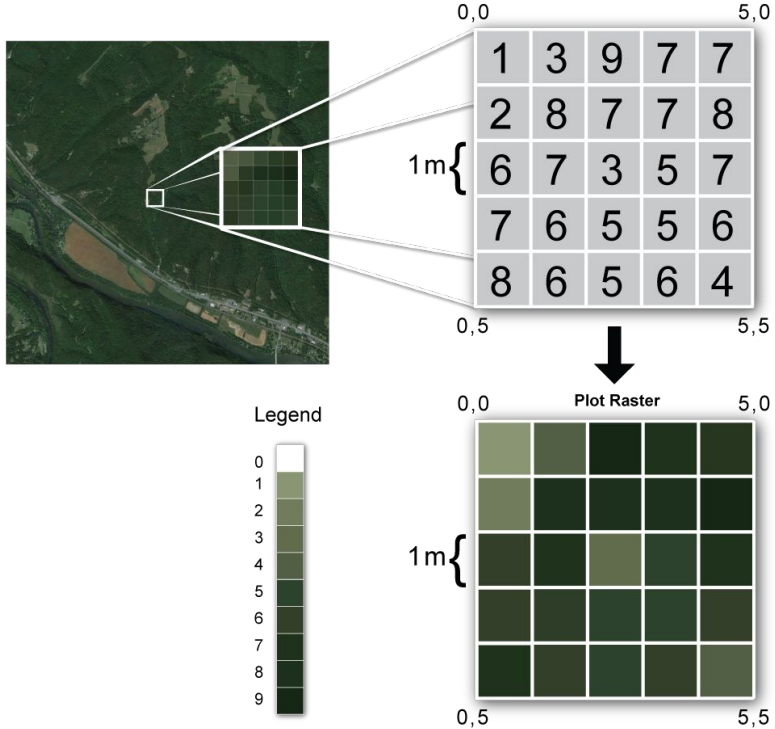
HAS Tools:
raster data & labeled arrays

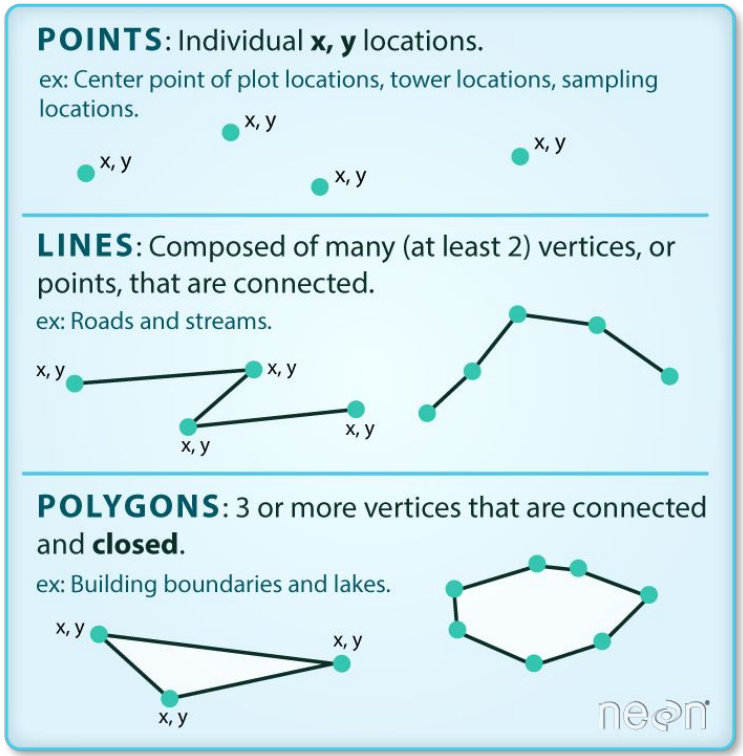
October 14, 2024

Geographic Information Systems (GIS) have 2 main data representations

Raster data



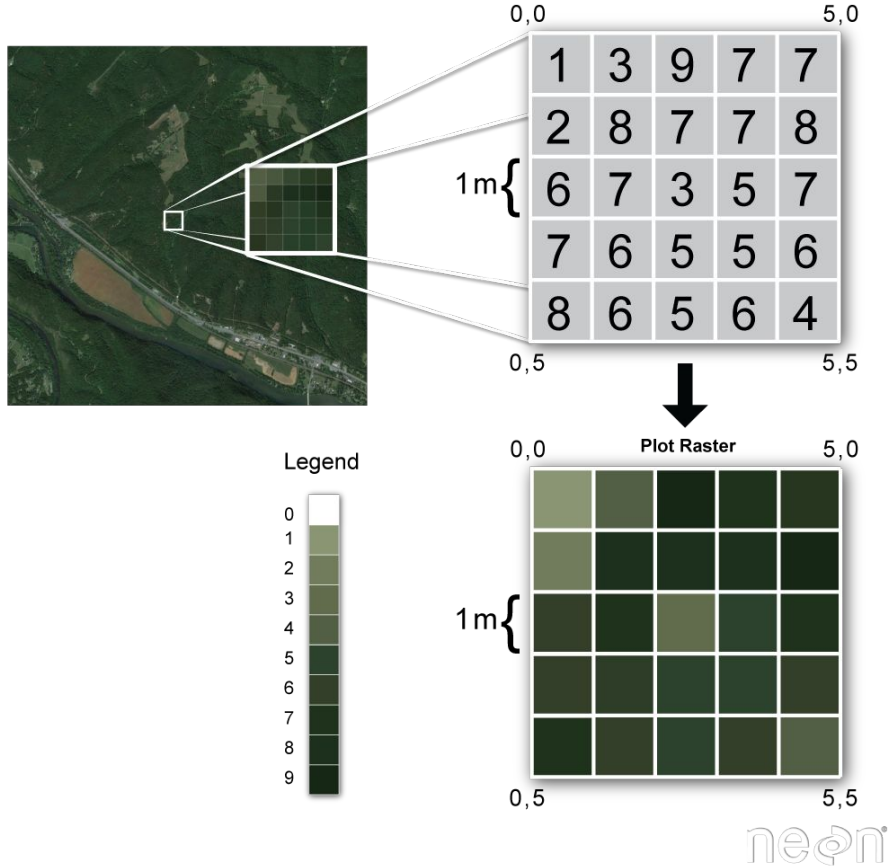
Vector data



https://datacarpentry.org/organization-geospatial/

Raster data

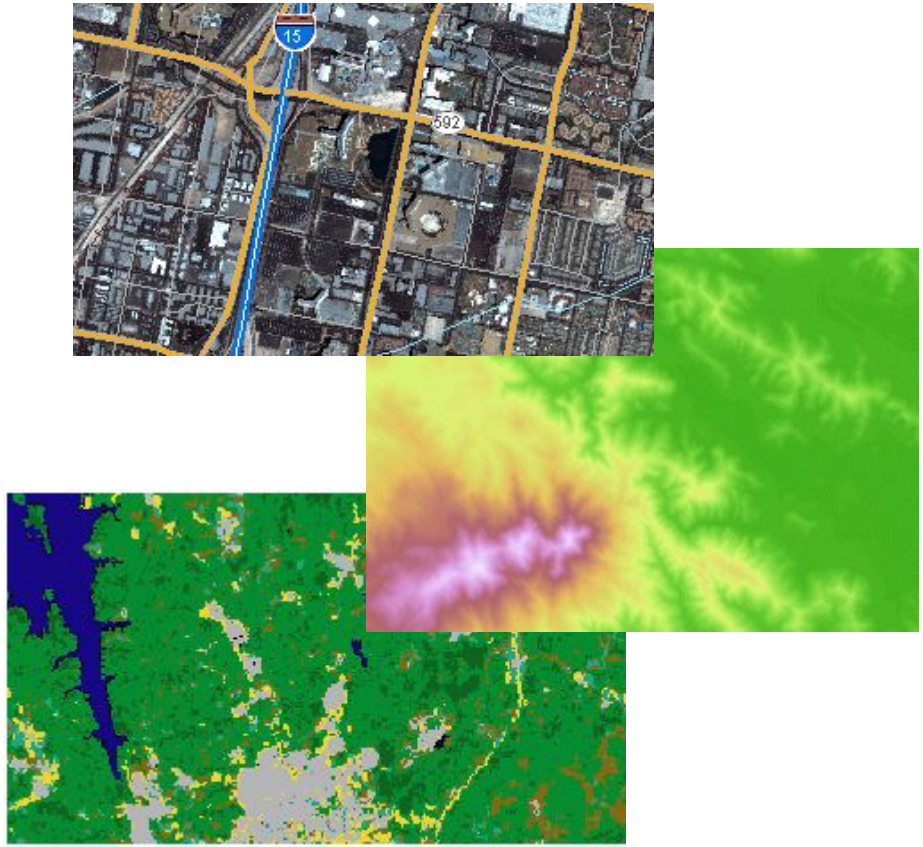
- Raster are simply arrays or matrices of data where each data cell is approximately the same size/shape (in a given CRS)



https://datacarpentry.org/organization-geospatial/

Raster data

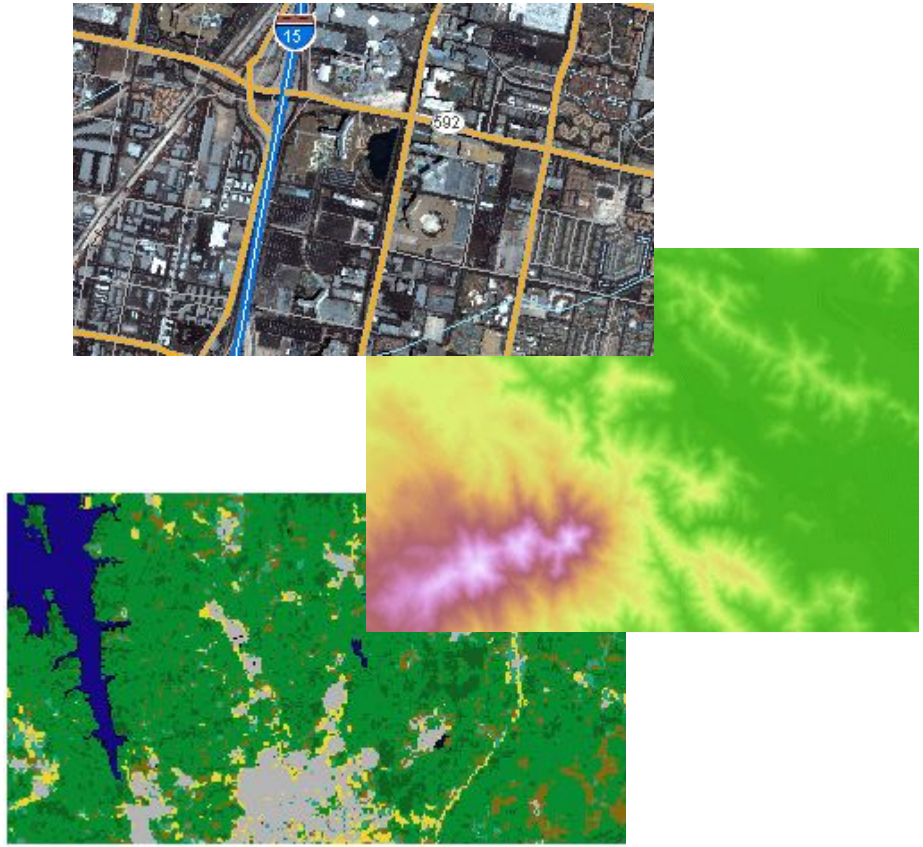
- Raster are simply arrays or matrices of data where each data cell is approximately the same size/shape (in a given CRS)
- Some examples include aerial and satellite data, elevation maps, and soil classification maps



https://desktop.arcgis.com/en/arcmap/10.3/manage-data/raster-and-images/what-is-raster-data.htm

Raster data

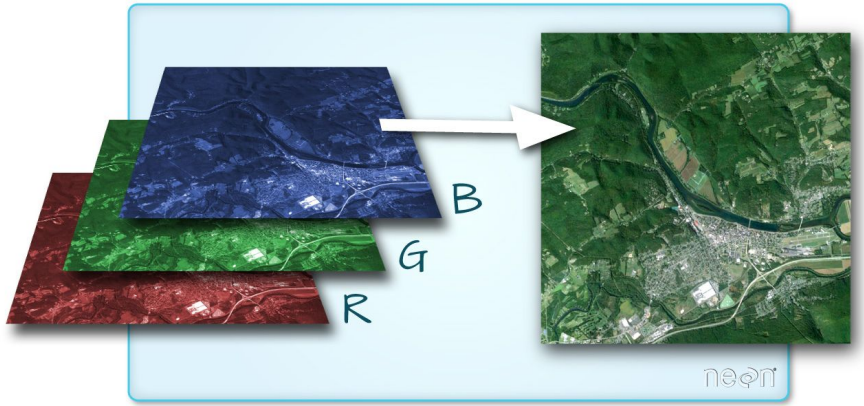
- Raster are simply arrays or matrices of data where each data cell is approximately the same size/shape (in a given CRS)
- Some examples include aerial and satellite data, elevation maps, and soil classification maps
- Data in raster cells/pixels can be continuous, discrete, or categorical



https://desktop.arcgis.com/en/arcmap/10.3/manage-data/raster-and-images/what-is-raster-data.htm

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



https://datacarpentry.org/organization-geospatial/

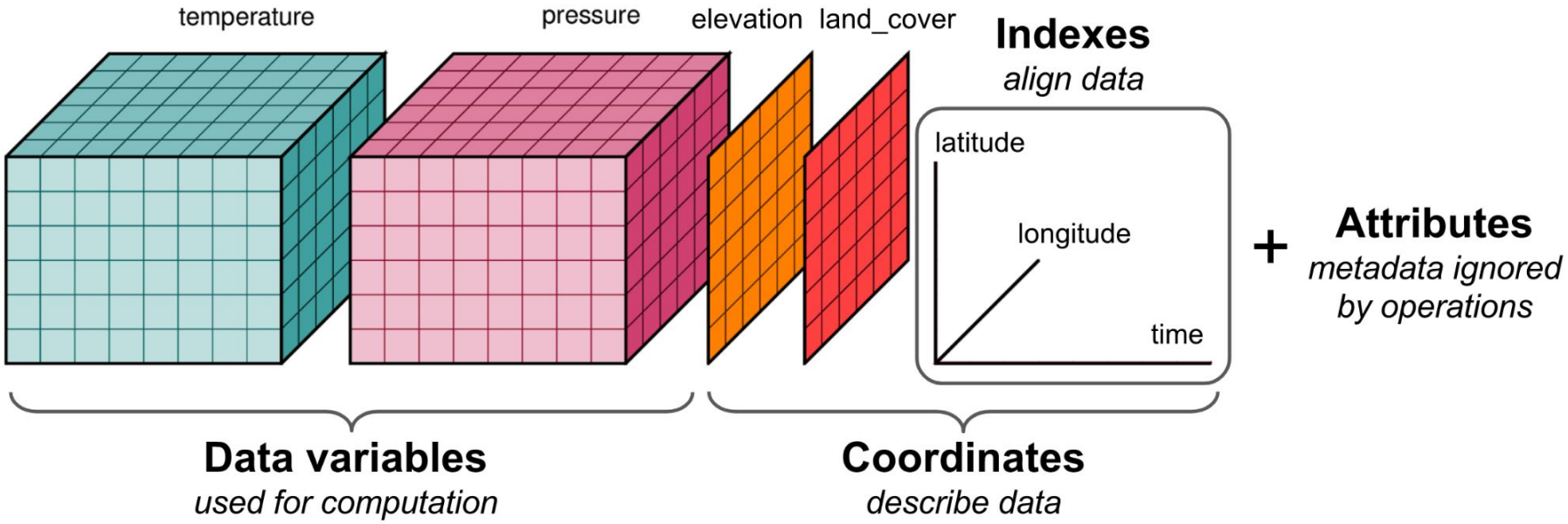
Working with raster data in python

- 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

https://docs.xarray.dev/en/stable/getting-started-guide/why-xarray.html

Dimensions and coordinates can represent many things!

- Reminder - dimensions refer to different “axes” of the arrays underlying them
- I’ve been giving examples of what dimensions can represent - name some
- Spatial maps (2d)! (lat, lon) or (latitude, longitude)
- Spatiotemporal data (3d)! (time, lat, lon) or (lat, lon, time) or etc
- Bands in remote sensing! (band, x, y)
- Ensemble forecasting (4d)! (ens_member, time, x, y)

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

https://training.galaxyproject.org/training-material/topics/climate/tutorials/pangeo-notebook/slides-plain.html

That’s all for today

- Rest of time today you can work on forecasts & geopandas assignment
- On Wednesday we will do a hands-on tutorial of working with xarray & raster data