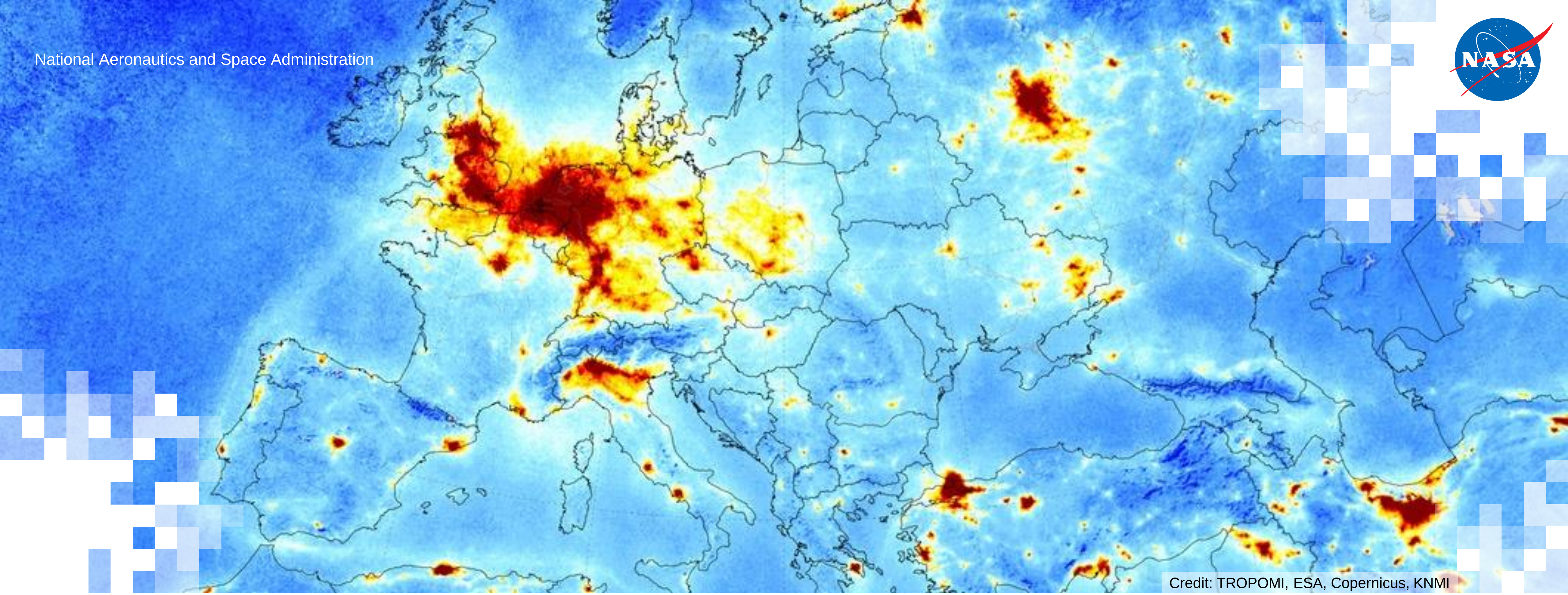
National Aeronautics and Space Administration



Credit: TROPOMI, ESA, Copernicus, KNMI

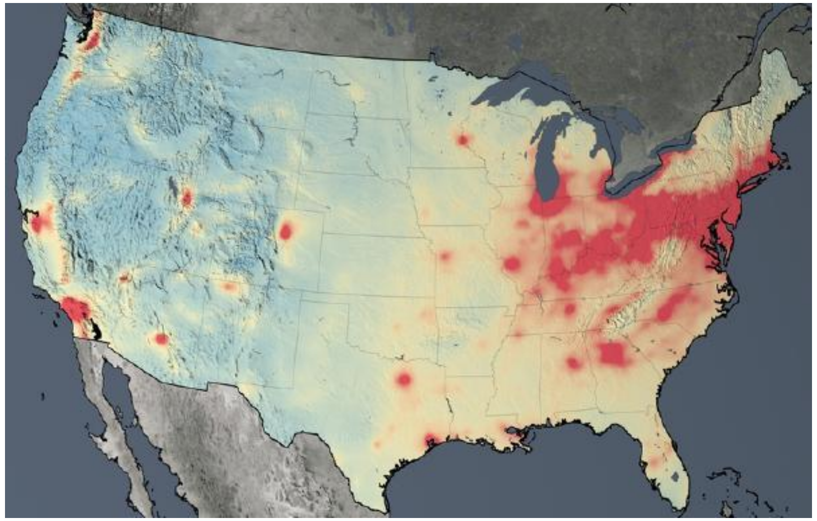
Python Tools for Analyzing NO_2 Data

Pawan Gupta and Melanie Follette-Cook

Advanced Webinar: High Resolution NO2 Monitoring From Space with TROPOMI, May 2019

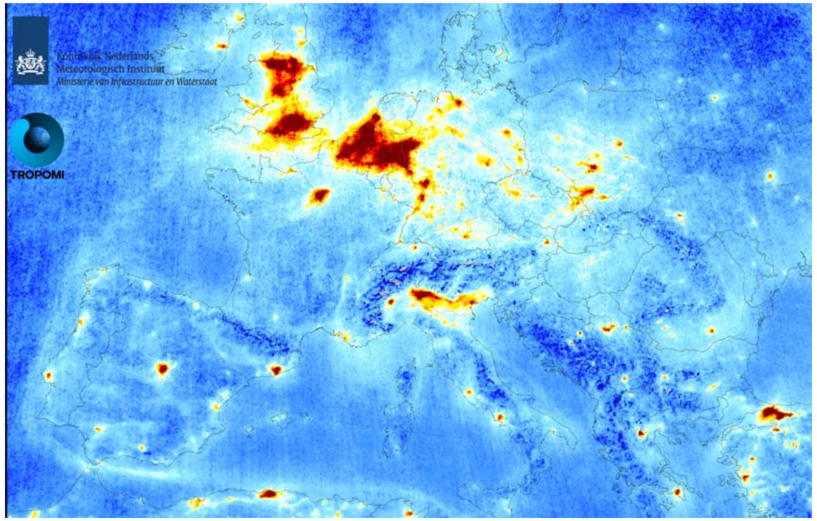
Webinar Agenda

Session 1



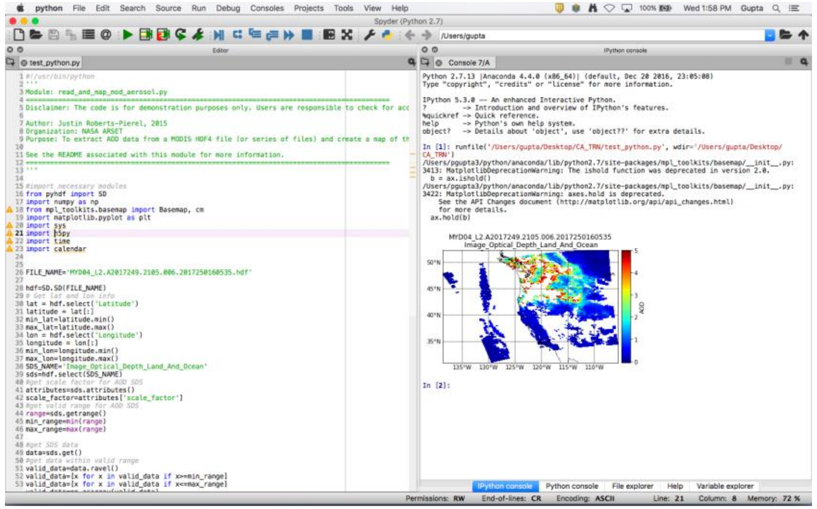
Remote sensing of NO_2, OMI Data Products, and Tools

Session 2



Introducing TROPOMI - High Resolution NO_2 Observations from Space

Session 3



```
#!/usr/bin/python
'''
Module: read_and_map_mod_aerosol.py
==========================================================================================
Disclaimer: The code is for demonstration purposes only. Users are responsible to check for acc

Author: Justin Roberts-Pierel, 2015
Organization: NASA ARSET
Purpose: To extract AOD data from a MODIS HDF4 file (or series of files) and create a map of th

See the README associated with this module for more information.
==========================================================================================
'''

#import necessary modules
from pyhdf import SD
import numpy as np
from mpl_toolkits.basemap import Basemap, cm
import matplotlib.pyplot as plt
import sys
import h5py
import time
import calendar


FILE_NAME='MYD04_L2.A2017249.2105.006.2017250160535.hdf'

hdf=SD.SD(FILE_NAME)
# Get lat and lon info
lat = hdf.select('Latitude')
latitude = lat[:]
min_lat=latitude.min()
max_lat=latitude.max()
lon = hdf.select('Longitude')
longitude = lon[:]
min_lon=longitude.min()
max_lon=longitude.max()
SDS_NAME='Image_Optical_Depth_Land_And_Ocean'
sds=hdf.select(SDS_NAME)
#get scale factor for AOD SDS
attributes=sds.attributes()
scale_factor=attributes['scale_factor']
#get valid range for AOD SDS
range=sds.getrange()
min_range=min(range)
max_range=max(range)

#get SDS data
data=sds.get()
#get data within valid range
valid_data=data.ravel()
valid_data=[x for x in valid_data if x>=min_range]
valid_data=[x for x in valid_data if x<=max_range]
```

```
Python 2.7.13 |Anaconda 4.4.0 (x86_64)| (default, Dec 20 2016, 23:05:08)
Type "copyright", "credits" or "license" for more information.

IPython 5.3.0 -- An enhanced Interactive Python.
?         -> Introduction and overview of IPython's features.
%quickref -> Quick reference.
help      -> Python's own help system.
object?   -> Details about 'object', use 'object??' for extra details.

In [1]: runfile('/Users/gupta/Desktop/CA_TRN/test_python.py', wdir='/Users/gupta/Desktop/
CA_TRN')
/Users/pgupta3/python/anaconda/lib/python2.7/site-packages/mpl_toolkits/basemap/__init__.py:
3413: MatplotlibDeprecationWarning: The ishold function was deprecated in version 2.0.
  b = ax.ishold()
/Users/pgupta3/python/anaconda/lib/python2.7/site-packages/mpl_toolkits/basemap/__init__.py:
3422: MatplotlibDeprecationWarning: axes.hold is deprecated.
    See the API Changes document (http://matplotlib.org/api/api_changes.html)
    for more details.
  ax.hold(b)

In [2]:
```

Python Tools - TROPOMI

Session 3

Introduction to Python tools for Tropospheric Monitoring Instrument (TROPOMI) Data

- Read NetCDF file and learn about SDS
- Read and map NO_2 data
- Read and extract NO_2 data at a location
- Read NetCDF and extract data into ascii format

Learning Objectives

By the end of this presentation, you will be able to:

- Read, extract and map TROPOMI NO_2 data sets

Data Sets & Tasks

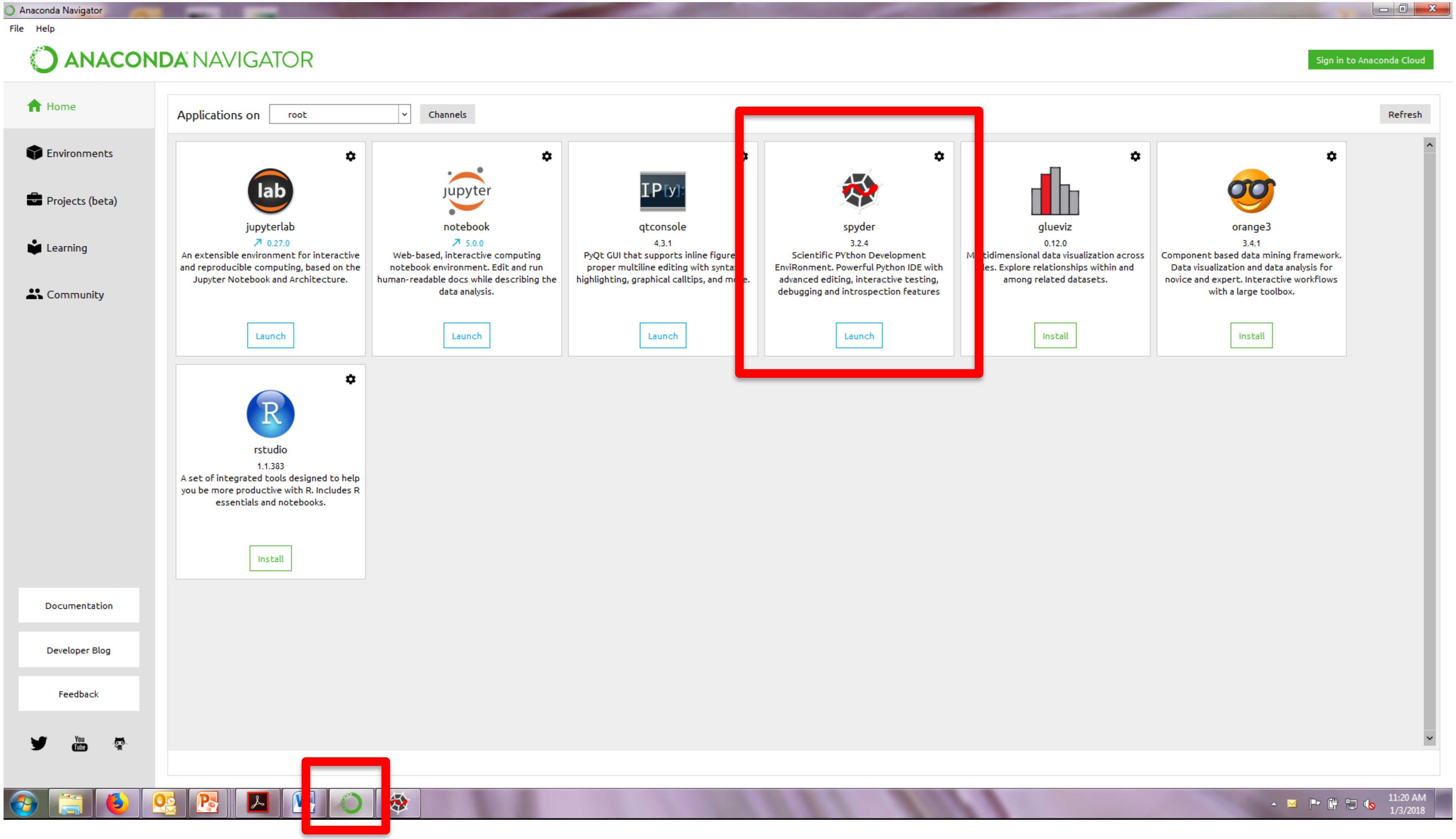
- **Data**
  - – OMI NO_2 data
  - – TROPOMI NO_2 data

- **Tasks**
  - – Read sds (scientific data sets) and list them
  - – Read and map the data
  - – Read and extract data over specific location
  - – Read and output data in a csv file

Data & Codes Required

- Screenshot of ARSET page once material is posted

Anaconda & Spyder Editor



jupyterlab: An extensible environment for interactive and reproducible computing, based on the Jupyter Notebook and Architecture.

notebook: Web-based, interactive computing notebook environment. Edit and run human-readable docs while describing the data analysis.

qtconsole: PyQt GUI that supports inline figure proper multiline editing with synta highlighting, graphical calltips, and m e.

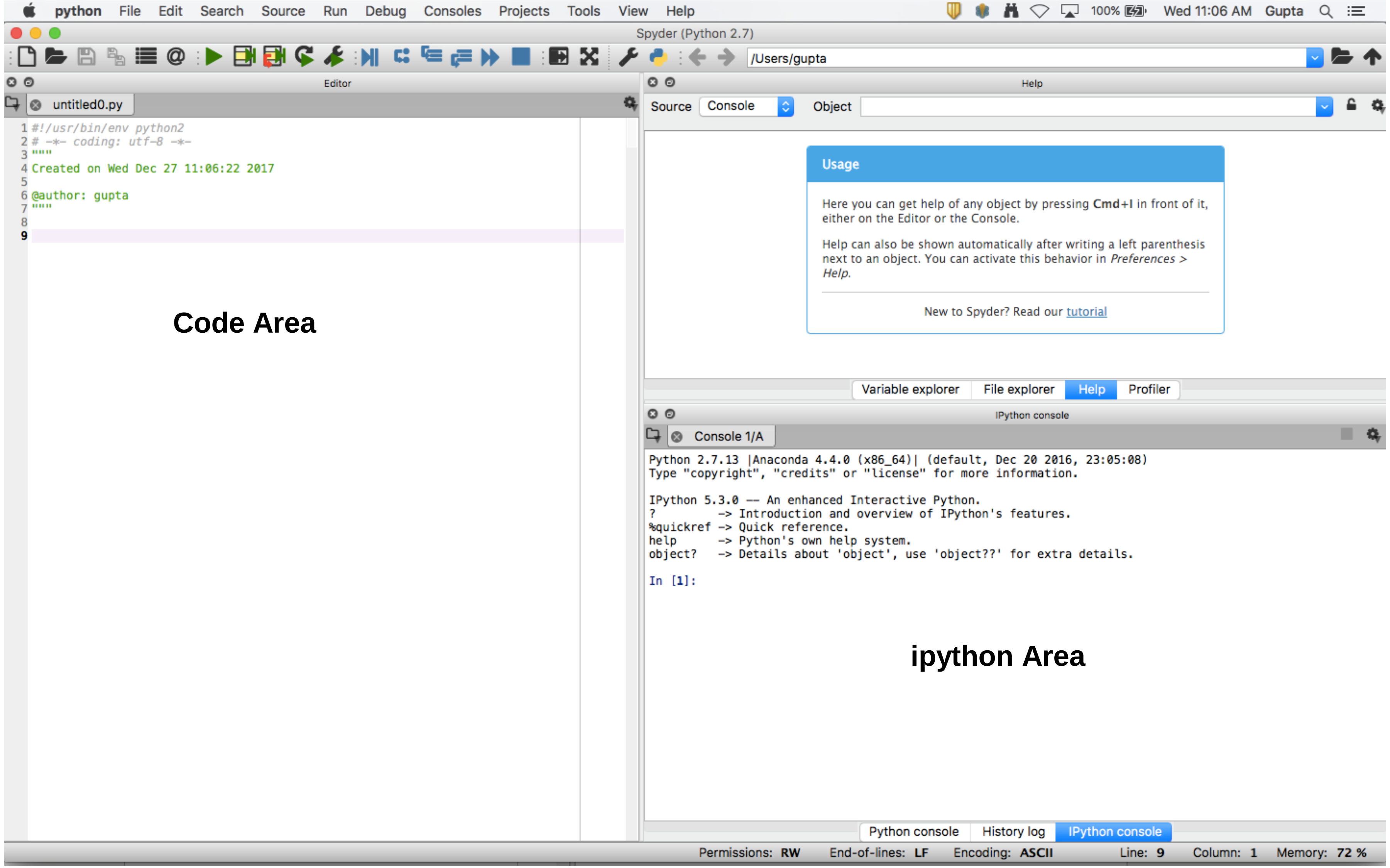
spyder: Scientific PYthon Development EnviRonment. Powerful Python IDE with advanced editing, interactive testing, debugging and introspection features

glueviz: M tidimensional data visualization across les. Explore relationships within and among related datasets.

orange3: Component based data mining framework. Data visualization and data analysis for novice and expert. Interactive workflows with a large toolbox.

rstudio: A set of integrated tools designed to help you be more productive with R. Includes R essentials and notebooks.

Spyder View



python File Edit Search Source Run Debug Consoles Projects Tools View Help
100% Wed 11:06 AM Gupta
Spyder (Python 2.7)
/Users/gupta
Editor
untitled0.py
Help
Source Console Object
Variable explorer File explorer Help Profiler
IPython console
Console 1/A
Python console History log IPython console
Permissions: RW End-of-lines: LF Encoding: ASCII Line: 9 Column: 1 Memory: 72 %

```
#!/usr/bin/env python2
# -*- coding: utf-8 -*-
"""
Created on Wed Dec 27 11:06:22 2017

@author: gupta
"""

```

Code Area

Usage

Here you can get help of any object by pressing **Cmd+I** in front of it, either on the Editor or the Console.

Help can also be shown automatically after writing a left parenthesis next to an object. You can activate this behavior in *Preferences > Help*.

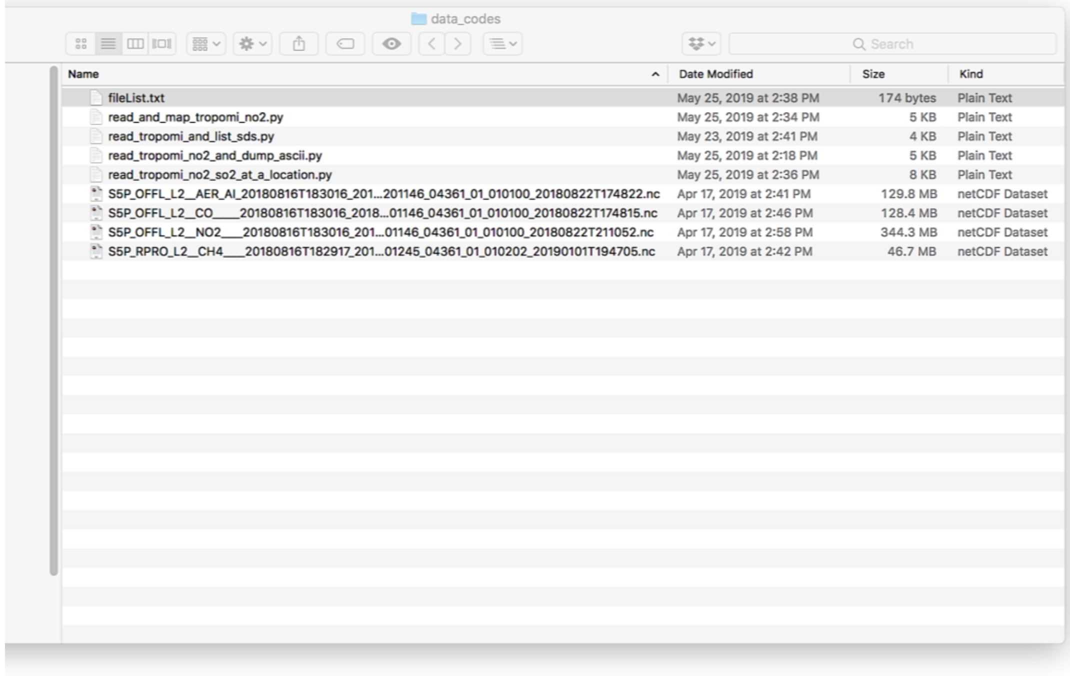
New to Spyder? Read our tutorial

```
Python 2.7.13 |Anaconda 4.4.0 (x86_64)| (default, Dec 20 2016, 23:05:08)
Type "copyright", "credits" or "license" for more information.

IPython 5.3.0 -- An enhanced Interactive Python.
?         -> Introduction and overview of IPython's features.
%quickref -> Quick reference.
help      -> Python's own help system.
object?   -> Details about 'object', use 'object??' for extra details.

In [1]:
```

ipython Area

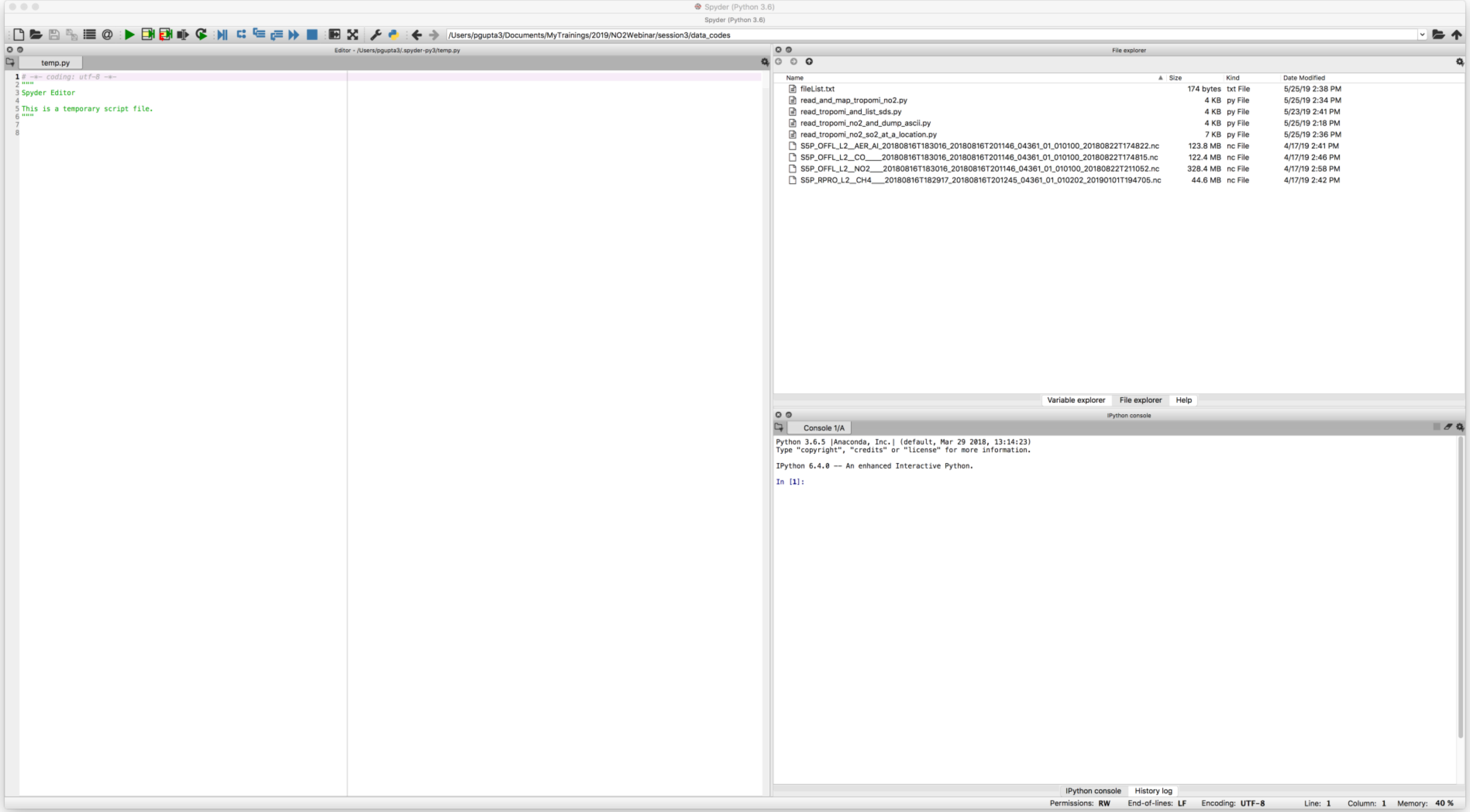
Current Directory View & fileList.txt

- In a text file, create a list of each netcdf file of interest and name it, ‘fileList.txt’
- The same directory should have
  - All the python codes
  - All the netcdf (.nc) data files
  - A file named ‘fileList.txt’ that contains a list of each netcdf filename



Name	Date Modified	Size	Kind
fileList.txt	May 25, 2019 at 2:38 PM	174 bytes	Plain Text
read_and_map_tropomi_no2.py	May 25, 2019 at 2:34 PM	5 KB	Plain Text
read_tropomi_and_list_sds.py	May 23, 2019 at 2:41 PM	4 KB	Plain Text
read_tropomi_no2_and_dump_ascii.py	May 25, 2019 at 2:18 PM	5 KB	Plain Text
read_tropomi_no2_so2_at_a_location.py	May 25, 2019 at 2:36 PM	8 KB	Plain Text
S5P_OFFL_L2__AER_AI_20180816T183016_201...201146_04361_01_010100_20180822T174822.nc	Apr 17, 2019 at 2:41 PM	129.8 MB	netCDF Dataset
S5P_OFFL_L2__CO____20180816T183016_2018...01146_04361_01_010100_20180822T174815.nc	Apr 17, 2019 at 2:46 PM	128.4 MB	netCDF Dataset
S5P_OFFL_L2__NO2___20180816T183016_201...01146_04361_01_010100_20180822T211052.nc	Apr 17, 2019 at 2:58 PM	344.3 MB	netCDF Dataset
S5P_RPRO_L2__CH4___20180816T182917_201...01245_04361_01_010202_20190101T194705.nc	Apr 17, 2019 at 2:42 PM	46.7 MB	netCDF Dataset

Spyder View



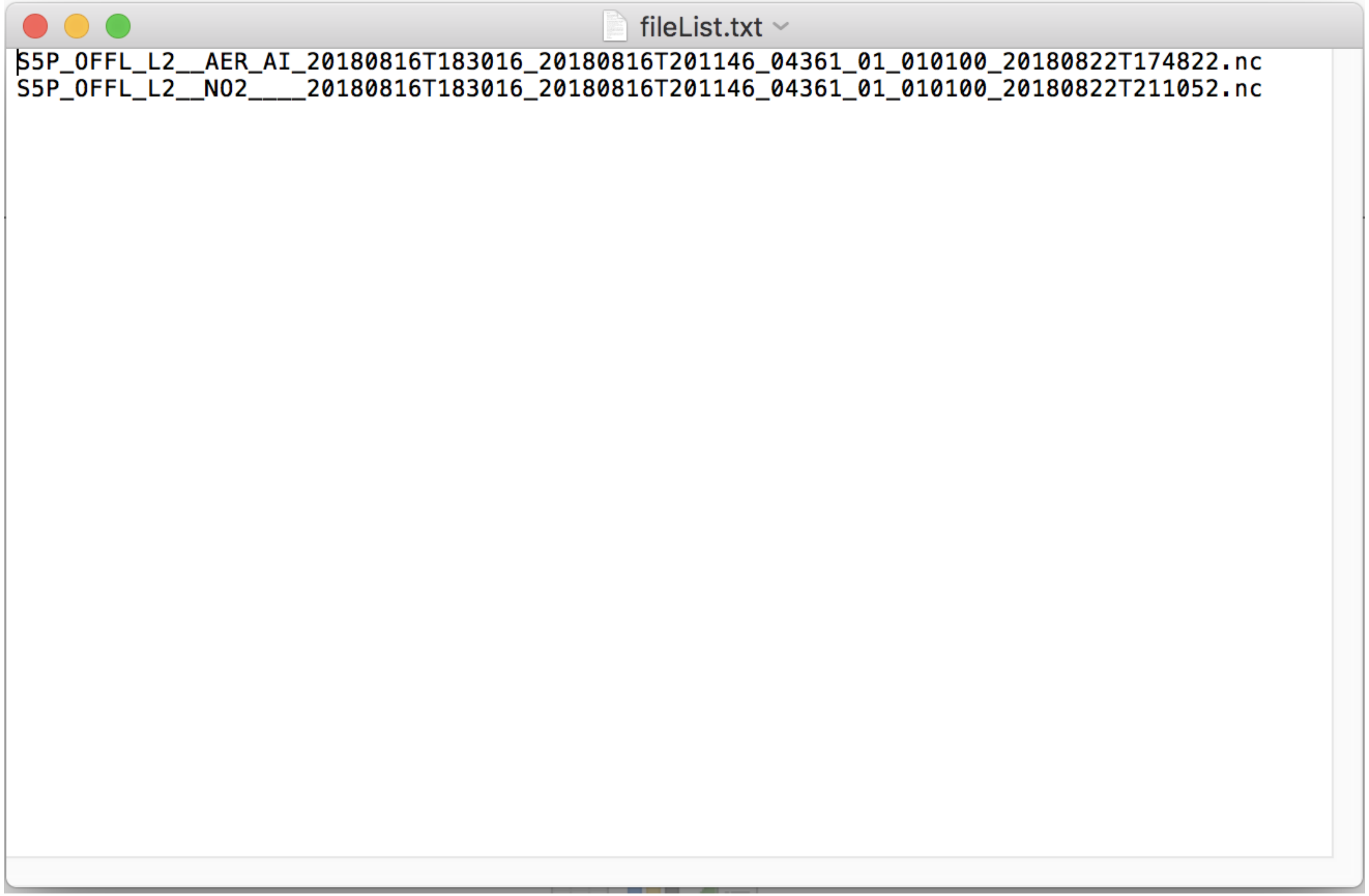
```
# -*- coding: utf-8 -*-
"""
Spyder Editor

This is a temporary script file.
"""
```

Name	Size	Kind	Date Modified
fileList.txt	174 bytes	txt File	5/25/19 2:38 PM
read_and_map_tropomi_no2.py	4 KB	py File	5/25/19 2:34 PM
read_tropomi_and_list_sds.py	4 KB	py File	5/23/19 2:41 PM
read_tropomi_no2_and_dump_ascii.py	4 KB	py File	5/25/19 2:18 PM
read_tropomi_no2_so2_at_a_location.py	7 KB	py File	5/25/19 2:36 PM
S5P_OFFL_L2__AER_AI_20180816T183016_20180816T201146_04361_01_010100_20180822T174822.nc	123.8 MB	nc File	4/17/19 2:41 PM
S5P_OFFL_L2__CO_____20180816T183016_20180816T201146_04361_01_010100_20180822T174815.nc	122.4 MB	nc File	4/17/19 2:46 PM
S5P_OFFL_L2__NO2____20180816T183016_20180816T201146_04361_01_010100_20180822T211052.nc	328.4 MB	nc File	4/17/19 2:58 PM
S5P_RPRO_L2__CH4____20180816T182917_20180816T201245_04361_01_010202_20190101T194705.nc	44.6 MB	nc File	4/17/19 2:42 PM

```
Python 3.6.5 |Anaconda, Inc.| (default, Mar 29 2018, 13:14:23)
Type "copyright", "credits" or "license" for more information.

IPython 6.4.0 -- An enhanced Interactive Python.

In [1]:
```

Permissions: **RW** End-of-lines: **LF** Encoding: **UTF-8** Line: **1** Column: **1** Memory: **40 %**

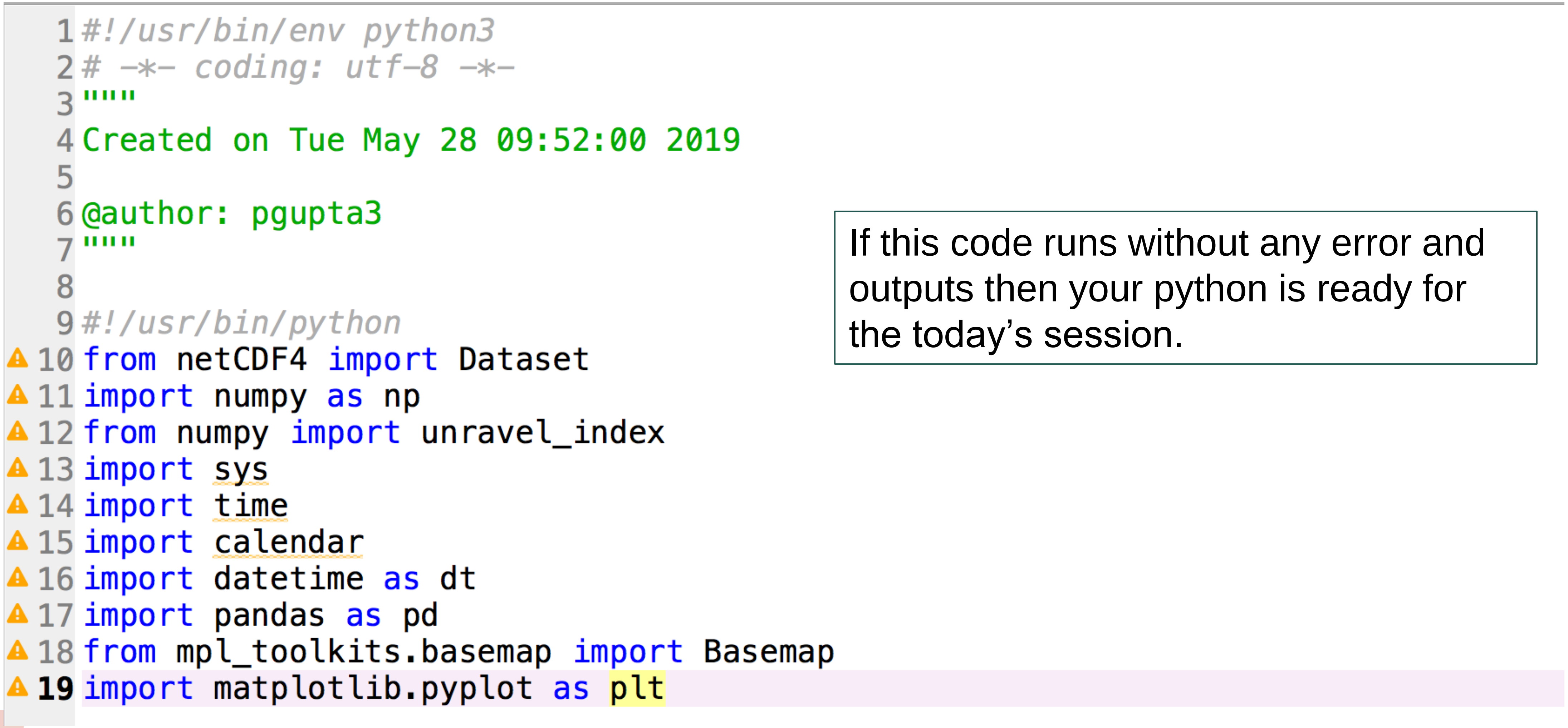
fileList.txt



```
S5P_OFFL_L2__AER_AI_20180816T183016_20180816T201146_04361_01_010100_20180822T174822.nc
S5P_OFFL_L2__NO2____20180816T183016_20180816T201146_04361_01_010100_20180822T211052.nc
```

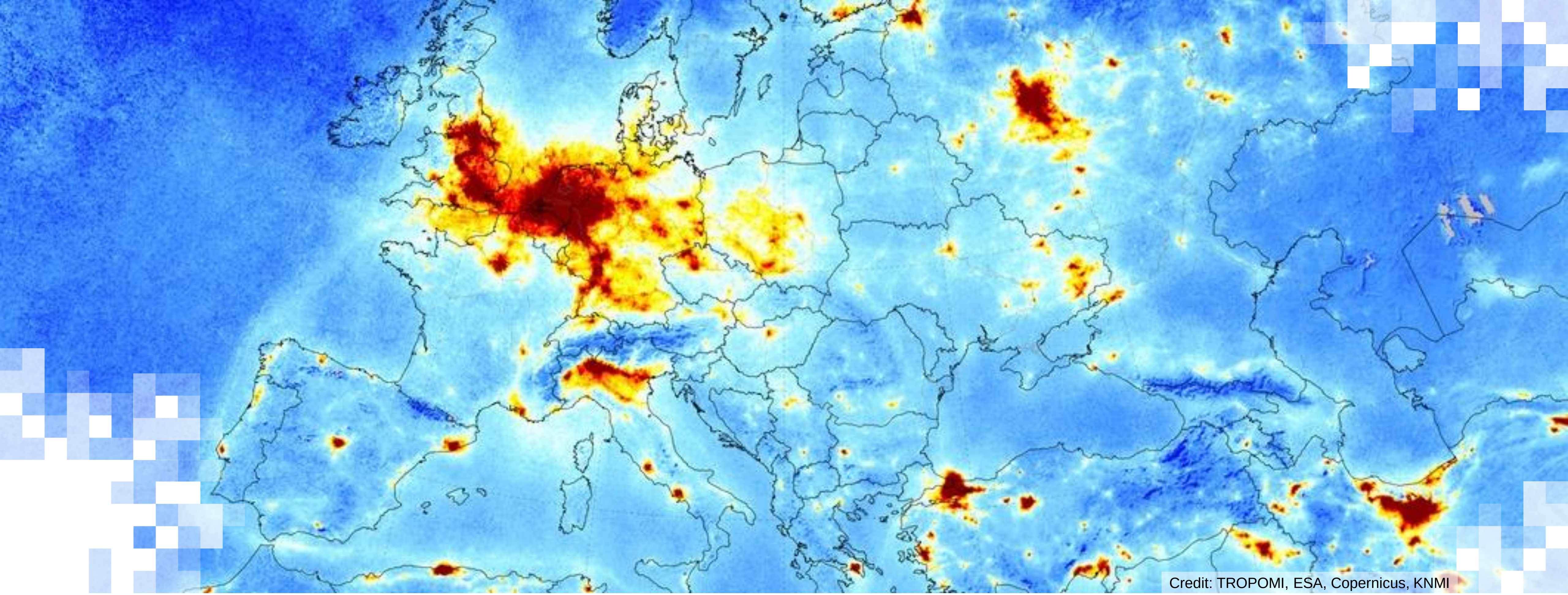
Python Packages & Test code

Open the test code and run it



```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
"""
Created on Tue May 28 09:52:00 2019

@author: pgupta3
"""

#!/usr/bin/python
from netCDF4 import Dataset
import numpy as np
from numpy import unravel_index
import sys
import time
import calendar
import datetime as dt
import pandas as pd
from mpl_toolkits.basemap import Basemap
import matplotlib.pyplot as plt
```

If this code runs without any error and outputs then your python is ready for the today's session.


Credit: TROPOMI, ESA, Copernicus, KNMI

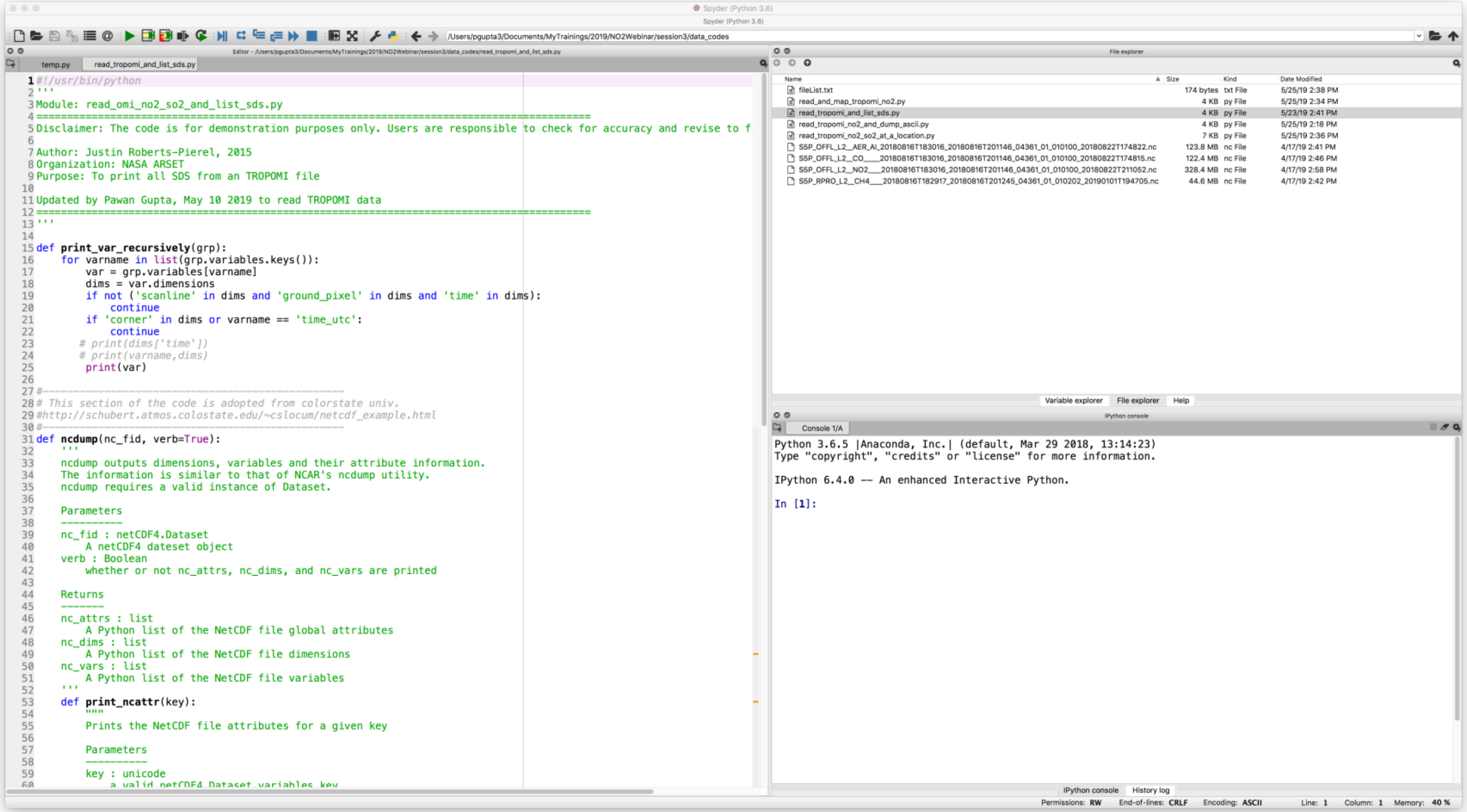
Read a TROPOMI NO_2 File (nc) and Print SDS List

Print Scientific Data Sets (SDSs)

read_tropomi_and_list_sds.py

Purpose: read TROPOMI level 2 data files in netcdf format and print all the **Scientific Data Sets** (SDS).

In their current form, all of these codes work for *only level 2 products, not gridded products*. The code is tested for NO_2 data and may require to modify to work with other TROPOMI data sets.



Spyder (Python 3.6)
/Users/pgupta3/Documents/MyTrainings/2019/NO2Webinar/session3/data_codes
Editor - /Users/pgupta3/Documents/MyTrainings/2019/NO2Webinar/session3/data_codes/read_tropomi_and_list_sds.py
temp.py
read_tropomi_and_list_sds.py

```python
#!/usr/bin/python
'''
Module: read_omi_no2_so2_and_list_sds.py
==========================================================================================
Disclaimer: The code is for demonstration purposes only. Users are responsible to check for accuracy and revise to f

Author: Justin Roberts-Pierel, 2015
Organization: NASA ARSET
Purpose: To print all SDS from an TROPOMI file

Updated by Pawan Gupta, May 10 2019 to read TROPOMI data
==========================================================================================
'''

def print_var_recursively(grp):
    for varname in list(grp.variables.keys()):
        var = grp.variables[varname]
        dims = var.dimensions
        if not ('scanline' in dims and 'ground_pixel' in dims and 'time' in dims):
            continue
        if 'corner' in dims or varname == 'time_utc':
            continue
       # print(dims['time'])
       # print(varname,dims)
        print(var)

#--------------------------------------------------
# This section of the code is adopted from colorstate univ.
#http://schubert.atmos.colostate.edu/~cslocum/netcdf_example.html
#--------------------------------------------------
def ncdump(nc_fid, verb=True):
    '''
    ncdump outputs dimensions, variables and their attribute information.
    The information is similar to that of NCAR's ncdump utility.
    ncdump requires a valid instance of Dataset.

    Parameters
    ----------
    nc_fid : netCDF4.Dataset
        A netCDF4 dateset object
    verb : Boolean
        whether or not nc_attrs, nc_dims, and nc_vars are printed

    Returns
    -------
    nc_attrs : list
        A Python list of the NetCDF file global attributes
    nc_dims : list
        A Python list of the NetCDF file dimensions
    nc_vars : list
        A Python list of the NetCDF file variables
    '''
    def print_ncattr(key):
        """
        Prints the NetCDF file attributes for a given key

        Parameters
        ----------
        key : unicode
            a valid netCDF4.Dataset.variables key
```

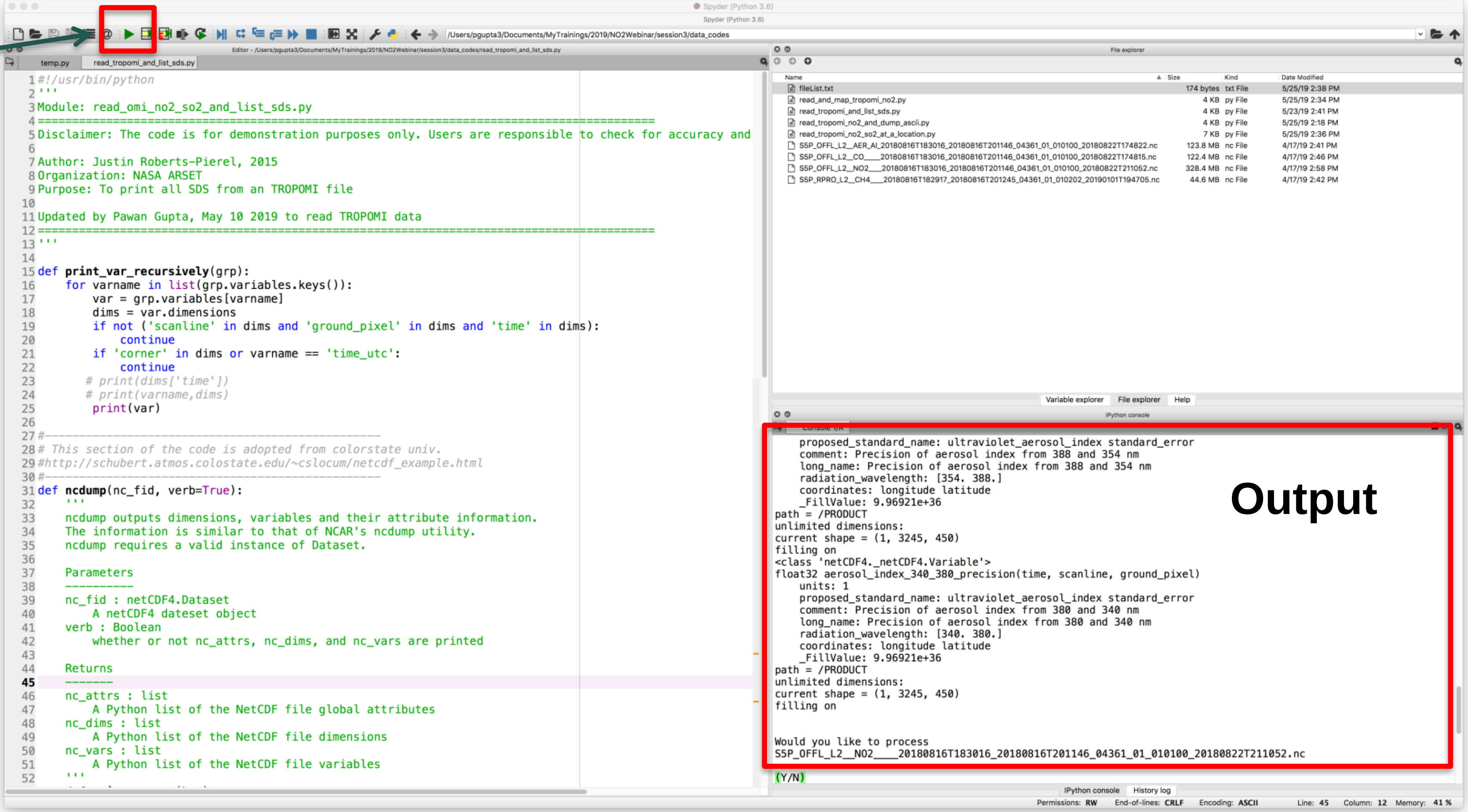
Name	Size	Kind	Date Modified
fileList.txt	174 bytes	txt File	5/25/19 2:38 PM
read_and_map_tropomi_no2.py	4 KB	py File	5/25/19 2:34 PM
read_tropomi_and_list_sds.py	4 KB	py File	5/23/19 2:41 PM
read_tropomi_no2_and_dump_ascii.py	4 KB	py File	5/25/19 2:18 PM
read_tropomi_no2_so2_at_a_location.py	7 KB	py File	5/25/19 2:36 PM
S5P_OFFL_L2__AER_AI_20180816T183016_20180816T201146_04361_01_010100_20180822T174822.nc	123.8 MB	nc File	4/17/19 2:41 PM
S5P_OFFL_L2__CO_____20180816T183016_20180816T201146_04361_01_010100_20180822T174815.nc	122.4 MB	nc File	4/17/19 2:46 PM
S5P_OFFL_L2__NO2____20180816T183016_20180816T201146_04361_01_010100_20180822T211052.nc	328.4 MB	nc File	4/17/19 2:58 PM
S5P_RPRO_L2__CH4____20180816T182917_20180816T201245_04361_01_010202_20190101T194705.nc	44.6 MB	nc File	4/17/19 2:42 PM

```
Python 3.6.5 |Anaconda, Inc.| (default, Mar 29 2018, 13:14:23)
Type "copyright", "credits" or "license" for more information.

IPython 6.4.0 -- An enhanced Interactive Python.

In [1]:
```

Permissions: RW End-of-lines: CRLF Encoding: ASCII Line: 1 Column: 1 Memory: 40 %

Running and Output

- Click the green arrow to run the code
- The code will process all of the files in **fileList.txt** one-by-one
- Follow the instructions on the **ipython** terminal (i.e. enter ‘Y’ or ‘N’ when prompted and hit enter)


Spyder (Python 3.6)
/Users/pgupta3/Documents/MyTrainings/2019/NO2Webinar/session3/data_codes
Editor - /Users/pgupta3/Documents/MyTrainings/2019/NO2Webinar/session3/data_codes/read_tropomi_and_list_sds.py
temp.py
read_tropomi_and_list_sds.py
File explorer
Variable explorer
File explorer
Help
IPython console
Console 1/A
IPython console
History log
Output

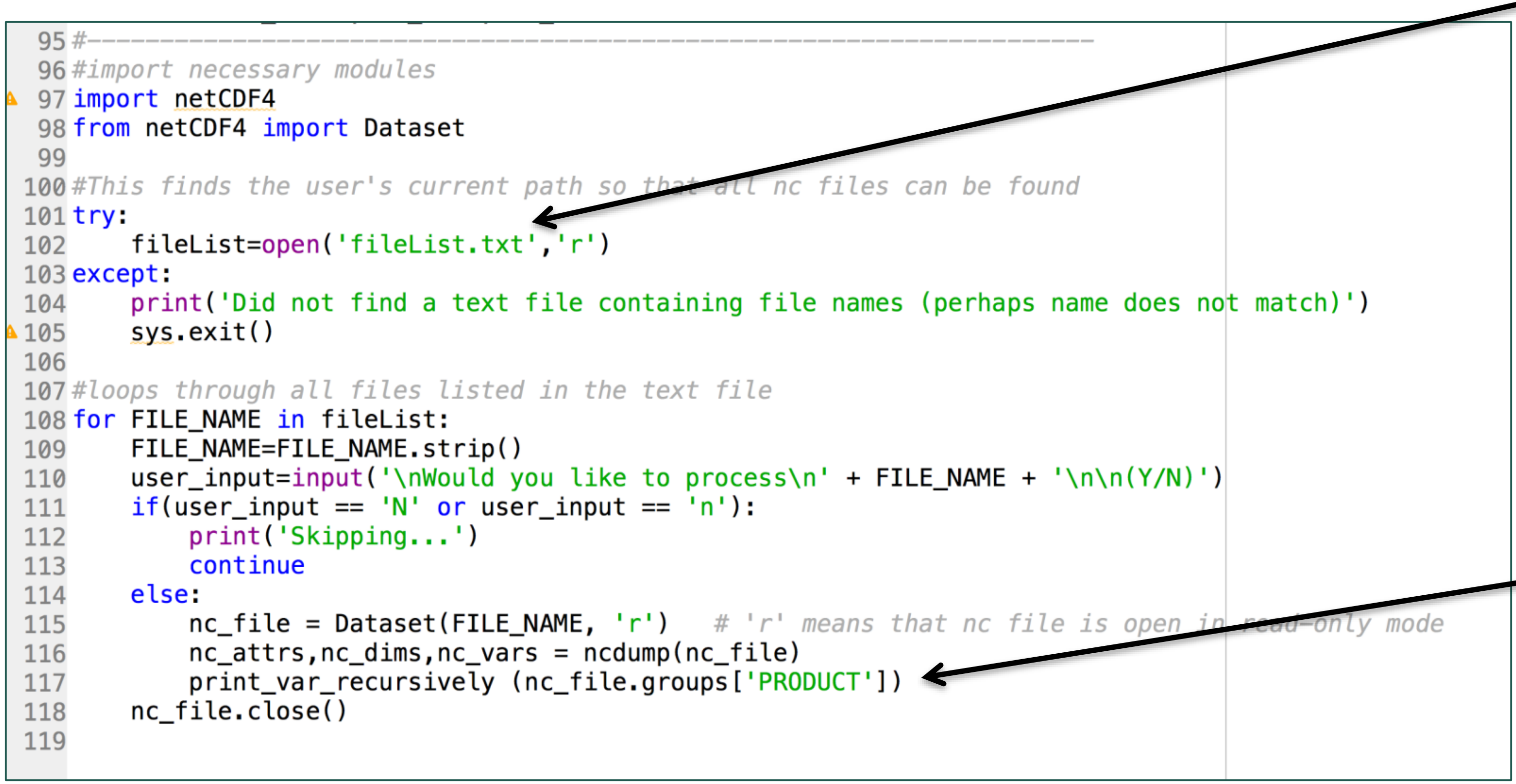
```python
#!/usr/bin/python
'''
Module: read_omi_no2_so2_and_list_sds.py
==========================================================================================
Disclaimer: The code is for demonstration purposes only. Users are responsible to check for accuracy and

Author: Justin Roberts-Pierel, 2015
Organization: NASA ARSET
Purpose: To print all SDS from an TROPOMI file

Updated by Pawan Gupta, May 10 2019 to read TROPOMI data
==========================================================================================
'''

def print_var_recursively(grp):
    for varname in list(grp.variables.keys()):
        var = grp.variables[varname]
        dims = var.dimensions
        if not ('scanline' in dims and 'ground_pixel' in dims and 'time' in dims):
            continue
        if 'corner' in dims or varname == 'time_utc':
            continue
       # print(dims['time'])
       # print(varname,dims)
        print(var)

#-----------------------------------------------------
# This section of the code is adopted from colorstate univ.
#http://schubert.atmos.colostate.edu/~cslocum/netcdf_example.html
#-----------------------------------------------------
def ncdump(nc_fid, verb=True):
    '''
    ncdump outputs dimensions, variables and their attribute information.
    The information is similar to that of NCAR's ncdump utility.
    ncdump requires a valid instance of Dataset.

    Parameters
    ----------
    nc_fid : netCDF4.Dataset
        A netCDF4 dateset object
    verb : Boolean
        whether or not nc_attrs, nc_dims, and nc_vars are printed

    Returns
    -------
    nc_attrs : list
        A Python list of the NetCDF file global attributes
    nc_dims : list
        A Python list of the NetCDF file dimensions
    nc_vars : list
        A Python list of the NetCDF file variables
    '''
```

Name	Size	Kind	Date Modified
fileList.txt	174 bytes	txt File	5/25/19 2:38 PM
read_and_map_tropomi_no2.py	4 KB	py File	5/25/19 2:34 PM
read_tropomi_and_list_sds.py	4 KB	py File	5/23/19 2:41 PM
read_tropomi_no2_and_dump_ascii.py	4 KB	py File	5/25/19 2:18 PM
read_tropomi_no2_so2_at_a_location.py	7 KB	py File	5/25/19 2:36 PM
S5P_OFFL_L2__AER_AI_20180816T183016_20180816T201146_04361_01_010100_20180822T174822.nc	123.8 MB	nc File	4/17/19 2:41 PM
S5P_OFFL_L2__CO_____20180816T183016_20180816T201146_04361_01_010100_20180822T174815.nc	122.4 MB	nc File	4/17/19 2:46 PM
S5P_OFFL_L2__NO2____20180816T183016_20180816T201146_04361_01_010100_20180822T211052.nc	328.4 MB	nc File	4/17/19 2:58 PM
S5P_RPRO_L2__CH4____20180816T182917_20180816T201245_04361_01_010202_20190101T194705.nc	44.6 MB	nc File	4/17/19 2:42 PM

```
    proposed_standard_name: ultraviolet_aerosol_index standard_error
    comment: Precision of aerosol index from 388 and 354 nm
    long_name: Precision of aerosol index from 388 and 354 nm
    radiation_wavelength: [354. 388.]
    coordinates: longitude latitude
    _FillValue: 9.96921e+36
path = /PRODUCT
unlimited dimensions:
current shape = (1, 3245, 450)
filling on
<class 'netCDF4._netCDF4.Variable'>
float32 aerosol_index_340_380_precision(time, scanline, ground_pixel)
    units: 1
    proposed_standard_name: ultraviolet_aerosol_index standard_error
    comment: Precision of aerosol index from 380 and 340 nm
    long_name: Precision of aerosol index from 380 and 340 nm
    radiation_wavelength: [340. 380.]
    coordinates: longitude latitude
    _FillValue: 9.96921e+36
path = /PRODUCT
unlimited dimensions:
current shape = (1, 3245, 450)
filling on


Would you like to process
S5P_OFFL_L2__NO2____20180816T183016_20180816T201146_04361_01_010100_20180822T211052.nc
(Y/N)
```

Permissions: RW   End-of-lines: CRLF   Encoding: ASCII   Line: 45   Column: 12   Memory: 41 %

Editing the Code



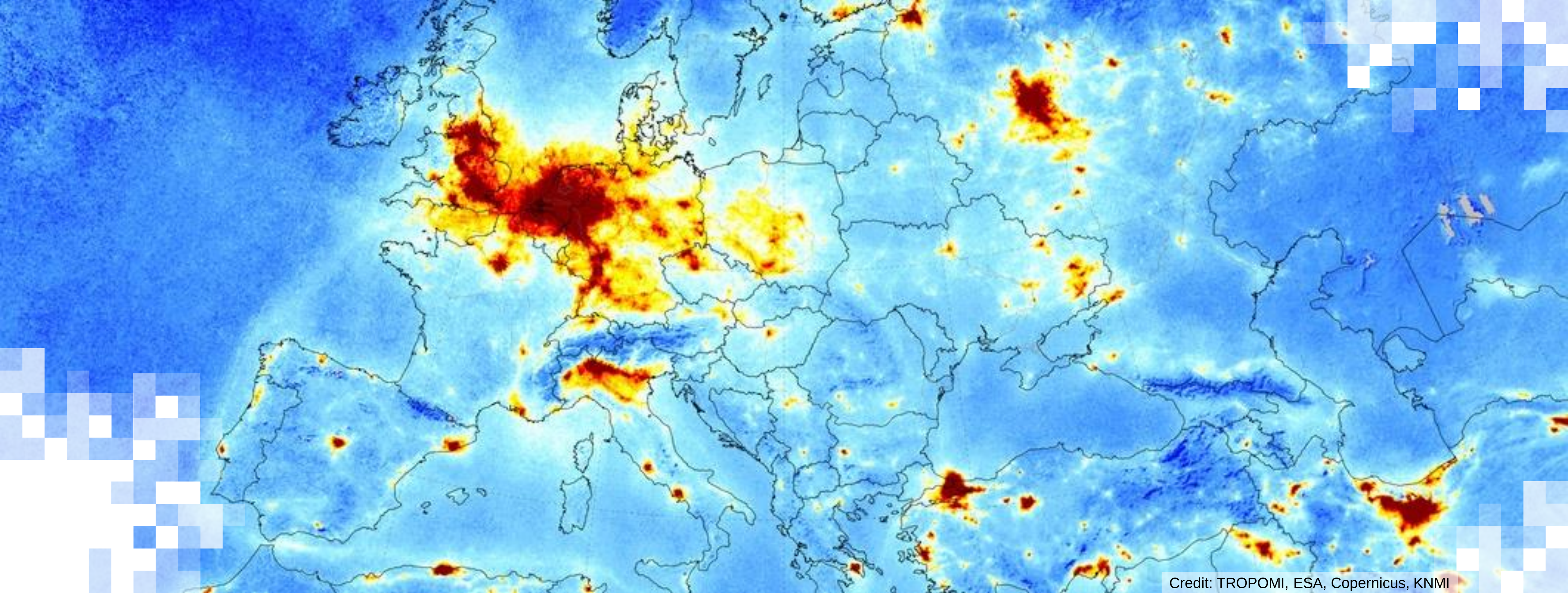
```
#----------------------------------------------------------------------
#import necessary modules
import netCDF4
from netCDF4 import Dataset

#This finds the user's current path so that all nc files can be found
try:
    fileList=open('fileList.txt','r')
except:
    print('Did not find a text file containing file names (perhaps name does not match)')
    sys.exit()

#loops through all files listed in the text file
for FILE_NAME in fileList:
    FILE_NAME=FILE_NAME.strip()
    user_input=input('\nWould you like to process\n' + FILE_NAME + '\n\n(Y/N)')
    if(user_input == 'N' or user_input == 'n'):
        print('Skipping...')
        continue
    else:
        nc_file = Dataset(FILE_NAME, 'r')   # 'r' means that nc file is open in read-only mode
        nc_attrs,nc_dims,nc_vars = ncdump(nc_file)
        print_var_recursively (nc_file.groups['PRODUCT'])
    nc_file.close()

```

Change the name of fileList.txt to anything you'd like

The group name in TROPOMI where data are stored is called 'PRODUCT'. There are other groups in the data file.

Applications

- TROPOMI Level 2 NO_2 and other data are provided in netCDF (.nc) file
- Each nc file contains several geophysical parameters
- Special codes/tools are required to open the nc files
- This code helps users see the names and dimensions of the available SDSs inside an nc file for further analysis



Credit: TROPOMI, ESA, Copernicus, KNMI

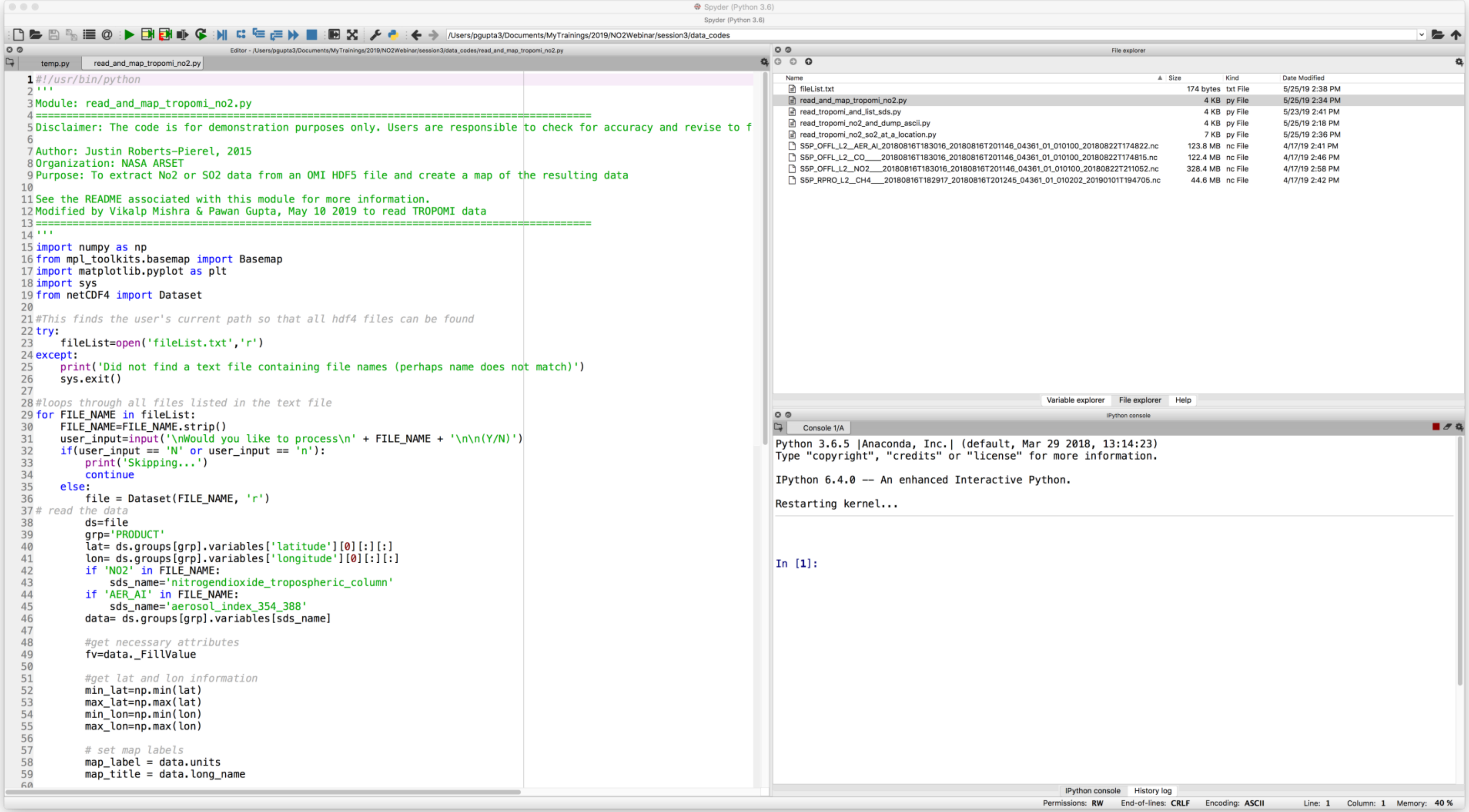
Map NO_2

Reminders

- Close the earlier code in Spyder
- Restart the ipython kernel

Plot and save a map of TROPOMI AI & NO_2

read_and_map_tropomi_no2_ai.py


Spyder (Python 3.6)
/Users/pgupta3/Documents/MyTrainings/2019/NO2Webinar/session3/data_codes
Editor - /Users/pgupta3/Documents/MyTrainings/2019/NO2Webinar/session3/data_codes/read_and_map_tropomi_no2.py
temp.py
read_and_map_tropomi_no2.py
File explorer
Variable explorer
File explorer
Help
IPython console
Console 1/A
IPython console
History log

```python
#!/usr/bin/python
'''
Module: read_and_map_tropomi_no2.py
==========================================================================================
Disclaimer: The code is for demonstration purposes only. Users are responsible to check for accuracy and revise to f

Author: Justin Roberts-Pierel, 2015
Organization: NASA ARSET
Purpose: To extract No2 or SO2 data from an OMI HDF5 file and create a map of the resulting data

See the README associated with this module for more information.
Modified by Vikalp Mishra & Pawan Gupta, May 10 2019 to read TROPOMI data
==========================================================================================
'''
import numpy as np
from mpl_toolkits.basemap import Basemap
import matplotlib.pyplot as plt
import sys
from netCDF4 import Dataset

#This finds the user's current path so that all hdf4 files can be found
try:
    fileList=open('fileList.txt','r')
except:
    print('Did not find a text file containing file names (perhaps name does not match)')
    sys.exit()

#loops through all files listed in the text file
for FILE_NAME in fileList:
    FILE_NAME=FILE_NAME.strip()
    user_input=input('\nWould you like to process\n' + FILE_NAME + '\n\n(Y/N)')
    if(user_input == 'N' or user_input == 'n'):
        print('Skipping...')
        continue
    else:
        file = Dataset(FILE_NAME, 'r')
# read the data
        ds=file
        grp='PRODUCT'
        lat= ds.groups[grp].variables['latitude'][0][:][:]
        lon= ds.groups[grp].variables['longitude'][0][:][:]
        if 'NO2' in FILE_NAME:
            sds_name='nitrogendioxide_tropospheric_column'
        if 'AER_AI' in FILE_NAME:
            sds_name='aerosol_index_354_388'
        data= ds.groups[grp].variables[sds_name]

        #get necessary attributes
        fv=data._FillValue

        #get lat and lon information
        min_lat=np.min(lat)
        max_lat=np.max(lat)
        min_lon=np.min(lon)
        max_lon=np.max(lon)

        # set map labels
        map_label = data.units
        map_title = data.long_name
```

Name	Size	Kind	Date Modified
fileList.txt	174 bytes	txt File	5/25/19 2:38 PM
read_and_map_tropomi_no2.py	4 KB	py File	5/25/19 2:34 PM
read_tropomi_and_list_sds.py	4 KB	py File	5/23/19 2:41 PM
read_tropomi_no2_and_dump_ascii.py	4 KB	py File	5/25/19 2:18 PM
read_tropomi_no2_so2_at_a_location.py	7 KB	py File	5/25/19 2:36 PM
S5P_OFFL_L2__AER_AI_20180816T183016_20180816T201146_04361_01_010100_20180822T174822.nc	123.8 MB	nc File	4/17/19 2:41 PM
S5P_OFFL_L2__CO_____20180816T183016_20180816T201146_04361_01_010100_20180822T174815.nc	122.4 MB	nc File	4/17/19 2:46 PM
S5P_OFFL_L2__NO2____20180816T183016_20180816T201146_04361_01_010100_20180822T211052.nc	328.4 MB	nc File	4/17/19 2:58 PM
S5P_RPRO_L2__CH4____20180816T182917_20180816T201245_04361_01_010202_20190101T194705.nc	44.6 MB	nc File	4/17/19 2:42 PM

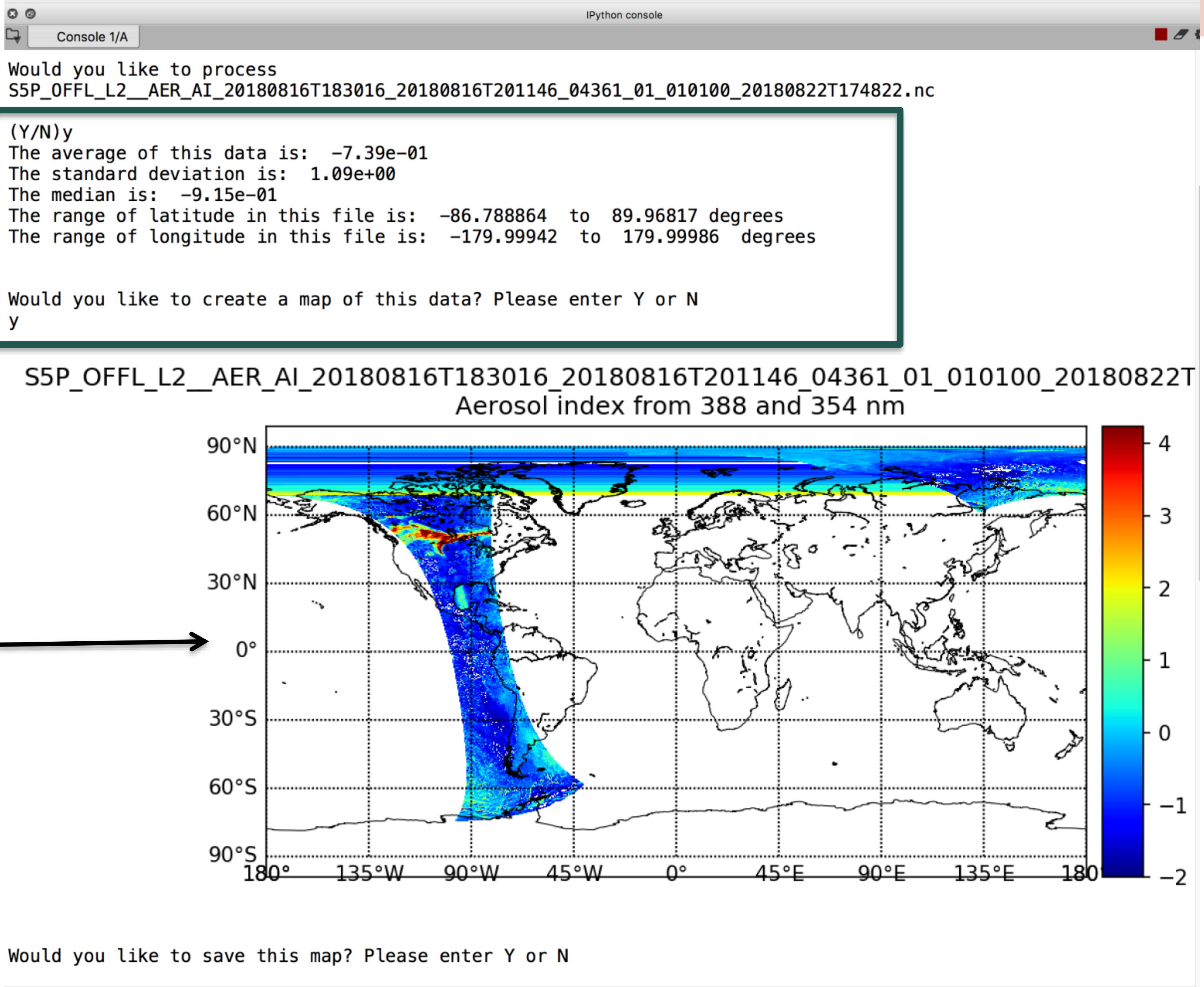
```
Python 3.6.5 |Anaconda, Inc.| (default, Mar 29 2018, 13:14:23)
Type "copyright", "credits" or "license" for more information.

IPython 6.4.0 -- An enhanced Interactive Python.

Restarting kernel...



In [1]:
```

Permissions: RW   End-of-lines: CRLF   Encoding: ASCII   Line: 1   Column: 1   Memory: 40 %

Running and Output

AI/NO_2 Statistics

Output map



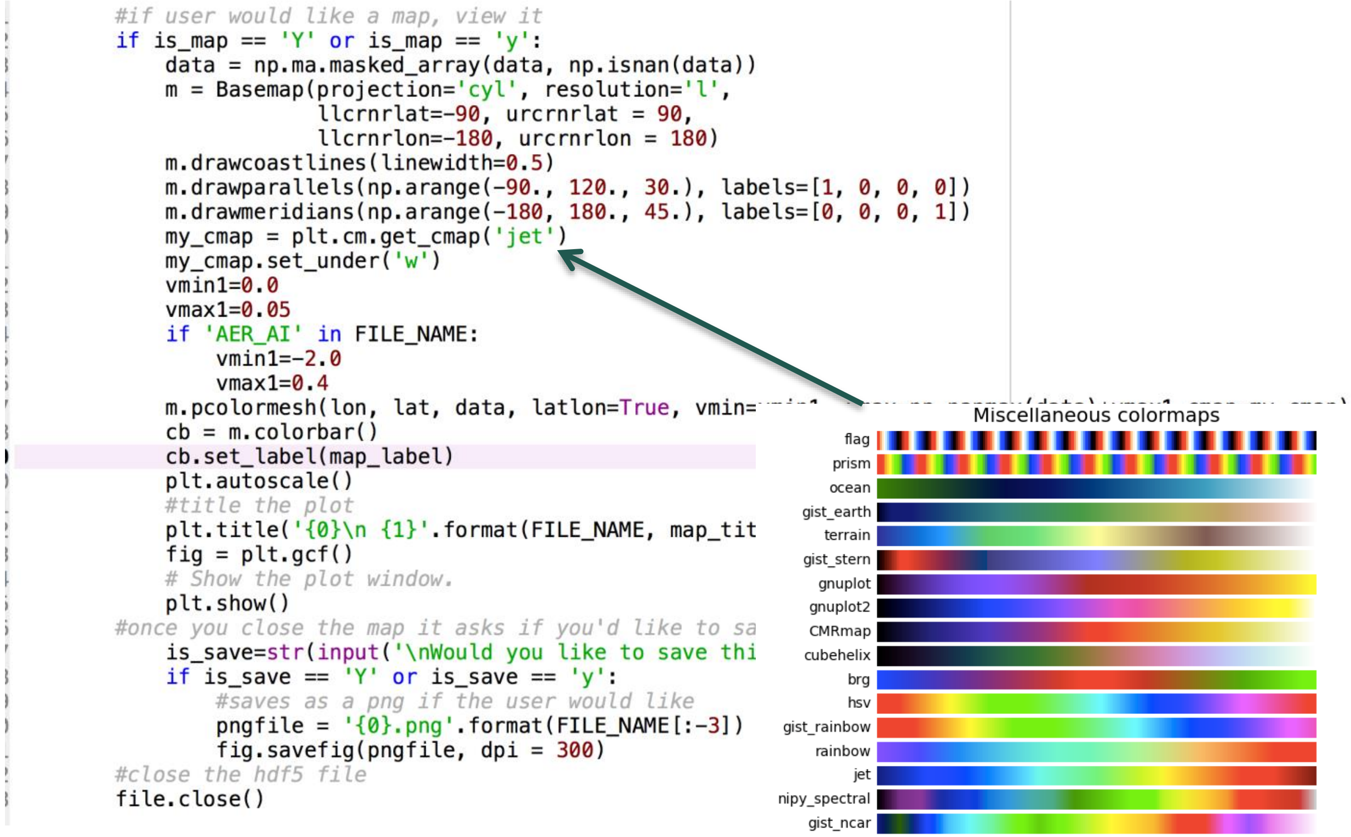
```
Would you like to process
S5P_OFFL_L2__AER_AI_20180816T183016_20180816T201146_04361_01_010100_20180822T174822.nc
(Y/N)y
The average of this data is:  -7.39e-01
The standard deviation is:  1.09e+00
The median is:  -9.15e-01
The range of latitude in this file is:  -86.788864  to  89.96817 degrees
The range of longitude in this file is:  -179.99942  to  179.99986  degrees


Would you like to create a map of this data? Please enter Y or N
y
```

S5P_OFFL_L2__AER_AI_20180816T183016_20180816T201146_04361_01_010100_20180822T
Aerosol index from 388 and 354 nm
90°N
60°N
30°N
0°
30°S
60°S
90°S
180°
135°W
90°W
45°W
0°
45°E
90°E
135°E
180°
4
3
2
1
0
−1
−2

```
Would you like to save this map? Please enter Y or N
```

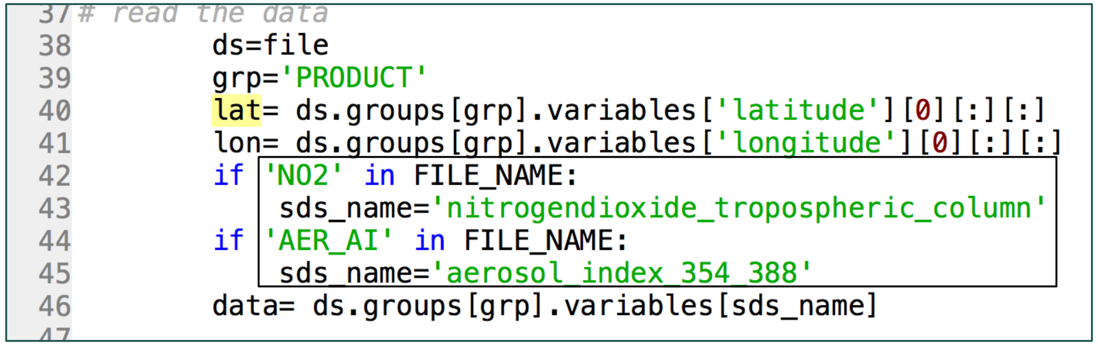
Editing the Code

Change the color scale



```
#if user would like a map, view it
if is_map == 'Y' or is_map == 'y':
    data = np.ma.masked_array(data, np.isnan(data))
    m = Basemap(projection='cyl', resolution='l',
                llcrnrlat=-90, urcrnrlat = 90,
                llcrnrlon=-180, urcrnrlon = 180)
    m.drawcoastlines(linewidth=0.5)
    m.drawparallels(np.arange(-90., 120., 30.), labels=[1, 0, 0, 0])
    m.drawmeridians(np.arange(-180, 180., 45.), labels=[0, 0, 0, 1])
    my_cmap = plt.cm.get_cmap('jet')
    my_cmap.set_under('w')
    vmin1=0.0
    vmax1=0.05
    if 'AER_AI' in FILE_NAME:
        vmin1=-2.0
        vmax1=0.4
    m.pcolormesh(lon, lat, data, latlon=True, vmin=
    cb = m.colorbar()
    cb.set_label(map_label)
    plt.autoscale()
    #title the plot
    plt.title('{0}\n {1}'.format(FILE_NAME, map_tit
    fig = plt.gcf()
    # Show the plot window.
    plt.show()
#once you close the map it asks if you'd like to sa
    is_save=str(input('\nWould you like to save thi
    if is_save == 'Y' or is_save == 'y':
        #saves as a png if the user would like
        pngfile = '{0}.png'.format(FILE_NAME[:-3])
        fig.savefig(pngfile, dpi = 300)
#close the hdf5 file
file.close()
```

Change the SDS to plot

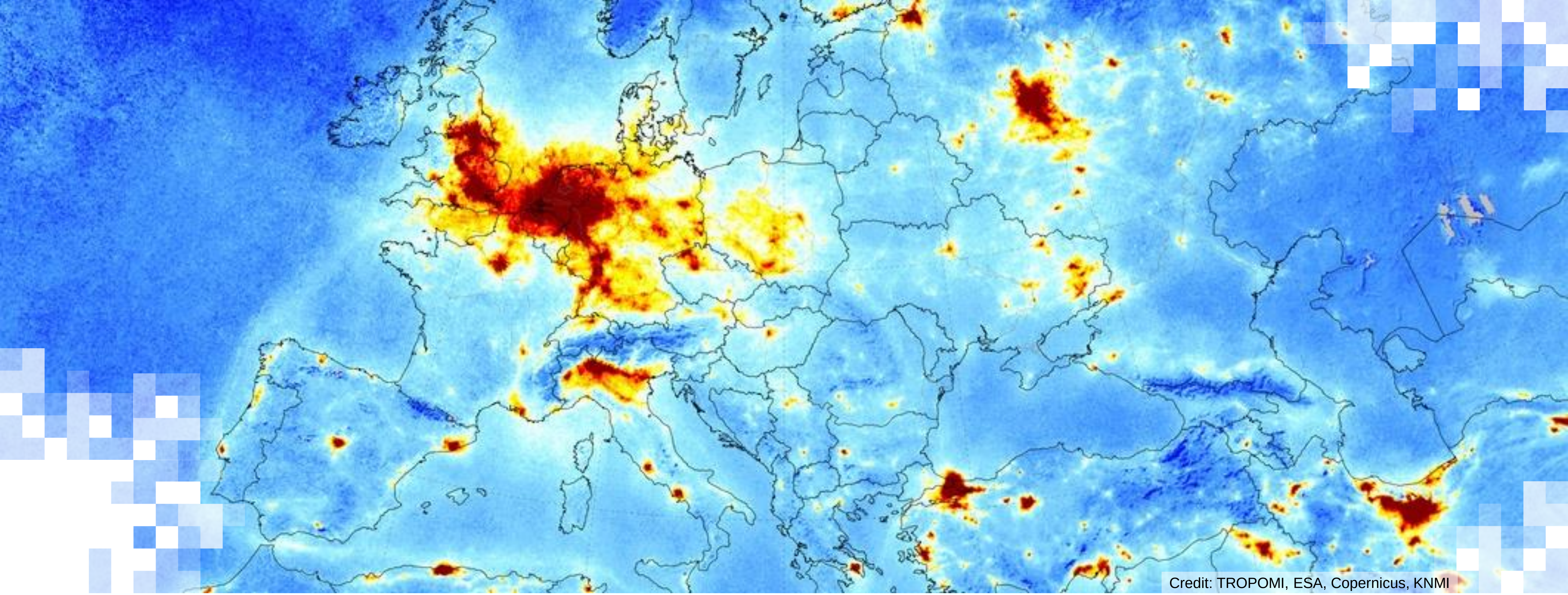


```
# read the data
        ds=file
        grp='PRODUCT'
        lat= ds.groups[grp].variables['latitude'][0][:][:]
        lon= ds.groups[grp].variables['longitude'][0][:][:]
        if 'NO2' in FILE_NAME:
            sds_name='nitrogendioxide_tropospheric_column'
        if 'AER_AI' in FILE_NAME:
            sds_name='aerosol_index_354_388'
        data= ds.groups[grp].variables[sds_name]
```

https://matplotlib.org/examples/color/colormaps_reference.html

Applications

- This is a sample code to read and map the TROPOMI Level 2 NO_2 and AI data
- The code can be modified to address various mapping needs
- User can create daily maps of trace gas columns over certain regions and start analyzing changes over time
- These maps can also help identify regions of high pollution



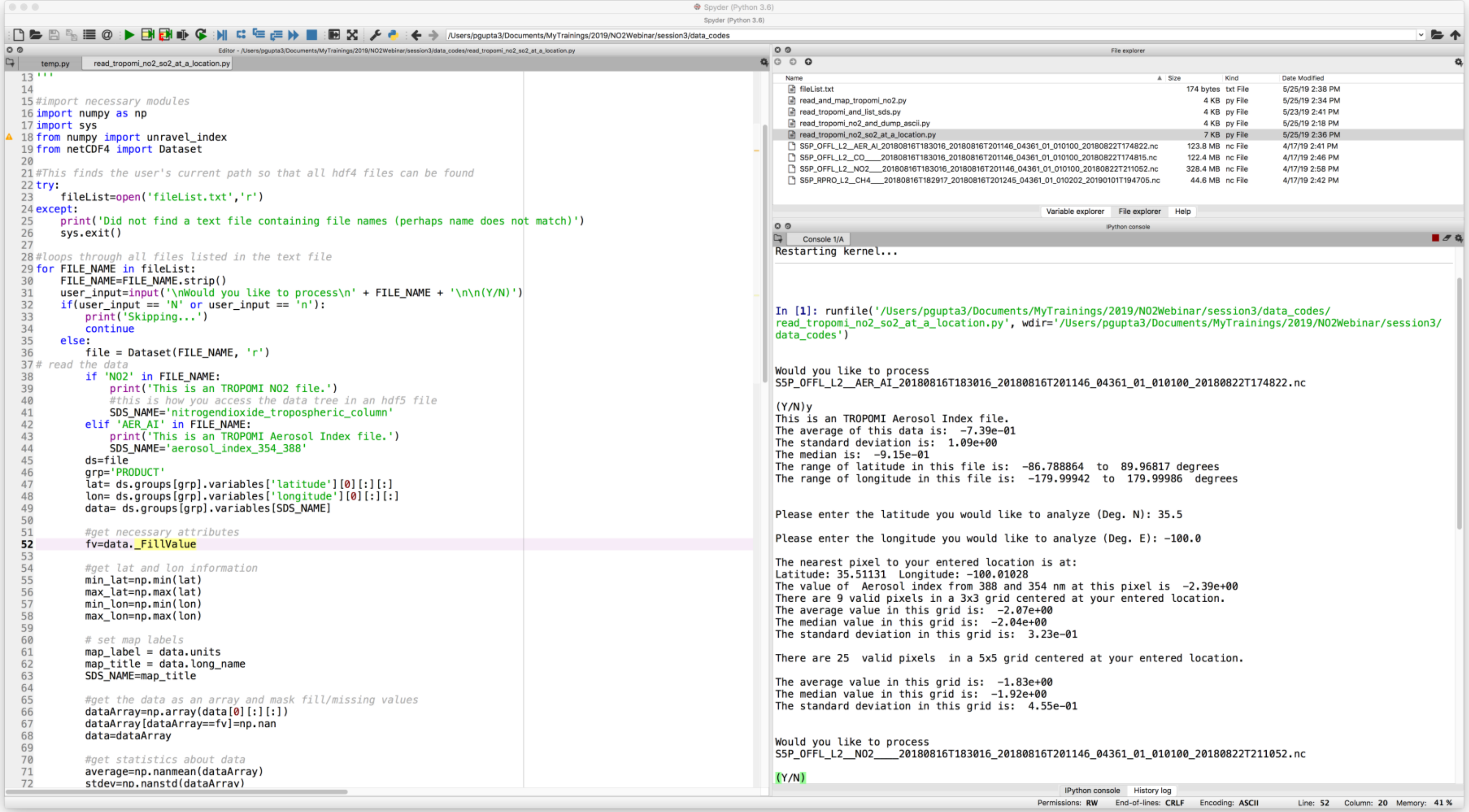
Credit: TROPOMI, ESA, Copernicus, KNMI

Extract NO_2/AI at a given location

Extract NO_2 Values from TROPOMI Level 2 Data

read_tropomi_no2_ai_at_a_location.py

- **Purpose**: read a TROPOMI NO_2/AI level 2 data file in nc format and extract values at a given ground location



Spyder (Python 3.6)
/Users/pgupta3/Documents/MyTrainings/2019/NO2Webinar/session3/data_codes
Editor - /Users/pgupta3/Documents/MyTrainings/2019/NO2Webinar/session3/data_codes/read_tropomi_no2_so2_at_a_location.py
temp.py
read_tropomi_no2_so2_at_a_location.py
File explorer
Variable explorer
File explorer
Help
IPython console
Console 1/A
IPython console
History log
Permissions: RW
End-of-lines: CRLF
Encoding: ASCII
Line: 52
Column: 20
Memory: 41 %

```python
'''

#import necessary modules
import numpy as np
import sys
from numpy import unravel_index
from netCDF4 import Dataset

#This finds the user's current path so that all hdf4 files can be found
try:
    fileList=open('fileList.txt','r')
except:
    print('Did not find a text file containing file names (perhaps name does not match)')
    sys.exit()

#loops through all files listed in the text file
for FILE_NAME in fileList:
    FILE_NAME=FILE_NAME.strip()
    user_input=input('\nWould you like to process\n' + FILE_NAME + '\n\n(Y/N)')
    if(user_input == 'N' or user_input == 'n'):
        print('Skipping...')
        continue
    else:
        file = Dataset(FILE_NAME, 'r')
# read the data
        if 'NO2' in FILE_NAME:
            print('This is an TROPOMI NO2 file.')
            #this is how you access the data tree in an hdf5 file
            SDS_NAME='nitrogendioxide_tropospheric_column'
        elif 'AER_AI' in FILE_NAME:
            print('This is an TROPOMI Aerosol Index file.')
            SDS_NAME='aerosol_index_354_388'
        ds=file
        grp='PRODUCT'
        lat= ds.groups[grp].variables['latitude'][0][:][:]
        lon= ds.groups[grp].variables['longitude'][0][:][:]
        data= ds.groups[grp].variables[SDS_NAME]

        #get necessary attributes
        fv=data._FillValue

        #get lat and lon information
        min_lat=np.min(lat)
        max_lat=np.max(lat)
        min_lon=np.min(lon)
        max_lon=np.max(lon)

        # set map labels
        map_label = data.units
        map_title = data.long_name
        SDS_NAME=map_title

        #get the data as an array and mask fill/missing values
        dataArray=np.array(data[0][:][:])
        dataArray[dataArray==fv]=np.nan
        data=dataArray

        #get statistics about data
        average=np.nanmean(dataArray)
        stdev=np.nanstd(dataArray)
```

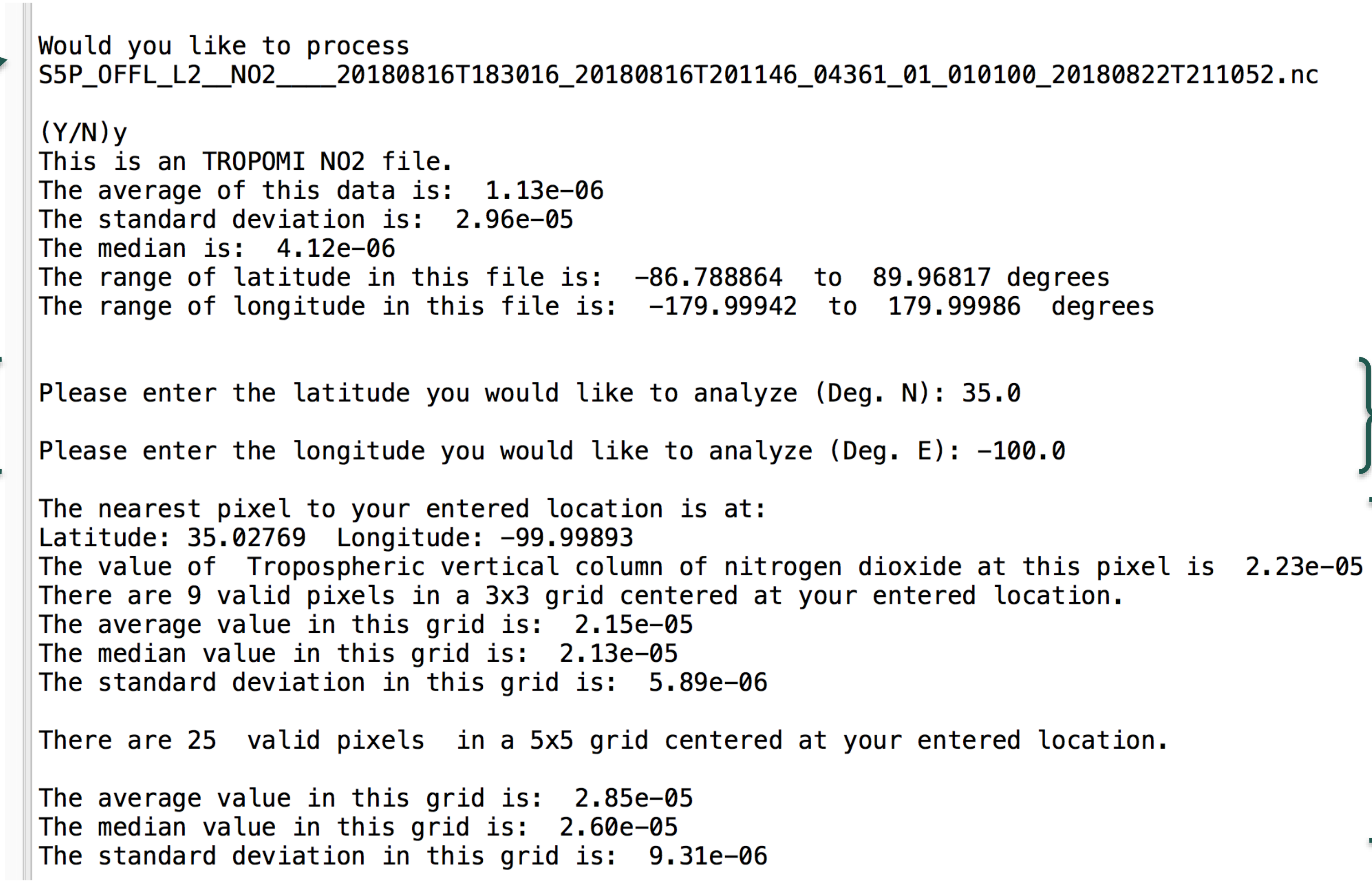
Name	Size	Kind	Date Modified
fileList.txt	174 bytes	txt File	5/25/19 2:38 PM
read_and_map_tropomi_no2.py	4 KB	py File	5/25/19 2:34 PM
read_tropomi_and_list_sds.py	4 KB	py File	5/23/19 2:41 PM
read_tropomi_no2_and_dump_ascii.py	4 KB	py File	5/25/19 2:18 PM
read_tropomi_no2_so2_at_a_location.py	7 KB	py File	5/25/19 2:36 PM
S5P_OFFL_L2__AER_AI_20180816T183016_20180816T201146_04361_01_010100_20180822T174822.nc	123.8 MB	nc File	4/17/19 2:41 PM
S5P_OFFL_L2__CO_____20180816T183016_20180816T201146_04361_01_010100_20180822T174815.nc	122.4 MB	nc File	4/17/19 2:46 PM
S5P_OFFL_L2__NO2____20180816T183016_20180816T201146_04361_01_010100_20180822T211052.nc	328.4 MB	nc File	4/17/19 2:58 PM
S5P_RPRO_L2__CH4____20180816T182917_20180816T201245_04361_01_010202_20190101T194705.nc	44.6 MB	nc File	4/17/19 2:42 PM

```
Restarting kernel...



In [1]: runfile('/Users/pgupta3/Documents/MyTrainings/2019/NO2Webinar/session3/data_codes/
read_tropomi_no2_so2_at_a_location.py', wdir='/Users/pgupta3/Documents/MyTrainings/2019/NO2Webinar/session3/
data_codes')


Would you like to process
S5P_OFFL_L2__AER_AI_20180816T183016_20180816T201146_04361_01_010100_20180822T174822.nc

(Y/N)y
This is an TROPOMI Aerosol Index file.
The average of this data is:  -7.39e-01
The standard deviation is:  1.09e+00
The median is:  -9.15e-01
The range of latitude in this file is:  -86.788864  to  89.96817 degrees
The range of longitude in this file is:  -179.99942  to  179.99986  degrees


Please enter the latitude you would like to analyze (Deg. N): 35.5

Please enter the longitude you would like to analyze (Deg. E): -100.0

The nearest pixel to your entered location is at:
Latitude: 35.51131  Longitude: -100.01028
The value of  Aerosol index from 388 and 354 nm at this pixel is  -2.39e+00
There are 9 valid pixels in a 3x3 grid centered at your entered location.
The average value in this grid is:  -2.07e+00
The median value in this grid is:  -2.04e+00
The standard deviation in this grid is:  3.23e-01

There are 25  valid pixels  in a 5x5 grid centered at your entered location.

The average value in this grid is:  -1.83e+00
The median value in this grid is:  -1.92e+00
The standard deviation in this grid is:  4.55e-01


Would you like to process
S5P_OFFL_L2__NO2____20180816T183016_20180816T201146_04361_01_010100_20180822T211052.nc

(Y/N)
```

Running and Output

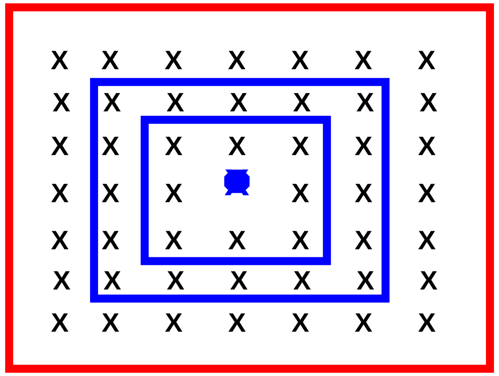
Type “Y” to process file, “N” to skip

Lat & Lon of station

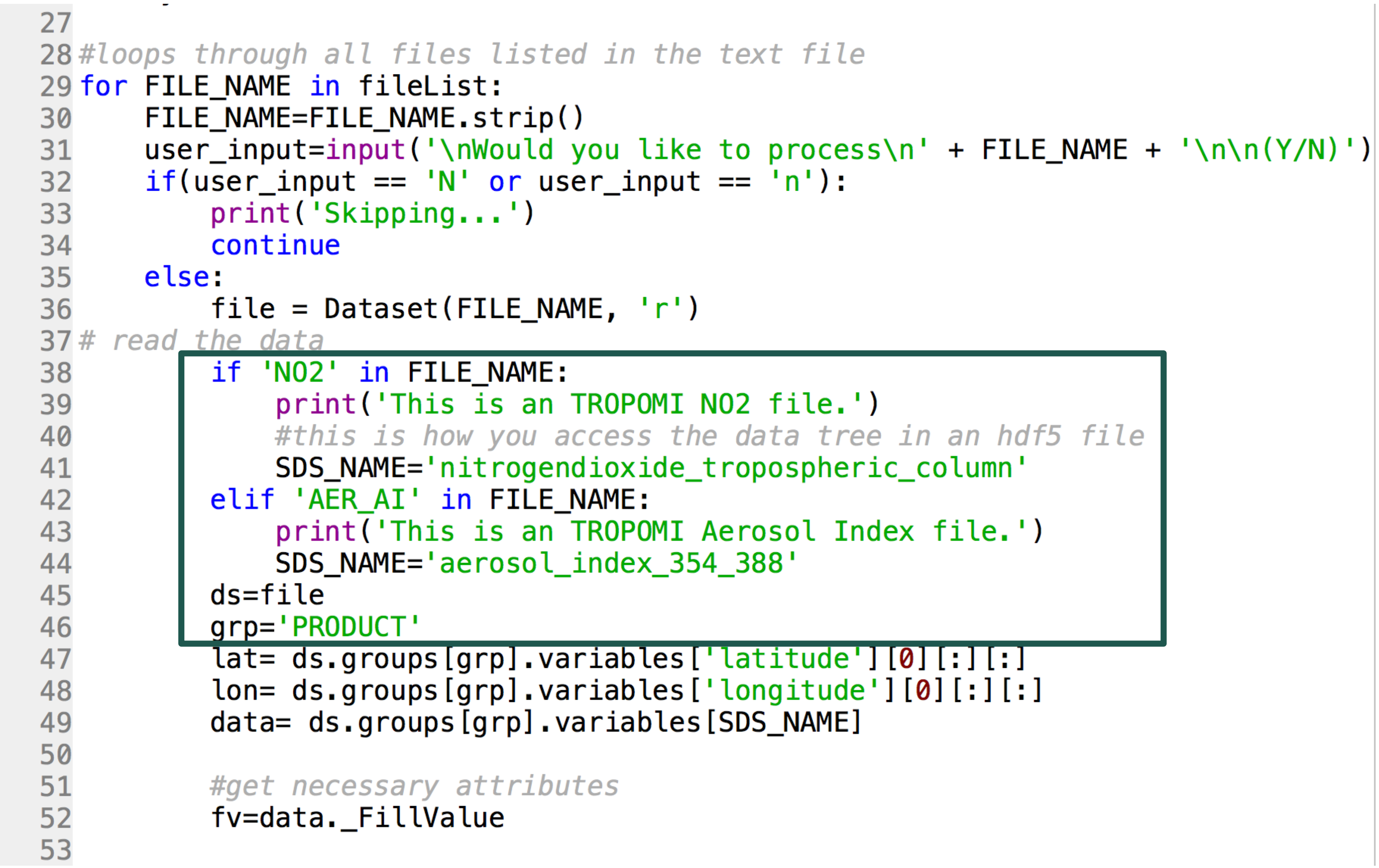


```
Would you like to process
S5P_OFFL_L2__NO2____20180816T183016_20180816T201146_04361_01_010100_20180822T211052.nc

(Y/N)y
This is an TROPOMI NO2 file.
The average of this data is:  1.13e-06
The standard deviation is:  2.96e-05
The median is:  4.12e-06
The range of latitude in this file is:  -86.788864  to  89.96817 degrees
The range of longitude in this file is:  -179.99942  to  179.99986  degrees


Please enter the latitude you would like to analyze (Deg. N): 35.0

Please enter the longitude you would like to analyze (Deg. E): -100.0

The nearest pixel to your entered location is at:
Latitude: 35.02769  Longitude: -99.99893
The value of  Tropospheric vertical column of nitrogen dioxide at this pixel is  2.23e-05
There are 9 valid pixels in a 3x3 grid centered at your entered location.
The average value in this grid is:  2.15e-05
The median value in this grid is:  2.13e-05
The standard deviation in this grid is:  5.89e-06

There are 25  valid pixels  in a 5x5 grid centered at your entered location.

The average value in this grid is:  2.85e-05
The median value in this grid is:  2.60e-05
The standard deviation in this grid is:  9.31e-06
```

Outputs



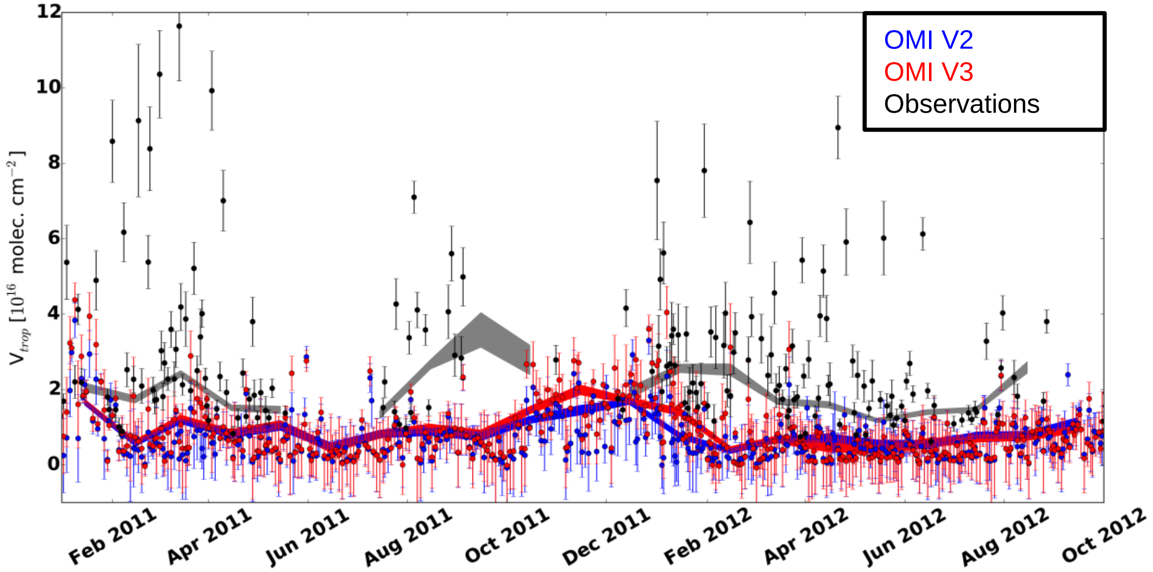
Editing the Code – Change the SDS



```

#loops through all files listed in the text file
for FILE_NAME in fileList:
    FILE_NAME=FILE_NAME.strip()
    user_input=input('\nWould you like to process\n' + FILE_NAME + '\n\n(Y/N)')
    if(user_input == 'N' or user_input == 'n'):
        print('Skipping...')
        continue
    else:
        file = Dataset(FILE_NAME, 'r')
# read the data
        if 'NO2' in FILE_NAME:
            print('This is an TROPOMI NO2 file.')
            #this is how you access the data tree in an hdf5 file
            SDS_NAME='nitrogendioxide_tropospheric_column'
        elif 'AER_AI' in FILE_NAME:
            print('This is an TROPOMI Aerosol Index file.')
            SDS_NAME='aerosol_index_354_388'
        ds=file
        grp='PRODUCT'
        lat= ds.groups[grp].variables['latitude'][0][:][:]
        lon= ds.groups[grp].variables['longitude'][0][:][:]
        data= ds.groups[grp].variables[SDS_NAME]

        #get necessary attributes
        fv=data._FillValue

```

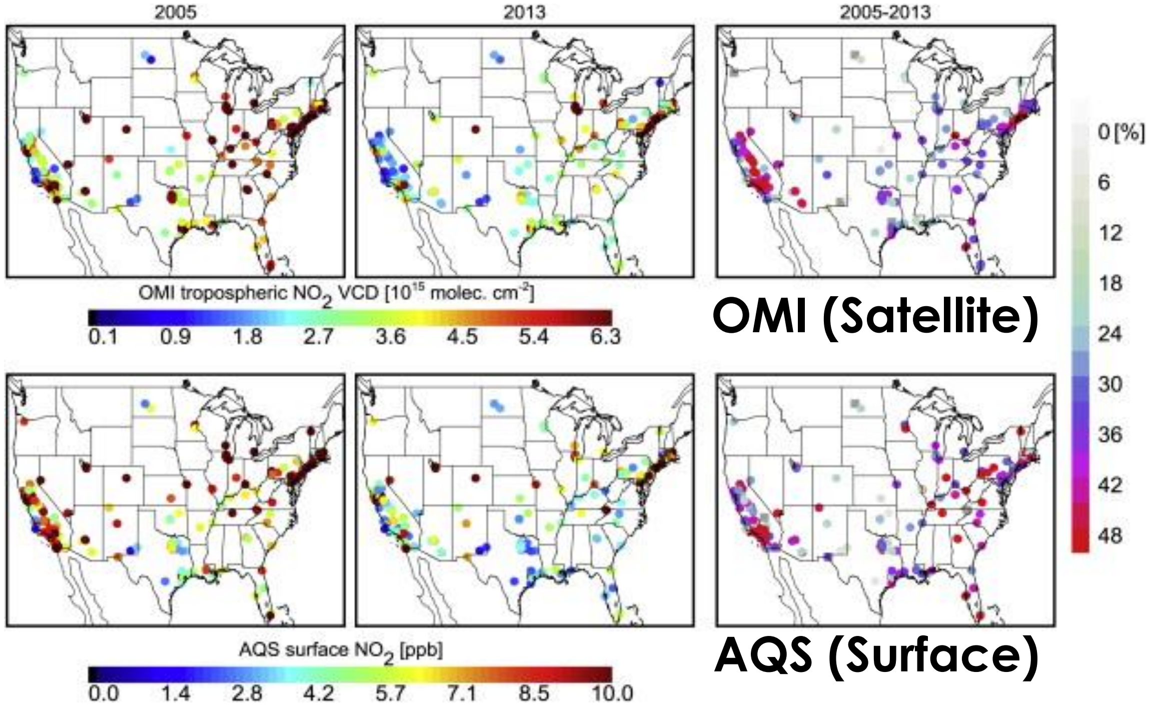
Applications

Satellite Validation

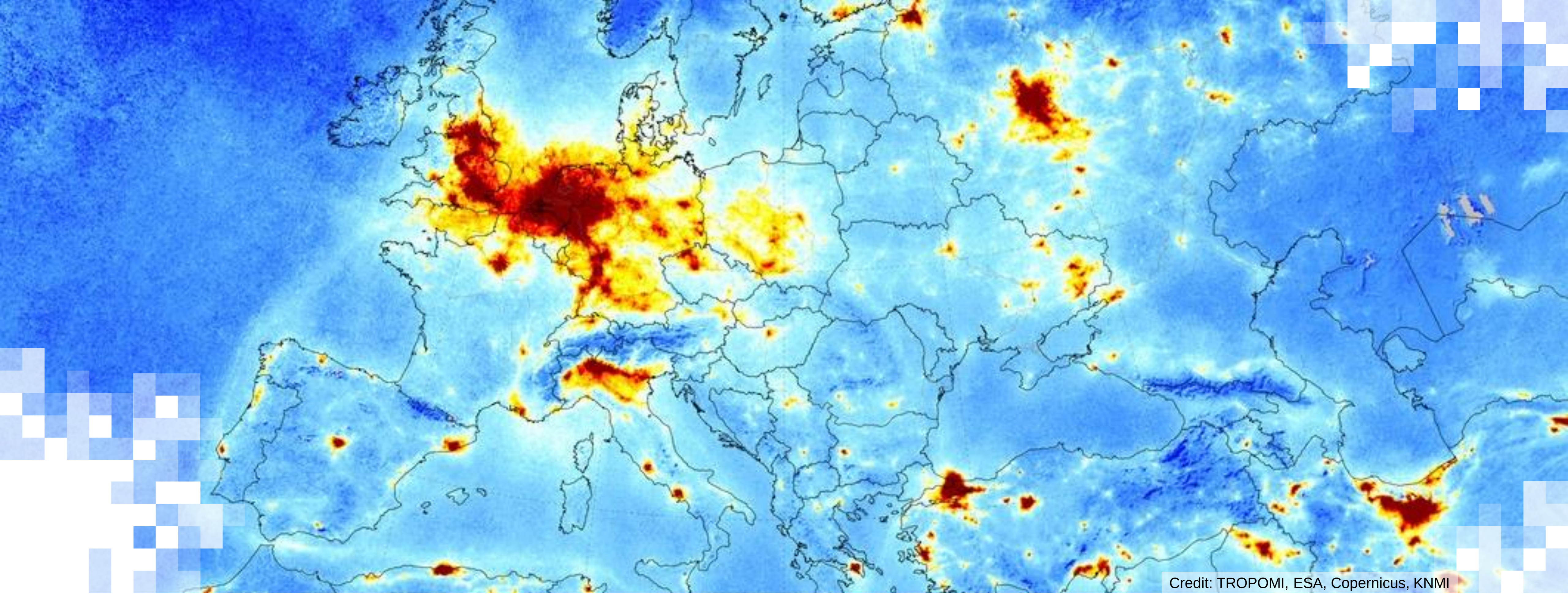


Source: Krotkov et al. (2017)

Column vs. Surface Relationship and Trends



Source: Lamsal, L.N. et al. (2016)



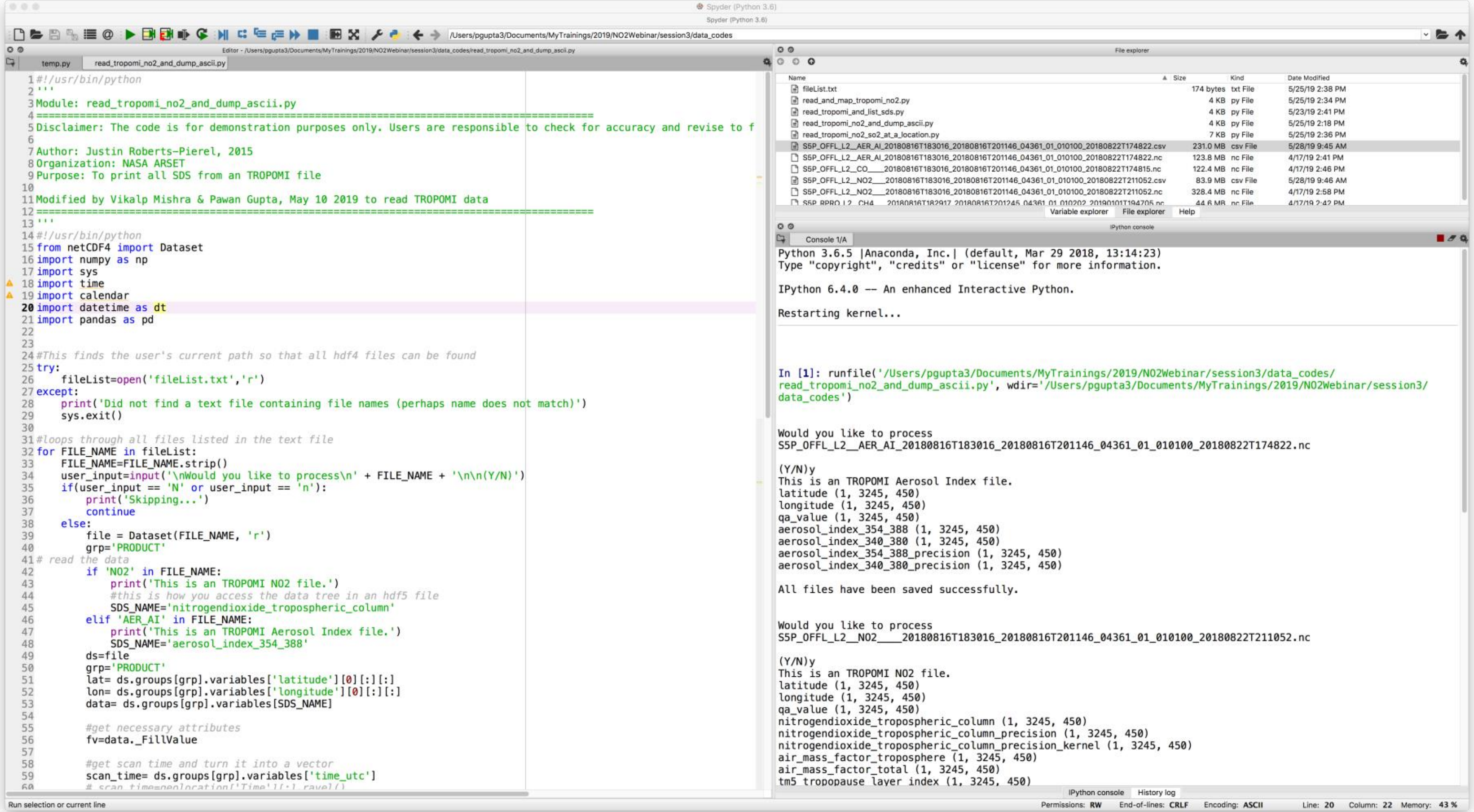
Credit: TROPOMI, ESA, Copernicus, KNMI

Output nc variables to CSV

Output TROPOMI NO_2/AI nc variables to a CSV file

read_tropomi_no2_ai_and_dump_ascii.py

- **Purpose**: read a TROPOMI level 2 NO_2 or AI data file in netCDF format and write certain SDSs into a csv (text) file



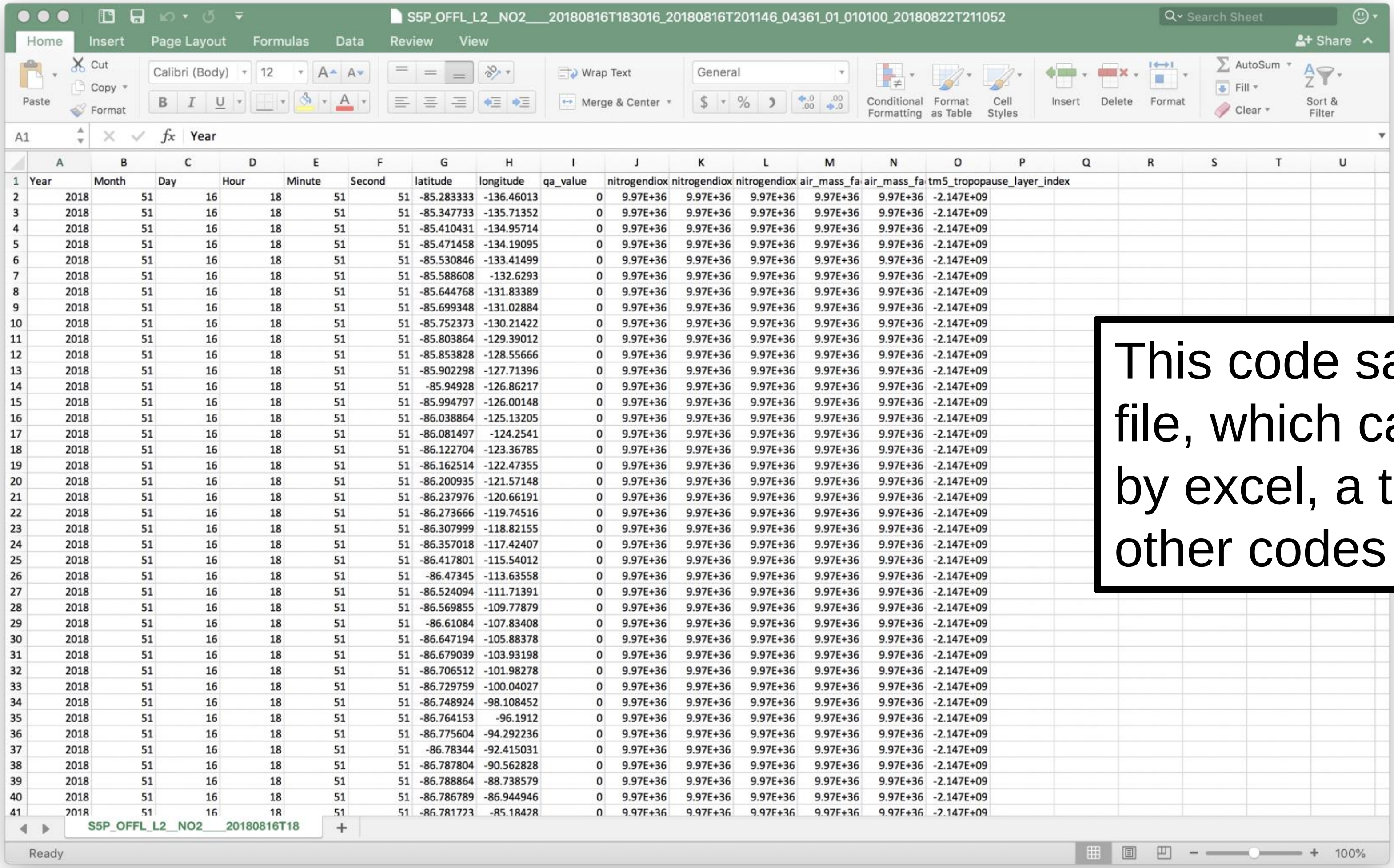
Spyder (Python 3.6)
/Users/pgupta3/Documents/MyTrainings/2019/NO2Webinar/session3/data_codes
Editor - /Users/pgupta3/Documents/MyTrainings/2019/NO2Webinar/session3/data_codes/read_tropomi_no2_and_dump_ascii.py
temp.py
read_tropomi_no2_and_dump_ascii.py

```python
#!/usr/bin/python
'''
Module: read_tropomi_no2_and_dump_ascii.py
==========================================================================================
Disclaimer: The code is for demonstration purposes only. Users are responsible to check for accuracy and revise to f

Author: Justin Roberts-Pierel, 2015
Organization: NASA ARSET
Purpose: To print all SDS from an TROPOMI file

Modified by Vikalp Mishra & Pawan Gupta, May 10 2019 to read TROPOMI data
==========================================================================================
'''
#!/usr/bin/python
from netCDF4 import Dataset
import numpy as np
import sys
import time
import calendar
import datetime as dt
import pandas as pd


#This finds the user's current path so that all hdf4 files can be found
try:
    fileList=open('fileList.txt','r')
except:
    print('Did not find a text file containing file names (perhaps name does not match)')
    sys.exit()

#loops through all files listed in the text file
for FILE_NAME in fileList:
    FILE_NAME=FILE_NAME.strip()
    user_input=input('\nWould you like to process\n' + FILE_NAME + '\n\n(Y/N)')
    if(user_input == 'N' or user_input == 'n'):
        print('Skipping...')
        continue
    else:
        file = Dataset(FILE_NAME, 'r')
        grp='PRODUCT'
# read the data
        if 'NO2' in FILE_NAME:
            print('This is an TROPOMI NO2 file.')
            #this is how you access the data tree in an hdf5 file
            SDS_NAME='nitrogendioxide_tropospheric_column'
        elif 'AER_AI' in FILE_NAME:
            print('This is an TROPOMI Aerosol Index file.')
            SDS_NAME='aerosol_index_354_388'
        ds=file
        grp='PRODUCT'
        lat= ds.groups[grp].variables['latitude'][0][:][:]
        lon= ds.groups[grp].variables['longitude'][0][:][:]
        data= ds.groups[grp].variables[SDS_NAME]

        #get necessary attributes
        fv=data._FillValue

        #get scan time and turn it into a vector
        scan_time= ds.groups[grp].variables['time_utc']
        # scan_time=geolocation['Time'][:].ravel()
```

Name	Size	Kind	Date Modified
fileList.txt	174 bytes	txt File	5/25/19 2:38 PM
read_and_map_tropomi_no2.py	4 KB	py File	5/25/19 2:34 PM
read_tropomi_and_list_sds.py	4 KB	py File	5/23/19 2:41 PM
read_tropomi_no2_and_dump_ascii.py	4 KB	py File	5/25/19 2:18 PM
read_tropomi_no2_so2_at_a_location.py	7 KB	py File	5/25/19 2:36 PM
S5P_OFFL_L2__AER_AI_20180816T183016_20180816T201146_04361_01_010100_20180822T174822.csv	231.0 MB	csv File	5/28/19 9:45 AM
S5P_OFFL_L2__AER_AI_20180816T183016_20180816T201146_04361_01_010100_20180822T174822.nc	123.8 MB	nc File	4/17/19 2:41 PM
S5P_OFFL_L2__CO_____20180816T183016_20180816T201146_04361_01_010100_20180822T174815.nc	122.4 MB	nc File	4/17/19 2:46 PM
S5P_OFFL_L2__NO2____20180816T183016_20180816T201146_04361_01_010100_20180822T211052.csv	83.9 MB	csv File	5/28/19 9:46 AM
S5P_OFFL_L2__NO2____20180816T183016_20180816T201146_04361_01_010100_20180822T211052.nc	328.4 MB	nc File	4/17/19 2:58 PM
S5P_RPRO_L2__CH4____20180816T182917_20180816T201245_04361_01_010202_20190101T194705.nc	44.6 MB	nc File	4/17/19 2:42 PM

```
Python 3.6.5 |Anaconda, Inc.| (default, Mar 29 2018, 13:14:23)
Type "copyright", "credits" or "license" for more information.

IPython 6.4.0 -- An enhanced Interactive Python.

Restarting kernel...



In [1]: runfile('/Users/pgupta3/Documents/MyTrainings/2019/NO2Webinar/session3/data_codes/
read_tropomi_no2_and_dump_ascii.py', wdir='/Users/pgupta3/Documents/MyTrainings/2019/NO2Webinar/session3/
data_codes')


Would you like to process
S5P_OFFL_L2__AER_AI_20180816T183016_20180816T201146_04361_01_010100_20180822T174822.nc

(Y/N)y
This is an TROPOMI Aerosol Index file.
latitude (1, 3245, 450)
longitude (1, 3245, 450)
qa_value (1, 3245, 450)
aerosol_index_354_388 (1, 3245, 450)
aerosol_index_340_380 (1, 3245, 450)
aerosol_index_354_388_precision (1, 3245, 450)
aerosol_index_340_380_precision (1, 3245, 450)

All files have been saved successfully.


Would you like to process
S5P_OFFL_L2__NO2____20180816T183016_20180816T201146_04361_01_010100_20180822T211052.nc

(Y/N)y
This is an TROPOMI NO2 file.
latitude (1, 3245, 450)
longitude (1, 3245, 450)
qa_value (1, 3245, 450)
nitrogendioxide_tropospheric_column (1, 3245, 450)
nitrogendioxide_tropospheric_column_precision (1, 3245, 450)
nitrogendioxide_tropospheric_column_precision_kernel (1, 3245, 450)
air_mass_factor_troposphere (1, 3245, 450)
air_mass_factor_total (1, 3245, 450)
tm5_tropopause_layer_index (1, 3245, 450)
```

Output



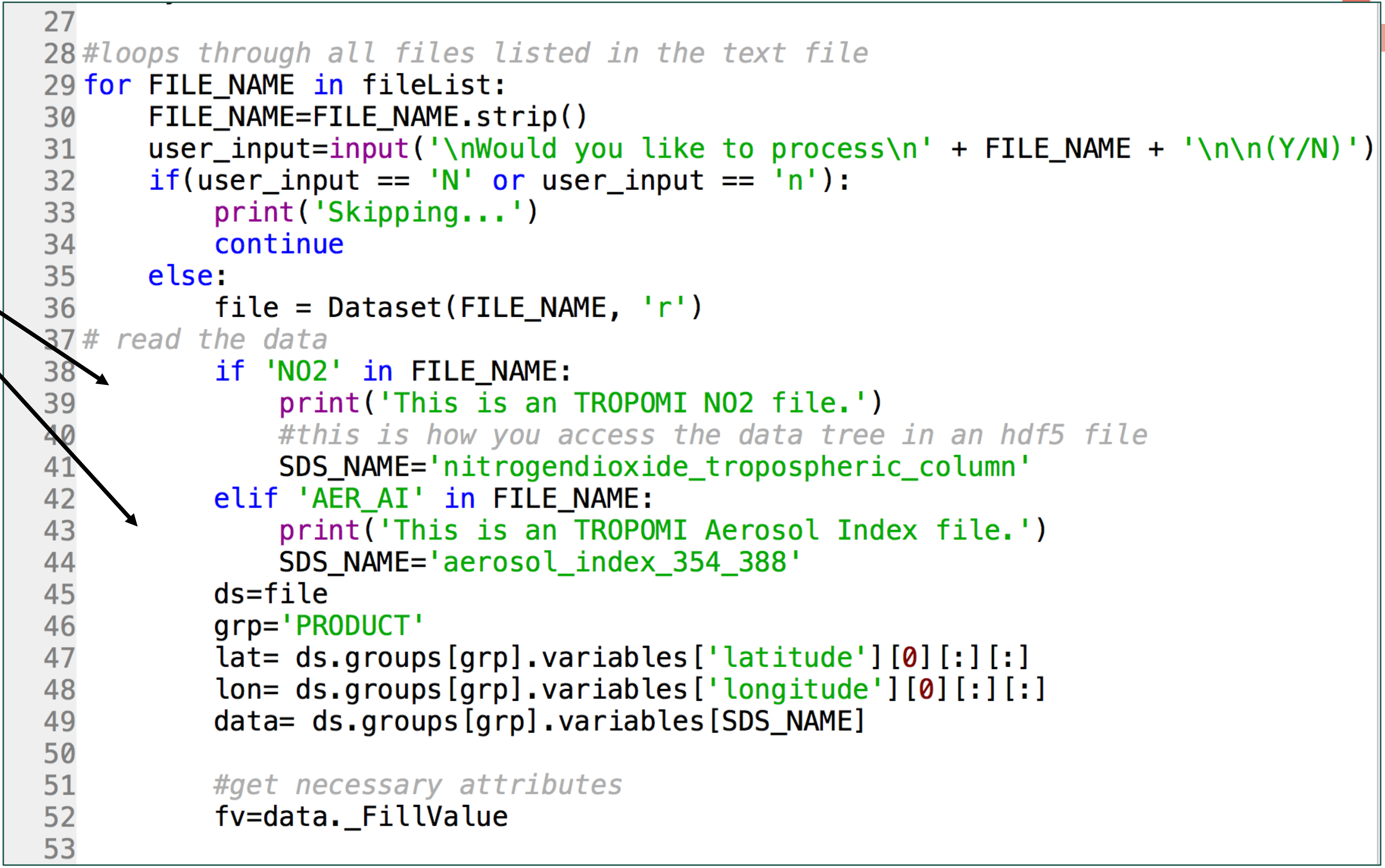
	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
1	Year	Month	Day	Hour	Minute	Second	latitude	longitude	qa_value	nitrogendiox	nitrogendiox	nitrogendiox	air_mass_fa	air_mass_fa	tm5_tropopause_layer_index
2	2018	51	16	18	51	51	-85.283333	-136.46013	0	9.97E+36	9.97E+36	9.97E+36	9.97E+36	9.97E+36	-2.147E+09
3	2018	51	16	18	51	51	-85.347733	-135.71352	0	9.97E+36	9.97E+36	9.97E+36	9.97E+36	9.97E+36	-2.147E+09
4	2018	51	16	18	51	51	-85.410431	-134.95714	0	9.97E+36	9.97E+36	9.97E+36	9.97E+36	9.97E+36	-2.147E+09
5	2018	51	16	18	51	51	-85.471458	-134.19095	0	9.97E+36	9.97E+36	9.97E+36	9.97E+36	9.97E+36	-2.147E+09
6	2018	51	16	18	51	51	-85.530846	-133.41499	0	9.97E+36	9.97E+36	9.97E+36	9.97E+36	9.97E+36	-2.147E+09
7	2018	51	16	18	51	51	-85.588608	-132.6293	0	9.97E+36	9.97E+36	9.97E+36	9.97E+36	9.97E+36	-2.147E+09
8	2018	51	16	18	51	51	-85.644768	-131.83389	0	9.97E+36	9.97E+36	9.97E+36	9.97E+36	9.97E+36	-2.147E+09
9	2018	51	16	18	51	51	-85.699348	-131.02884	0	9.97E+36	9.97E+36	9.97E+36	9.97E+36	9.97E+36	-2.147E+09
10	2018	51	16	18	51	51	-85.752373	-130.21422	0	9.97E+36	9.97E+36	9.97E+36	9.97E+36	9.97E+36	-2.147E+09
11	2018	51	16	18	51	51	-85.803864	-129.39012	0	9.97E+36	9.97E+36	9.97E+36	9.97E+36	9.97E+36	-2.147E+09
12	2018	51	16	18	51	51	-85.853828	-128.55666	0	9.97E+36	9.97E+36	9.97E+36	9.97E+36	9.97E+36	-2.147E+09
13	2018	51	16	18	51	51	-85.902298	-127.71396	0	9.97E+36	9.97E+36	9.97E+36	9.97E+36	9.97E+36	-2.147E+09
14	2018	51	16	18	51	51	-85.94928	-126.86217	0	9.97E+36	9.97E+36	9.97E+36	9.97E+36	9.97E+36	-2.147E+09
15	2018	51	16	18	51	51	-85.994797	-126.00148	0	9.97E+36	9.97E+36	9.97E+36	9.97E+36	9.97E+36	-2.147E+09
16	2018	51	16	18	51	51	-86.038864	-125.13205	0	9.97E+36	9.97E+36	9.97E+36	9.97E+36	9.97E+36	-2.147E+09
17	2018	51	16	18	51	51	-86.081497	-124.2541	0	9.97E+36	9.97E+36	9.97E+36	9.97E+36	9.97E+36	-2.147E+09
18	2018	51	16	18	51	51	-86.122704	-123.36785	0	9.97E+36	9.97E+36	9.97E+36	9.97E+36	9.97E+36	-2.147E+09
19	2018	51	16	18	51	51	-86.162514	-122.47355	0	9.97E+36	9.97E+36	9.97E+36	9.97E+36	9.97E+36	-2.147E+09
20	2018	51	16	18	51	51	-86.200935	-121.57148	0	9.97E+36	9.97E+36	9.97E+36	9.97E+36	9.97E+36	-2.147E+09
21	2018	51	16	18	51	51	-86.237976	-120.66191	0	9.97E+36	9.97E+36	9.97E+36	9.97E+36	9.97E+36	-2.147E+09
22	2018	51	16	18	51	51	-86.273666	-119.74516	0	9.97E+36	9.97E+36	9.97E+36	9.97E+36	9.97E+36	-2.147E+09
23	2018	51	16	18	51	51	-86.307999	-118.82155	0	9.97E+36	9.97E+36	9.97E+36	9.97E+36	9.97E+36	-2.147E+09
24	2018	51	16	18	51	51	-86.357018	-117.42407	0	9.97E+36	9.97E+36	9.97E+36	9.97E+36	9.97E+36	-2.147E+09
25	2018	51	16	18	51	51	-86.417801	-115.54012	0	9.97E+36	9.97E+36	9.97E+36	9.97E+36	9.97E+36	-2.147E+09
26	2018	51	16	18	51	51	-86.47345	-113.63558	0	9.97E+36	9.97E+36	9.97E+36	9.97E+36	9.97E+36	-2.147E+09
27	2018	51	16	18	51	51	-86.524094	-111.71391	0	9.97E+36	9.97E+36	9.97E+36	9.97E+36	9.97E+36	-2.147E+09
28	2018	51	16	18	51	51	-86.569855	-109.77879	0	9.97E+36	9.97E+36	9.97E+36	9.97E+36	9.97E+36	-2.147E+09
29	2018	51	16	18	51	51	-86.61084	-107.83408	0	9.97E+36	9.97E+36	9.97E+36	9.97E+36	9.97E+36	-2.147E+09
30	2018	51	16	18	51	51	-86.647194	-105.88378	0	9.97E+36	9.97E+36	9.97E+36	9.97E+36	9.97E+36	-2.147E+09
31	2018	51	16	18	51	51	-86.679039	-103.93198	0	9.97E+36	9.97E+36	9.97E+36	9.97E+36	9.97E+36	-2.147E+09
32	2018	51	16	18	51	51	-86.706512	-101.98278	0	9.97E+36	9.97E+36	9.97E+36	9.97E+36	9.97E+36	-2.147E+09
33	2018	51	16	18	51	51	-86.729759	-100.04027	0	9.97E+36	9.97E+36	9.97E+36	9.97E+36	9.97E+36	-2.147E+09
34	2018	51	16	18	51	51	-86.748924	-98.108452	0	9.97E+36	9.97E+36	9.97E+36	9.97E+36	9.97E+36	-2.147E+09
35	2018	51	16	18	51	51	-86.764153	-96.1912	0	9.97E+36	9.97E+36	9.97E+36	9.97E+36	9.97E+36	-2.147E+09
36	2018	51	16	18	51	51	-86.775604	-94.292236	0	9.97E+36	9.97E+36	9.97E+36	9.97E+36	9.97E+36	-2.147E+09
37	2018	51	16	18	51	51	-86.78344	-92.415031	0	9.97E+36	9.97E+36	9.97E+36	9.97E+36	9.97E+36	-2.147E+09
38	2018	51	16	18	51	51	-86.787804	-90.562828	0	9.97E+36	9.97E+36	9.97E+36	9.97E+36	9.97E+36	-2.147E+09
39	2018	51	16	18	51	51	-86.788864	-88.738579	0	9.97E+36	9.97E+36	9.97E+36	9.97E+36	9.97E+36	-2.147E+09
40	2018	51	16	18	51	51	-86.786789	-86.944946	0	9.97E+36	9.97E+36	9.97E+36	9.97E+36	9.97E+36	-2.147E+09
41	2018	51	16	18	51	51	-86.781723	-85.18428	0	9.97E+36	9.97E+36	9.97E+36	9.97E+36	9.97E+36	-2.147E+09

This code saves a .csv file, which can be opened by excel, a text editor, or other codes or software

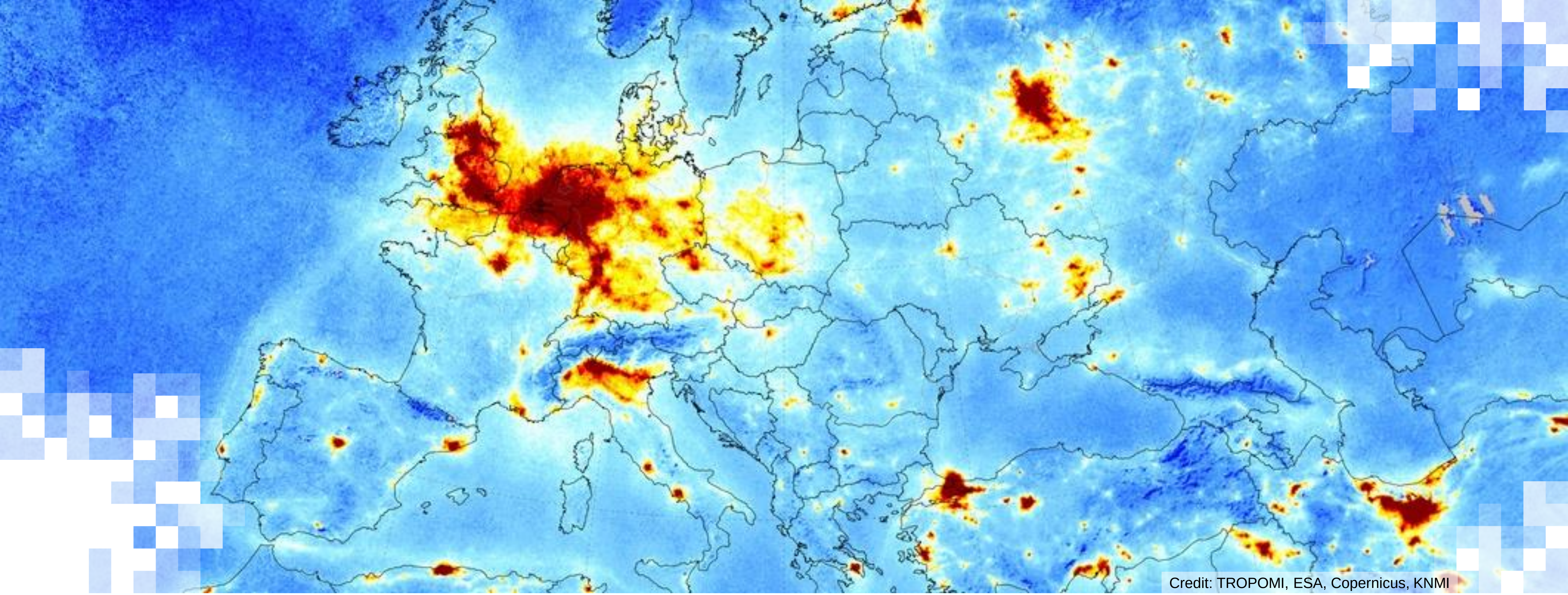
Editing the Code

Change the SDS SDS to be written as output

NOTE: This code will only work when all the variables listed are the same dimension. Use the “list SDS” code to view the variable dimensions

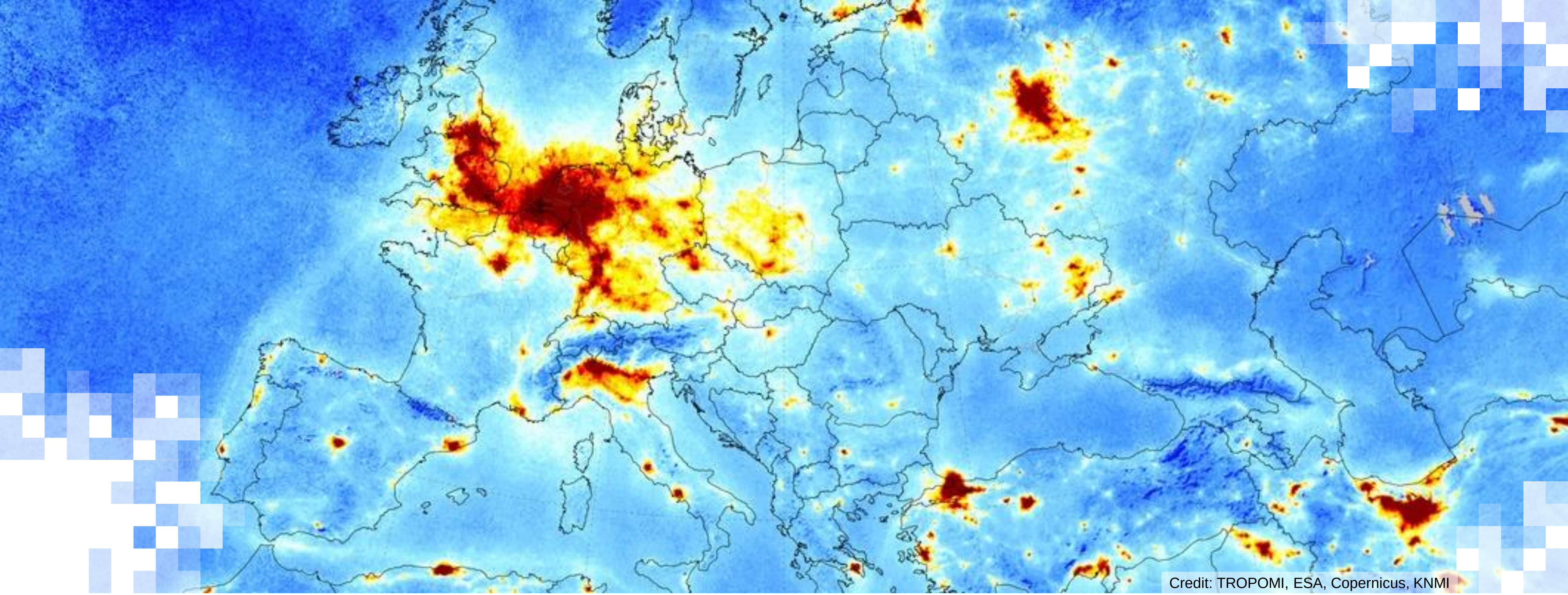


```

#loops through all files listed in the text file
for FILE_NAME in fileList:
    FILE_NAME=FILE_NAME.strip()
    user_input=input('\nWould you like to process\n' + FILE_NAME + '\n\n(Y/N)')
    if(user_input == 'N' or user_input == 'n'):
        print('Skipping...')
        continue
    else:
        file = Dataset(FILE_NAME, 'r')
# read the data
        if 'NO2' in FILE_NAME:
            print('This is an TROPOMI NO2 file.')
            #this is how you access the data tree in an hdf5 file
            SDS_NAME='nitrogendioxide_tropospheric_column'
        elif 'AER_AI' in FILE_NAME:
            print('This is an TROPOMI Aerosol Index file.')
            SDS_NAME='aerosol_index_354_388'
        ds=file
        grp='PRODUCT'
        lat= ds.groups[grp].variables['latitude'][0][:][:]
        lon= ds.groups[grp].variables['longitude'][0][:][:]
        data= ds.groups[grp].variables[SDS_NAME]

        #get necessary attributes
        fv=data._FillValue

```



Credit: TROPOMI, ESA, Copernicus, KNMI

Transition to OMI Data


Credit: TROPOMI, ESA, Copernicus, KNMI

Read an OMI NO_2 File (he5) and Print SDS List

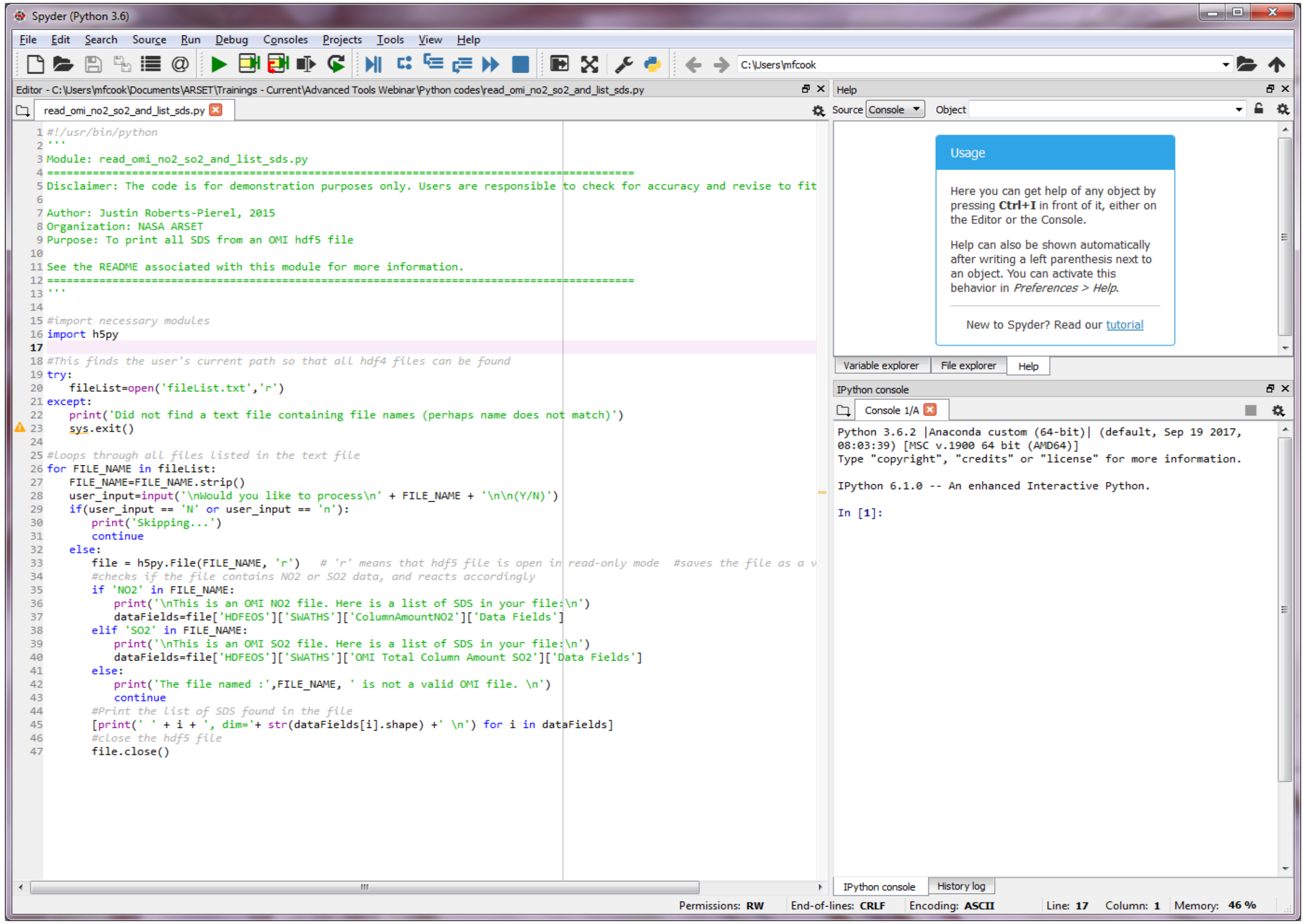
Print Scientific Data Sets (SDSs)

read_omi_no2_so2_and_list_sds.py
read_omi_no2_so2_and_list_sds_geo.py

Purpose: read OMI NO_2 or SO_2 level 2 data files in hdf format and print all the **Scientific Data Sets** (SDS).

In their current form, all of these codes work for *only level 2 products, not gridded products*.

The '_geo.py' code lists all of the geolocation fields



```
#!/usr/bin/python
'''
Module: read_omi_no2_so2_and_list_sds.py
==========================================================================================
Disclaimer: The code is for demonstration purposes only. Users are responsible to check for accuracy and revise to fit

Author: Justin Roberts-Pierel, 2015
Organization: NASA ARSET
Purpose: To print all SDS from an OMI hdf5 file

See the README associated with this module for more information.
==========================================================================================
'''

#import necessary modules
import h5py

#This finds the user's current path so that all hdf4 files can be found
try:
    fileList=open('fileList.txt','r')
except:
    print('Did not find a text file containing file names (perhaps name does not match)')
    sys.exit()

#loops through all files listed in the text file
for FILE_NAME in fileList:
    FILE_NAME=FILE_NAME.strip()
    user_input=input('\nWould you like to process\n' + FILE_NAME + '\n\n(Y/N)')
    if(user_input == 'N' or user_input == 'n'):
        print('Skipping...')
        continue
    else:
        file = h5py.File(FILE_NAME, 'r')   # 'r' means that hdf5 file is open in read-only mode  #saves the file as a v
        #checks if the file contains NO2 or SO2 data, and reacts accordingly
        if 'NO2' in FILE_NAME:
            print('\nThis is an OMI NO2 file. Here is a list of SDS in your file:\n')
            dataFields=file['HDFEOS']['SWATHS']['ColumnAmountNO2']['Data Fields']
        elif 'SO2' in FILE_NAME:
            print('\nThis is an OMI SO2 file. Here is a list of SDS in your file:\n')
            dataFields=file['HDFEOS']['SWATHS']['OMI Total Column Amount SO2']['Data Fields']
        else:
            print('The file named :',FILE_NAME, ' is not a valid OMI file. \n')
            continue
        #Print the list of SDS found in the file
        [print(' ' + i + ', dim='+ str(dataFields[i].shape) +' \n') for i in dataFields]
        #close the hdf5 file
        file.close()
```

Here you can get help of any object by pressing **Ctrl+I** in front of it, either on the Editor or the Console.

Help can also be shown automatically after writing a left parenthesis next to an object. You can activate this behavior in *Preferences > Help*.

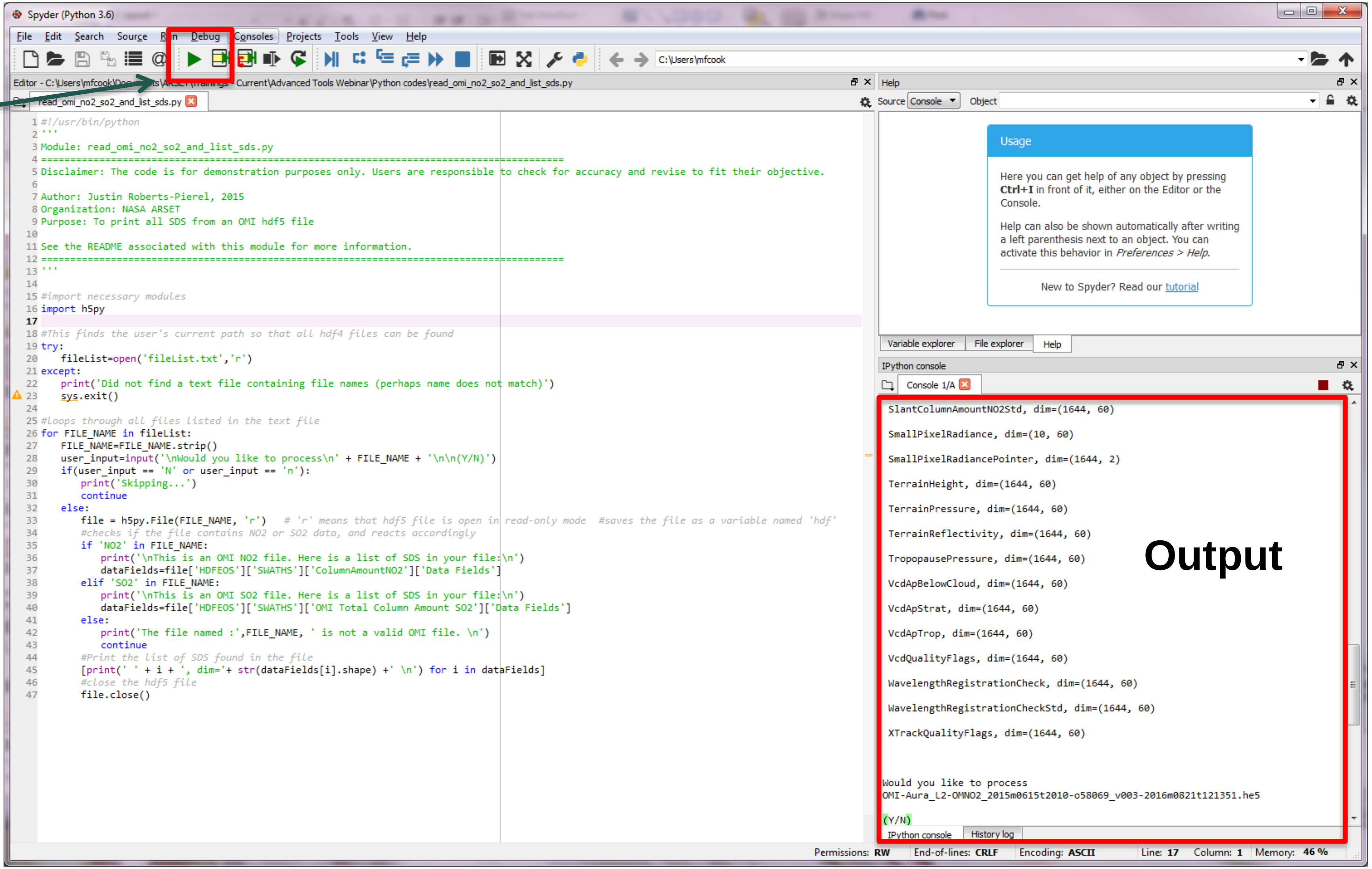
New to Spyder? Read our tutorial

```
Python 3.6.2 |Anaconda custom (64-bit)| (default, Sep 19 2017,
08:03:39) [MSC v.1900 64 bit (AMD64)]
Type "copyright", "credits" or "license" for more information.

IPython 6.1.0 -- An enhanced Interactive Python.

In [1]:
```

Permissions: RW   End-of-lines: CRLF   Encoding: ASCII   Line: 17   Column: 1   Memory: 46 %

Running and Output

- Click the green arrow to run the code
- The code will process all of the files in **fileList.txt** one-by-one
- Follow the instructions on the **ipython** terminal (i.e. enter ‘Y’ or ‘N’ when prompted and hit enter)



```
#!/usr/bin/python
'''
Module: read_omi_no2_so2_and_list_sds.py
==========================================================================================
Disclaimer: The code is for demonstration purposes only. Users are responsible to check for accuracy and revise to fit their objective.

Author: Justin Roberts-Pierel, 2015
Organization: NASA ARSET
Purpose: To print all SDS from an OMI hdf5 file

See the README associated with this module for more information.
==========================================================================================
'''

#import necessary modules
import h5py

#This finds the user's current path so that all hdf4 files can be found
try:
    fileList=open('fileList.txt','r')
except:
    print('Did not find a text file containing file names (perhaps name does not match)')
    sys.exit()

#loops through all files listed in the text file
for FILE_NAME in fileList:
    FILE_NAME=FILE_NAME.strip()
    user_input=input('\nWould you like to process\n' + FILE_NAME + '\n\n(Y/N)')
    if(user_input == 'N' or user_input == 'n'):
        print('Skipping...')
        continue
    else:
        file = h5py.File(FILE_NAME, 'r')   # 'r' means that hdf5 file is open in read-only mode  #saves the file as a variable named 'hdf'
        #checks if the file contains NO2 or SO2 data, and reacts accordingly
        if 'NO2' in FILE_NAME:
            print('\nThis is an OMI NO2 file. Here is a list of SDS in your file:\n')
            dataFields=file['HDFEOS']['SWATHS']['ColumnAmountNO2']['Data Fields']
        elif 'SO2' in FILE_NAME:
            print('\nThis is an OMI SO2 file. Here is a list of SDS in your file:\n')
            dataFields=file['HDFEOS']['SWATHS']['OMI Total Column Amount SO2']['Data Fields']
        else:
            print('The file named :',FILE_NAME, ' is not a valid OMI file. \n')
            continue
        #Print the list of SDS found in the file
        [print(' ' + i + ', dim='+ str(dataFields[i].shape) +' \n') for i in dataFields]
        #close the hdf5 file
        file.close()
```

Usage

Here you can get help of any object by pressing **Ctrl+I** in front of it, either on the Editor or the Console.

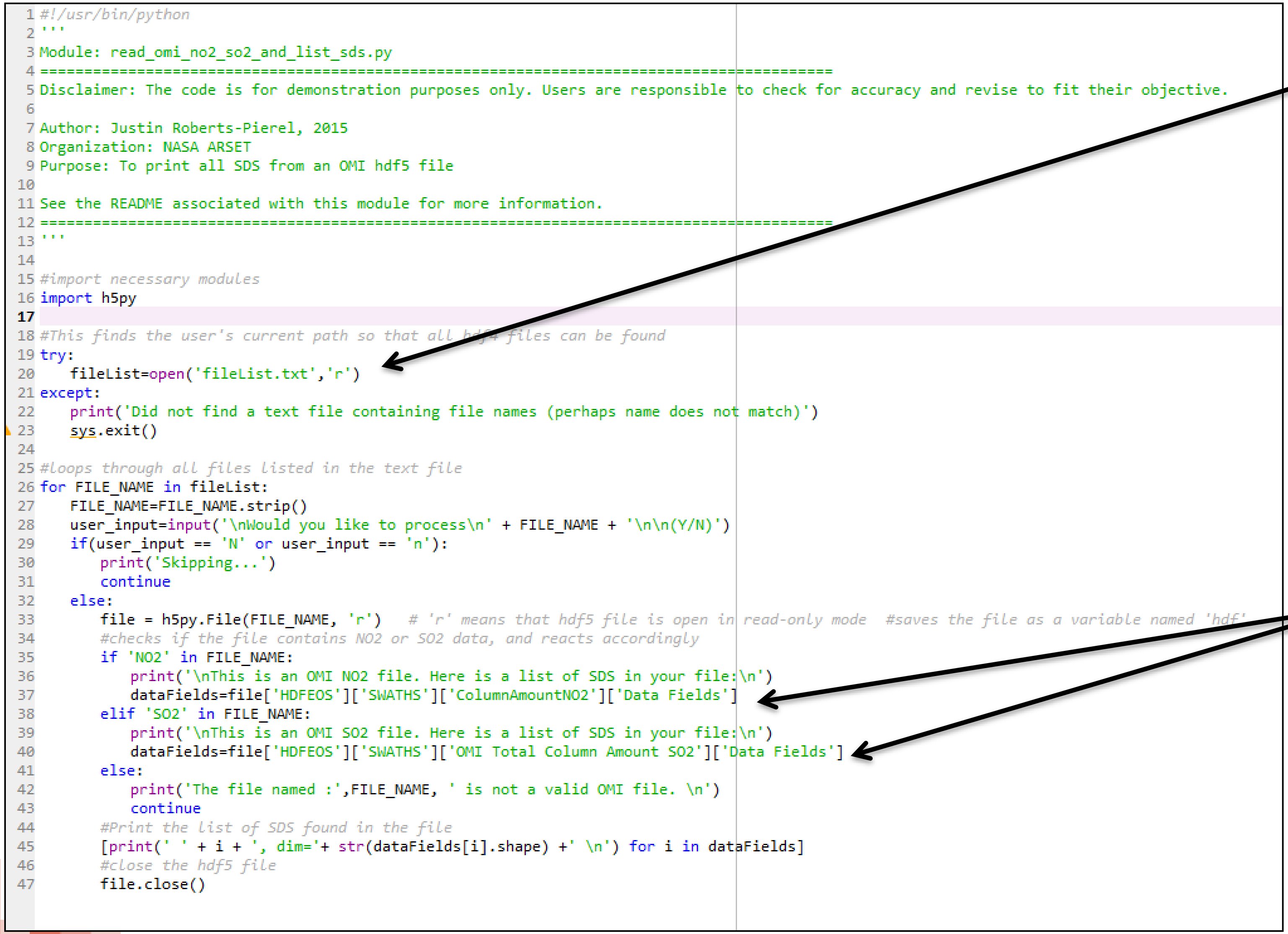
Help can also be shown automatically after writing a left parenthesis next to an object. You can activate this behavior in *Preferences > Help*.

New to Spyder? Read our tutorial

```
 SlantColumnAmountNO2Std, dim=(1644, 60)

 SmallPixelRadiance, dim=(10, 60)

 SmallPixelRadiancePointer, dim=(1644, 2)

 TerrainHeight, dim=(1644, 60)

 TerrainPressure, dim=(1644, 60)

 TerrainReflectivity, dim=(1644, 60)

 TropopausePressure, dim=(1644, 60)

 VcdApBelowCloud, dim=(1644, 60)

 VcdApStrat, dim=(1644, 60)

 VcdApTrop, dim=(1644, 60)

 VcdQualityFlags, dim=(1644, 60)

 WavelengthRegistrationCheck, dim=(1644, 60)

 WavelengthRegistrationCheckStd, dim=(1644, 60)

 XTrackQualityFlags, dim=(1644, 60)



Would you like to process
OMI-Aura_L2-OMNO2_2015m0615t2010-o58069_v003-2016m0821t121351.he5

(Y/N)
```

Permissions: RW   End-of-lines: CRLF   Encoding: ASCII   Line: 17   Column: 1   Memory: 46 %

Editing the Code



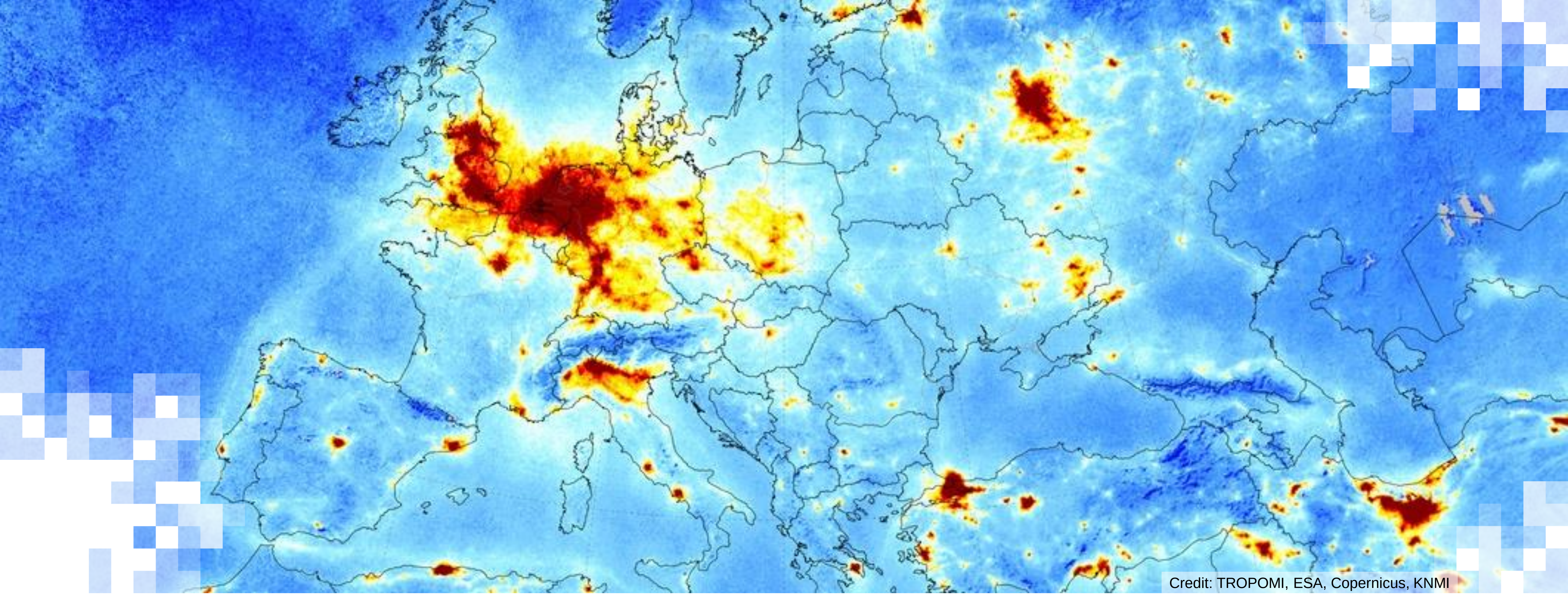
```python
#!/usr/bin/python
'''
Module: read_omi_no2_so2_and_list_sds.py
==========================================================================================
Disclaimer: The code is for demonstration purposes only. Users are responsible to check for accuracy and revise to fit their objective.

Author: Justin Roberts-Pierel, 2015
Organization: NASA ARSET
Purpose: To print all SDS from an OMI hdf5 file

See the README associated with this module for more information.
==========================================================================================
'''

#import necessary modules
import h5py

#This finds the user's current path so that all hdf4 files can be found
try:
    fileList=open('fileList.txt','r')
except:
    print('Did not find a text file containing file names (perhaps name does not match)')
    sys.exit()

#loops through all files listed in the text file
for FILE_NAME in fileList:
    FILE_NAME=FILE_NAME.strip()
    user_input=input('\nWould you like to process\n' + FILE_NAME + '\n\n(Y/N)')
    if(user_input == 'N' or user_input == 'n'):
        print('Skipping...')
        continue
    else:
        file = h5py.File(FILE_NAME, 'r')   # 'r' means that hdf5 file is open in read-only mode  #saves the file as a variable named 'hdf'
        #checks if the file contains NO2 or SO2 data, and reacts accordingly
        if 'NO2' in FILE_NAME:
            print('\nThis is an OMI NO2 file. Here is a list of SDS in your file:\n')
            dataFields=file['HDFEOS']['SWATHS']['ColumnAmountNO2']['Data Fields']
        elif 'SO2' in FILE_NAME:
            print('\nThis is an OMI SO2 file. Here is a list of SDS in your file:\n')
            dataFields=file['HDFEOS']['SWATHS']['OMI Total Column Amount SO2']['Data Fields']
        else:
            print('The file named :',FILE_NAME, ' is not a valid OMI file. \n')
            continue
        #Print the list of SDS found in the file
        [print(' ' + i + ', dim='+ str(dataFields[i].shape) +' \n') for i in dataFields]
        #close the hdf5 file
        file.close()
```

Change the name of fileList.txt to anything you'd like

By changing the location of dataFields to geolocation (found in other codes) this can also list the available geolocation variables

Applications

- OMI Level 2 NO_2 and SO_2 data are provided in hdf file
- Each HDF file contains several geophysical parameters
- Special codes/tools are required to open the hdf files
- This code helps users see the names and dimensions of the available SDSs inside an hdf file for further analysis

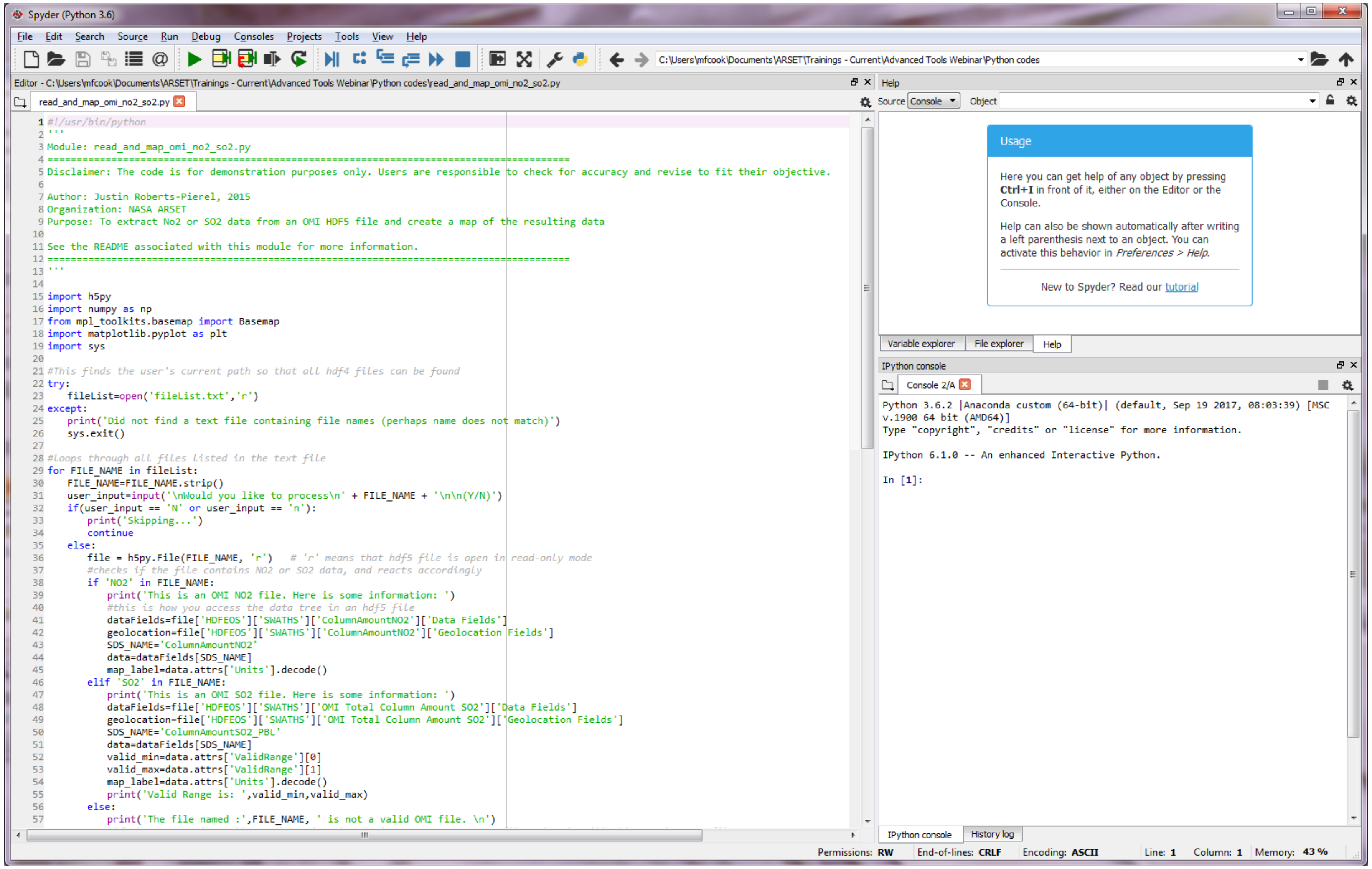


Credit: TROPOMI, ESA, Copernicus, KNMI

Map NO_2 or SO_2

Plot and save a map of OMI NO_2 or SO_2

read_and_map_omi_so2_no2.py



Spyder (Python 3.6)
File Edit Search Source Run Debug Consoles Projects Tools View Help
C:\Users\mfcook\Documents\ARSET\Trainings - Current\Advanced Tools Webinar\Python codes
Editor - C:\Users\mfcook\Documents\ARSET\Trainings - Current\Advanced Tools Webinar\Python codes\read_and_map_omi_no2_so2.py
read_and_map_omi_no2_so2.py
Help
Source Console
Object
Variable explorer
File explorer
Help
IPython console
Console 2/A
IPython console
History log

```python
#!/usr/bin/python
'''
Module: read_and_map_omi_no2_so2.py
==================================================================================
Disclaimer: The code is for demonstration purposes only. Users are responsible to check for accuracy and revise to fit their objective.

Author: Justin Roberts-Pierel, 2015
Organization: NASA ARSET
Purpose: To extract No2 or SO2 data from an OMI HDF5 file and create a map of the resulting data

See the README associated with this module for more information.
==================================================================================
'''

import h5py
import numpy as np
from mpl_toolkits.basemap import Basemap
import matplotlib.pyplot as plt
import sys

#This finds the user's current path so that all hdf4 files can be found
try:
    fileList=open('fileList.txt','r')
except:
    print('Did not find a text file containing file names (perhaps name does not match)')
    sys.exit()

#loops through all files listed in the text file
for FILE_NAME in fileList:
    FILE_NAME=FILE_NAME.strip()
    user_input=input('\nWould you like to process\n' + FILE_NAME + '\n\n(Y/N)')
    if(user_input == 'N' or user_input == 'n'):
        print('Skipping...')
        continue
    else:
        file = h5py.File(FILE_NAME, 'r')   # 'r' means that hdf5 file is open in read-only mode
        #checks if the file contains NO2 or SO2 data, and reacts accordingly
        if 'NO2' in FILE_NAME:
            print('This is an OMI NO2 file. Here is some information: ')
            #this is how you access the data tree in an hdf5 file
            dataFields=file['HDFEOS']['SWATHS']['ColumnAmountNO2']['Data Fields']
            geolocation=file['HDFEOS']['SWATHS']['ColumnAmountNO2']['Geolocation Fields']
            SDS_NAME='ColumnAmountNO2'
            data=dataFields[SDS_NAME]
            map_label=data.attrs['Units'].decode()
        elif 'SO2' in FILE_NAME:
            print('This is an OMI SO2 file. Here is some information: ')
            dataFields=file['HDFEOS']['SWATHS']['OMI Total Column Amount SO2']['Data Fields']
            geolocation=file['HDFEOS']['SWATHS']['OMI Total Column Amount SO2']['Geolocation Fields']
            SDS_NAME='ColumnAmountSO2_PBL'
            data=dataFields[SDS_NAME]
            valid_min=data.attrs['ValidRange'][0]
            valid_max=data.attrs['ValidRange'][1]
            map_label=data.attrs['Units'].decode()
            print('Valid Range is: ',valid_min,valid_max)
        else:
            print('The file named :',FILE_NAME, ' is not a valid OMI file. \n')
```

Usage

Here you can get help of any object by pressing **Ctrl+I** in front of it, either on the Editor or the Console.

Help can also be shown automatically after writing a left parenthesis next to an object. You can activate this behavior in *Preferences > Help*.

New to Spyder? Read our tutorial

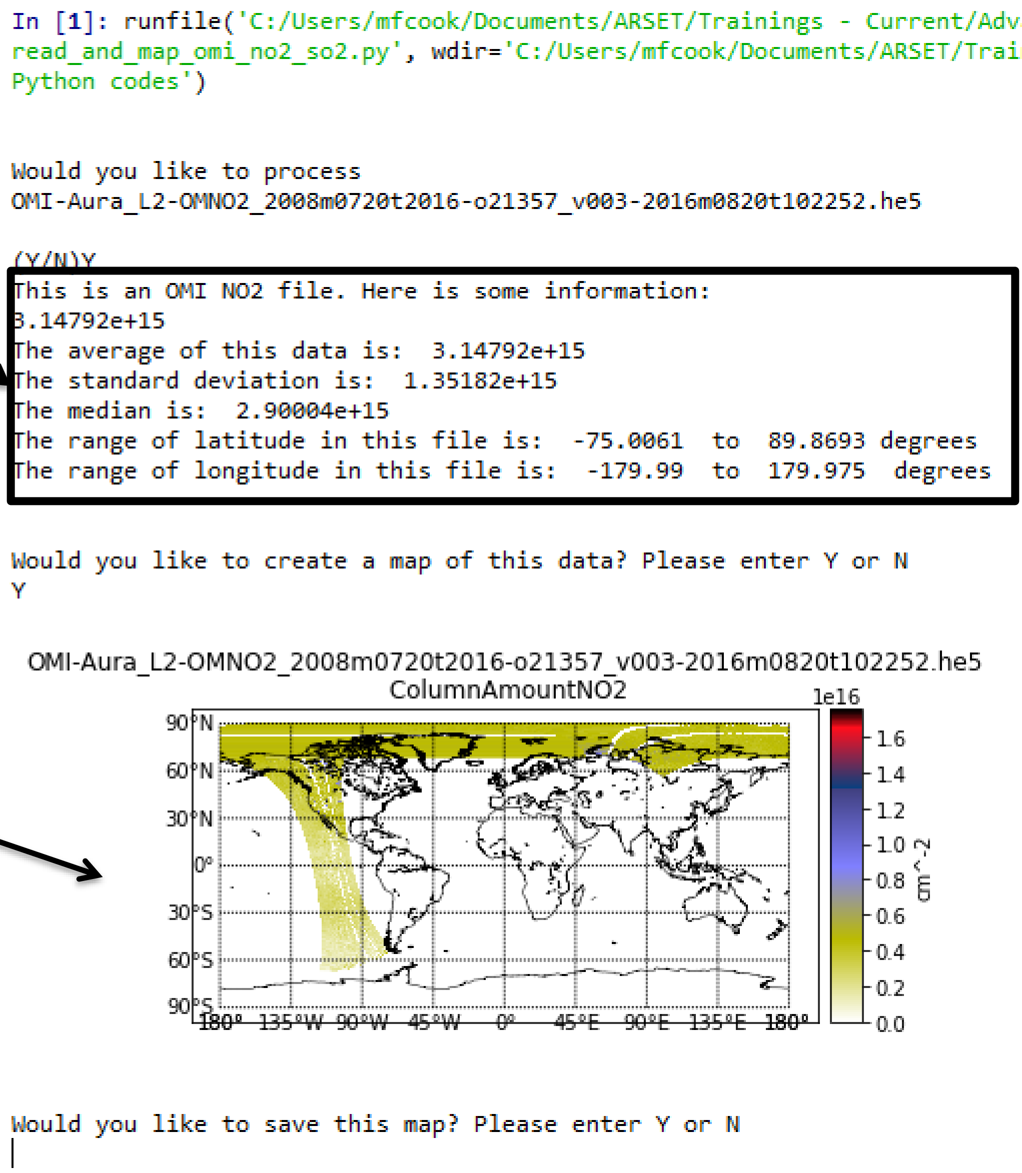
```
Python 3.6.2 |Anaconda custom (64-bit)| (default, Sep 19 2017, 08:03:39) [MSC
v.1900 64 bit (AMD64)]
Type "copyright", "credits" or "license" for more information.

IPython 6.1.0 -- An enhanced Interactive Python.

In [1]:
```

Permissions: RW End-of-lines: CRLF Encoding: ASCII Line: 1 Column: 1 Memory: 43 %

Running and Output

NO_2/SO_2 Statistics

Output map

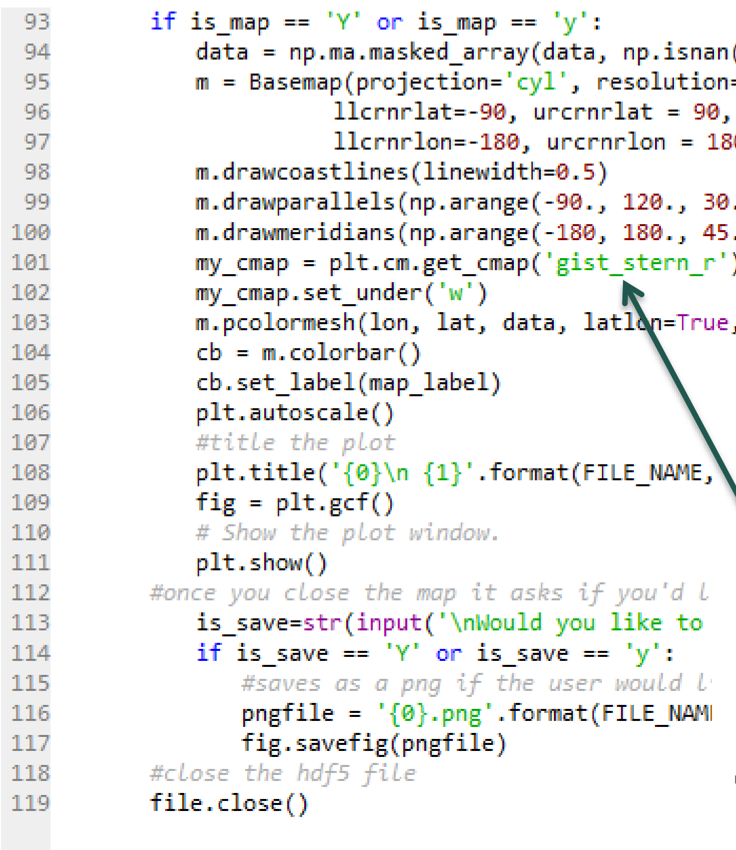


```
In [1]: runfile('C:/Users/mfcook/Documents/ARSET/Trainings - Current/Advanced Tools Webinar/Python codes/
read_and_map_omi_no2_so2.py', wdir='C:/Users/mfcook/Documents/ARSET/Trainings - Current/Advanced Tools Webinar/
Python codes')


Would you like to process
OMI-Aura_L2-OMNO2_2008m0720t2016-o21357_v003-2016m0820t102252.he5

(Y/N)Y
This is an OMI NO2 file. Here is some information:
3.14792e+15
The average of this data is:  3.14792e+15
The standard deviation is:  1.35182e+15
The median is:  2.90004e+15
The range of latitude in this file is:  -75.0061  to  89.8693 degrees
The range of longitude in this file is:  -179.99  to  179.975  degrees

Would you like to create a map of this data? Please enter Y or N
Y
```

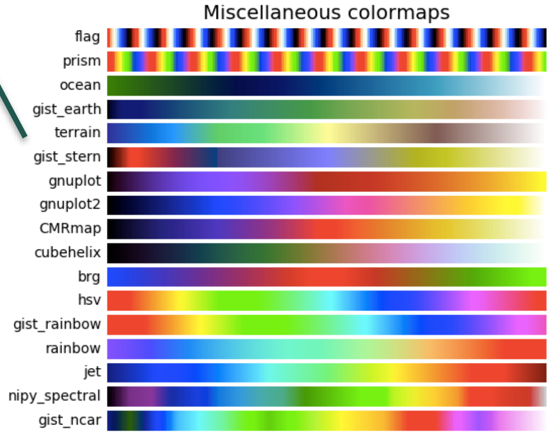
```
Would you like to save this map? Please enter Y or N
```

Editing the Code

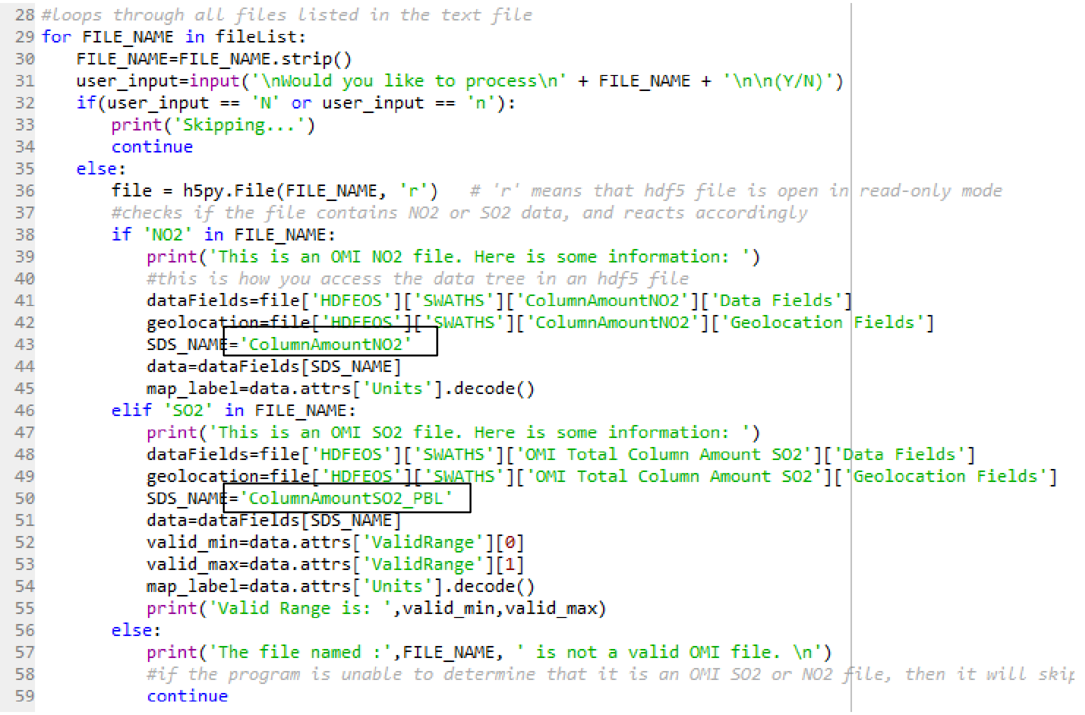
Change the color scale



```
if is_map == 'Y' or is_map == 'y':
    data = np.ma.masked_array(data, np.isnan(data))
    m = Basemap(projection='cyl', resolution='l',
              llcrnrlat=-90, urcrnrlat = 90,
              llcrnrlon=-180, urcrnrlon = 180)
    m.drawcoastlines(linewidth=0.5)
    m.drawparallels(np.arange(-90., 120., 30.), labels=[1, 0, 0, 0])
    m.drawmeridians(np.arange(-180, 180., 45.), labels=[0, 0, 0, 1])
    my_cmap = plt.cm.get_cmap('gist_stern_r')
    my_cmap.set_under('w')
    m.pcolormesh(lon, lat, data, latlon=True, vmin=0, vmax=np.nanmax(dat
    cb = m.colorbar()
    cb.set_label(map_label)
    plt.autoscale()
    #title the plot
    plt.title('{0}\n {1}'.format(FILE_NAME,
    fig = plt.gcf()
    # Show the plot window.
    plt.show()
#once you close the map it asks if you'd l
    is_save=str(input('\nWould you like to
    if is_save == 'Y' or is_save == 'y':
        #saves as a png if the user would l
        pngfile = '{0}.png'.format(FILE_NAM
        fig.savefig(pngfile)
#close the hdf5 file
file.close()
```



Change the SDS to plot

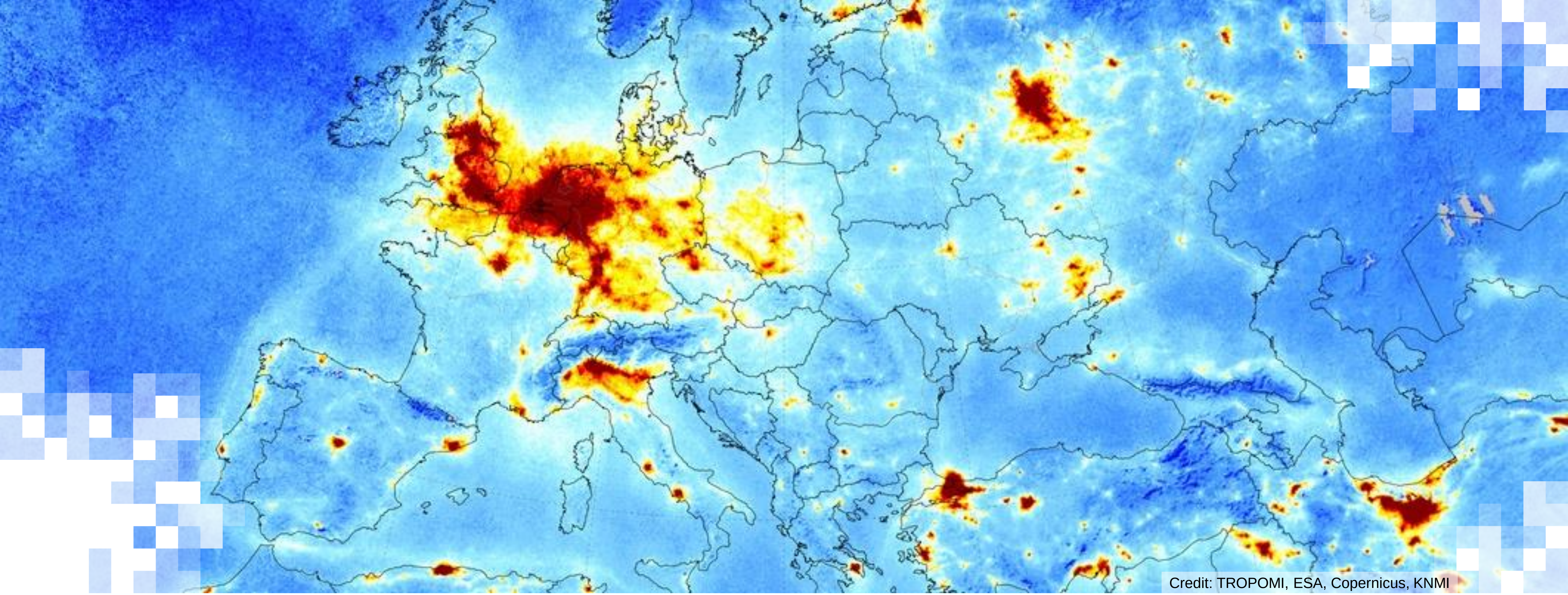


```
#loops through all files listed in the text file
for FILE_NAME in fileList:
    FILE_NAME=FILE_NAME.strip()
    user_input=input('\nWould you like to process\n' + FILE_NAME + '\n\n(Y/N)')
    if(user_input == 'N' or user_input == 'n'):
        print('Skipping...')
        continue
    else:
        file = h5py.File(FILE_NAME, 'r')   # 'r' means that hdf5 file is open in read-only mode
        #checks if the file contains NO2 or SO2 data, and reacts accordingly
        if 'NO2' in FILE_NAME:
            print('This is an OMI NO2 file. Here is some information: ')
            #this is how you access the data tree in an hdf5 file
            dataFields=file['HDFEOS']['SWATHS']['ColumnAmountNO2']['Data Fields']
            geolocation=file['HDFEOS']['SWATHS']['ColumnAmountNO2']['Geolocation Fields']
            SDS_NAME='ColumnAmountNO2'
            data=dataFields[SDS_NAME]
            map_label=data.attrs['Units'].decode()
        elif 'SO2' in FILE_NAME:
            print('This is an OMI SO2 file. Here is some information: ')
            dataFields=file['HDFEOS']['SWATHS']['OMI Total Column Amount SO2']['Data Fields']
            geolocation=file['HDFEOS']['SWATHS']['OMI Total Column Amount SO2']['Geolocation Fields']
            SDS_NAME='ColumnAmountSO2_PBL'
            data=dataFields[SDS_NAME]
            valid_min=data.attrs['ValidRange'][0]
            valid_max=data.attrs['ValidRange'][1]
            map_label=data.attrs['Units'].decode()
            print('Valid Range is: ',valid_min,valid_max)
        else:
            print('The file named :',FILE_NAME, ' is not a valid OMI file. \n')
            #if the program is unable to determine that it is an OMI SO2 or NO2 file, then it will skip
            continue
```

https://matplotlib.org/examples/color/colormaps_reference.html

Applications

- This is a sample code to read and map the OMI Level 2 NO_2 and SO_2 data
- The code can be modified to address various mapping needs
- User can create daily maps of trace gas columns over certain regions and start analyzing changes over time
- These maps can also help identify regions of high pollution



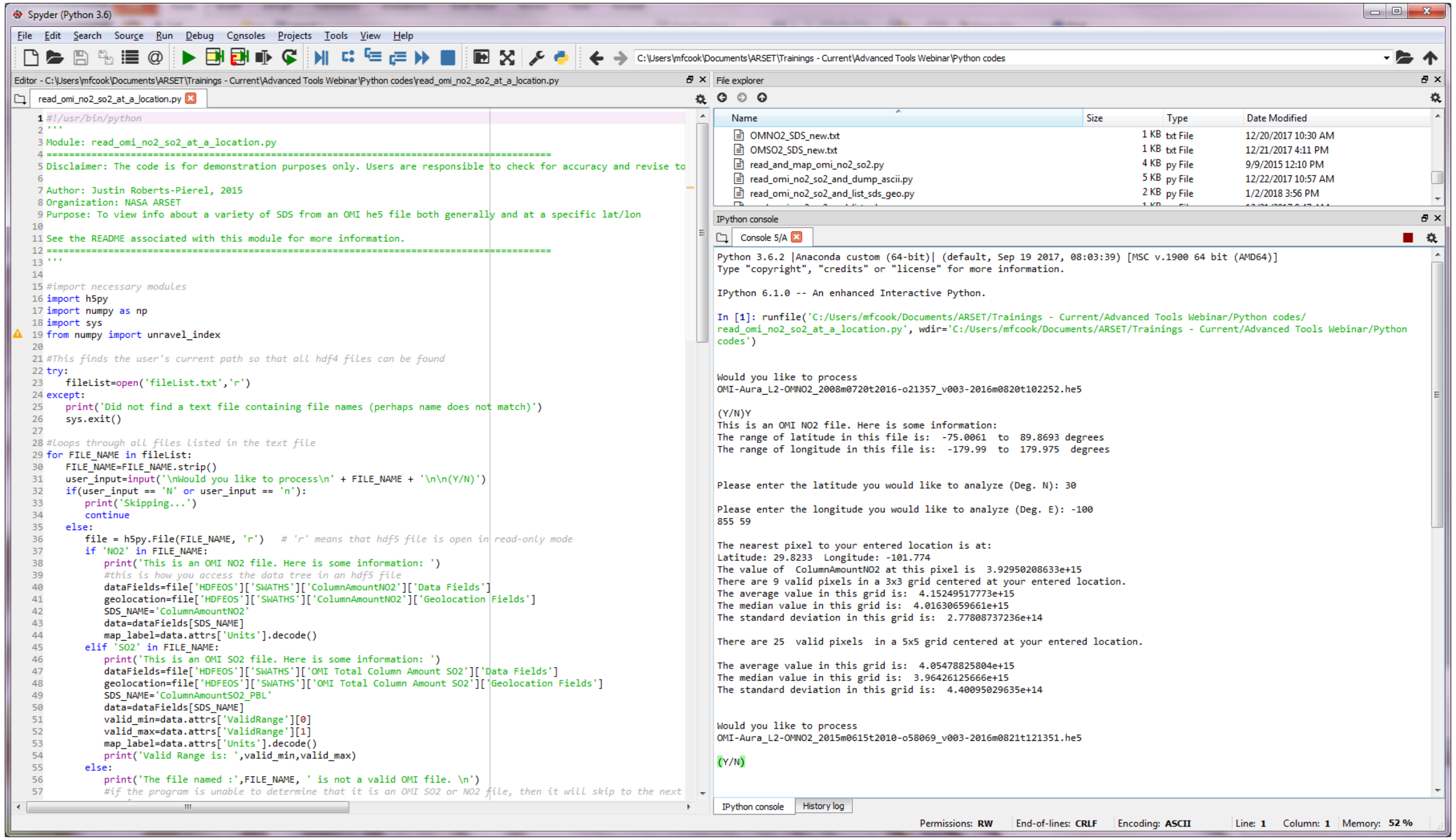
Credit: TROPOMI, ESA, Copernicus, KNMI

Extract NO_2/SO_2 at a given location

Extract AOD Values from MODIS Aerosol Level 2 Data

read_mod_aerosol_and_list_sds.py

- **Purpose**: read an OMI NO_2/SO_2 level 2 data file in HDF format and extract values at a given ground location


Spyder (Python 3.6)
File
Edit
Search
Source
Run
Debug
Consoles
Projects
Tools
View
Help
C:\Users\mfcook\Documents\ARSET\Trainings - Current\Advanced Tools Webinar\Python codes
Editor - C:\Users\mfcook\Documents\ARSET\Trainings - Current\Advanced Tools Webinar\Python codes\read_omi_no2_so2_at_a_location.py
read_omi_no2_so2_at_a_location.py
File explorer
IPython console
Console 5/A
IPython console
History log

```python
#!/usr/bin/python
'''
Module: read_omi_no2_so2_at_a_location.py
==========================================================================================
Disclaimer: The code is for demonstration purposes only. Users are responsible to check for accuracy and revise to

Author: Justin Roberts-Pierel, 2015
Organization: NASA ARSET
Purpose: To view info about a variety of SDS from an OMI he5 file both generally and at a specific lat/lon

See the README associated with this module for more information.
==========================================================================================
'''

#import necessary modules
import h5py
import numpy as np
import sys
from numpy import unravel_index

#This finds the user's current path so that all hdf4 files can be found
try:
   fileList=open('fileList.txt','r')
except:
   print('Did not find a text file containing file names (perhaps name does not match)')
   sys.exit()

#loops through all files listed in the text file
for FILE_NAME in fileList:
   FILE_NAME=FILE_NAME.strip()
   user_input=input('\nWould you like to process\n' + FILE_NAME + '\n\n(Y/N)')
   if(user_input == 'N' or user_input == 'n'):
      print('Skipping...')
      continue
   else:
      file = h5py.File(FILE_NAME, 'r')   # 'r' means that hdf5 file is open in read-only mode
      if 'NO2' in FILE_NAME:
         print('This is an OMI NO2 file. Here is some information: ')
         #this is how you access the data tree in an hdf5 file
         dataFields=file['HDFEOS']['SWATHS']['ColumnAmountNO2']['Data Fields']
         geolocation=file['HDFEOS']['SWATHS']['ColumnAmountNO2']['Geolocation Fields']
         SDS_NAME='ColumnAmountNO2'
         data=dataFields[SDS_NAME]
         map_label=data.attrs['Units'].decode()
      elif 'SO2' in FILE_NAME:
         print('This is an OMI SO2 file. Here is some information: ')
         dataFields=file['HDFEOS']['SWATHS']['OMI Total Column Amount SO2']['Data Fields']
         geolocation=file['HDFEOS']['SWATHS']['OMI Total Column Amount SO2']['Geolocation Fields']
         SDS_NAME='ColumnAmountSO2_PBL'
         data=dataFields[SDS_NAME]
         valid_min=data.attrs['ValidRange'][0]
         valid_max=data.attrs['ValidRange'][1]
         map_label=data.attrs['Units'].decode()
         print('Valid Range is: ',valid_min,valid_max)
      else:
         print('The file named :',FILE_NAME, ' is not a valid OMI file. \n')
         #if the program is unable to determine that it is an OMI SO2 or NO2 file, then it will skip to the next
```

Name	Size	Type	Date Modified
OMNO2_SDS_new.txt	1 KB	txt File	12/20/2017 10:30 AM
OMSO2_SDS_new.txt	1 KB	txt File	12/21/2017 4:11 PM
read_and_map_omi_no2_so2.py	4 KB	py File	9/9/2015 12:10 PM
read_omi_no2_so2_and_dump_ascii.py	5 KB	py File	12/22/2017 10:57 AM
read_omi_no2_so2_and_list_sds_geo.py	2 KB	py File	1/2/2018 3:56 PM

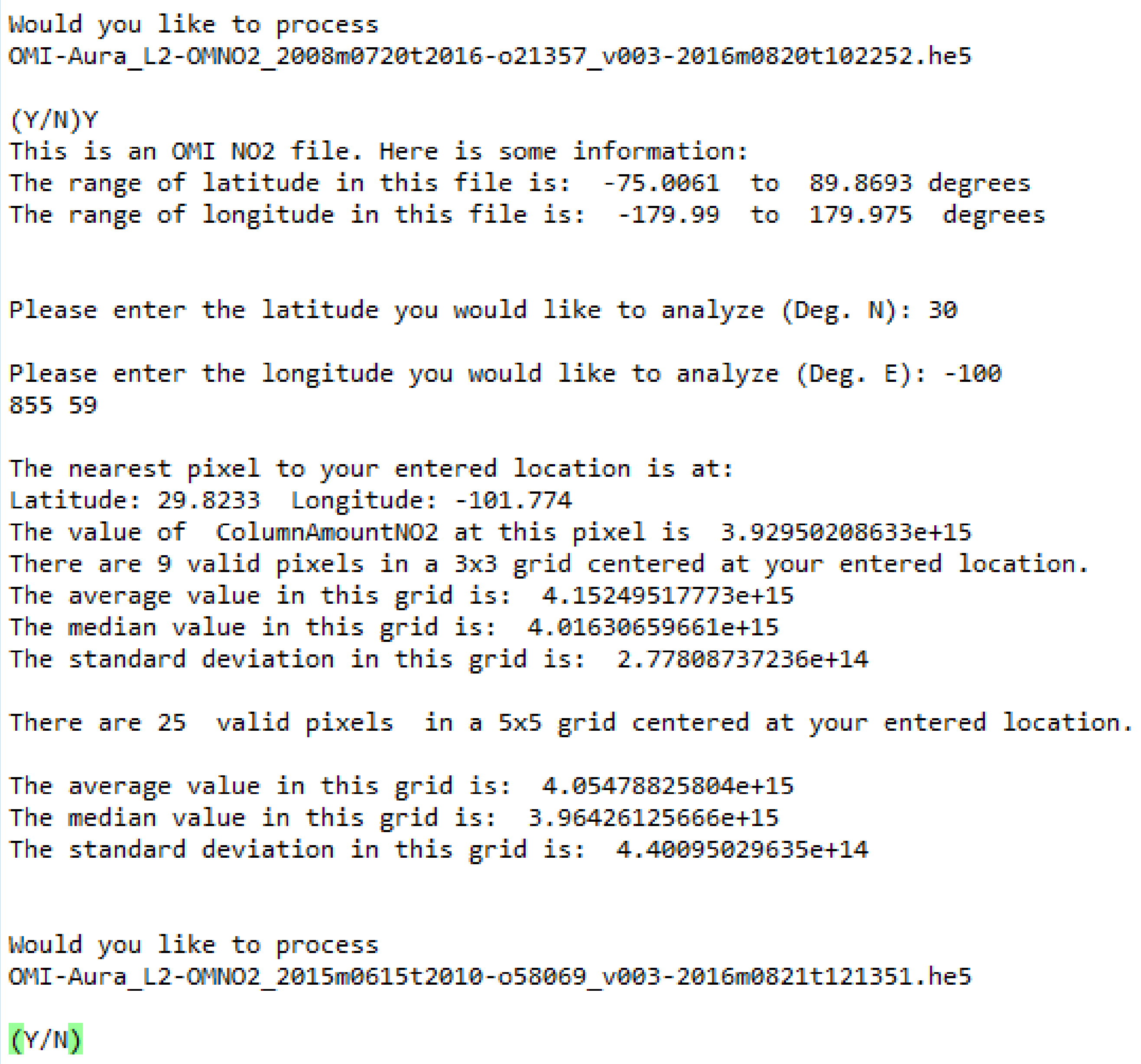
```
Python 3.6.2 |Anaconda custom (64-bit)| (default, Sep 19 2017, 08:03:39) [MSC v.1900 64 bit (AMD64)]
Type "copyright", "credits" or "license" for more information.

IPython 6.1.0 -- An enhanced Interactive Python.

In [1]: runfile('C:/Users/mfcook/Documents/ARSET/Trainings - Current/Advanced Tools Webinar/Python codes/
read_omi_no2_so2_at_a_location.py', wdir='C:/Users/mfcook/Documents/ARSET/Trainings - Current/Advanced Tools Webinar/Python
codes')


Would you like to process
OMI-Aura_L2-OMNO2_2008m0720t2016-o21357_v003-2016m0820t102252.he5

(Y/N)Y
This is an OMI NO2 file. Here is some information:
The range of latitude in this file is:  -75.0061  to  89.8693 degrees
The range of longitude in this file is:  -179.99  to  179.975  degrees


Please enter the latitude you would like to analyze (Deg. N): 30

Please enter the longitude you would like to analyze (Deg. E): -100
855 59

The nearest pixel to your entered location is at:
Latitude: 29.8233  Longitude: -101.774
The value of  ColumnAmountNO2 at this pixel is  3.92950208633e+15
There are 9 valid pixels in a 3x3 grid centered at your entered location.
The average value in this grid is:  4.15249517773e+15
The median value in this grid is:  4.01630659661e+15
The standard deviation in this grid is:  2.77808737236e+14

There are 25  valid pixels  in a 5x5 grid centered at your entered location.

The average value in this grid is:  4.05478825804e+15
The median value in this grid is:  3.96426125666e+15
The standard deviation in this grid is:  4.40095029635e+14


Would you like to process
OMI-Aura_L2-OMNO2_2015m0615t2010-o58069_v003-2016m0821t121351.he5

(Y/N)
```

Permissions: RW End-of-lines: CRLF Encoding: ASCII Line: 1 Column: 1 Memory: 52 %

Running and Output

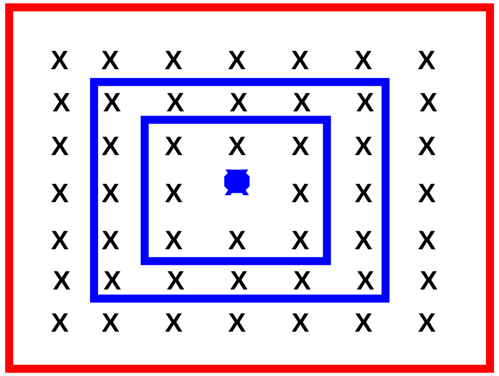
Type “Y” to process file, “N” to skip

Lat & Lon of station

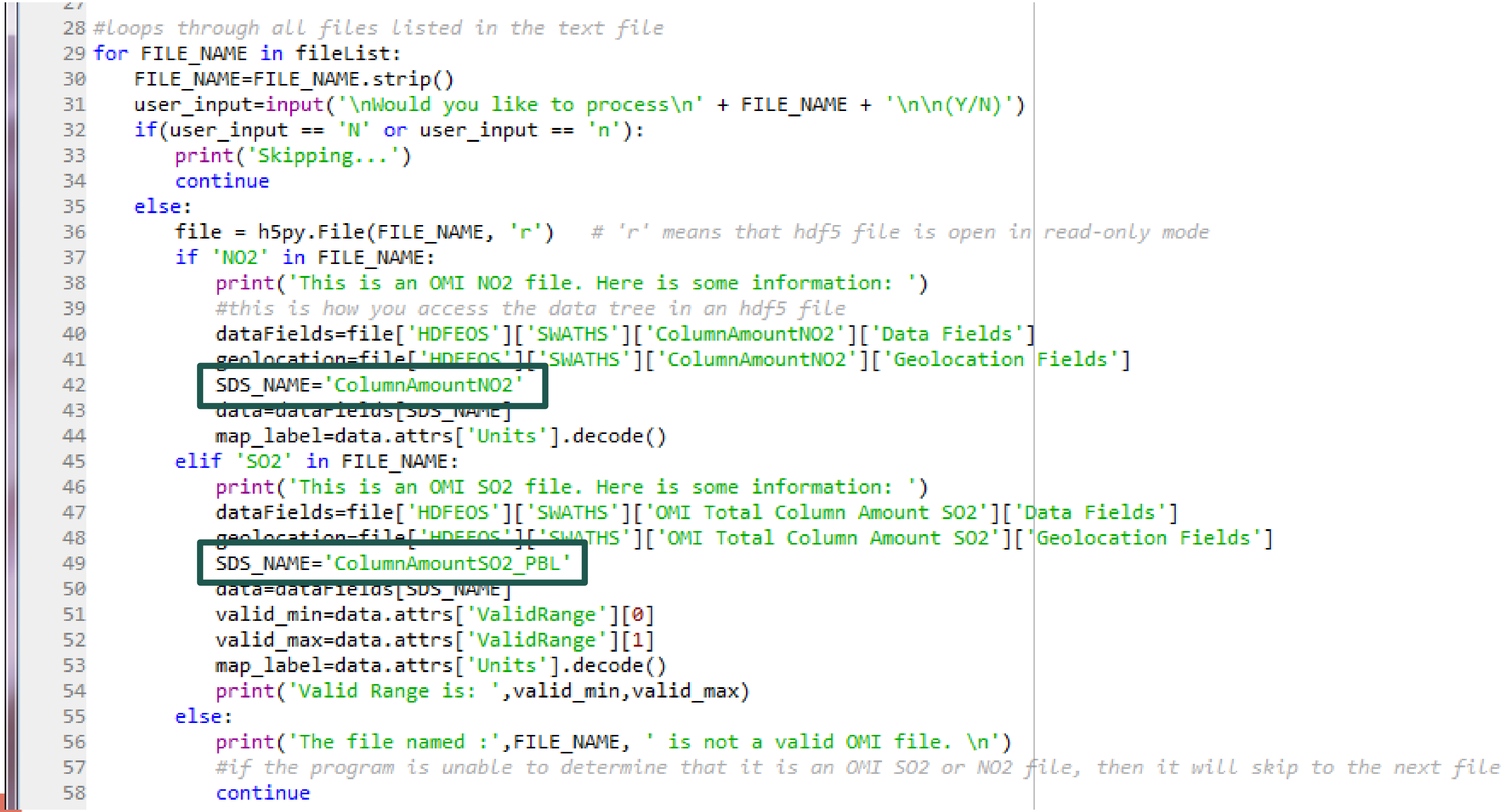


```
Would you like to process
OMI-Aura_L2-OMNO2_2008m0720t2016-o21357_v003-2016m0820t102252.he5

(Y/N)Y
This is an OMI NO2 file. Here is some information:
The range of latitude in this file is:  -75.0061  to  89.8693 degrees
The range of longitude in this file is:  -179.99  to  179.975  degrees


Please enter the latitude you would like to analyze (Deg. N): 30

Please enter the longitude you would like to analyze (Deg. E): -100
855 59

The nearest pixel to your entered location is at:
Latitude: 29.8233  Longitude: -101.774
The value of  ColumnAmountNO2 at this pixel is  3.92950208633e+15
There are 9 valid pixels in a 3x3 grid centered at your entered location.
The average value in this grid is:  4.15249517773e+15
The median value in this grid is:  4.01630659661e+15
The standard deviation in this grid is:  2.77808737236e+14

There are 25  valid pixels  in a 5x5 grid centered at your entered location.

The average value in this grid is:  4.05478825804e+15
The median value in this grid is:  3.96426125666e+15
The standard deviation in this grid is:  4.40095029635e+14


Would you like to process
OMI-Aura_L2-OMNO2_2015m0615t2010-o58069_v003-2016m0821t121351.he5

(Y/N)
```

Outputs



Editing the Code – Change the SDS



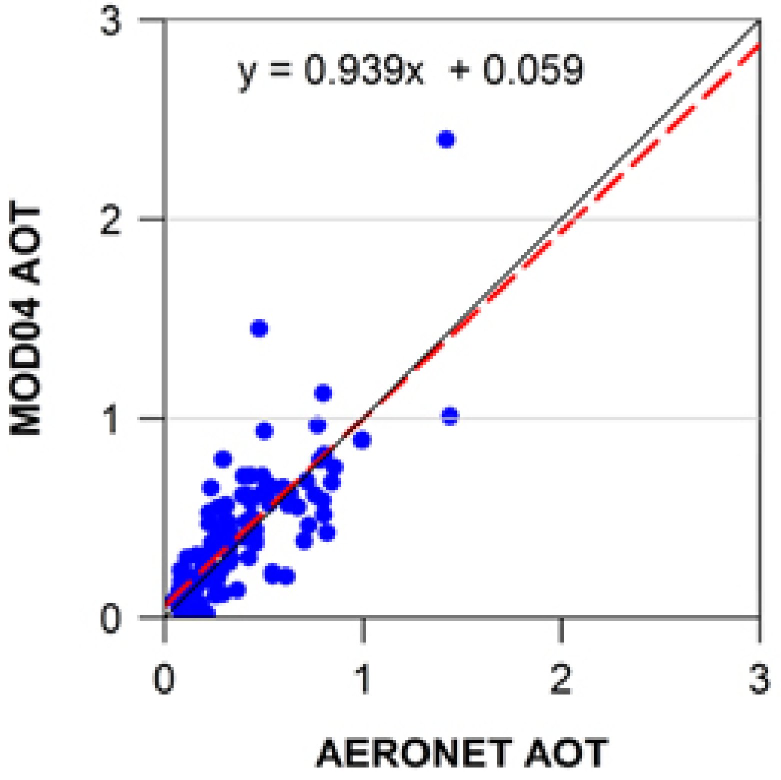
```python
#loops through all files listed in the text file
for FILE_NAME in fileList:
    FILE_NAME=FILE_NAME.strip()
    user_input=input('\nWould you like to process\n' + FILE_NAME + '\n\n(Y/N)')
    if(user_input == 'N' or user_input == 'n'):
        print('Skipping...')
        continue
    else:
        file = h5py.File(FILE_NAME, 'r')   # 'r' means that hdf5 file is open in read-only mode
        if 'NO2' in FILE_NAME:
            print('This is an OMI NO2 file. Here is some information: ')
            #this is how you access the data tree in an hdf5 file
            dataFields=file['HDFEOS']['SWATHS']['ColumnAmountNO2']['Data Fields']
            geolocation=file['HDFEOS']['SWATHS']['ColumnAmountNO2']['Geolocation Fields']
            SDS_NAME='ColumnAmountNO2'
            data=dataFields[SDS_NAME]
            map_label=data.attrs['Units'].decode()
        elif 'SO2' in FILE_NAME:
            print('This is an OMI SO2 file. Here is some information: ')
            dataFields=file['HDFEOS']['SWATHS']['OMI Total Column Amount SO2']['Data Fields']
            geolocation=file['HDFEOS']['SWATHS']['OMI Total Column Amount SO2']['Geolocation Fields']
            SDS_NAME='ColumnAmountSO2_PBL'
            data=dataFields[SDS_NAME]
            valid_min=data.attrs['ValidRange'][0]
            valid_max=data.attrs['ValidRange'][1]
            map_label=data.attrs['Units'].decode()
            print('Valid Range is: ',valid_min,valid_max)
        else:
            print('The file named :',FILE_NAME, ' is not a valid OMI file. \n')
            #if the program is unable to determine that it is an OMI SO2 or NO2 file, then it will skip to the next file
            continue
```

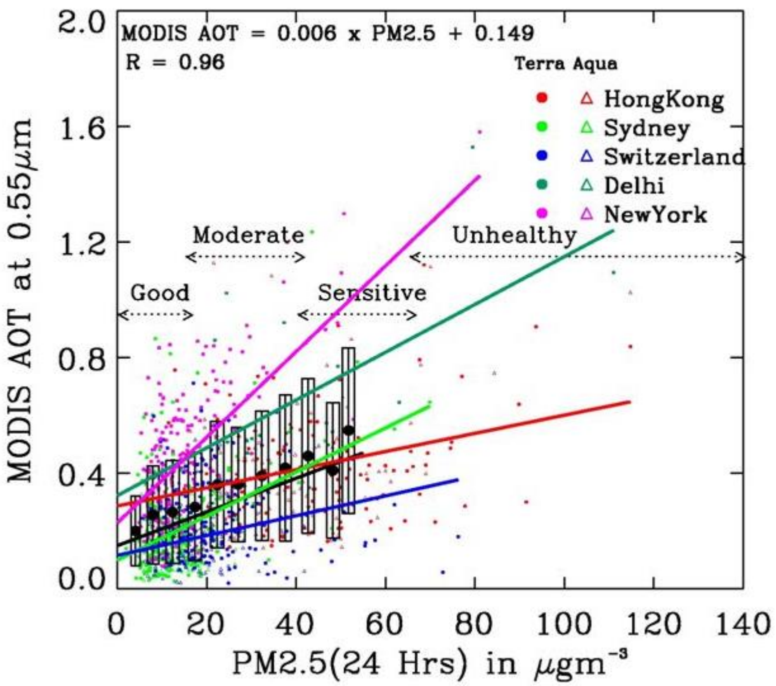
Applications

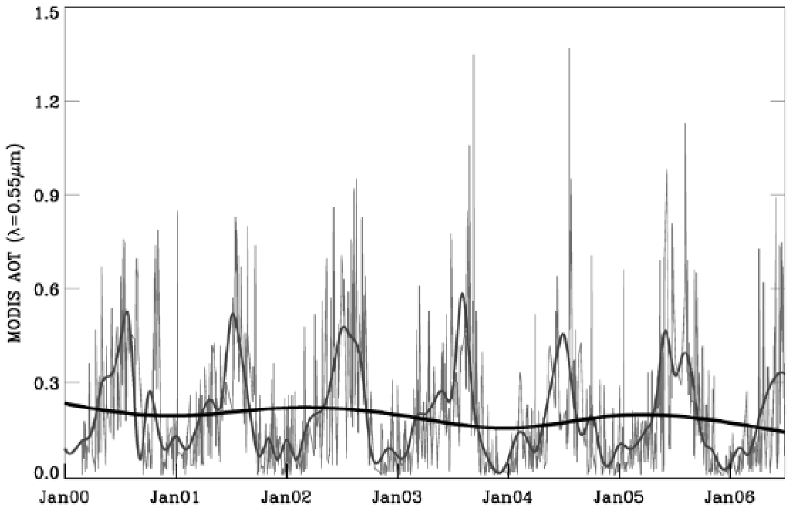
Satellite AOD Validation

AOD-PM$_{2.5}$ Relationship

Time Series Analysis

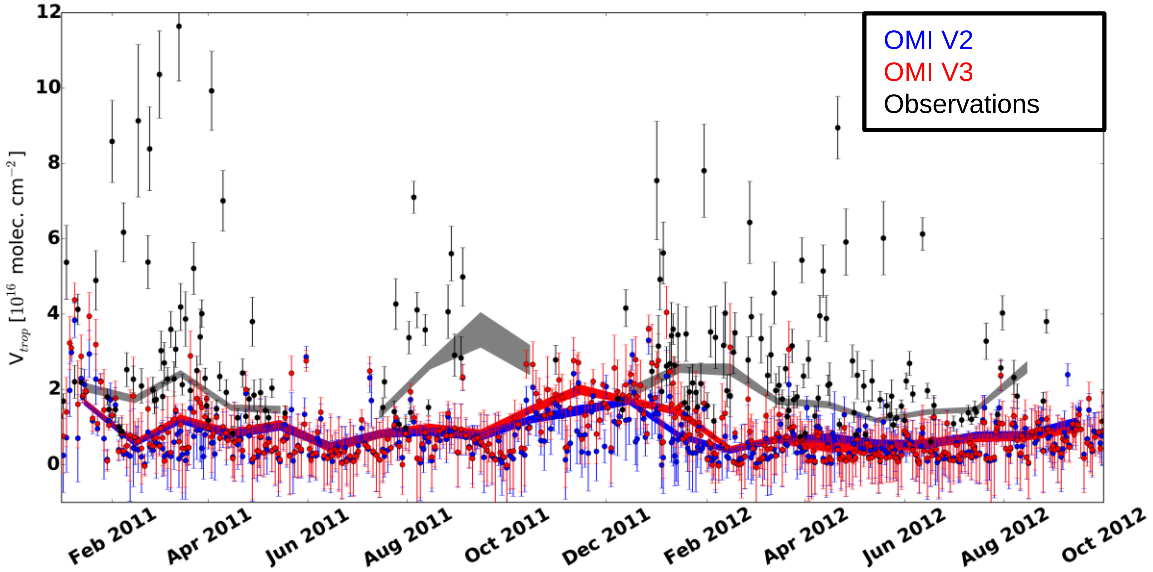






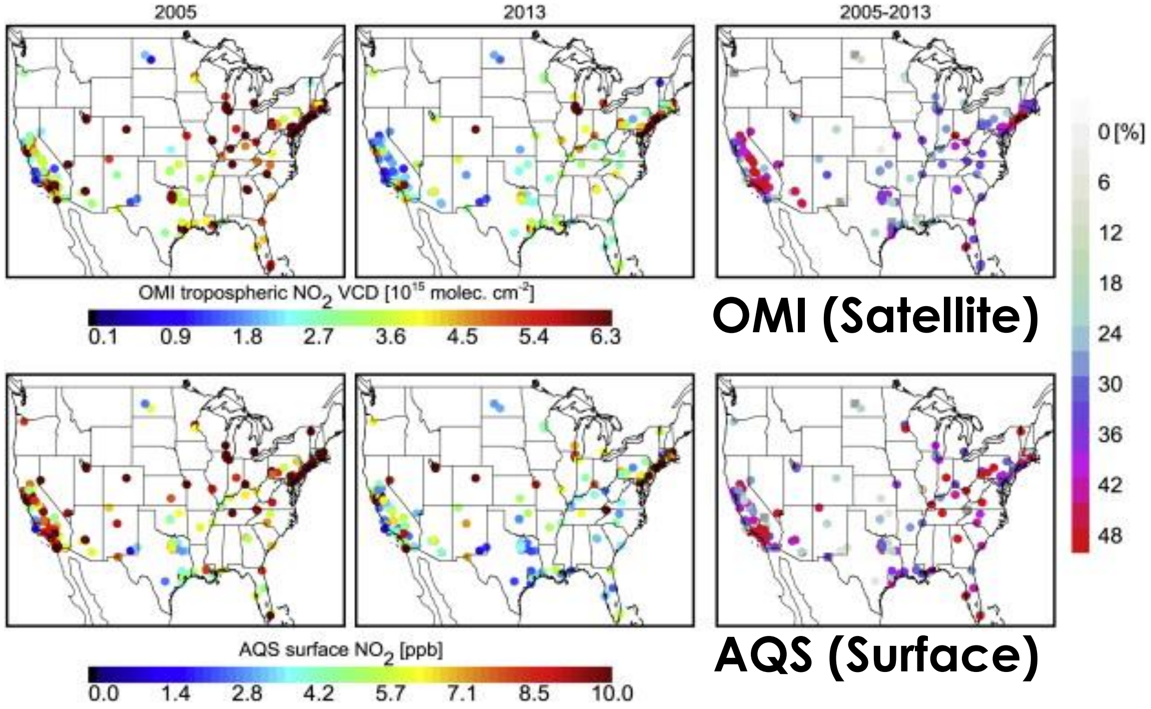
Applications

Satellite Validation

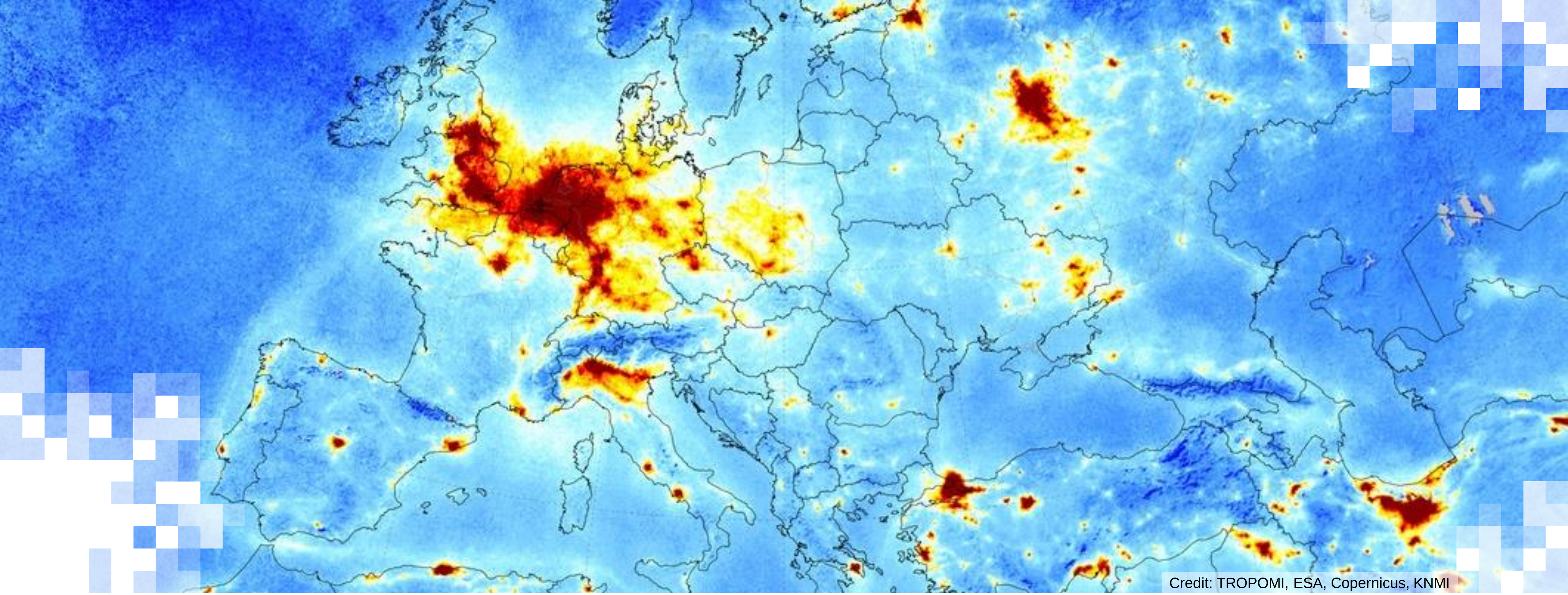


Source: Krotkov et al. (2017)

Column vs. Surface
Relationship and Trends



Source: Lamsal, L.N. et al. (2016)



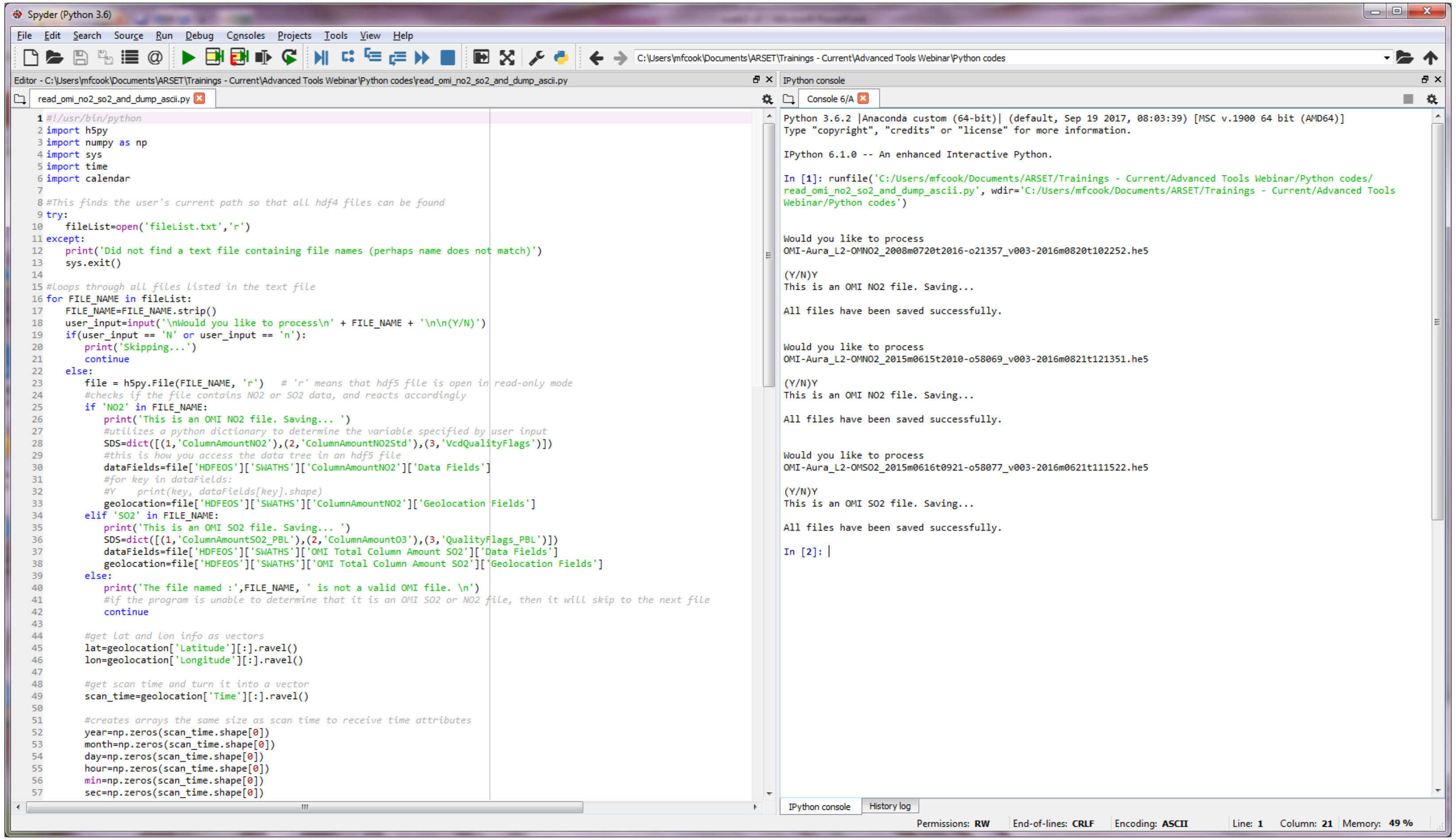
Credit: TROPOMI, ESA, Copernicus, KNMI

Output HDF variables to CSV

Output OMI NO_2/SO_2 HDF variables to a CSV file

read_omi_no2_so2_and_dump_ascii.py

- **Purpose**: read an OMI level 2 NO_2 or SO_2 data file in HDF format and write certain SDSs into a csv (text) file

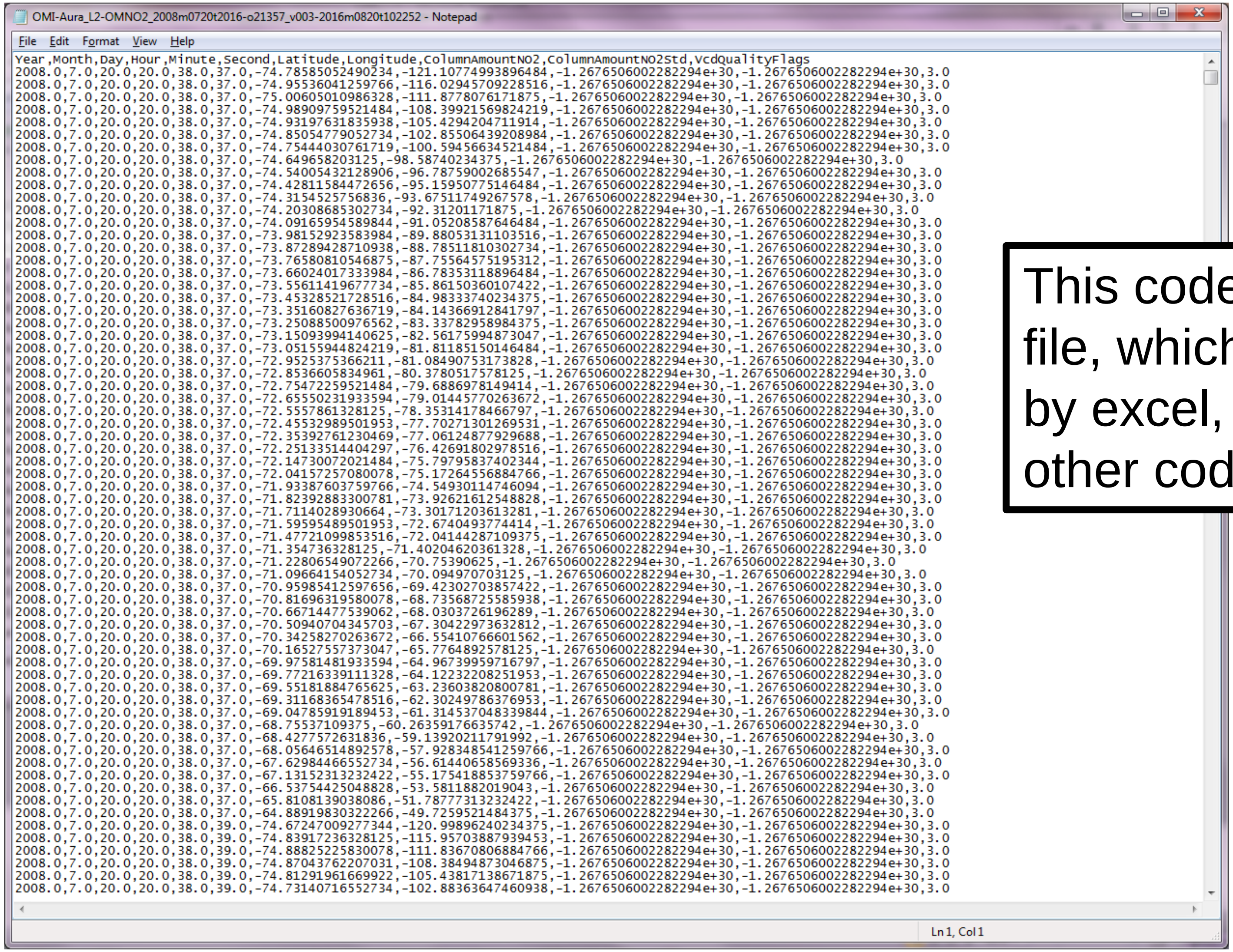

Spyder (Python 3.6)
File Edit Search Source Run Debug Consoles Projects Tools View Help
C:\Users\mfcook\Documents\ARSET\Trainings - Current\Advanced Tools Webinar\Python codes
Editor - C:\Users\mfcook\Documents\ARSET\Trainings - Current\Advanced Tools Webinar\Python codes\read_omi_no2_so2_and_dump_ascii.py
read_omi_no2_so2_and_dump_ascii.py
IPython console
Console 6/A
IPython console
History log

```python
#!/usr/bin/python
import h5py
import numpy as np
import sys
import time
import calendar

#This finds the user's current path so that all hdf4 files can be found
try:
    fileList=open('fileList.txt','r')
except:
    print('Did not find a text file containing file names (perhaps name does not match)')
    sys.exit()

#loops through all files listed in the text file
for FILE_NAME in fileList:
    FILE_NAME=FILE_NAME.strip()
    user_input=input('\nWould you like to process\n' + FILE_NAME + '\n\n(Y/N)')
    if(user_input == 'N' or user_input == 'n'):
        print('Skipping...')
        continue
    else:
        file = h5py.File(FILE_NAME, 'r')   # 'r' means that hdf5 file is open in read-only mode
        #checks if the file contains NO2 or SO2 data, and reacts accordingly
        if 'NO2' in FILE_NAME:
            print('This is an OMI NO2 file. Saving... ')
            #utilizes a python dictionary to determine the variable specified by user input
            SDS=dict([(1,'ColumnAmountNO2'),(2,'ColumnAmountNO2Std'),(3,'VcdQualityFlags')])
            #this is how you access the data tree in an hdf5 file
            dataFields=file['HDFEOS']['SWATHS']['ColumnAmountNO2']['Data Fields']
            #for key in dataFields:
            #Y    print(key, dataFields[key].shape)
            geolocation=file['HDFEOS']['SWATHS']['ColumnAmountNO2']['Geolocation Fields']
        elif 'SO2' in FILE_NAME:
            print('This is an OMI SO2 file. Saving... ')
            SDS=dict([(1,'ColumnAmountSO2_PBL'),(2,'ColumnAmountO3'),(3,'QualityFlags_PBL')])
            dataFields=file['HDFEOS']['SWATHS']['OMI Total Column Amount SO2']['Data Fields']
            geolocation=file['HDFEOS']['SWATHS']['OMI Total Column Amount SO2']['Geolocation Fields']
        else:
            print('The file named :',FILE_NAME, ' is not a valid OMI file. \n')
            #if the program is unable to determine that it is an OMI SO2 or NO2 file, then it will skip to the next file
            continue

        #get lat and lon info as vectors
        lat=geolocation['Latitude'][:].ravel()
        lon=geolocation['Longitude'][:].ravel()

        #get scan time and turn it into a vector
        scan_time=geolocation['Time'][:].ravel()

        #creates arrays the same size as scan time to receive time attributes
        year=np.zeros(scan_time.shape[0])
        month=np.zeros(scan_time.shape[0])
        day=np.zeros(scan_time.shape[0])
        hour=np.zeros(scan_time.shape[0])
        min=np.zeros(scan_time.shape[0])
        sec=np.zeros(scan_time.shape[0])
```

```
Python 3.6.2 |Anaconda custom (64-bit)| (default, Sep 19 2017, 08:03:39) [MSC v.1900 64 bit (AMD64)]
Type "copyright", "credits" or "license" for more information.

IPython 6.1.0 -- An enhanced Interactive Python.

In [1]: runfile('C:/Users/mfcook/Documents/ARSET/Trainings - Current/Advanced Tools Webinar/Python codes/
read_omi_no2_so2_and_dump_ascii.py', wdir='C:/Users/mfcook/Documents/ARSET/Trainings - Current/Advanced Tools
Webinar/Python codes')


Would you like to process
OMI-Aura_L2-OMNO2_2008m0720t2016-o21357_v003-2016m0820t102252.he5

(Y/N)Y
This is an OMI NO2 file. Saving...

All files have been saved successfully.


Would you like to process
OMI-Aura_L2-OMNO2_2015m0615t2010-o58069_v003-2016m0821t121351.he5

(Y/N)Y
This is an OMI NO2 file. Saving...

All files have been saved successfully.


Would you like to process
OMI-Aura_L2-OMSO2_2015m0616t0921-o58077_v003-2016m0621t111522.he5

(Y/N)Y
This is an OMI SO2 file. Saving...

All files have been saved successfully.

In [2]:
```

Permissions: RW   End-of-lines: CRLF   Encoding: ASCII   Line: 1   Column: 21   Memory: 49 %

Output


OMI-Aura_L2-OMNO2_2008m0720t2016-o21357_v003-2016m0820t102252 - Notepad
File Edit Format View Help
Ln 1, Col 1

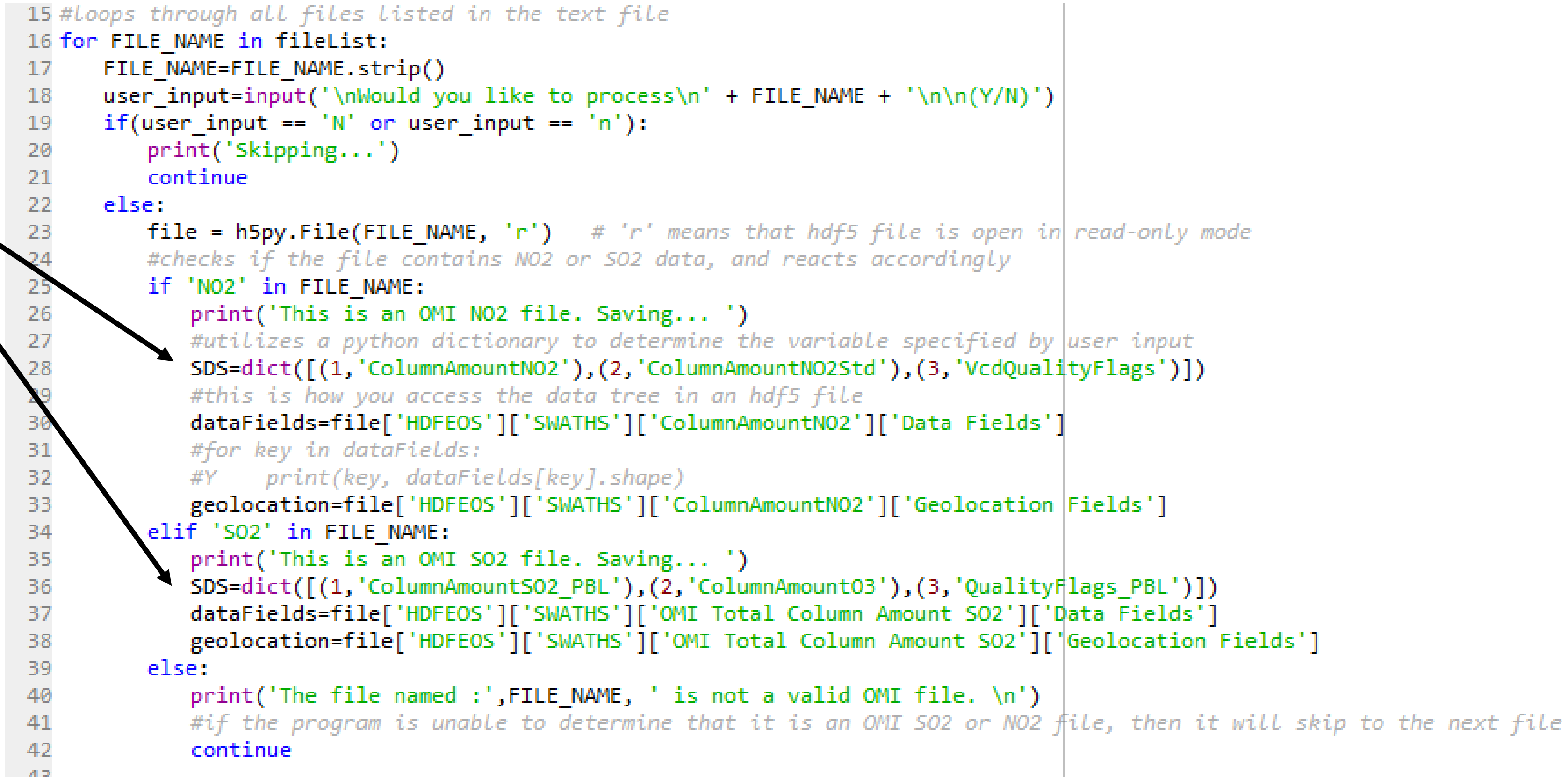
```
Year,Month,Day,Hour,Minute,Second,Latitude,Longitude,ColumnAmountNO2,ColumnAmountNO2Std,VcdQualityFlags
2008.0,7.0,20.0,20.0,38.0,37.0,-74.78585052490234,-121.10774993896484,-1.2676506002282294e+30,-1.2676506002282294e+30,3.0
2008.0,7.0,20.0,20.0,38.0,37.0,-74.95536041259766,-116.02945709228516,-1.2676506002282294e+30,-1.2676506002282294e+30,3.0
2008.0,7.0,20.0,20.0,38.0,37.0,-75.00605010986328,-111.8778076171875,-1.2676506002282294e+30,-1.2676506002282294e+30,3.0
2008.0,7.0,20.0,20.0,38.0,37.0,-74.98909759521484,-108.39921569824219,-1.2676506002282294e+30,-1.2676506002282294e+30,3.0
2008.0,7.0,20.0,20.0,38.0,37.0,-74.93197631835938,-105.4294204711914,-1.2676506002282294e+30,-1.2676506002282294e+30,3.0
2008.0,7.0,20.0,20.0,38.0,37.0,-74.85054779052734,-102.85506439208984,-1.2676506002282294e+30,-1.2676506002282294e+30,3.0
2008.0,7.0,20.0,20.0,38.0,37.0,-74.75444030761719,-100.59456634521484,-1.2676506002282294e+30,-1.2676506002282294e+30,3.0
2008.0,7.0,20.0,20.0,38.0,37.0,-74.649658203125,-98.58740234375,-1.2676506002282294e+30,-1.2676506002282294e+30,3.0
2008.0,7.0,20.0,20.0,38.0,37.0,-74.54005432128906,-96.78759002685547,-1.2676506002282294e+30,-1.2676506002282294e+30,3.0
2008.0,7.0,20.0,20.0,38.0,37.0,-74.42811584472656,-95.15950775146484,-1.2676506002282294e+30,-1.2676506002282294e+30,3.0
2008.0,7.0,20.0,20.0,38.0,37.0,-74.3154525756836,-93.67511749267578,-1.2676506002282294e+30,-1.2676506002282294e+30,3.0
2008.0,7.0,20.0,20.0,38.0,37.0,-74.20308685302734,-92.31201171875,-1.2676506002282294e+30,-1.2676506002282294e+30,3.0
2008.0,7.0,20.0,20.0,38.0,37.0,-74.09165954589844,-91.05208587646484,-1.2676506002282294e+30,-1.2676506002282294e+30,3.0
2008.0,7.0,20.0,20.0,38.0,37.0,-73.98152923583984,-89.88053131103516,-1.2676506002282294e+30,-1.2676506002282294e+30,3.0
2008.0,7.0,20.0,20.0,38.0,37.0,-73.87289428710938,-88.78511810302734,-1.2676506002282294e+30,-1.2676506002282294e+30,3.0
2008.0,7.0,20.0,20.0,38.0,37.0,-73.76580810546875,-87.75564575195312,-1.2676506002282294e+30,-1.2676506002282294e+30,3.0
2008.0,7.0,20.0,20.0,38.0,37.0,-73.66024017333984,-86.78353118896484,-1.2676506002282294e+30,-1.2676506002282294e+30,3.0
2008.0,7.0,20.0,20.0,38.0,37.0,-73.55611419677734,-85.86150360107422,-1.2676506002282294e+30,-1.2676506002282294e+30,3.0
2008.0,7.0,20.0,20.0,38.0,37.0,-73.45328521728516,-84.98333740234375,-1.2676506002282294e+30,-1.2676506002282294e+30,3.0
2008.0,7.0,20.0,20.0,38.0,37.0,-73.35160827636719,-84.14366912841797,-1.2676506002282294e+30,-1.2676506002282294e+30,3.0
2008.0,7.0,20.0,20.0,38.0,37.0,-73.25088500976562,-83.33782958984375,-1.2676506002282294e+30,-1.2676506002282294e+30,3.0
2008.0,7.0,20.0,20.0,38.0,37.0,-73.15093994140625,-82.56175994873047,-1.2676506002282294e+30,-1.2676506002282294e+30,3.0
2008.0,7.0,20.0,20.0,38.0,37.0,-73.05155944824219,-81.81185150146484,-1.2676506002282294e+30,-1.2676506002282294e+30,3.0
2008.0,7.0,20.0,20.0,38.0,37.0,-72.9525375366211,-81.08490753173828,-1.2676506002282294e+30,-1.2676506002282294e+30,3.0
2008.0,7.0,20.0,20.0,38.0,37.0,-72.8536605834961,-80.3780517578125,-1.2676506002282294e+30,-1.2676506002282294e+30,3.0
2008.0,7.0,20.0,20.0,38.0,37.0,-72.75472259521484,-79.6886978149414,-1.2676506002282294e+30,-1.2676506002282294e+30,3.0
2008.0,7.0,20.0,20.0,38.0,37.0,-72.65550231933594,-79.01445770263672,-1.2676506002282294e+30,-1.2676506002282294e+30,3.0
2008.0,7.0,20.0,20.0,38.0,37.0,-72.5557861328125,-78.35314178466797,-1.2676506002282294e+30,-1.2676506002282294e+30,3.0
2008.0,7.0,20.0,20.0,38.0,37.0,-72.45532989501953,-77.70271301269531,-1.2676506002282294e+30,-1.2676506002282294e+30,3.0
2008.0,7.0,20.0,20.0,38.0,37.0,-72.35392761230469,-77.06124877929688,-1.2676506002282294e+30,-1.2676506002282294e+30,3.0
2008.0,7.0,20.0,20.0,38.0,37.0,-72.25133514404297,-76.42691802978516,-1.2676506002282294e+30,-1.2676506002282294e+30,3.0
2008.0,7.0,20.0,20.0,38.0,37.0,-72.14730072021484,-75.79795837402344,-1.2676506002282294e+30,-1.2676506002282294e+30,3.0
2008.0,7.0,20.0,20.0,38.0,37.0,-72.04157257080078,-75.17264556884766,-1.2676506002282294e+30,-1.2676506002282294e+30,3.0
2008.0,7.0,20.0,20.0,38.0,37.0,-71.93387603759766,-74.54930114746094,-1.2676506002282294e+30,-1.2676506002282294e+30,3.0
2008.0,7.0,20.0,20.0,38.0,37.0,-71.82392883300781,-73.92621612548828,-1.2676506002282294e+30,-1.2676506002282294e+30,3.0
2008.0,7.0,20.0,20.0,38.0,37.0,-71.7114028930664,-73.30171203613281,-1.2676506002282294e+30,-1.2676506002282294e+30,3.0
2008.0,7.0,20.0,20.0,38.0,37.0,-71.59595489501953,-72.6740493774414,-1.2676506002282294e+30,-1.2676506002282294e+30,3.0
2008.0,7.0,20.0,20.0,38.0,37.0,-71.47721099853516,-72.04144287109375,-1.2676506002282294e+30,-1.2676506002282294e+30,3.0
2008.0,7.0,20.0,20.0,38.0,37.0,-71.354736328125,-71.40204620361328,-1.2676506002282294e+30,-1.2676506002282294e+30,3.0
2008.0,7.0,20.0,20.0,38.0,37.0,-71.22806549072266,-70.75390625,-1.2676506002282294e+30,-1.2676506002282294e+30,3.0
2008.0,7.0,20.0,20.0,38.0,37.0,-71.09664154052734,-70.094970703125,-1.2676506002282294e+30,-1.2676506002282294e+30,3.0
2008.0,7.0,20.0,20.0,38.0,37.0,-70.95985412597656,-69.42302703857422,-1.2676506002282294e+30,-1.2676506002282294e+30,3.0
2008.0,7.0,20.0,20.0,38.0,37.0,-70.81696319580078,-68.73568725585938,-1.2676506002282294e+30,-1.2676506002282294e+30,3.0
2008.0,7.0,20.0,20.0,38.0,37.0,-70.66714477539062,-68.0303726196289,-1.2676506002282294e+30,-1.2676506002282294e+30,3.0
2008.0,7.0,20.0,20.0,38.0,37.0,-70.50940704345703,-67.30422973632812,-1.2676506002282294e+30,-1.2676506002282294e+30,3.0
2008.0,7.0,20.0,20.0,38.0,37.0,-70.34258270263672,-66.55410766601562,-1.2676506002282294e+30,-1.2676506002282294e+30,3.0
2008.0,7.0,20.0,20.0,38.0,37.0,-70.16527557373047,-65.7764892578125,-1.2676506002282294e+30,-1.2676506002282294e+30,3.0
2008.0,7.0,20.0,20.0,38.0,37.0,-69.97581481933594,-64.96739959716797,-1.2676506002282294e+30,-1.2676506002282294e+30,3.0
2008.0,7.0,20.0,20.0,38.0,37.0,-69.77216339111328,-64.12232208251953,-1.2676506002282294e+30,-1.2676506002282294e+30,3.0
2008.0,7.0,20.0,20.0,38.0,37.0,-69.55181884765625,-63.23603820800781,-1.2676506002282294e+30,-1.2676506002282294e+30,3.0
2008.0,7.0,20.0,20.0,38.0,37.0,-69.31168365478516,-62.30249786376953,-1.2676506002282294e+30,-1.2676506002282294e+30,3.0
2008.0,7.0,20.0,20.0,38.0,37.0,-69.04785919189453,-61.314537048339844,-1.2676506002282294e+30,-1.2676506002282294e+30,3.0
2008.0,7.0,20.0,20.0,38.0,37.0,-68.75537109375,-60.26359176635742,-1.2676506002282294e+30,-1.2676506002282294e+30,3.0
2008.0,7.0,20.0,20.0,38.0,37.0,-68.4277572631836,-59.13920211791992,-1.2676506002282294e+30,-1.2676506002282294e+30,3.0
2008.0,7.0,20.0,20.0,38.0,37.0,-68.05646514892578,-57.928348541259766,-1.2676506002282294e+30,-1.2676506002282294e+30,3.0
2008.0,7.0,20.0,20.0,38.0,37.0,-67.62984466552734,-56.61440658569336,-1.2676506002282294e+30,-1.2676506002282294e+30,3.0
2008.0,7.0,20.0,20.0,38.0,37.0,-67.13152313232422,-55.175418853759766,-1.2676506002282294e+30,-1.2676506002282294e+30,3.0
2008.0,7.0,20.0,20.0,38.0,37.0,-66.53754425048828,-53.5811882019043,-1.2676506002282294e+30,-1.2676506002282294e+30,3.0
2008.0,7.0,20.0,20.0,38.0,37.0,-65.8108139038086,-51.78777313232422,-1.2676506002282294e+30,-1.2676506002282294e+30,3.0
2008.0,7.0,20.0,20.0,38.0,37.0,-64.88919830322266,-49.7259521484375,-1.2676506002282294e+30,-1.2676506002282294e+30,3.0
2008.0,7.0,20.0,20.0,38.0,39.0,-74.67247009277344,-120.99896240234375,-1.2676506002282294e+30,-1.2676506002282294e+30,3.0
2008.0,7.0,20.0,20.0,38.0,39.0,-74.83917236328125,-115.95703887939453,-1.2676506002282294e+30,-1.2676506002282294e+30,3.0
2008.0,7.0,20.0,20.0,38.0,39.0,-74.88825225830078,-111.83670806884766,-1.2676506002282294e+30,-1.2676506002282294e+30,3.0
2008.0,7.0,20.0,20.0,38.0,39.0,-74.87043762207031,-108.38494873046875,-1.2676506002282294e+30,-1.2676506002282294e+30,3.0
2008.0,7.0,20.0,20.0,38.0,39.0,-74.81291961669922,-105.43817138671875,-1.2676506002282294e+30,-1.2676506002282294e+30,3.0
2008.0,7.0,20.0,20.0,38.0,39.0,-74.73140716552734,-102.88363647460938,-1.2676506002282294e+30,-1.2676506002282294e+30,3.0
```

This code saves a .csv file, which can be opened by excel, a text editor, or other codes or software

Editing the Code

Change the SDS SDS to be written as output

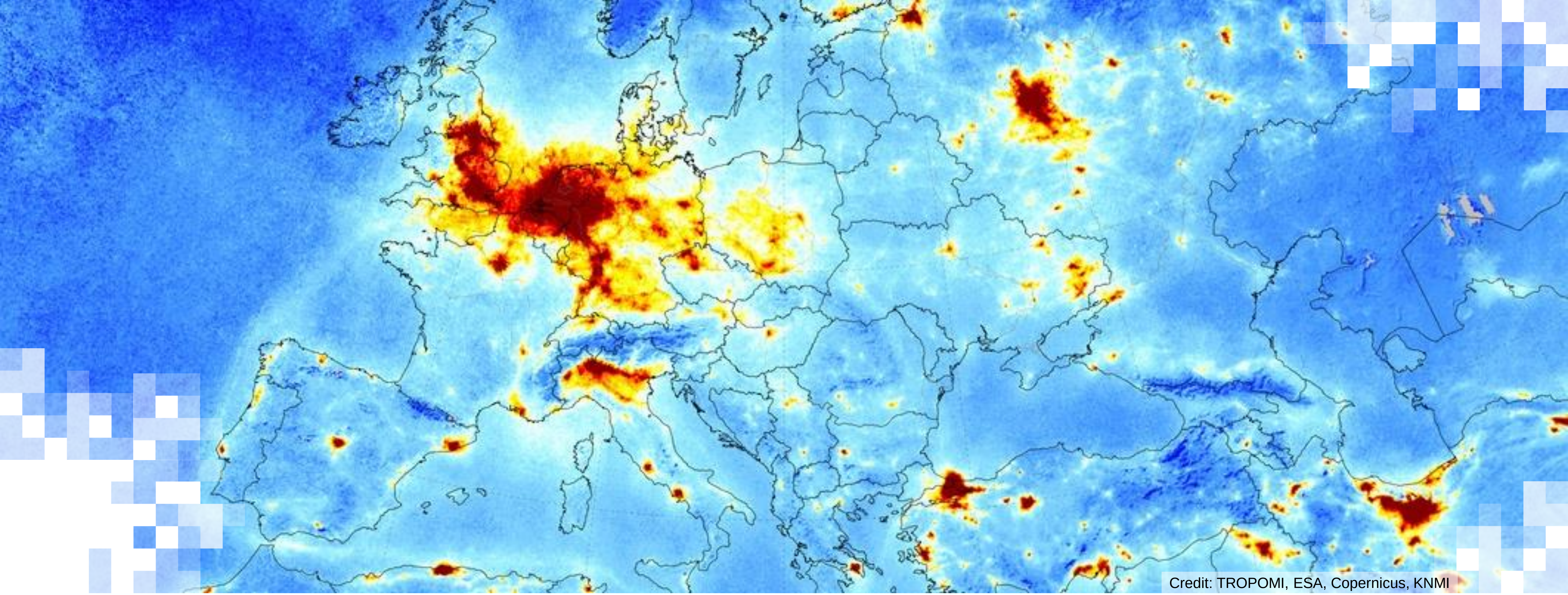
NOTE: This code will only work when all the variables listed are the same dimension. Use the “list SDS” code to view the variable dimensions



```
#loops through all files listed in the text file
for FILE_NAME in fileList:
    FILE_NAME=FILE_NAME.strip()
    user_input=input('\nWould you like to process\n' + FILE_NAME + '\n\n(Y/N)')
    if(user_input == 'N' or user_input == 'n'):
        print('Skipping...')
        continue
    else:
        file = h5py.File(FILE_NAME, 'r')   # 'r' means that hdf5 file is open in read-only mode
        #checks if the file contains NO2 or SO2 data, and reacts accordingly
        if 'NO2' in FILE_NAME:
            print('This is an OMI NO2 file. Saving... ')
            #utilizes a python dictionary to determine the variable specified by user input
            SDS=dict([(1,'ColumnAmountNO2'),(2,'ColumnAmountNO2Std'),(3,'VcdQualityFlags')])
            #this is how you access the data tree in an hdf5 file
            dataFields=file['HDFEOS']['SWATHS']['ColumnAmountNO2']['Data Fields']
            #for key in dataFields:
            #Y    print(key, dataFields[key].shape)
            geolocation=file['HDFEOS']['SWATHS']['ColumnAmountNO2']['Geolocation Fields']
        elif 'SO2' in FILE_NAME:
            print('This is an OMI SO2 file. Saving... ')
            SDS=dict([(1,'ColumnAmountSO2_PBL'),(2,'ColumnAmountO3'),(3,'QualityFlags_PBL')])
            dataFields=file['HDFEOS']['SWATHS']['OMI Total Column Amount SO2']['Data Fields']
            geolocation=file['HDFEOS']['SWATHS']['OMI Total Column Amount SO2']['Geolocation Fields']
        else:
            print('The file named :',FILE_NAME, ' is not a valid OMI file. \n')
            #if the program is unable to determine that it is an OMI SO2 or NO2 file, then it will skip to the next file
            continue
```

Applications

- This is a sample code to read and extract OMI Level 2 NO_2 and SO_2 data
- The code can be modified to extract varying SDSs into a single .csv file
- The code be easily modified to extract data over a certain region
- The output file can be opened in excel, or any other data analysis tool



Credit: TROPOMI, ESA, Copernicus, KNMI

Question & Answers