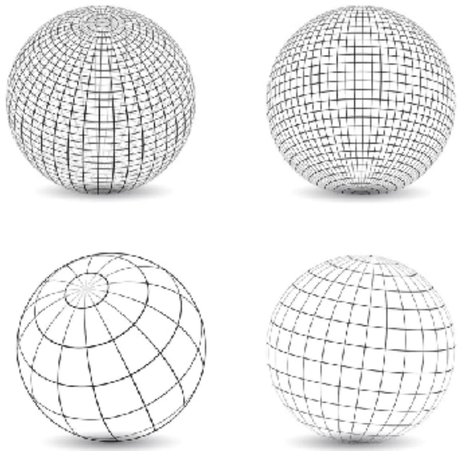
Geospatial Data Science Tutorial, IC2S2'23

Part 1: Data Handling
Data & Geometry, CRS, Libraries

Michael Szell
Jul 17, 2023

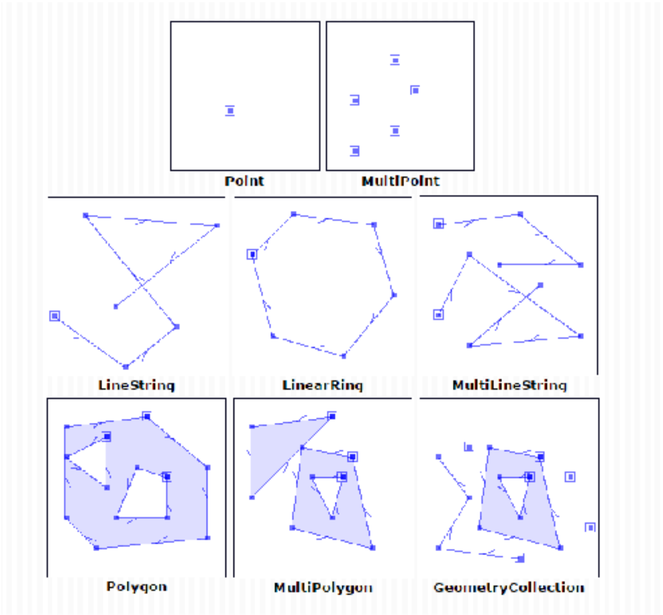


IT UNIVERSITY OF COPENHAGEN

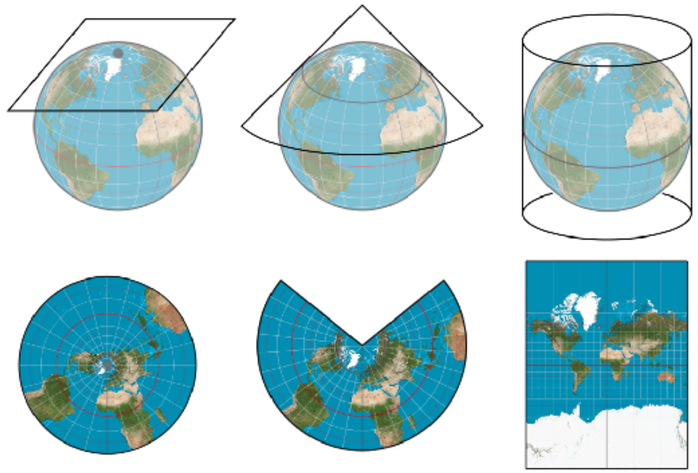
In this session you will learn about geospatial data basics

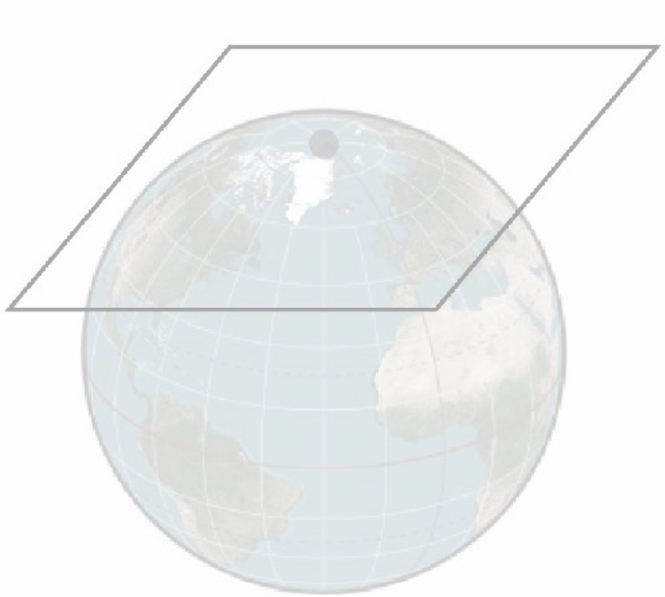
GeoPandas / How to work with vector data

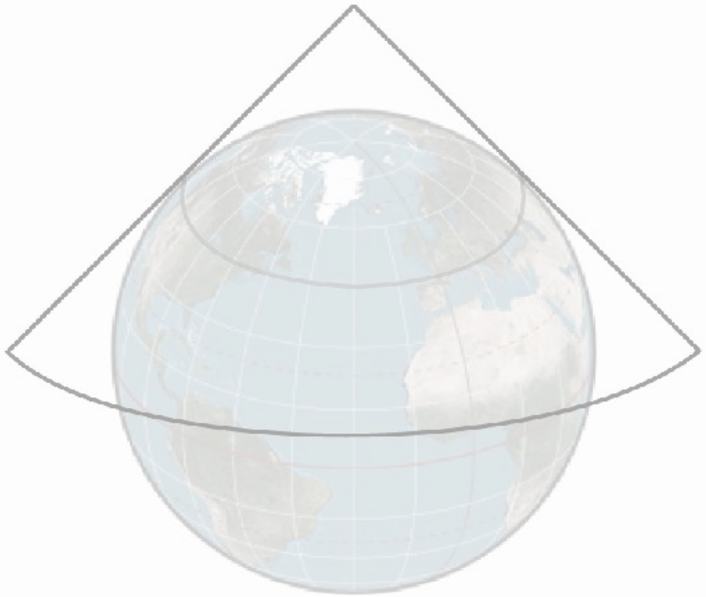
Shapely / Manipulating geometry

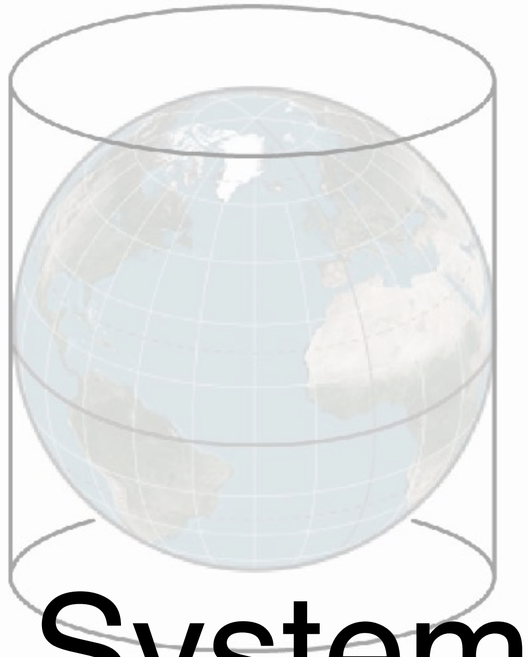


Coordinate reference systems (CRS)

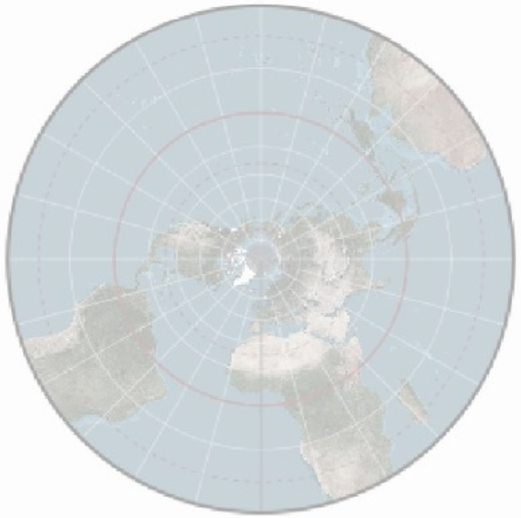


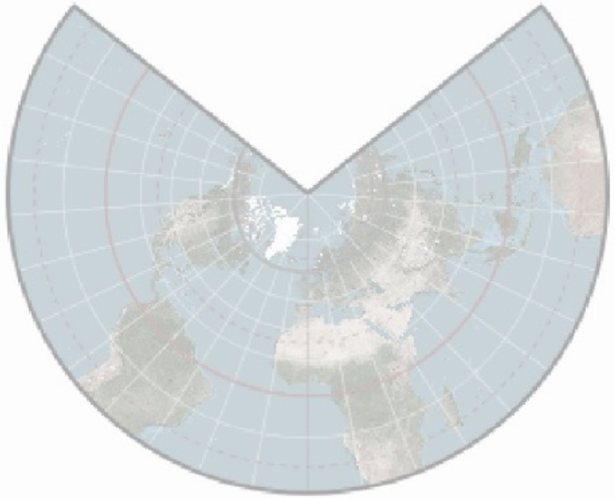






Coordinate Reference Systems





Coordinate Reference System (CRS)

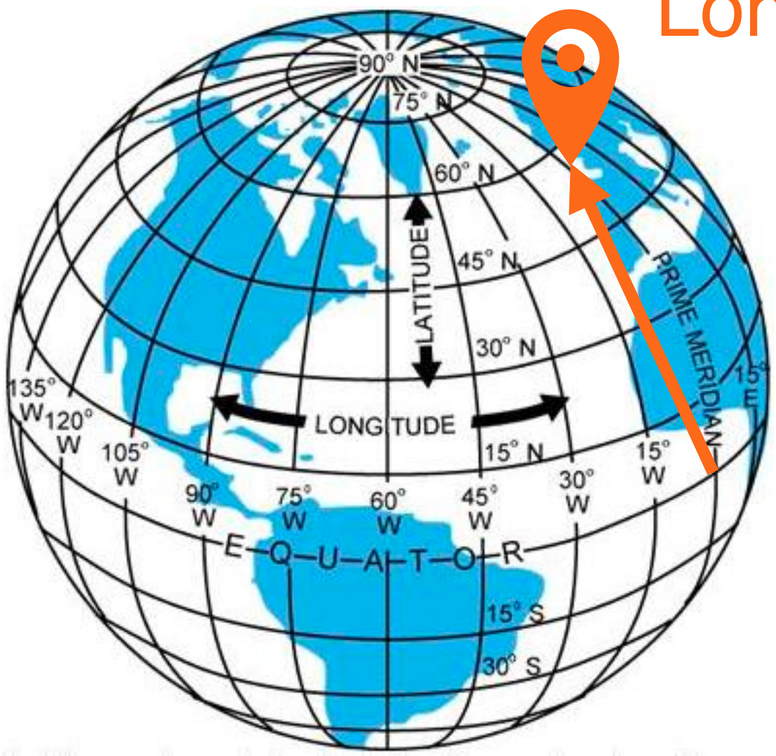
Geographic RS

(ϕ, λ) = (-0.1,51.5)

Longitude	Degrees N/S from equator
Latitude	Degrees W/E from meridian

Coordinate Reference System (CRS)

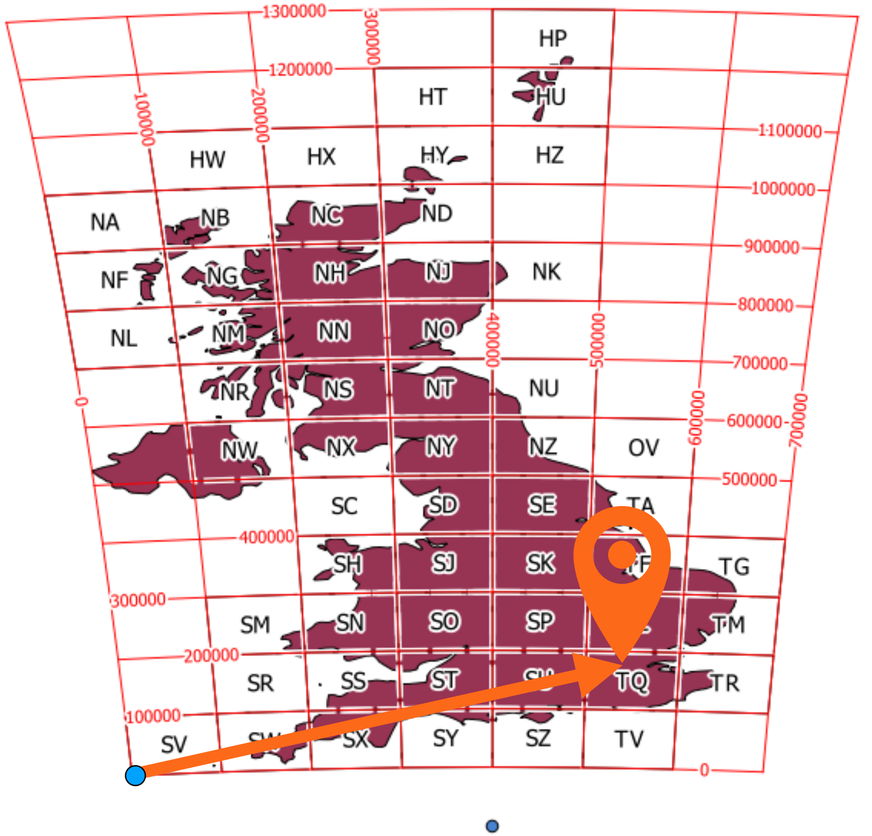
Geographic RS



(ϕ, λ) = (-0.1,51.5)

Longitude — Degrees N/S from equator
Latitude — Degrees W/E from meridian

Projected RS

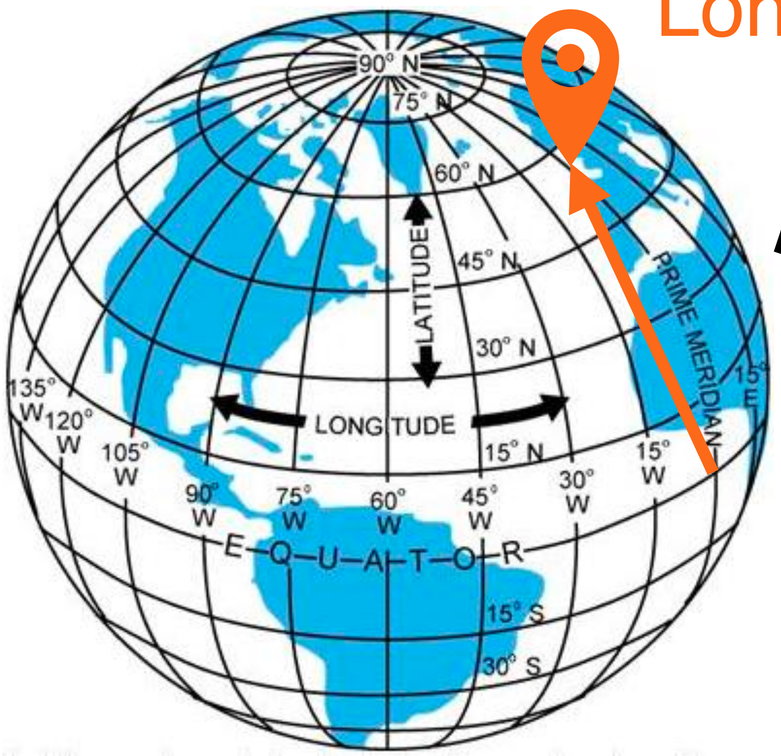


(x,y) = (530000,180000)

Easting — Meters east from origin (bottom left)
Northing — Meters north from origin (bottom left)

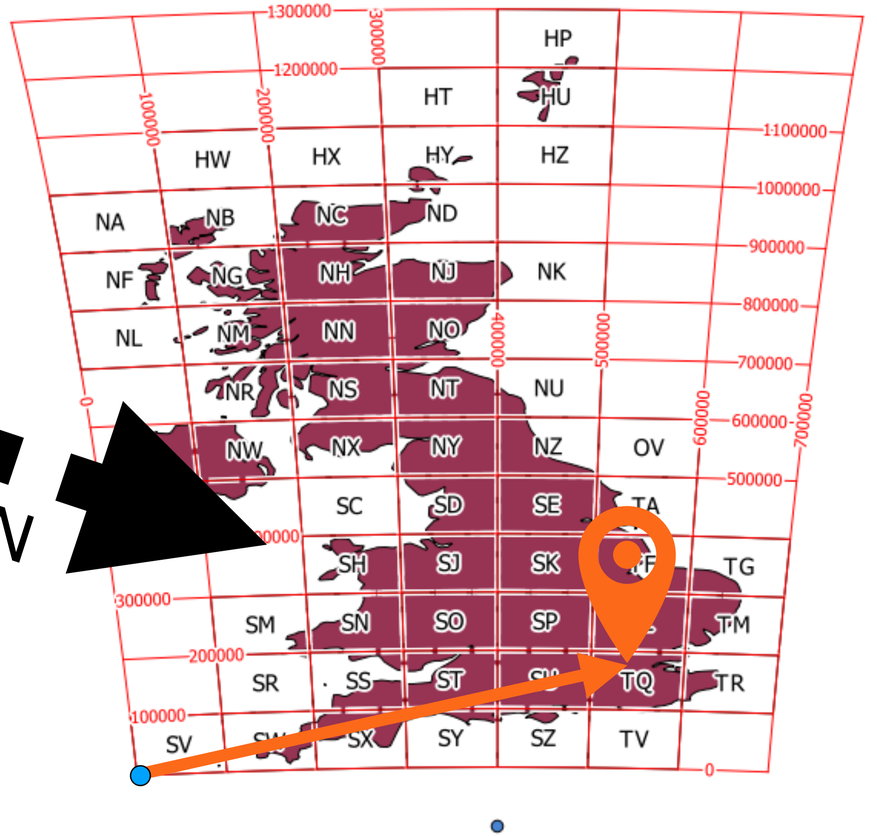
Coordinate Reference System (CRS)

Geographic RS



MAP PROJECTION

Projected RS



(ϕ, λ) = (-0.1,51.5)

Longitude — Degrees N/S from equator
Latitude — Degrees W/E from meridian

(x,y) = (530000,180000)

Easting — Meters east from origin (bottom left)
Northing — Meters north from origin (bottom left)

Common world-spanning reference systems

WGS 84 (World Geodetic System 1984) / EPSG:4326

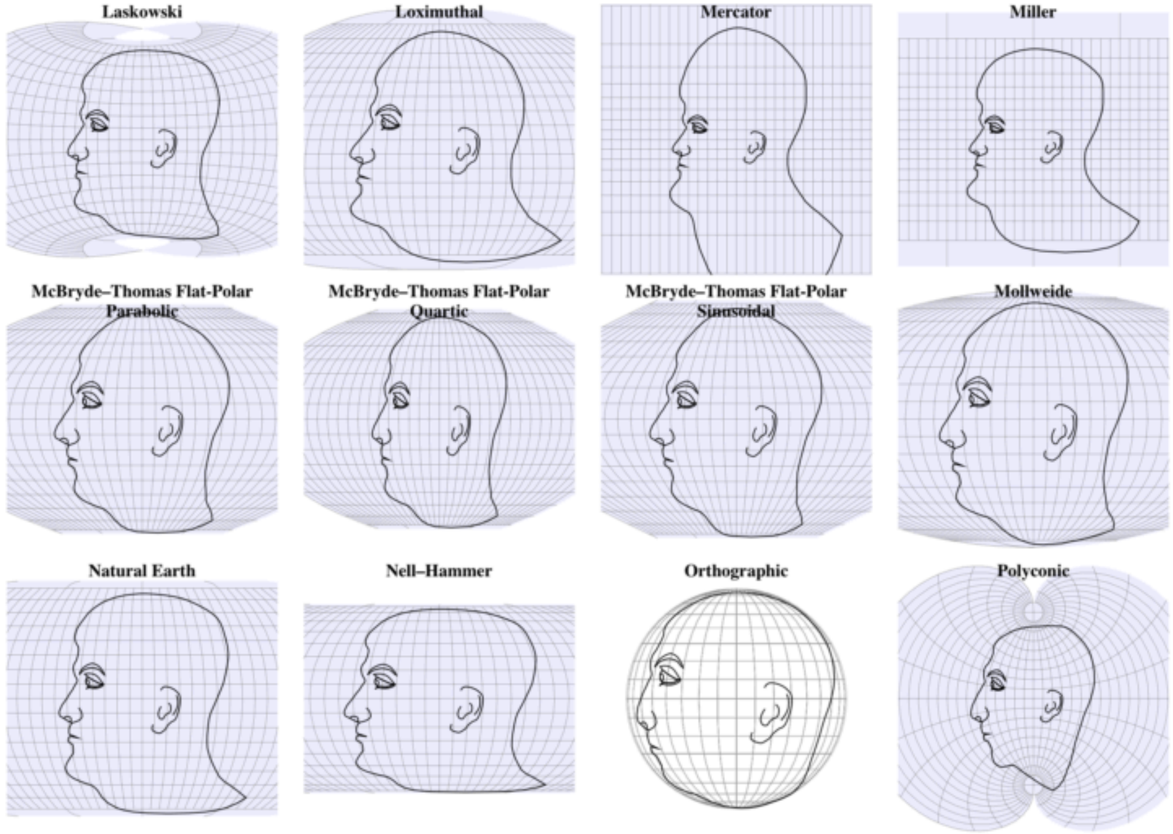
WGS 84 - Web/Pseudo-Mercator / EPSG:3857

Google Maps

epsg.io

From MapTiler Team

All map projections are wrong (but some are useful)



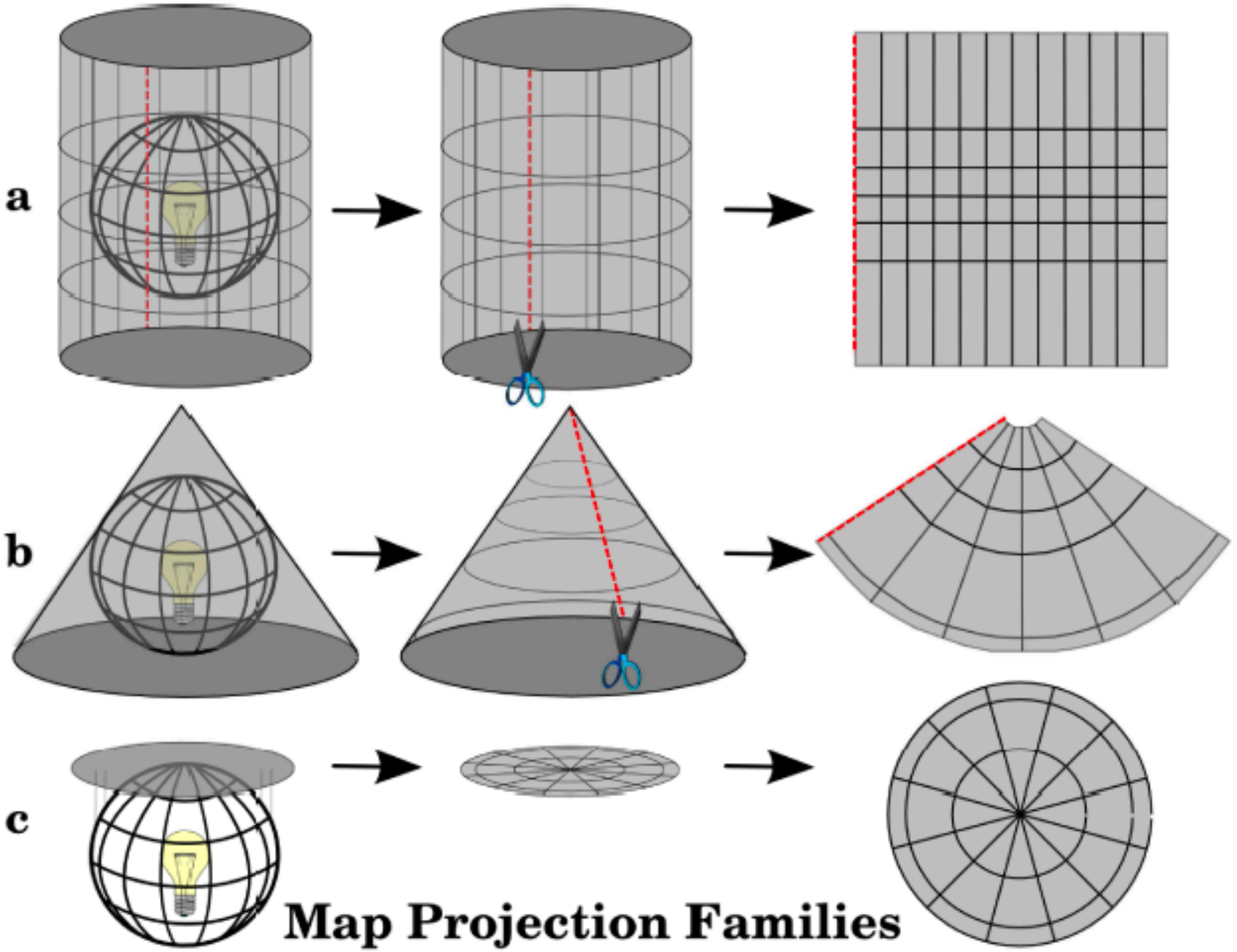
https://www.thetruesize.com/

3 families of map projections

Cylindrical

Conical

Planar



https://tomroth.com.au/projections/

Map projections are optimized for different things

Azimuthal: preserve directions

Conformal: preserve local angular relationships

Equal area or equivalent: preserve relative areas

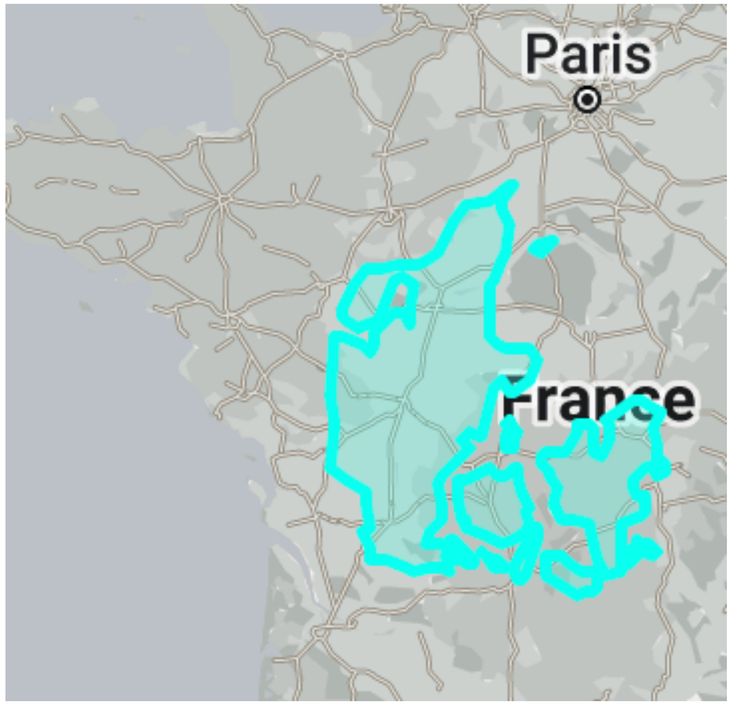
Equidistant: preserve distances

Compromise: does not preserve either areas or angles, but finds a compromise between the two, typically to present a more aesthetically pleasing map while reducing excessive distortion

https://gistbok.ucgis.org/bok-topics/map-projections

The biggest pitfall: Wrong or missing CRS!

Check your CRS!



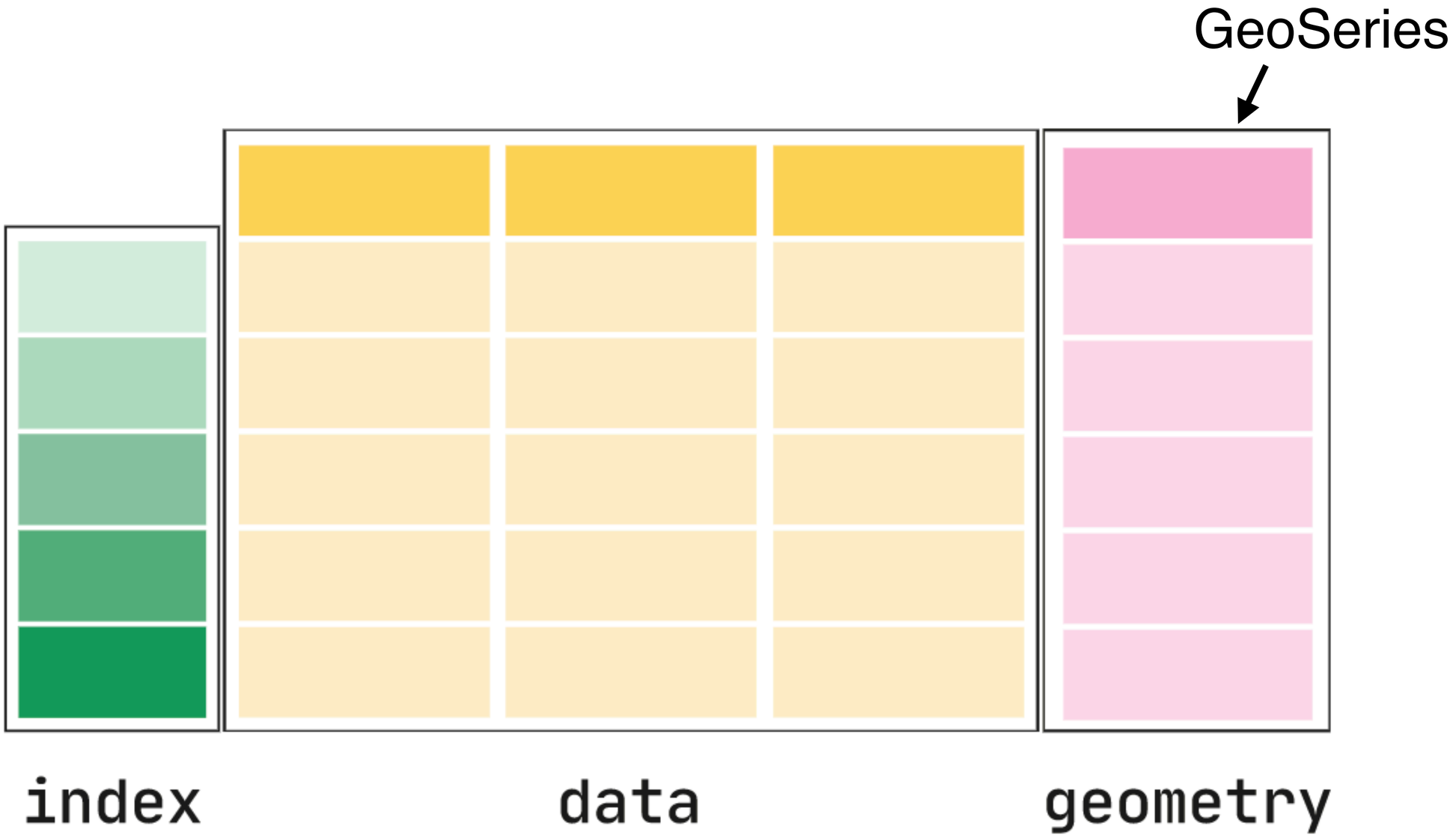
The 2nd biggest pitfall: (lat,lon) vs. (lon,lat)

lon, lat	lat, lon
formats • GeoJSON ref • KML ref • Shapefile ref • WKT ref • WKB ref • geobuf ref	formats • GeoRSS ref • Encoded Polylines (Google) ref • iCalendar ref
javascript apis • OpenLayers ref • d3 ref • ArcGIS API for JavaScript ref • Mapbox GL JS ref	javascript apis • Leaflet ref • Google Maps API ref

GeoPandas

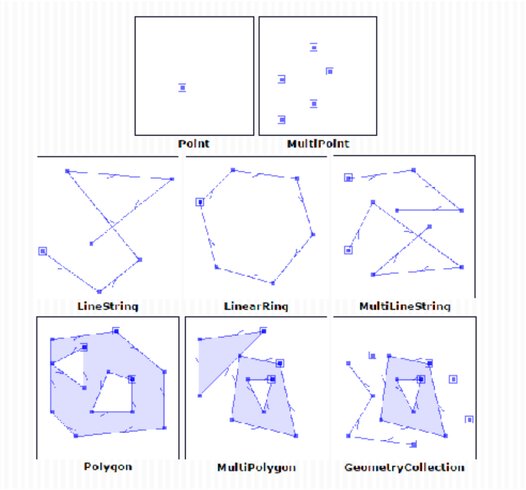
The spatial extension of pandas

The GeoDataFrame (gdf)



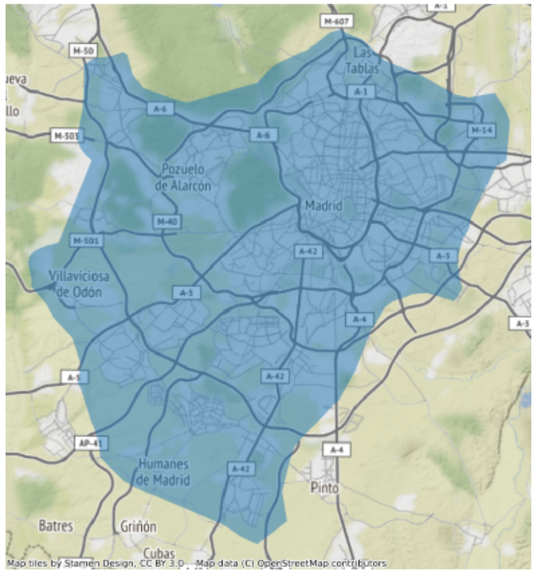
GeoPandas features

Uses Shapely for geometries

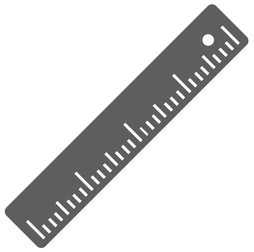


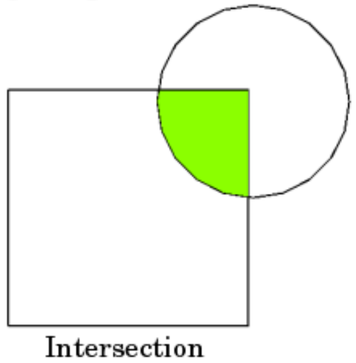
CRS support

Static maps



Spatial operations

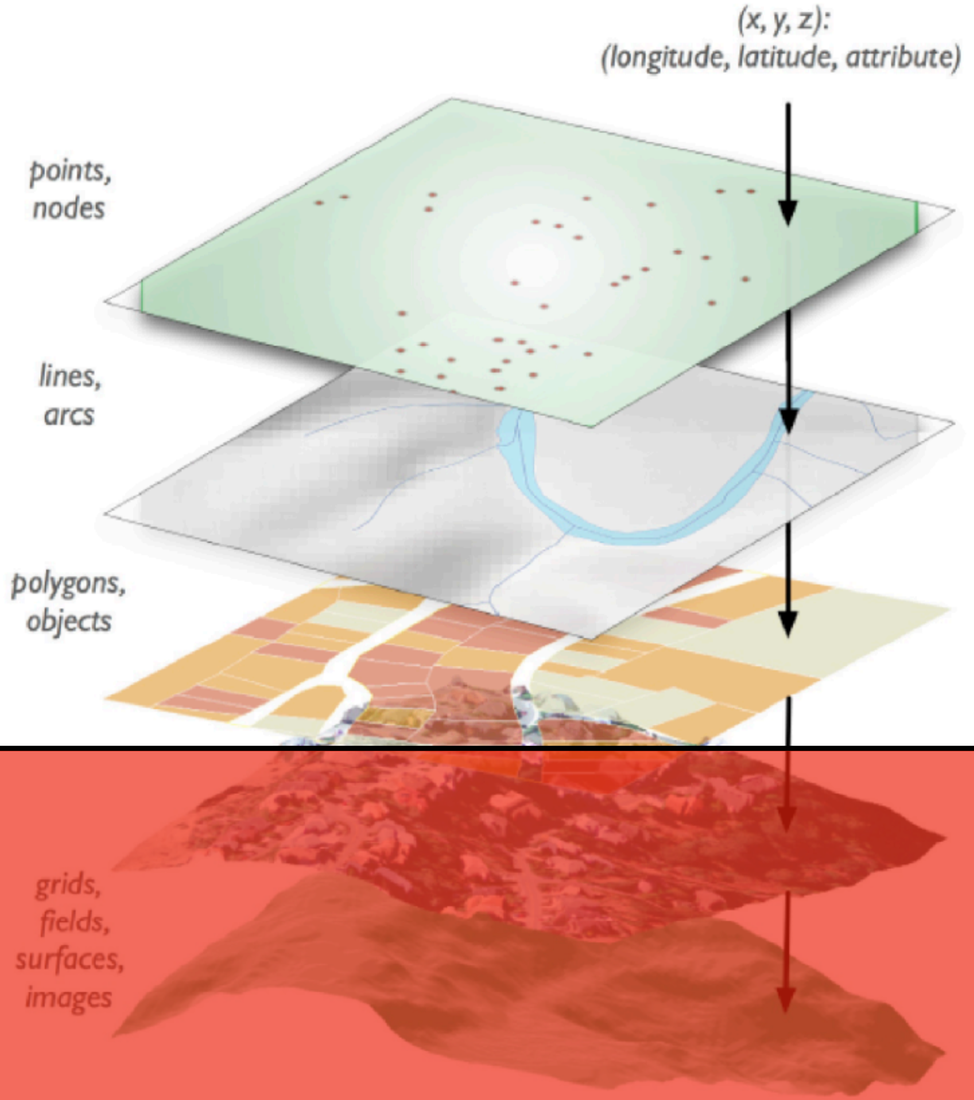




GeoPandas handles vector data

Vector: Geometric objects

.gpkg, .shp, .svg, geojson



Raster: Grid of pixels

.tif, .jpg, .png, .bmp

Use rasterio for raster data

Image: Maggi Kelly, UC Berkeley

File formats matter

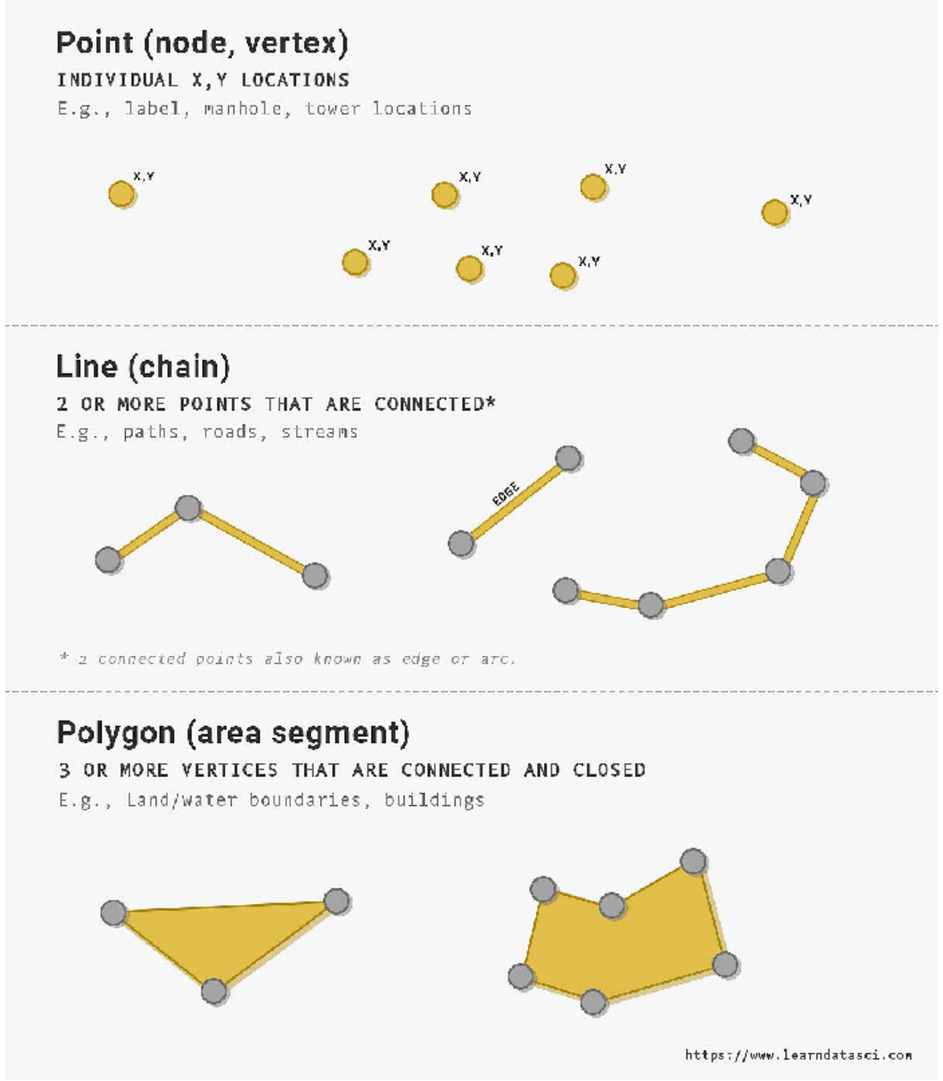
Geopackage - ‘universal’

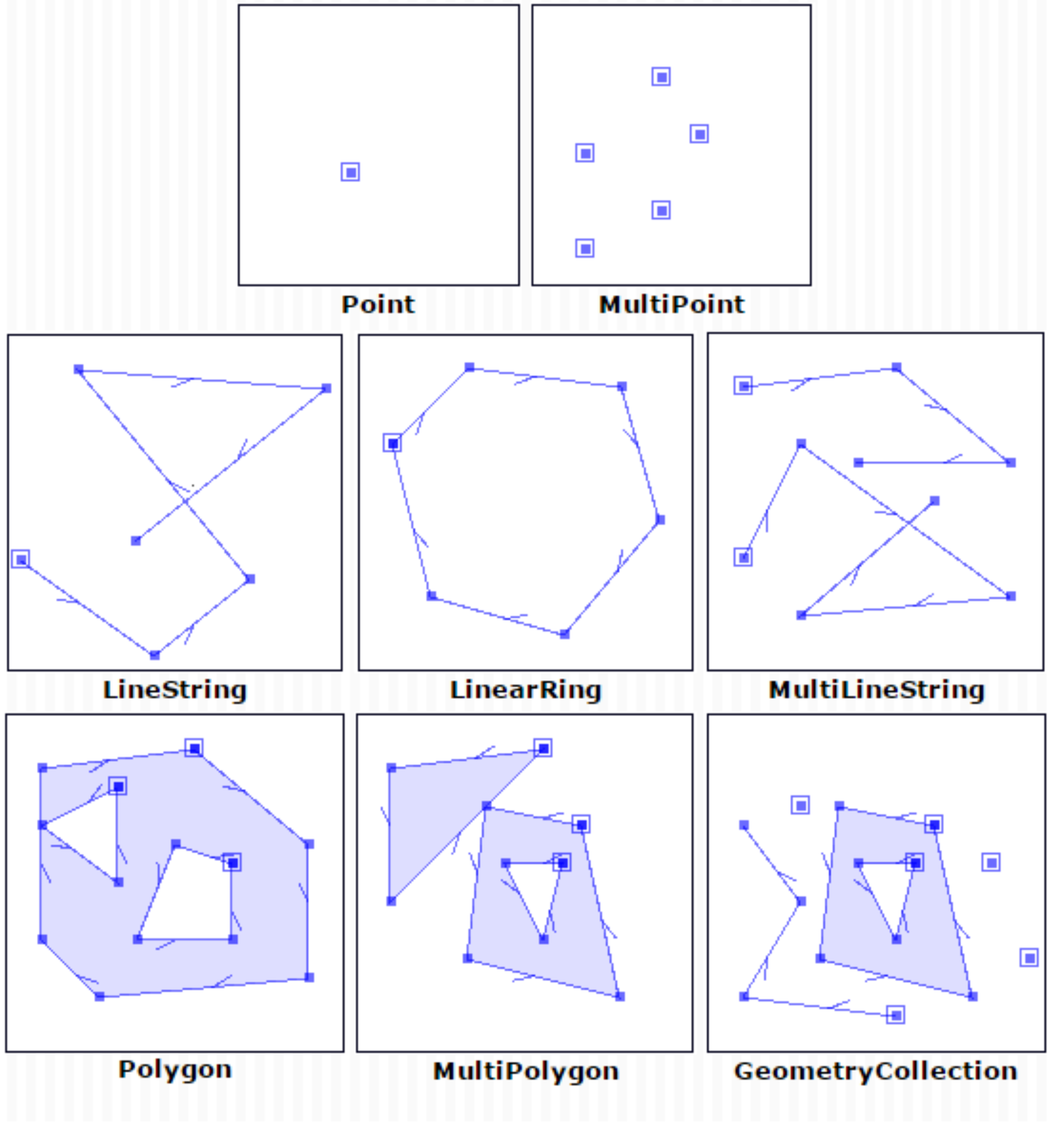
GeoJSON - web-optimized

CSV

Shapefiles - old classic

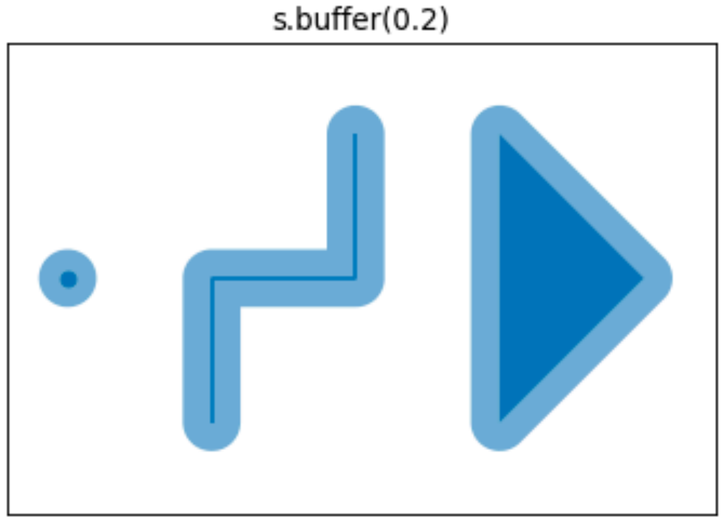
The basic geometric objects are handled by shapely



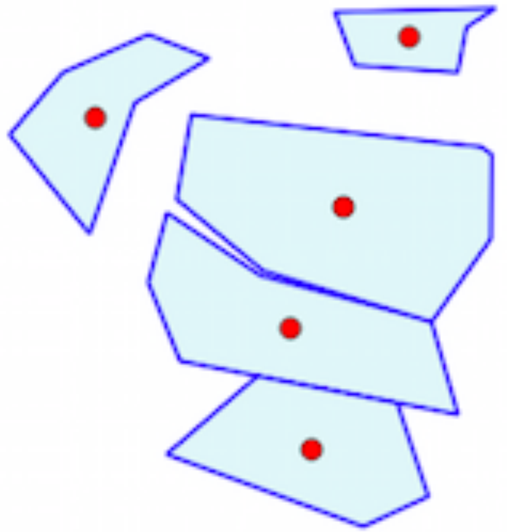


Spatial operations in GeoPandas/Shapely

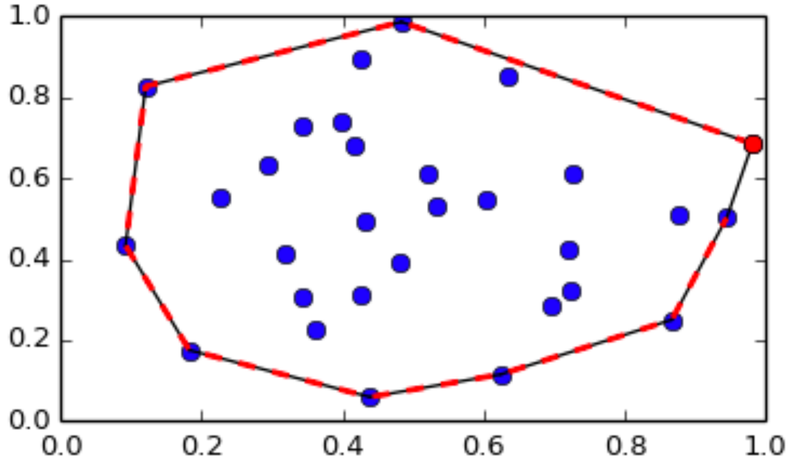
Geometric Manipulations



`gdf.buffer(10)`



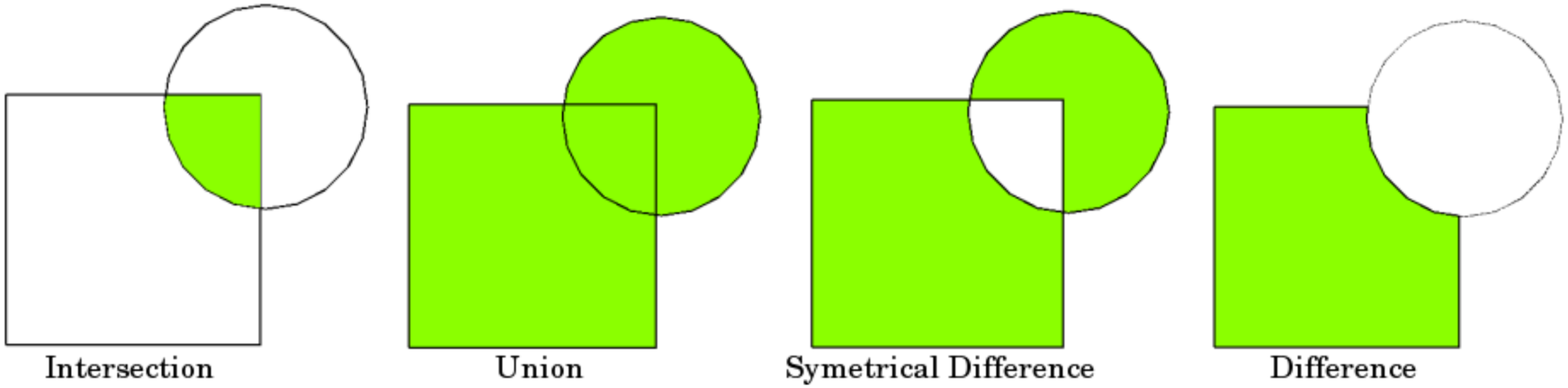
`gdf.centroid`



`gdf.convex_hull`

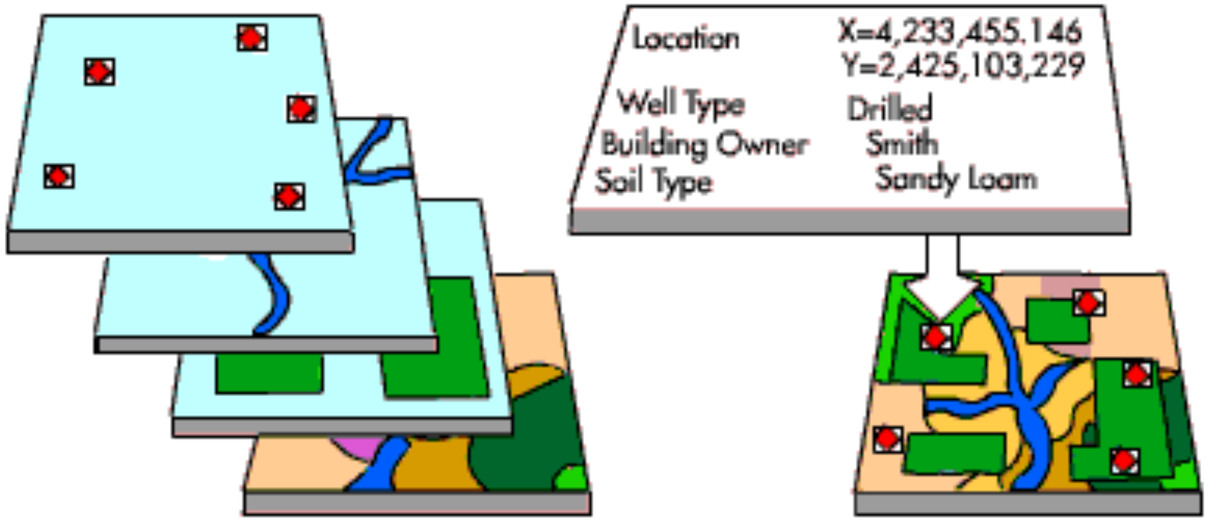
Spatial operations in GeoPandas/Shapely

Set based operations




```
intersection = gdf1.overlay(gdf2, how='intersection')
```

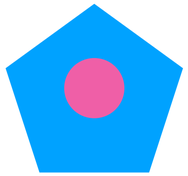
Spatial join



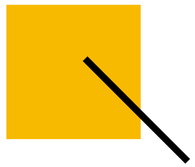

```
join = point_gdf.sjoin(poly_gdf)
```

Spatial queries

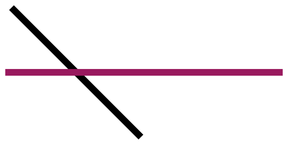
```
point.within(poly)
```



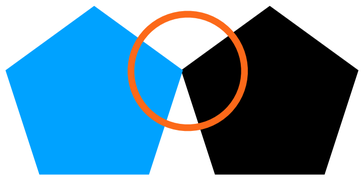
```
line.intersects(poly)
```



```
line1.crosses(line2)
```



```
poly1.touches(poly2)
```

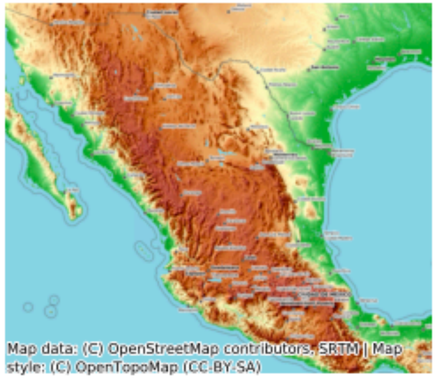


etc….

Contextily adds a basemap to your data

OpenStreetMap.Mapnik

OpenTopoMap



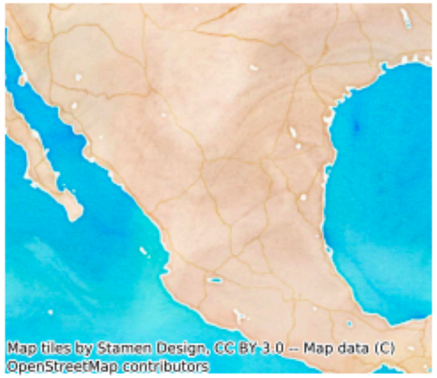
Stamen.Toner

Stamen.TonerLite

Stamen.Terrain

Stamen.TerrainBackground

Stamen.Watercolor



NASAGIBS.ViirsEarthAtNight2012

CartoDB.Positron

CartoDB.Voyager

https://contextily.readthedocs.io/en/latest/

Jupyter

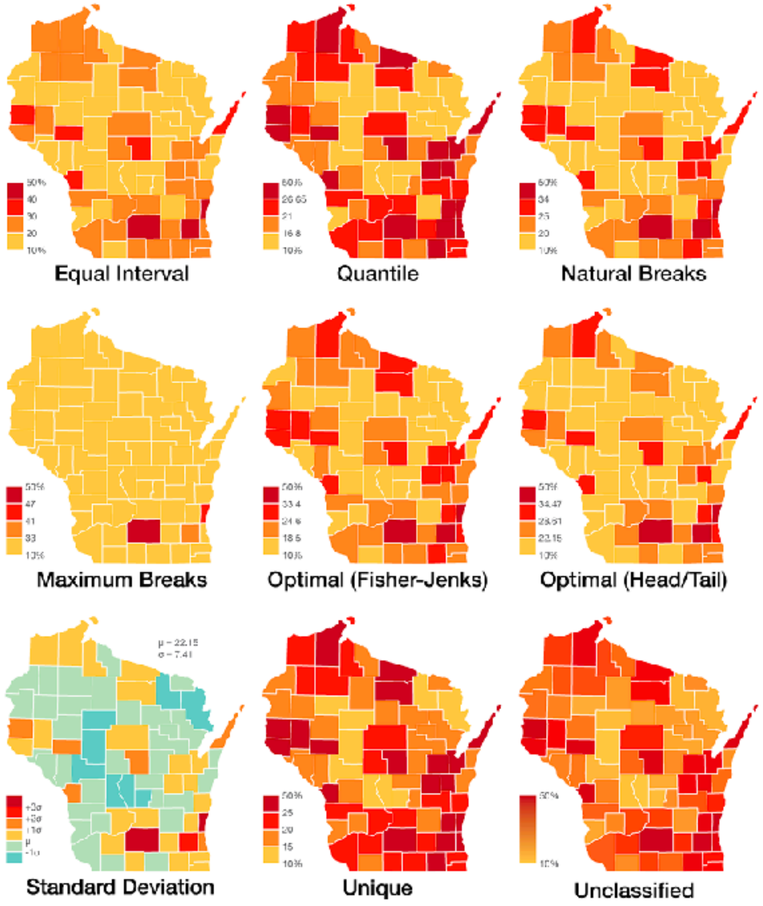
part1/part1datahandling.ipynb

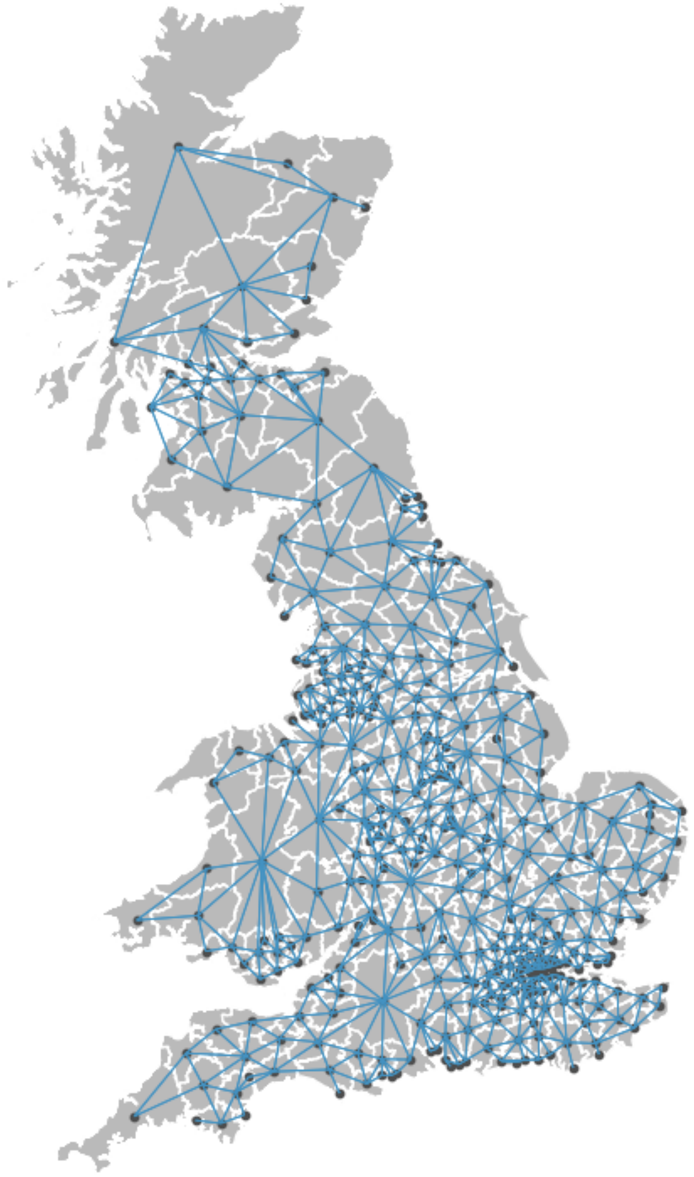
github.com/NERDSITU/gdstutorial

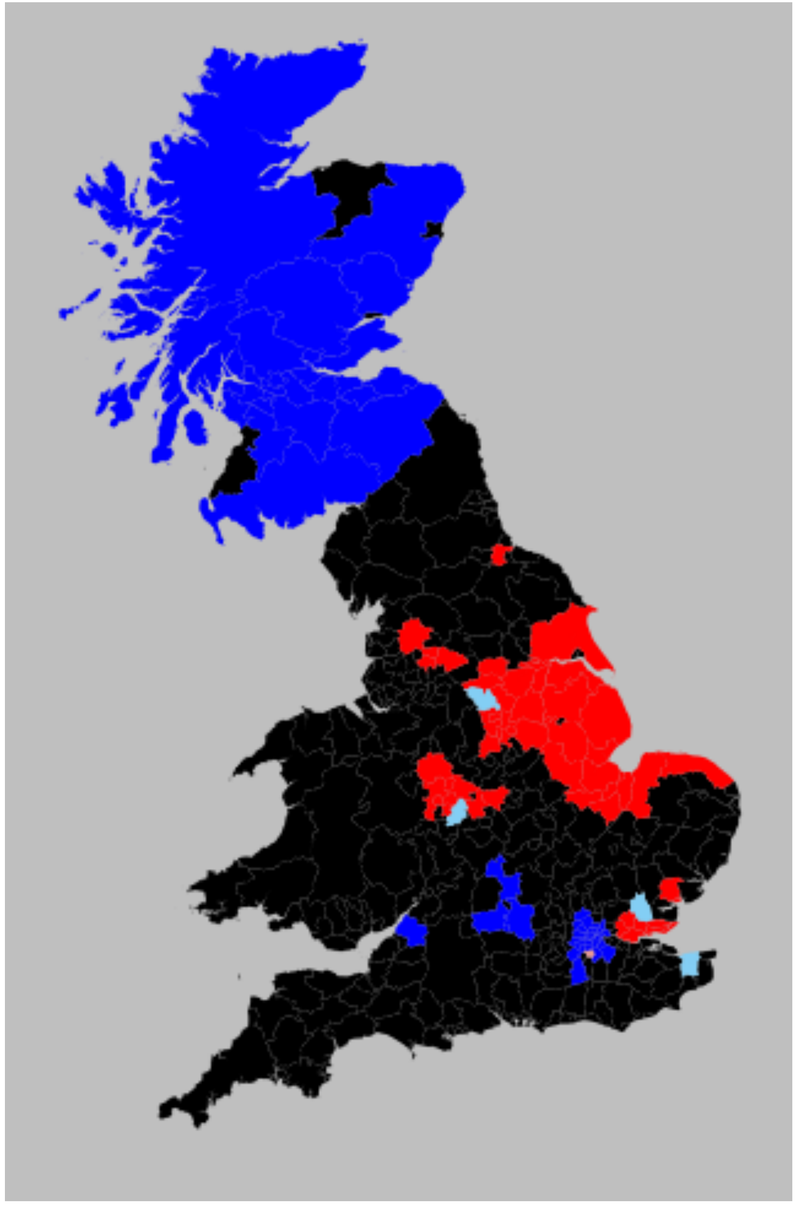
We now know how to…

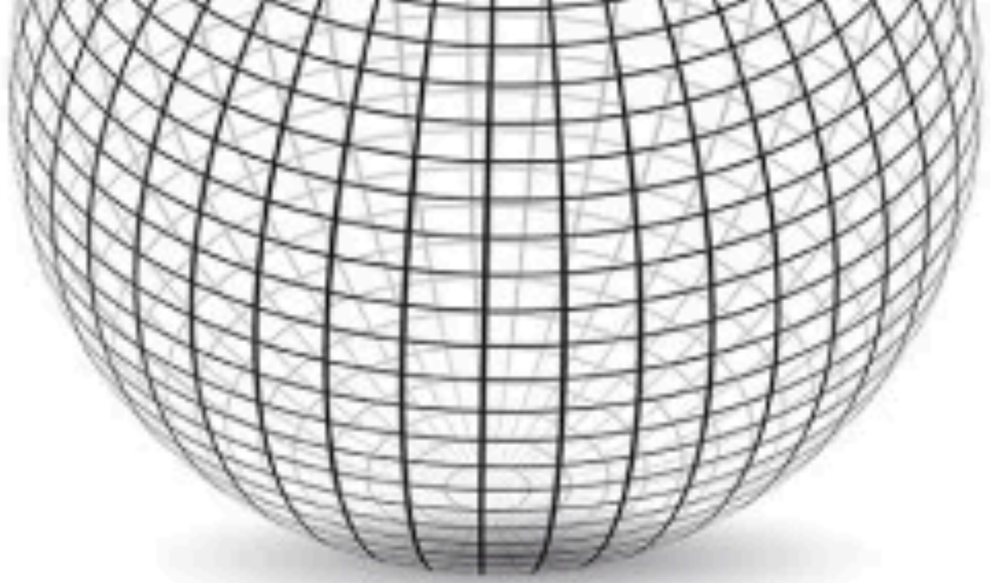
- Read vector data with *GeoPandas*
- Make simple maps with *matplotlib* and *contextily*
- Do spatial operations like buffering
- Understand and modify the data's CRS

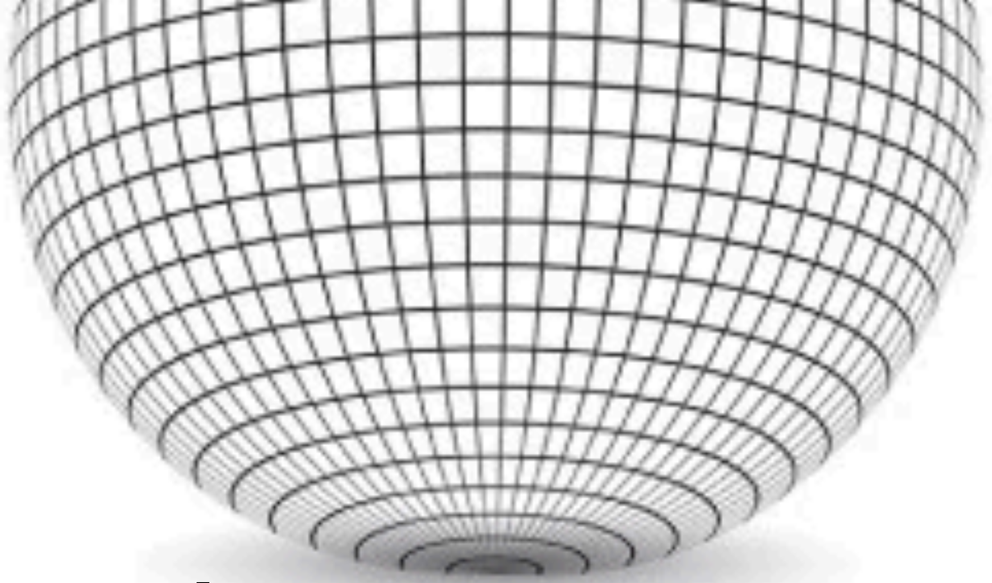
Next session: Choropleth maps, Spatial autocorrelation











10 min break

See you at 9:50