Addressing Global Sustainability Challenges with Jupyter Technologies and Cloud-based Geospatial Data Platforms

Tyler Erickson, Ph.D.

linkedin.com/in/tylere

github.com/tylere/JupyterCon-2023

Background

We have a global grand challenge of making living on Earth sustainable, as it undergoes significant change due to human activity. The world's inhabitants are already experiencing many negative impacts, such as displacement from sea-level rise, ecosystem degradation, and increases in extreme weather events. People need access to better information so they can understand the impacts that are already occurring, mitigate changes we can prevent from occurring, predict what changes will occur in the future, and prepare to adapt to those changes that we cannot mitigate.

Fortunately, in recent decades there has been a rapid increase in data about the state of the Earth. Satellites have been continuously observing the globe for over 50 years, and Earth System Models (ESMs) generate weather predictions of the coming weeks and climate projections of what may happen in future decades. However, several barriers to progress remain:

- Data volume & velocity is a major challenge for most potential users. Satellites and ESMs generate terabytes/day and archives are petabytes in size.
- Raw data needs to be processed into actionable information that is appropriate for decision makers.

Jupyter Widgets

Widgets are "*eventful python objects that have a representation in the browser*" (Jupyter Widget Docs). Widgets enable bidirectional communication between the client side UI (JavaScript) and server side (ipython kernel) components.

Widgets can be used to enable rapid interaction with large (terabyte to petabyte) geospatial datasets via the following process:

1. Notebook users interact with widget representations in the browser, triggering one or more *widget events*.
2. Widget events trigger Python code that requests representations of geospatial data from a remote server. The remote server processes data to a small size (~100KB) before returning it to the browser.

Geospatial Data Analysis Systems

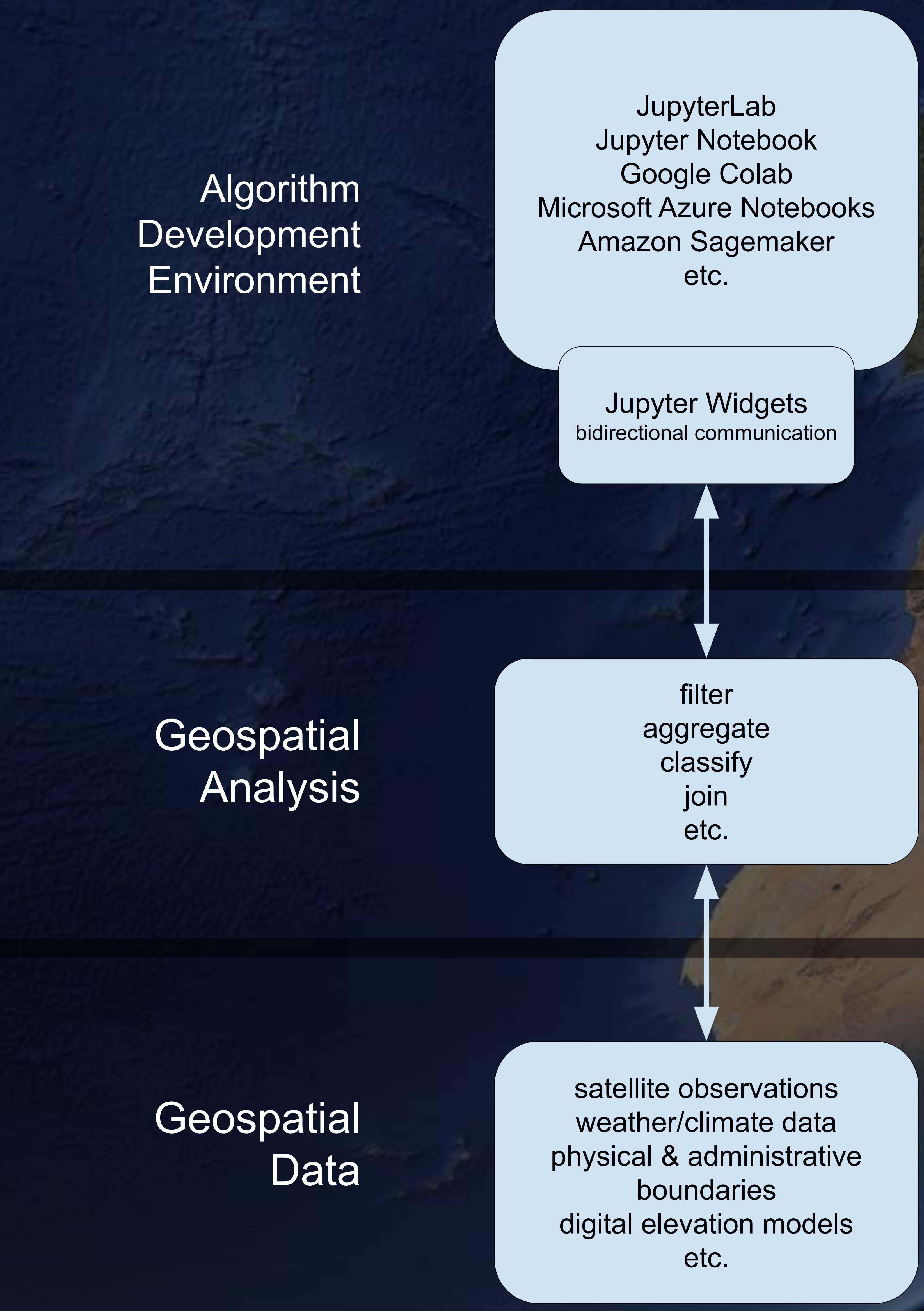
Barriers are being addressed through the use of ***data-proximate computing*** systems that are optimized for geospatial data analysis. These systems move compute close to data storage, and are an improvement over the historical practice (and bottleneck) of downloading large datasets for local analysis.

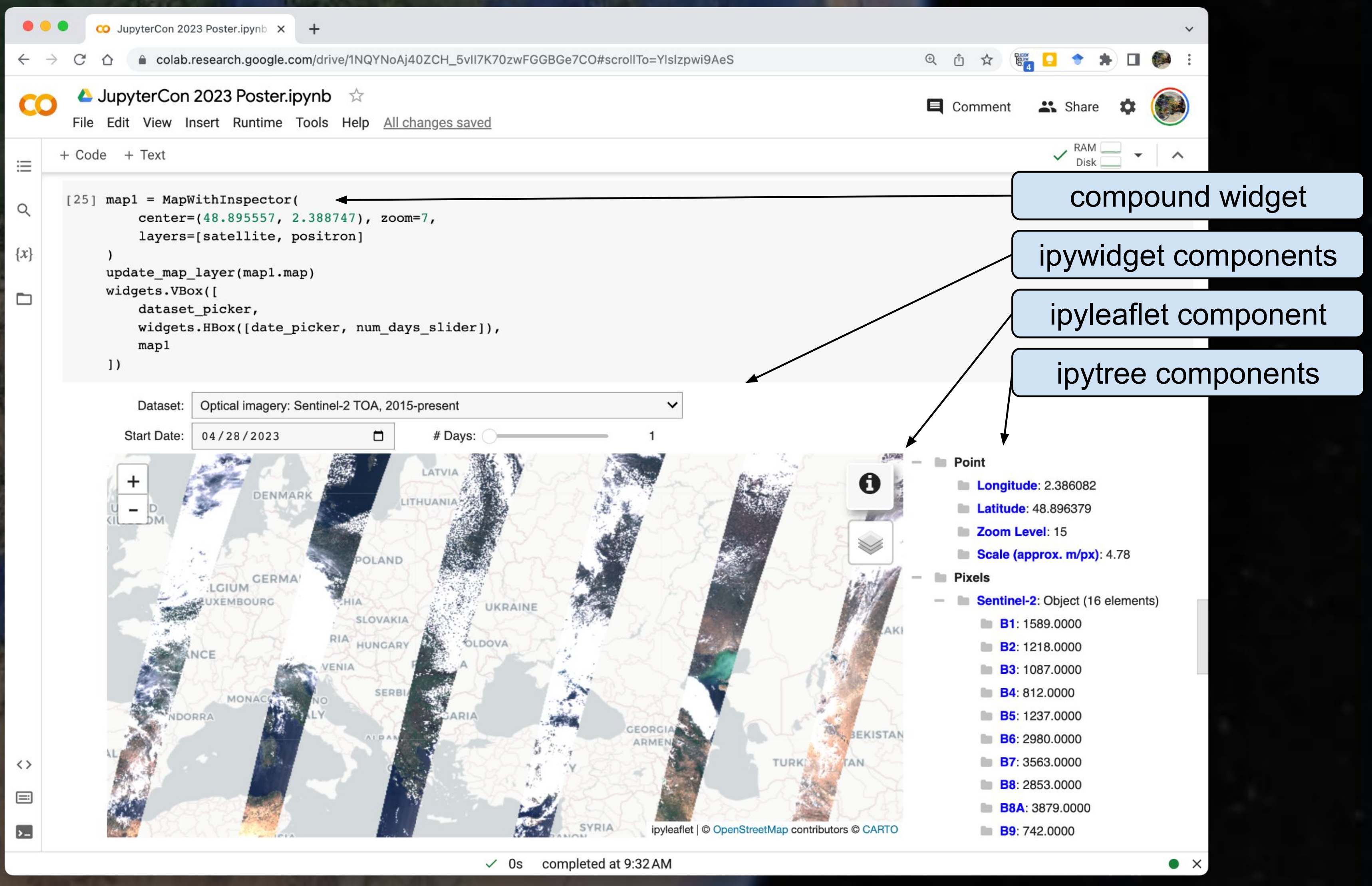
Some examples of these systems include

- Pangeo (pangeo.io/) - an open source software stack consisting of Jupyter, Dask, and Xarray.
- Google Earth Engine (developers.google.com/earth-engine) - a cloud-based geospatial analysis system, including a >50 PB data catalog.

Ideas for Future Improvements

1. Make it easier to interact with common spatio-temporal datasets by creating additional compound widgets that exploit the structure of the datasets. For example, compound widgets could be created for polar orbiting satellite imagery, stationary sampling stations, nested watersheds, etc.
2. Deploy examples/tutorials using JupyterLite, preloaded with widgets and geospatial libraries, to remove the need for a hosted geospatial server (for some use cases).




JupyterCon 2023 Poster.ipynb
colab.research.google.com/drive/1NQYNoAj40ZCH_5vIl7K70zwFGGBGe7CO#scrollTo=YlslzpwI9AeS
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
+ Code + Text
RAM
Disk
compound widget
ipywidget components
ipyleaflet component
ipytree components
ipyleaflet | © OpenStreetMap contributors © CARTO

```
[25] map1 = MapWithInspector(
         center=(48.895557, 2.388747), zoom=7,
         layers=[satellite, positron]
     )
     update_map_layer(map1.map)
     widgets.VBox([
         dataset_picker,
         widgets.HBox([date_picker, num_days_slider]),
         map1
     ])
```

Dataset: Optical imagery: Sentinel-2 TOA, 2015-present

Start Date: 04/28/2023 # Days: 1

- Point
  - **Longitude**: 2.386082
  - **Latitude**: 48.896379
  - **Zoom Level**: 15
  - **Scale (approx. m/px)**: 4.78
- Pixels
  - **Sentinel-2**: Object (16 elements)
    - **B1**: 1589.0000
    - **B2**: 1218.0000
    - **B3**: 1087.0000
    - **B4**: 812.0000
    - **B5**: 1237.0000
    - **B6**: 2980.0000
    - **B7**: 3563.0000
    - **B8**: 2853.0000
    - **B8A**: 3879.0000
    - **B9**: 742.0000

✓ 0s completed at 9:32 AM

xarray Google Earth Engine dask Microsoft Planetary Computer

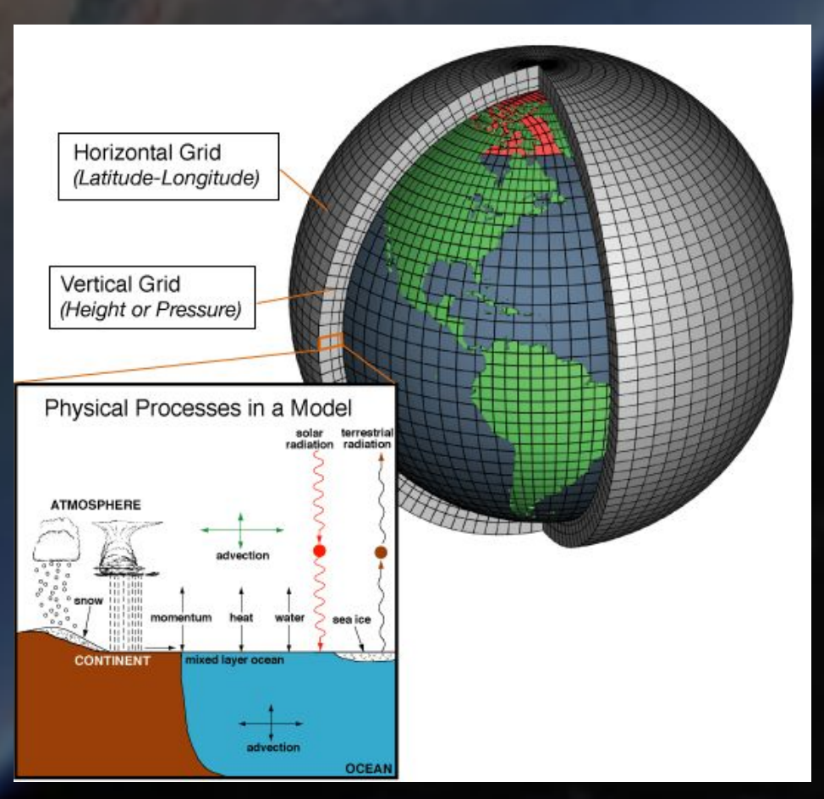
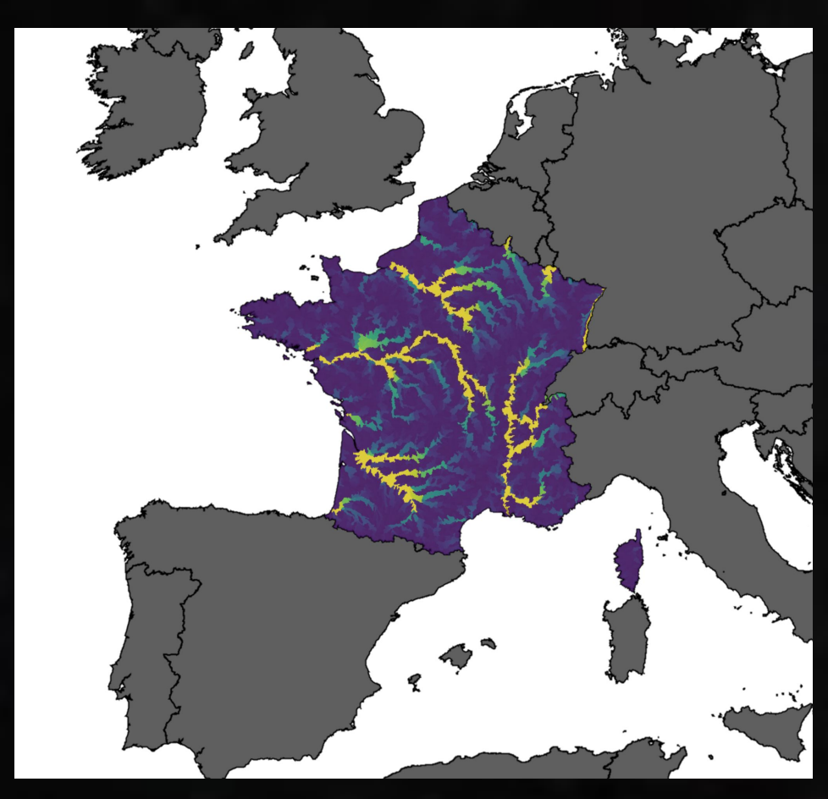

Horizontal Grid (Latitude-Longitude)
Vertical Grid (Height or Pressure)
Physical Processes in a Model

Image credit: Climate.gov



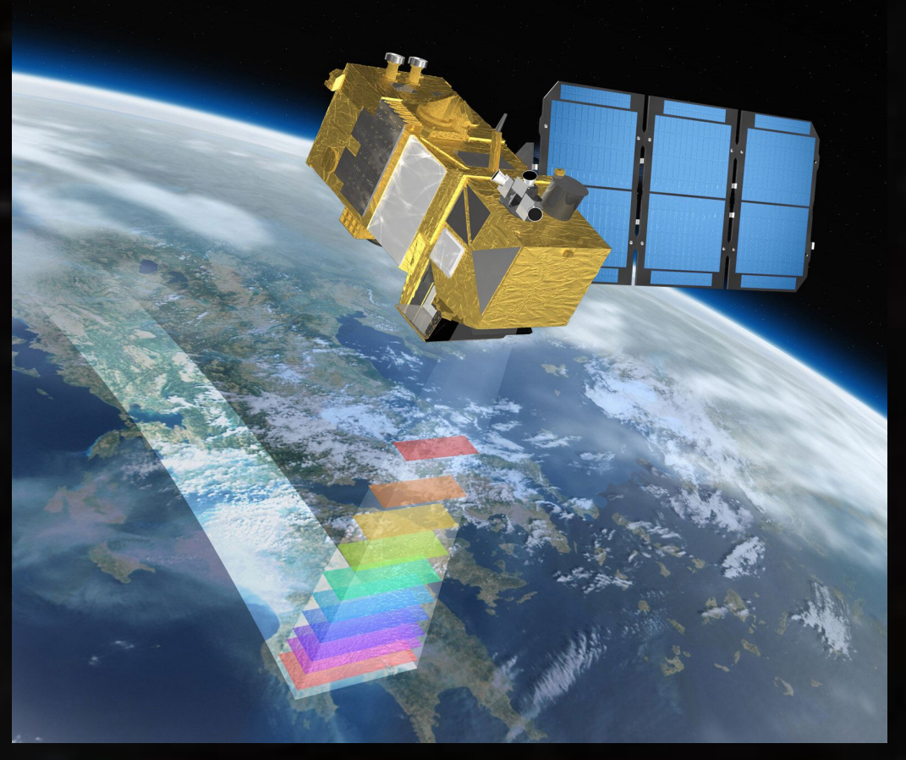


Image credit: ESA

aws Google Cloud Microsoft Azure