European Commission

The European Commission's science and knowledge service
Joint Research Centre

From Jupyter notebooks to web dashboards for big geospatial data analysis

Davide De Marchi, Armin Burger, Pierre Soille

Joint Research Centre

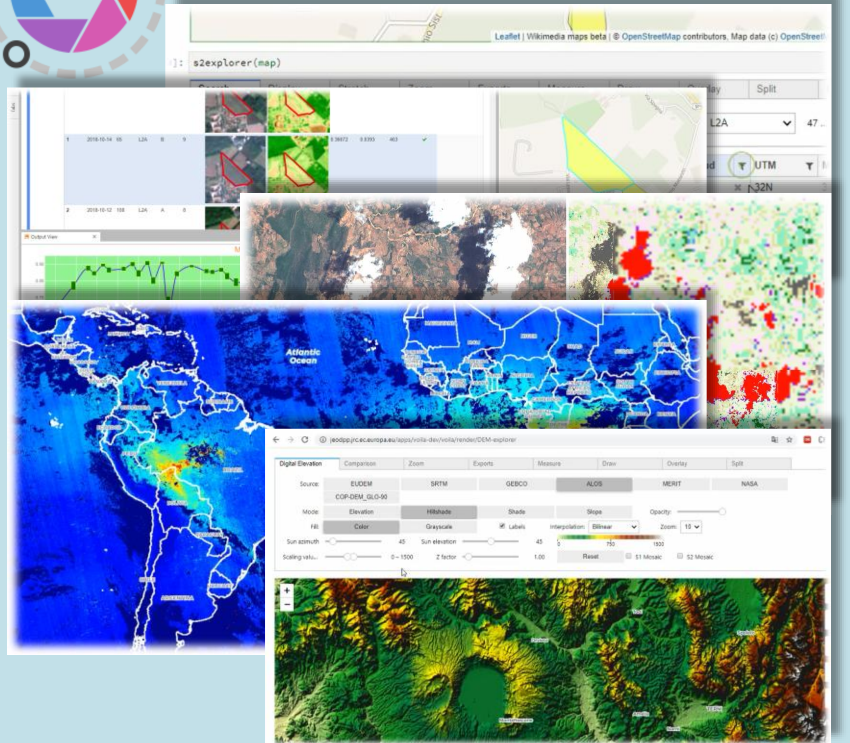
CS3 2020 Copenhagen

Summary:

European Commission

JEODPP

- **JEODPP big data platform**
- **Jupyter notebook GUI applications**
- **From interactive to batch processing and return**
- **Classification and Machine Learning**
- **New datasets and services**
- **Voila’ dashboarding**

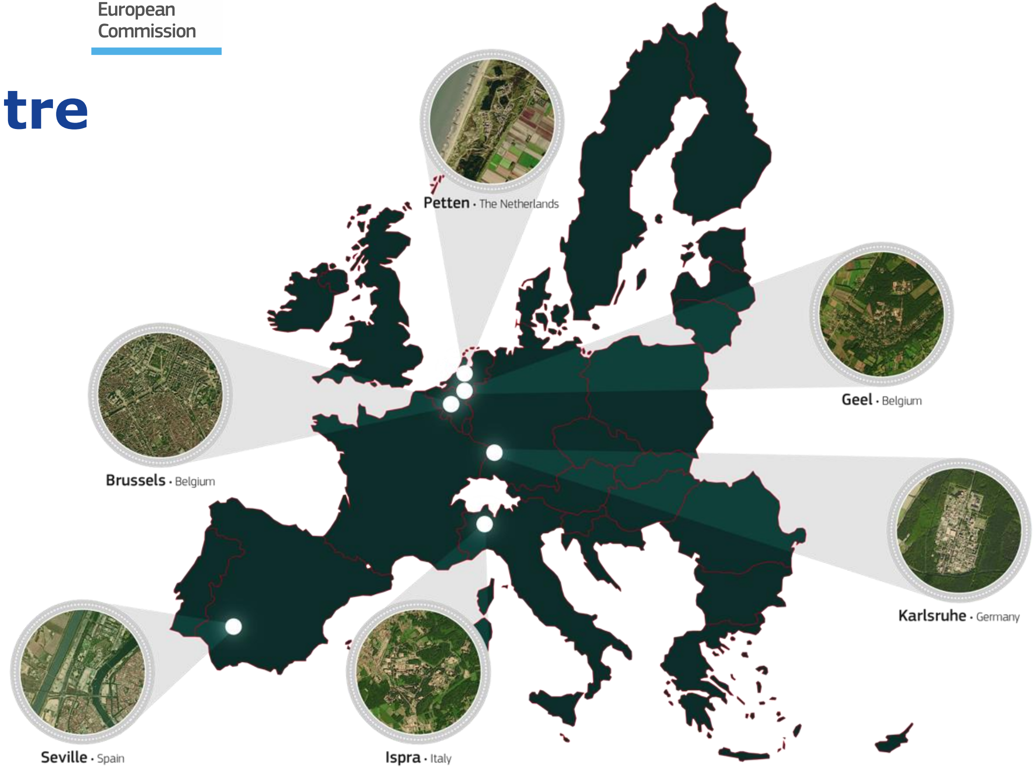


Joint Research Centre

The Joint Research Centre at a glance

3000 staff

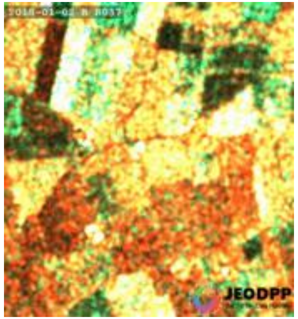
Almost 75% are scientists and researchers.
Headquarters in Brussels and research facilities located in 5 Member States.



JEODPP big data platform

With the **Copernicus Programme**, Earth Observation truly enters the big data era (currently **25 TB/day** or **10 PB/year** of full, free, and open data).

Copernicus Sentinel satellites deliver dense time series:

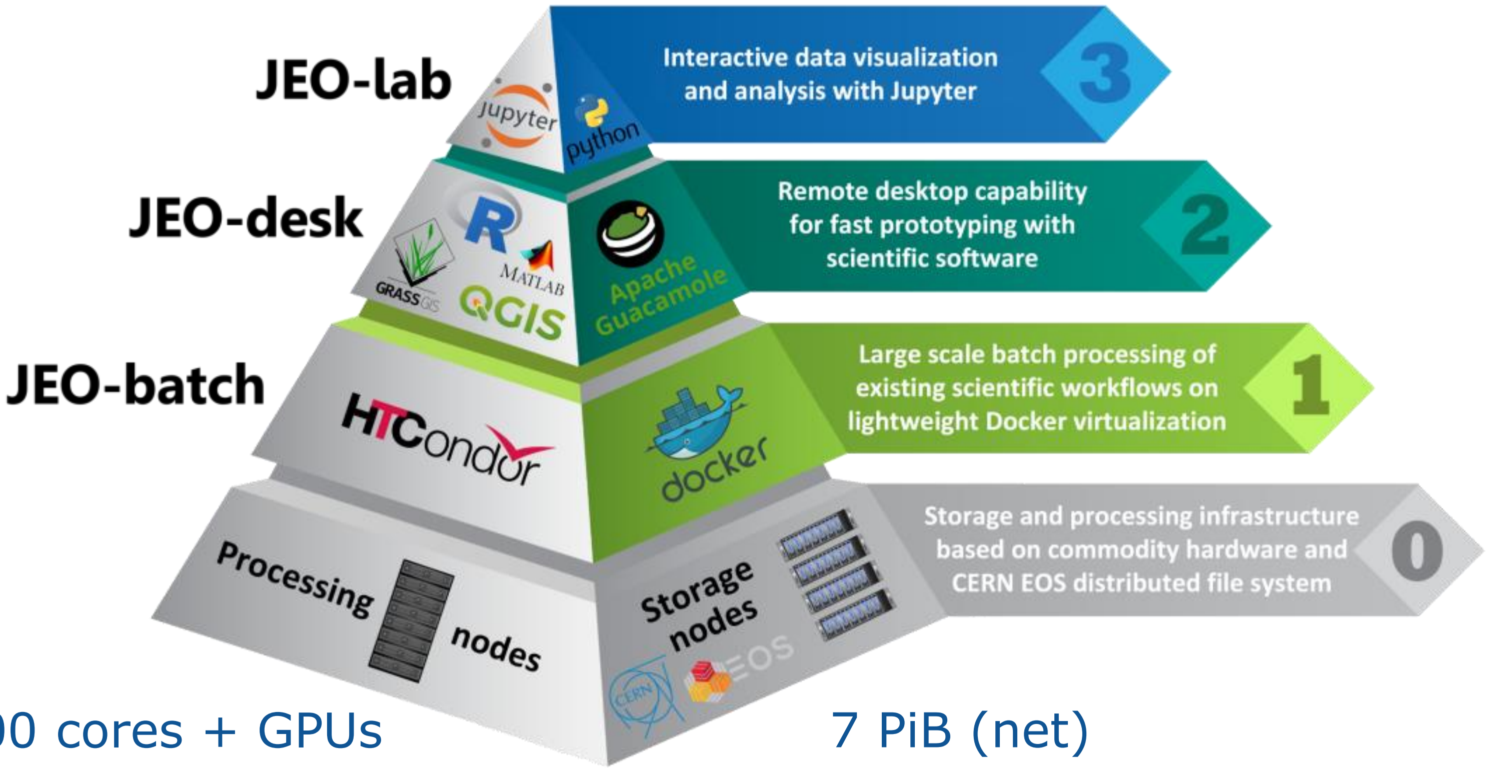


Need for a platform where EO data can be stored, processed, analyzed and maintained

Big Data Analytics project and **JEODPP platform** for JRC and EC

JEODPP big data platform

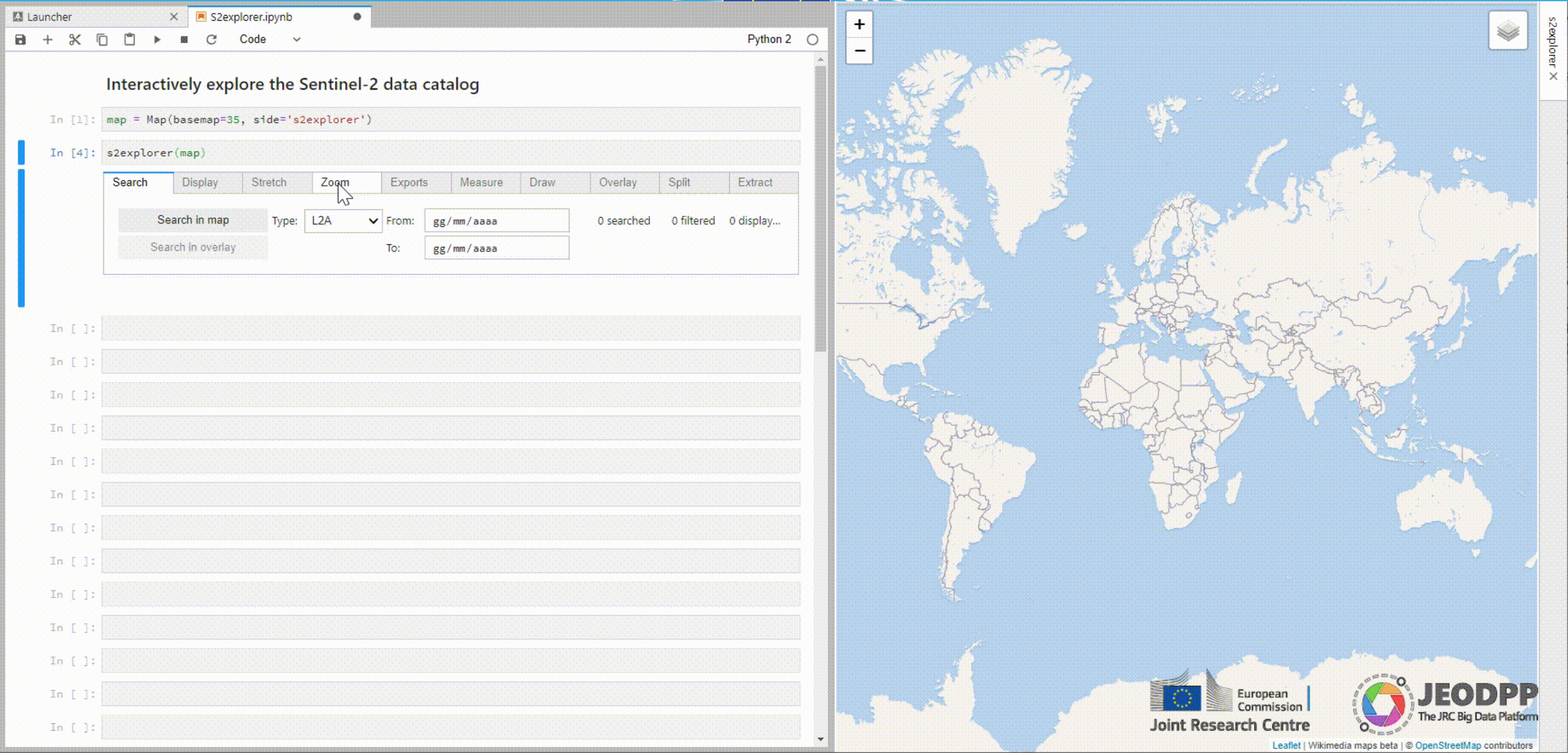
DOI:10.1016/j.future.2017.11.007



2,000 cores + GPUs

7 PiB (net)

Jupyter notebook GUI applications

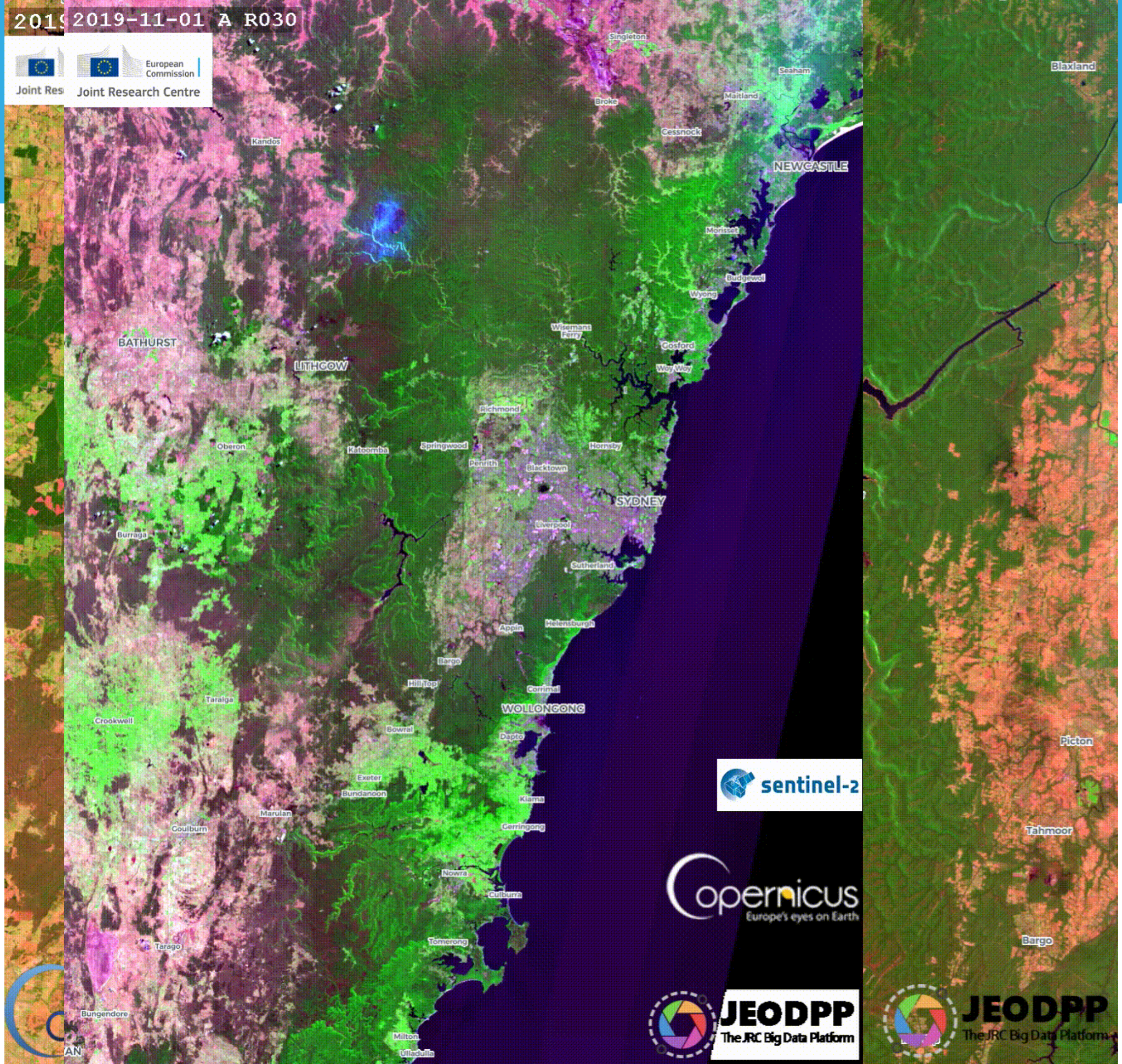


Jupyter notebook GUI applications

S2explorer:

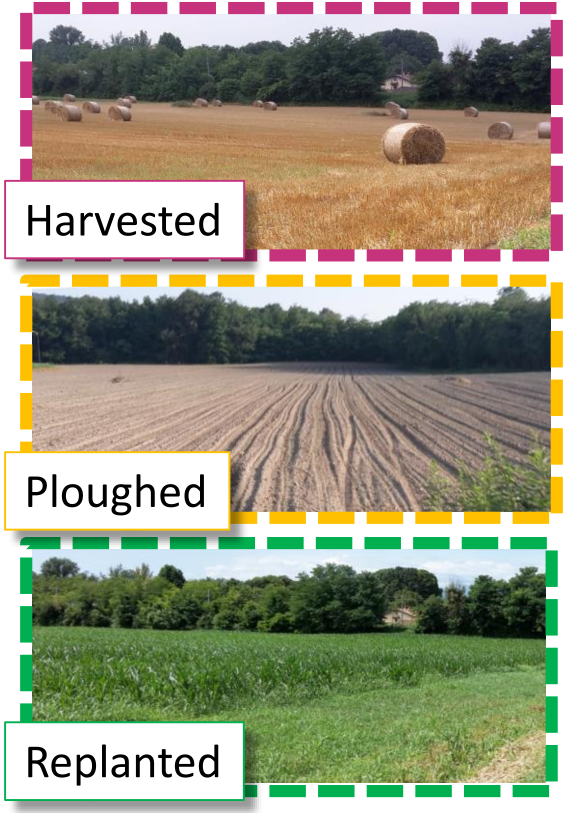
Create and export multi-temporal videos

Examples on the recent Australia fires

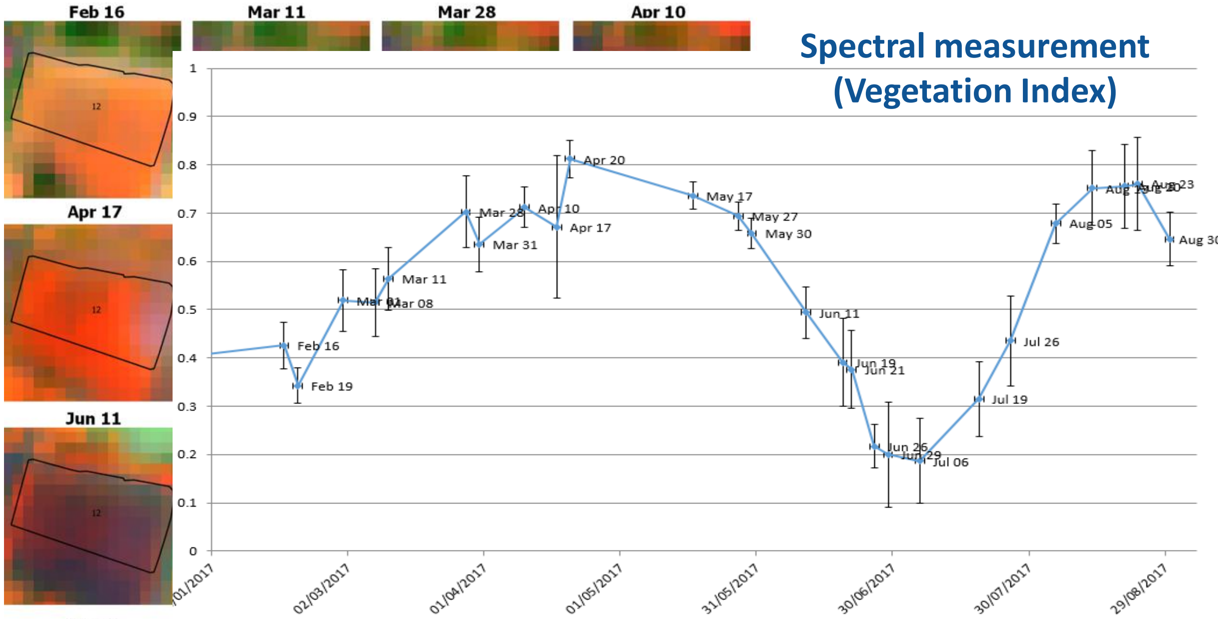


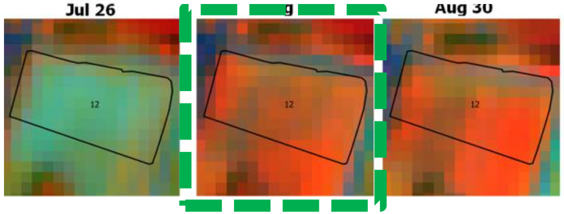
Jupyter notebook GUI applications

CAP: agriculture use case



Spectral measurement (Vegetation Index)





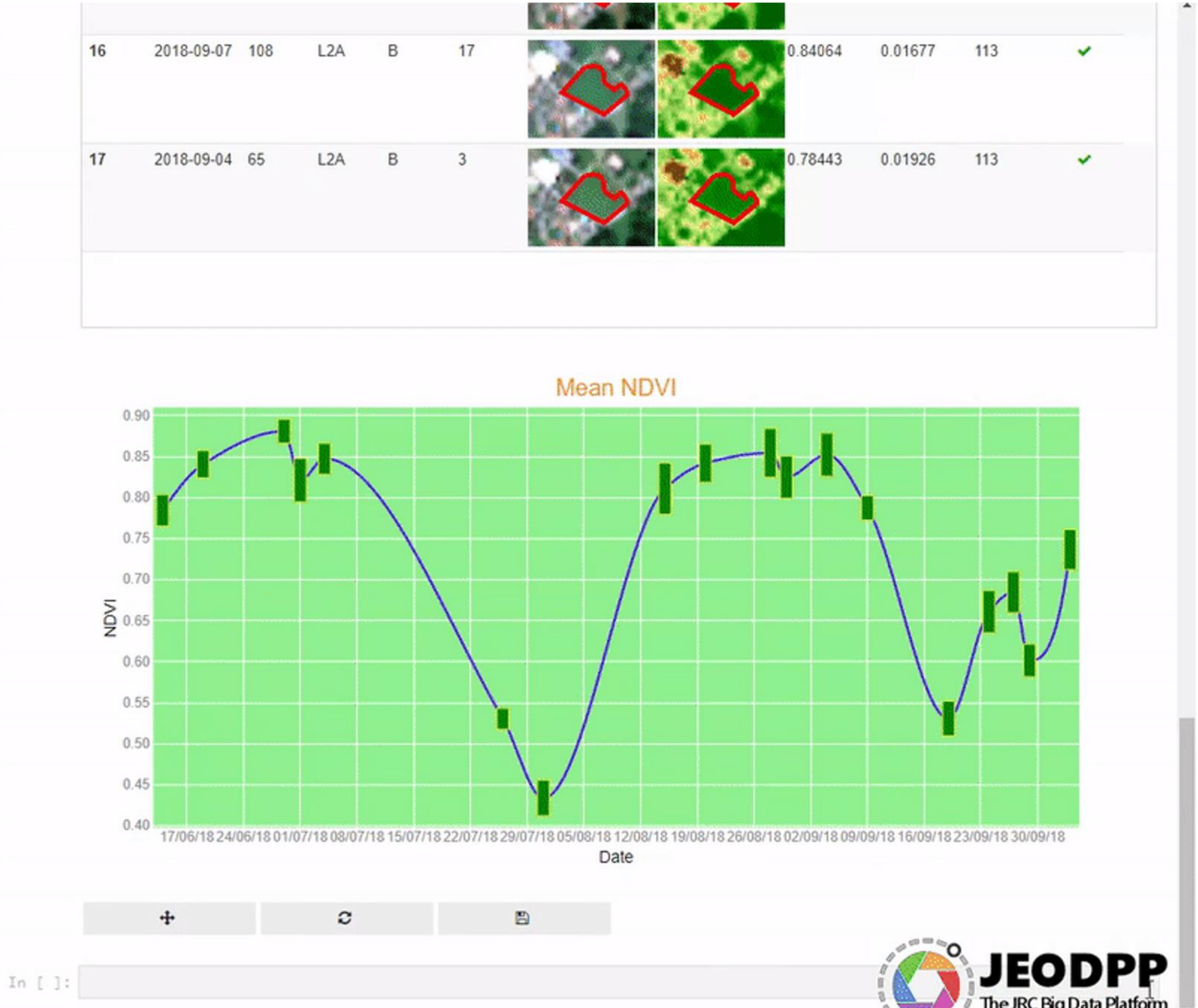
Following crop condition and calendar from satellite

JRC D.5 - Food Security, GTCAP (Guidance and Tools for the CAP)

Jupyter notebook GUI applications

S2explorer:

Added extraction features to automatically create the vegetation profile on a polygon parcel

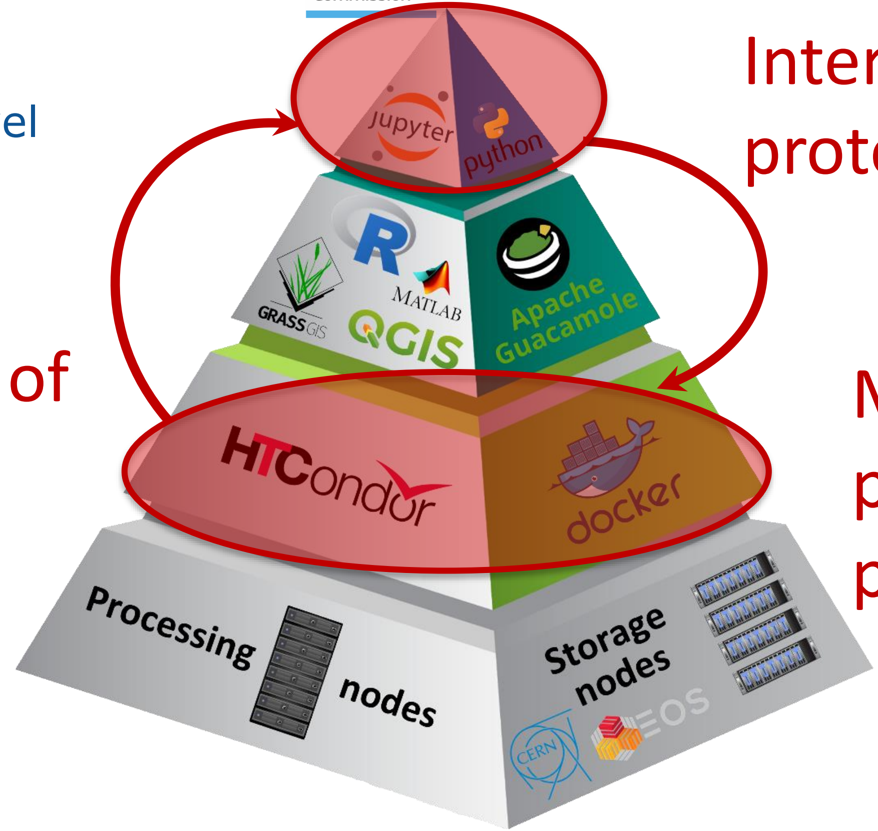


JEODPP
The JRC Big Data Platform

From interactive to batch processing and return

Extend the vegetation profiles calculation at regional or national level

Interactive evaluation of batch processing



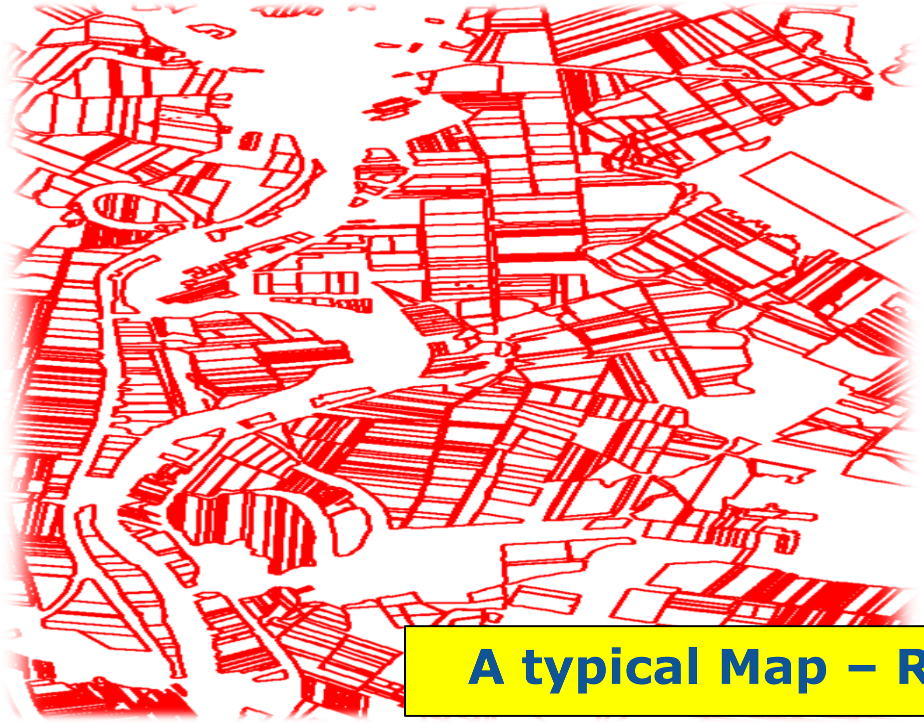
Interactive prototyping

Massive parallel processing

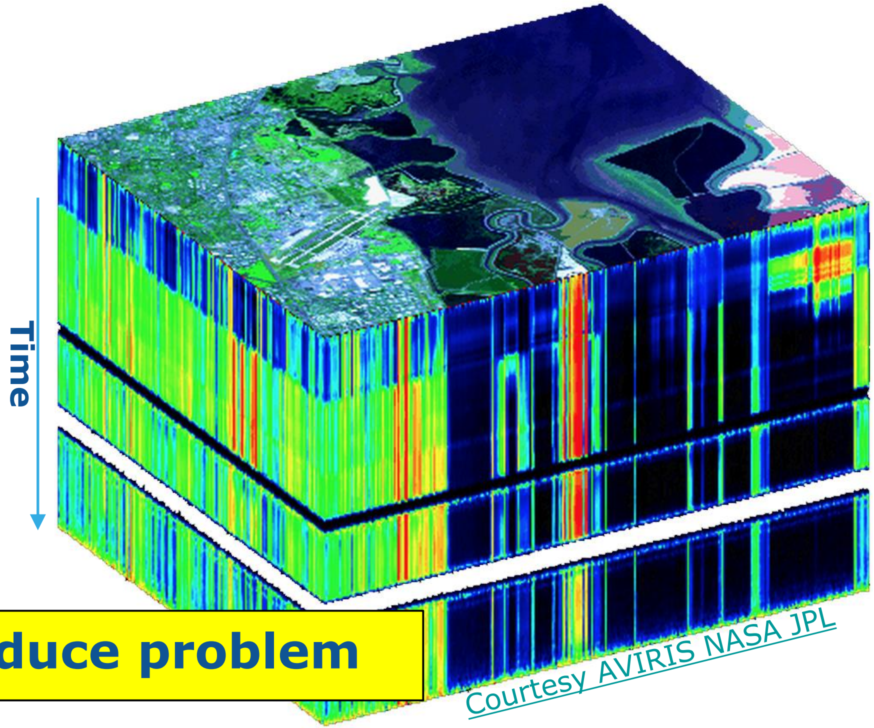
From interactive to batch services and return

European Commission

Many thousands of polygons



Hundreds of acquired images



A typical Map – Reduce problem

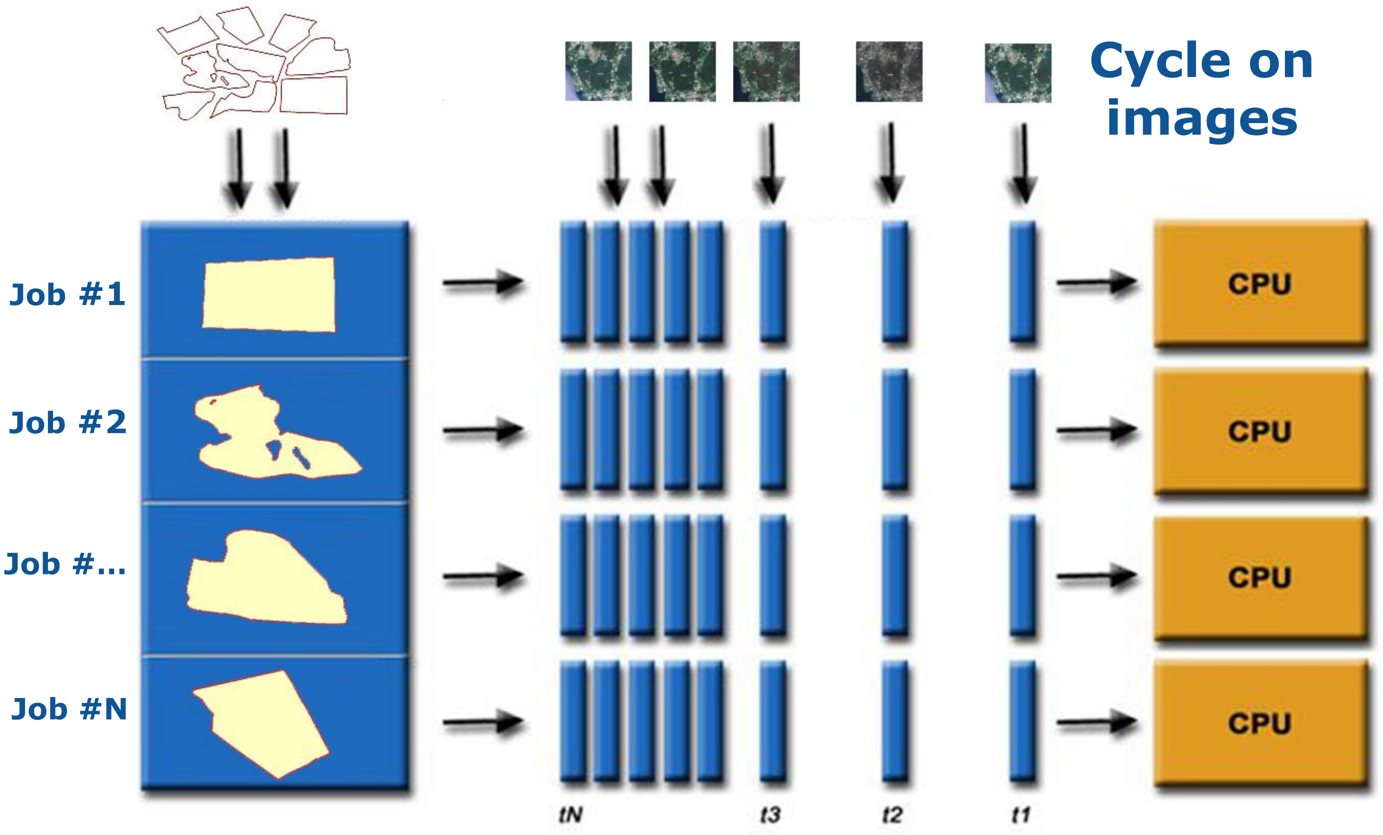
From interactive to batch services and return

Parallelize on polygons

Each image is read at the same time by many parallel jobs

GDAL library requests many stat() calls for each access to JP2000 images

EOS metadata server receives > 100K stat() request per second



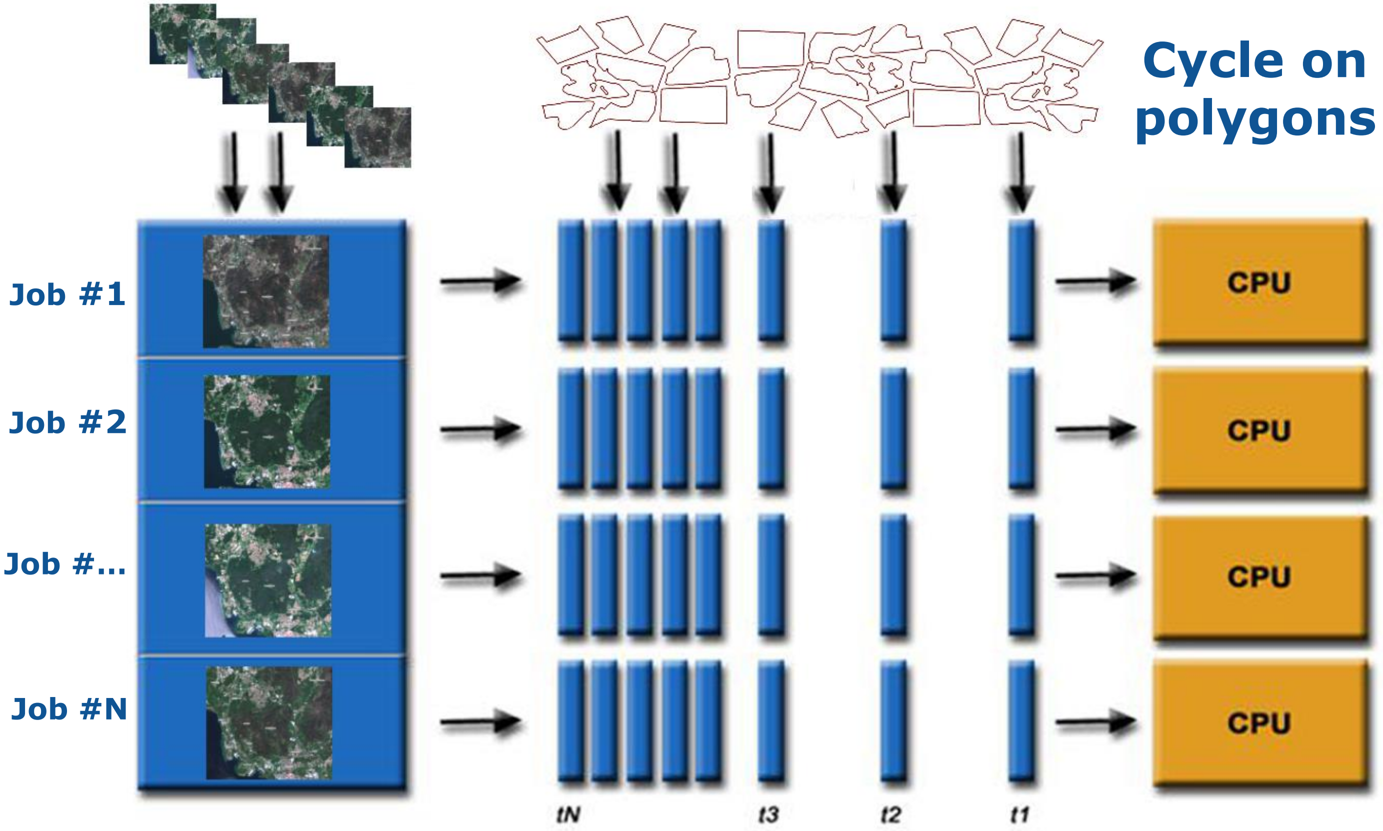
From interactive to batch services and return

Parallelize on images

Each image is read by a single job

GDAL library behaves much better with vector data parallel access

1M polygons for a full year (~1K images) processed in 5 hours using 400 cores



Cycle on polygons

Launcher | ExtractBatch_HU2.ipynb

Code | Python 2

Copernicus for EU Common Agriculture Policy Monitoring

- The dense time-series and global coverage of EU Copernicus satellite imagery opens new ways of conceiving the monitoring of agriculture in the context of the EU Common Agriculture Policy.
- This notebook illustrates tests performed on the JRC Big Data Platform (JEODPP) with batch extraction of NDVI profiles for 10K agricultural parcels in Hungary using JEO-batch. The collected data is accessible in JEO-lab to interactively explore the multi-temporal NDVI plot and the imagettes extracted for each polygon over all the Sentinel2 L2A not-cloudy images of the first seven months of 2018.

Activity performed through collaboration between the JRC Text and Data Mining and Food & Security Units.

European Commission Joint Research Centre

JEODPP The JRC Big Data Platform

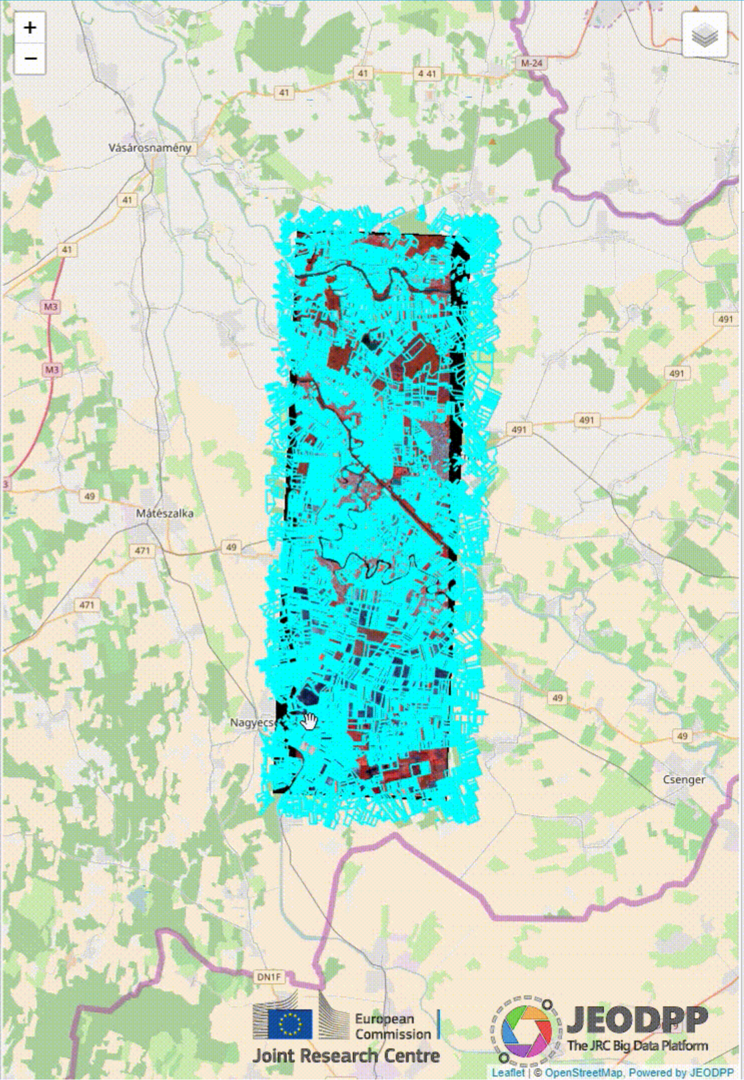
In [1]:
```
map = Map(side='Batch Extraction evaluation', center=[47.95, 22.45], zoom=12)
p = loadUserRaster('wirnhca','HU/2018/tif/hu2018_1.tif', epsg=23700)
map.addLayer(p.toLayer(), name="WorldView-4 21/04/2018")

ExtractReadLogs('/eos/jeodpp/htcondor/processing_logs/BigDataEOSS_CORE/davide/hu2018/log_image2/', True)
```

```
Reading data from binary index... 223,639 extractions found
```

Out[1]: True

In [8]:
```
ExtractBatchEvaluate(map, 'hu2.shp', lineColor='cyan', labelAttribute='crop_cod_1', minZoomLabels=15,
                     S2Display='None', autoWidth='medium', showTooltip=True,
                     histogramDimension='medium')
```



Batch Extraction evaluation

Classification and Machine learning

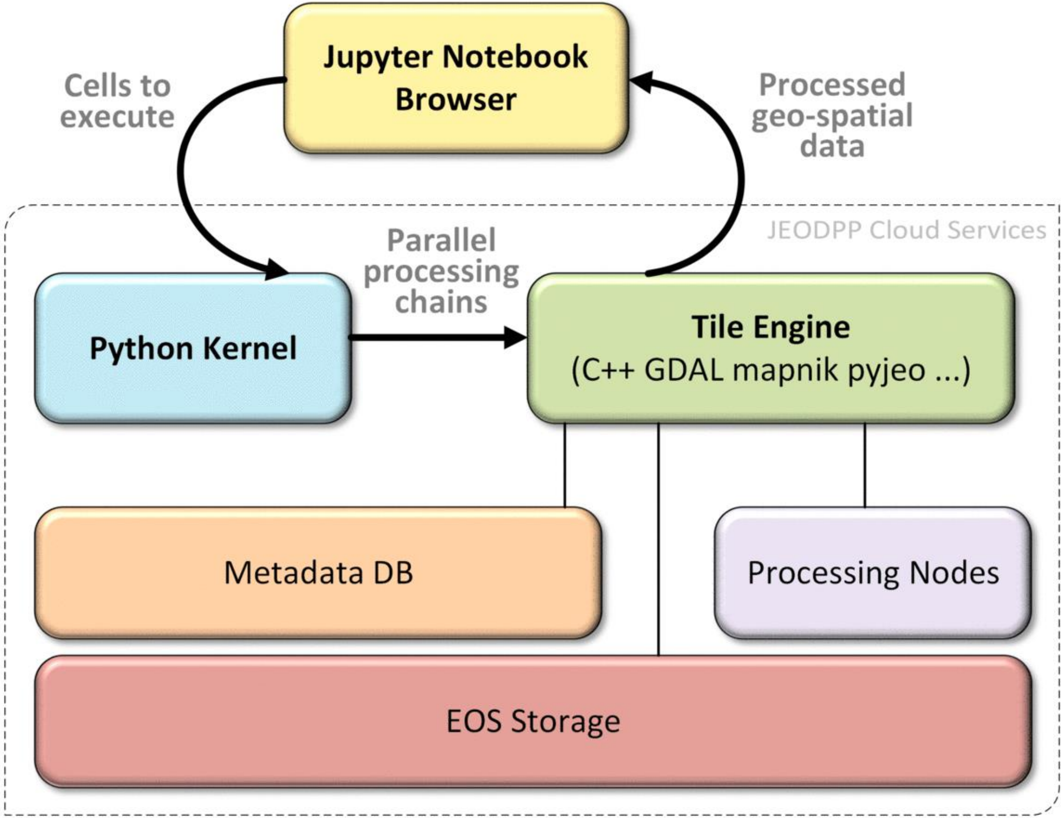
How the JEO-lab works:

3 components: **Jupyter Client**, **Python Kernel** and **Tile Engine**

The **Client** sends cells containing python code to the kernel

The **Python Kernel** transforms the requests in calls to the Tile Engine parallel library (written in C++ and using many standard geoprocessing libraries like GDAL, mapnik and JRC libraries like pyjeo, mialib, etc.)

The **Tile Engine** executes the processing chains in a highly parallel environment: reads raw data, transforms it and sends the results back to the client browser

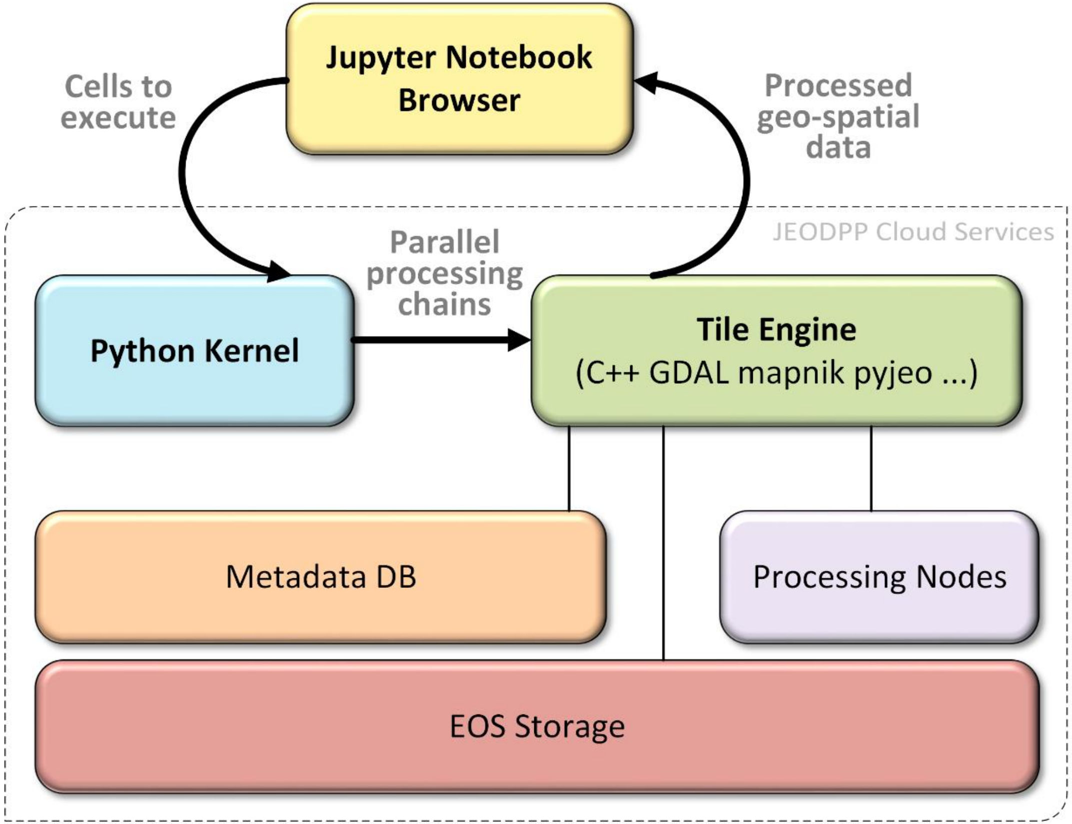


Classification and Machine learning

Need to increase user flexibility and use available python libraries

Solution: enable injection of custom python code to the server-side Tile Engine running in the HPC

```
def maskpy(img, n):
    return img[img<=n] = 0
```



Launcher | Rule-based_CloudDetector.i

Code | Python 2

Rule based cloud detector implemented in python using numpy code injected in the Tile Engine

In [ ]:
```
map = Map(side="map2")
z = map.zoomToExtent(search("Kenya"))
```

In [ ]:
```
coll = inter.ImageCollection("S2")
coll = coll.filterOnPoint(map.center[1],map.center[0])

coll = coll.filterOn("cloudCover","<",80)
coll = coll.filterOn("jrc_filepath","<>","").limit(1)
```

In [ ]:
```
p1 = coll.process().bands("B04","B03","B02")
```

In [ ]:
```
def doCloudMask():
    global img
    #use band0 as numpy array containing fist band
    #use band1 as numpy array containing second band
    #calculate your result as a numpy array
    #copy your numpy array result in the variable 'img'

    BLU = numpy.float64(band0)/10000.
    GREEN = numpy.float64(band1)/10000.
    RED = numpy.float64(band2)/10000.
    NIR = numpy.float64(band3)/10000.
    SWIR1 = numpy.float64(band4)/10000.
    SWIR2 = numpy.float64(band5)/10000.

    DATAMASK = numpy.logical_and(numpy.logical_and(RED>0, NIR>0, BLU>0), GREEN>0,
                             numpy.logical_and(SWIR1>0, SWIR2>0) )

    th_NDVI_MAX_WATER = 0
    th_NDVI_MIN_VEGE = 0.45

    th_NDVI_SATURATION=0.0037
    th_NDVI_MIN_CLOUD_BARE=0.35
    th_NDVI_MIN_VEGE=0.45

    th_SHALLOW_WATER=-0.1
    th_RANGELAND=0.49
    th_GRASS=0.53
    th_SHRUB=0.63
    th_TREES=0.78

    min123 = numpy.minimum.reduce([BLU, GREEN, RED])
    min1234 = numpy.minimum(min123, NIR)
    min12345 = numpy.minimum(min1234, SWIR1)
    min123457 = numpy.minimum(min12345, SWIR2)
    min234 = numpy.minimum.reduce([GREEN, RED, NIR])
    max234 = numpy.maximum.reduce([GREEN, RED, NIR])
    max1234 = numpy.maximum(max234, BLU)
    max57 = numpy.maximum(SWIR1, SWIR2)
    max123457 = numpy.maximum(max1234, max57)

    BLUgtGREEN = BLU>GREEN
    BLUgteGREEN = BLU>=GREEN
    GREENgtRED  = GREEN>RED
    GREENlteRED = GREEN<=RED
    GREENgteRED = GREEN>=RED
    REDlteNIR = RED<=NIR
    BLUlteNIR = BLU <= NIR

    REDsubtractGREEN = numpy.abs(RED-GREEN)
    BLUsubtractNIR = BLU-NIR
```

European Commission
Joint Research Centre

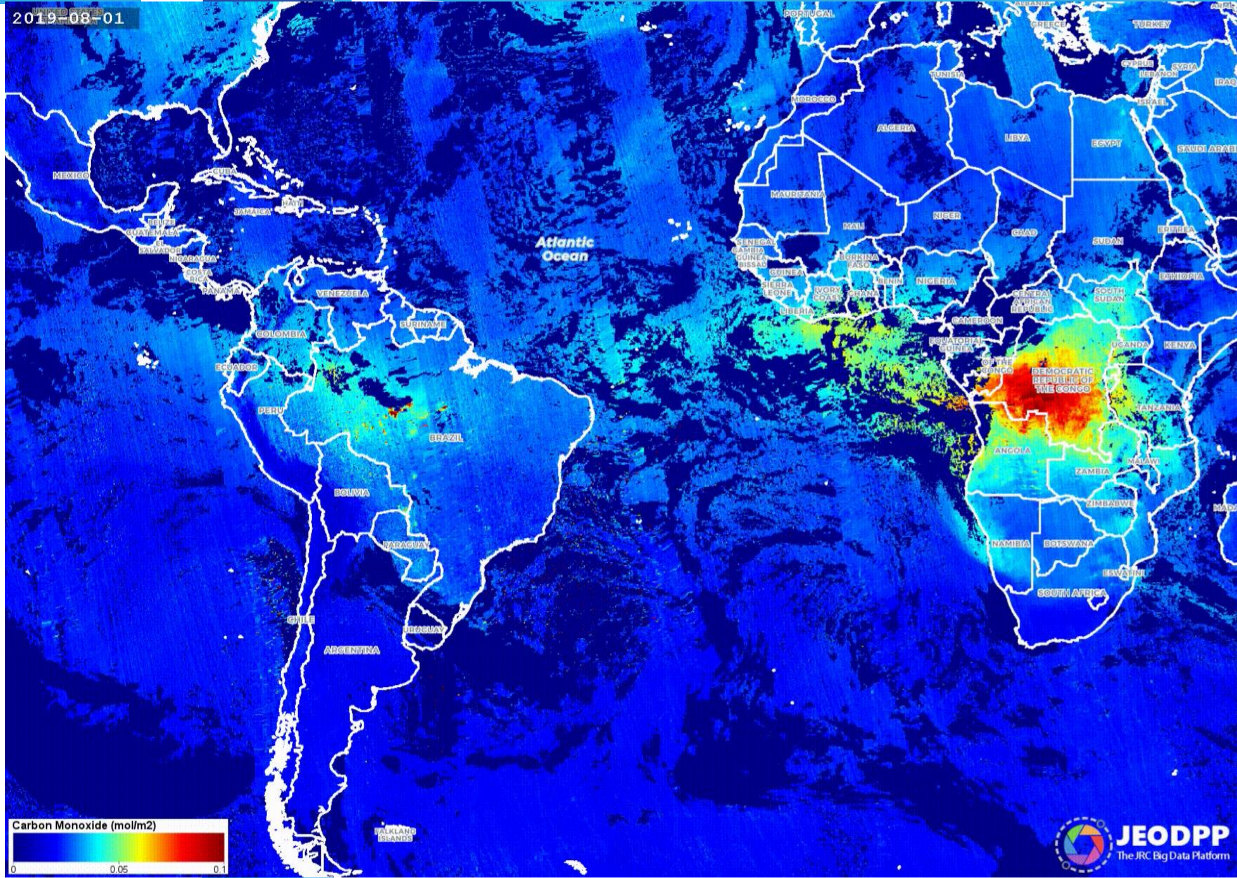
JEODPP
The JRC Big Data Platform

New datasets and services

Sentinel-5P satellite provides important insights for operational monitoring of air quality and climate

CO – Carbon monoxide
NO2 – Nitrogen dioxide
CH4 – Methane
...

Example: global CO emissions for August 2019

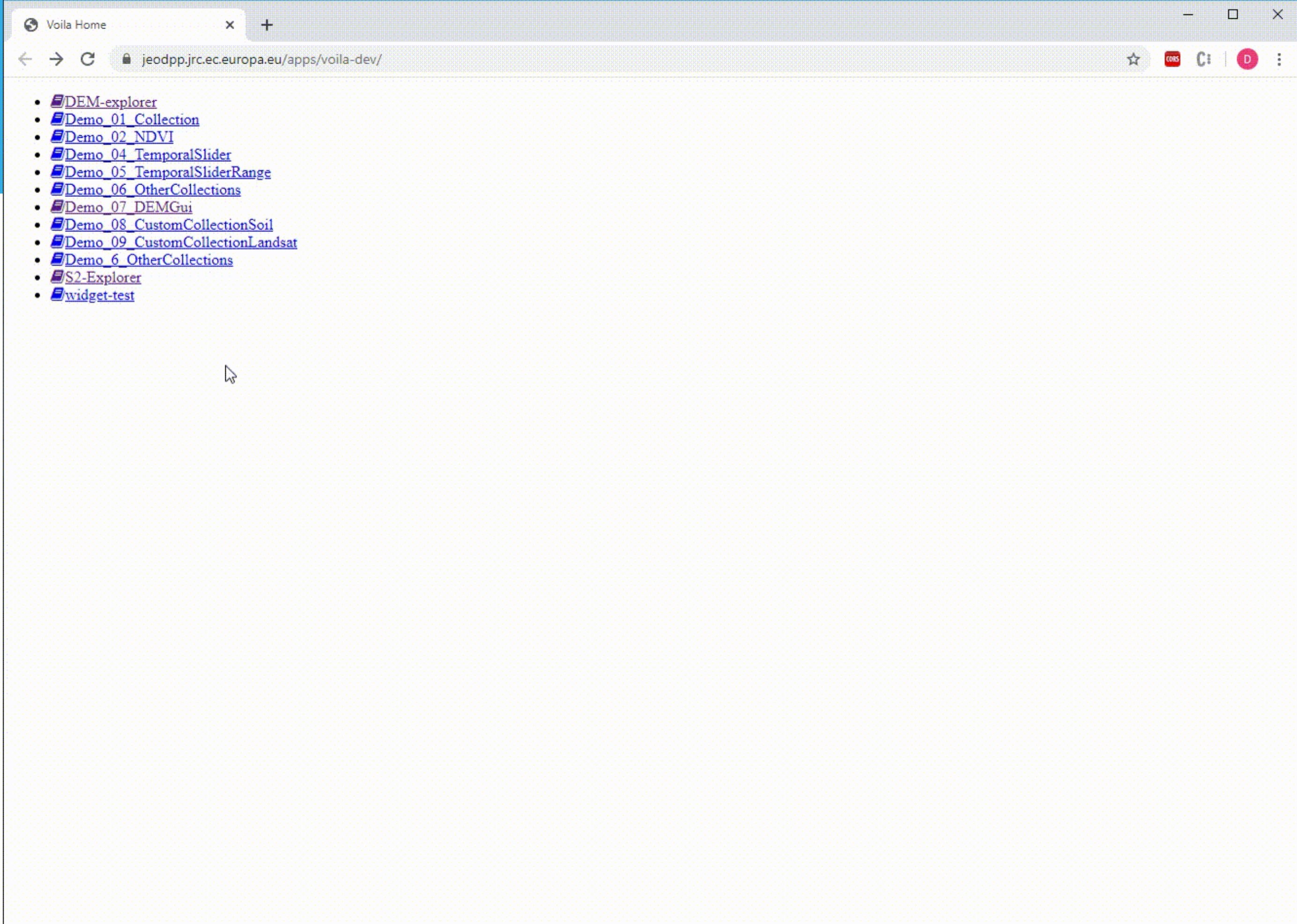


Voila' dashboards

Voilà turns Jupyter notebooks into standalone web applications

Way to open-up apps to non-registered users

Less threads on security (no python code is visible inside the HTML page)



European Commission
Joint Research Centre
JEODPP
The JRC Big Data Platform

Launcher | demexplorer.ipynb

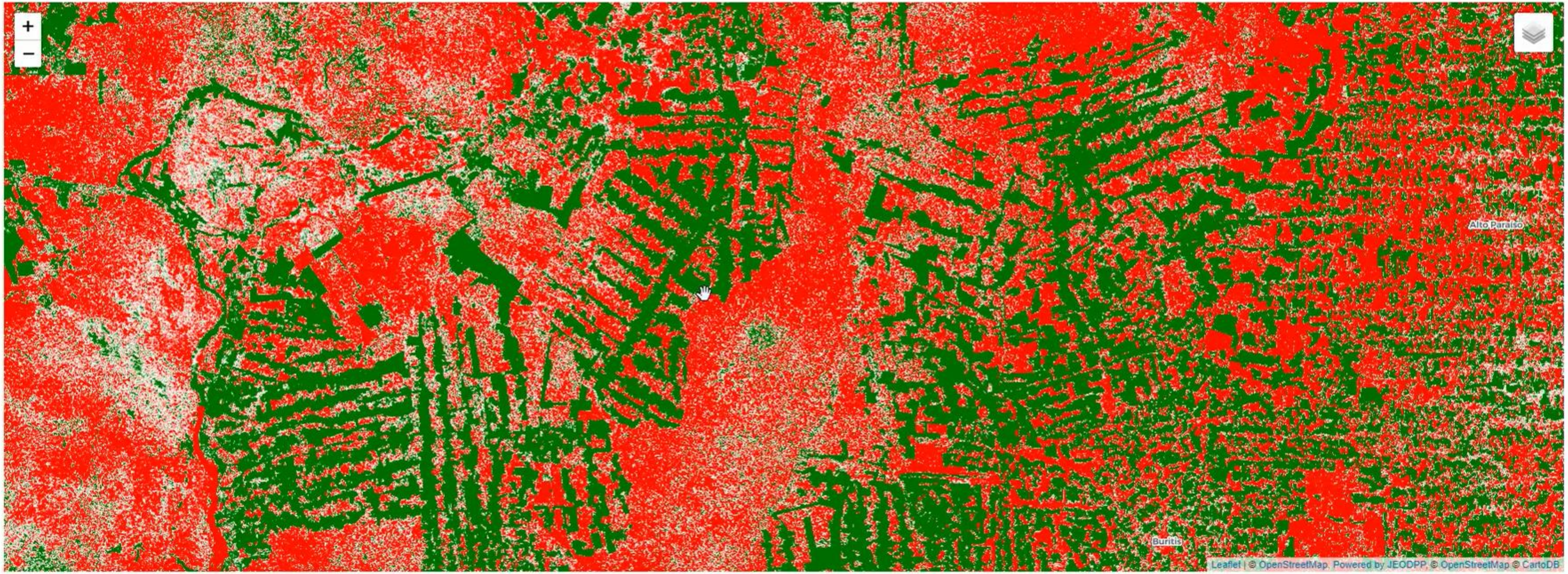
Code | Python 2

DEM explorer

In [3]:
```
demexplorer()
```

Digital Elevation | Comparison | Zoom | Exports | Measure | Draw | Overlay | Split

First DEM:	EUDEM	SRTM	GEBCO	ALOS	MERIT	NASA	Firt DEM over: Green	Difference: 4
Second DEM:	EUDEM	SRTM	GEBCO	ALOS	MERIT	NASA	Second DEM over: Red	Zoom: 10 Sharp difference



In [ ]:

In [ ]:

In [ ]:

European Commission
Joint Research Centre

JEODPP
The JRC Big Data Platform

European Commission

Stay in touch

EU Science Hub:
ec.europa.eu/jrc

Twitter:
@EU_ScienceHub

YouTube:
EU Science Hub

Facebook:
EU Science Hub – Joint Research Centre

LinkedIn:
Joint Research Centre

JEODPP
The JRC Big Data Platform

https://doi.org/10.1016/j.future.2017.11.007

Publication list:
https://cidportal.jrc.ec.europa.eu/home/publications

EPSO IT specialist competition (deadline 6th February): Cloud, Infrastructure, HPC, ...

Joint Research Centre