

Analyzing COVID-19 using Python and remote sensing images

Josep Sitjar

@josepsitjar

josep.sitjar@udg.edu

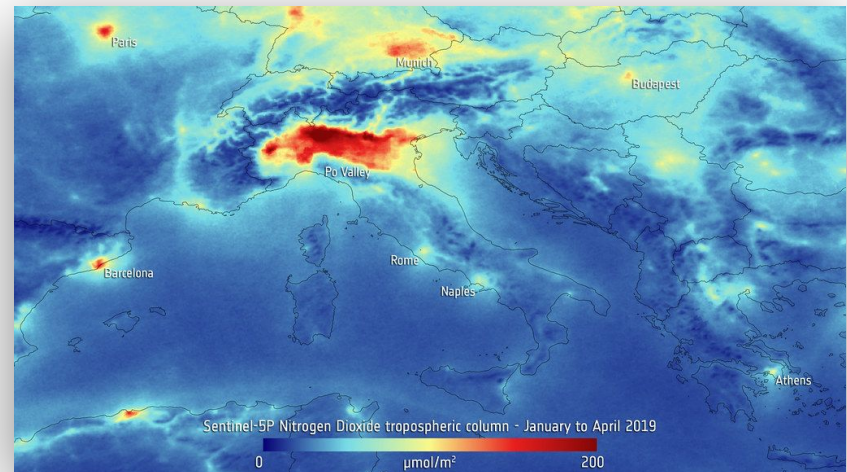
SIGTE - Universitat de Girona

Index

- Summary
- Sentinel5p and TROPOMI
- Sentinelsat - Download images
- Harp - Process Images
- Visan - Visualize Images
- Automate a workflow using a python script

Summary

The Covid-19 pandemic and its consequent lockdown has had a significant impact on industrial activity and traffic over many regions of the world. Both human activities are responsible for the **NO₂** emissions into the atmosphere, so its reduction also involved a reduction of this polluting gas.

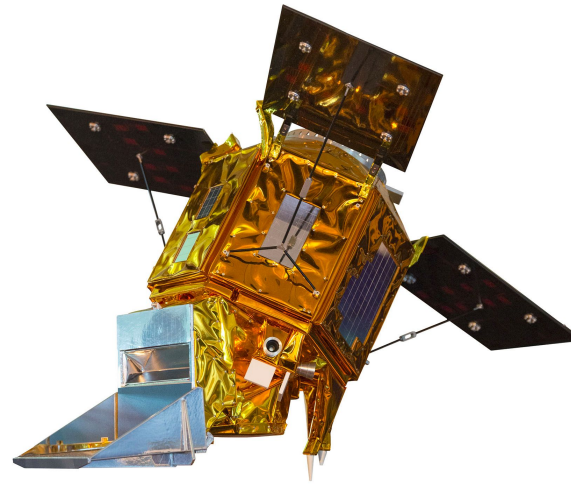


We can analyze and visualize this impact using **Python** and **Remote Sensing Images**.

Copernicus Sentinel 5P & TROPOMI

Sentinel 5P -*one of the satellites from the Sentinel constellation*- carries on board the TROPOMI sensor. This sensor is prepared to measure many atmospheric gases, among which is NO₂.

With a **daily revisit period** and a **spatial resolution about 7km²**, Sentinel 5p offers the possibility to accurately track the evolution of NO₂ concentration over the world.



Get the images

[Copernicus Open Acces Hub](#) provides **complete**, **free** and **open** access to Sentinel 5p products.

There's a GUI to search, filter and download this products, but if we want to automate the process because we require long series of images, we can code a script using the **sentinel**sat Python library.

Sentinelsat is a library that facilitates the process of searching, downloading and retrieving metadata of Sentinel satellite images from the Copernicus Open Acces Hub.

The connection with the image repository can be done through a generic account with this lines of code:

```
# connect to the API
from sentinelat import SentinelAPI, read_geojson, geojson_to_wkt
from datetime import date

api = SentinelAPI(s5pguest, s5pguest, 'https://s5phub.copernicus.eu/dhus')
```

In order to search images from a certain area with **sentinelsat**, we can use the following code:

```
# search by polygon, time, and SciHub query keywords
footprint = geojson_to_wkt(read_geojson('/path/to/map.geojson'))
products = api.query(footprint,
                     date=(date(2019, 12, 23), date(2020, 6, 29)),
                     producttype='L2_N02_---',
                     platformname='Sentinel-5')
```

products variable will contain the reference to all products (scenes) captured by Sentinel-5p over Spain from december 2019 to june 2020.

This instruction will download all this products:

```
# download all results from the search
api.download_all(products)
```

In order to cover the Spanish region, **2** Sentinel-5p products are required.

From december 2019 to june 2020 we'll download more than 190 products (2 each day). Each product weights about 400Mb.

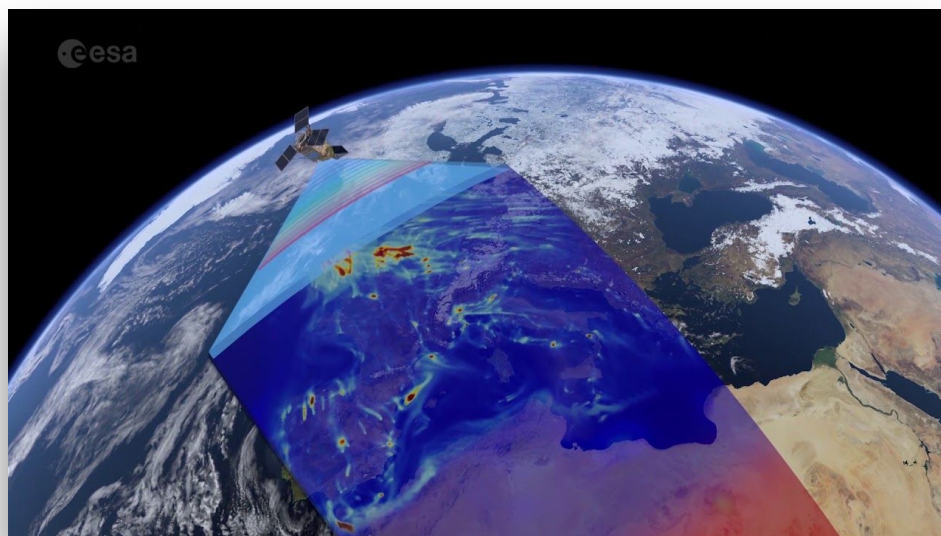


Image that shows the Sentinel 5p footprint

Processing Sentinel-5p images with HARP

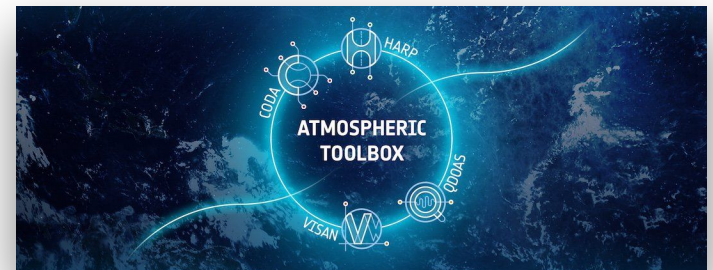
HARP is a toolkit for reading, processing and inter-comparing satellite remote sensing data, model data, in-situ data, and ground based remote sensing data.

HARP is part of Atmospheric Toolbox, a set of [several](#) software components that help scientifics to work with atmospheric data.

HARP can be installed into a **Conda** virtual environment.

In this occasion we are going to use HARP to:

- **mosaic** the downloaded daily products
- create a **temporal animation** of this images
- apply a **mask**
- **filter** the data



- Create a mosaic using HARP

First of all, we'll import the downloaded products (which are in [NetCDF](#) format) into HARP. This import will be done using the HARP ***import_product*** function. This function is also useful to clip the image extension, select the variable to represent, apply filters, etc.

```
harp.import_product(base_file + ext,  
                    operations="latitude > -55 [degree_north]; latitude < 80 [degree_north];  
tropospheric_NO2_column_number_density_validity > 75;bin_spatial(231,-55,0.5,721,-180,0.5)",  
                    post_operations="bin();squash(time, (latitude,longitude))")
```

bin_spatial() operation homogenize all product variables onto a regular grid, allowing its correct visualization.

We also apply a filter to ***tropospheric_NO2_column_number_density_validity*** variable, in order to only import pixels with a higher value than 75. It's important to apply this filter to discard the presence of clouds, snow or ice.

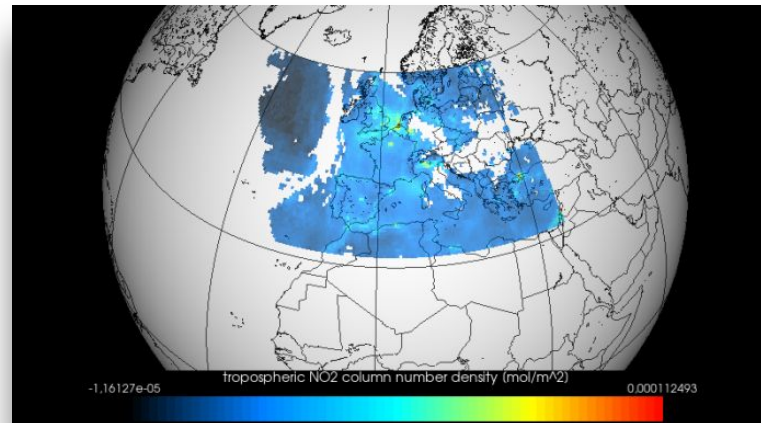
We can import into HARP as many products as necessary. We can append them into a python list to join later.

```
# store products into a list
# both products cover Spanish area
products = [no2_escena1, no2_escena2]

# join the products
product_bin = harp.exe
```

The code should be executed for each one of the days to be analyzed.

This creates a mosaic with the 2 products that cover the study area for a single day.



- **Merge the mosaics keeping the temporal dimension**

After executing the last step, we have more than 150 files, one for each analyzed day. Executing this command line from the terminal, we can merge all them into one keeping the temporal variable. So, we'll be able to visualize them in an animated way.

```
harpmerge merged_files*.nc temporal_mosaic.nc
```

- **Optimize product size**

A Sentinel-5p product weights 400Mb.

As 2 products are necessary to cover the study area, we need more than 800Mb of disk to store them. Just for a single day.

After importing the products into HARP, apply a filter to them and cut its extension, this size decrease significantly.

The last image, processed with HARP and which represents the NO2 value over the Spanish area on 20 may 2020, weighs 123Mb. A significantly smaller size.

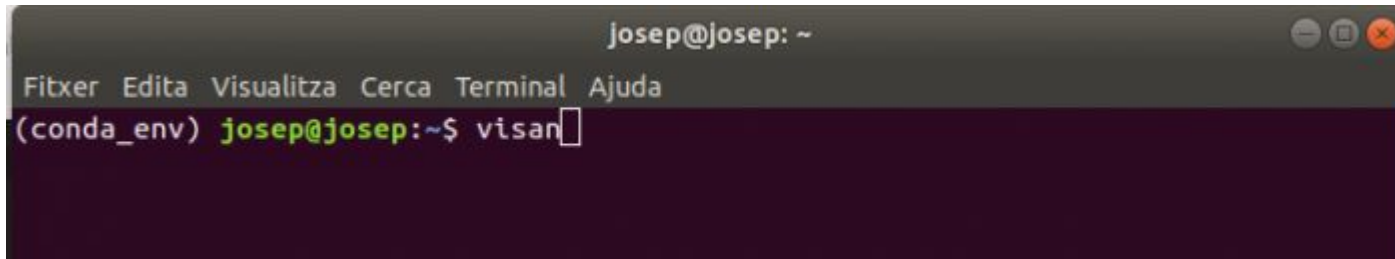
If we have not enough storage space on our system, we can program a function to delete the original products once they have been used to create the daily mosaic.

Visualize images using VISAN

The visualisation of the results can be done using **VISAN**. This python library provides powerful visualization functionalities for 2D plots and geospatial world plots, which can be executed using Python.

VISAN is also part of Atmospheric Toolbox.

When VISAN is installed on a Conda environment, we can run the software from a Linux terminal:



```
josep@josep: ~  
Fitxer Edita Visualitza Cerca Terminal Ajuda  
(conda_env) josep@josep:~$ visan
```

The software offers a command line to introduce python instructions.

A product can be imported into VISAN using:

```
product = harp.import_product(r "/home/josep/COVID/mosaic_file.nc")
```

And we can visualize the image:

```
wplot(product)
```

Programming a Python routine

SentinelSat, VISAN and HARP are useful libraries to automate a workflow to search, download, process and visualize remote sensing images. .

In this [github repository](#) I've uploaded a script prepared to create an animated images of NO2 values for a specific period of time.

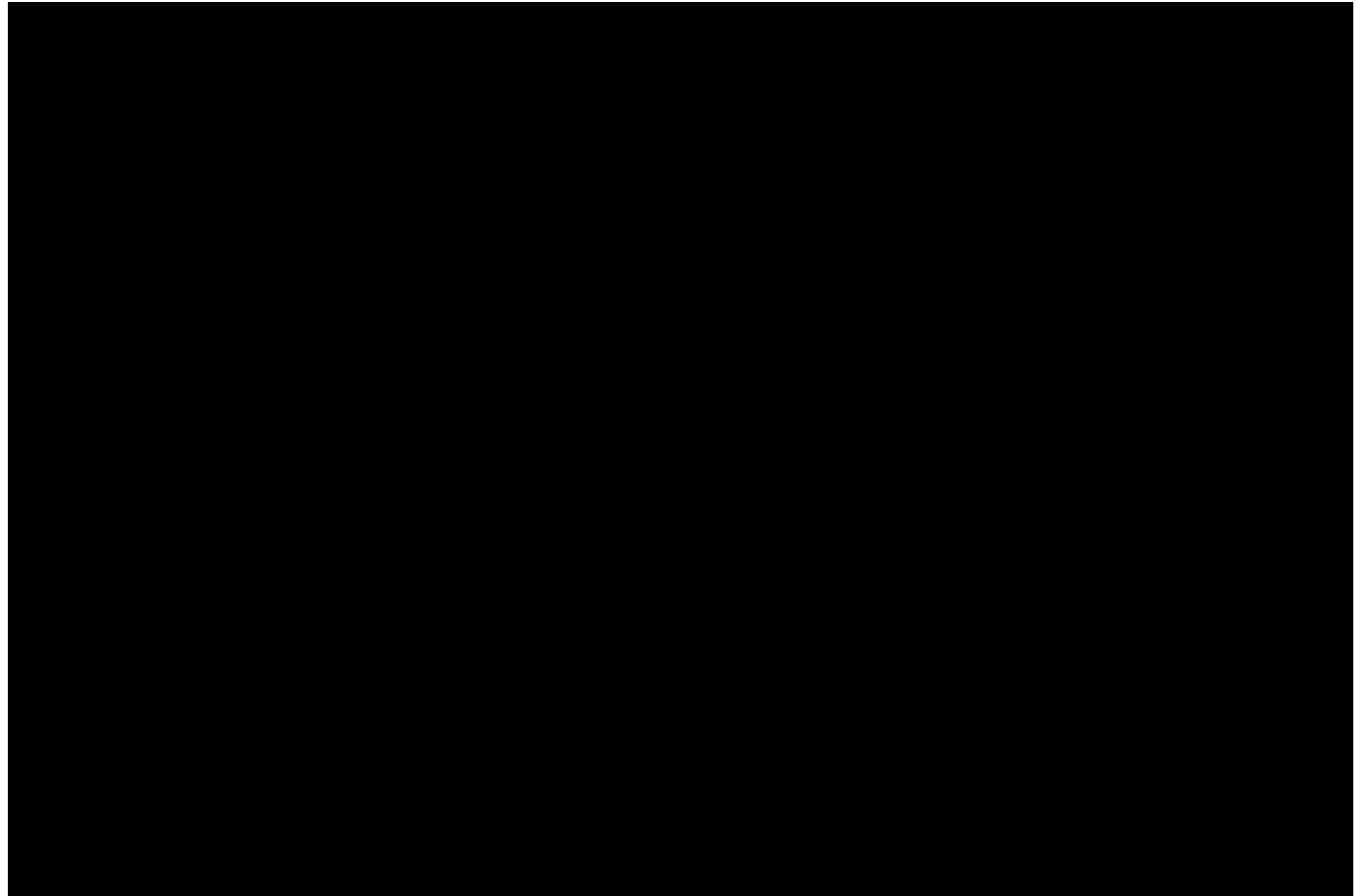
A single day image from Sentinel-5p is not significant enough to reflect the atmospheric NO2 registered values, and also atmospheric conditions such as the presence of clouds can invalidate the obtained results. For this reason, is more realistic to work with **averaged values** (7 days, for example).

The cited script includes a **function that calculates this averaged values**.

After each 7 day averaged values mosaic is created, **the original files are deleted**. So the system doesn't requires to have a large storage space.

Running this script requires a lot of time (at least 8 hours).

**Timelapse that
demonstrates the
reduction of
polluting gases
during the
lockdown.**



Any questions?

Thanks!

josep.sitjar@udg.edu