

Temporal aggregations

- Statistical mosaics:
 - Filter images and calculate a stack statistic
 - Filter collection (DOY, Cloud Cover, ...)
 - Mask clouds (FMASK, NDCI, ...)
 - Calculate a statistic (mean, median, etc.)
 - <https://code.earthengine.google.com/47f2685d752c1614fe6b842942c7e3dc>
- Best available pixel (BAP) mosaics:
 - Sort each pixel so the best is at the top
 - Distance to clouds and shadows masks
 - Atmospheric Opacity
 - Day of the year, ...
 - <https://code.earthengine.google.com/e27240a92ecf64bbadf8a082b91c711c?hideCode=true>



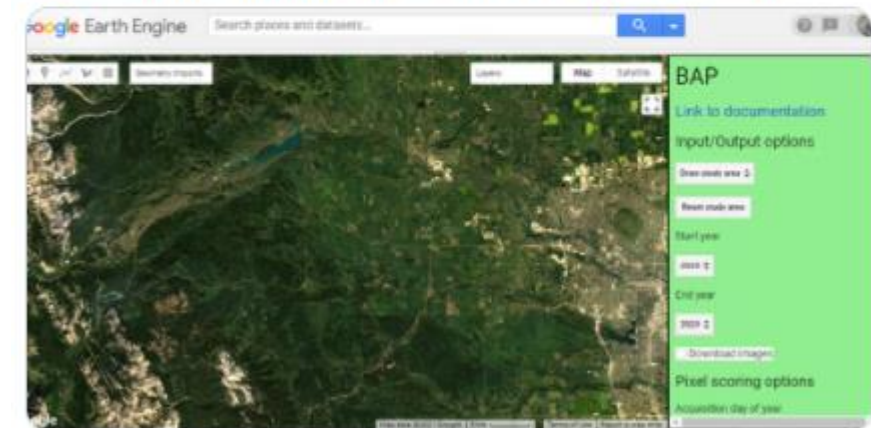
Mike Wulder
@mikewulder

The best-available-pixel (BAP) tool you have been waiting for! Implemented on [#GoogleEarthEngine](#) ([#GEE](#)). [#Landsat](#)

In [#GEEBAP](#) can tune composite parameters, create a [#timeseries](#), set area of interest, AND download surface reflectance outcomes!

Try it out:

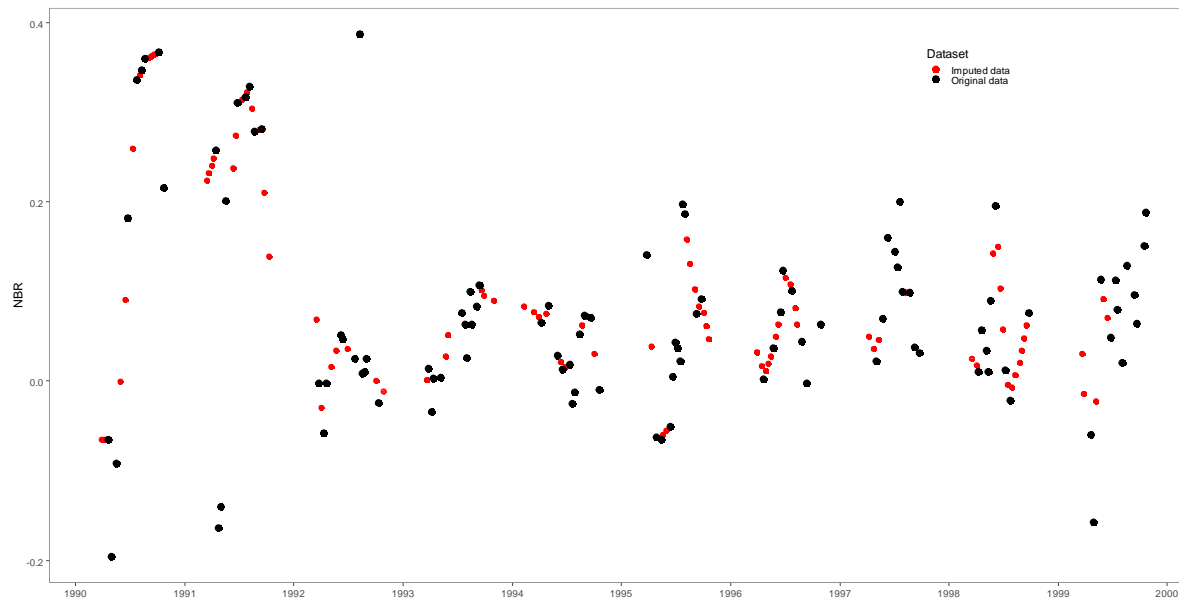
code.earthengine.google.com/e27240a92ecf64bbadf8a082b91c711c?hideCode=true



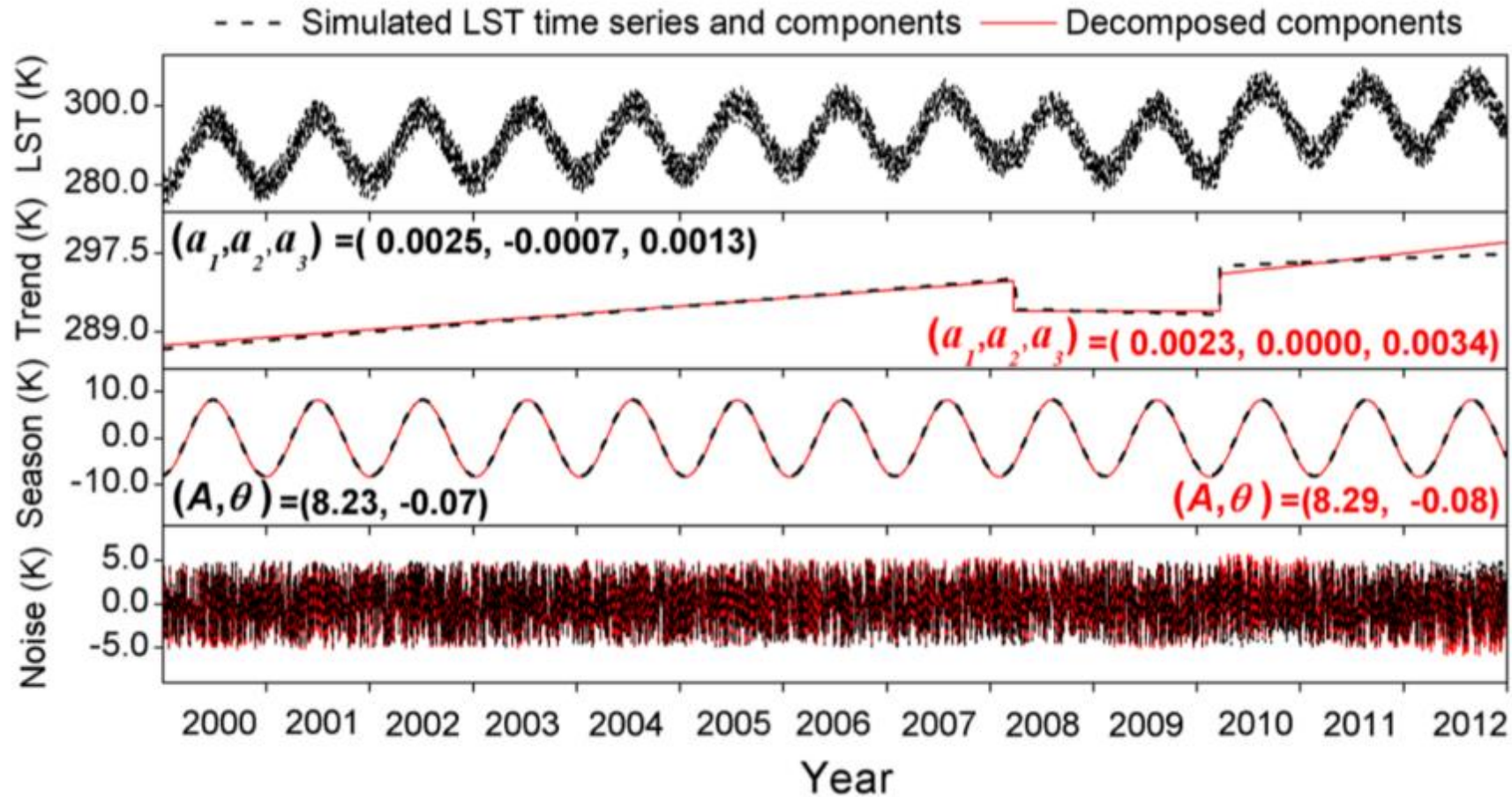
Saverio Francini and 7 others

11:00 AM - Apr 29, 2021 - Twitter Web App

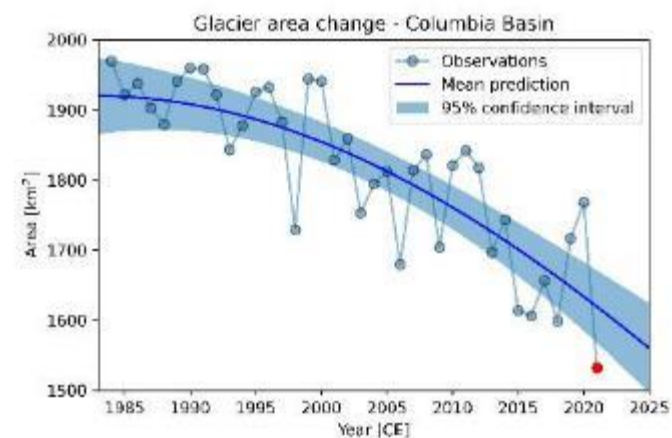
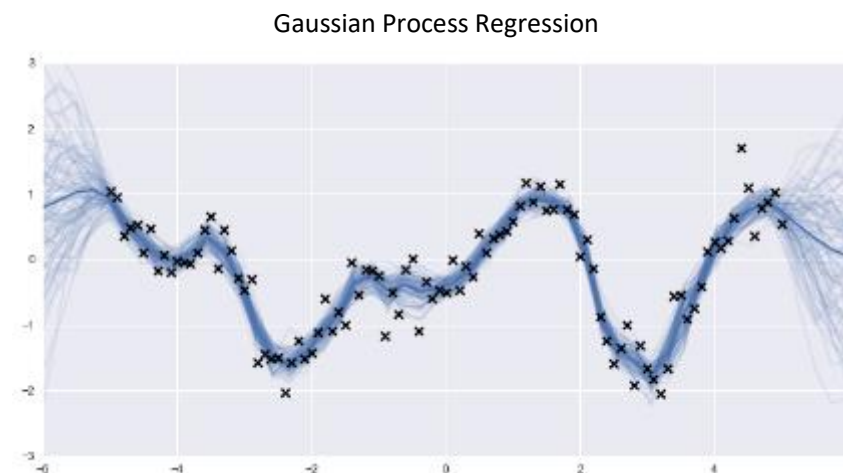
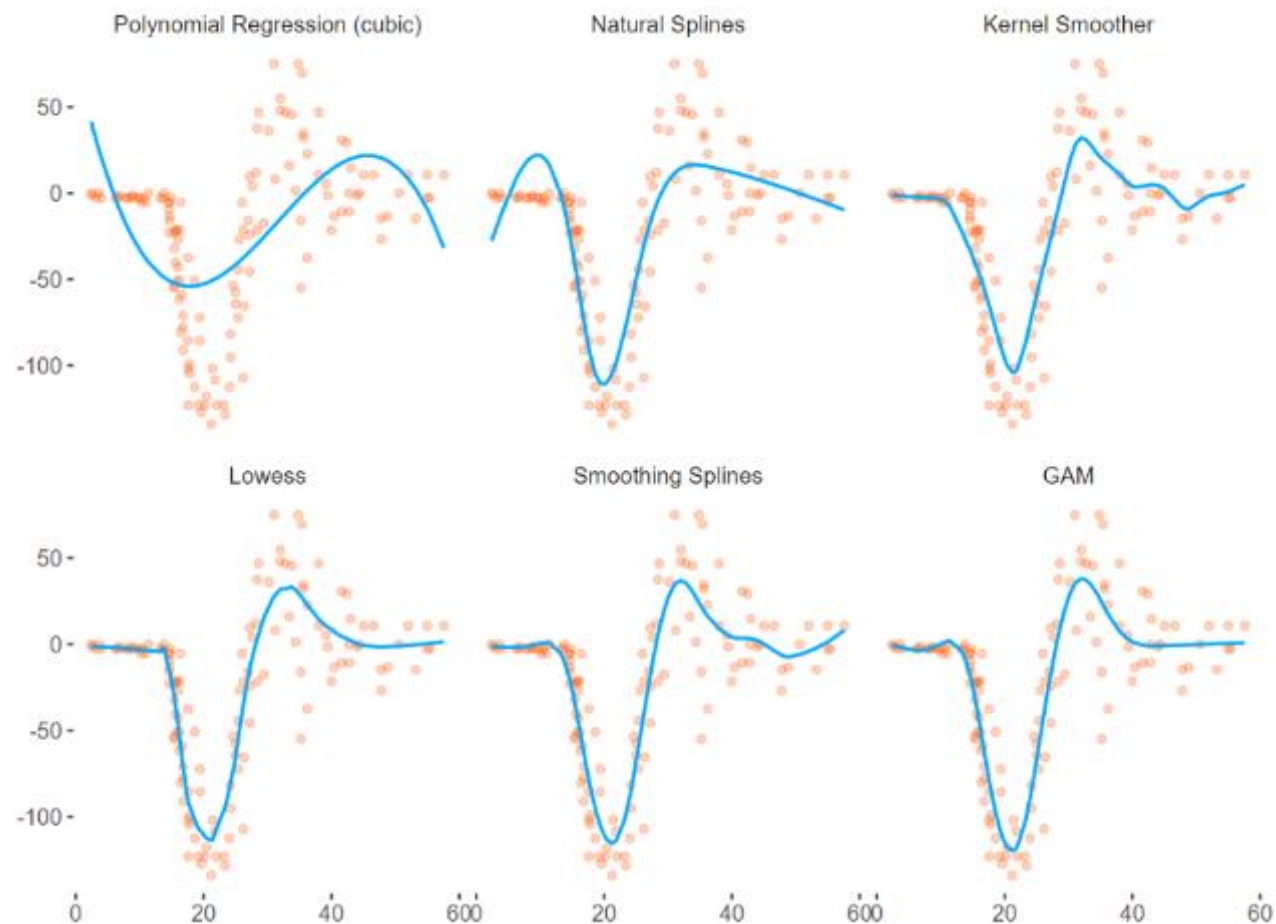
Gap filling: Impute NoData



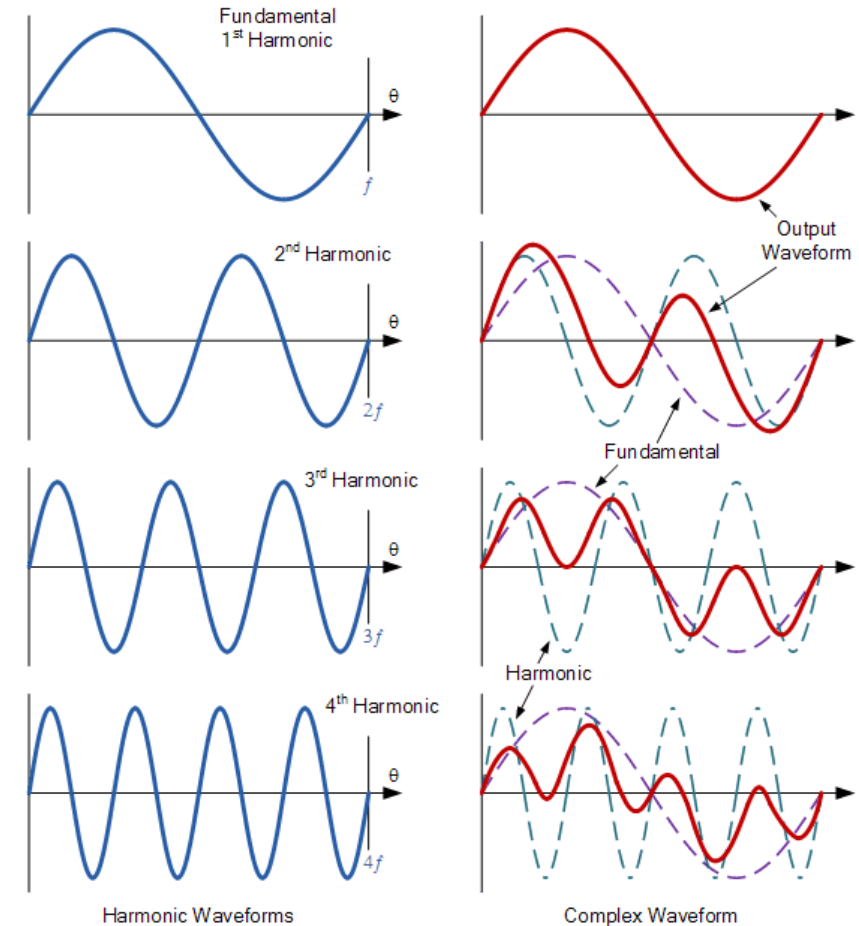
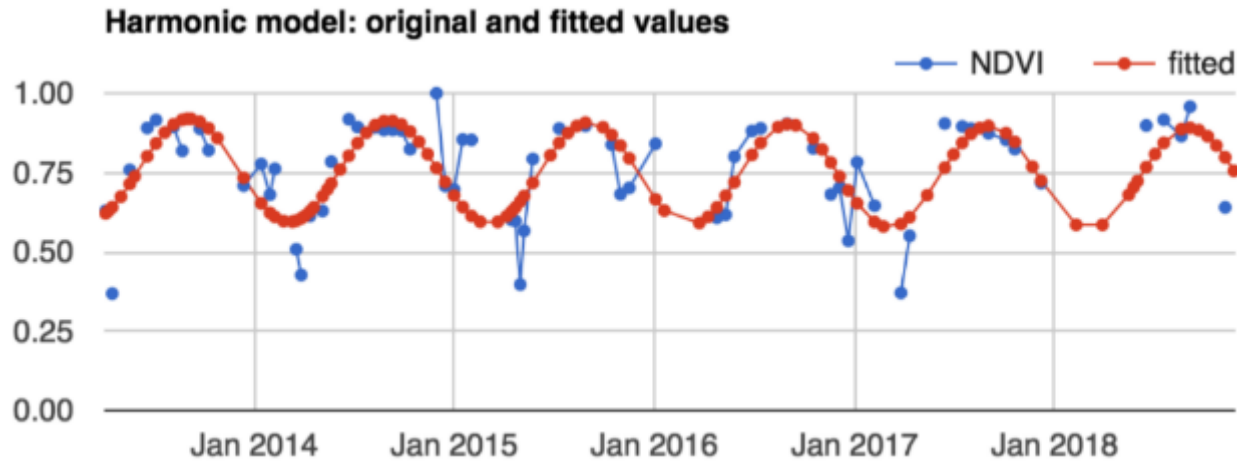
Detrend a timeseries



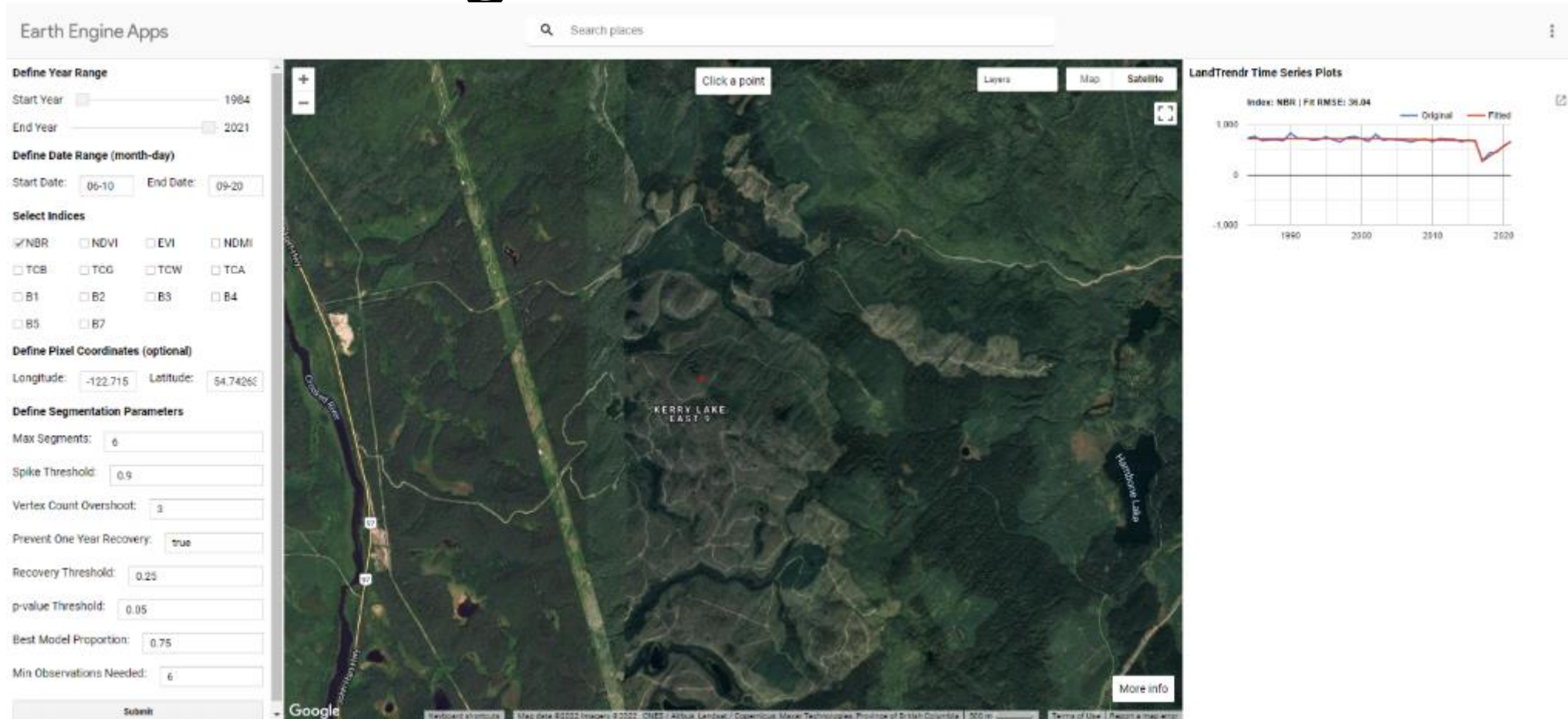
Time series smoothing and interpolation



Harmonic and Seasonal Interpolation



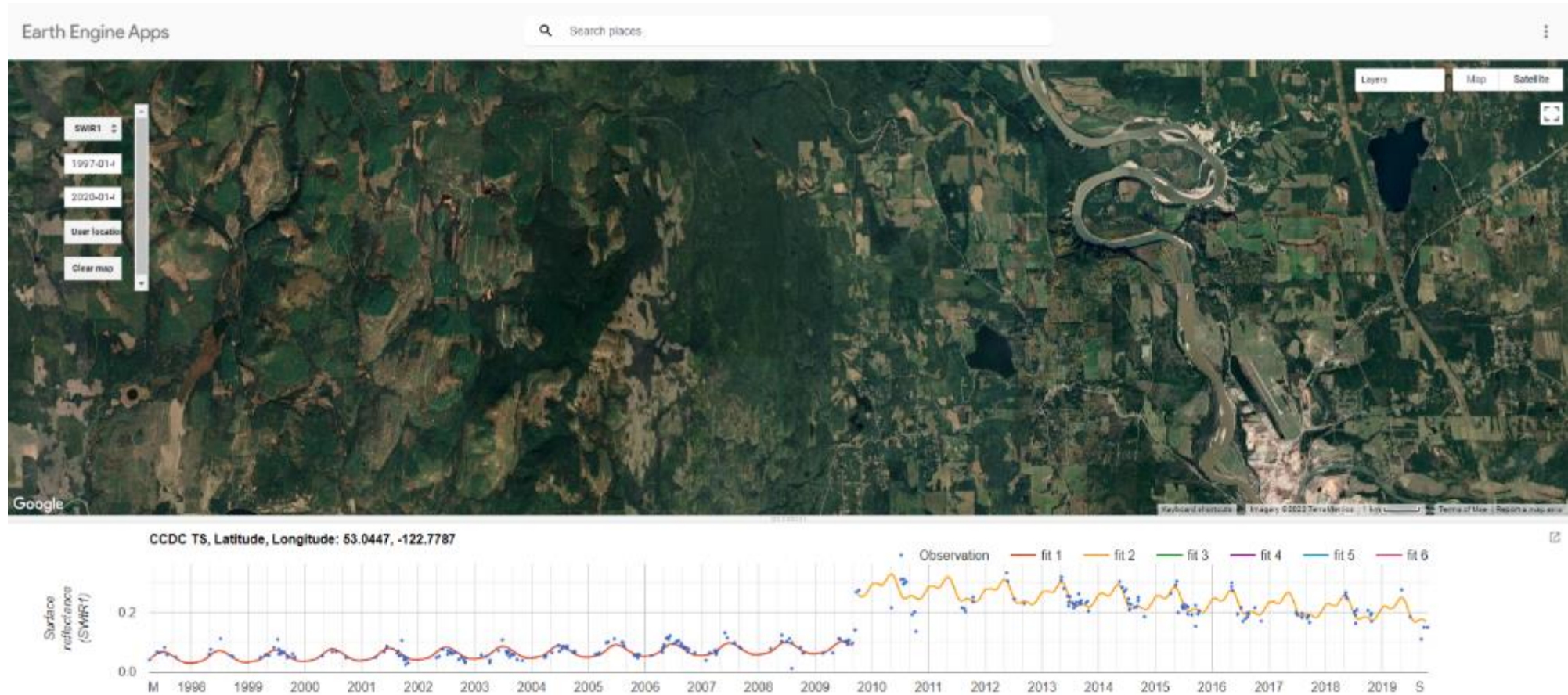
LandTrendR Algorithm



<https://emapr.github.io/LT-GEE/ui-applications.html#ui-landtrendr-pixel-time-series-plotter>

<https://emaprlab.users.earthengine.app/view/lt-gee-pixel-time-series>

Continuous Change Detection and Classification (CCDC) Algorithm



Dynamic time warping



Journal of Statistical Software

MMMMMM YYYY, Volume VV, Issue II. <http://www.jstatsoft.org/>

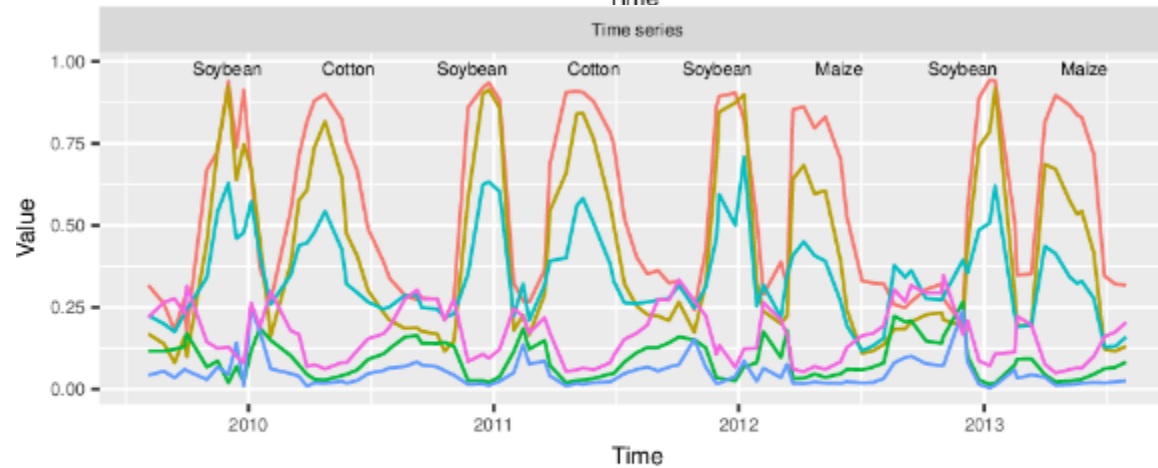
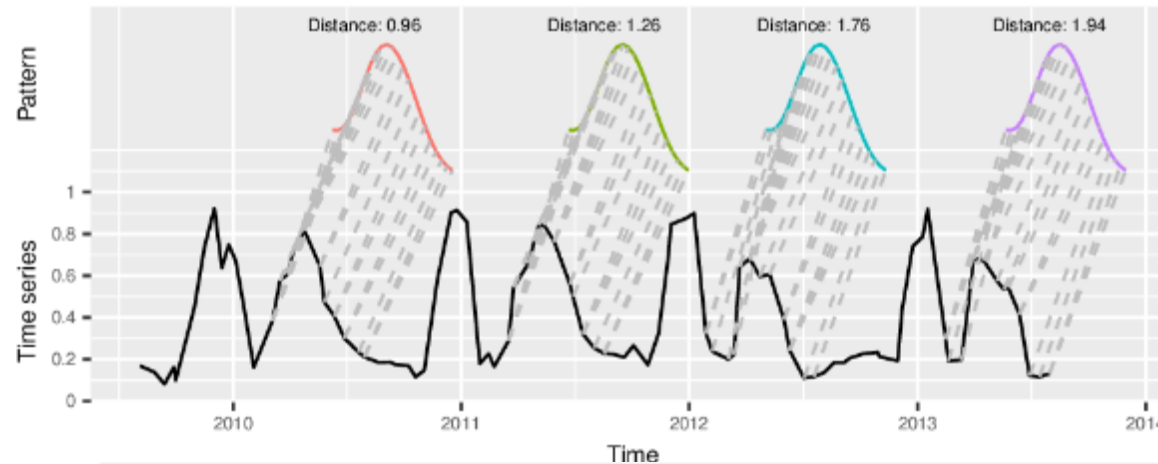
dtwSat: Time-Weighted Dynamic Time Warping for
Satellite Image Time Series Analysis in R

Victor Maus
University of Münster

Gilberto Câmara
INPE

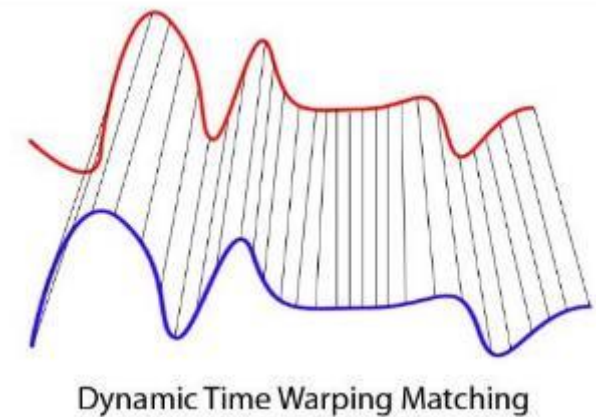
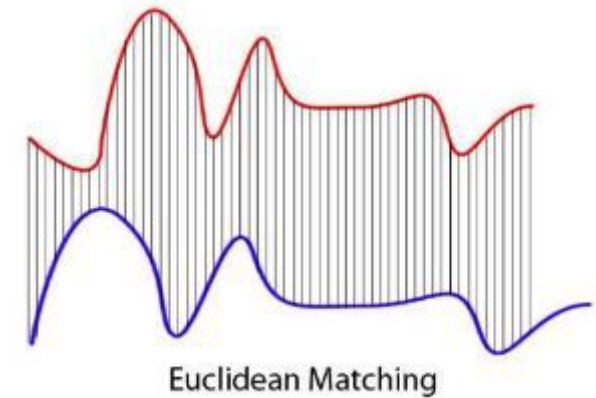
Marius Appel
University of Münster

Edzer Pebesma
University of Münster



Bands

ndvi	red	blue
evi	nir	mir



<https://towardsdatascience.com/dynamic-time-warping-3933f25fcdd>



Working with `large` datasets

March 7, 2024

Alexandre.Bevington@gov.bc.ca

What are large datasets?

- More capabilities = Bigger questions

```
from sentinelsat import SentinelAPI
api = SentinelAPI('username', 'pw')
```

```
S1 = api.query(date=('2021-01-01T00:00:00Z', '2021-12-31T23:59:59Z'), platformname='Sentinel-1', producttype='SLC', sensoroperationalmode='IW')
```

```
api.get_products_size(S1)
```

~ 2.07 Petabyte of Sentinel-1 IW SLC data (NOT INCLUDING GRD)

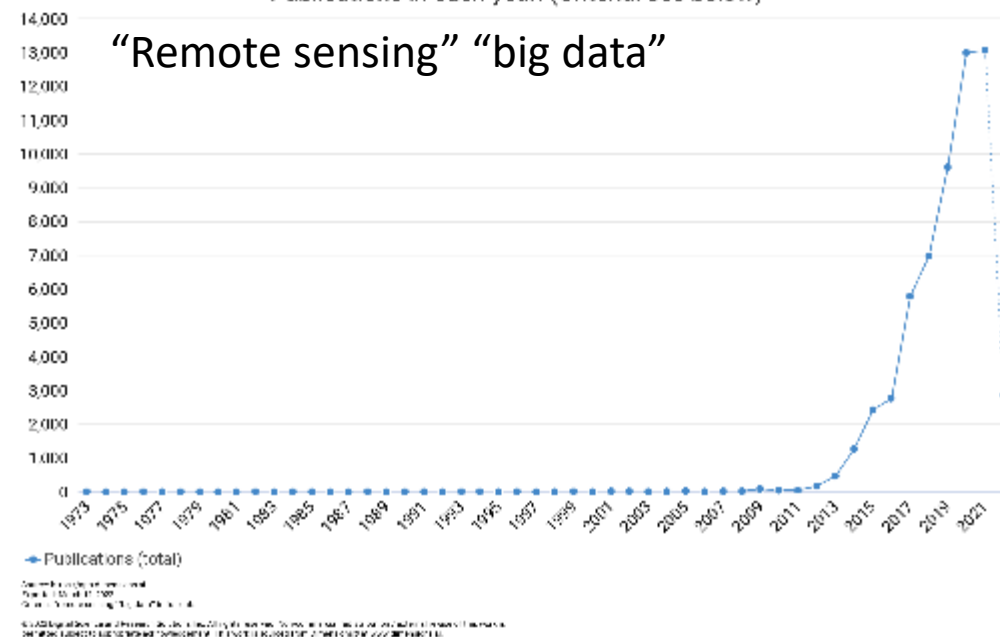
```
S2 = api.query(date=('2021-01-01T00:00:00Z', '2021-12-31T23:59:59Z'), platformname='Sentinel-2', producttype='S2MSI1C')
```

```
api.get_products_size(S2)
```

~ 0.94 Petabyte of Sentinel-2 L1C dat

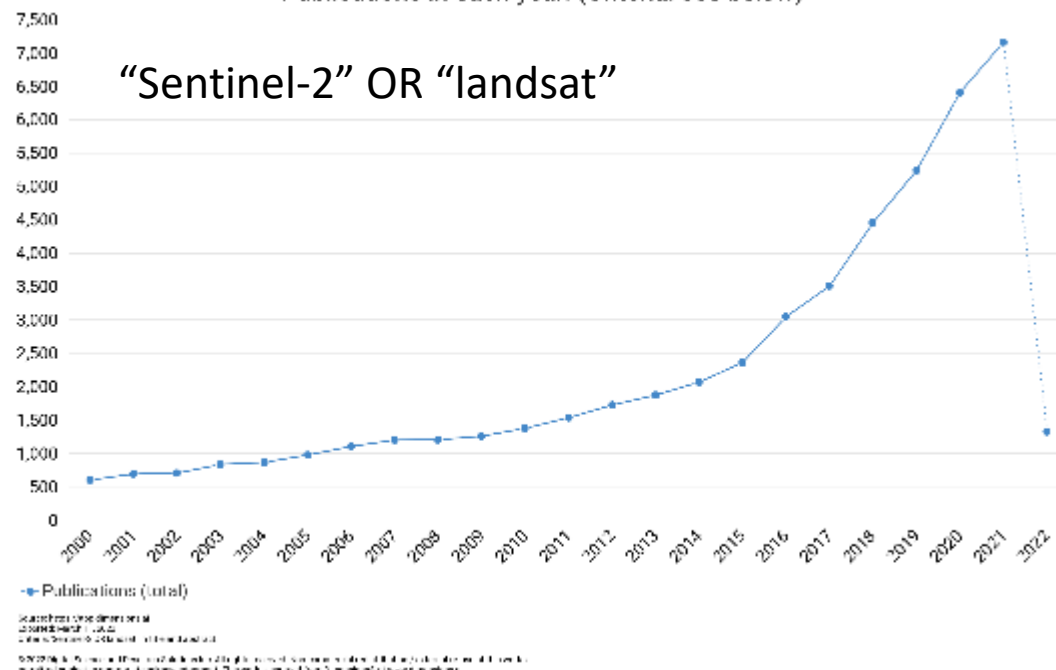
Publications in each year. (Criteria: see below)

“Remote sensing” “big data”

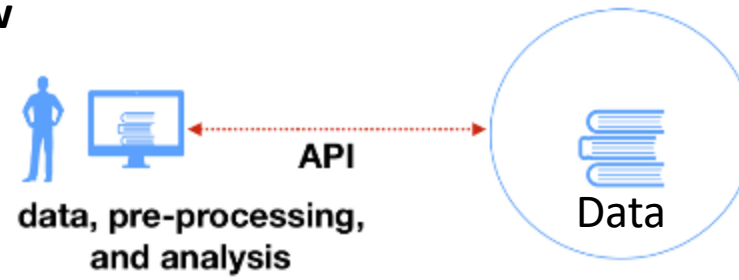


Publications in each year. (Criteria: see below)

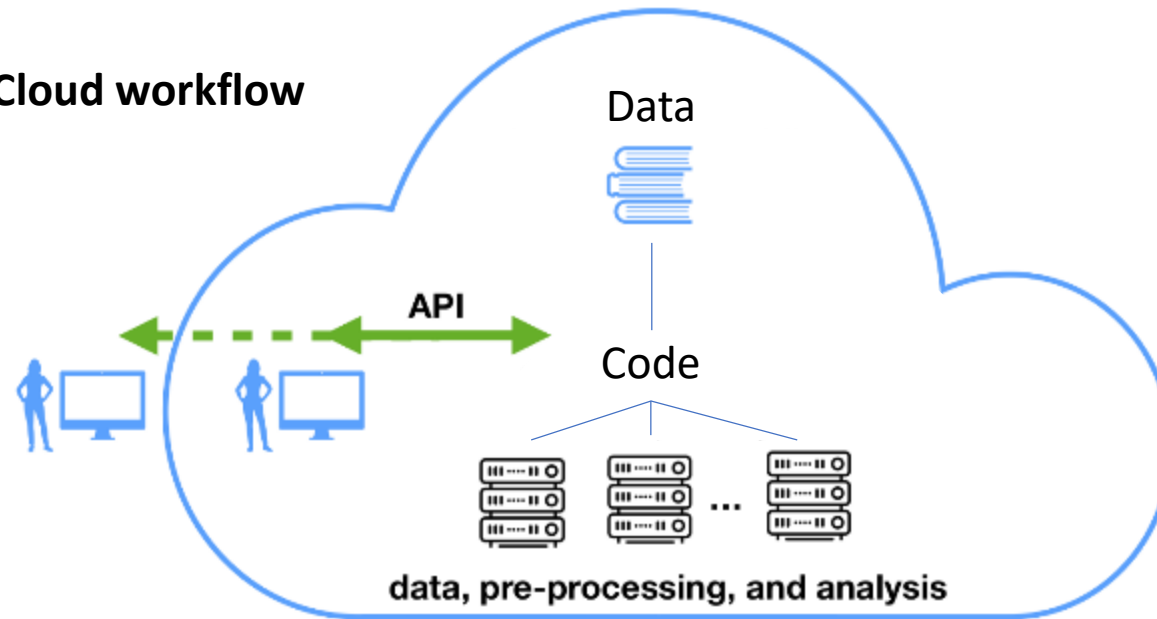
“Sentinel-2” OR “landsat”



Local workflow



Cloud workflow



Computer speed

- **CPU – Central Processing Unit**
Speed of processor, number of cores (How many workers)
- **RAM – Random Access Memory**
Store working data and machine code (Multitasking)
- **SSD vs HDD – Solid State vs Hard Disk**
Read/write speed and total storage
- **GPU - Graphics Processing Unit**
Speeds up image visualization and processing, optimal for some tasks
- etc



- 8 bit = 256
- 10 bit = 1024
- 12 bit = 4096
- 16 bit = 65,536

Res (m)	Pixels	8 bit	16 bit
1	1 trillion	1 TB	2 TB
10	10 billion	10 GB	20 GB
100	100 million	100 MB	200 MB
1000	1 million	1 MB	1 MB

- **Serial computing**

- A problem is broken into instructions
- Executed sequentially on a single processor
- One instruction executed at a time

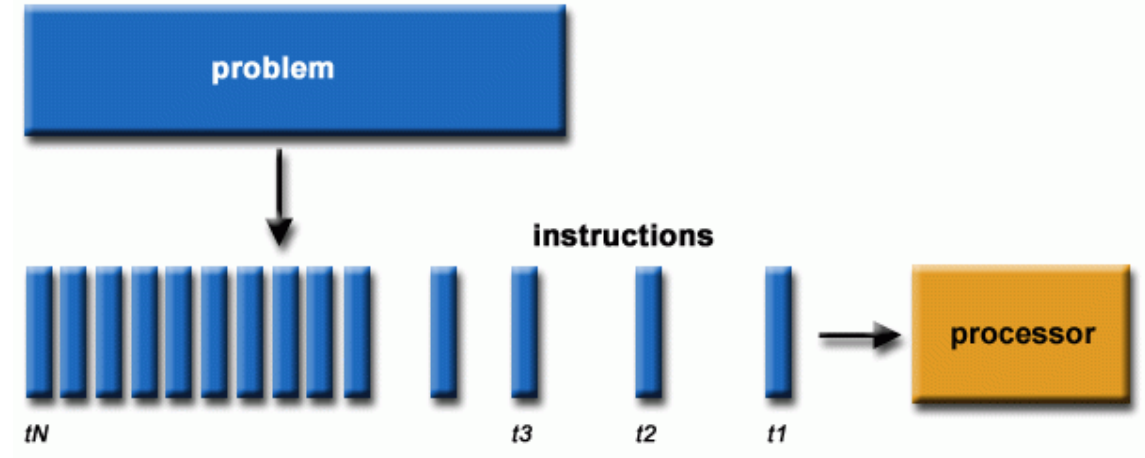
- **Parallel Computing**

- A problem is broken into parts
- Each part is broken into instructions
- Execute simultaneously on different processors
- Requires orchestration, sometimes not worth it

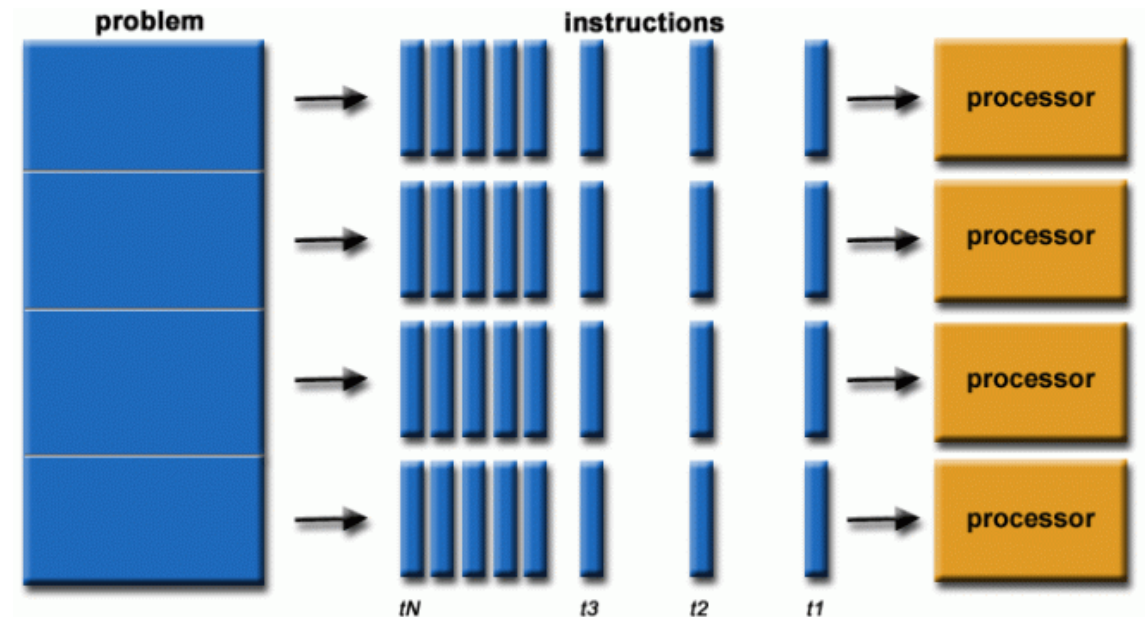
- **Hyper-threading**

- Better task scheduling
- Minimizes processor downtime
- Works for both serial and parallel computing
- Not equivalent to more cores

Serial computing

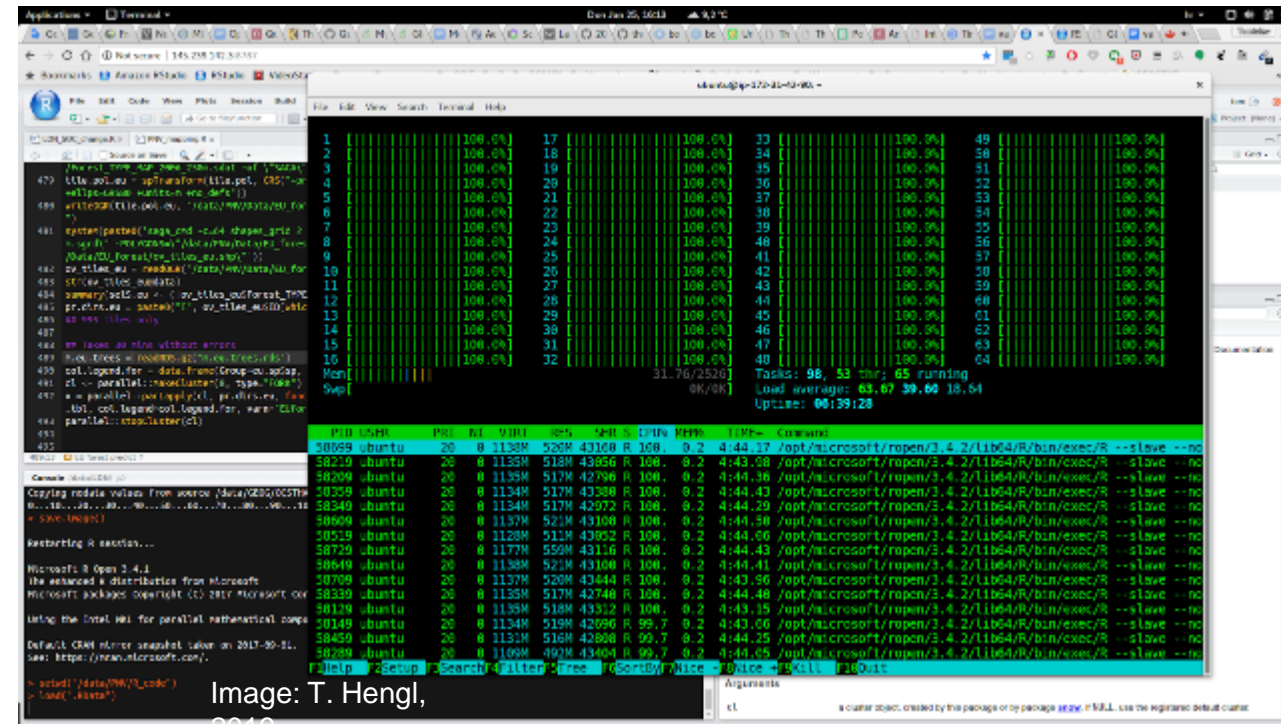


Parallel Computing



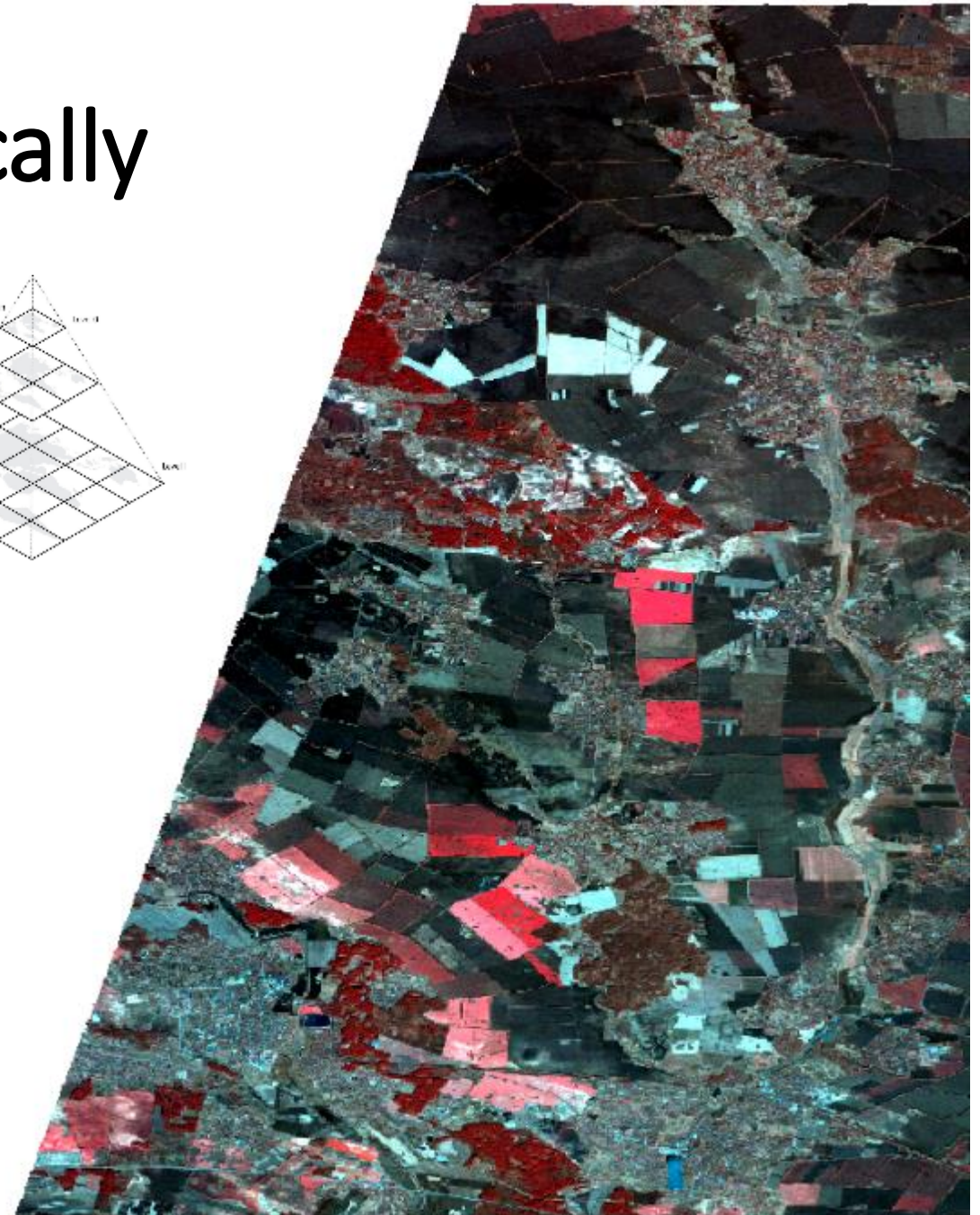
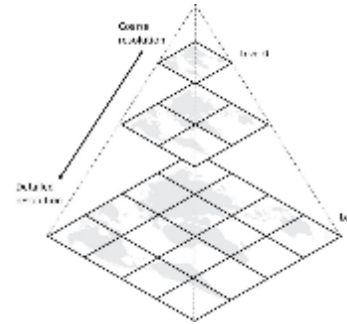
How to work in Parallel?

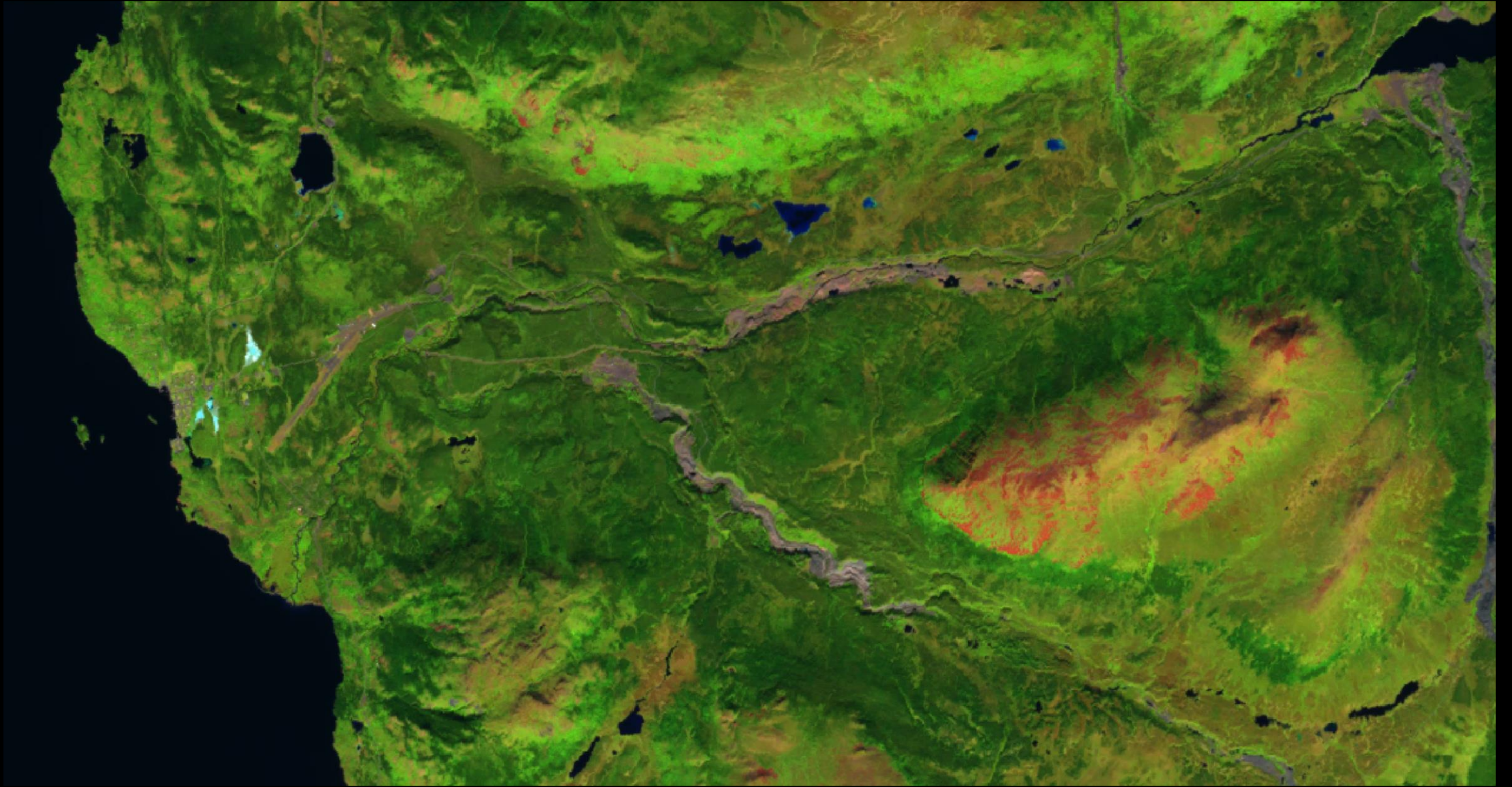
- Not all functions can be run in parallel
- SAGA GIS runs in parallel by default
- GRASS can be parallel with OpenMP
- Python: use `Dask`
- R: use `Future`
- ArcGIS: Available for some functions, not all



Managing large images locally

- Tiles
 - Easier to manage as small tiles
- Visualize all tiles at once
 - **Virtual raster (gdalbuildvrt)**
Creates index of all tiles (small file)
 - **Mosaic (gdalmerge)**
Combines all images into a large file
- Speed up visualization
 - **Overviews (gdaladdo)**
Creates multiple reduced resolution layers that are used at different zoom levels, speeds up visualization. Layers stored in *.ovr file

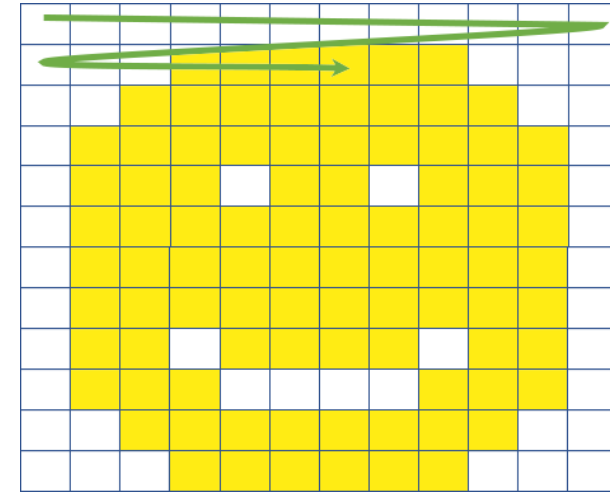




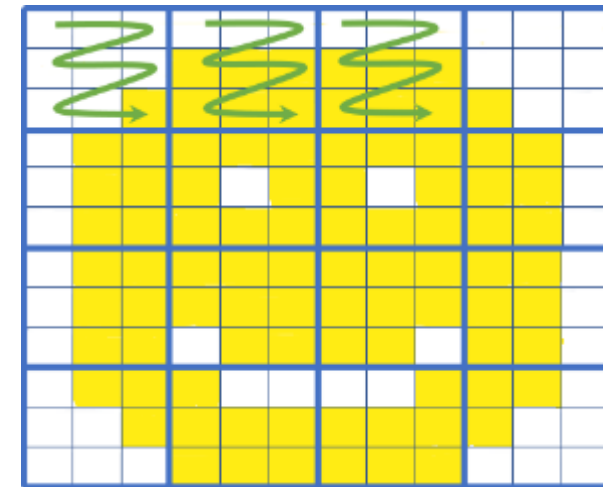
Cloud optimized geotiffs

- A COG is a regular GeoTIFF
- COGs have an internal organization that supports efficient access via HTTP GET range requests
- Supports overviews
- Clip rasters BEFORE processing downloading
- Used by:
 - STAC Index
 - Google Earth Engine
 - DigitalGlobe/Mazar
 - USGS
 - etc.

Typical



COG





+ Code + Text Copy to Drive

For this demo, we will use data from <https://www.maxar.com/open-data/california-colorado-fires> for mapping California and Colorado. List of COGs can be found [here](#).

```
[ ] import ee
import geemap
```

```
[ ] Map = geemap.Map()
Map
```

```
Map(center=[40, -100], controls=(WidgetControl(options=['position'], widget=HBox(children=(ToggleButton(value=
```

```
[ ] url = 'https://opendata.digitalglobe.com/events/california-fire-2020/pre-event/2018-02-16/pine-gulch-fire2018-02-16'
```

```
[ ] geemap.cog_bounds(url)
```

```
[-108.63447456563128,
 38.963980238226654,
-108.38008268561431,
 40.025815049929754]
```

```
[ ] geemap.cog_center(url)
```

```
(-108.5072786256228, 39.49489764407821)
```

```
[ ] geemap.cog_bands(url)
```

```
['band1', 'band2', 'band3']
```

```
[ ] geemap.cog_tile(url)
```

```
'https://titiler.xyzcog/tiles/WebMercatorQuad/{z}/{x}/{y}@1x?url=https%3A%2F%2Fopendata.digitalglobe.com%2Fevents%2Fcalifornia-fire-2020%2Fpre-event%2F2018-02-16%2Fpine-gulch-fire2018-02-16'
```

```
[ ] Map.add_cog_layer(url, name="Fire (pre-event)")
```

```
[ ] url2 = 'https://opendata.digitalglobe.com/events/california-fire-2020/post-event/2020-08-14/pine-gulch-fire2020-08-14'
```

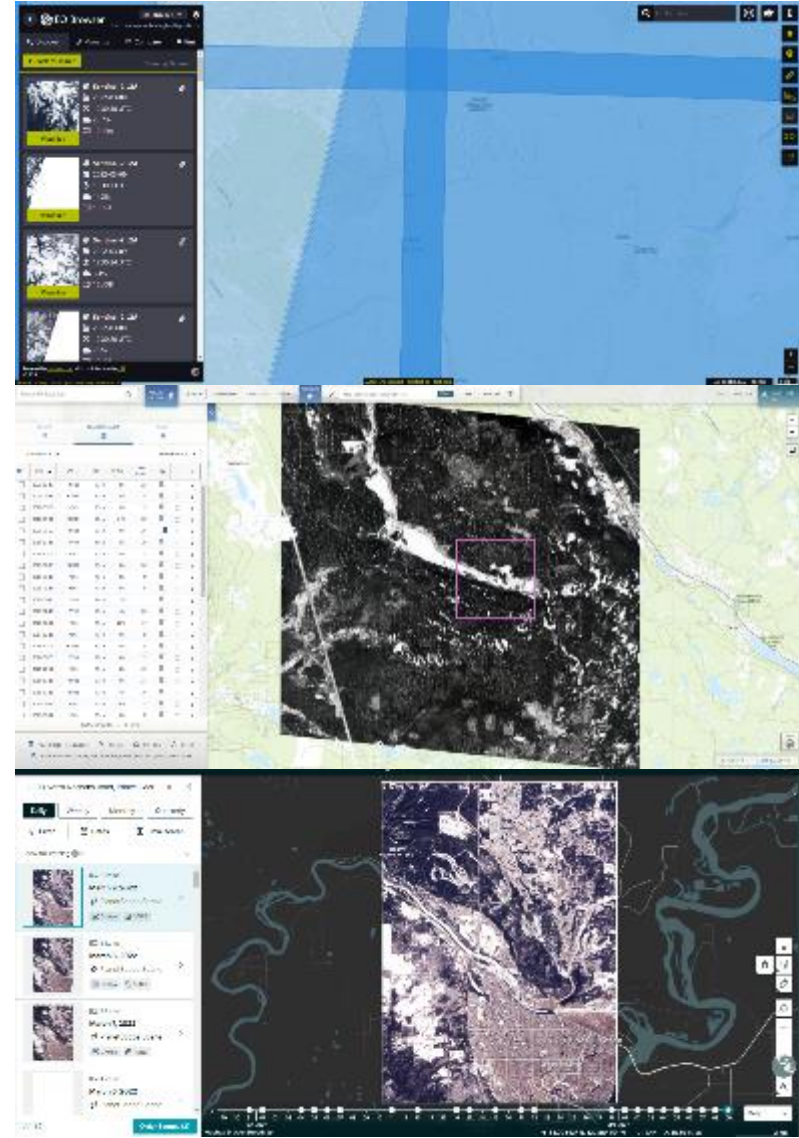
```
[ ] Map.add_cog_layer(url2, name="Fire (post-event)")
```

DEMO

https://colab.research.google.com/github/giswqs/geemap/blob/master/examples/notebooks/44_cog_stac.ipynb

Spatiotemporal Asset Catalog (STAC)

- Manage geospatial data with a single language
- Ideal for searching and managing datasets (not for processing or visualization)
- Built for the cloud using an open standard format for simple geographical features, along with their non-spatial attributes (GeoJSON)
- STAC consists of:
 - Catalogue with collections (e.g. Sentinel-2),
 - Collection has Items (e.g. Multiband image)
 - Items have assets (e.g. Single band image)

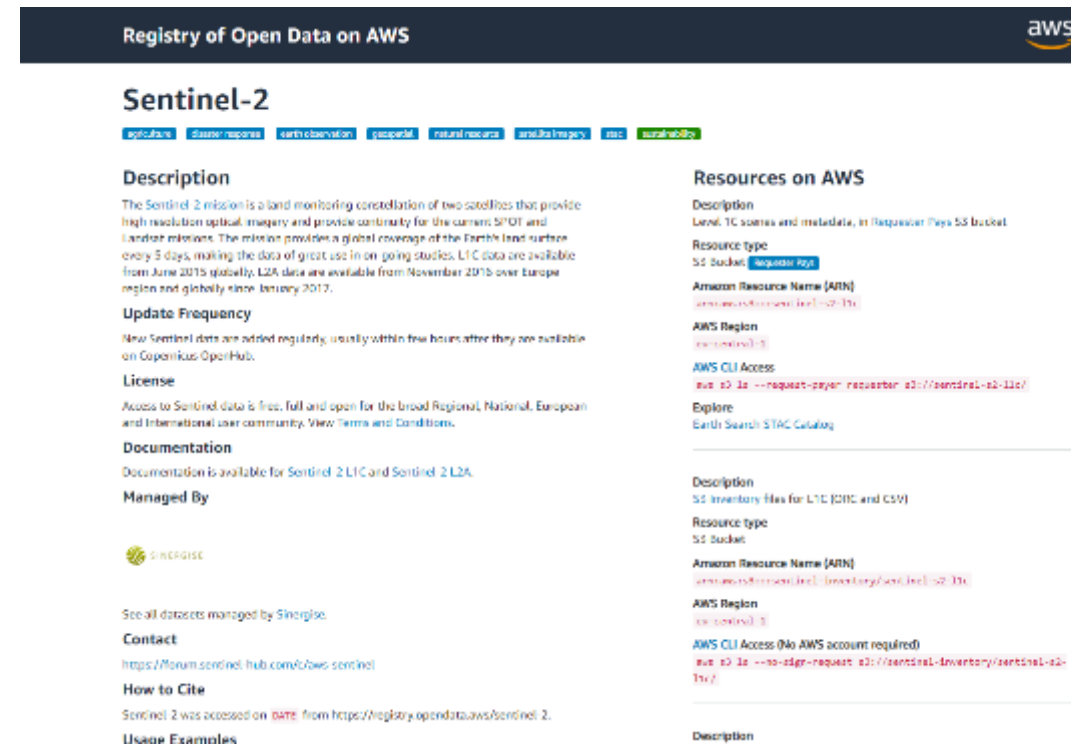




<https://stacindex.org/catalogs/planet-labs-stac-catalog#/?t=1>

Amazon Web Services (AWS)

- Common home for large geospatial datasets (e.g. Climate, Imagery, OpenStreetMap, Terrain)
- Typically data are hosted as “COGs” using “STAC”
- Access bands individually in seconds
- Can be accessed from a PC or from another cloud service
- There is a COST
- E.g.:
<https://aws.amazon.com/earth/>



The screenshot displays the 'Registry of Open Data on AWS' page for Sentinel-2. The page is organized into several sections: 'Description', 'Update Frequency', 'License', 'Documentation', 'Managed By', 'Resources on AWS', and 'Usage Examples'. The 'Description' section explains that Sentinel-2 is a land monitoring constellation of two satellites providing high-resolution optical imagery. The 'Update Frequency' section states that new data is added regularly, usually within a few hours. The 'License' section notes that access to Sentinel data is free, full, and open. The 'Documentation' section mentions that documentation is available for Sentinel-2 L1C and Sentinel-2 L2A. The 'Managed By' section identifies the data as being managed by 'GEOLOGIS'. The 'Resources on AWS' section provides details about the data's location in an S3 bucket, including the Amazon Resource Name (ARN), AWS Region, and AWS CLI Access command. The 'Usage Examples' section provides a link to the Sentinel-2 data on the AWS website.

Registry of Open Data on AWS

Sentinel-2

[Applications](#) [Dataset reports](#) [Earth observation](#) [Imagery](#) [Natural resource](#) [Satellite imagery](#) [Science](#) [Spatial data](#)

Description

The Sentinel-2 mission is a land monitoring constellation of two satellites that provide high-resolution optical imagery and provide continuity for the current SPOT and Landsat missions. The mission provides a global coverage of the Earth's land surface every 5 days, making the data of great use in on-going studies. L1C data are available from June 2015 globally. L2A data are available from November 2015 over Europe region and globally since January 2017.

Update Frequency

New Sentinel data are added regularly, usually within few hours after they are available on Copernicus OpenHub.


License

Access to Sentinel data is free, full and open for the broad Regional, National, European and International user community. View [Terms and Conditions](#).

Documentation

Documentation is available for [Sentinel-2 L1C](#) and [Sentinel-2 L2A](#).

Managed By

 [GEOLOGIS](#)

See all datasets managed by [GEOLOGIS](#).

Contact

<https://forum.sentinel-hub.com/two-sentinel>

How to Cite

Sentinel-2 was accessed on [date](#) from <https://registry.opendata.aws/sentinel-2>.

Usage Examples

Resources on AWS

Description

Level-1C scenes and metadata, in Requester Pays S3 bucket.

Resource type

S3 bucket [requester pays](#)

Amazon Resource Name (ARN)

[arn:aws:s3:::sentinel-2-l1c/](#)

AWS Region

[eu-central-1](#)

AWS CLI Access

`aws s3 ls --request-payer requester --s3://sentinel-2-l1c/`

Explore

[Earth Search STAC Catalog](#)

Description

S3 inventory files for L1C (S3C and CSV)

Resource type

S3 bucket

Amazon Resource Name (ARN)

[arn:aws:s3:::sentinel-2-l1c/inventory/sentinel-2-l1c/](#)

AWS Region

[eu-central-1](#)

AWS CLI Access (No AWS account required)

`aws s3 ls --no-sign-request s3://sentinel-inventory/sentinel-2-l1c/`

Description

Google Earth Engine

- Available via: Python (geemap), R (rgee) and JavaScript
- Will use in next lab
- Very common in research and not-for-profit remote sensing
- Free for research education and not-for-profit use, must apply for commercial license



Google Earth Engine: Planetary-scale geospatial analysis for everyone

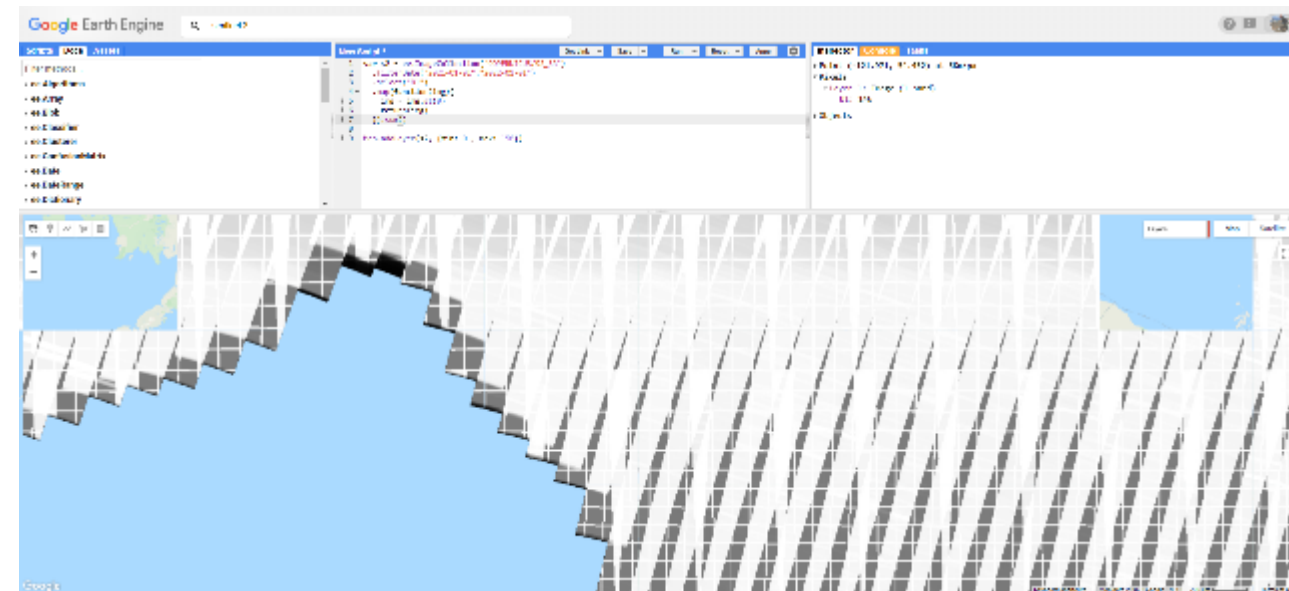
Noel Gorelick^{a,*}, Matt Hancher^b, Mike Dixon^b, Simon Ilyushchenko^b, David Thau^b, Rebecca Moore^b

^a Google Switzerland, Brändelstrasse 170, Zurich 8002, Switzerland

^b Google Inc., 1600 Amphitheater Parkway, Mountain View, CA, 94041, USA



Cited >4k times!



High-Resolution Global Maps of 21st-Century Forest Cover Change

M. C. Hansen,^{1*} P. V. Potapov,¹ R. Moore,² M. Hancher,² S. A. Turubanova,¹ A. Tyukavina,¹ D. Thau,² S. V. Stehman,³ S. J. Goetz,⁴ T. R. Loveland,⁵ A. Kommareddy,⁶ A. Egorov,⁶ L. Chini,¹ C. O. Justice,¹ J. R. G. Townshend¹

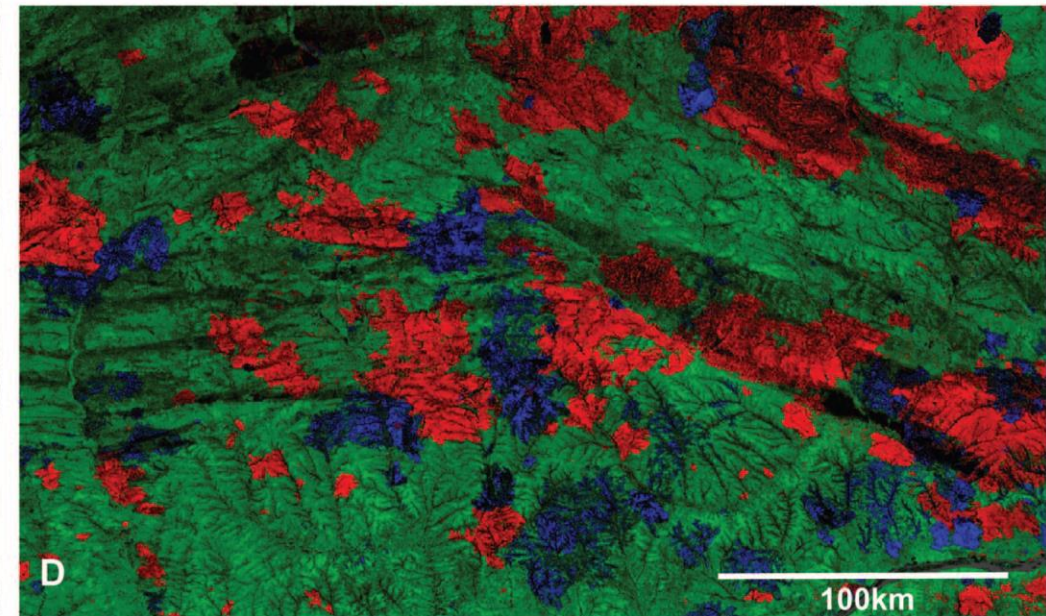
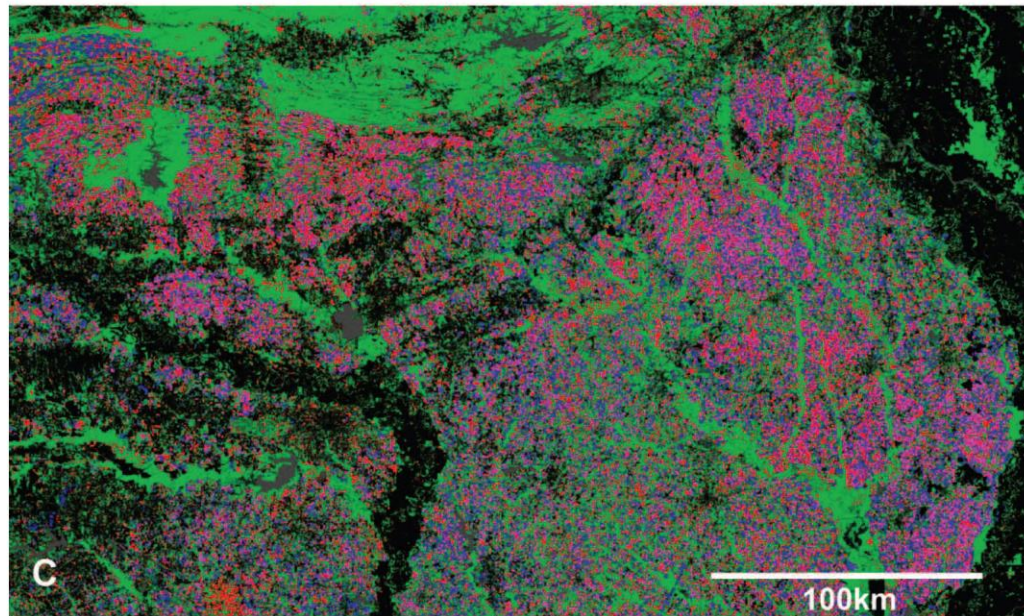
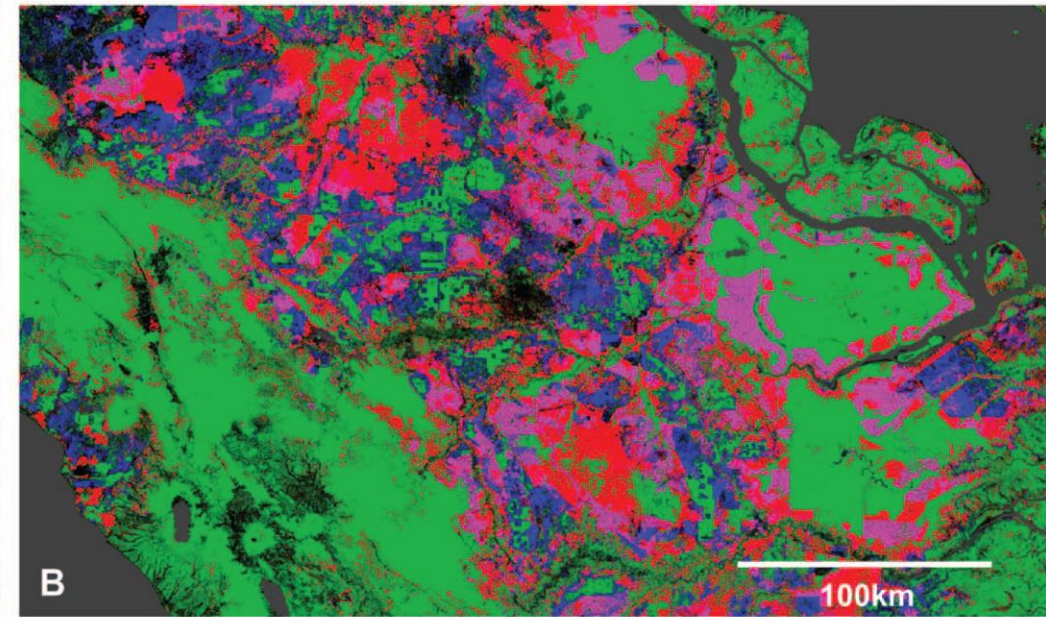
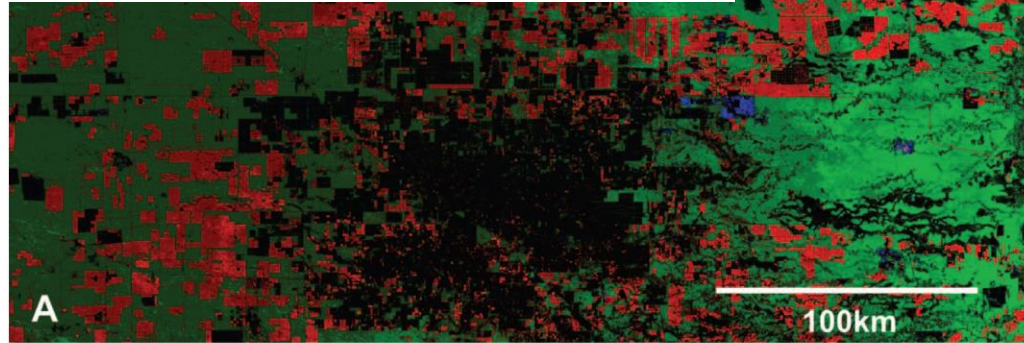
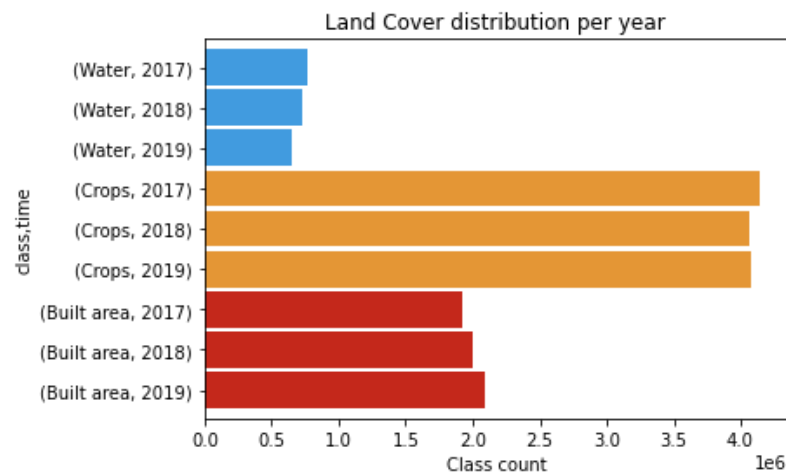
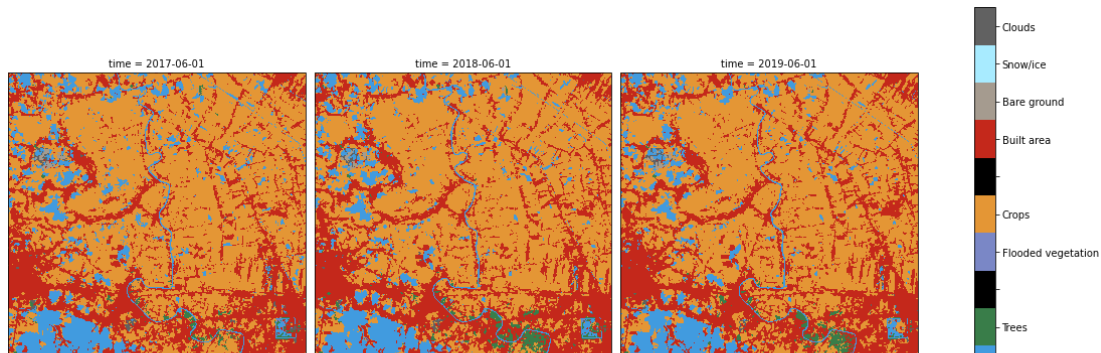


Fig. 2. Regional subsets of 2000 tree cover and 2000 to 2012 forest loss and gain. (A) Paraguay, centered at 21.9°S, 59.8°W; **(B)** Indonesia, centered at 0.4°S, 101.5°E; **(C)** the United States, centered at 33.8°N, 93.3°W; and **(D)** Russia, centered at 62.1°N, 123.4°E.

Cited >8k times!

Microsoft Planetary Computer

The Planetary Computer combines a multi-petabyte catalog of global environmental data with intuitive APIs, a flexible scientific environment that allows users to answer global questions about that data, and applications that put those answers in the hands of conservation stakeholders.



Choose your environment



CPU - Python

4 cores, 32 GiB of memory. [Pangeo Notebook environment](#).



R

8 cores, 64 GiB of memory. [R geospatial environment](#).



GPU - PyTorch

4 cores, 28 GiB of memory, [T4 GPU](#). This has a longer startup time.



GPU - Tensorflow

4 cores, 28 GiB of memory, [T4 GPU](#). This has a longer startup time.



QGIS (preview)

4 cores, 32 GiB of memory. [QGIS geospatial environment](#). Currently in preview mode. [Contact us](#) with feedback.

Start

Conclusion

- Tricks exists to make local processing more efficient
 - More cores, more RAM, better GPU
 - Tiling large datasets
 - For visualization: 8-bit compression, VRTs, Overviews
 - Parallelization processing
- Cloud infrastructure keeps data close to processing
 - COGS/STAC to query catalogues and grab only the data you need
 - Some free/cheap options for cloud computing
 - Commercial solutions can be very efficient but can also be costly