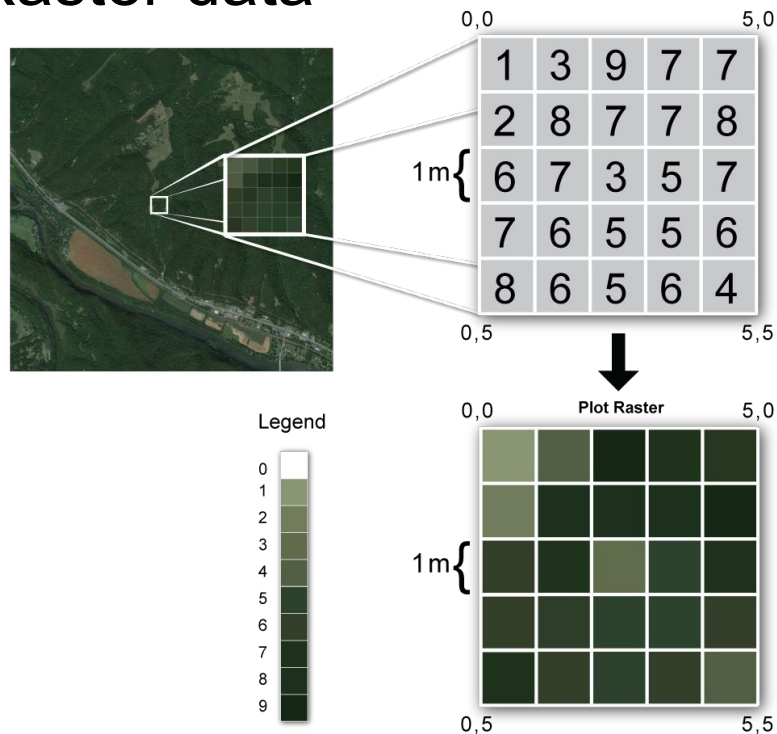


Geospatial data analysis: raster data and xarray

HAS Tools

Geographic Information Systems (GIS) have 2 main data representations

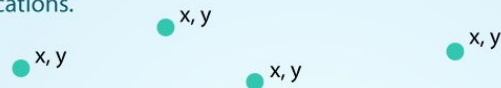
Raster data



Vector data

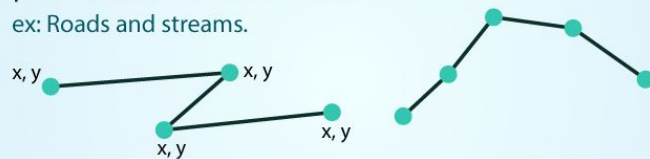
POINTS: Individual x, y locations.

ex: Center point of plot locations, tower locations, sampling locations.



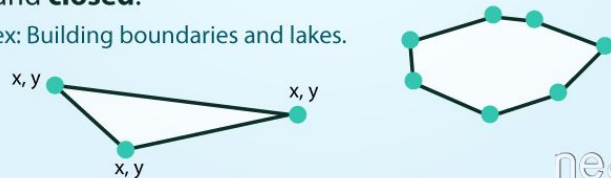
LINES: Composed of many (at least 2) vertices, or points, that are connected.

ex: Roads and streams.



POLYGONS: 3 or more vertices that are connected and **closed**.

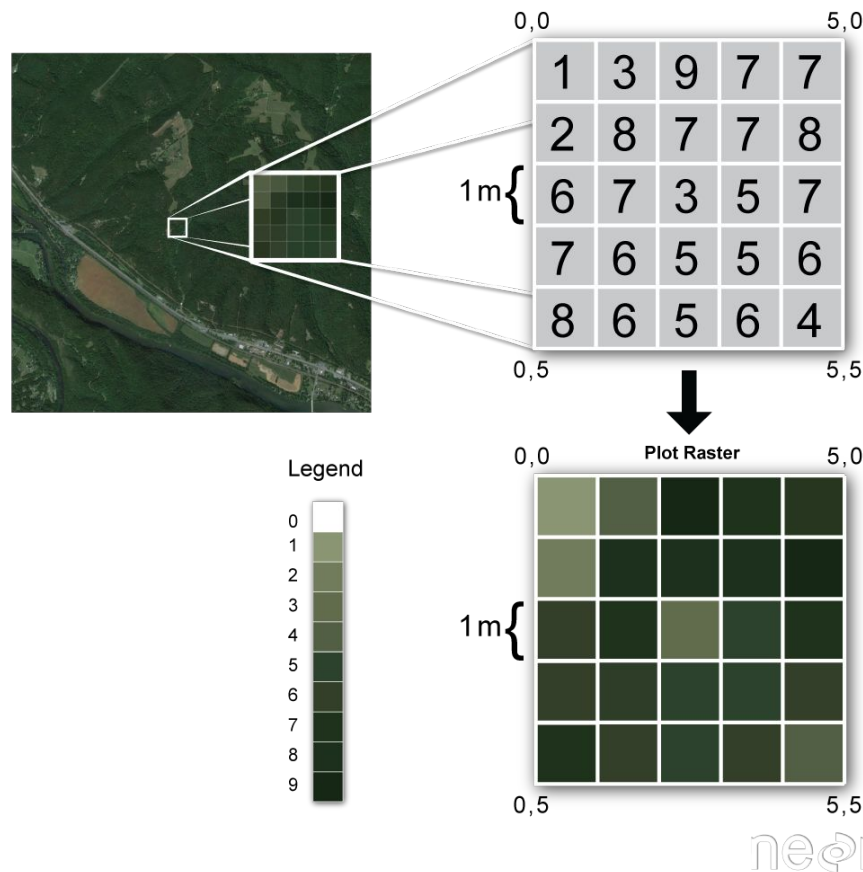
ex: Building boundaries and lakes.



neon

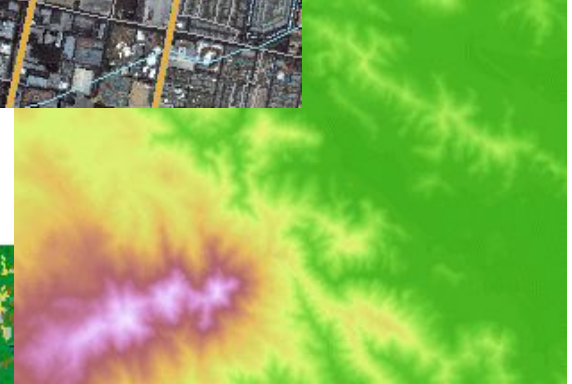
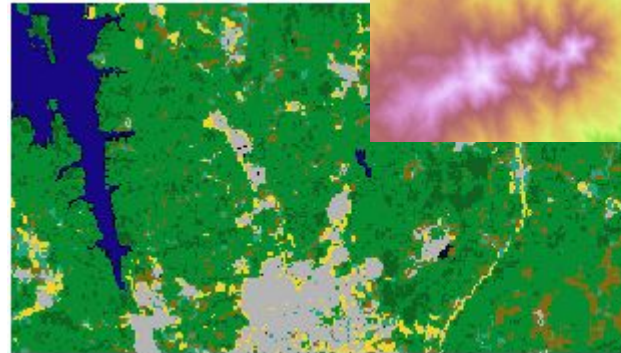
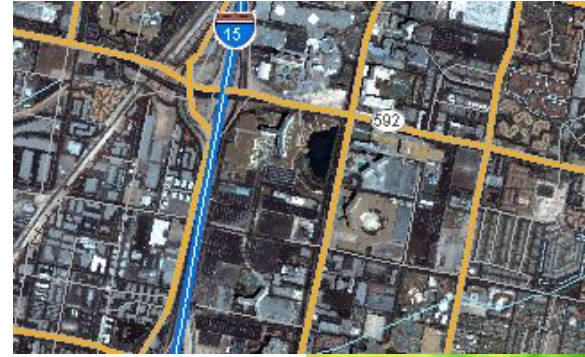
Raster data

- Raster are simply arrays or matrices of data where each data cell is approximately the same size/shape (in a given CRS)



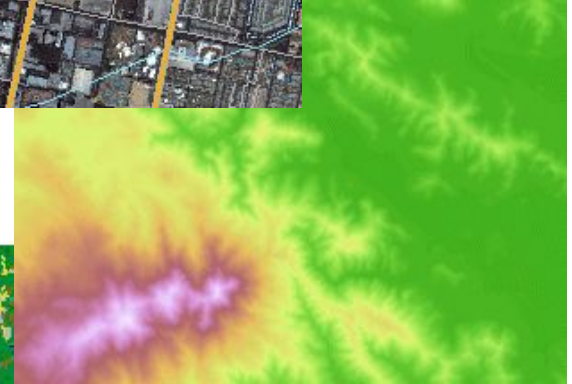
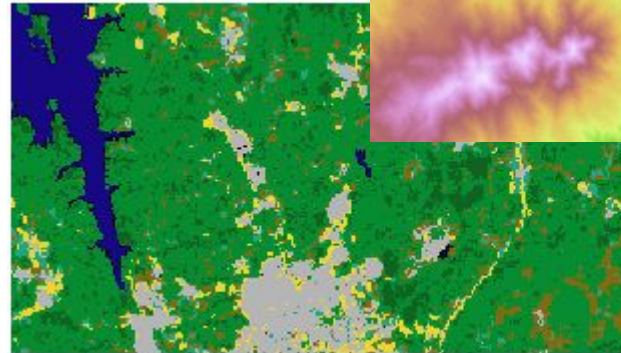
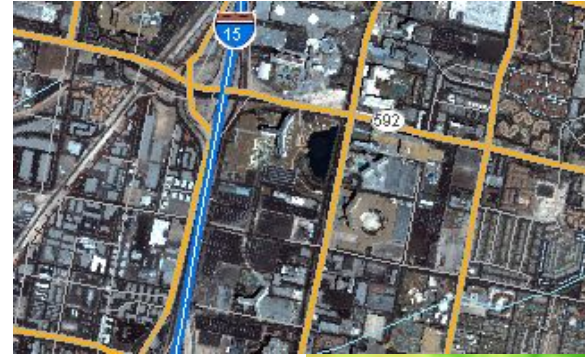
Raster data

- Raster are simply arrays or matrices of data where each data cell is approximately the same size/shape (in a given CRS)
- Some examples include aerial and satellite data, elevation maps, and soil classification maps



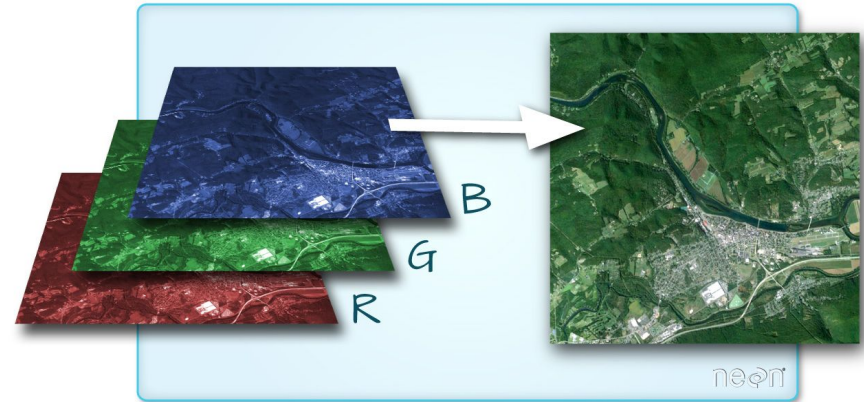
Raster data

- Raster are simply arrays or matrices of data where each data cell is approximately the same size/shape (in a given CRS)
- Some examples include aerial and satellite data, elevation maps, and soil classification maps
- Data in raster cells/pixels can be continuous, discrete, or categorical



Raster data formats

- There are many formats for that can store raster data:
 - TIFF, GRIB, BIL, NetCDF, zarr
- Raster data formats contain metadata along with the underlying array data, which includes things like the extent, resolution, CRS, etc
- Rasters may contain multiple “bands” or “variables” which can be used in multiple ways



Working with raster data in python

- As mentioned in the vector data lecture, there are bindings to GDAL in python:

<https://pypi.org/project/osgeo/>

- More modern+pythonic interface via rasterio:

<https://rasterio.readthedocs.io/en/latest/>

- Less geospatial emphasis:

<https://docs.xarray.dev/en/stable/>

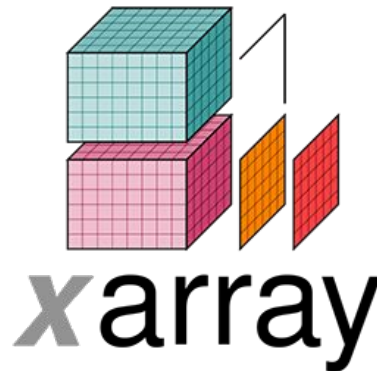
- Combining xarray and rasterio:

<https://github.com/corteva/rioxarray>

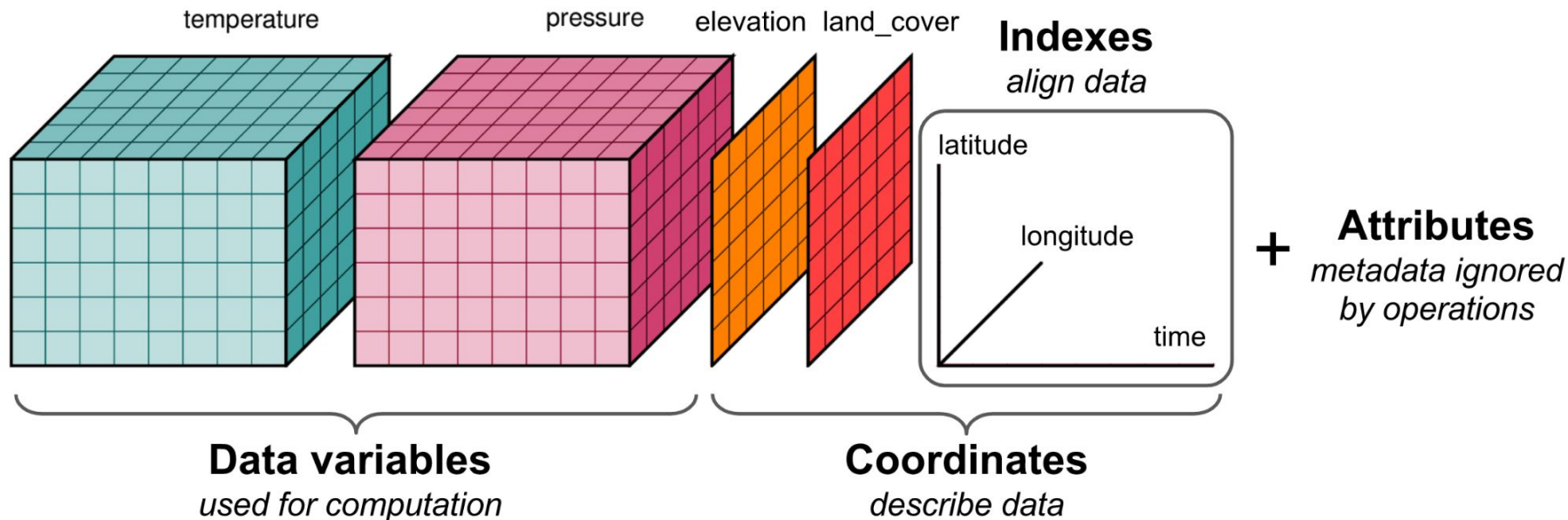


rasterio/rasterio

Rasterio reads and writes geospatial raster datasets



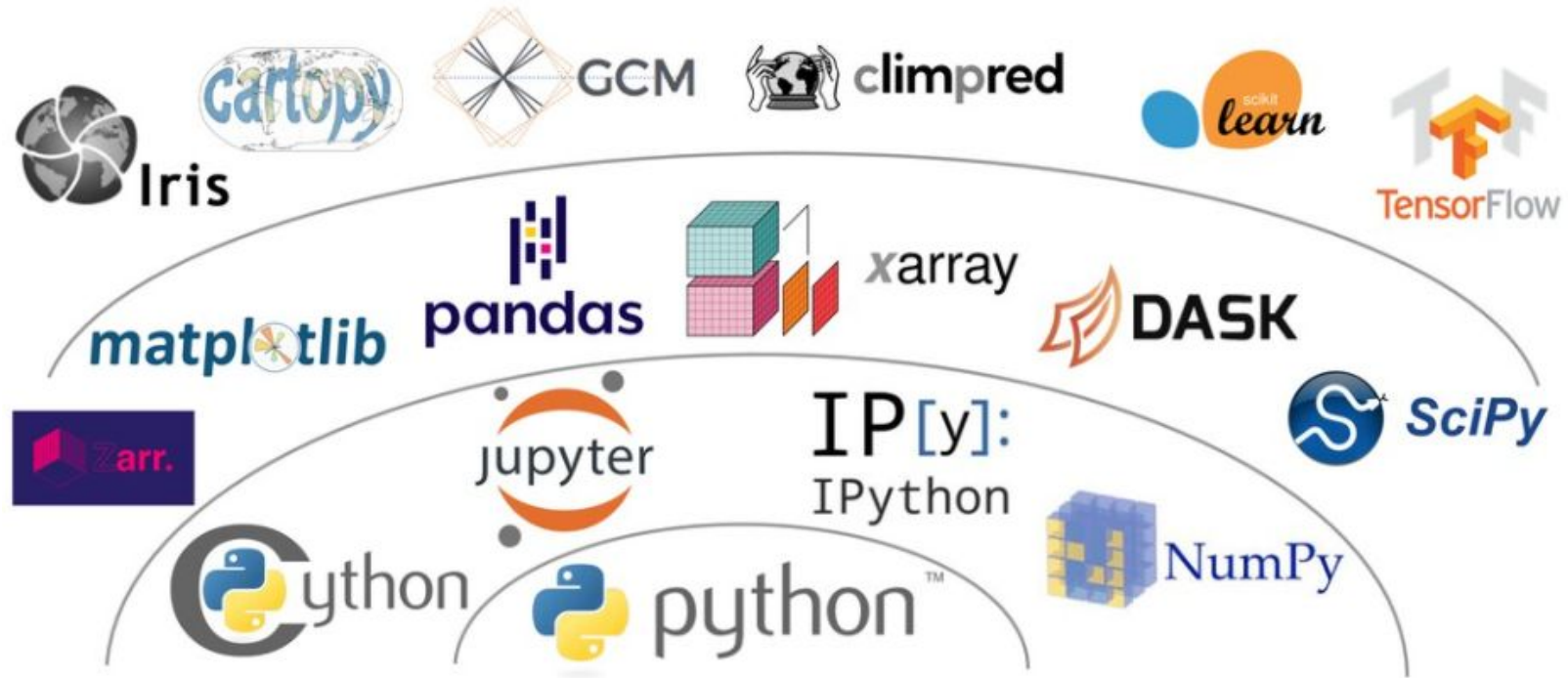
Moving away from *strictly* raster data: The NetCDF/xarray data model



Why do we want labeled dimensions and coordinates?

- Apply operations over dimensions by name:
 - Xarray: `x.sum(dim='time')`
 - Numpy: `np.sum(x, dim=2)` # if xdims (lat, lon, time)
- Select values by label rather than integer index:
 - Xarray: `x.sel(time='2014-01-01')`
 - Numpy: `x[np.where(time_arr == '2014-01-01', drop=True)]`
- Easily apply groupby and rolling calculations:
 - Xarray: `x.groupby(x['time'].year).mean()`
 - Numpy: # no single line solution, requires loops!
- Mainly you can think of xarray as an extension of pandas to N-dimensional arrays

A brief aside: xarray works really well in the larger “Pangeo” ecosystem



Inspiration: Stephan Hoyer, Jake Vanderplas (SciPy 2015)

Link to the interactive xarray tutorial

<https://mybinder.org/v2/gh/xarray-contrib/xarray-tutorial/HEAD?labpath=workshops/scipy2022/index.ipynb>