Using Google Earth Engine and Python to Extract Data and Visualize Imagery

Victoria A. Walker

ORISE (USDA-ARS)

Tools Used

- Python: numpy, pandas, matplotlib, xarray, cartopy, geemap

- Google Colab: python notebooks in the cloud
  - saves automatically to Google Drive
  - can be exported to .ipynb or .py files

- Google Earth Engine (GEE) API
  - access GEE tools via python

Connecting to a Cloud Project

Announcement: On **November 13, 2024**, all users will need to use a Cloud project in order to access Earth Engine. After this date, continued individual access without a Cloud project will require an exception.

- See instructions from Google how to set up your project.
- Linking to your project in python is simple: use ee.Authenticate() to confirm your login and then ee.Initialize() with the project name.

```
1 import ee
2 ee.Authenticate()
3 ee.Initialize(project="ee-vawalker") # insert your GEE cloud project name here
```

Basics of the Google Earth Engine API

- Documentation of the GEE API is available online along with guides for use in JavaScript and Python.
- The API runs server-side operations. These process in your cloud project and are what make it so you don't have to download the datasets yourself.
- .getInfo() fetches a result from server-side to client-side (onto your local machine). Only use this if you really need to.

ImageCollection

- There are many publicly available datasets. Each one has a homepage that includes how to call it via the API as well its description, data availability, bands, creator, and terms of use.
- As the name suggests, an ImageCollection is a collection of type Image. Each Image is made up of Bands.
- Bands are named variables. These layers can be literal satellite bands (e.g., the numerical “B1” – “B11” in LandSat 8) but they don’t have to be (e.g., the categorical “landcover” in the 2021 NLCD).

Example: Selecting NAIP data

```
1 ## Construct buffered point (AoI)
2 res = 250 # width of AOI bounds in meters
3 aoi = ee.Geometry.Point([lon, lat]).buffer(res/2).bounds()
```

Define an area of interest (e.g., a buffered point)

```
14 collection = ee.ImageCollection('USDA/NAIP/DOQQ') \
15                 .filterDate('2005-01-01', '2020-12-31') \
16                 .filterBounds(aoi) \
17                 .sort('system:time_start') \
18                 .select(["R","G","B"]) \
19                 .map(lambda x: x.clip(aoi))
```

Name of collection

Filter images to those within the AoI

Clip images to AoI

Filter between two dates

Subset to select bands

Integration with xArray

- xArray is a package that allows for labeled multi-dimensional arrays (think pandas except with n-D arrays)
- The xee package lets you import GEE data into xArray.

Example: Pull hourly NLDAS precip and resample to daily.

Publication-quality maps with cartoee

- cartopy is a common python package for creating publication-quality maps. cartoee integrates that functionality with GEE.
- Available as a stand-alone package. Also bundled in geemap.

Example: Plotting the 2021 NLCD.

Animations with geemap

- geemap is a package for analysis and visualization of GEE data.
- Many built-in options; not just for maps anymore.

Example: Timeseries of NAIP imagery around a point.

More on geemap

- Dr. Qiusheng Wu (the creator of geemap) has a youtube channel with many video tutorials. There are also examples in the main documentation.

Questions?

Materials are available on Simon Kraatz's github.