GEOSPATIAL STATISTICS

An Introduction
Georg Heiler UII 2023

Georg Heiler: georgheiler.com

Co-founder of startup for time series prediction

Senior Sofware Engineer with a specialization in data @MagentaTelekom

big geospatial analytics

Lecturer (R-Summer schoo, UII, DHBW)

Organizing data science meetups in Vienna Data Science Group (VDSG) & board member

Post Doc Researcher @Complexity Science Hub

Agenda

- Propties of spatial data
- Examples of spatial data
- Geospatial usecases
- Spatial analytics
- Scaling geospatial data handling
- Geo statistics

What is Geo processing?

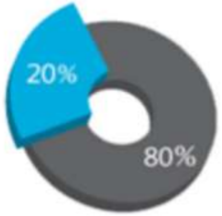
- Operations to manipulate spatial data
- Operations include **geographic feature overlay**, feature selection and analysis, **topology processing**, raster processing, and data conversion

- Geospatial statistics: statistics with spatiotemporal data

Source: Wikipedia

Geography matters: 60 % of all data is spatial

> 80% of the informational needs of local government are related to geographic location.



- 80 % of all data is spatial, is a commonly repeated myth (a phrase used for selling geospatial data to governments: Williams, 1987).
- However, after investigating, scientists have estimated that **approximately 60 %** of all the data is geospatially referenced
    - **Still, it's a lot**
    - With the same logic, one might reason that at least ~60 % of all Sustainable Development Goals and indicators have a geographical dimension in them, hence, spatial data science inherently has a lot to give for SDGs

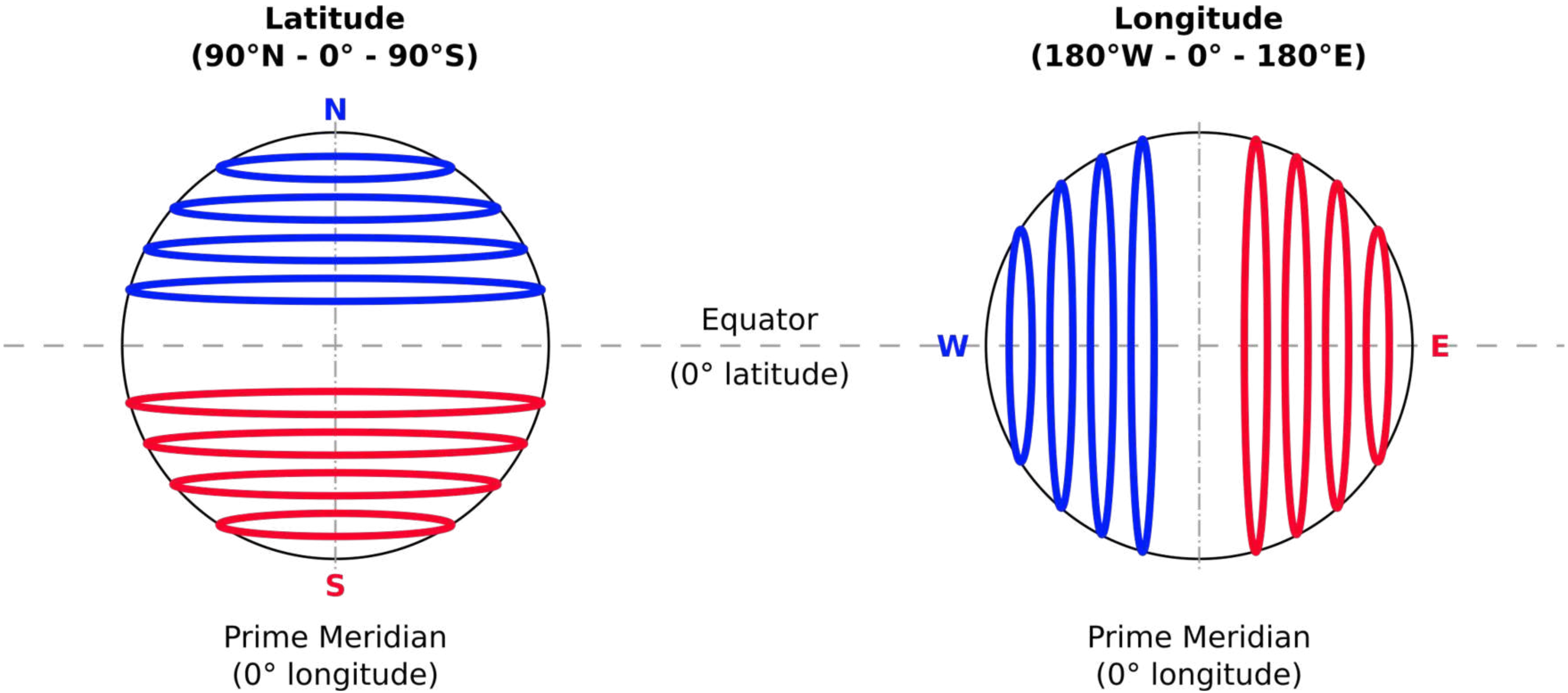
Properties
of spatial data

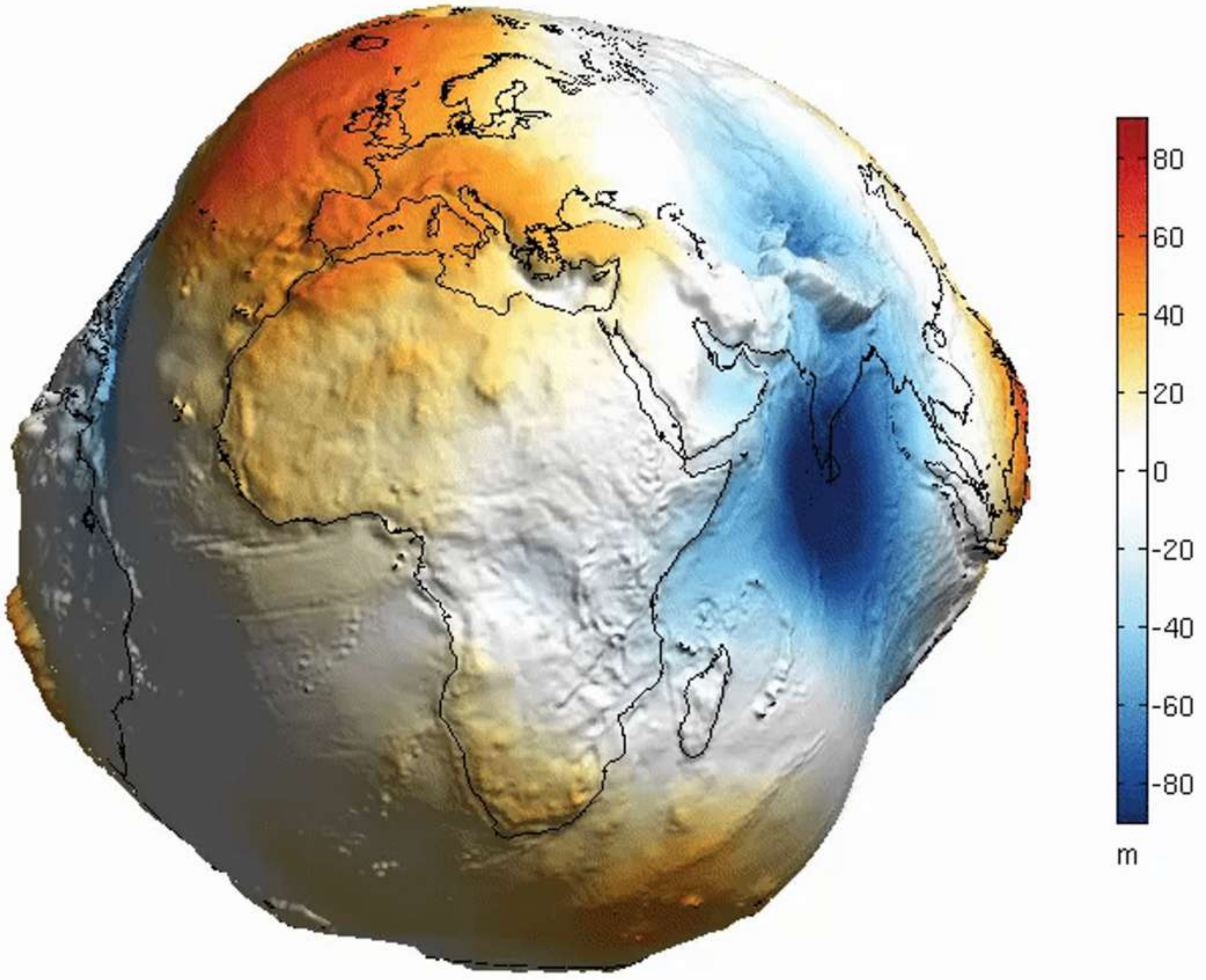
Spatial data

- Data for spatial reference
- Geometries (points, polygons, lines), 1D or 2D
- Often latitude, longitude as x, y spatial reference
- Point clouds (3D, 4D) of LIDAR scans

geometry	value	thing
POLYGON ((-97.019...	31	Cumin
POLYGON ((-123.43...	53	Wahkiaku
POLYGON ((-104.56...	35	De Bac
POLYGON ((-96.910...	31	Lancaste

What is latitude and longitude?





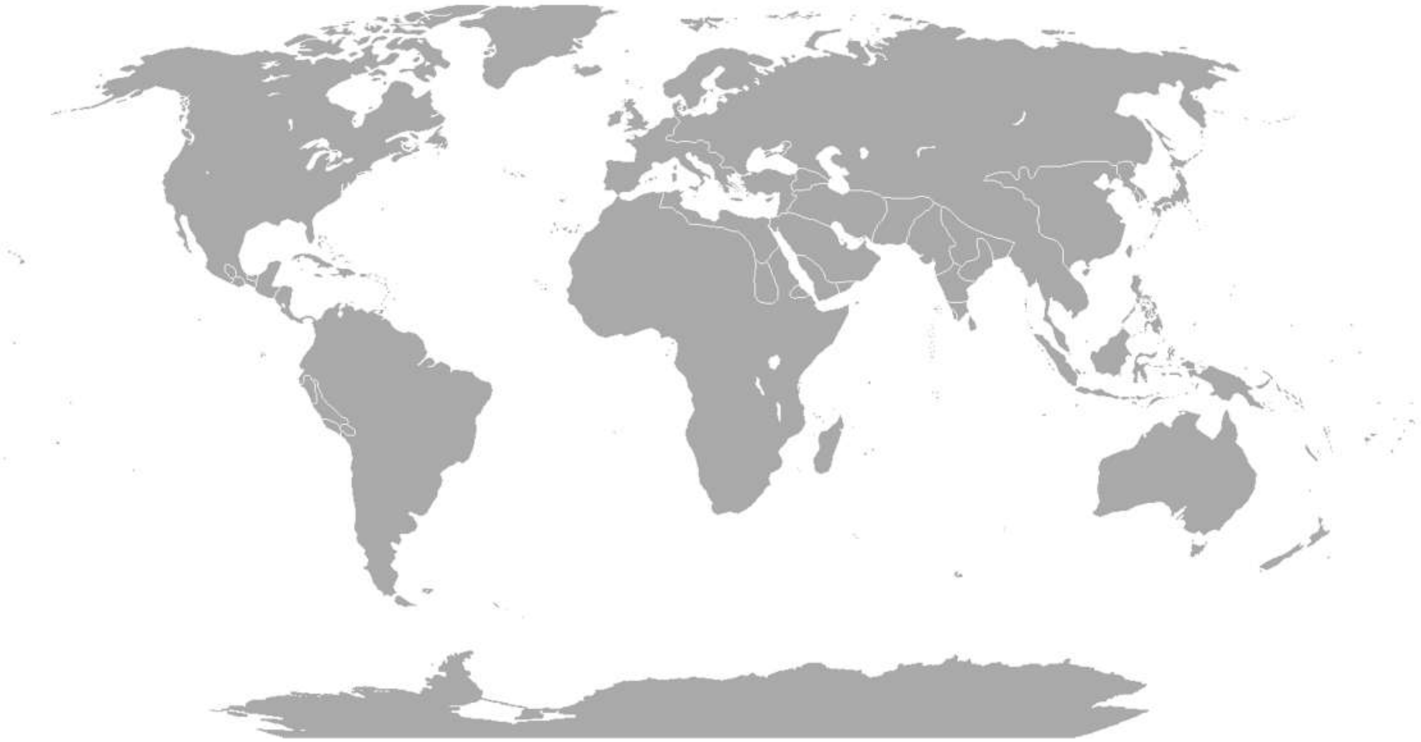
Source: imgur.com

Geoid height (EGM2008, nmax=500)

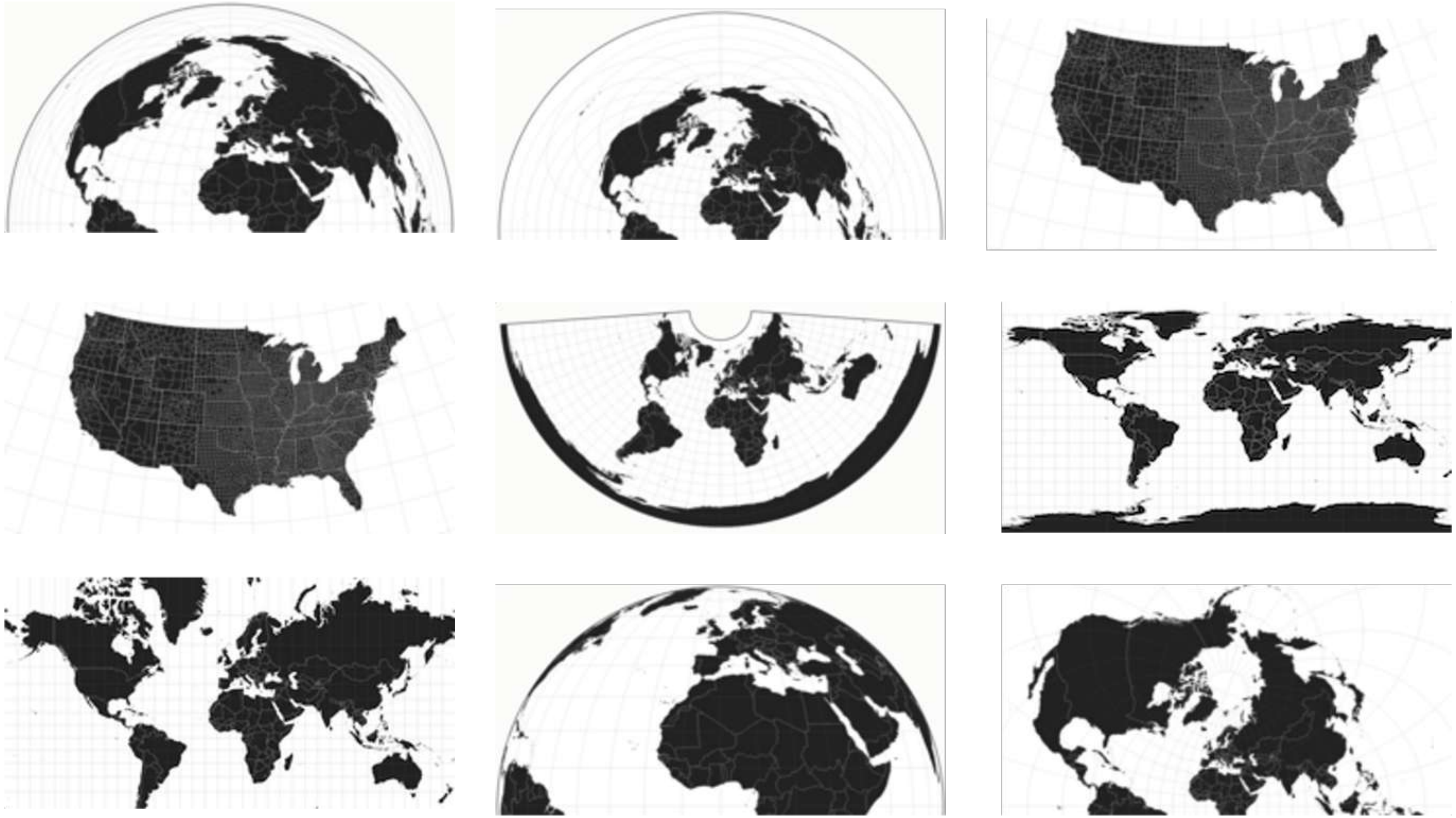
3D Coordinates

- Earth is not a perfect sphere!
- Can be approximated by a biaxial ellipsoid
- 3D coordinates need a reference ellipsoid
- Widely used is the **World Geodetic System (WGS84)** used by GPS
- Minimal positioning error on the surface

3D or 2D



Going to 2 Dimensions



2D Projections

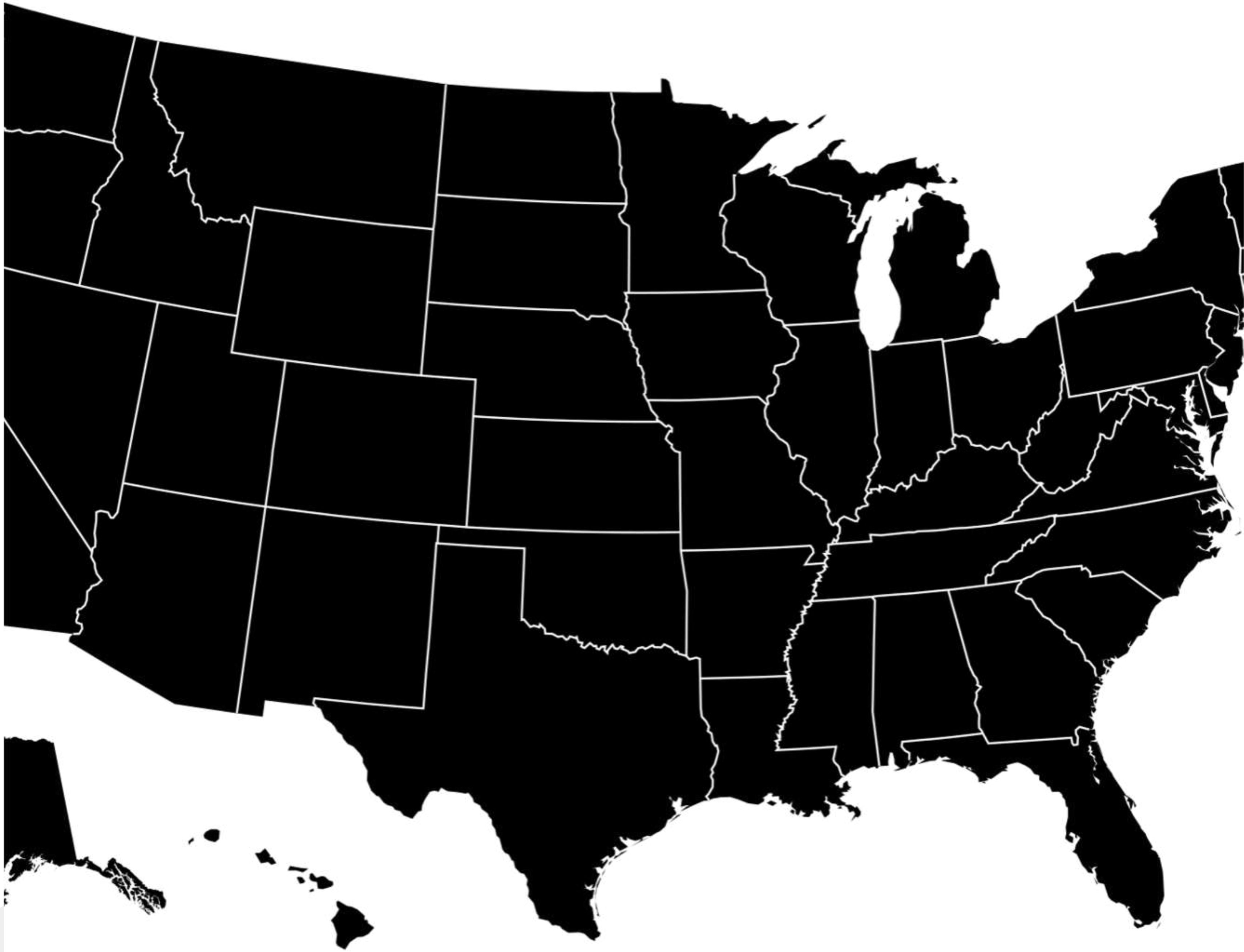
- The earth cannot be displayed on a 2D map without distortion
- Mapping to the surface of other 3D Volumes
  - Cylindrical
  - Conical
  - Azimuthal

2D Projections

- The earth cannot be displayed on a 2D map without distortion
- Every mapping has its tradeoff
  - Length Preserving (Equidistant)
  - Area Preserving (Equal Area)
  - Angle Preserving (Conformal)

2D Projections

- Commonly used in Austria: MGI Austria Lambert (equal area)
- Commonly used in the US: Albers USA projection (equal area)

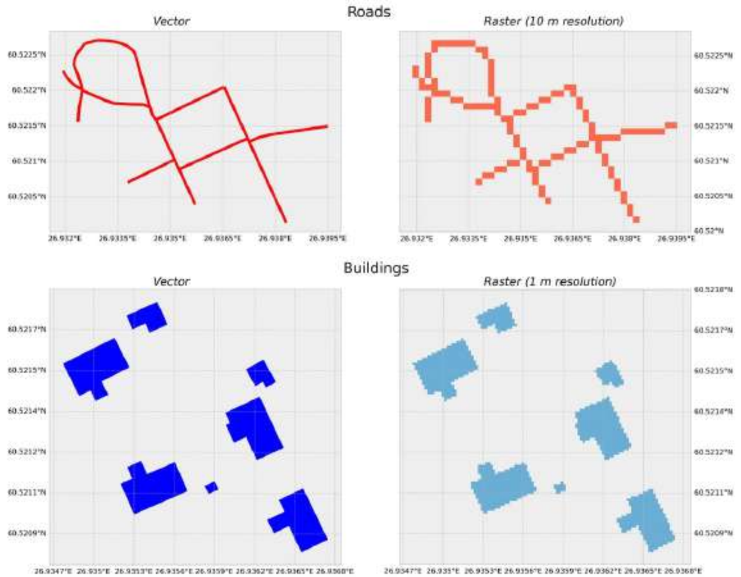


Projections

- Remember different coordinate systems
- Make sure all data sets use the same CRS!!!
    - If not apply reprojection

Data formats

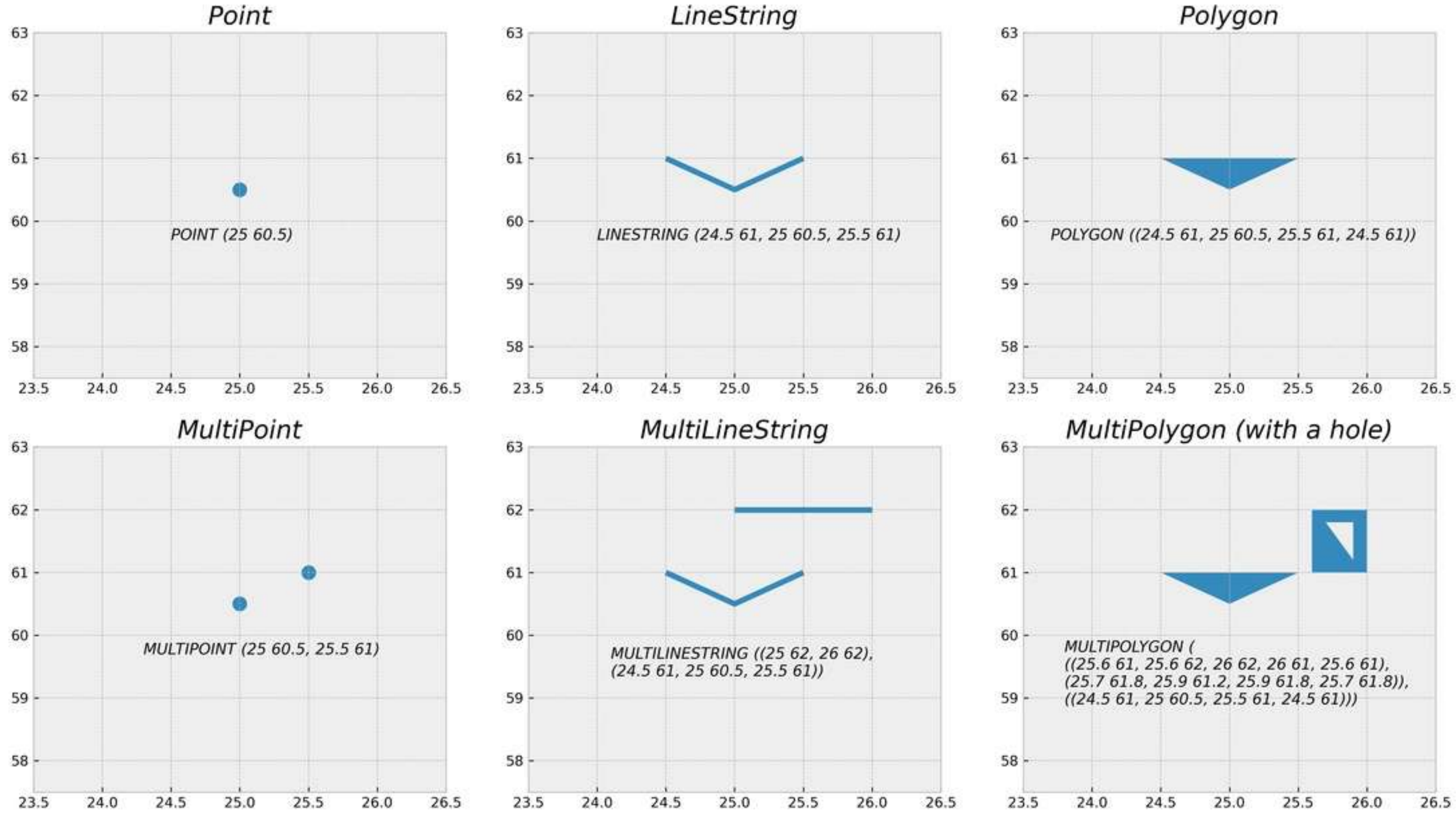
- Make the data format readable by your tool of choice
- Vector
  - Well Known Text
  - GeoJson
  - Shapefile
  - Geodatabase (file)
  - Points
  - Geoparquet
- Raster
  - GeoTiff
- 3D
  - LIDAR data



There is a huge variety of formats! https://pro.arcgis.com/de/pro-app/help/data/imagery/supported-raster-dataset-file-formats.htm you will probably need to convert data from different sources to a single shared format

https://pythongis.org/part2/chapter-05/nb/01-introduction-to-geographic-data-in-python.html

Geometry types



https://pythongis.org/part2/chapter-05/nb/01-introduction-to-geographic-data-in-python.html

GDAL - Transform Shapefiles to CSV

```
ogr2ogr -f CSV output.csv \
        input.shp \
        -lco GEOMETRY=AS_WKT \
        -lco SEPARATOR=SEMICOLON \
        -oo ENCODING=UTF-8
```

GDAL - Use spatial queries

ogr2ogr -sql "SELECT A.* FROM shape1 A, shape2 B WHERE **ST_Intersects**(A.geo, B.geo)" \

-dialect SQLITE \

data input_dir \

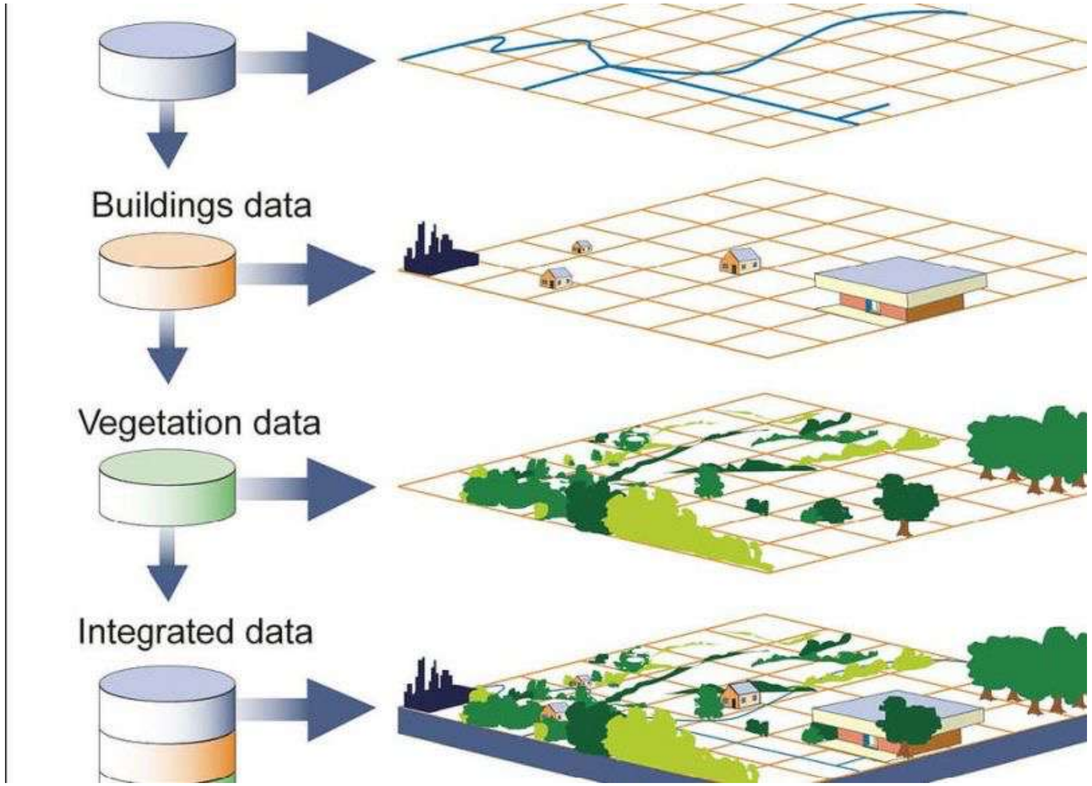
-nln output.shp

More complex preprocessing

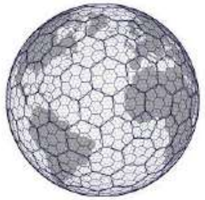
- Custom scripts with geospatial libraries of choice in programming language of choice
  - Python (scripts, exploratory analytics)
  - JVM based (production pipeline)

Examples
for spatial data

Geospatial Information System

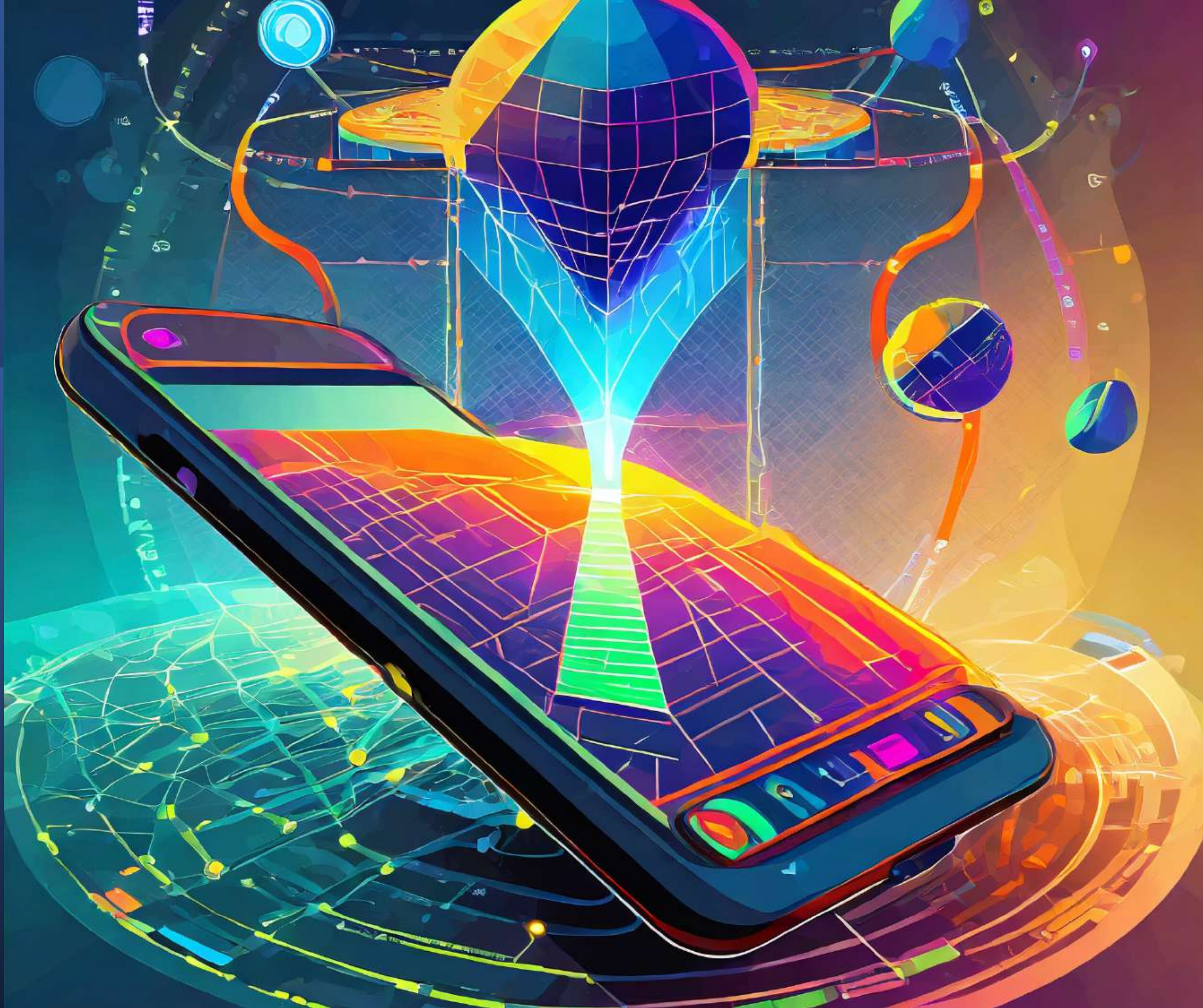


- Government published open data
- Geo-marketing
- Raster
  - Topology (federal states, municipality, postal codes)
  - Mathematical. (S3, H3, geoHash)

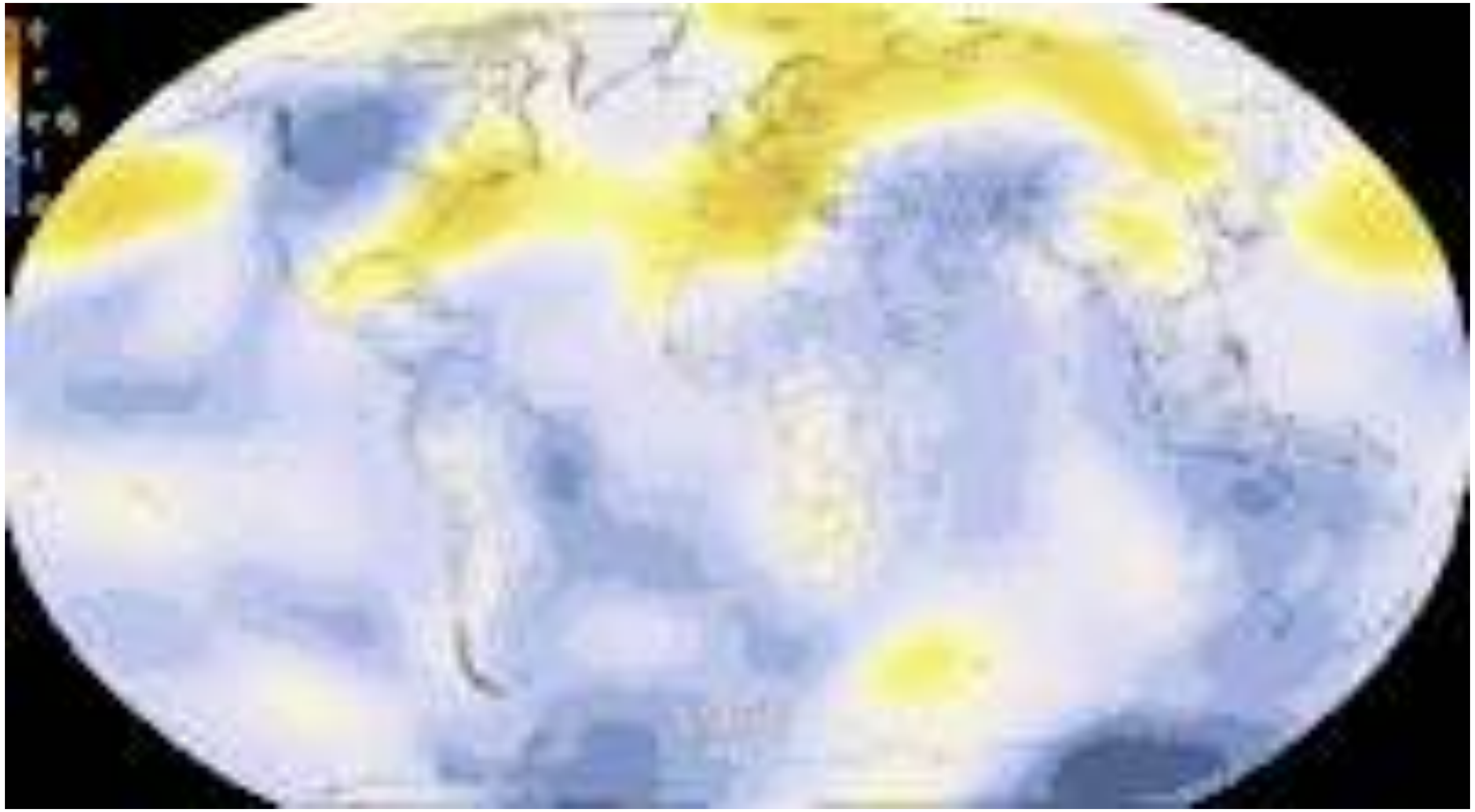


https://education.nationalgeographic.org/resource/geographic-information-system-gis/

Global Positioning System



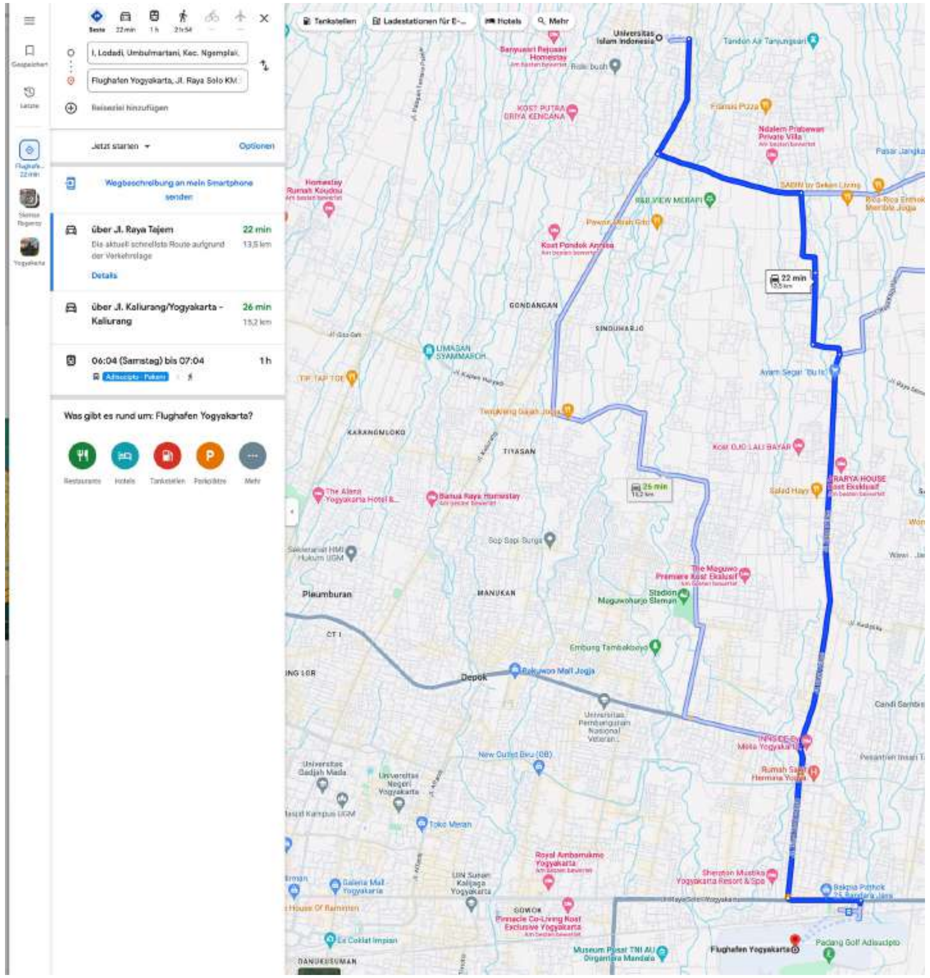
Remote sensing satellite data Spatiotemporal data: Track over Time, Potentially Forecast the Future



REAL TIME STREAM OF IOT DATA

LIDAR Scan

- Self driving cars
- Drones
- Maintenance of industry plants

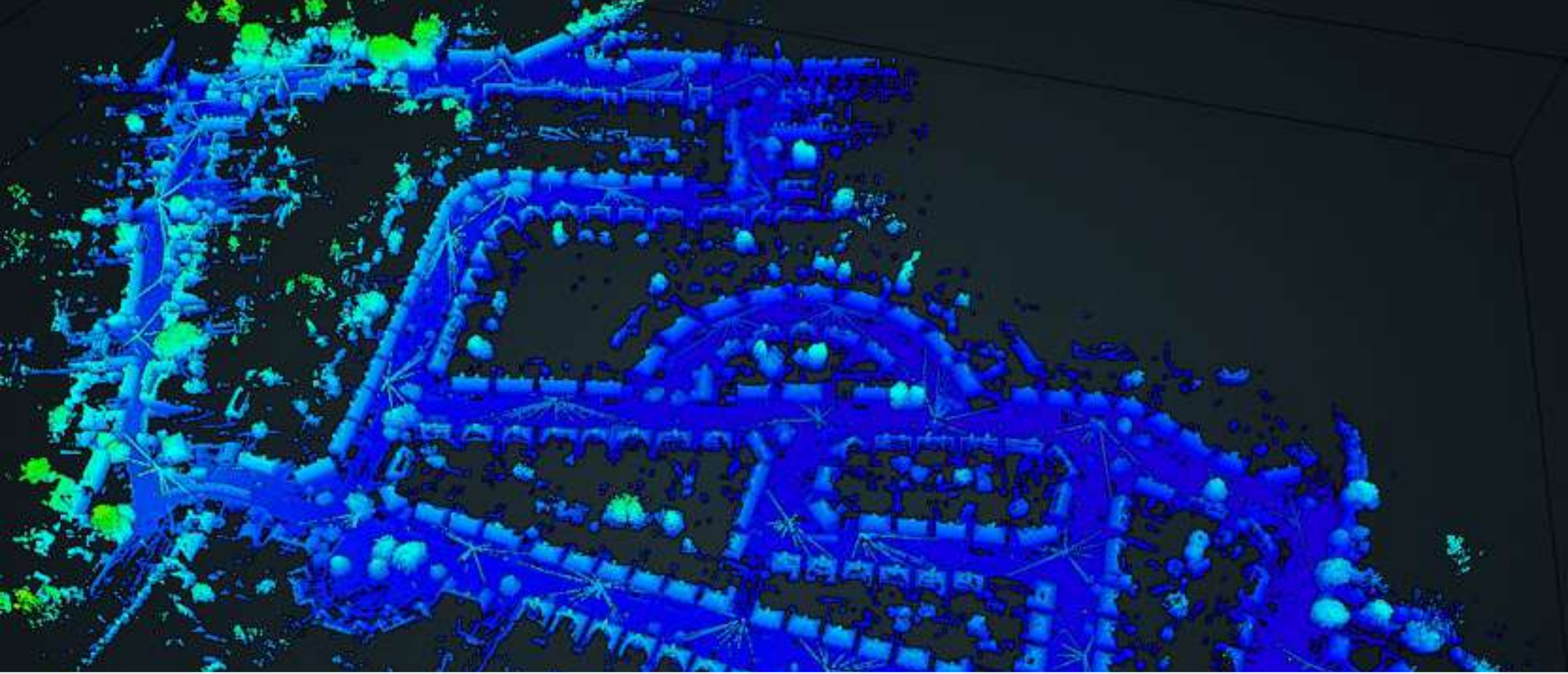


Topological data (routing)

Spatial usecases

Disaster recovery

- Awareness
- Mitigation
- Recovery



Urban planning

https://geoslam.com/blog/2021/10/05/why-is-geospatial-information-so-important/

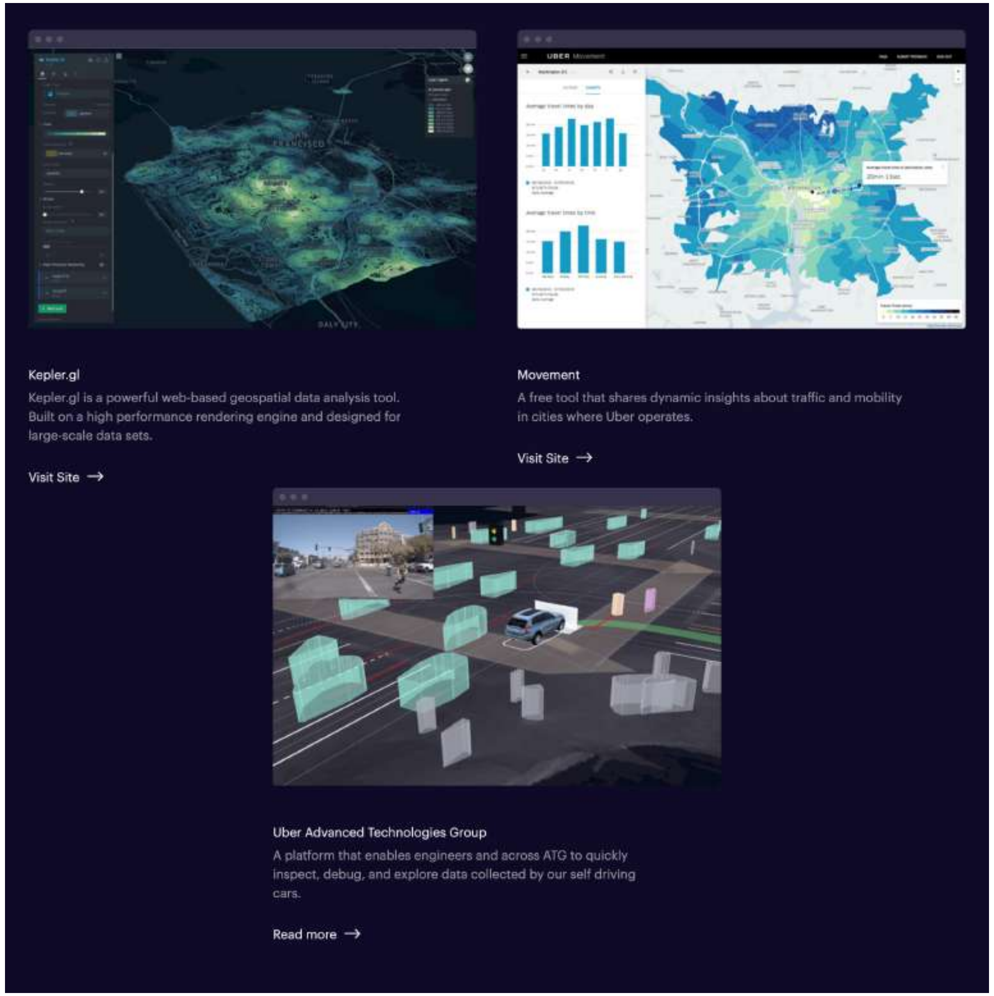

Agriculture precision farming
https://www.agrifac.com/sustainable-farming/precision-farming/

Use Case: Geo Processing @ telecommunications

- Network traffic analysis and optimization
- Signal performance along railway tracks
- Analysis of network coverage
- Footfall analytics

Use Case: Trips Analysis @ Uber

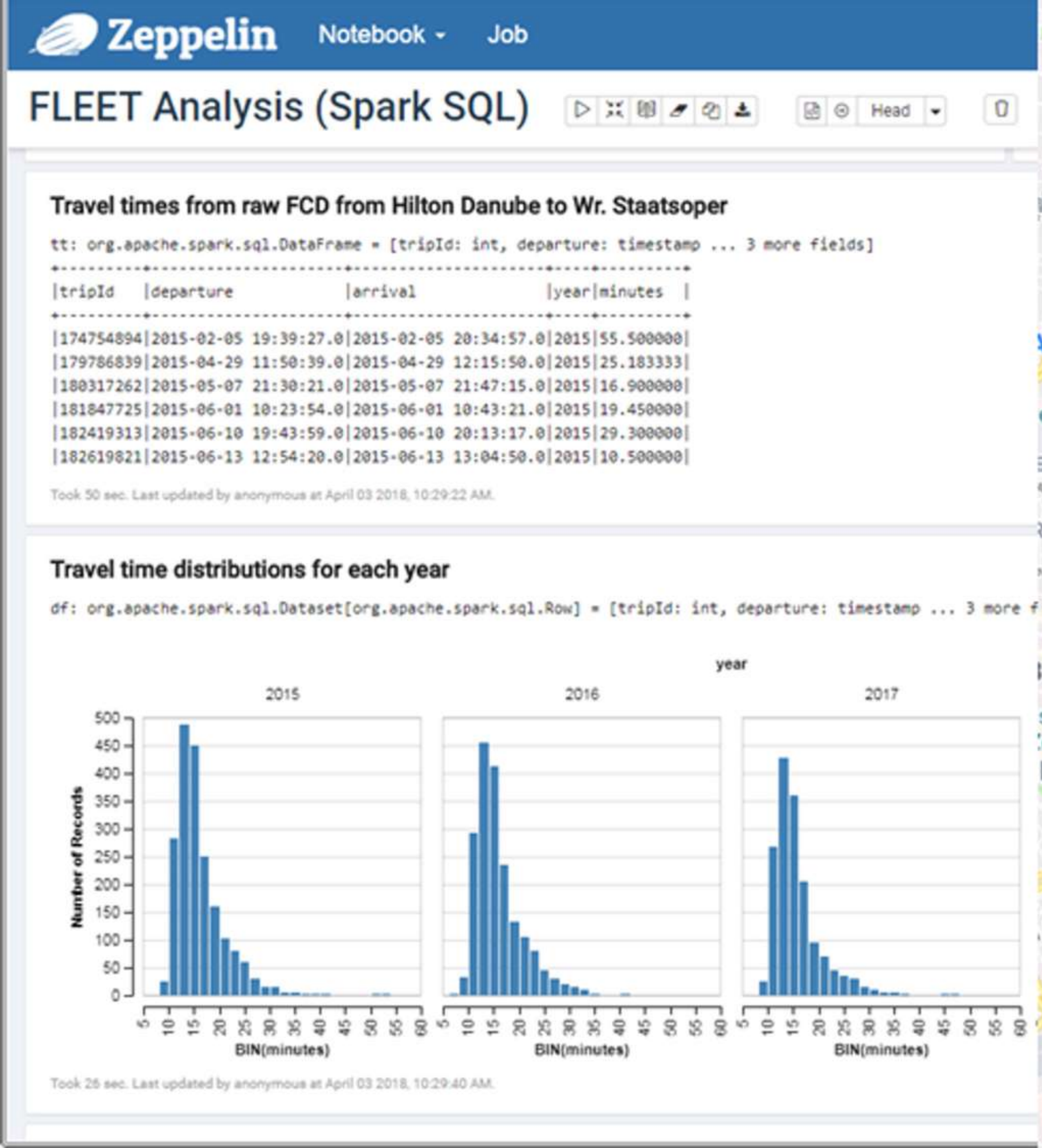
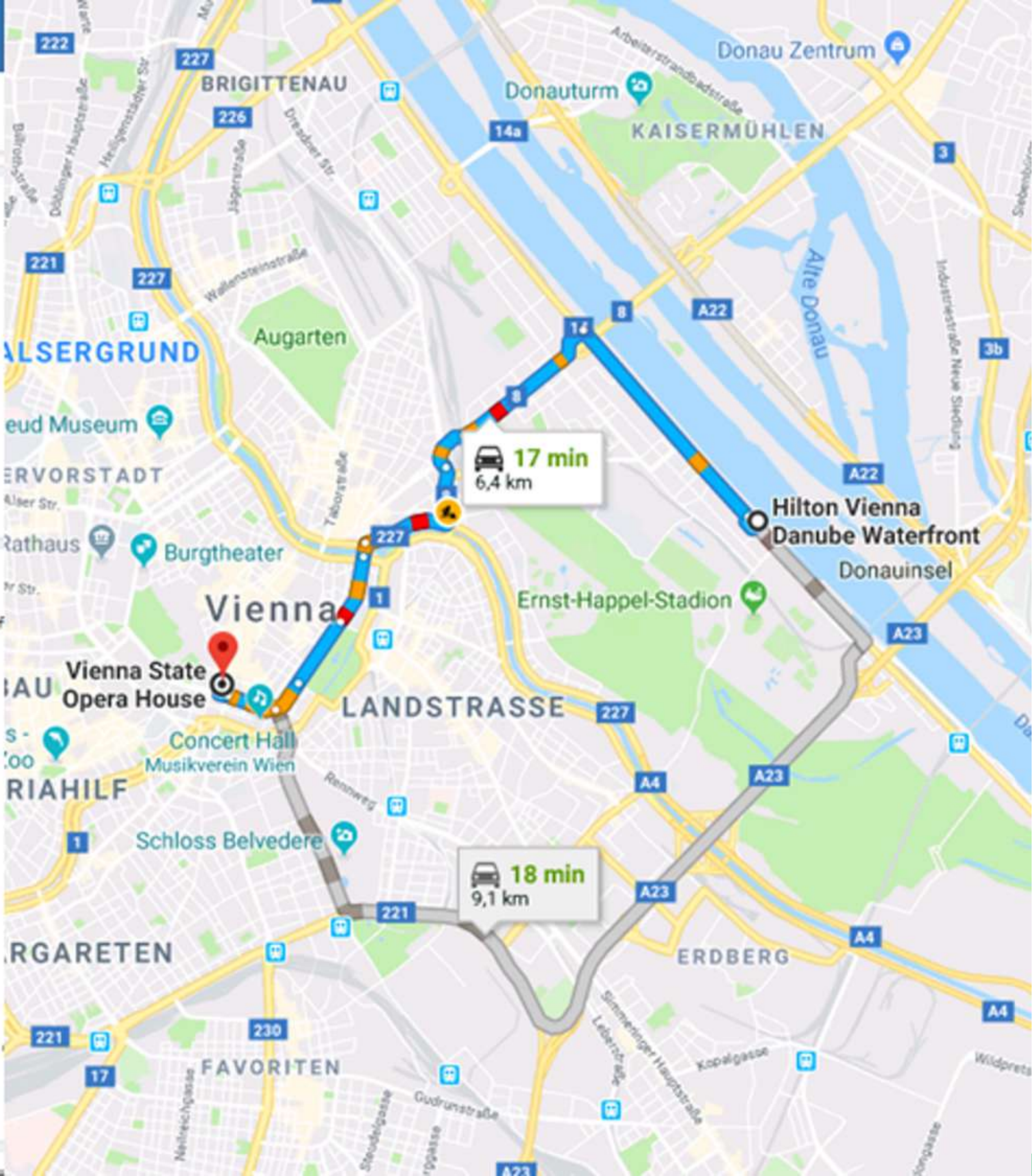
- What do trips look like?
- How can we reduce wait time and make more trips?
- Are there new products we should introduce?
- https://vis.gl/showcases



Source: slideshare.net, https://eng.uber.com/rethinking-gps/

Use Case: Traffic Jam Prediction based on GPS/FCD

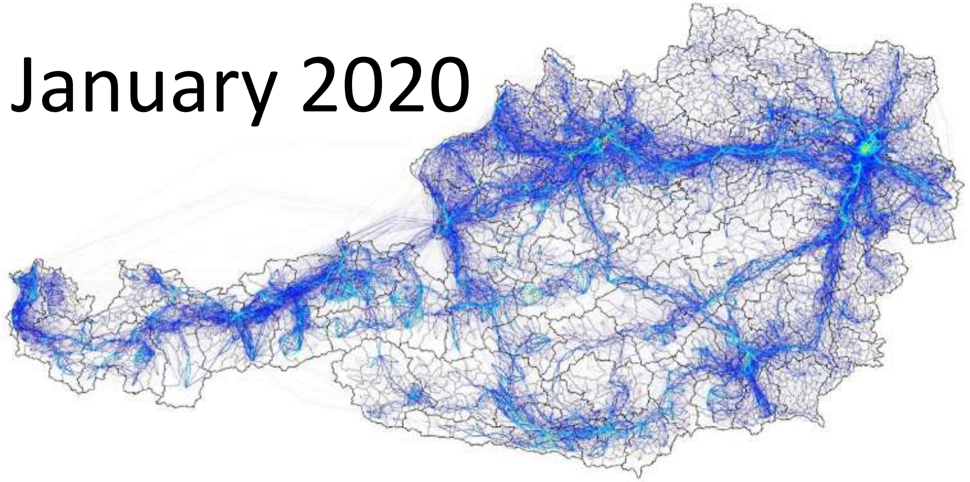
- Estimate average speed of cars on road
- Compare to the max speed on each street
- Use public traffic jam data as ground truth
- Train a model to predict traffic jams


Zeppelin
Notebook
Job
FLEET Analysis (Spark SQL)
Head
Travel times from raw FCD from Hilton Danube to Wr. Staatsoper
tt: org.apache.spark.sql.DataFrame = [tripId: int, departure: timestamp ... 3 more fields]
tripId	departure	arrival	year	minutes
174754894	2015-02-05 19:39:27.0	2015-02-05 20:34:57.0	2015	55.500000
179786839	2015-04-29 11:50:39.0	2015-04-29 12:15:50.0	2015	25.183333
180317262	2015-05-07 21:30:21.0	2015-05-07 21:47:15.0	2015	16.900000
181847725	2015-06-01 10:23:54.0	2015-06-01 10:43:21.0	2015	19.450000
182419313	2015-06-10 19:43:59.0	2015-06-10 20:13:17.0	2015	29.300000
182619821	2015-06-13 12:54:20.0	2015-06-13 13:04:50.0	2015	10.500000
Took 50 sec. Last updated by anonymous at April 03 2018, 10:29:22 AM.
Travel time distributions for each year
df: org.apache.spark.sql.Dataset[org.apache.spark.sql.Row] = [tripId: int, departure: timestamp ... 3 more f
year
2015
2016
2017
Number of Records
BIN(minutes)
Took 26 sec. Last updated by anonymous at April 03 2018, 10:29:40 AM.

17 min
6,4 km
18 min
9,1 km
Hilton Vienna
Danube Waterfront
Vienna State
Opera House
Vienna
BRIGITTENAU
KAISERMÜHLEN
LANDSTRASSE
FAVORITEN
ERDBERG
Augarten
Donauturm
Donau Zentrum
Burgtheater
Concert Hall
Musikverein Wien
Schloss Belvedere
Ernst-Happel-Stadion
Donauinsel
Alte Donau

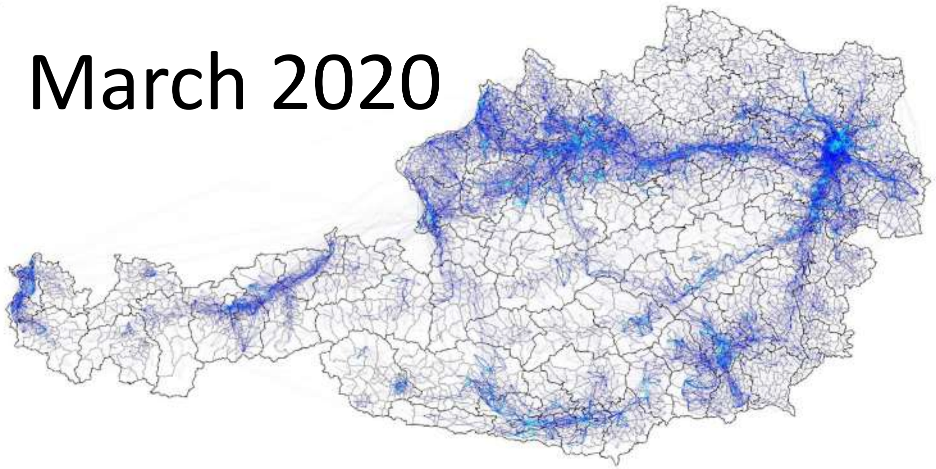
COVID-19 mobility insights

Powered by large scale geospatial algorithms

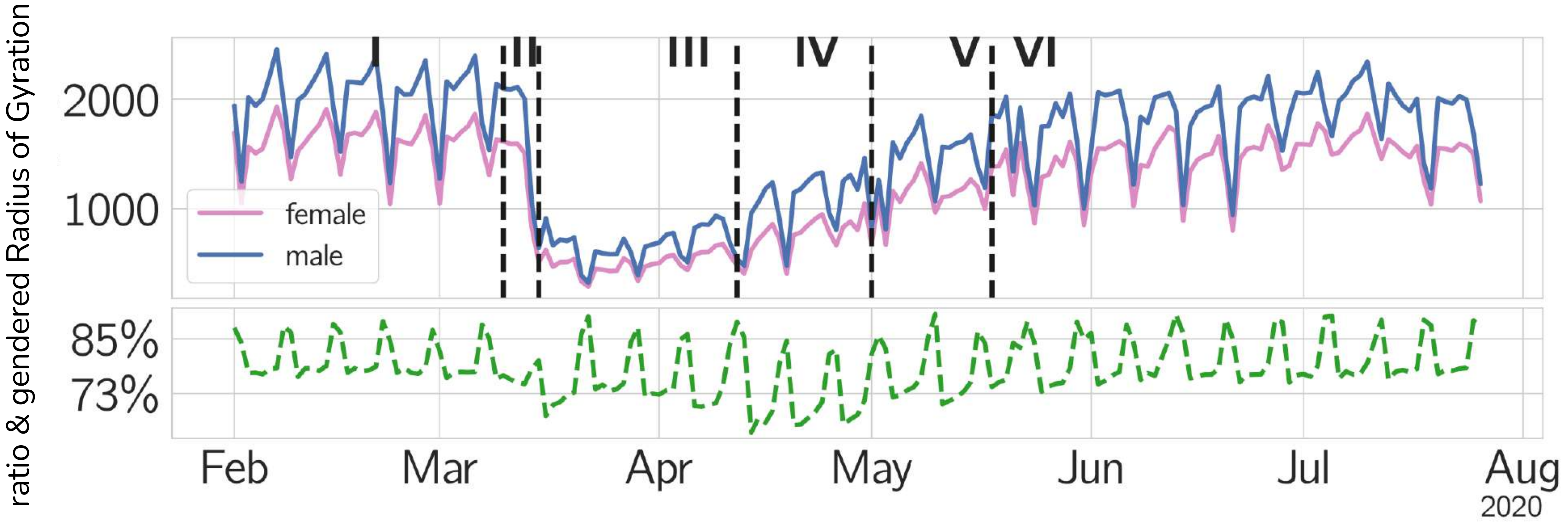
January 2020



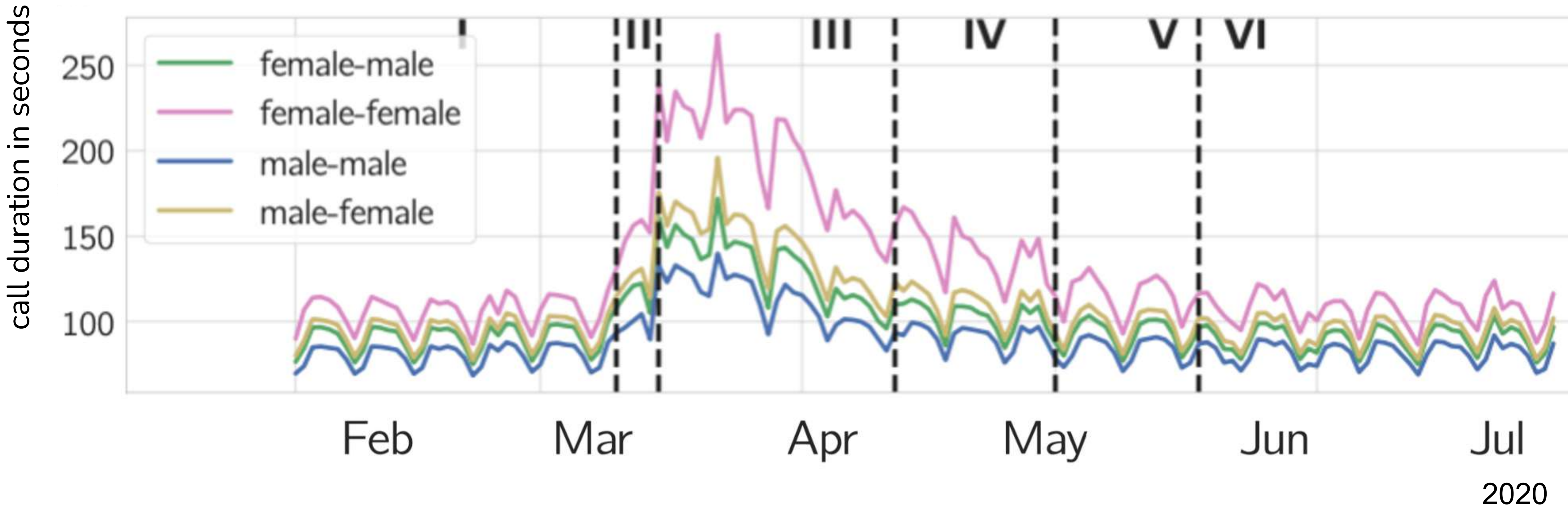
March 2020



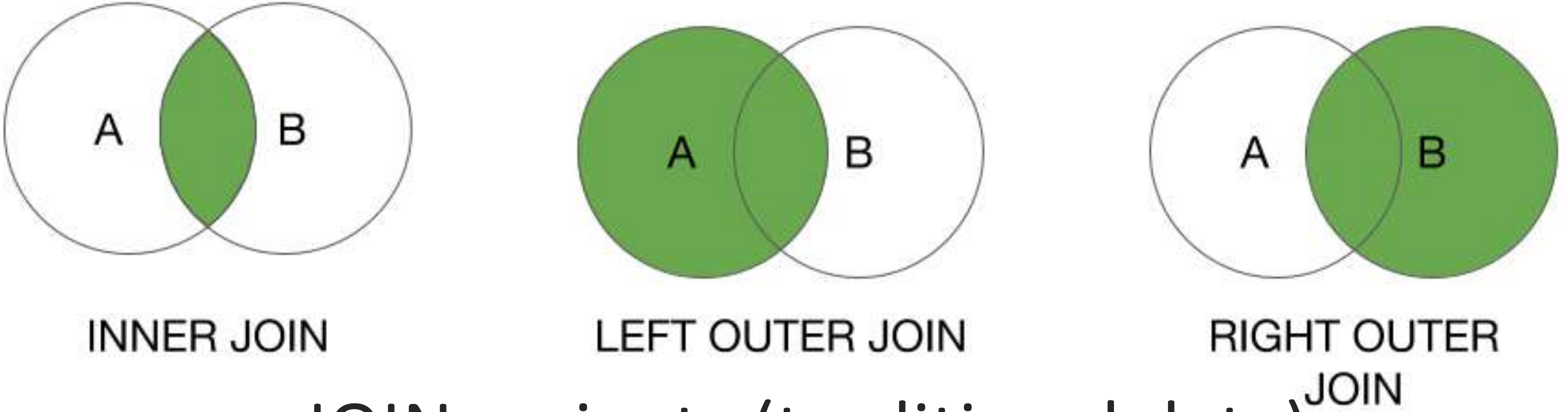
Gendered mobility



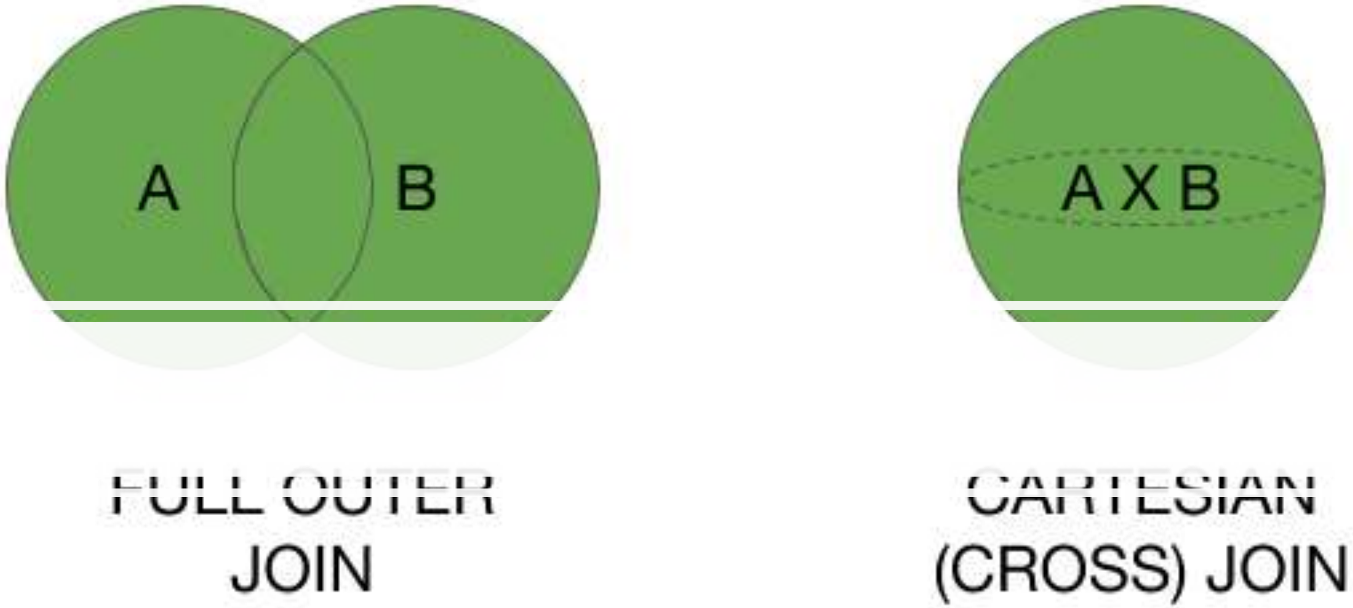
Gendered interactions



Spatial analytics

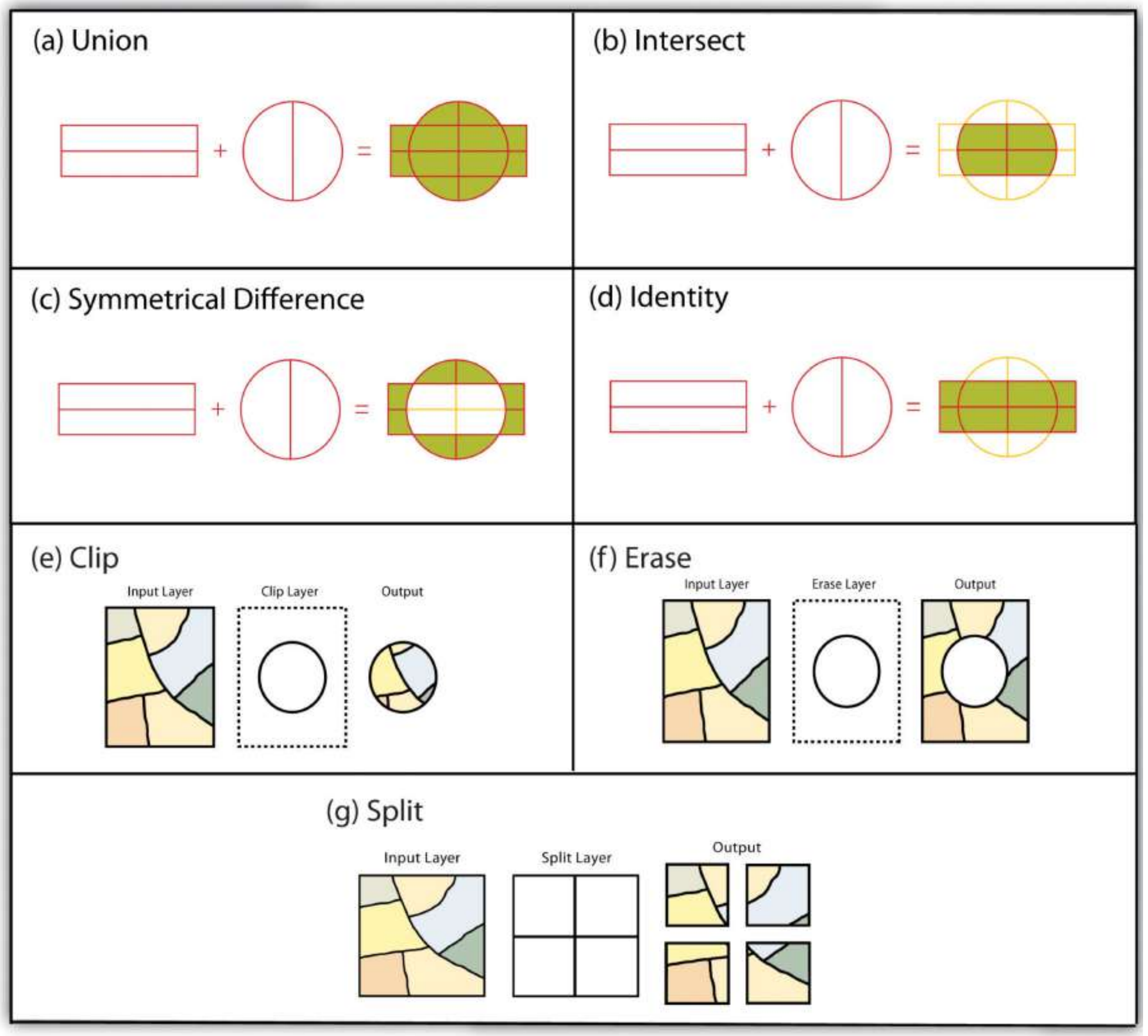


JOIN variants (traditional data)



https://en.wikipedia.org/wiki/Relational_algebra different JOIN types

Spatial operations



https://saylordotorg.github.io/text_essentials-of-geographic-information-systems/s11-02-multiple-layer-analysis.html

You know, like...

ORACLE is to PostgreSQL

as

ORACLE SPATIAL is to PostGIS

Geospatial SQL

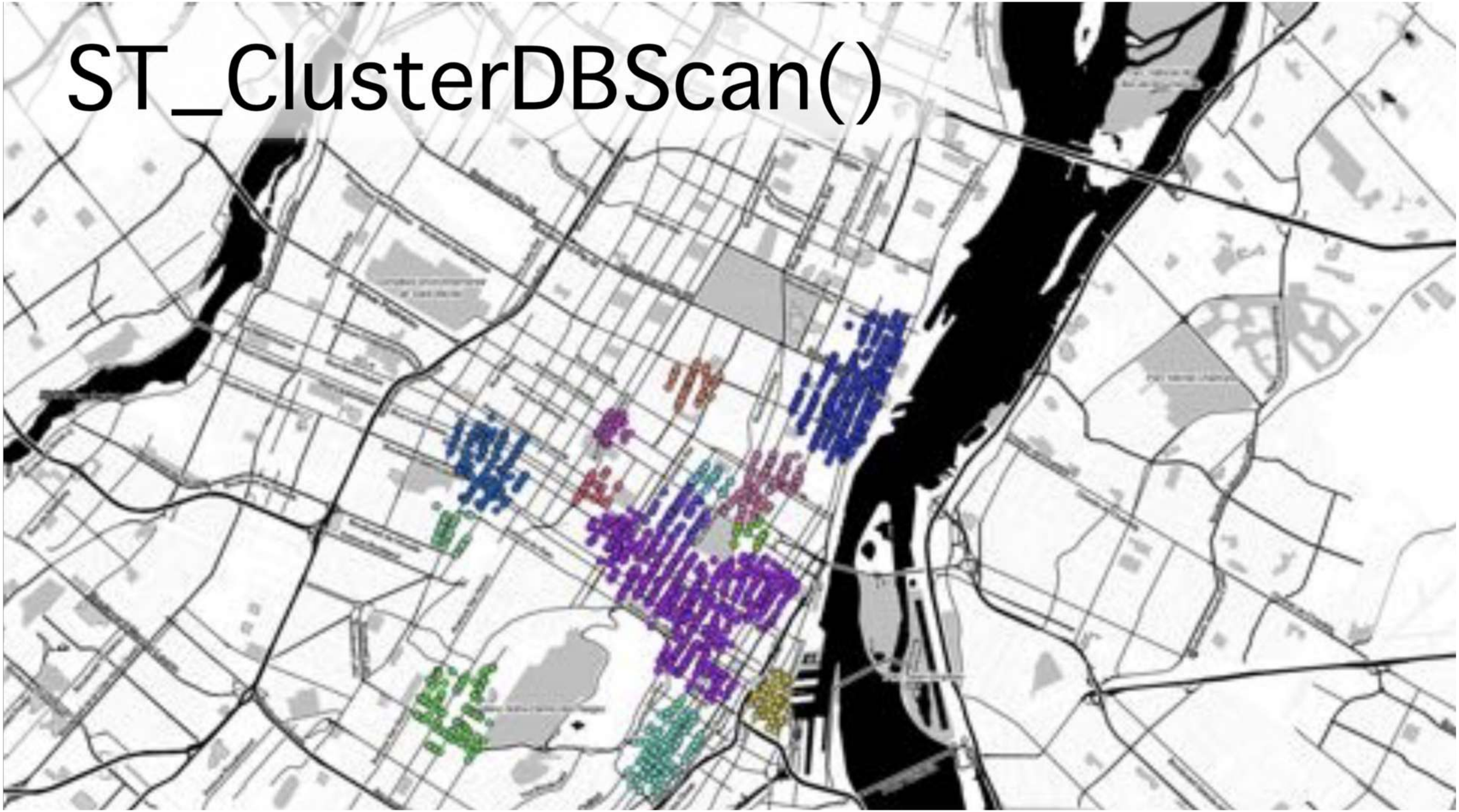
```sql
SELECT superhero.name
FROM city, superhero
WHERE ST_Contains(city.geom, superhero.geom)
AND city.name = 'Gotham';
```

http://workshops.boundlessgeo.com/postgis-intro/spatial_relationships.html

speeding up queries in a database

- create an (scalar) index
- But geospatial data is multi dimensional. Prevent complete cross product by filtering te data first:
  - geo hash
  - space filling curves (Hilbert Kurve, …)
  - R-tree
  - Quad-tree
  - KD-tree
  - RB-tree

ST_ClusterDBScan()



Postgres spatial addons

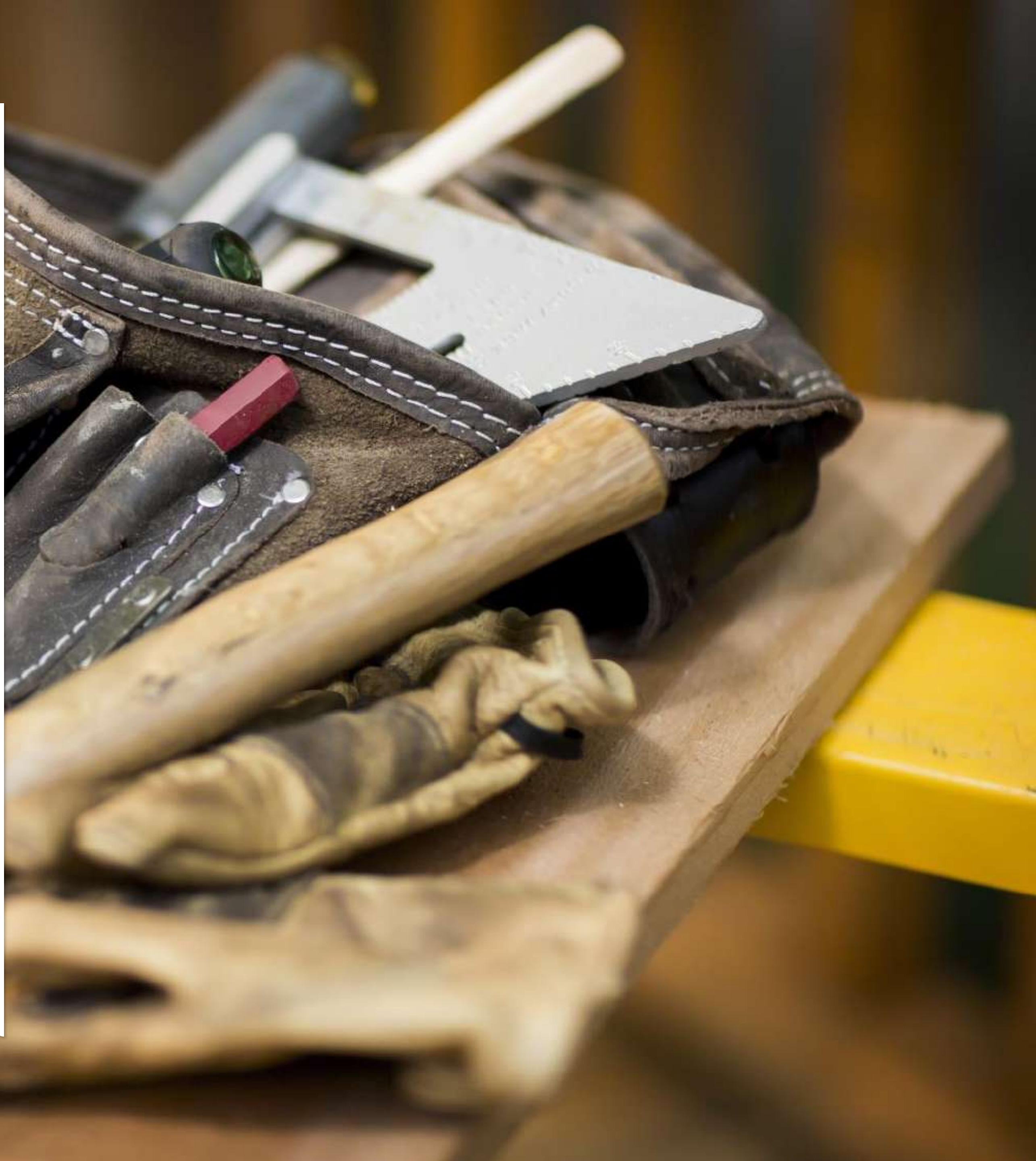
MobilityDB

pgRouting Project

pgRouting extends the PostGIS / PostgreSQL geospatial database to provide geospatial routing functionality.

Some tools

- python
  - geopandas
  - pysal
  - xarray
- R
  - rspatial
  - SP
- Notebooks:
  - jupyter
  - rmarkdown



Spatial Visualization

- **QGIS**
- ArcGIS
- **MapBox**
- Carto
- WMF/ WPS Services via geoserver
- [kepler.gl]

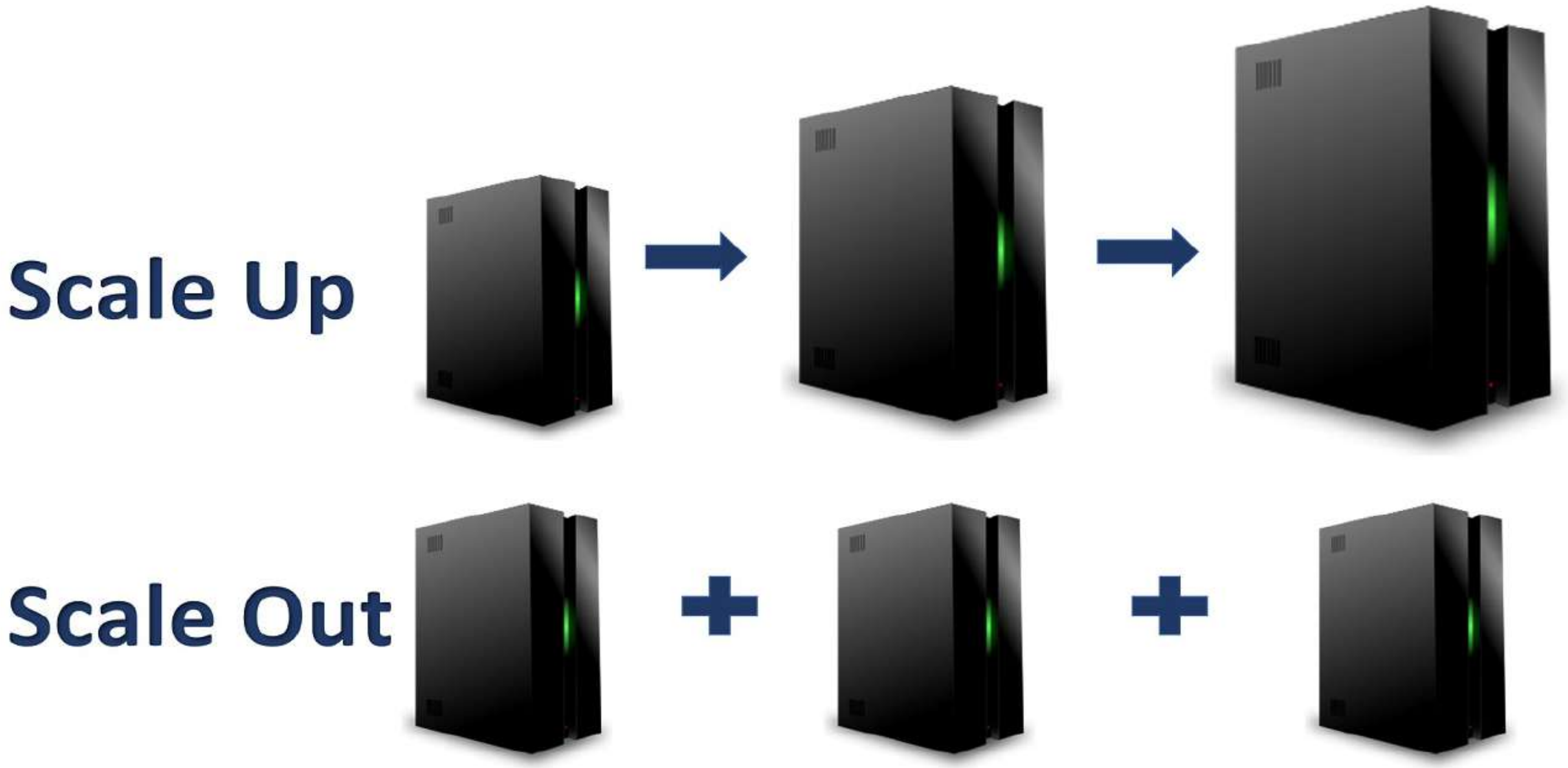
Common analytical tasks

- Clustering
- Watershed analytics
- Interpolation (kriging)
- Pattern detection
- Geospatial Forecasting

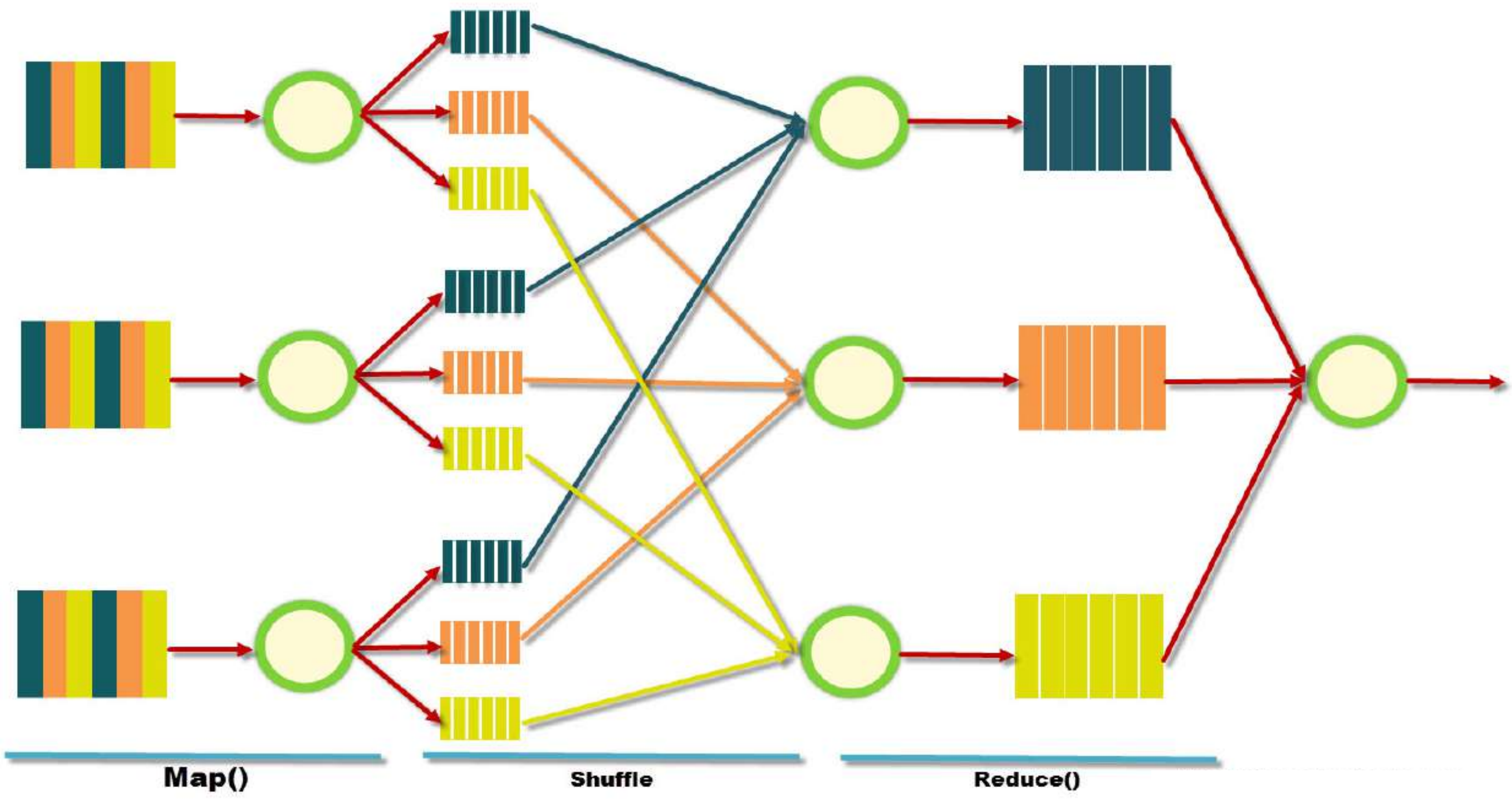
Scaling spatial data processing

LIMITATIONS OF CLASSICAL RDBMS

scalability
scale up only



https://www.syslink-xandria.com/blog/scaling-up-vs-scaling-out-sap-on-the-public-cloud



MAP-REDUCE PARADIGM

http://blog.sqlauthority.com

pola.rs

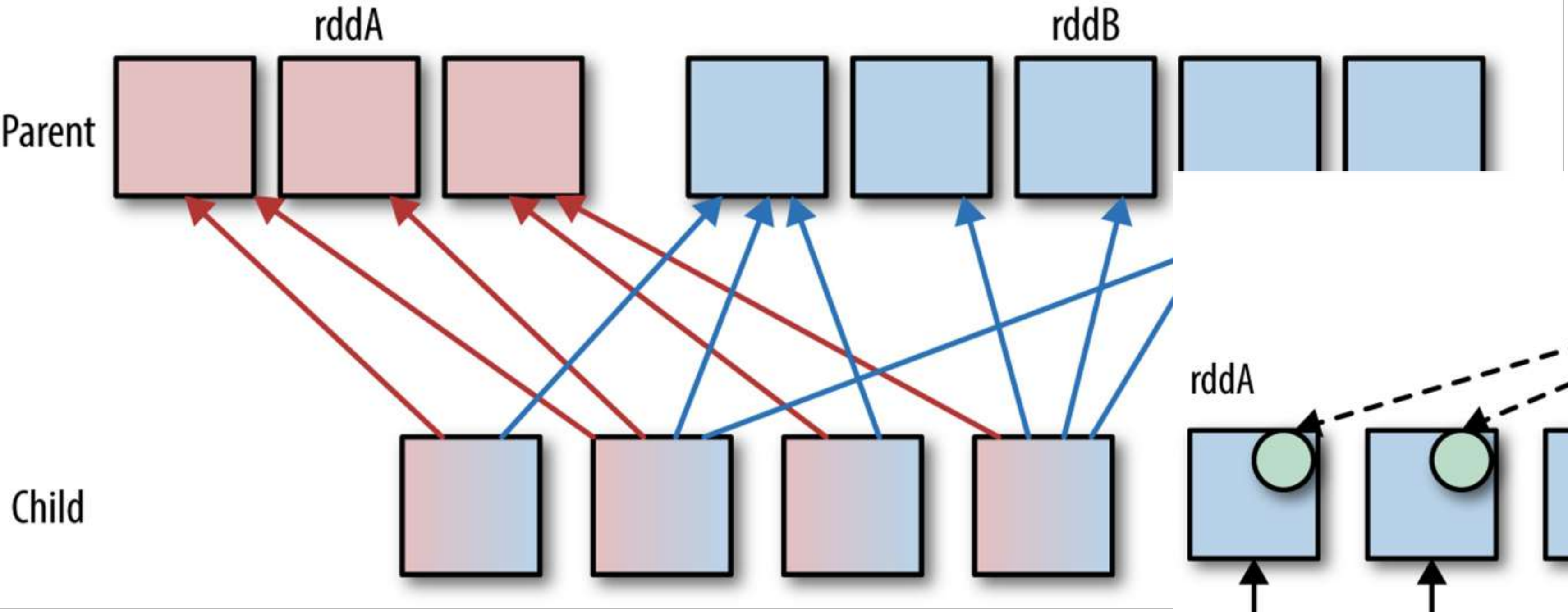
Making it faster
(medium sized data)

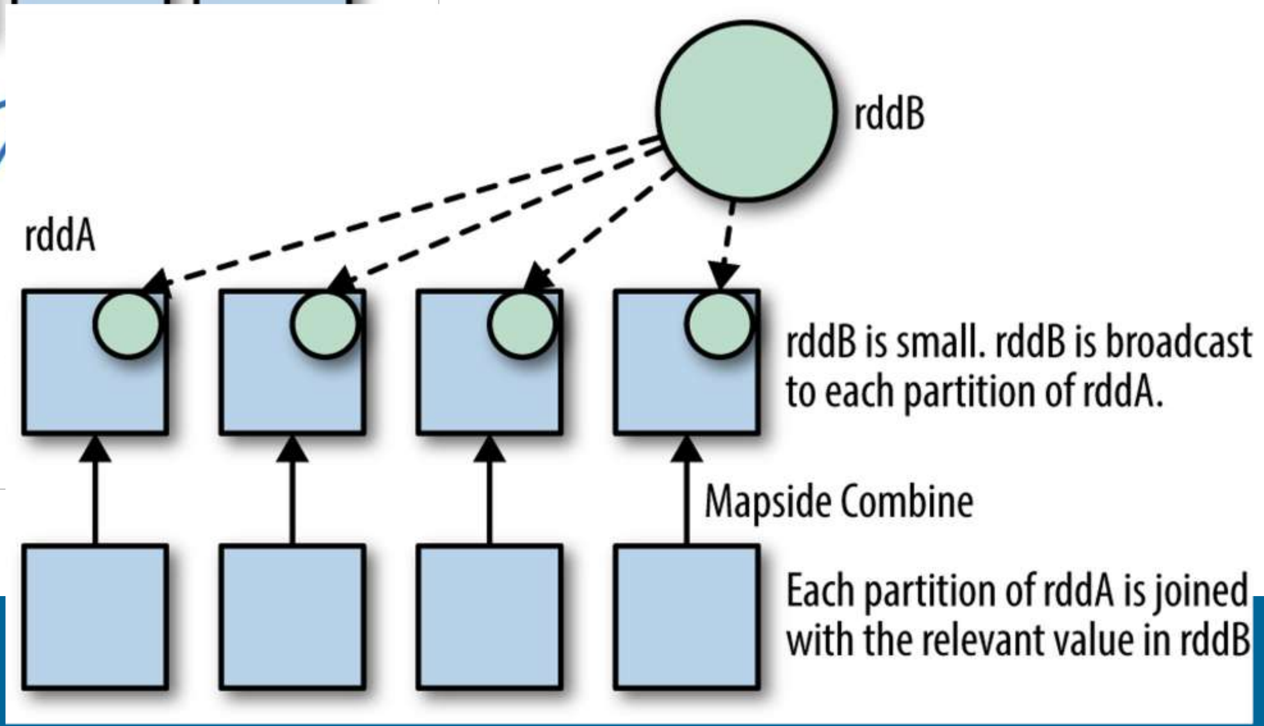
- From simple single node without concurrency (pandas)
- To LLVM native code (Ray, Modin)
- DuckDB
- Executed on multiple processes or a couple of machines (Dask, Modin)
- GPU acceleration using cuDf (RAPIDS)
- 100s of nodes Spark

Problem of large data & parallel processing

- PostGIS analytical operations only recently started to be parallelized
    - http://s3.cleverelephant.ca/2017-cdb-postgis.pdf, http://blog.cleverelephant.ca/2017/10/parallel-postgis-2.html
    - Fromt version 12 on http://blog.cleverelephant.ca/2019/05/parallel-postgis-4.html
- ArcGIS only recently added parallel processing https://www.esri.com/arcgis-blog/products/arcgis-pro/analytics/parallel-geoprocessing-in-arcgis-pro/
- Naive spatial join collapses easily due to full cross product
- Larger and faster spatial data needs more efficient and scalable processing
- Scaling data means partitioning – how to efficiently partition spatial data?
    - Hotspotting?
    - Borders?
- Cloud DWH (BigQuery & Snowflake) as of 2024 already support geospatial functionality
- sedona.apache.org for Spark available

Broadcast (spatial) join explained





scaling out processing

- Python: dask & geopandas https://r-shekhar.github.io/posts/spatial-joins-geopandas-dask.html
- DuckDB spatial extension https://duckdb.org/2023/04/28/spatial.html
- Hadoop is a cheap general purpose compute infrastructure
- full solutions:
  - geomesa (well supported)
  - geowave
  - rasterframes
- only distributed geospatial SQL, varying degree of optimizations (indices, spatial partitioning)
  - Apache Sedona (spark)
  - harsha2010/magellan (spark)
- Most are based on JTS (java topology suite) in some way
- Most offer a SQL based interface similar to postGIS
- https://github.com/rapidsai/cuspatial

Geostatistics

Spatial autocorrelation / Moran`s I

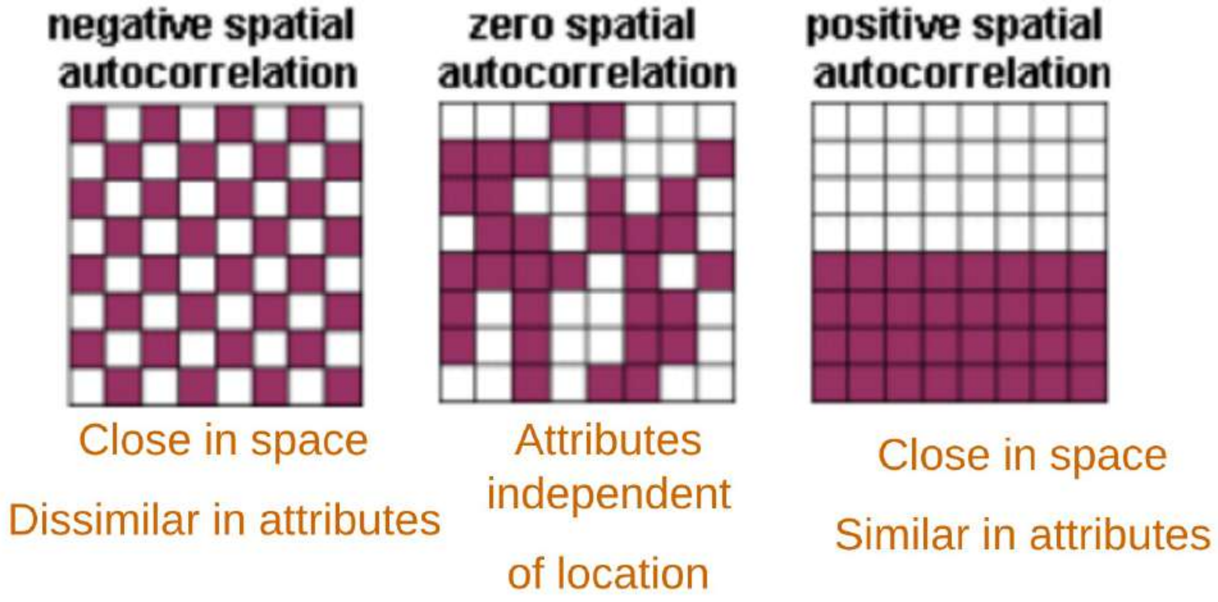
spatial dependence: Phenomena that are close to one another in space are more likely to have similar characteristics than those that are farther apart

Global Moran's I is a measure of the overall clustering of the spatial data. It is defined as

$$I = \frac{N}{W} \frac{\sum_{i=1}^{N} \sum_{j=1}^{N} w_{ij} (x_i - \bar{x})(x_j - \bar{x})}{\sum_{i=1}^{N} (x_i - \bar{x})^2}$$

where

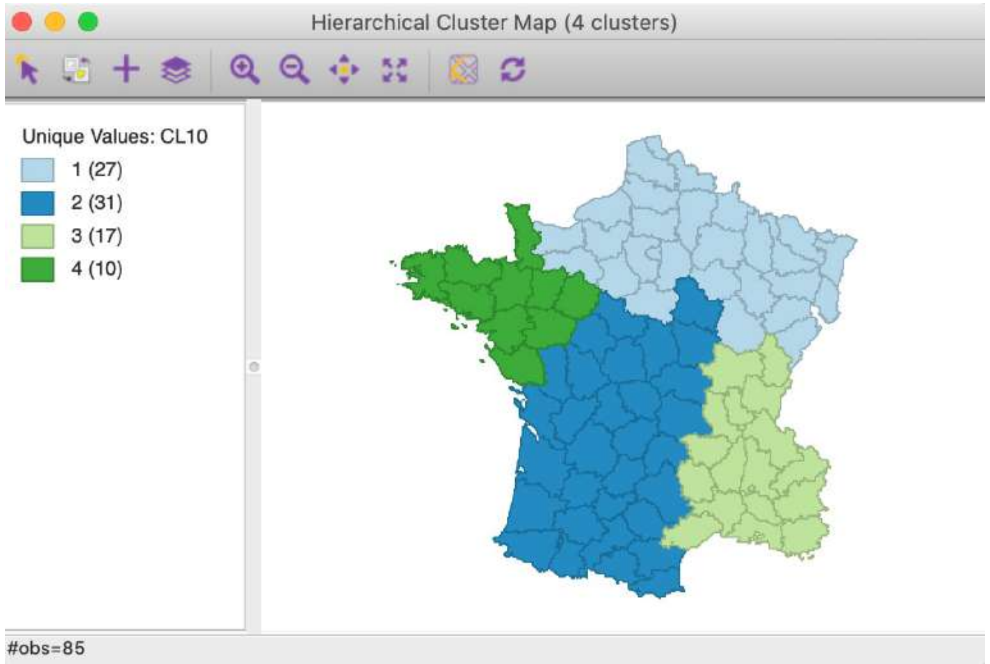
- N is the number of spatial units indexed by i and j;
- x is the variable of interest;
- $\bar{x}$ is the mean of x;
- w_{ij} are the elements of a matrix of spatial weights with zeroes on the diagonal (i.e., $w_{ii} = 0$);
- and W is the sum of all w_{ij} (i.e. $W = \sum_{i=1}^{N} \sum_{j=1}^{N} w_{ij}$).

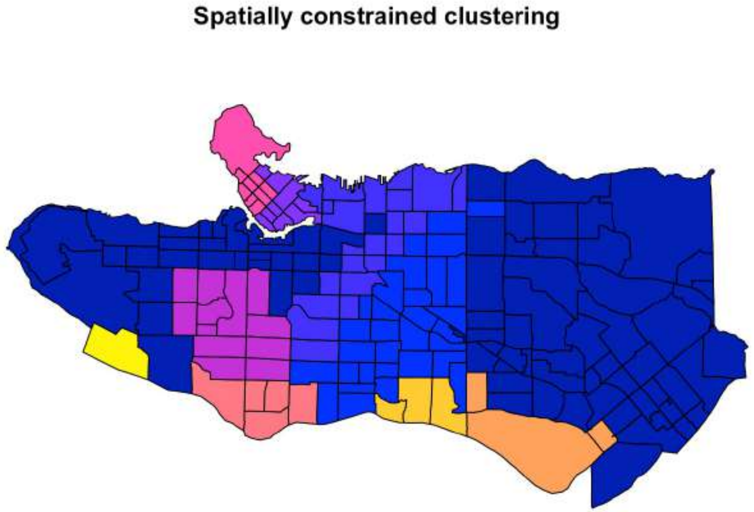


https://www.docsity.com/en/spatial-autocorrelation-gis-and-mapping-lecture-slides/380010/

Spatial clustering

- Geolocation
- Attributes
- Common methods (with spatial extensions)
  - Hierarchical
  - Kmeans
- SKATER



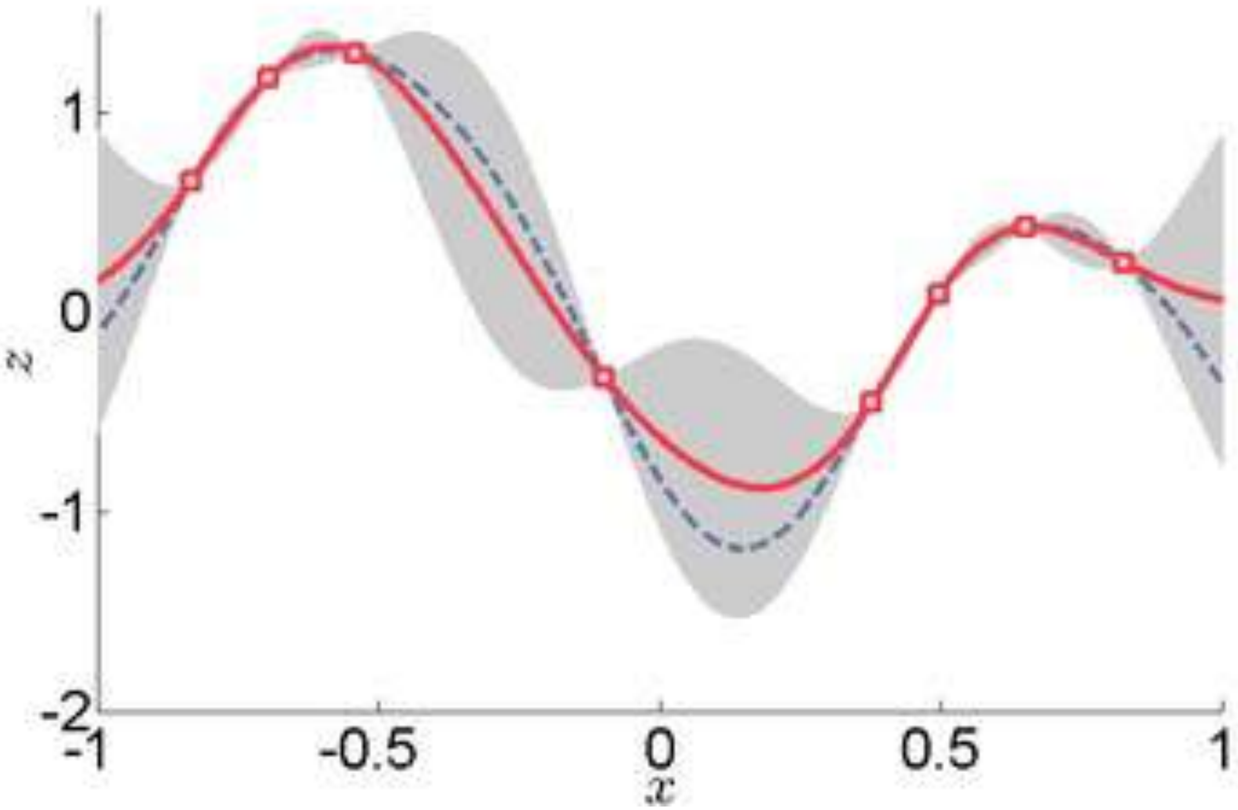


https://geodacenter.github.io/workbook/9a_spatial1/lab9a.htm
https://www.dshkol.com/post/spatially-constrained-clustering-and-regionalization/ l

Kriging

- Spatial interpolation
- Gaussian process regression

- Alternative [Integrated Nested Laplace Approximation]



https://en.wikipedia.org/wiki/Kriging

Getis-Ord GI*

- Z-score measuring spatial clustering
- Looking at each feature within the context of neighboring features
- Satistically significant hot spot must have a high value and be surrounded by other features with high values as well

The Getis-Ord local statistic is given as:

$$G_i^* = \frac{\sum_{j=1}^{n} w_{i,j} x_j - \bar{X} \sum_{j=1}^{n} w_{i,j}}{S \sqrt{\frac{\left[n \sum_{j=1}^{n} w_{i,j}^2 - \left(\sum_{j=1}^{n} w_{i,j} \right)^2 \right]}{n-1}}} \qquad (1)$$

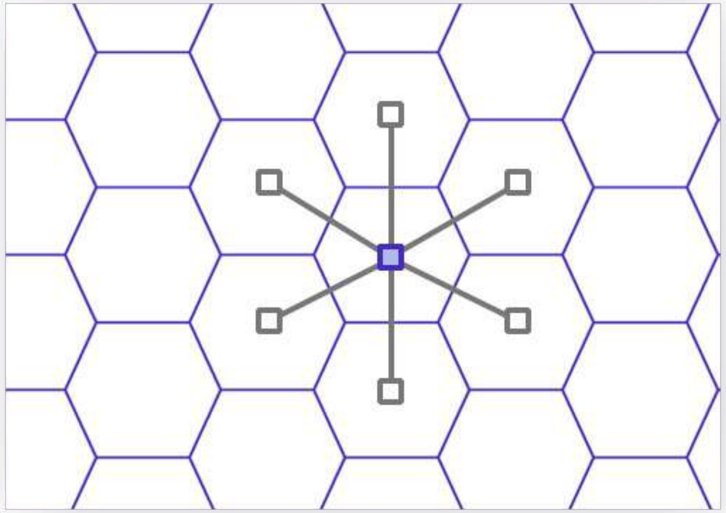
where x_j is the attribute value for feature j, $w_{i,j}$ is the spatial weight between feature i and j, n is equal to the total number of features and:

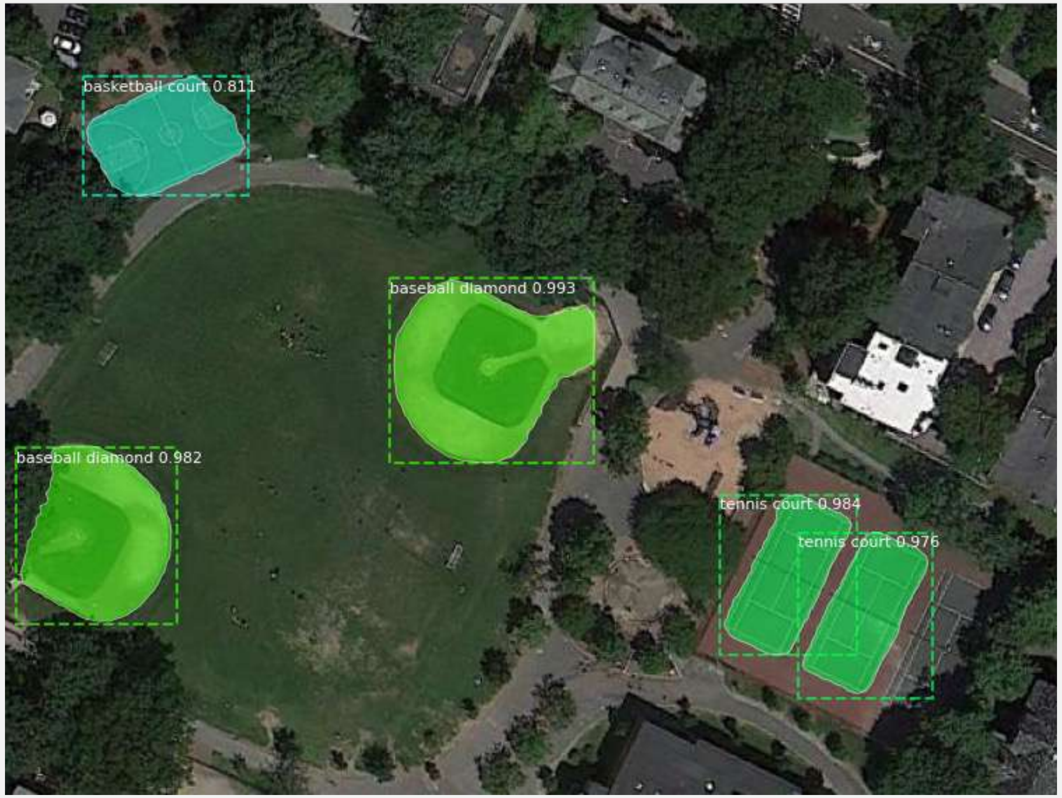
$$\bar{X} = \frac{\sum_{j=1}^{n} x_j}{n} \qquad (2)$$

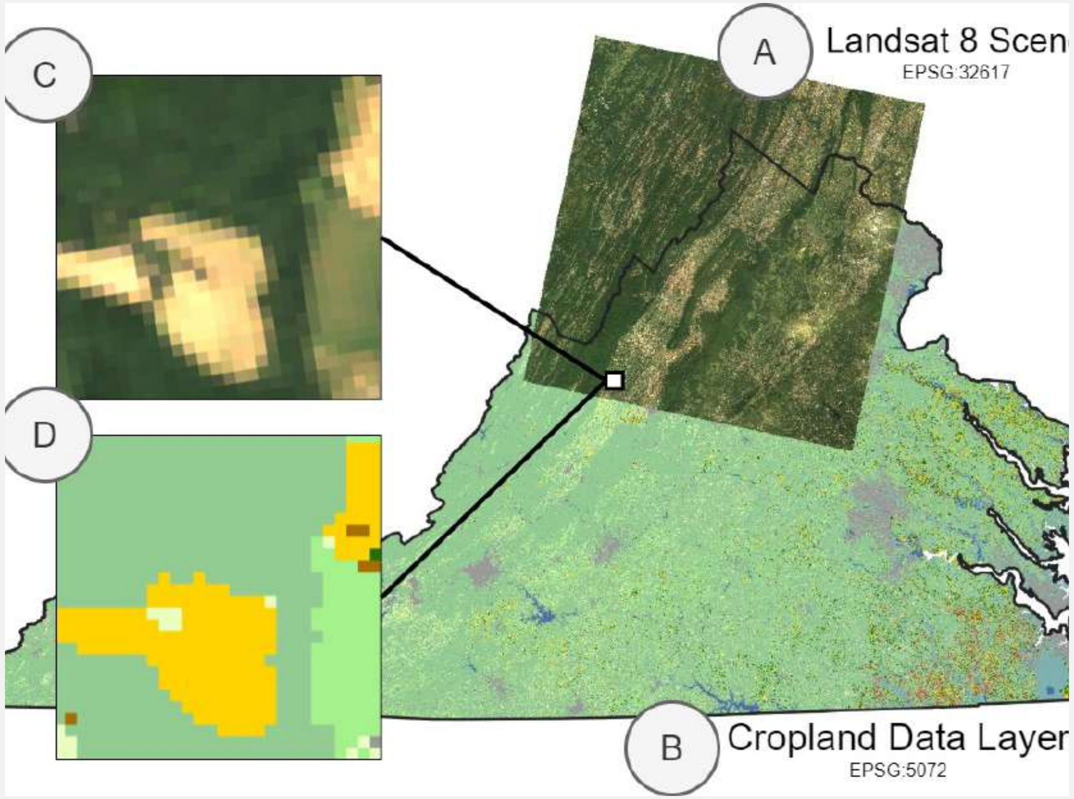
$$S = \sqrt{\frac{\sum_{j=1}^{n} x_j^2}{n} - (\bar{X})^2} \qquad (3)$$

The G_i^* statistic is a z-score so no further calculations are required.

H3 hexagon can make calculation
Of similar statistics much more efficient







Machine learning

https://github.com/microsoft/torchgeo

https://www.microsoft.com/en-us/research/project/geospatial-machine-learning/

GEOSPATIAL STATISTICS

An Introduction
Georg Heiler UII 2023

Great examples to learn from

- https://github.com/r-shekhar/NYC-transport, https://towardsdatascience.com/geospatial-operations-at-scale-with-dask-and-geopandas-4d92d00eb7e8
- https://github.com/gboeing/urban-data-science
- https://github.com/geoHeil/spatial-heatmaps
- https://automating-gis-processes.github.io/2016/index.html
- https://www.kaggle.com/headsortails/be-my-guest-recruit-restaurant-eda
- cythonized geopandas http://matthewrocklin.com/blog/work/2017/09/21/accelerating-geopandas-1
- https://geocompr.robinlovelace.net
- https://cran.r-project.org/web/views/Spatial.html
- https://cran.r-project.org/web/views/SpatioTemporal.html

Links & references

- https://de.slideshare.net/ChristophKrner/large-scale-geo-processing-on-hadoop
- http://blog.cleverelephant.ca/2017/10/parallel-postgis-2.html
- http://blog.cleverelephant.ca/2017/12/postgis-scaling.html
- http://s3.cleverelephant.ca/2018-postgis-for-managers.pdf
- https://docs.google.com/presentation/d/14lf1TsVO4Wq7ykgHjIiXYksvzWBW5XvuxJh2CrtraHc/edit#slide=id.g392f8bb753_0_561
- http://shop.oreilly.com/product/0636920032175.do
- http://www.highstat.com/index.php/beginner-s-guide-to-regression-models-with-spatial-and-temporal-correlation
- https://geostat-course.org
- https://github.com/andrewzm/FRK
- https://www.esri.com/arcgis-blog/products/arcgis-pro/analytics/new-clustering-tools-in-arcgis-pro-2-1-more-machine-learning-at-your-fingertips/?rmedium=redirect&rsource=blogs.esri.com%2Fesri%2Farcgis%2F2018%2F01%2F22%2Fpro-2-1-new-clustering-tools
- http://xarray.pydata.org/en/stable/ , https://ncar.github.io/PySpark4Climate/sparkxarray/overview/
- https://www.paradigm4.com/try_scidb/
- https://github.com/r-spatial/spdep/
- http://geonode.org
- https://carto.com/blog/inside/postgres-parallel/
- https://anitagraser.com
- https://www.wiley.com/en-us/Statistics+for+Spatio+Temporal+Data-p-9780471692744
- https://medium.com/@christoph.k.rieke/essential-geospatial-python-libraries-5d82fcc38731 https://github.com/sacridini/Awesome-Geospatial
- https://cloudnativegeo.org/

Links to tutorials

- https://pythongis.org/
- https://sustainability-gis.readthedocs.io/en/latest/lessons/L1/intro-to-python-geostack.html
- https://www.whiteboxgeo.com/
- https://courses.spatialthoughts.com/python-foundation.html
- https://geographicdata.science/book/notebooks/03_spatial_data.html
- https://geocompx.org/post/2023/ogh23/
- https://movingpandas.org