# Cloud based biophysical trait map retrieval: The **PyEOGPR** python package

**Dávid D.Kovács**[1] **Jochem Verrelst**[1]

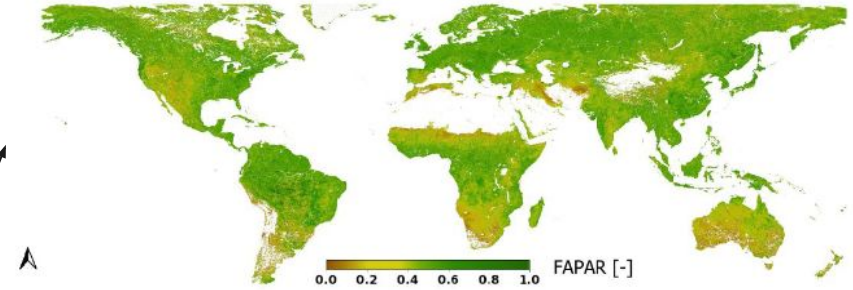
*1 Image Processing Laboratory (IPL), University of Valencia, Spain*

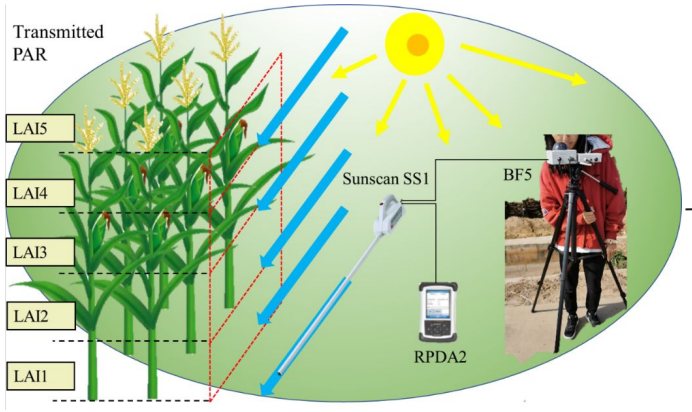
VNIVERSITAT DE VALÈNCIA

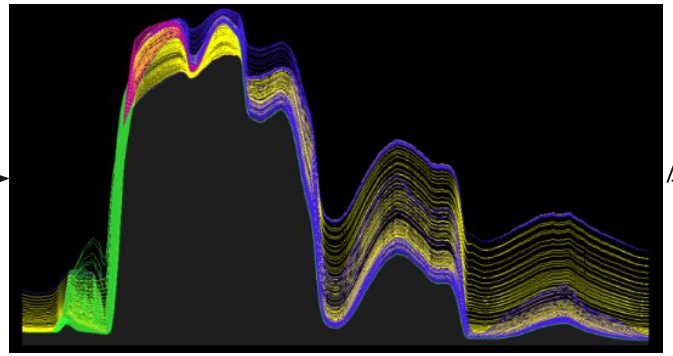
flexinel

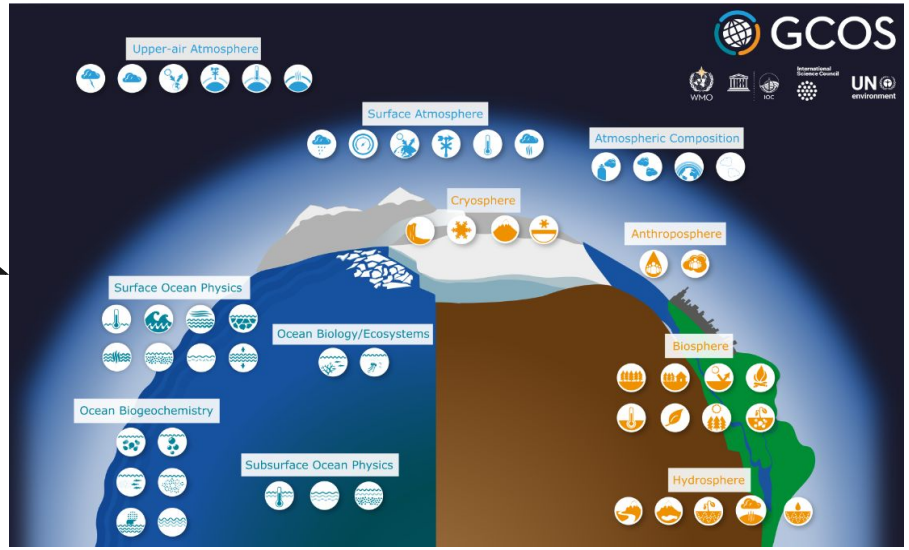
# Motivation

- Biophysical variable mapping is useful in climate studies, precision agriculture, vegetation monitoring
- What are biophysical variables?
  - Physical, chemical or biological variables critically contributes to the characterization of Earth's climate.
  - The need to be **measured locally**, can't be calculated like NDVI
  - vegetation biophysical variables have unique spectral signatures
- Data driven methods to obtain → machine learning methods.





# Motivation

- Biophysical variable mapping is useful in climate studies, precision agriculture, vegetation monitoring
  **However!**
- Local processing of satellite imagery: **complex computation and memory requirements Complicated**

  Solution:
- Machine learning and cloud computing **Run algorithms on EO data on a large remote cluster with PyEOGPR**

Receive **resulting vegetation trait maps**
Universal **netCDF** or **geoTIFF** format

✔ **No local processing needed**
✔ **Can be run from anywhere**
✔ **Can be run with minimal IT knowledge**
✔ **Ready-to-use results**

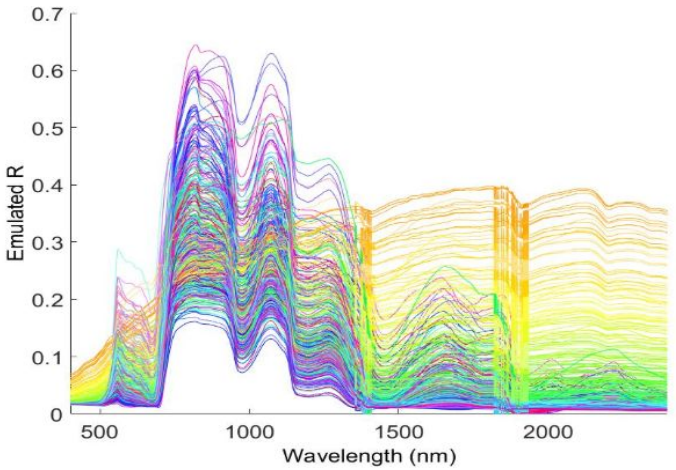
# Background

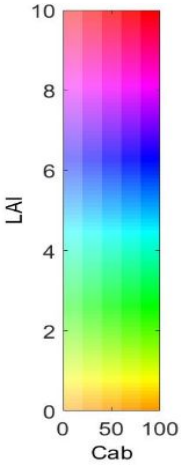
- Different vegetation traits have various **spectral signatures,** captured by satellite sensors
- Radiative Transfer Models in ARTMO simulate these spectra
- Machine learning algorithms **learn** the connections between **spectral signatures** and **plant traits**

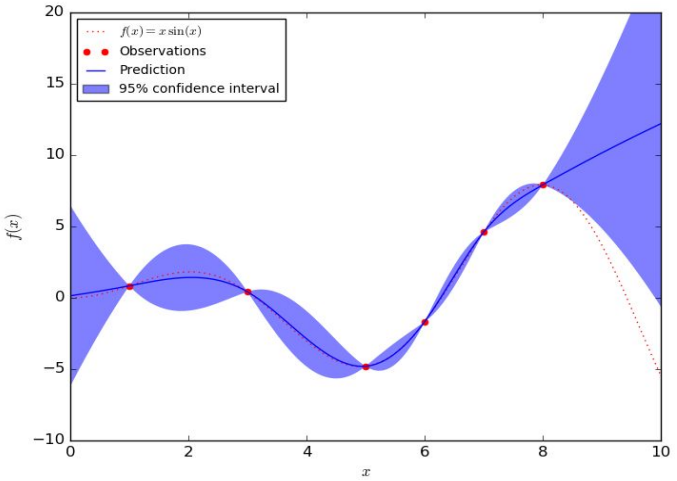
Hybrid models

Combining: **Machine Learning** with **RTMs**

Specifically: Gaussian Process regression (GPR) trained on RTM simulations from ARTMO



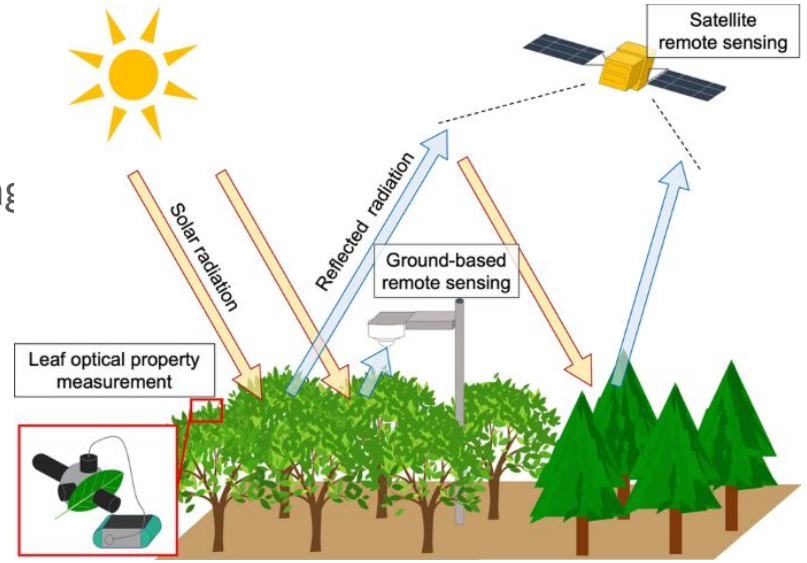



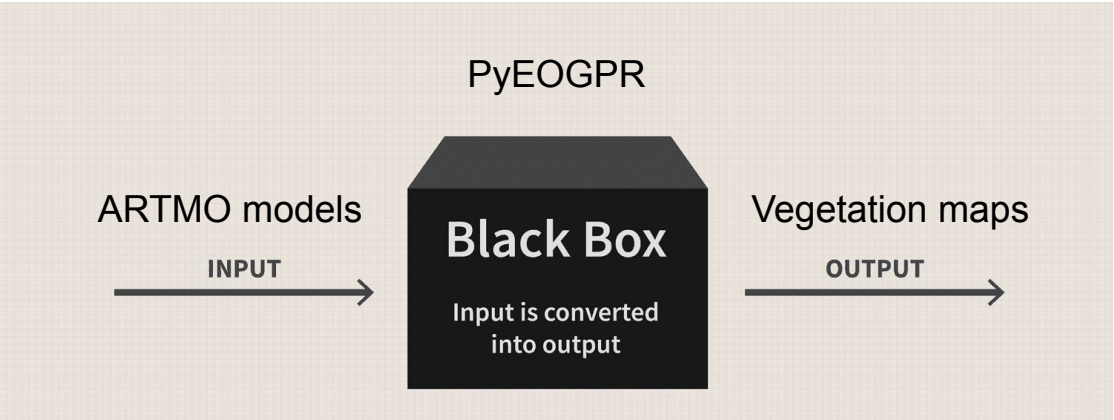
# Methodology

- Hybrid GPR models yielded from ARTMO
  - User can upload custom trained *.py* files
  - ..or choose from currently implemented 26 models
  - Leaf area index, fraction of absorbed photosynthetically active radiation, mang more!
  - Variable uncertainties from GPR *(under construction)*

- Satellite data from a multitude of missions:
  - Sentinel 2 MSI, Sentinel 3 OLCI and Landsat 8-9

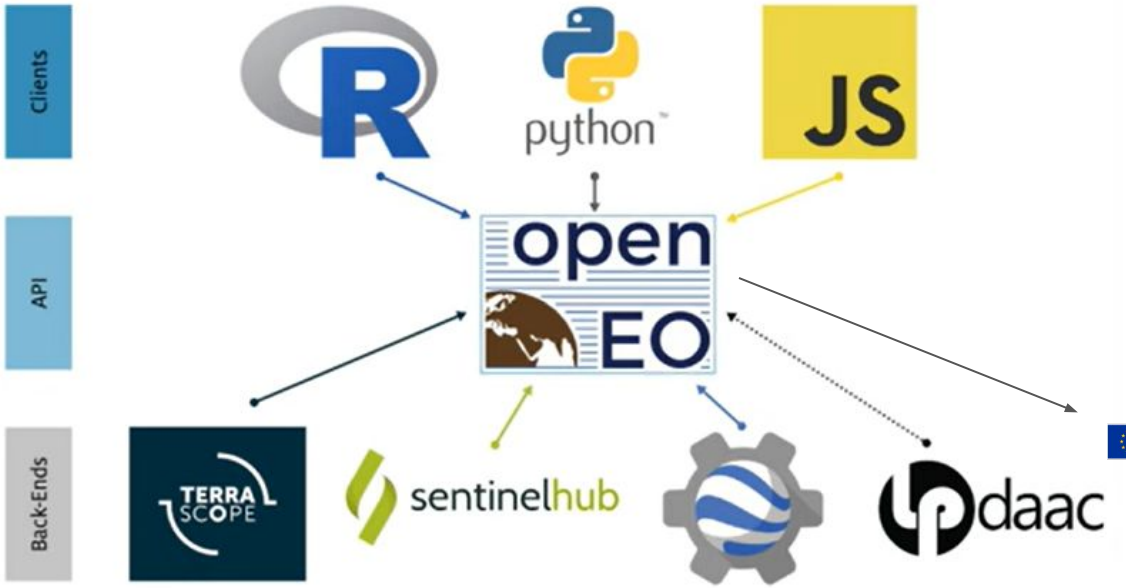



- PyEOGPR wraps the complex matrix multiplication of GPR
- Users don't need data science knowledge, „black box" structure
- openEO based back-end complex matrix multiplications



# Short tutorial

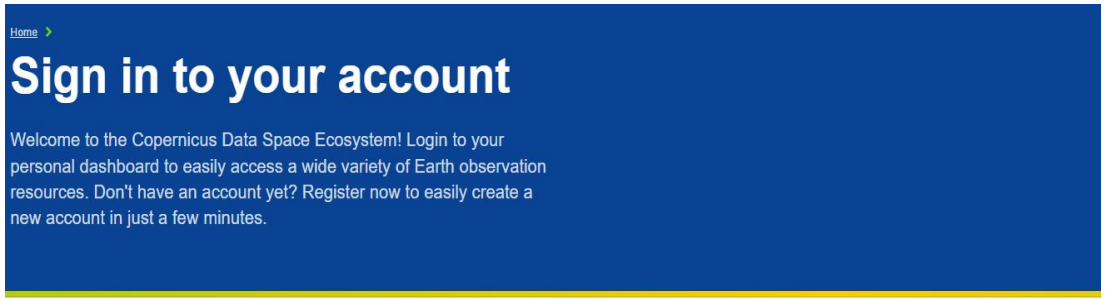
1. Free to use, registration to Copernicus Data Space Ecosystem required
2. Open **any** Python IDE!
3. *pip install pyeogpr*
4. And you are ready to go!

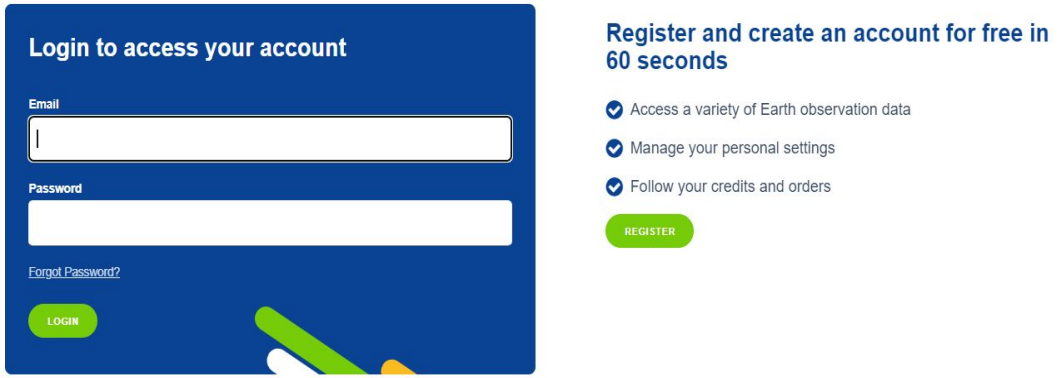



PROGRAMME OF THE EUROPEAN UNION

opernicus
Europe's eyes on Earth

esa




**https://pypi.org/project/pyeogpr/**

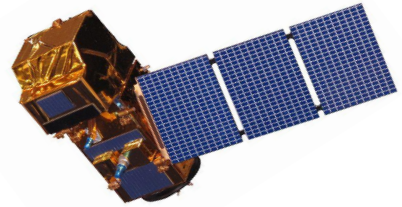
# Short tutorial

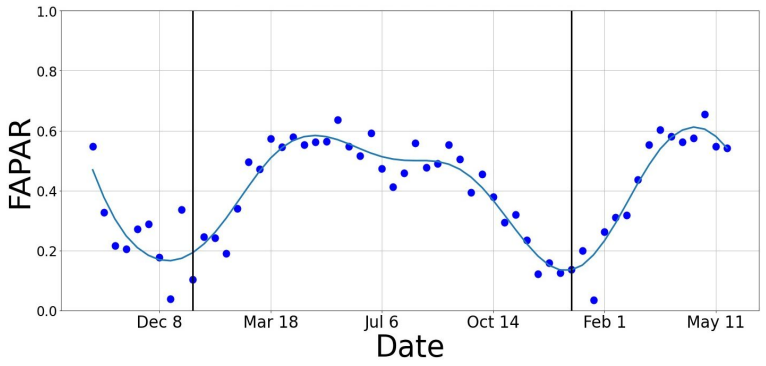
How to use the software?

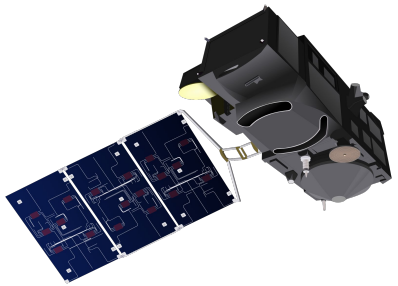
- User inputs:
  - Where?
    - bounding box *(e.g. geoJSON)*
  - When?
    - timespan *(e.g. 2020-05-01, 2020-08-01)*
  - What?
    - Variable: *e.g. LAI or FAPAR*
  - How, which mission?
    - Satellite sensor *e.g. Sentinel 2 MSI Level 2 C*

- Sentinel 2 Cloud flag
- Savitzky-Golay gap filling also implemented!

```
bounding_box = [
        -73.98605881463239,
         40.763066527718536,
        -73.94617017216025,
         40.80083669627726
      ]
```



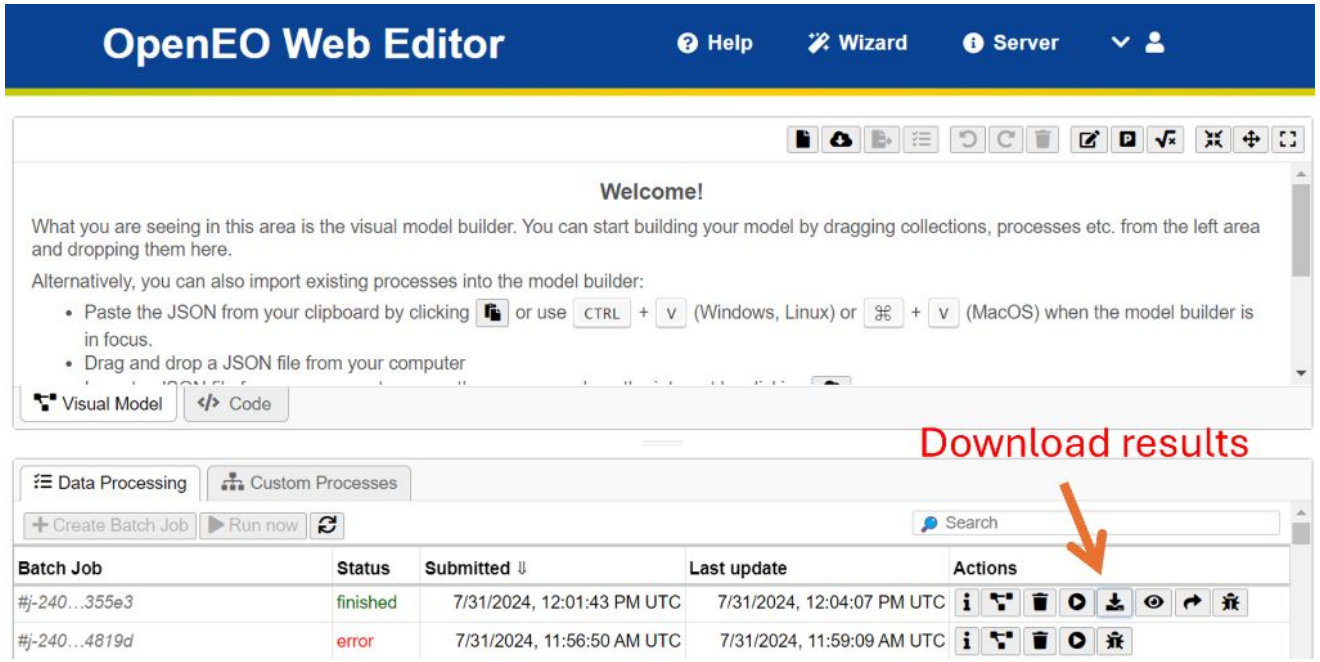


# Code snippet

```python
import pyeogpr

# Your region of interest
bounding_box = [
          -0.305543150556133,
          39.29253033906926,
          -0.28169853763617425,
          39.303338211248104
        ]

# Time window for processing Satellite observations
time_window = ["2022-05-01", "2022-06-01"]

dc = pyeogpr.Datacube(
    "SENTINEL2_L2A",  # Satellite sensor
    "FVC",            # Fractional Vegetation Cover
    bounding_box,
    time_window,
    cloudmask=True
)

dc.construct_datacube("dekad")  # Initiates openEO datacube

dc.process_map()  # Starts GPR processing
```

https://openeo.dataspace.copernicus.eu/



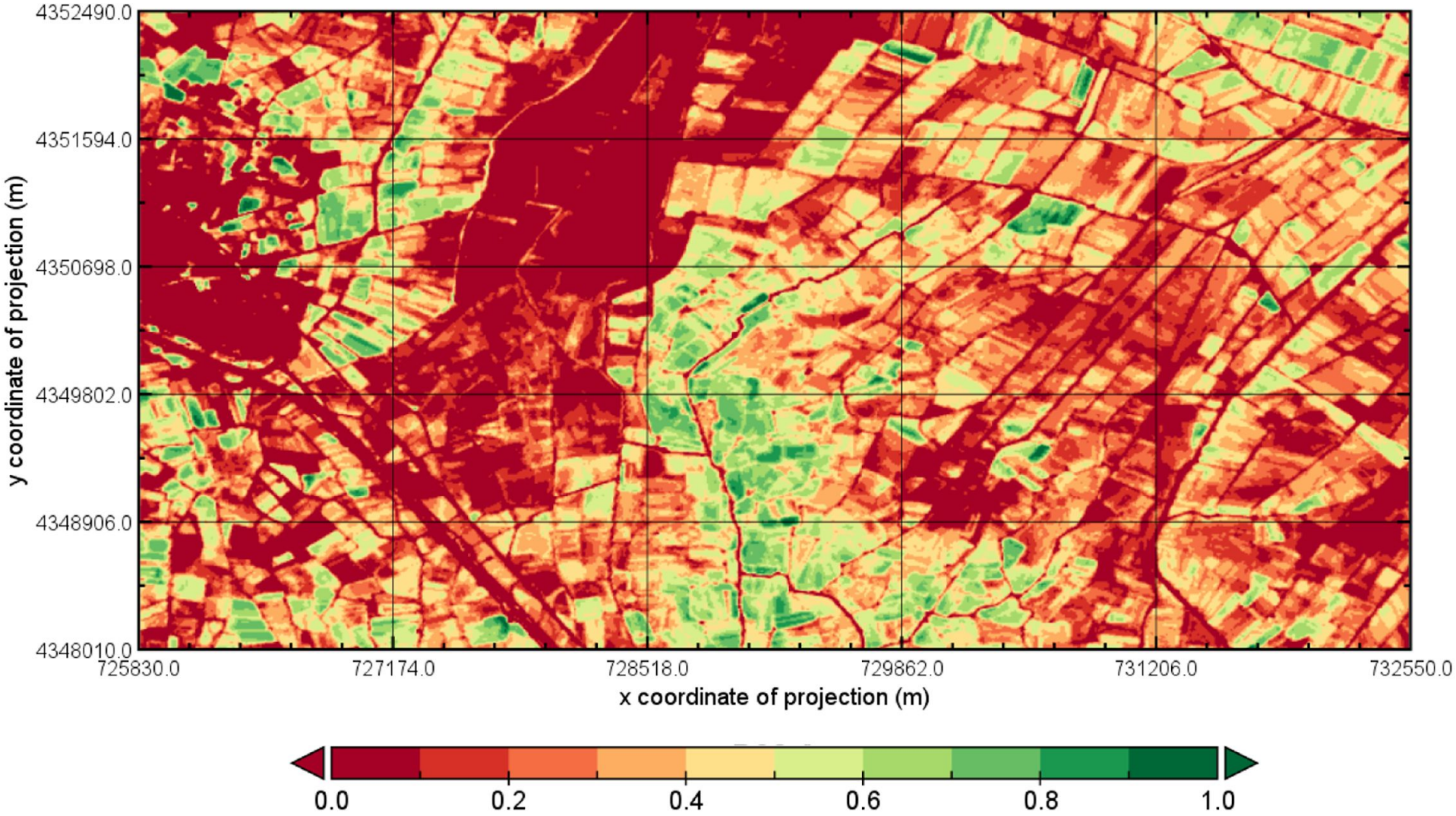

Download results

Sends process to back-end

Visualize results in e.g. QGIS or Panoply

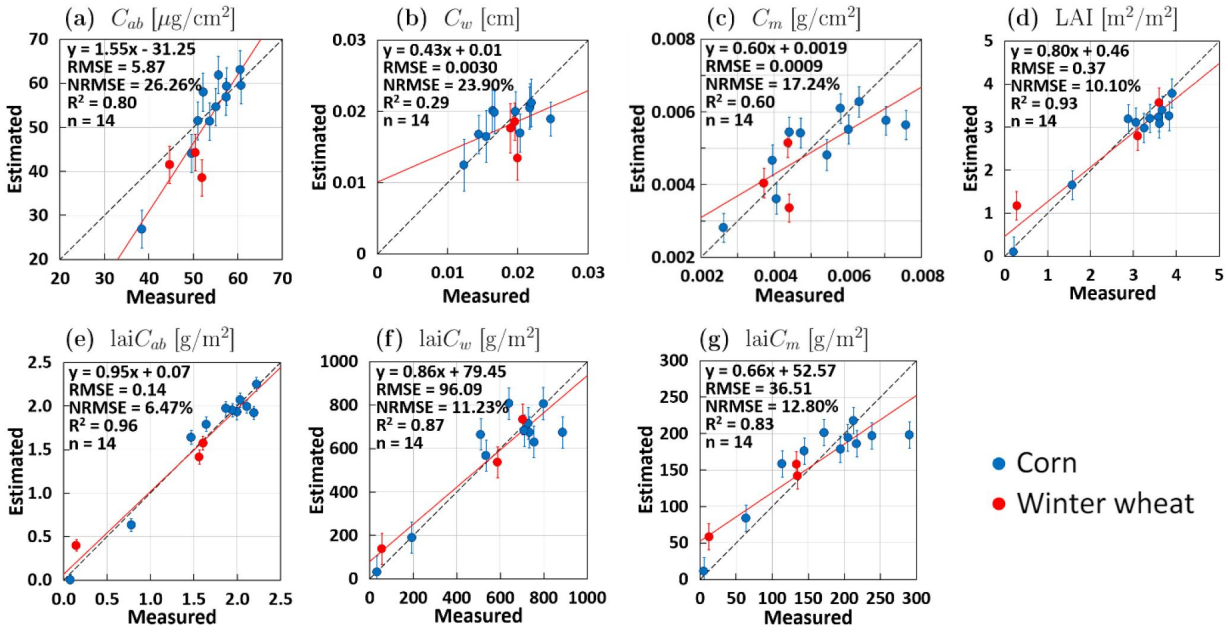
# Resulting map: Multi-temporal *netcdf or tiff*

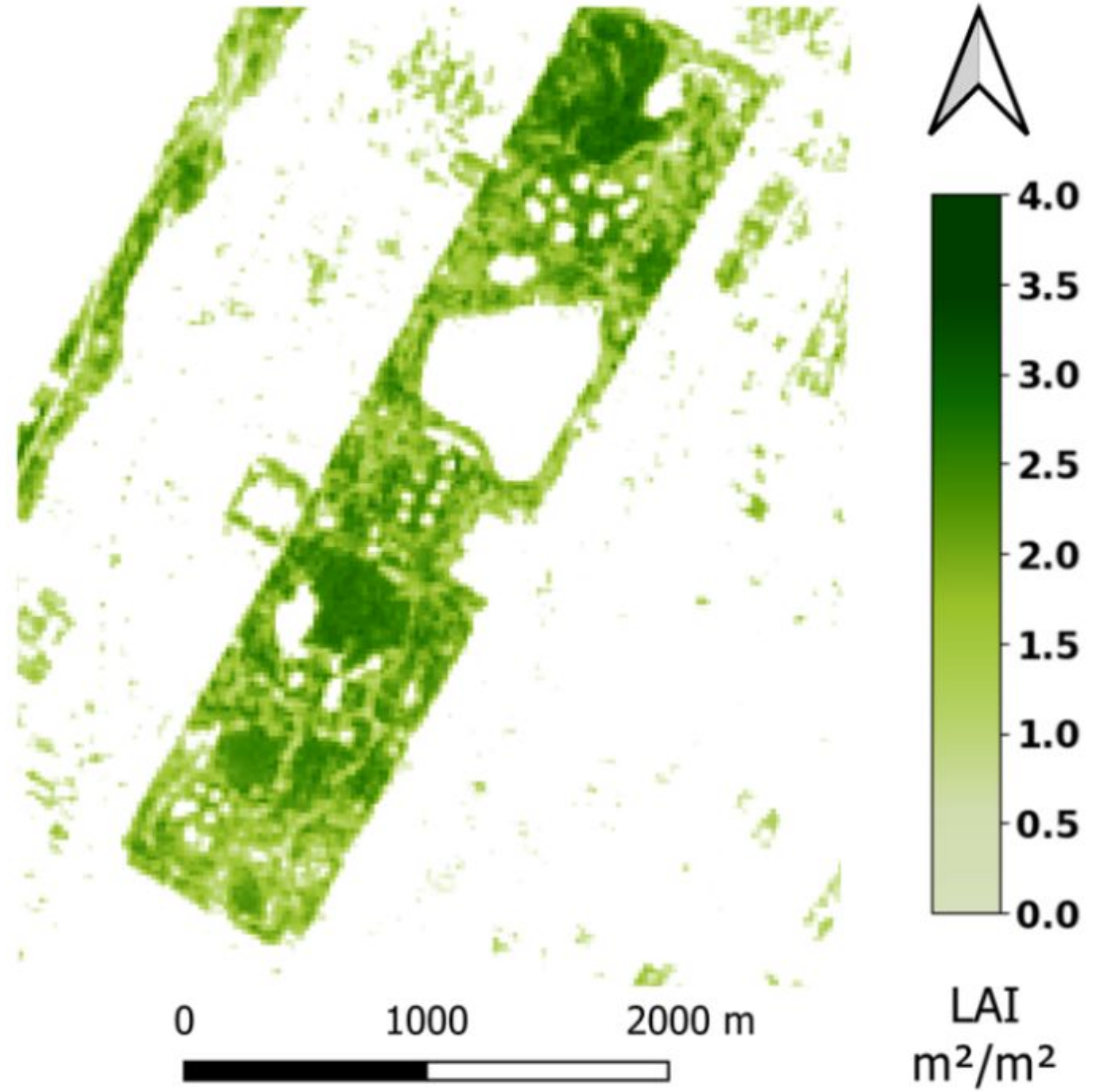
Sentinel 2 Level 2A - Fractional Vegetation Cover



# Sentinel 2 L2C based LAI map in NY Central Park

GPR processing



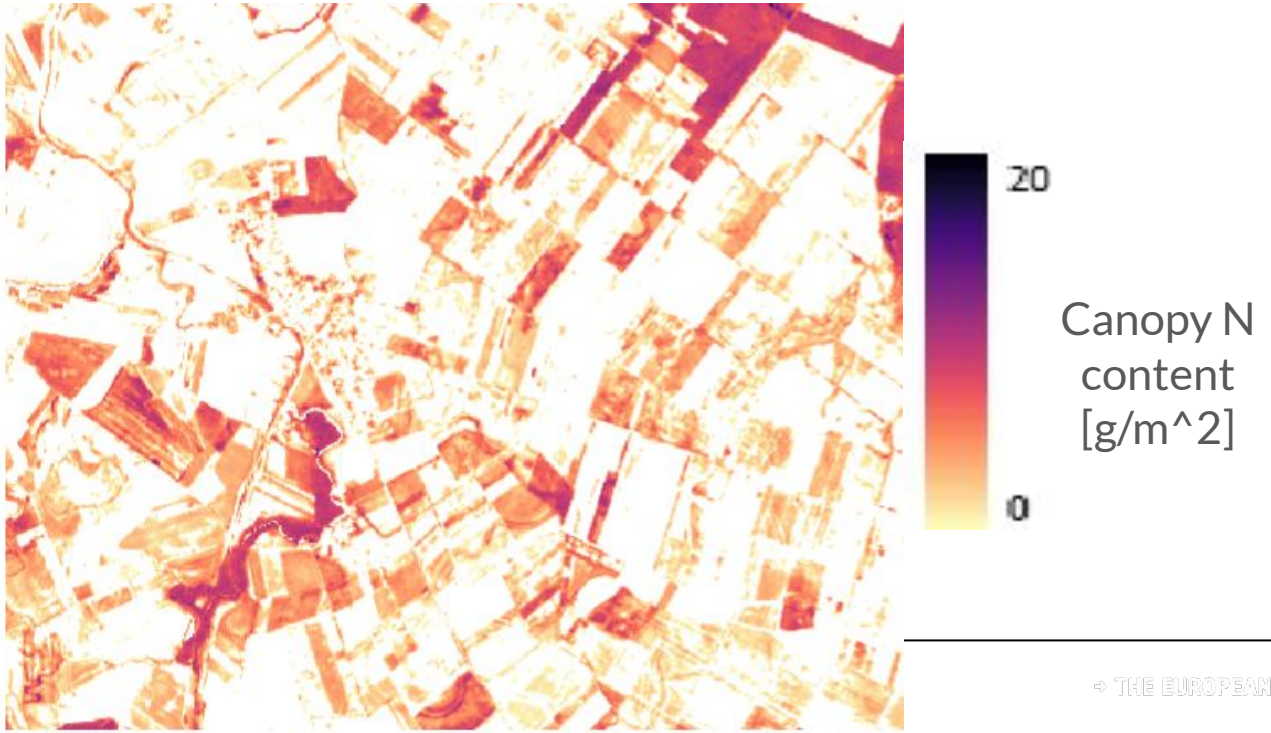
# User based model

- User can **train their own model** by using the ARTMO software package
- **Hyperspectral imagery** with PCA step will be included (for CHIME, FLEX)
- Only needed to add path of model in the argument

```
25
26 dc.process_map(gapfill=True, own_model ="C:\Users\Username\custom_ARTMO_model.py")
```

*Canopy Nitrogen content map Hungarian agricultural field*

*User defined model, imported from own script*



# Access

- GitHub – Feel free to fork the repo and collaborate
- openEO algorithm plaza
- ARTMO - needed for custom model creation
  - www.artmotoolbox.com

www.github.com/daviddkovacs/pyeogpr

# Conclusions

-openEO is a great alternative to Google Earth Engine, moreover it is open source and Python based

-PyEOGPR ultimately facilitates biophysical product retrieval

-Bypassing difficult ML coding, computation and data storage

-Community driven, free to use, user defined models

-Brings EO data to the everyday user

www.github.com/daviddkovacs/pyeogpr

Thank you for your attention!

david.kovacs@uv.es