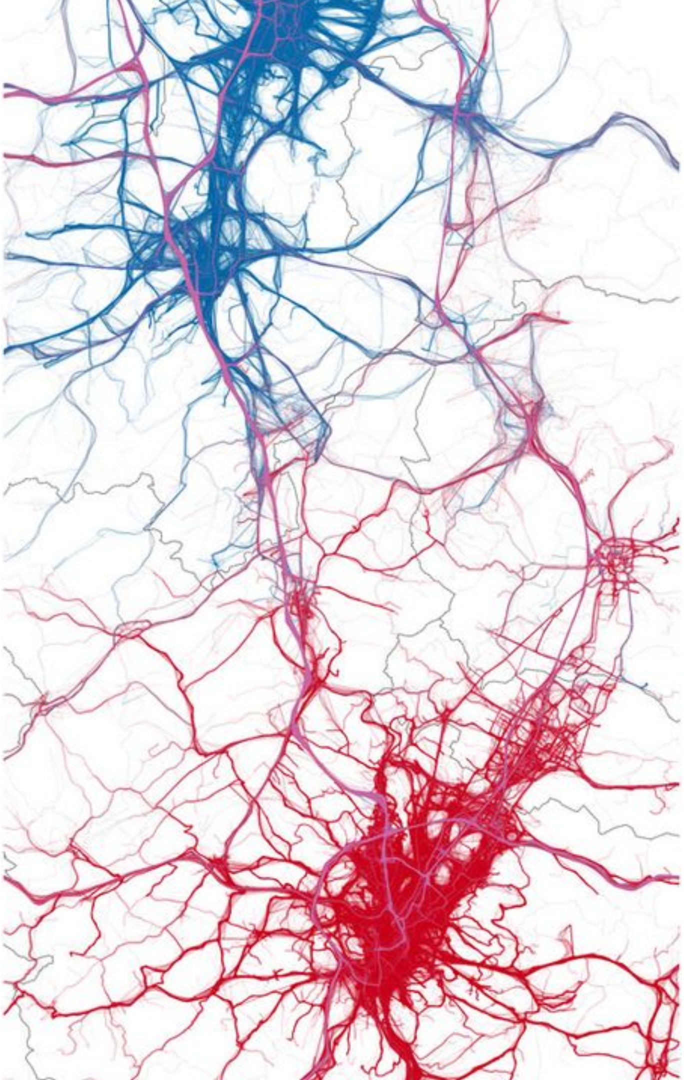


Consiglio Nazionale delle Ricerche

SCUOLA NORMALE SUPERIORE

03 Spatial Data Analysis

Todays contents

- A bit of (confusing) terminology

- Basic spatial data types
  - Raster vs. Vectorial

- Basic spatial operations
  - intersection, union, difference
  - buffering
  - spatial join

- Simple spatial patterns and concentration measures
  - Moran’s I
  - Geary’s C

Basic Terms and Concepts

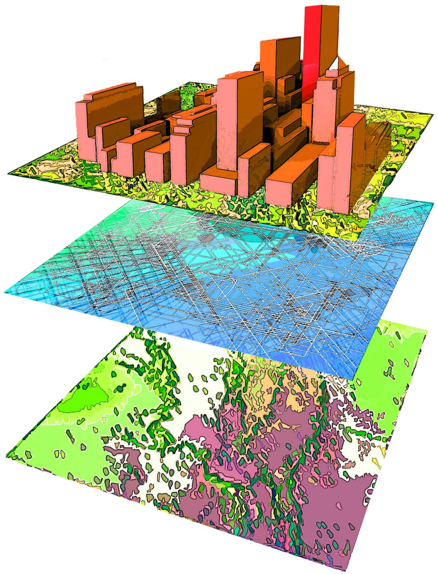
mainly coming from the GIS world

Layers

- Information is organized in separate “sheets”, named **(thematic) layers**
  - They refer to the same geographical area, but have independent lives

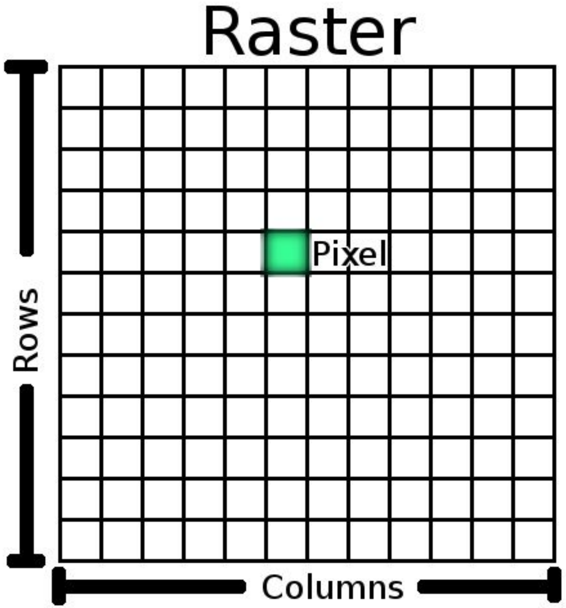
- Each layer contains a set of objects, usually of the same nature, type and/or logical function
  - E.g. a layer for the street network, one for the position of buildings, etc.

- Layers can be processed and combined together
  - E.g. to identify the buildings within a neighborhood,



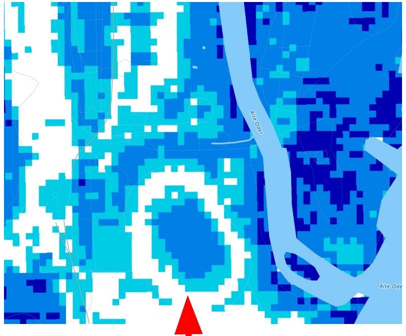
Raster Layers

- Divide the space into a regular grid of squares
  - Equivalent to an image made of pixels
- Associate some information at each cell / pixel
  - single-band raster = one attribute value
  - multi-band raster = several attributes
- The size of cells defines the resolution of the data
- Typically come from satellite sensors

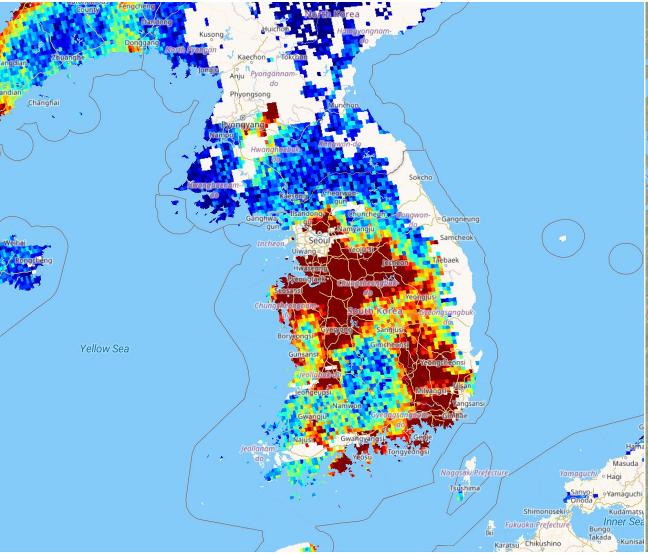


Raster Layers - examples

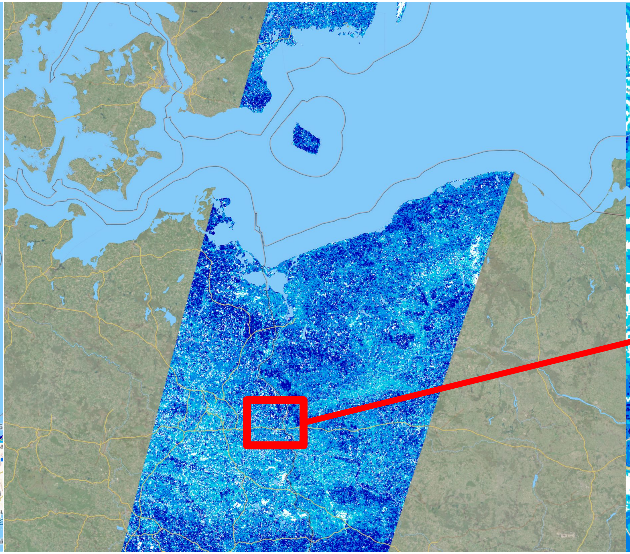
From Copernicus/Sentinel satellites

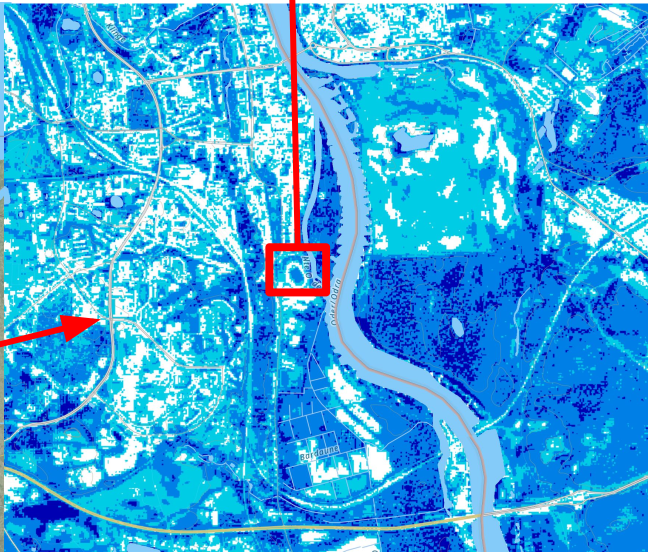


NO2 in S. Korea



Moisture in Poland



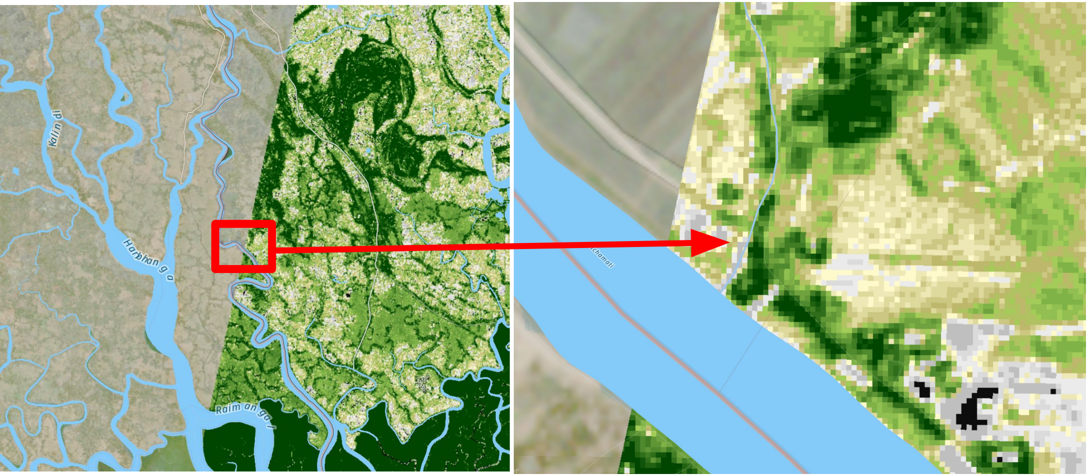


https://browser.dataspace.copernicus.eu/

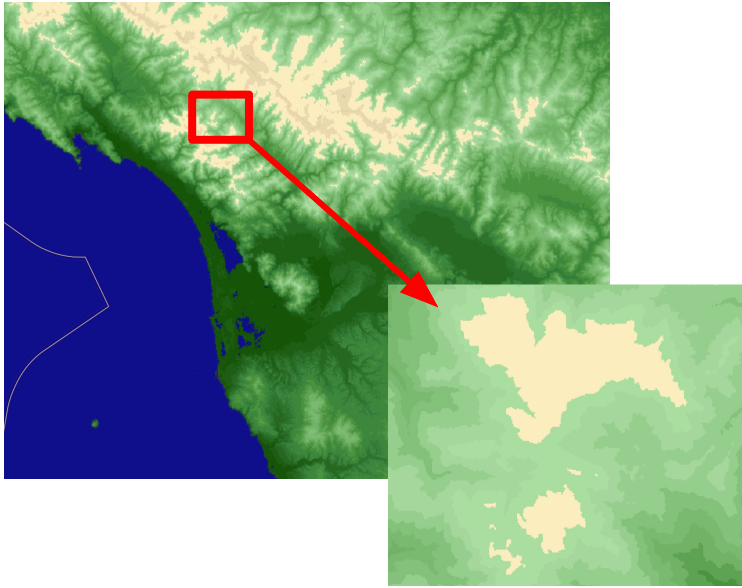
Raster Layers - examples

From Copernicus/Sentinel satellites

Normalized Difference Vegetation Index (NDVI)
in Sundarbans, Bangladesh



Digital Elevation Model (DEM)
in Tuscany



https://browser.dataspace.copernicus.eu/

Raster Layers

- In summary: a pixelized version of Earth
- Kind of Minecraft
  - yet not limited to visible channels

(unsurprisingly)

Vector data model

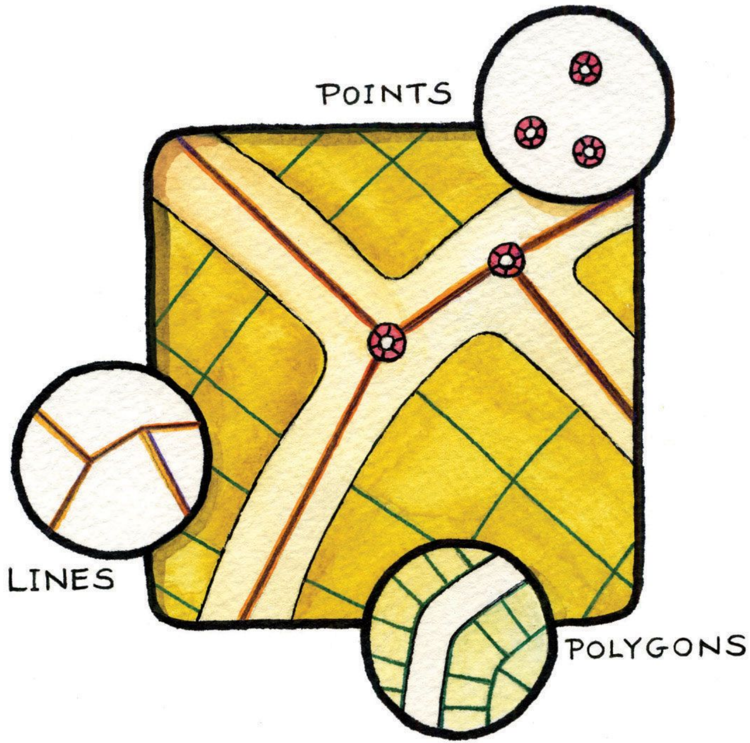
It uses discrete objects to represent spatial features:

1. representing `points`, `lines`, and `polygons` on an empty space

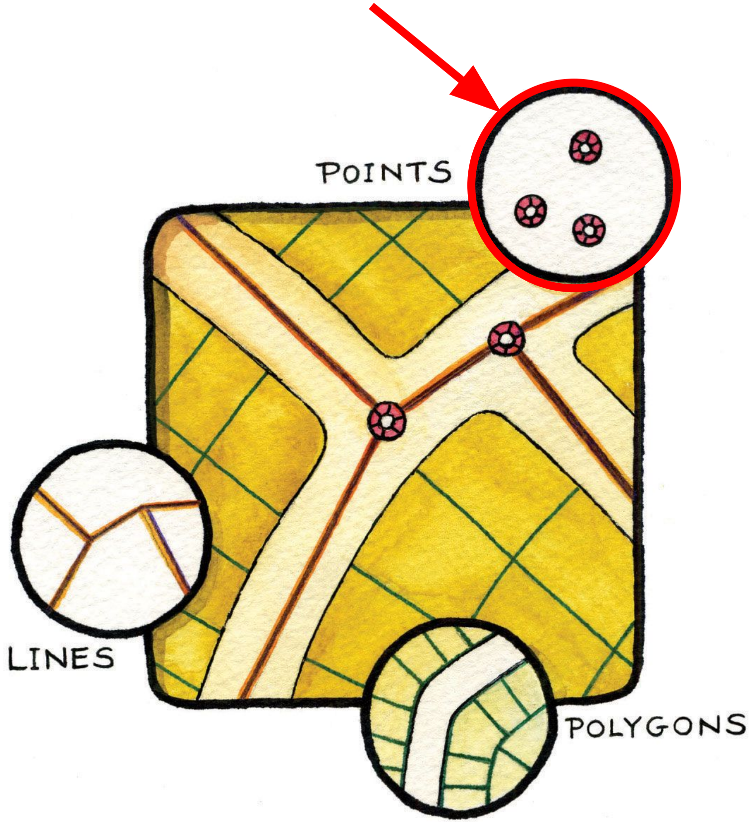
2. structuring the properties and spatial relationships of these geometric objects

3. coding and storing vector data in digital data files

Vector types

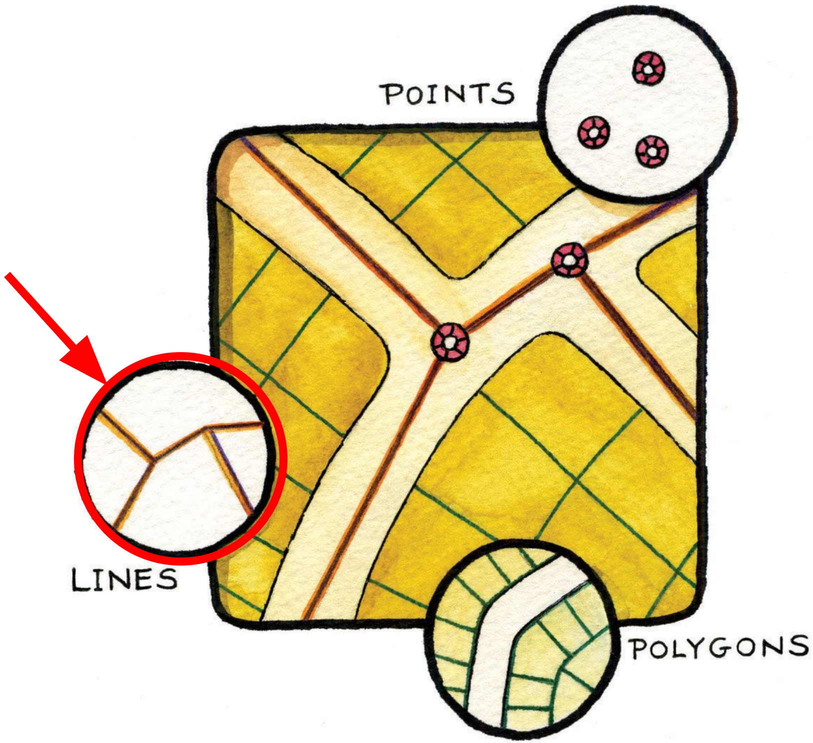


Vector types



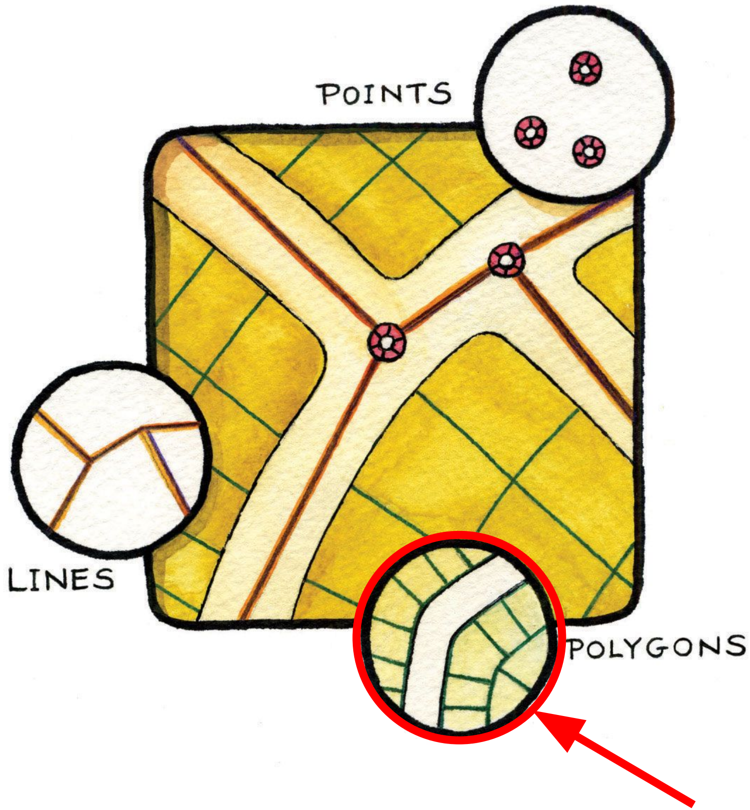
- `Point`: zero dimension
  - properties: *location* (xy coords)

Vector types

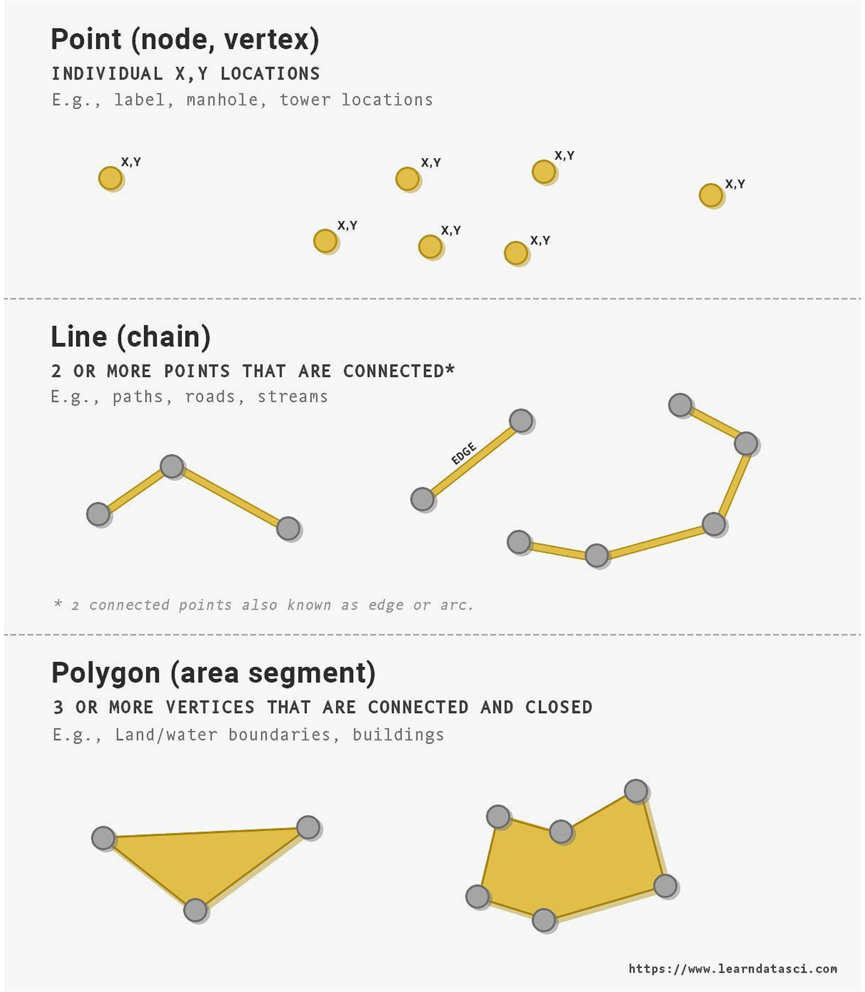


- `Point`: zero dimension
  - properties: *location* (xy coords)
- `Line`: one-dimensional
  - properties: *location* and *length*
  - has two end `Point`s
  - straight-line or curve

Vector types



- `Point`: zero dimension
  - properties: *location* (xy coords)
- `Line`: one-dimensional
  - properties: *location* and *length*
  - has two end `Point`s
  - straight-line or curve
- `Polygon`: two-dimensional
  - properties: *location*, *area*, *perimeter*
  - made of connected closed lines



- Google Maps
  - Point, Linestring, LinearRing, Polygon
- GeoJSON
  - Point, LineString, Polygon
- Shapely
  - Point, LineString, Polygon

Data Format Examples

- Google KML for visualization

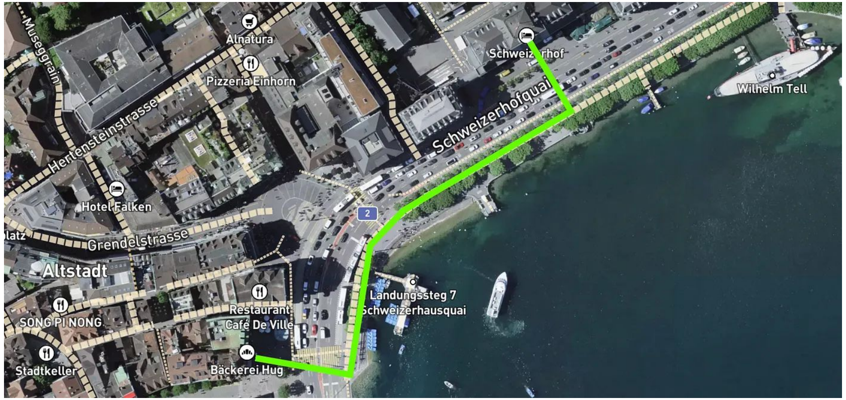
```
<?xml version="1.0" encoding="UTF-8"?>
<kml xmlns="http://www.opengis.net/kml/2.2" xmlns:gx="http://www.google.com/kml/ext/2.2"
xmlns:kml="http://www.opengis.net/kml/2.2" xmlns:atom="http://www.w3.org/2005/Atom">
<Document>

  <name>polygon.kml</name>
  <Style id="orange-5px">
    <LineStyle>
            <color>ff00aaff</color>
            <width>5</width>
    </LineStyle>
  </Style>


  <Placemark>
    <name>A polygon</name>
    <styleUrl>#orange-5px</styleUrl>

    <LineString>
            <tessellate>1</tessellate>
            <coordinates>
                8.542123809233731,47.36651432591258,0
                8.542020373307826,47.36684332453151,0
                8.544057950790664,47.36717881947375,0
                8.544133279150493,47.36684482636069,0
                8.542123809233731,47.36651432591258,0  <!--  = start point-->
            </coordinates>
    </LineString>

   </Placemark>
</Document>
</kml>
```

https://renenyffenegger.ch/notes/tools/Google-Earth/kml/index

Data Format Examples

- GeoJSON

```
{
  "coordinates":
    [
      [
        8.310242689008646,
        47.05429444841852
      ],
      [
        8.310504651721004,
        47.05399445514598
      ],
      [
        8.309440079847974,
        47.05356534791096
      ],
      [
        8.309228280208345,
        47.05341345071696
      ],
      [
        8.309100085689579,
        47.05285522481279
      ],
      [
        8.308498128818854,
        47.05292737678937
      ]
    ],
  "type": "LineString"
}
```



https://medium.com/@dmitry.sobolevsky/geojson-tutorial-for-beginners-ce810d3ff169

Data Format Examples

- Python’s Shapely library:
    - shapely.geometry.**LineString**(**[(2, 0.5), (1, 1), (-1, 0), (1, 0)]**)

- Through WKT (Well Known Text) standard format:
    - shapely.wkt.loads('**LINESTRING (0 1, 1 0, 2 0.5, 3 0, 4 0, 5 0.5, 6 -0.5, 7 -0.5, 7 1)**')

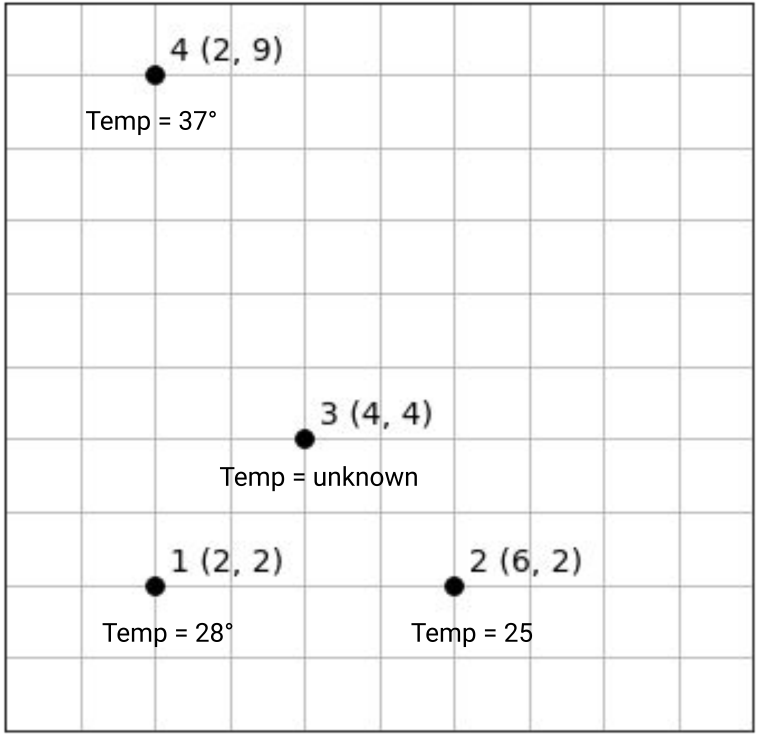
Type	Examples
Point	`POINT (30 10)`
LineString	`LINESTRING (30 10, 10 30, 40 40)`
Polygon	`POLYGON ((30 10, 40 40, 20 40, 10 20, 30 10))`
	`POLYGON ((35 10, 45 45, 15 40, 10 20, 35 10), (20 30, 35 35, 30 20, 20 30))`

https://geobgu.xyz/py-2022/shapely.html
https://en.wikipedia.org/wiki/Well-known_text_representation_of_geometry

Terminology

- Objects (points, lines, polygons, etc.) in a layer are also called **(spatial) features**
  - their presence is a “feature” of space

- The other variables (e.g. Temp in the fig.) are called **(non-spatial) attributes**
  - better not calling them just features…



Data Representation Model

How do we represent geometric objects in a computer?

- **Geo-relational** data model
  - stores geometries and attributes separately
  - associating attributes to an object requires some operations

- **Object-relational** data model
  - stores geometries and attributes together
  - attributes are part of the objects

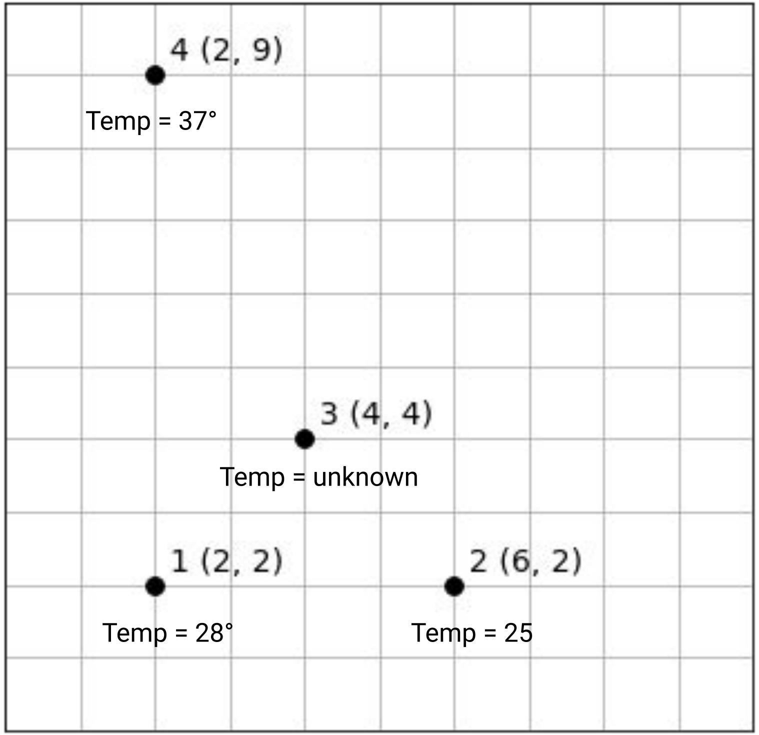
Geo-relational Model

Objects

ID	x, y
1	(2, 2)
2	(6, 2)
3	(4, 4)
4	(2, 9)

Attributes

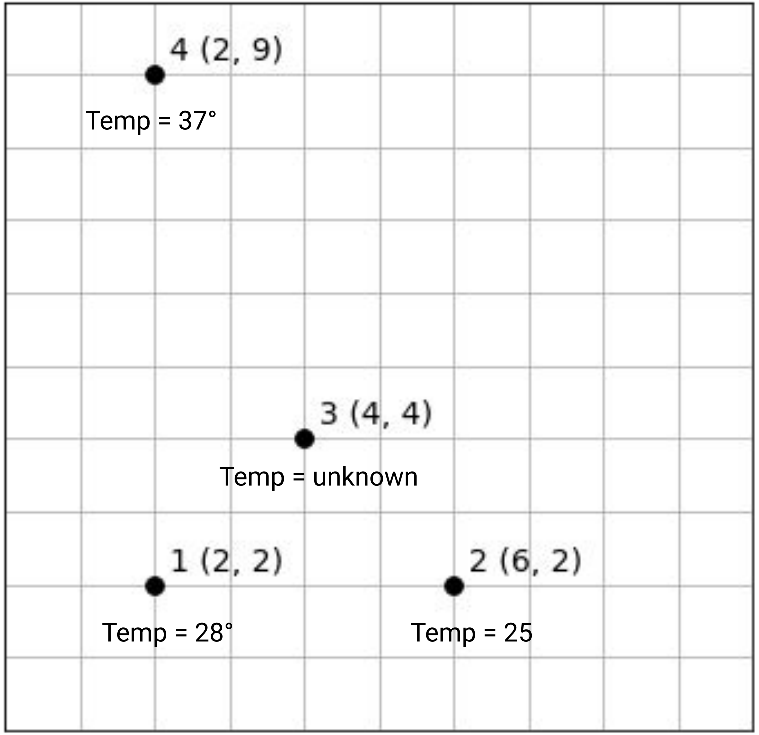
ID	Temp
1	28°
2	25°
4	37°



Object-relational Model

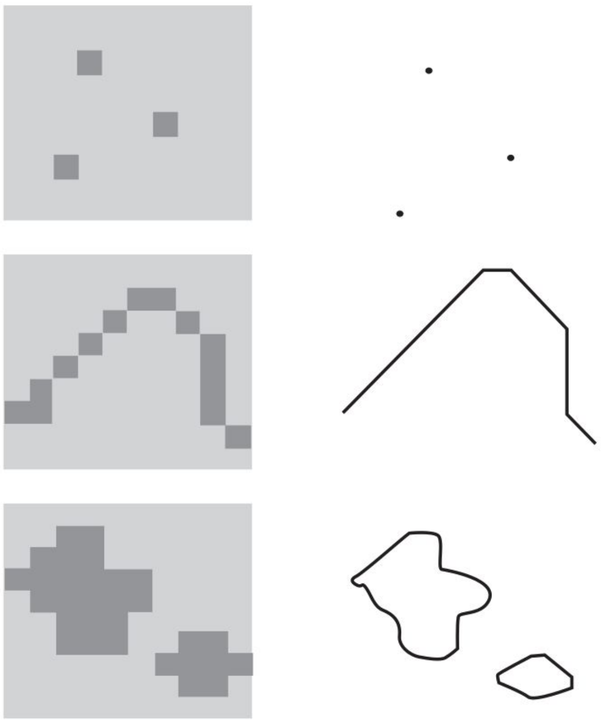
Objects

ID	x, y	Temp
1	(2, 2)	28°
2	(6, 2)	25°
3	(4, 4)	N/A
4	(2, 9)	37°



Raster or Vector?

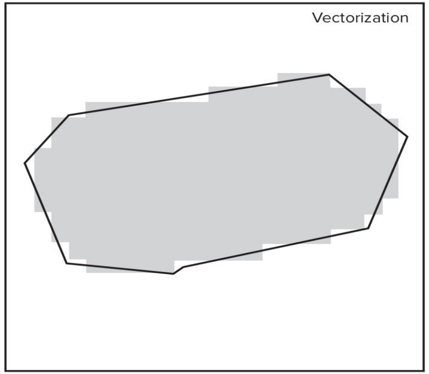
- In principle, vectors can model everything
- Yet, raster can be practical for “dense” data
    - In particular, more efficient
- What would you use for representing:
    - Points of interest (bars, cinemas, etc.) ?
    - The home location of people ?
    - The price of houses in the city ?
    - Animal ranges (areas where they move) ?



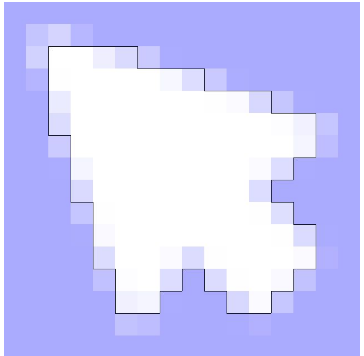
From Raster to Vector, and viceversa

- Rasterization: from vectors to raster images
  - Introduces approximations
  - Which pixels should be selected?

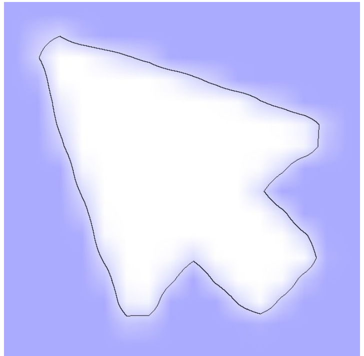
- Vectorization: from raster images to vectors
  - Can be very difficult



trivial solution
(take borders of selected pixels)



more sophisticated
(bilinear pixel interpolation)



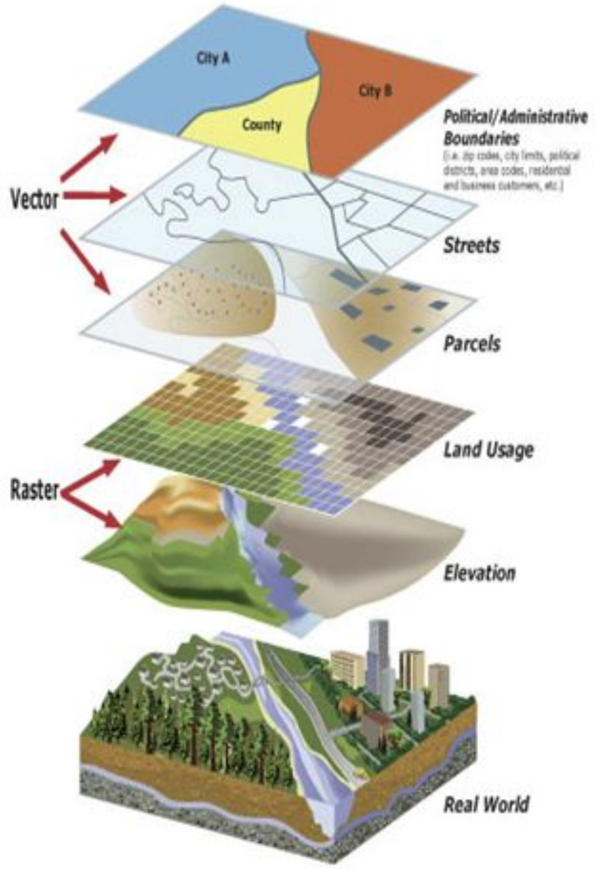
← https://wordsandbuttons.online/simple_image_vectorization.html

Vector and Raster layers used together

- Modeling reality often requires several layers

- Some objects are better modeled (or easier to find in data sources) as vector features:
  - administrative boundaries
  - street network
  - single locations

- Others are usually raster:
  - land usage
  - DEM



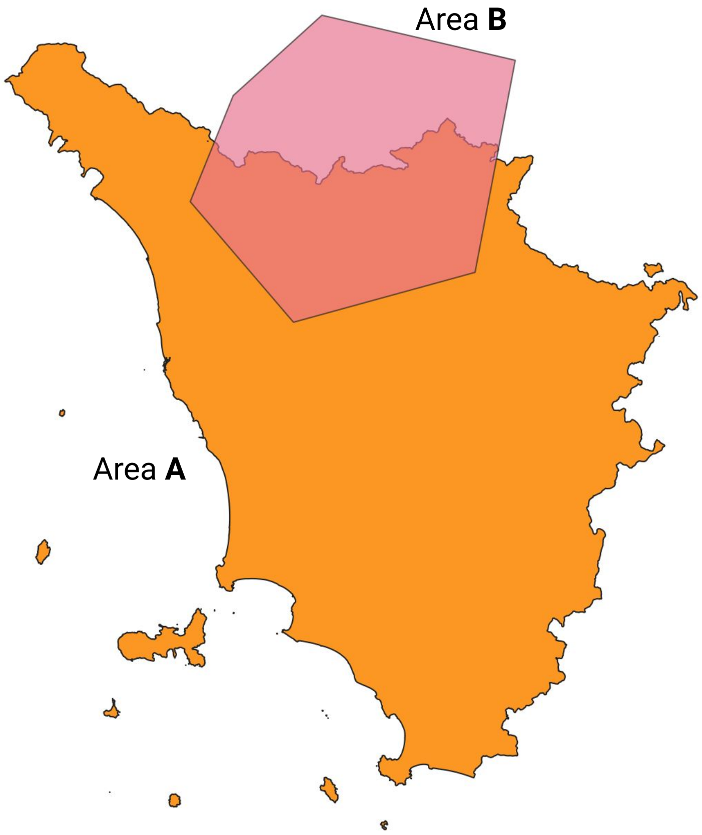
Spatial operations

intersection, union, buffering, spatial join

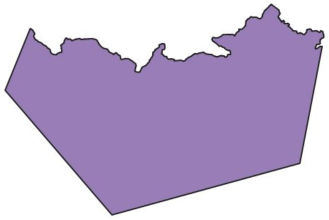
Overlay Spatial Operations

- Several operations are inherited from set theory, with the same meaning
- They belong to the family of Overlay operations, e.g.:
    - Intersection
    - Union
    - Difference

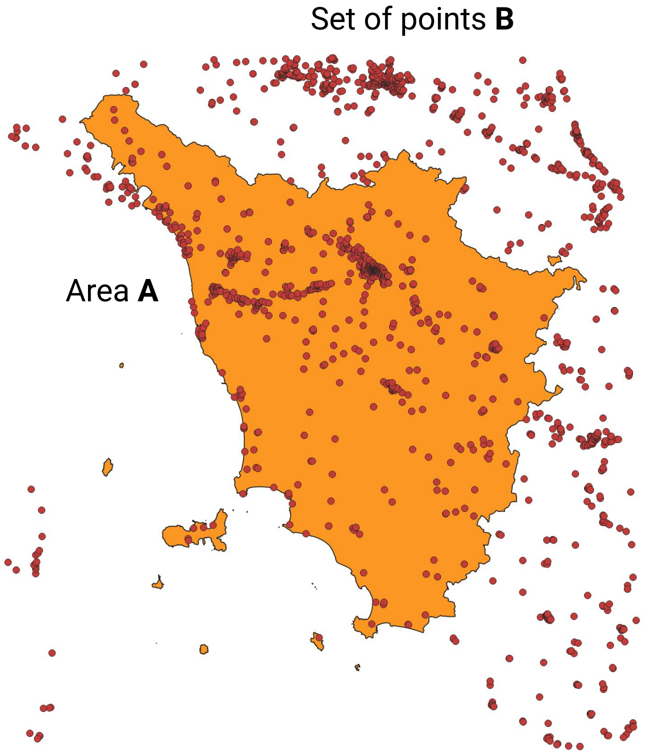
Intersection



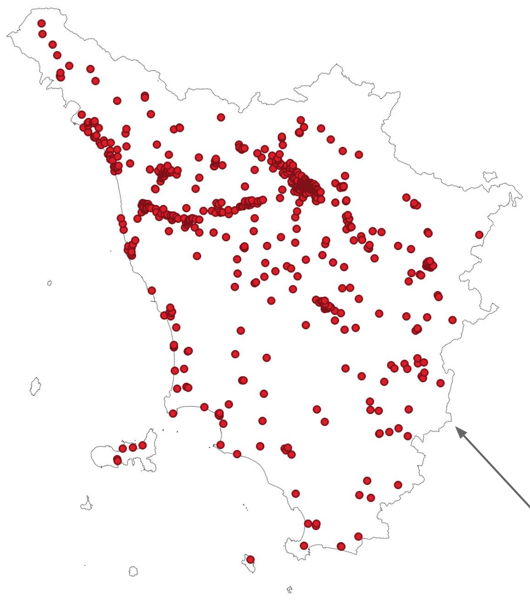
A ∩ B



Intersection/2

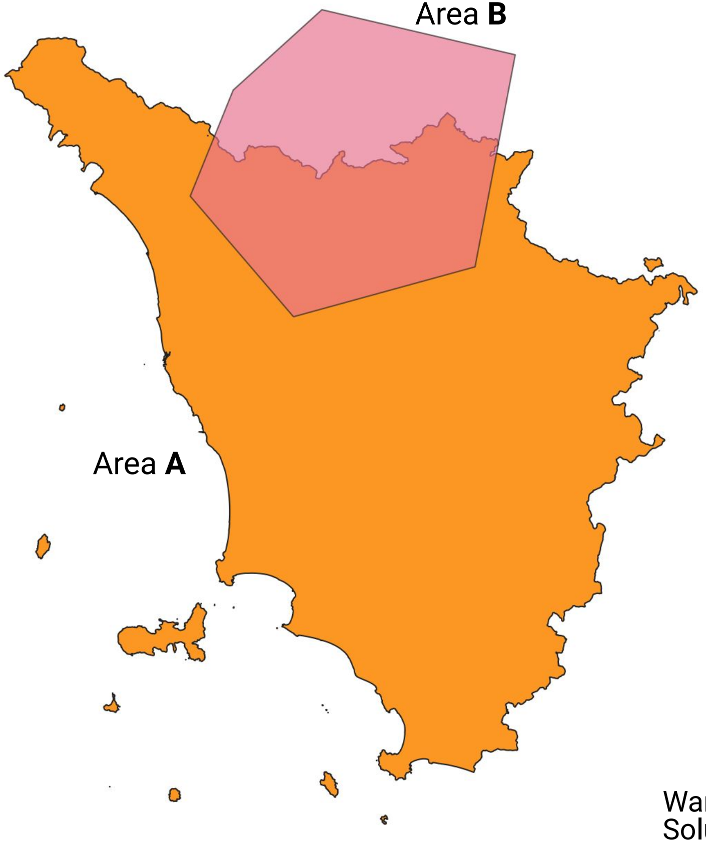


A ∩ B

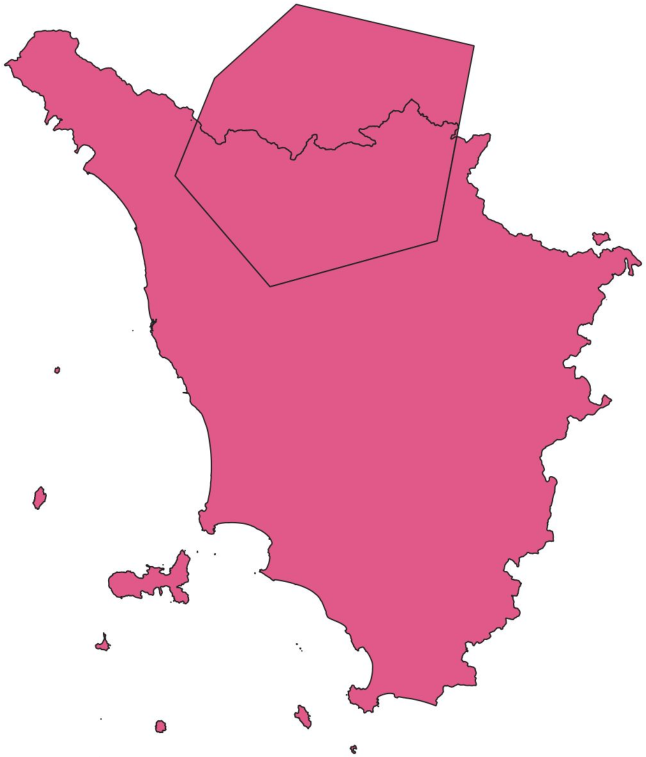


(The outline is just for readability.
It is not part of the output)

Union

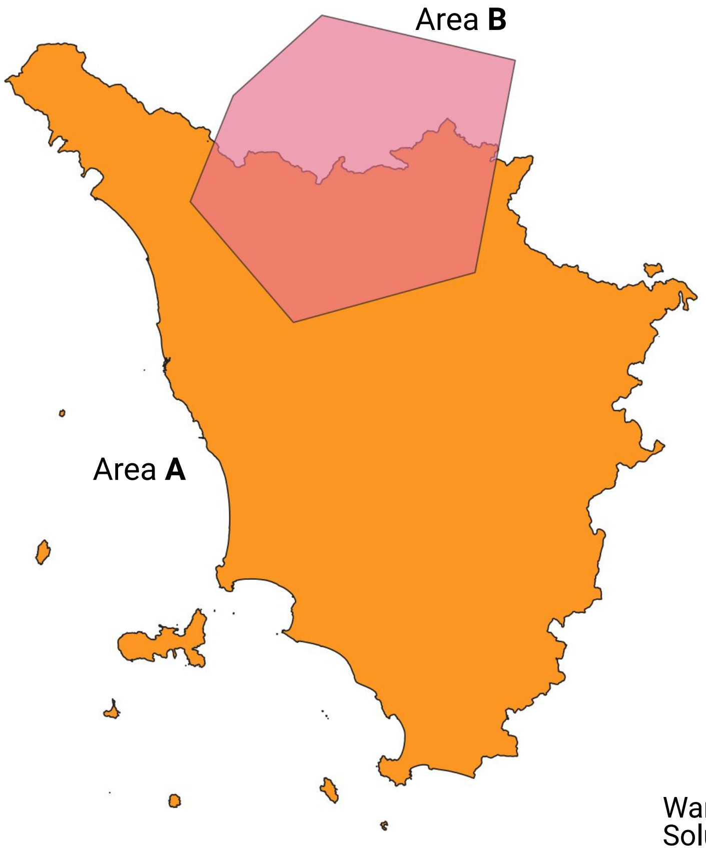


A ∪ B



Warning: some tools (e.g. QGIS) return a multipolygon. Here union just added a polygon.
Solution (if it is a problem): “dissolve” operation

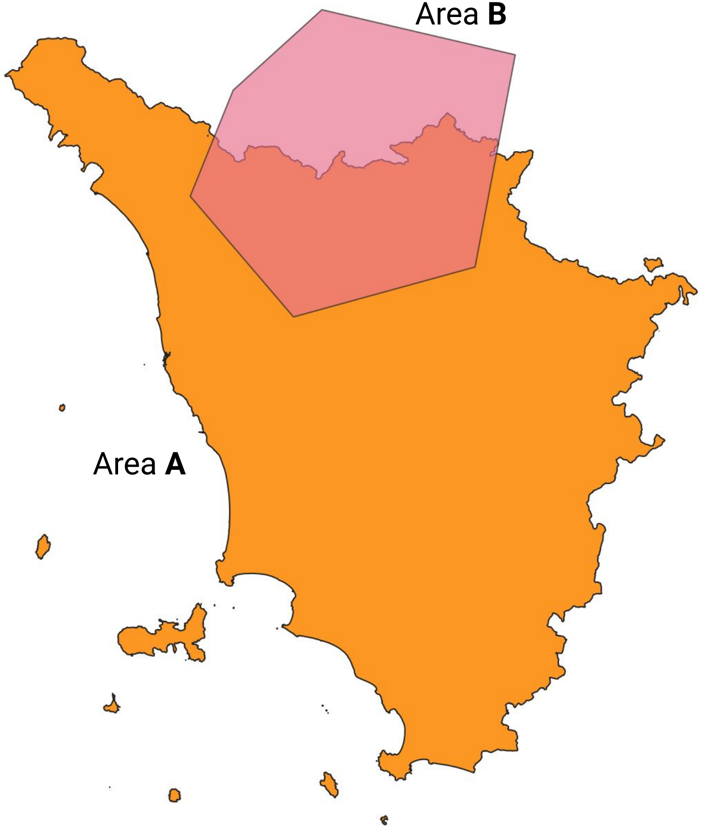
Union/2

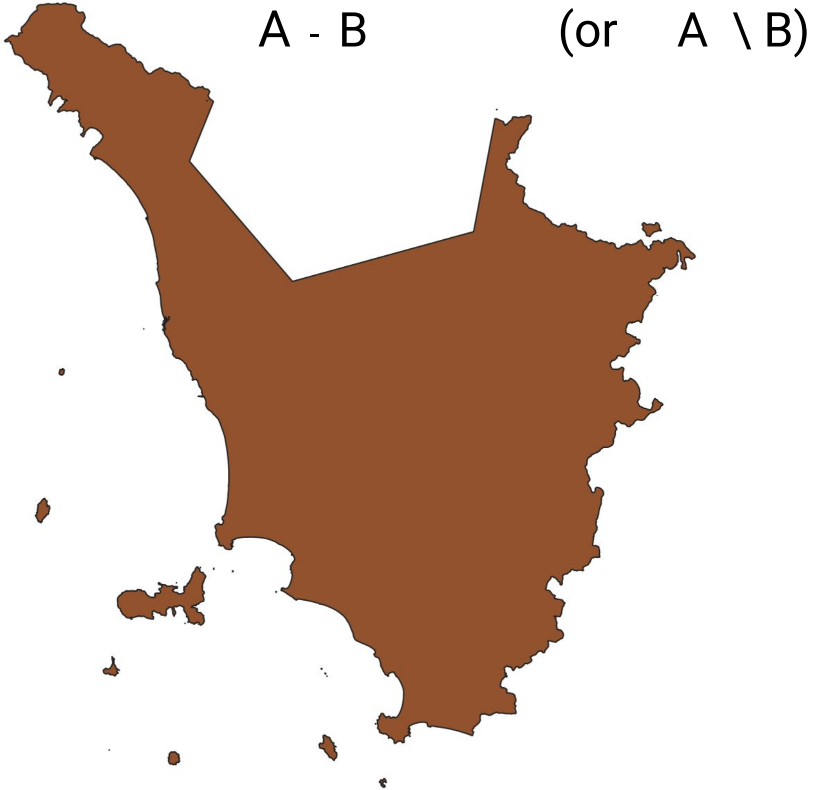


dissolve(A ∪ B)

Warning: some tools (e.g. QGIS) return a multipolygon. Here union just added a polygon.
Solution (if it is a problem): “dissolve” operation

Difference





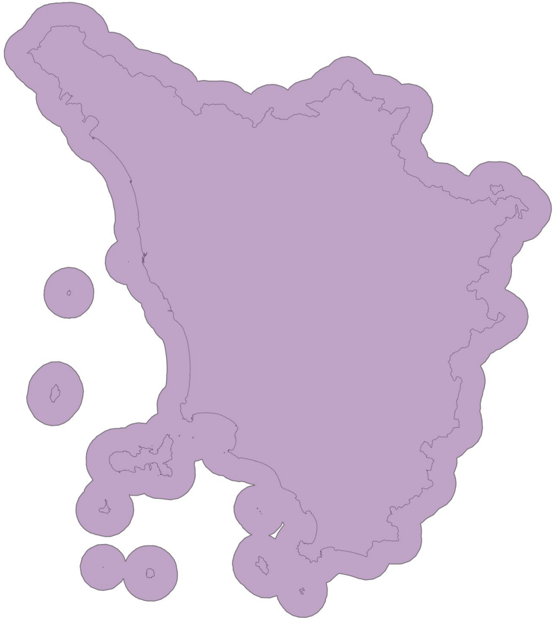
GIS-oriented Spatial Operations

- Some operations are more oriented to manipulate geometries or managing the non-spatial attributes
  - Creating buffers
  - Joining attributes of geometries
  - Overlays

Buffers

- Expand the shape by a given amount
- Equivalent to replace each point in the geometry by a circle

Buffer of 10 km

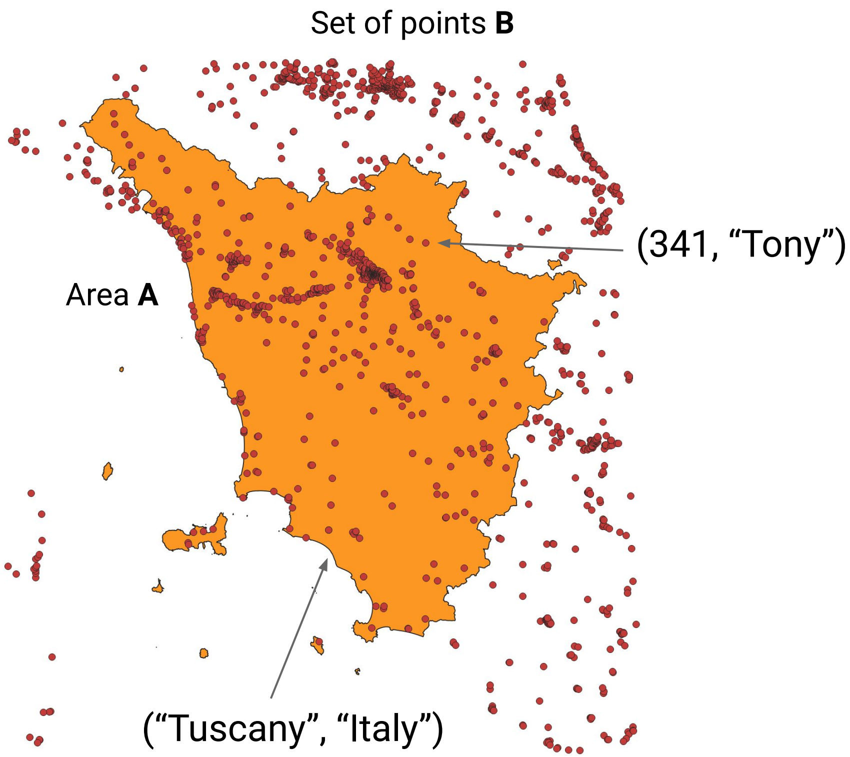


Spatial Join

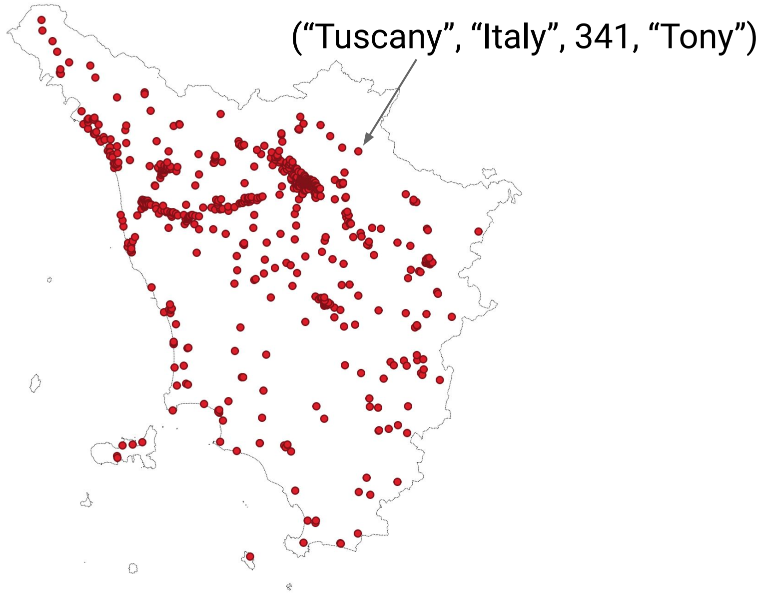
- As in databases, merge the information of two objects A and B if they “match”
    - Most common notion of “match”:
        - A **intersects** B
        - A **contains** B
        - A **=** B
        - A **touches** B (they are neighbors, though not intersecting)

- Using database terms, the join can be
    - **inner**: the output contains only pairs that match
    - **outer** / left / right: the output contains all non-matching objects
        - in this case the attributes added by the join are Null

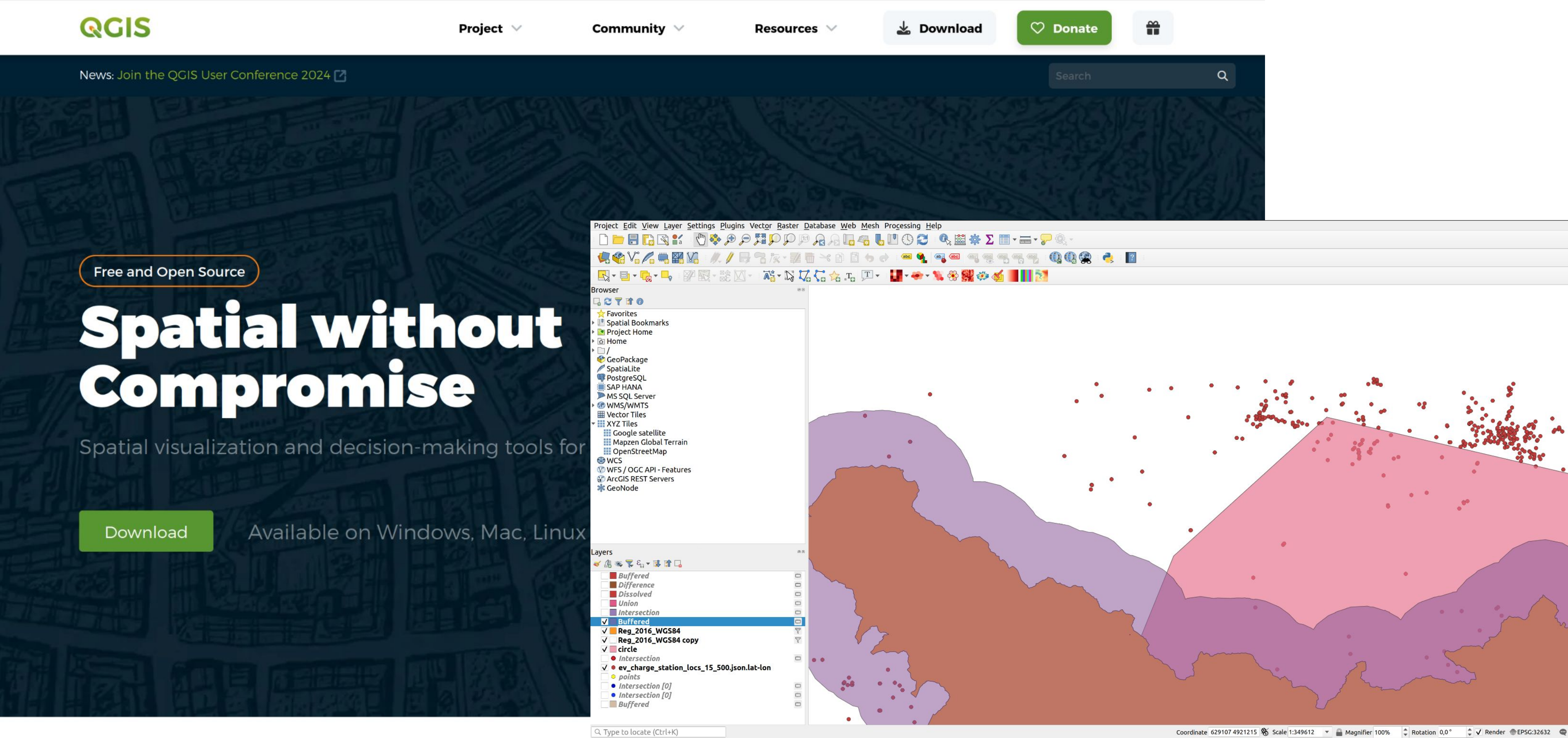
Spatial Join



Inner intersect join(A, B)



Suggested “point-n-click” software



Point spatial patterns and spatial correlation

density, NNs, Moran’s I & Geary’s C, etc.

Point spatial patterns

- General objective: understanding how objects are distributed in space
  - Not interested (yet) in non-spatial attributes

- In spatial point pattern analysis, spatial distribution patterns are typically categorized into three types:
  - uniform (discrete) distribution
  - random distribution
  - clustered distribution

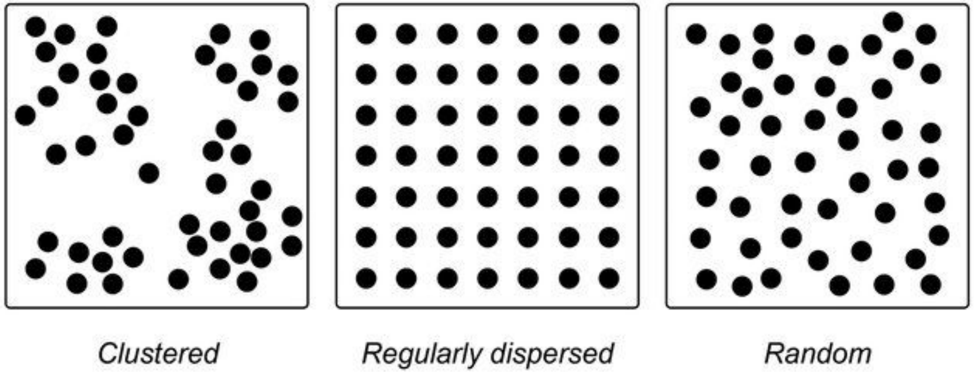
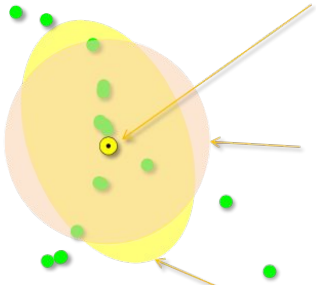


Image: http://dx.doi.org/10.4312/dp.47.27

Center and dispersion measures

- Exactly as in standard statistics, we can define
  - a center around which all objects are distributed
  - various dispersion indexes to measure how much dispersed around they are



Mean center is the computed average X and Y coordinate values.

$$\bar{s} = \left(\frac{\sum_{i=1}^{n} x_i}{n}, \frac{\sum_{i=1}^{n} y_i}{n} \right)$$

Standard distance is a measure of the variance between the average distance of the features to the mean center.

$$d = \sqrt{\frac{\sum_{i=1}^{n} (x_i - \mu_x)^2 + (y_i - \mu_y)^2}{n}}$$

Standard deviational ellipse computes separate standard distances for each axis.

$$\begin{cases} d_x = \sqrt{\frac{\sum_{i=1}^{n} (x_i - \mu_x)^2}{n}} \\ d_y = \sqrt{\frac{\sum_{i=1}^{n} (y_i - \mu_y)^2}{n}} \end{cases}$$

https://mgimond.github.io/Spatial/

Density estimation

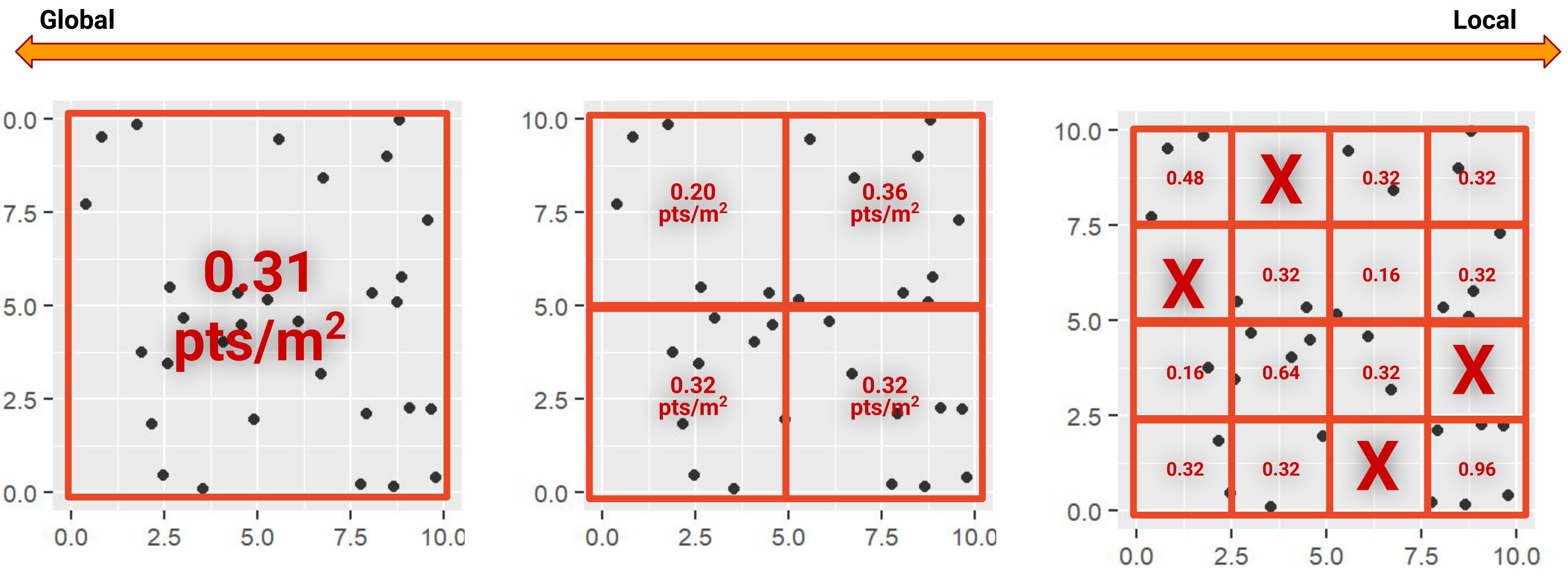
- Simple question: how many points (or objects) are in the same place?

- Issue: “same place” can have different meanings

- We will explore three of them:
  - **Global density**: computed over all the geographical area
    - E.g. number of restaurants per m^2
  - **Local density**: computed separately over the small cells of a tesselation
    - E.g. restaurants per m^2 for each municipality
  - **Kernel density**: the density of one cells is computed considering its neighborhood
    - E.g. restaurants in and around each municipality

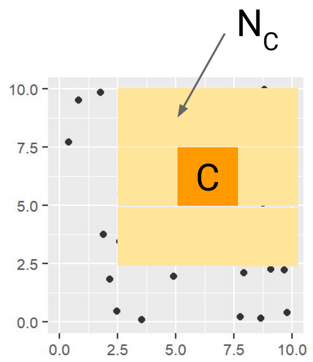
Global vs. Local density

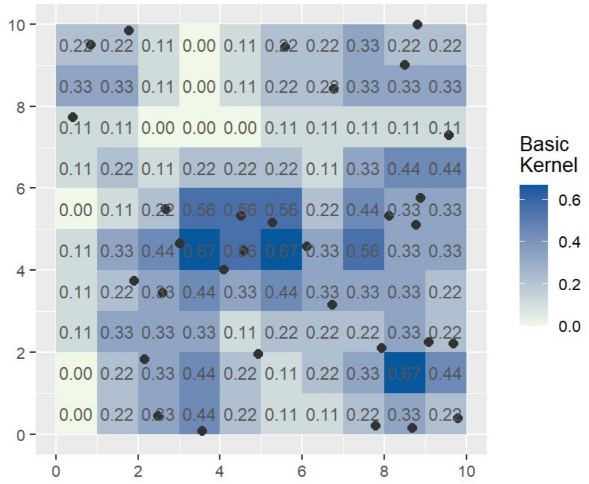
- N.points: 31
- 10m x 10m area



Kernel density

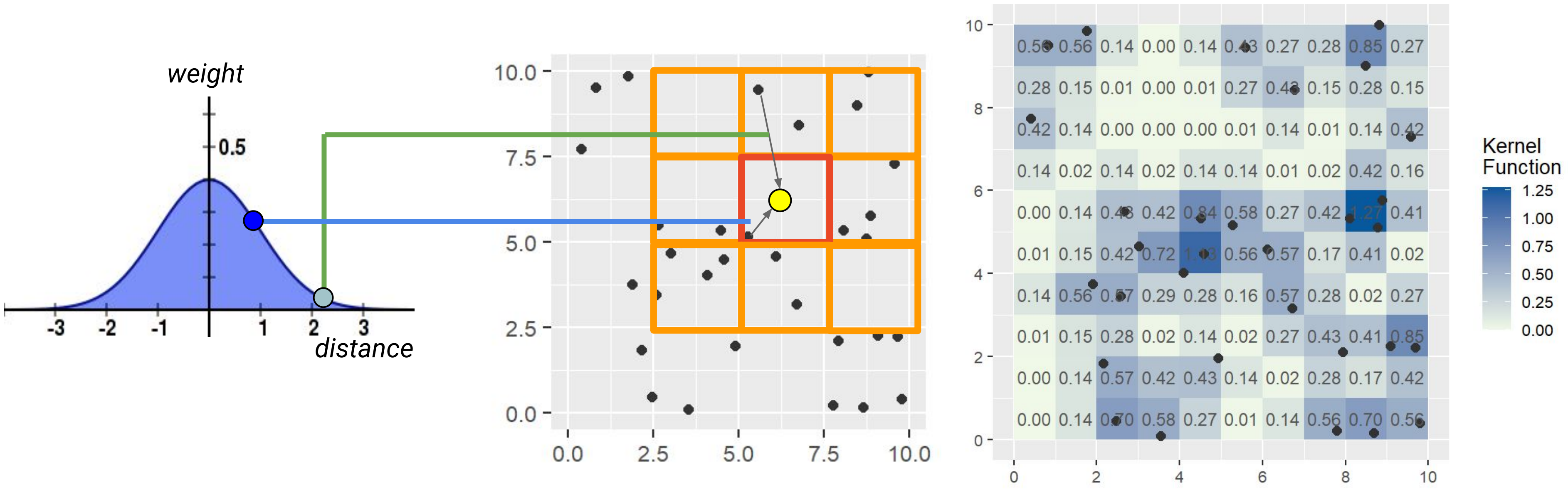
- Define a neighborhood N_C for each cell C
    - Typically, the 8 adjacent cells
    - Density of C = density of { C } U N_C
        - Smoothing effect similar to “moving average” in time series





Weighted Kernel density

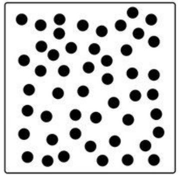
- Points in the neighborhood have a weight dependent on the distance from C’s center
- E.g. Gaussian function of distance: G(d)



Random vs. Pattern / 1

- **Average Nearest Neighbor (ANN)** analysis
  - Associate each point to its nearest neighbor distance (d_i)
  - Compute average d_i values (d_{obs})
  - Normalize w.r.t. expected d_{obs} over random points (d_{exp}):

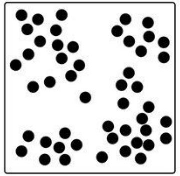
$$R = \frac{d_{\text{obs}}}{d_{\text{exp}}}$$



Random

R = 1

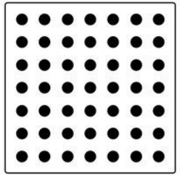
By definition, since we expect all random cases yield the same d_{obs}



Clustered

R < 1

Most points have a few very close neighbors, thus small d_{obs}



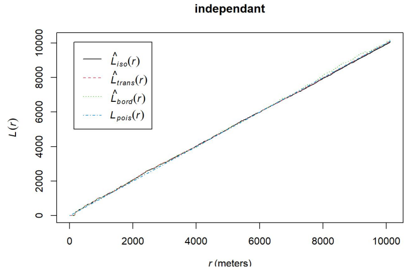
Regularly dispersed

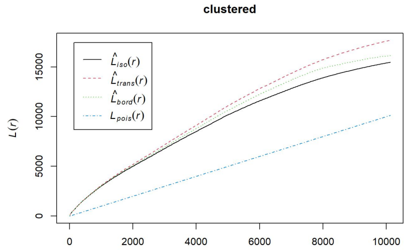
R > 1

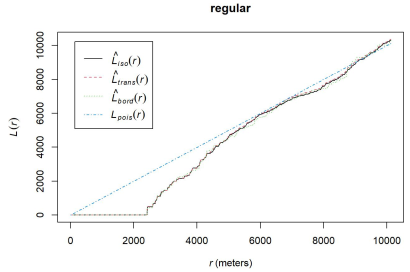
Dispersion maximizes distances, yielding larger d_{obs}

Random vs. Pattern / 2

- **L function** (a.k.a. standardized Ripley's K-function)
  Given **N** points in an area of size **A** and a distance parameter **d**:
  - Compute all N(N-1) distances between each pair of points
  - Compute the fraction φ of distances that are < d
  - Compute: $$L(d) = \sqrt{\frac{A}{\pi}\phi}$$
- Property: L(d)=d for random points
- Exploring L(d) for various d values allows understanding patterns at different spatial granularities







Spatial Autocorrelation

- Measures to assess the relations / dependencies between a non-spatial attribute of objects and their spatial location
  - E.g. how much is the temperature in one place influenced by temperature around it?

- **Autocorrelation**: correlation between values of the same variable (e.g. temperature) measured in different times or places
  - different times → time series → temporal autocorrelation
    - Values at each time t are correlated with value at $t + \delta$ (lagged correlation)
  - different places → geospatial data → spatial autocorrelation
    - Values at each location p are correlated with values in p’s neighborhood

- **Tobler’s first law of geography**: “everything is related to everything else, but near things are more related than distant things.”

Moran's I

- Autocorrelation between values of each point against all other points in its neighborhood

$$I = \frac{\sum_{i=1}^{n}\sum_{j=1}^{m} w_{ij}\left(x_i - \bar{x}\right)\left(x_j - \bar{x}\right)}{s^2 \sum_{i=1}^{n}\sum_{j=1}^{m} w_{ij}}$$

  - n = number of points in the dataset
  - m = number of neighbors (assuming it is constant)
  - s^2 = variance of values
  - w_{ij} = strength of dependency between points “i” and “j”
    - various ways to define it
    - most common: w_{ij} = 1/distance(i,j)

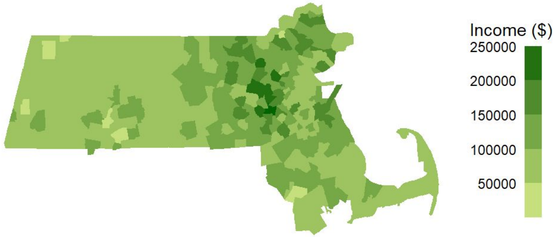
- I > 0 means positive correlation; I < 0 means negative correlation (w.r.t. nearby values)

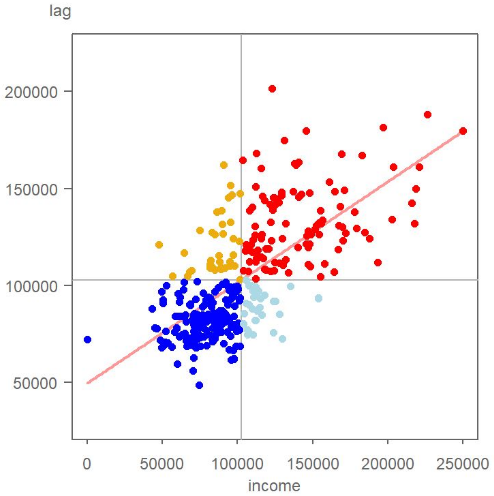
Moran's I

- Alternative reading: Moran's I = average of several "Local Moran's I"s:

$$\mathrm{I} = \frac{\sum_{i=1}^{n} \boxed{\sum_{j=1}^{m} w_{ij}} (x_i - \bar{x}) \boxed{(x_j - \bar{x})}}{s^2 \sum_{i=1}^{n} \sum_{j=1}^{m} w_{ij}}$$

- We can explore Local I values in a "spatial lag plot" (ref.: lag plots in time series)
  - lag = average value around the reference point





Geary's C

- Similar in concept to Moran's I:

$$C = \frac{(N-1)\sum_i \sum_j w_{ij}(x_i - x_j)^2}{2S_0 \sum_i (x_i - \bar{x})^2}$$

$S_0 = \sum_i \sum_j w_{ij}$

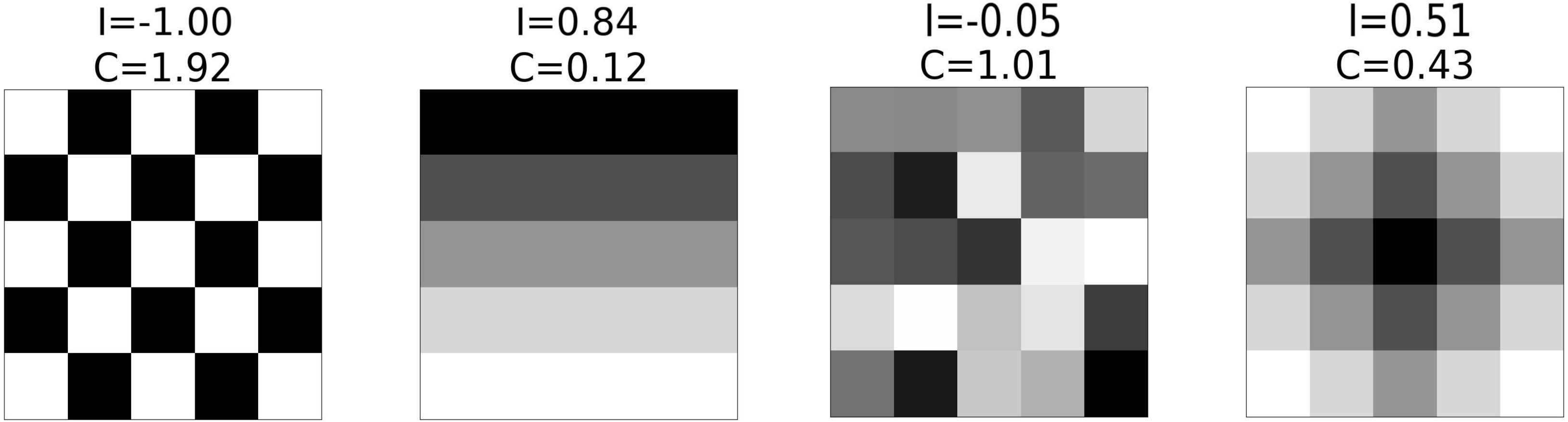
  - though it measures (almost) the opposite
  - the higher is C, the more different are nearby values

- It can be seen that Geary's C is less sensitive to linear associations

- Also here, a Local C can be explored

Geary's C

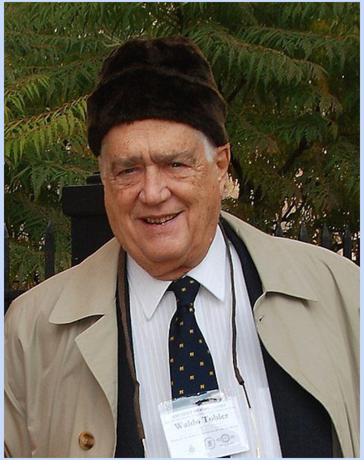
- A comparison:



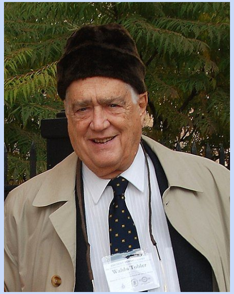
INTERVALLO

Waldo Rudolph Tobler

(1930-2018)



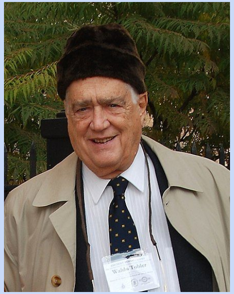
- Geographer and cartographer
- Father of Tobler's First Law of Geography
    - A top citation in geospatial studies (and often abused…)



INTERVALLO

Great contributions to many areas:

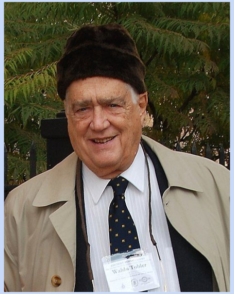
- analytical cartography
- early dev. of Geographic information systems (GIS)
- lay the groundwork for GIScience
- computer cartography
    - one of the first to use computers in geography
- map projections, choropleth maps, flow maps, cartograms, animated mapping
- mathematical modeling of geographic phenomena



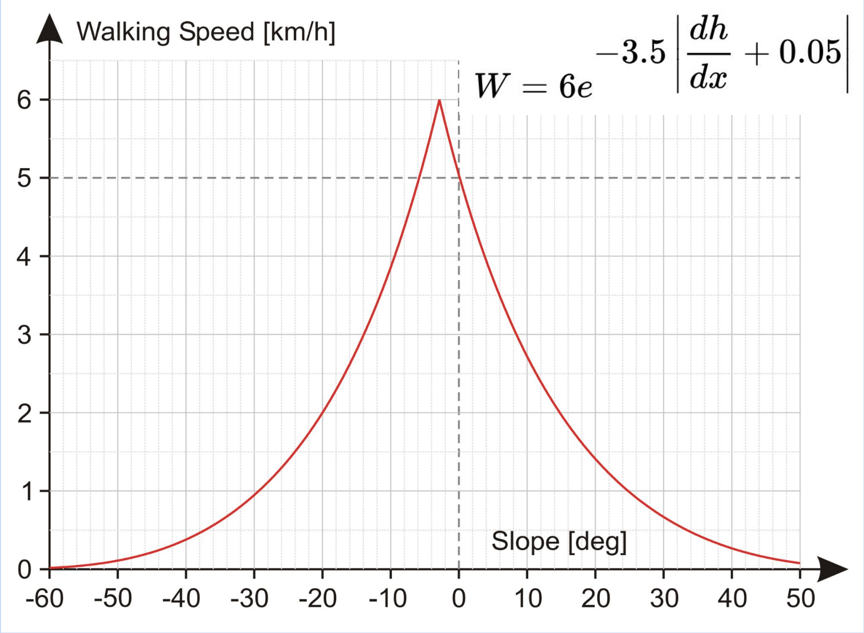
INTERVALLO

- W. R. Tobler. “A Computer Movie Simulating Urban Growth in the Detroit Region”. Economic Geography, 1970.

INTERVALLO



- Tobler's hiking function
  - models an average hiker walking speed on slopes
  - Tobler collected data himself
  - Published results in a 1993 paper

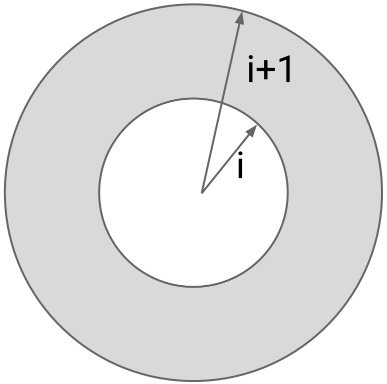


Food for thought

- The mobility of a vehicle is recorded through GPS. Should it be represented as a (set) of points or as lines (LineString or similar)?

- Let assume that a mobile phone app can "see" all the antennas that are within 1km from it, and that we have a list of all existing antennas with their location (lon, lat). Can the app infer the location of the phone? Can it do that using the vector operations seen in previous slides?

- In Moran's I the neighborhood is a parameter. Let define I_i = Moran's I with neighborhood equal to a ring of internal radius "i" km and external radius "i+1" km, and set all weights w_{ij} = 1. Compute I_i for several "i" values. How does I_i varies when "i" increases?



to study for the exam

Material

- [book chapter] Introduction to geographic information systems, Kang-Tsung Chang, McGraw-Hill
  - Sections 3.1, 3.3 (no subsections)
  - Sections 4.1, 4.2, 4.3, 4.7: Raster Data Model
  - Section 8.5: Spatial Join
  - Chapter 11: Vector Data Analysis
- [book chapter] Intro to GIS and Spatial Analysis, Manuel Gimond, online: https://mgimond.github.io/Spatial
  - Chapter 11: Pattern Analysis
  - Chapter 13: Spatial Autocorrelation
- [book section] Encyclopedia of GIS: Geary's C, Xiaobo Zhou & Henry Lin, online: https://doi.org/10.1007/978-0-387-35973-1_446