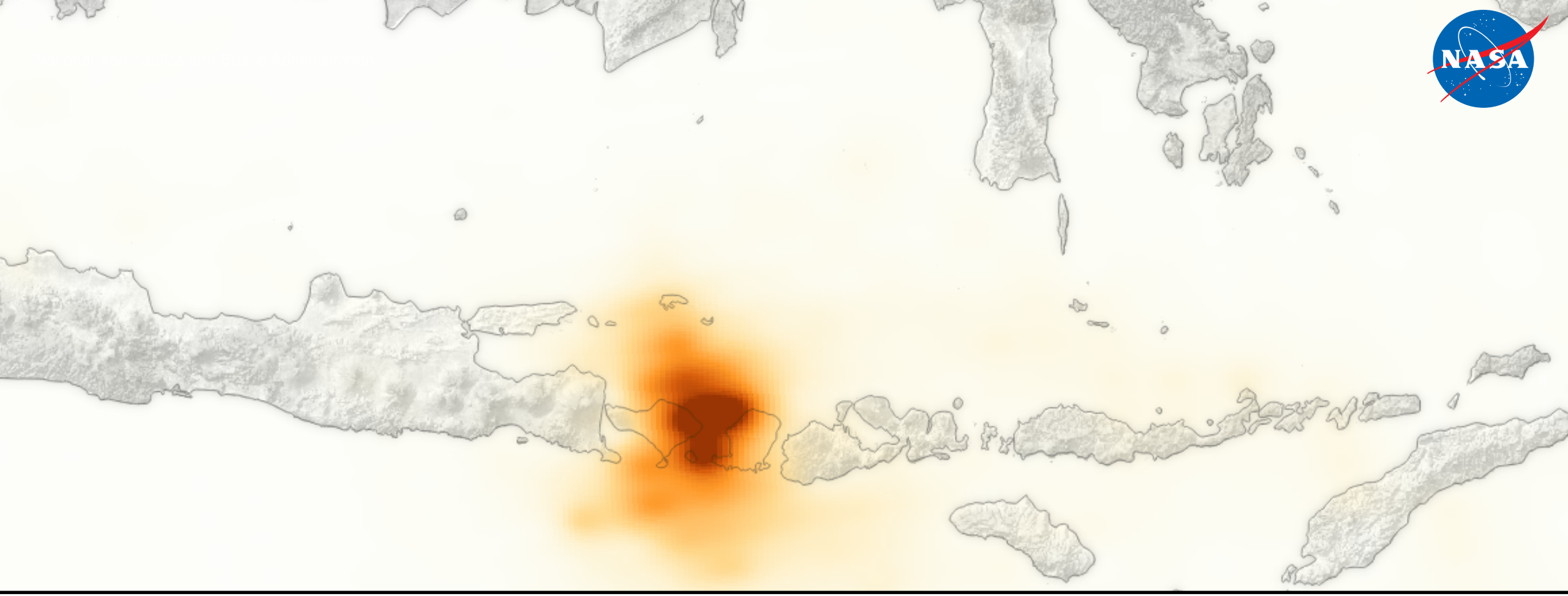
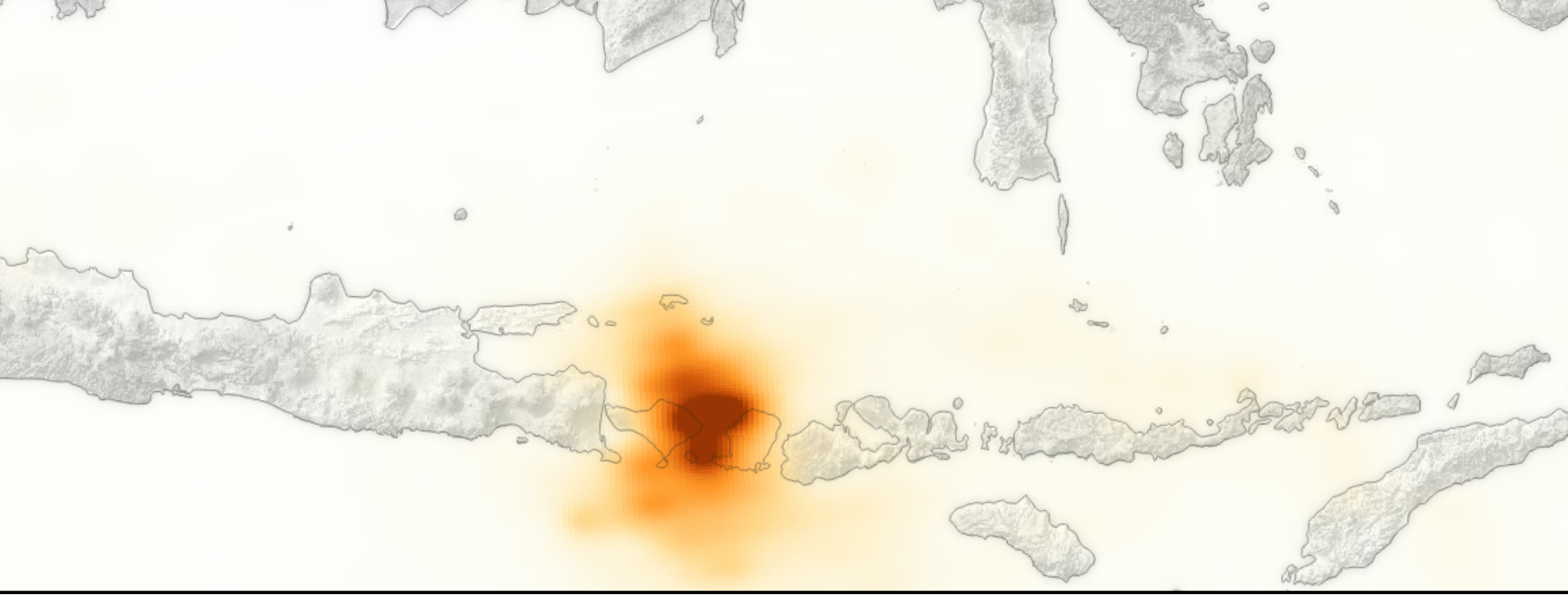


Read, Map, and Extract Level 2 OMI NO_2 and SO_2

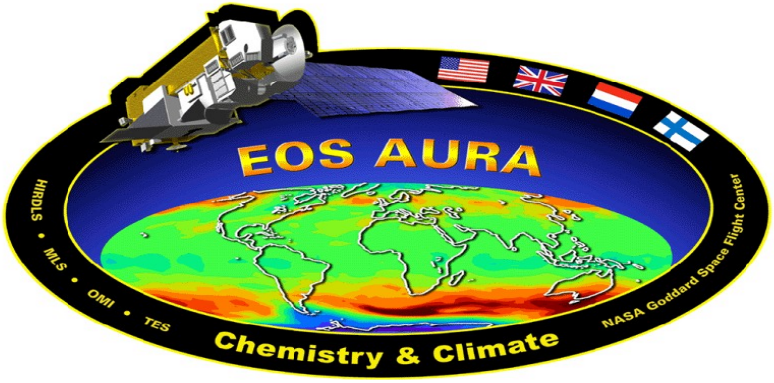
Data Analysis Tools for High Resolution Air Quality Satellite Datasets

Pawan Gupta & Melanie Follette-Cook, January 17-22, 2018



Ozone Monitoring Instrument (OMI)

Ozone Monitoring Instrument (OMI)

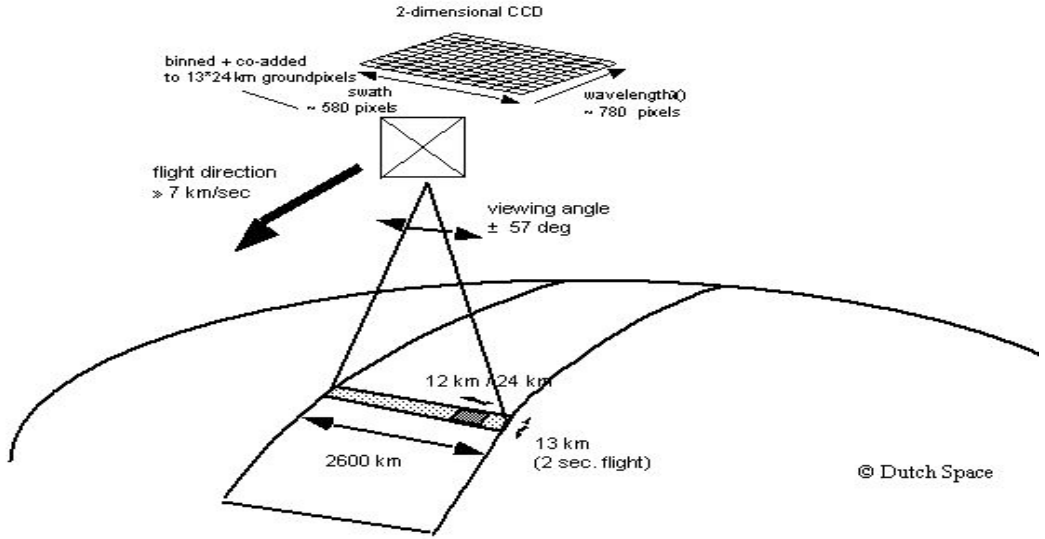


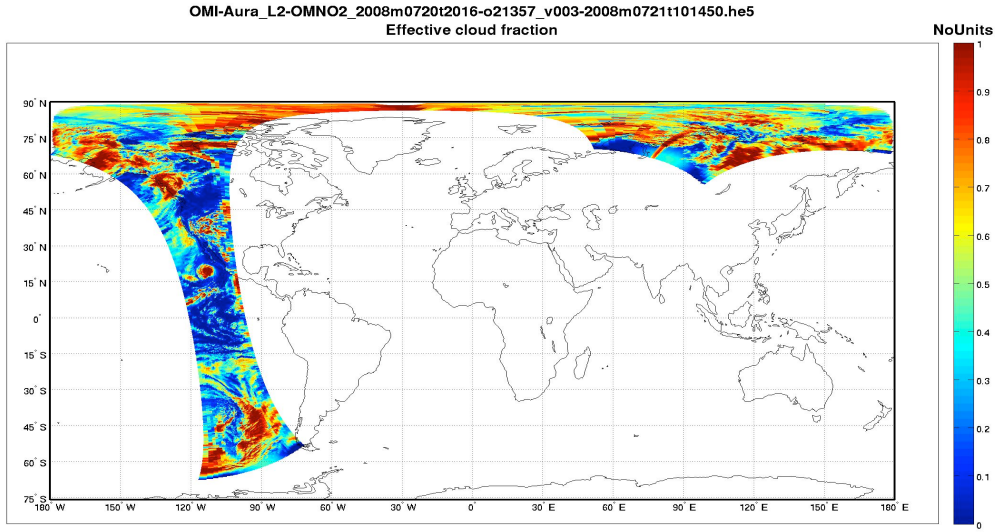
- Launched July 15, 2004
- NASA EOS Aura Satellite
- Nadir-viewing UV/Visible
  - 270 – 310 nm at 0.6 nm
  - 310 – 500 nm at 0.45 nm
- 1:45 p.m. equatorial crossing time
- 13x24 km^2 at nadir
- Daily global coverage

- Products
  - Total Column O_3
  - Tropospheric Column O_3
  - Aerosol optical depth (in UV)
  - Column Formaldehyde
  - Column NO_2
  - Tropospheric column NO_2
  - Column SO_2

Data Granule

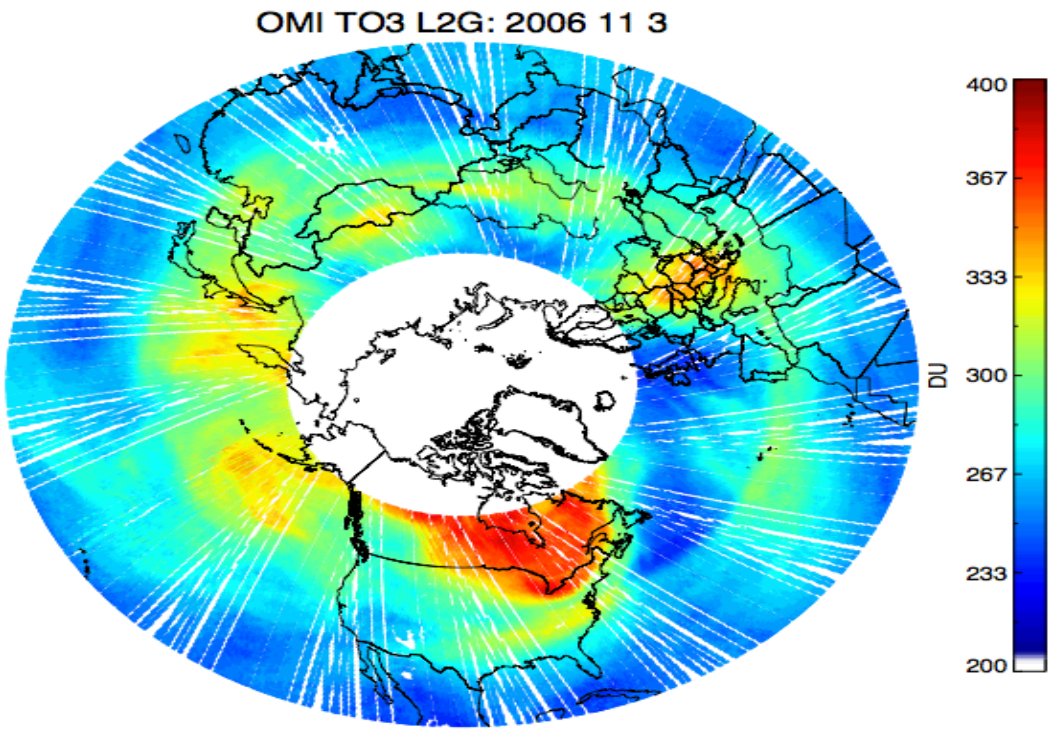
- Product File
  - covers sunlit portion of the orbit with an approx. 2,600 km wide swath
  - contains 60 binned pixels or scenes per viewing line
- 14 or 15 granules are produced daily, providing fully contiguous coverage of the globe

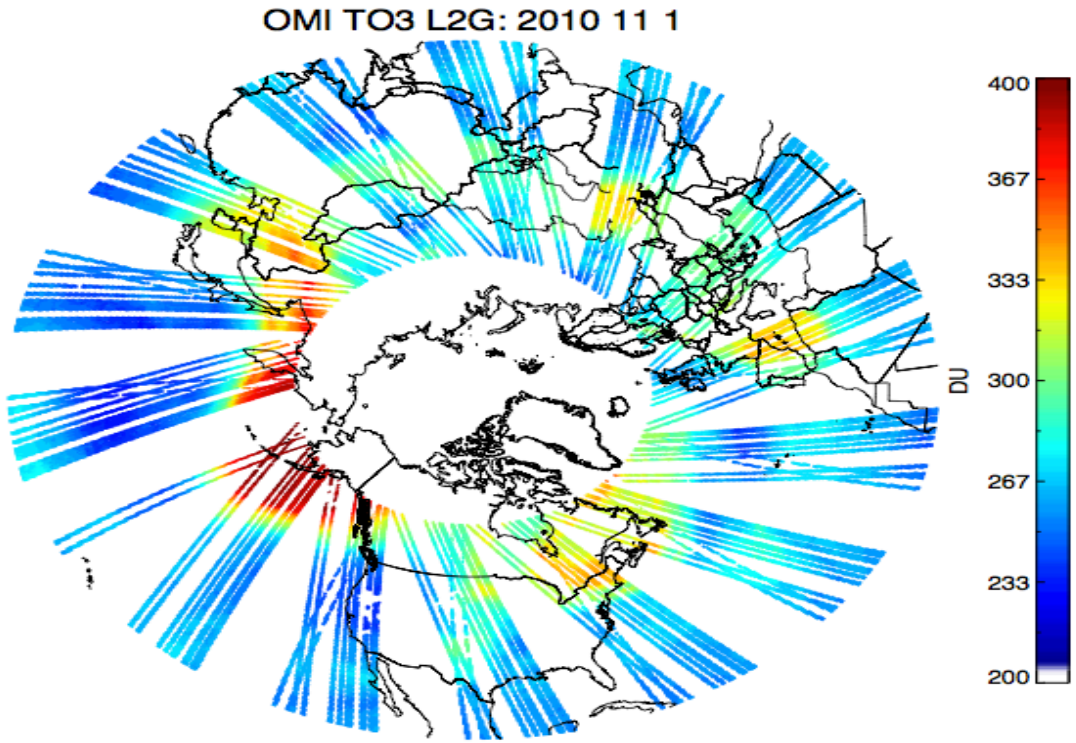




Important Information Regarding OMI

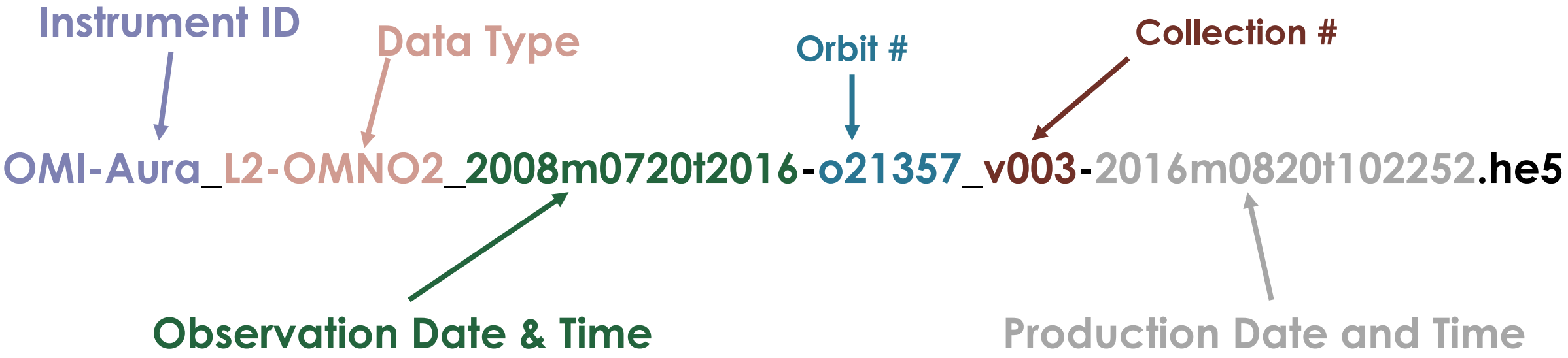
- Almost 50% data loss since 2008 (row anomaly effect)
- Affects all OMI products





Understanding an OMI File Name

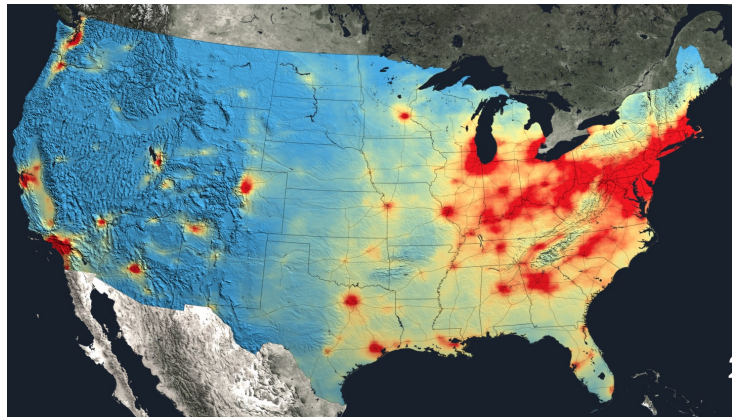
OMNO2, OMSO2



HDFLook, Panoply, IDL, Python, Fortran, MatLab, and more can be used to read the data

Nitrogen Dioxide (NO_2)

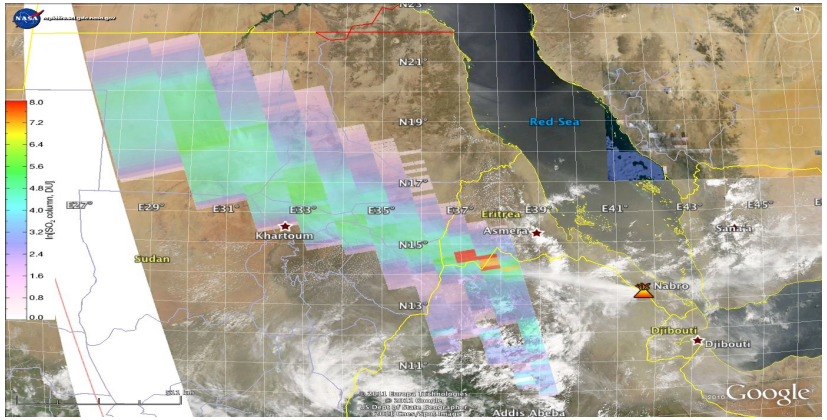
- EPA criteria pollutant, ozone precursor and health irritant
- Sources: Fires, industrial and transportation sources, stationary sources (e.g. power plants), lightning
- High concentrations in the planetary boundary layer (PBL) make tropospheric column amounts more useful for estimating surface levels



airquality.gsfc.nasa.gov

Sulfur Dioxide (SO_2)

- EPA Criteria pollutant
- Contributes to acid deposition and linked to adverse health effects
- Sources: Volcanoes, coal and oil burning



Aqua MODIS visible image of the Nabro (Eritrea) eruption on June 13, 2011 and the SO2 plume overlaid.

Quantification of Gas Abundances - Units

Satellite Tracer	Units
OMI O_3, SO_2	Dobson Units (DU)
OMI NO_2	Molecules/cm^2

$$1 \text{ DU} = 2.69 \times 10^{16} \text{ molec/cm}^2$$

OMI NO_2 Parameter (SDS) information (OMNO2)

SDS name	Description	Unit	Notes
ColumnAmountNO2Trop	Tropospheric Column NO_2	Molec / cm^2	• Use only rows 4-54 (where the first row = 0) • Use only scenes with: radiative cloud fraction < 0.5 solar zenith angle < 85° terrain reflectivity < 0.3
TerrainReflectivity		Unitless	Scale factor: 0.001
CloudRadianceFraction		Unitless	Scale factor: 0.001
SolarZenithAngle		Deg	In geolocation fields

- All fill values are high negative numbers: ($-2.^{100} \approx -1.26765\times10^{30}$)

OMI SO_2 Level-2 Product Summary (OMSO2)

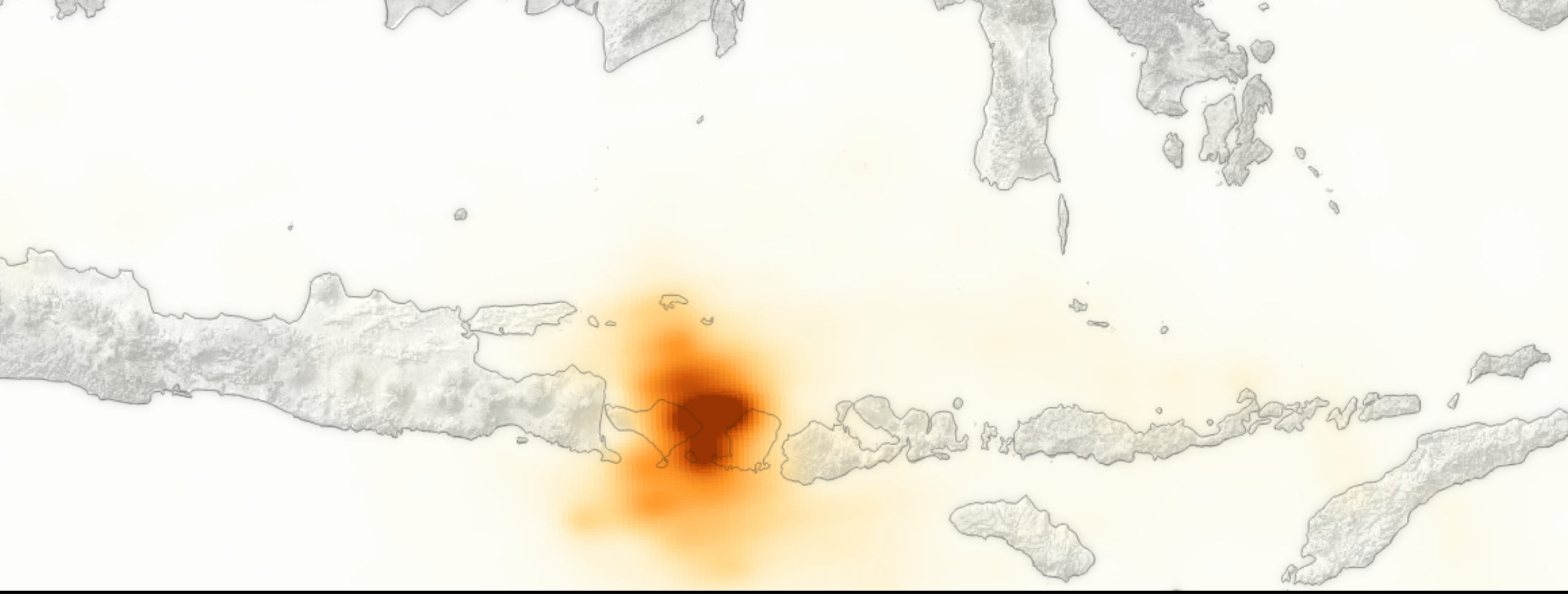
SO_2 Product	SDS Name	Estimated center of plume	Use
PBL SO_2	ColumnAmountSO2_PBL	0.9 km	Near-surface pollution
TRL SO_2	ColumnAmountSO2_TRL	3 km	Volcanic degassing
TRM SO_2	ColumnAmountSO2_TRM	8 km	Plumes from moderate eruptions, and long range pollution transport
STL SO_2	ColumnAmountSO2_STL	18 km	Explosive volcanic eruptions

Note: Each retrieval listed here yields total column values, and represents a different assumption of SO_2 plume height. These should therefore not be added together.

OMI SO_2 Parameter (SDS) information (OMSO2)

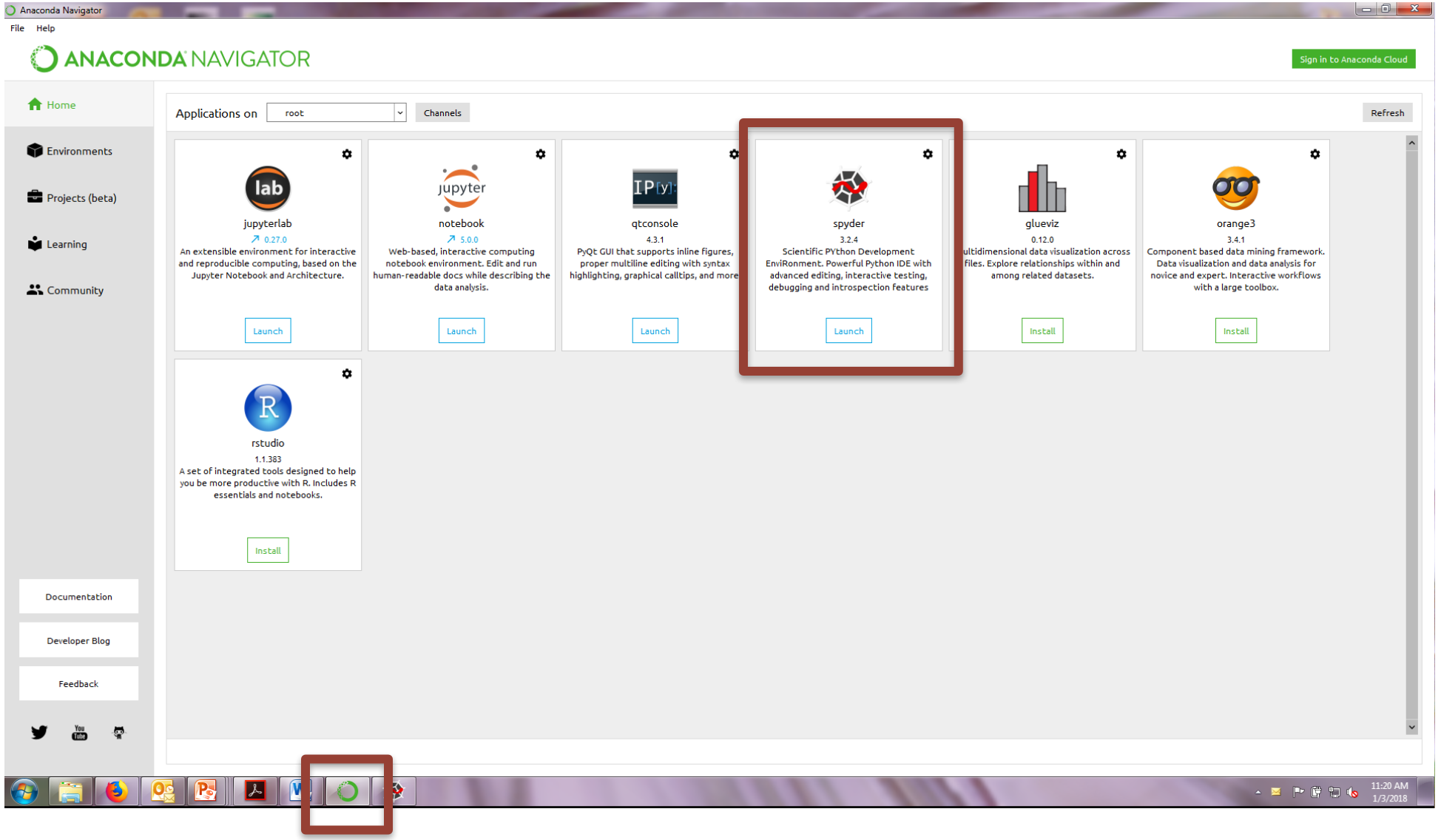
SDS name	Description	Unit	Notes
ColumnAmountSO2_PBL	Total Column SO2	DU	• use only rows 4-54 (where the first row = 0) • use only scenes with radiative cloud fraction < 0.3 solar zenith angle < 70°
ColumnAmountSO2_TRL /TRM/STL	Total Column SO2	DU	• All rows can be used • Use only scenes with solar zenith angle < 70°
RadiativeCloudFraction		Unitless	No scale factor
SolarZenithAngle		Deg	In geolocation fields

- As of the latest version (v1.3), the OMSO2 documentation does not recommend using the included data quality flags for screening
- All fill values are high negative numbers: ($-2.^{100} \approx -1.26765\times10^{30}$)

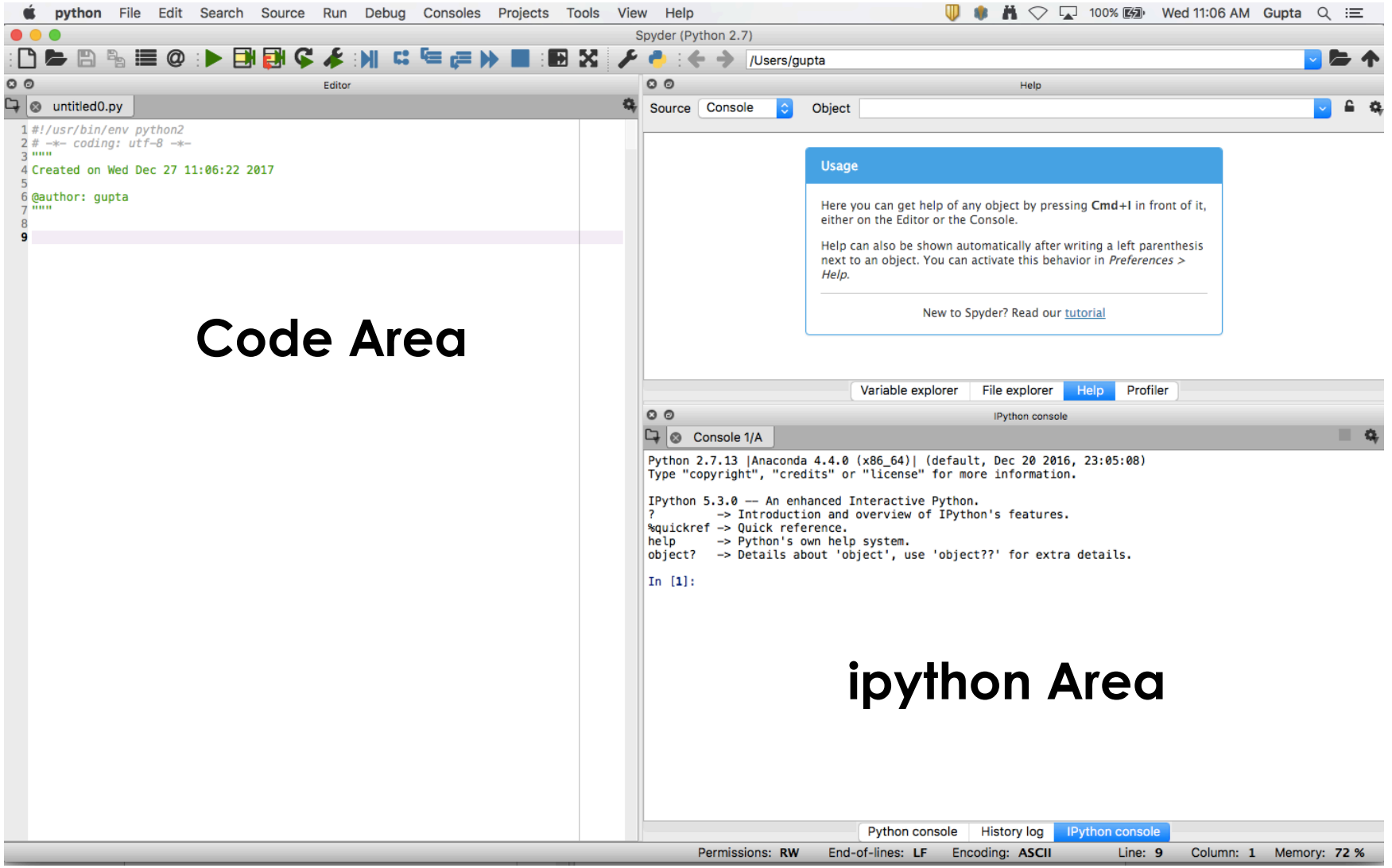


Getting Ready with Python

Anaconda & Spyder Editor

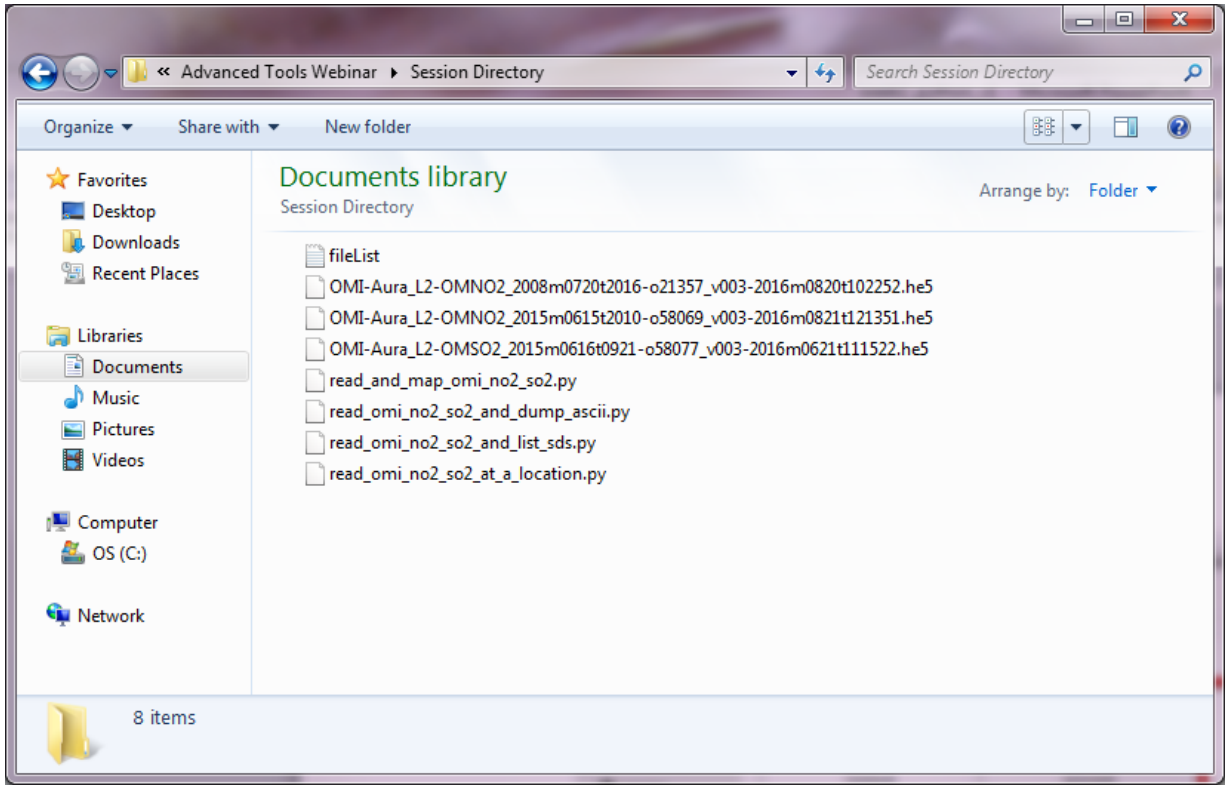


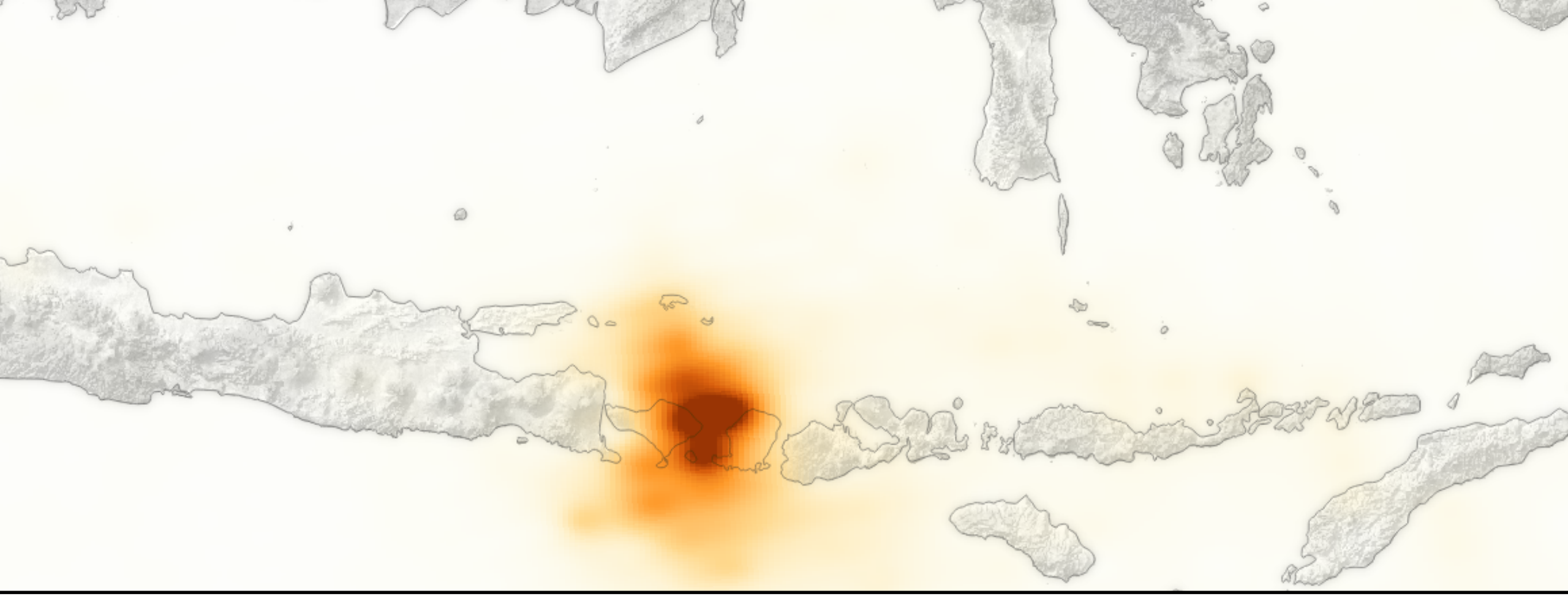
Spyder View


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
#!/usr/bin/env python2
-*- coding: utf-8 -*-
"""
Created on Wed Dec 27 11:06:22 2017
@author: gupta
"""
Code Area
Help
Source Console Object
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
Variable explorer File explorer Help Profiler
IPython console
Console 1/A
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
Python console History log IPython console
Permissions: RW End-of-lines: LF Encoding: ASCII Line: 9 Column: 1 Memory: 72 %

Current Directory View & fileList.txt

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





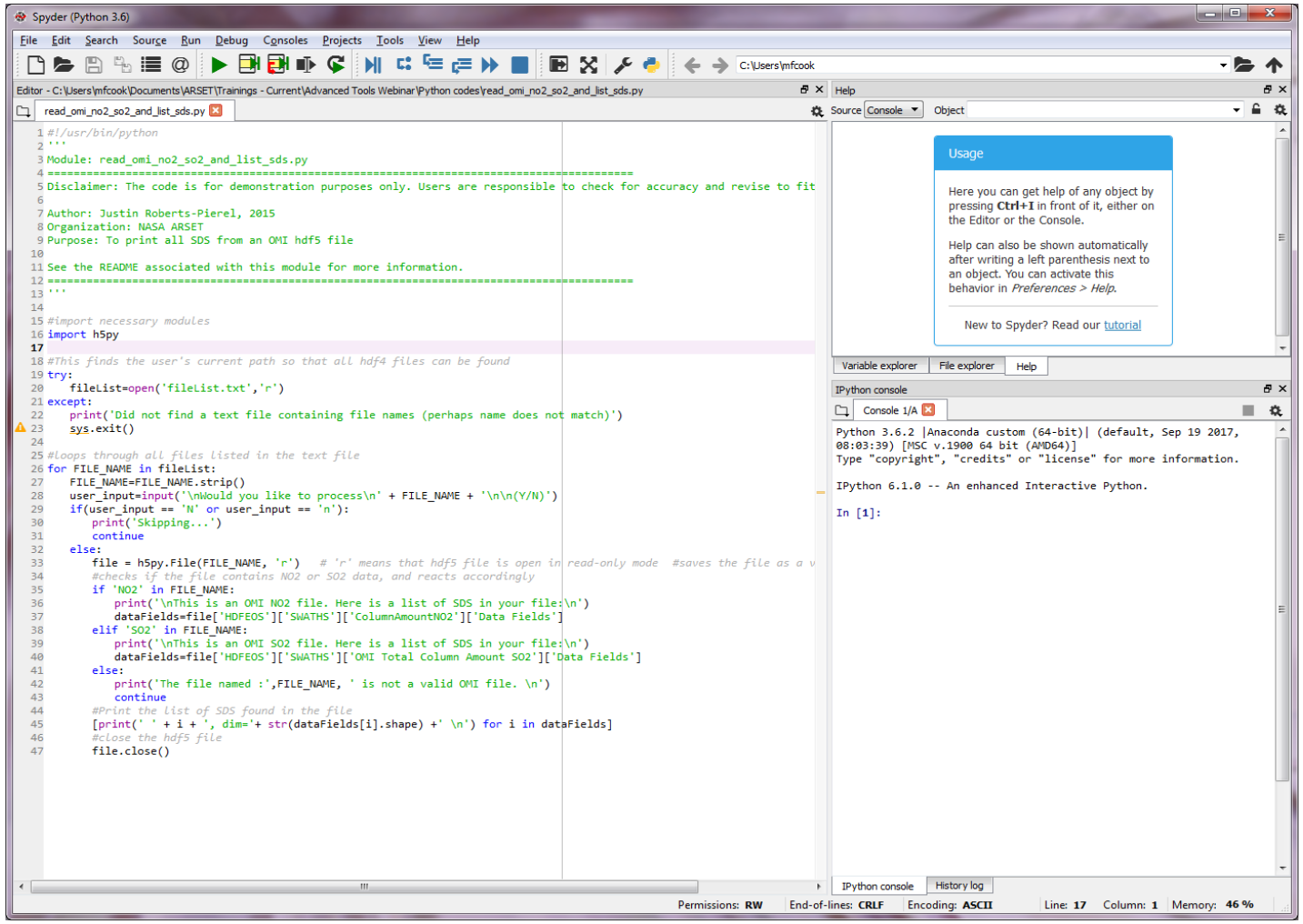
Read an OMI NO_2 File (HDF) and Print SDS List

Print Scientific Data Sets (SDSs)

read_omi_no2_so2_and_list_sds.py

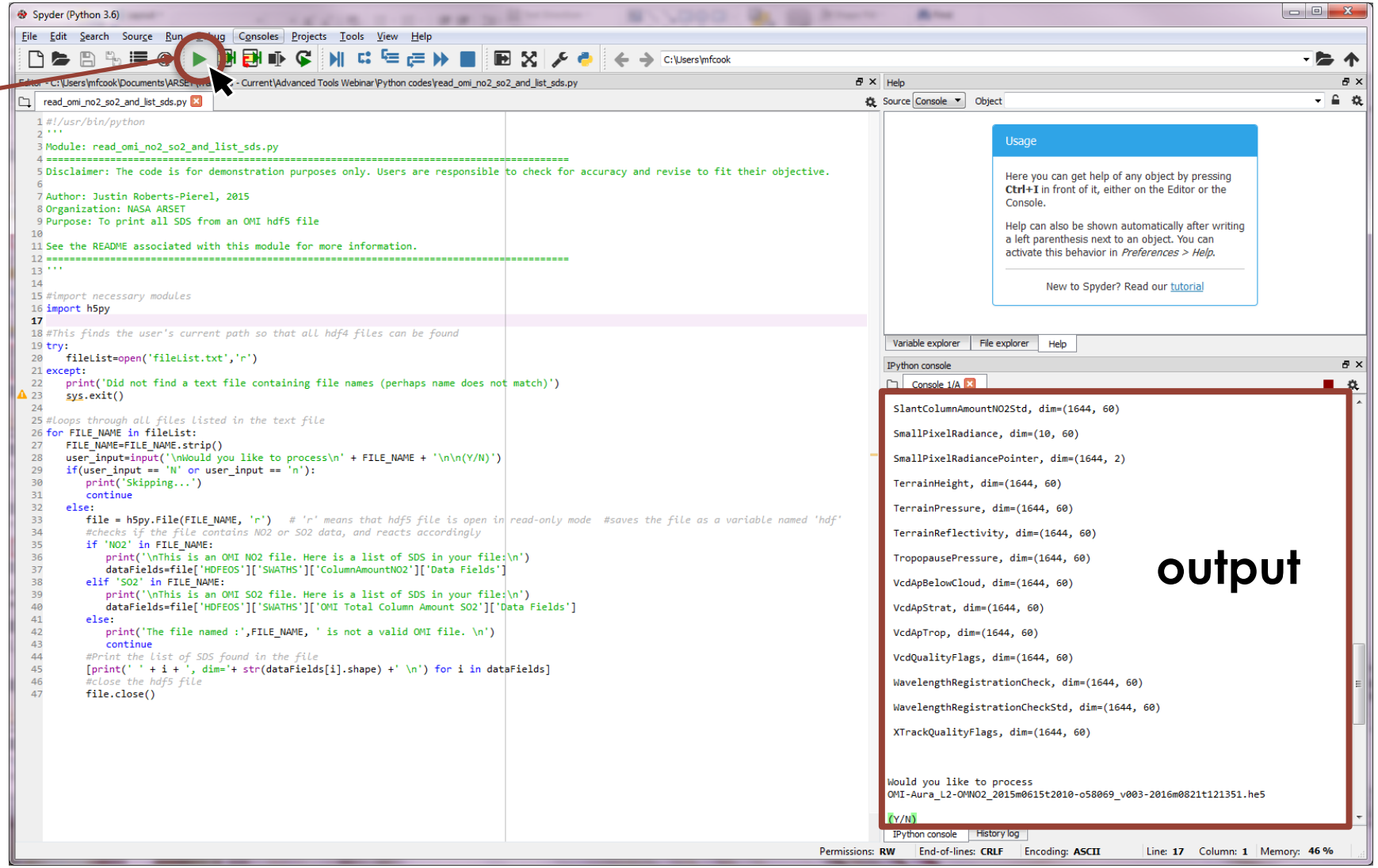
Purpose: read OMI NO_2 or SO_2 level 2 data files in HDF format and print all the **Scientific Data Sets** (SDS)

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



Running and Output

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



Editing the Code

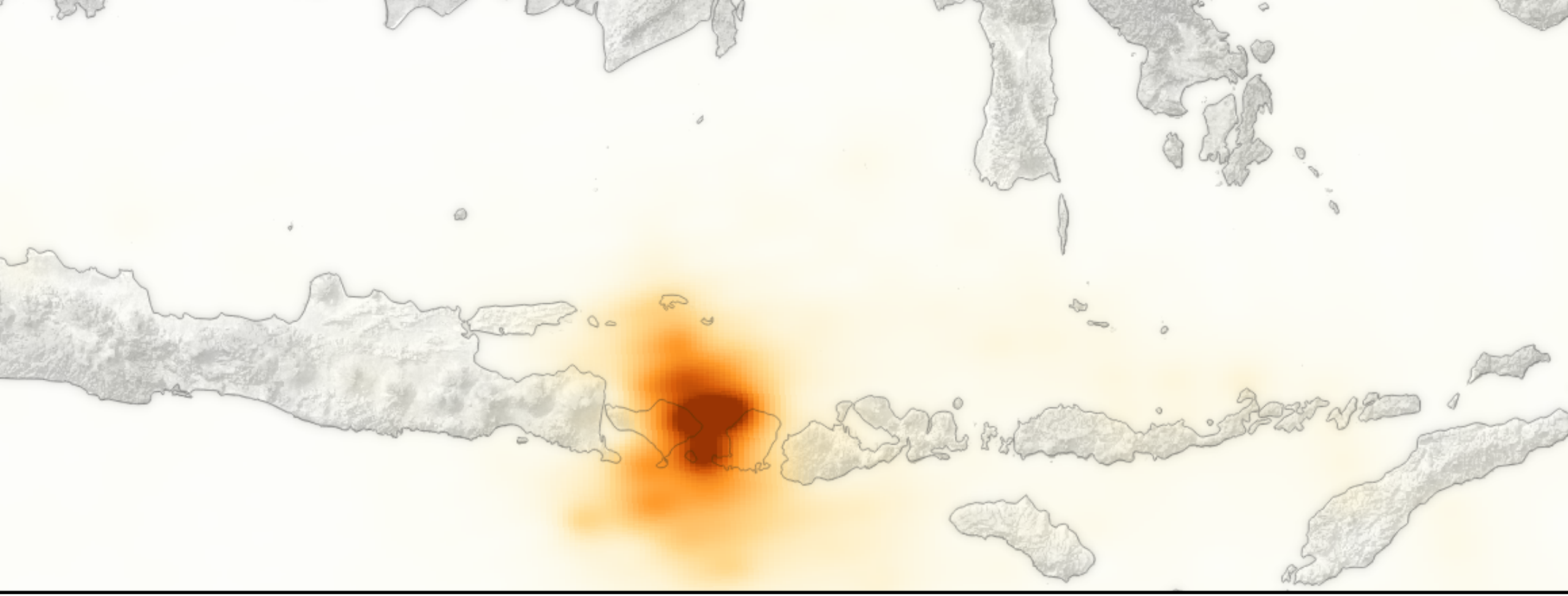
```python
#!/usr/bin/python
'''
Module: read_omi_no2_so2_and_list_sds.py
================================================================================
Disclaimer: The code is for demonstration purposes only. Users are responsible to check for accuracy and revise to fit their objective.

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 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()
```

Change the file to be read

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

Applications

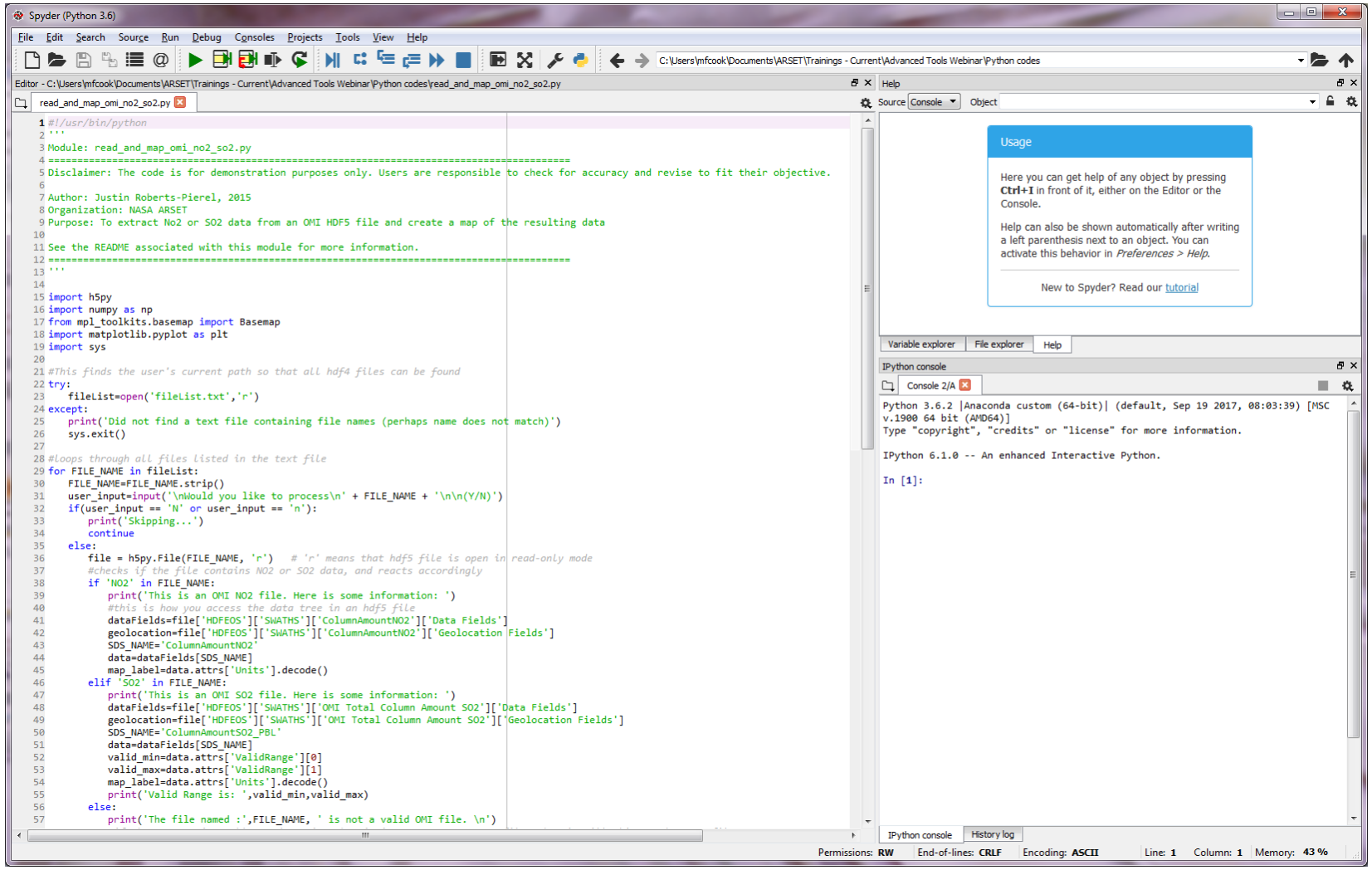
- OMI Level 2 NO_2 and SO_2 data are provided in the HDF file
- Each HDF file contains several geophysical parameters
- Special codes and tools are required to open the HDF files
- This code helps users see the names and dimensions of the available SDSs inside an HDF file for further analysis



Map NO_2 or SO_2

Plot and Save a Map of OMI NO_2 or SO_2

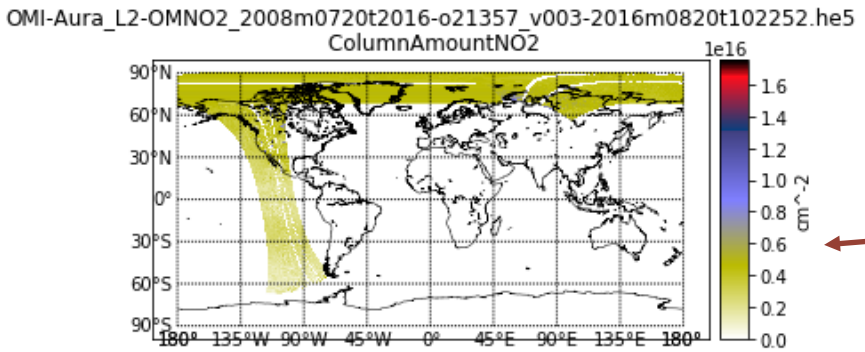
read_and_map_omi_so2_no2.py



Running and Output

NO_2/SO_2 Statistics

```
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
```



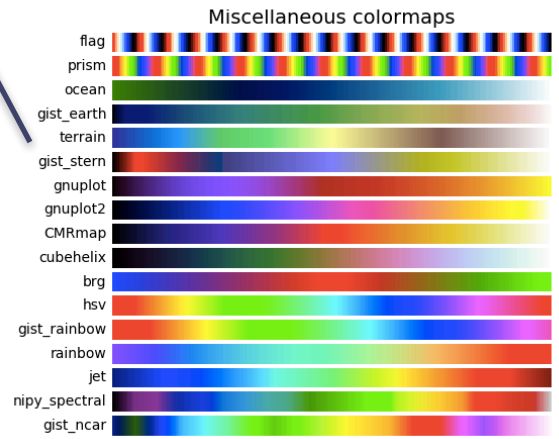
Output Map

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

Editing the Code

Change the color scale

```
93        if is_map == 'Y' or is_map == 'y':
94            data = np.ma.masked_array(data, np.isnan(data))
95            m = Basemap(projection='cyl', resolution='l',
96                        llcrnrlat=-90, urcrnrlat = 90,
97                        llcrnrlon=-180, urcrnrlon = 180)
98            m.drawcoastlines(linewidth=0.5)
99            m.drawparallels(np.arange(-90., 120., 30.), labels=[1, 0, 0, 0])
100           m.drawmeridians(np.arange(-180, 180., 45.), labels=[0, 0, 0, 1])
101           my_cmap = plt.cm.get_cmap('gist_stern_r')
102           my_cmap.set_under('w')
103           m.pcolormesh(lon, lat, data, latlon=True, vmin=0, vmax=np.nanmax(dat
104           cb = m.colorbar()
105           cb.set_label(map_label)
106           plt.autoscale()
107           #title the plot
108           plt.title('{0}\n {1}'.format(FILE_NAME,
109           fig = plt.gcf()
110           # Show the plot window.
111           plt.show()
112       #once you close the map it asks if you'd l
113           is_save=str(input('\nWould you like to
114           if is_save == 'Y' or is_save == 'y':
115               #saves as a png if the user would l
116               pngfile = '{0}.png'.format(FILE_NAM
117               fig.savefig(pngfile)
118       #close the hdf5 file
119       file.close()
```



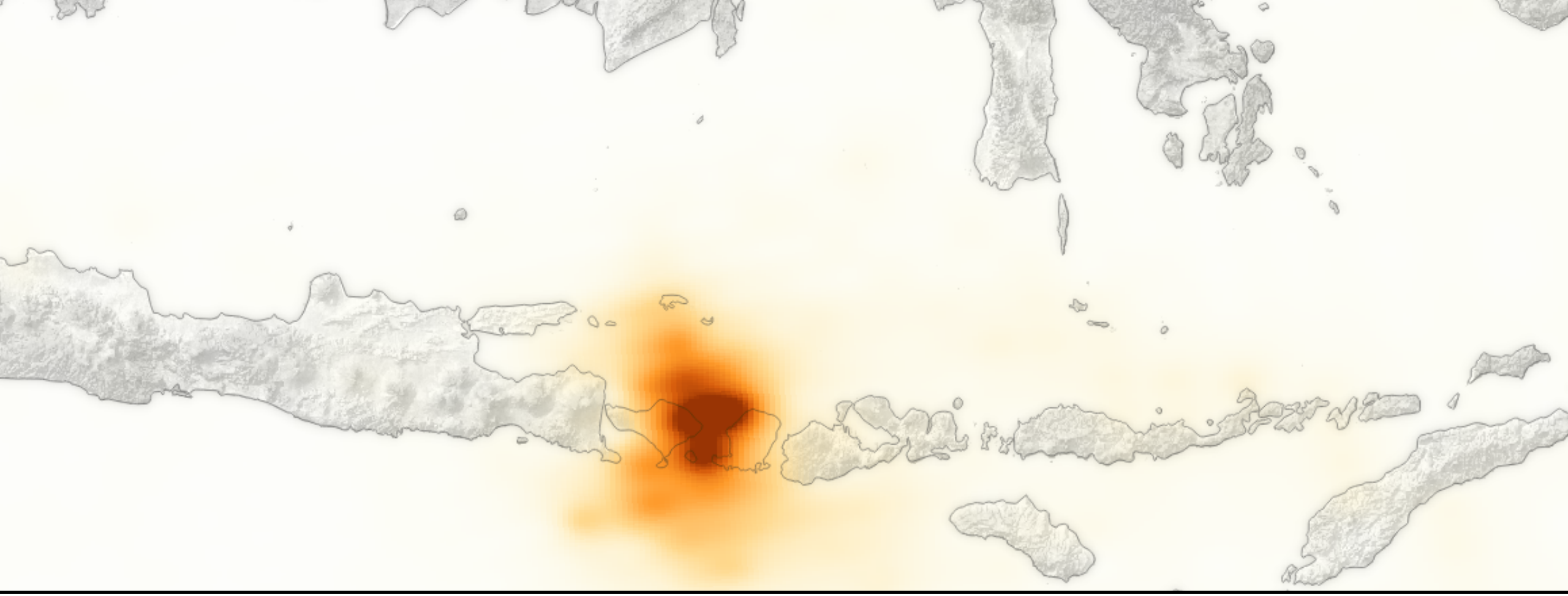
https://matplotlib.org/examples/color/colormaps_reference.html

Change the SDS to plot

```
28 #Loops through all files listed in the text file
29 for FILE_NAME in fileList:
30     FILE_NAME=FILE_NAME.strip()
31     user_input=input('\nWould you like to process\n' + FILE_NAME + '\n\n(Y/N)')
32     if(user_input == 'N' or user_input == 'n'):
33         print('Skipping...')
34         continue
35     else:
36         file = h5py.File(FILE_NAME, 'r')   # 'r' means that hdf5 file is open in read-only mode
37         #checks if the file contains NO2 or SO2 data, and reacts accordingly
38         if 'NO2' in FILE_NAME:
39             print('This is an OMI NO2 file. Here is some information: ')
40             #this is how you access the data tree in an hdf5 file
41             dataFields=file['HDFEOS']['SWATHS']['ColumnAmountNO2']['Data Fields']
42             geolocation=file['HDFEOS']['SWATHS']['ColumnAmountNO2']['Geolocation Fields']
43             SDS_NAME='ColumnAmountNO2'
44             data=dataFields[SDS_NAME]
45             map_label=data.attrs['Units'].decode()
46         elif 'SO2' in FILE_NAME:
47             print('This is an OMI SO2 file. Here is some information: ')
48             dataFields=file['HDFEOS']['SWATHS']['OMI Total Column Amount SO2']['Data Fields']
49             geolocation=file['HDFEOS']['SWATHS']['OMI Total Column Amount SO2']['Geolocation Fields']
50             SDS_NAME='ColumnAmountSO2_PBL'
51             data=dataFields[SDS_NAME]
52             valid_min=data.attrs['ValidRange'][0]
53             valid_max=data.attrs['ValidRange'][1]
54             map_label=data.attrs['Units'].decode()
55             print('Valid Range is: ',valid_min,valid_max)
56         else:
57             print('The file named :',FILE_NAME, ' is not a valid OMI file. \n')
58             #if the program is unable to determine that it is an OMI SO2 or NO2 file, then it will skip
59             continue
```

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

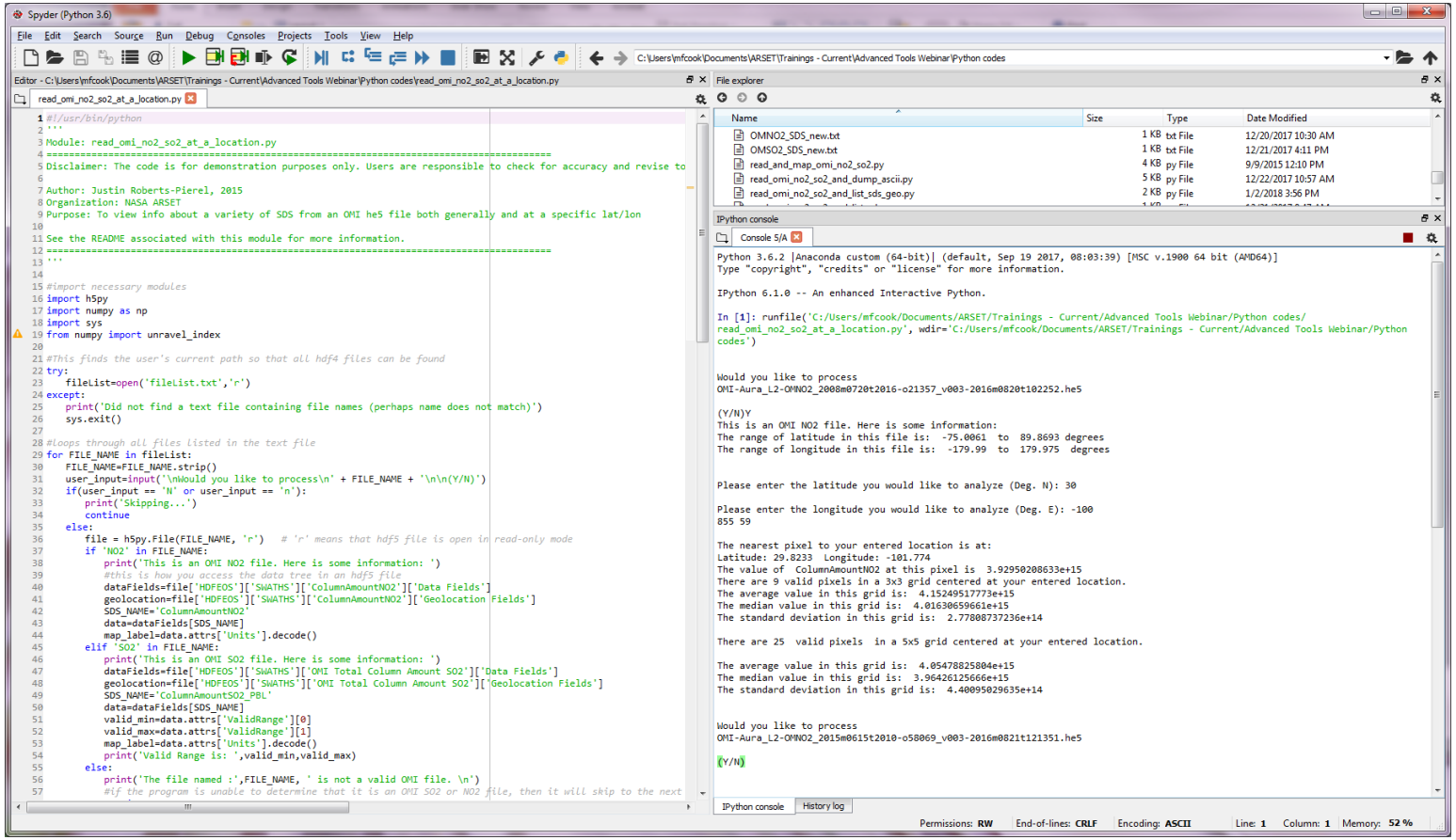


Extract NO_2/SO_2 at a given location

Extract Level 2 OMI NO_2/SO_2 Values

read_omi_no2_so2_at_a_location.py

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

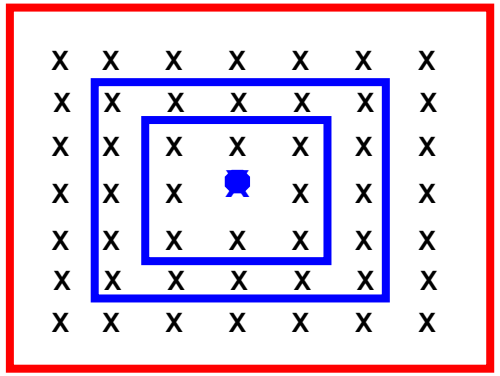


Running and Output

Type "Y" to process file, "N" to skip

Latitude and Longitude of the Station

Outputs



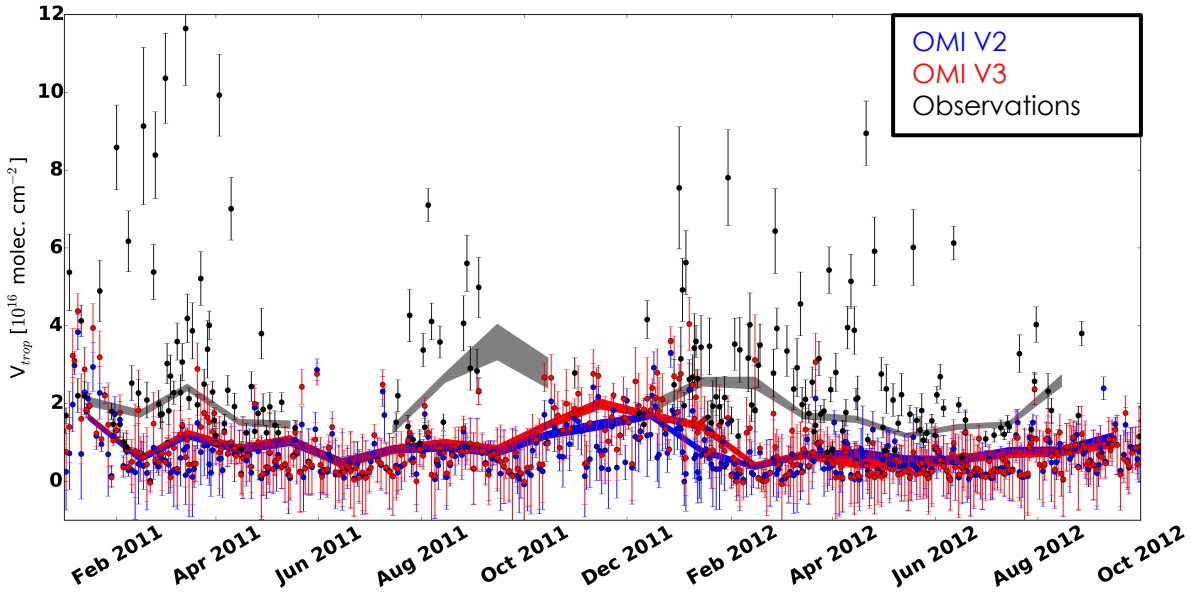
```
Would you like to process
OMI-Aura_L2-OMNO2_2008m0720t2016-o21357_v003-2016m0820t102252.he5

(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)
```

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 = 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
```

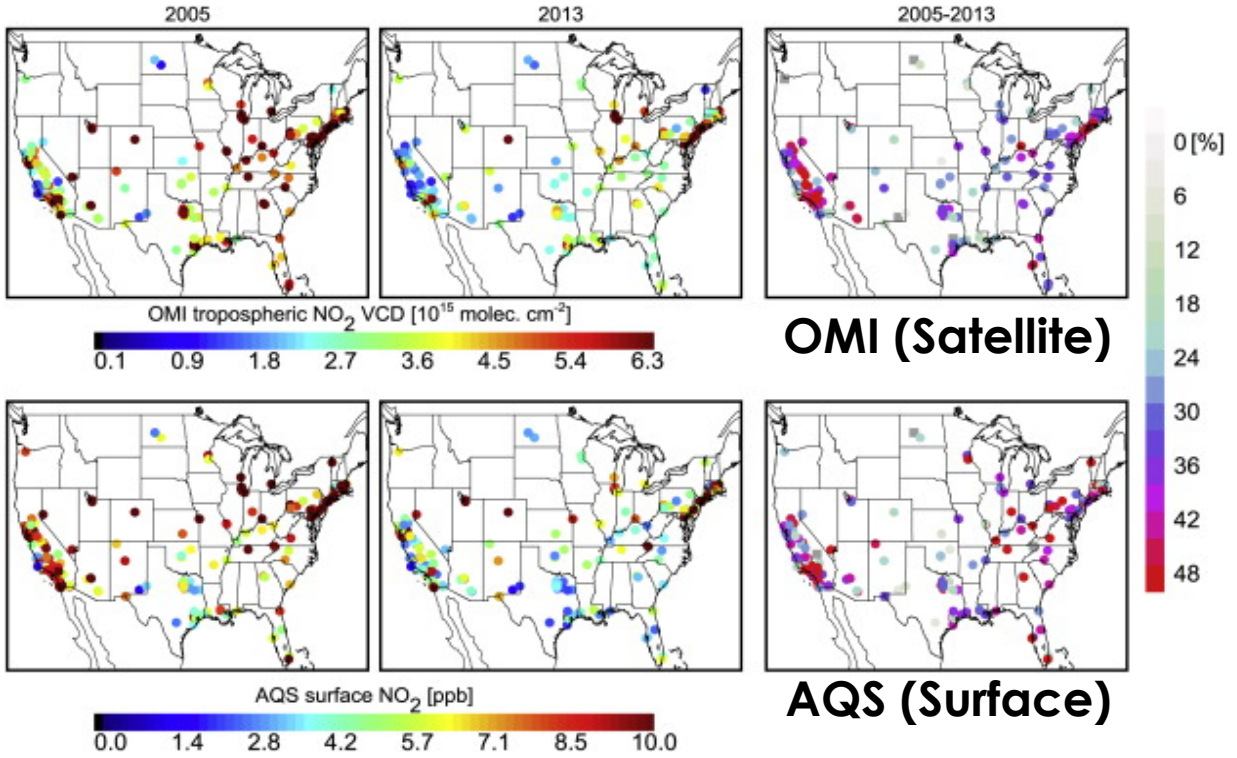

Applications

Satellite Validation

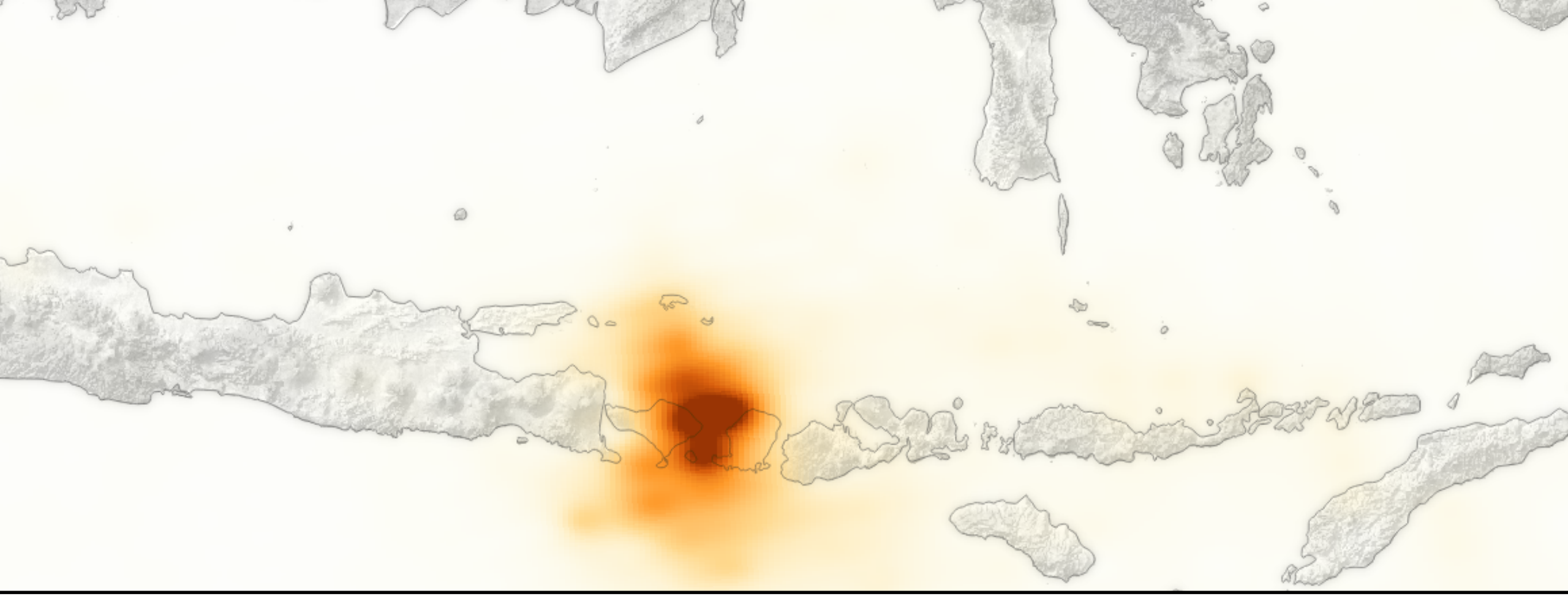


Source: Krotkov et al. (2017)

Column vs. Surface Relationship and Trends



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

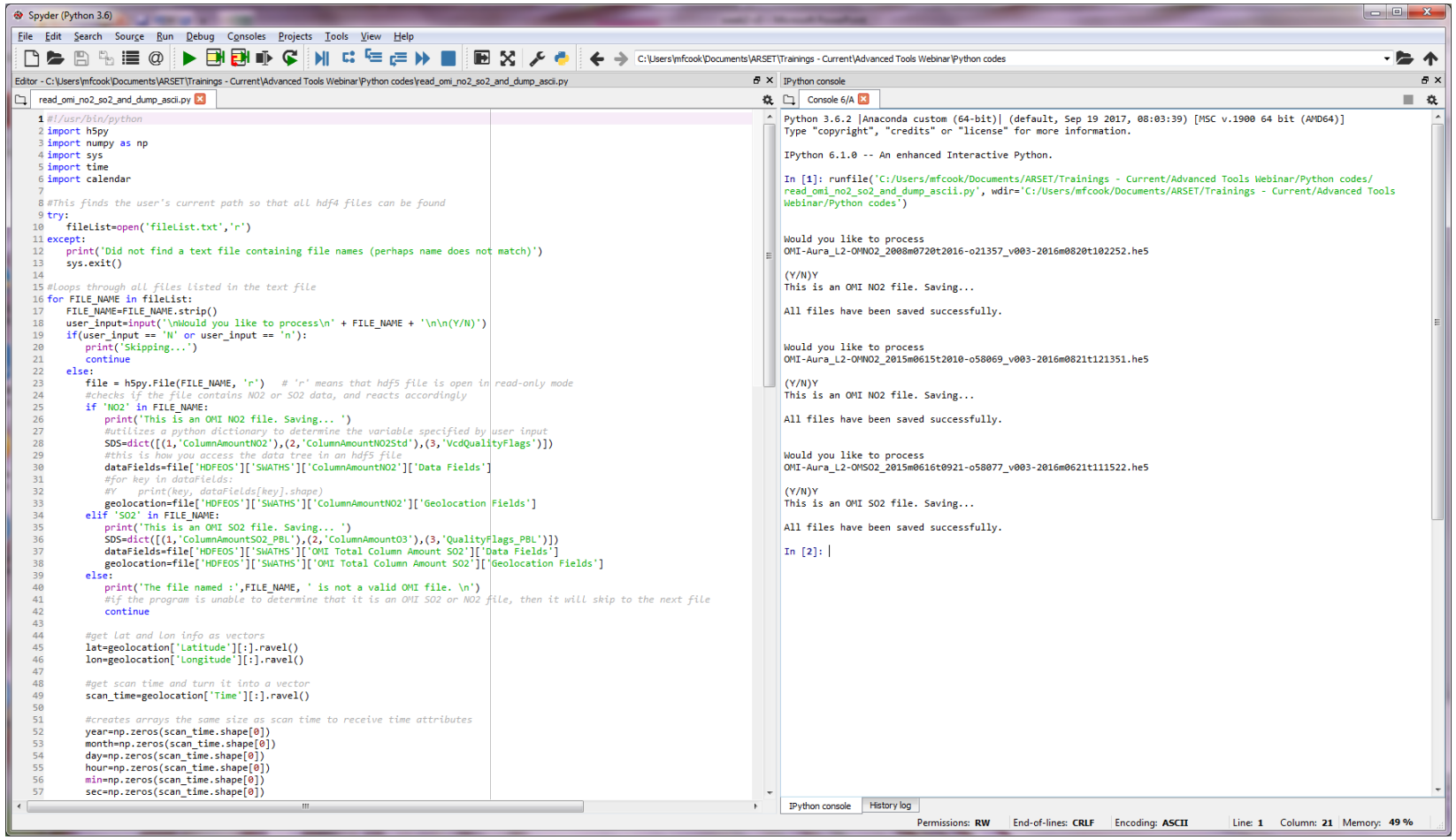


Output HDF variables to CSV

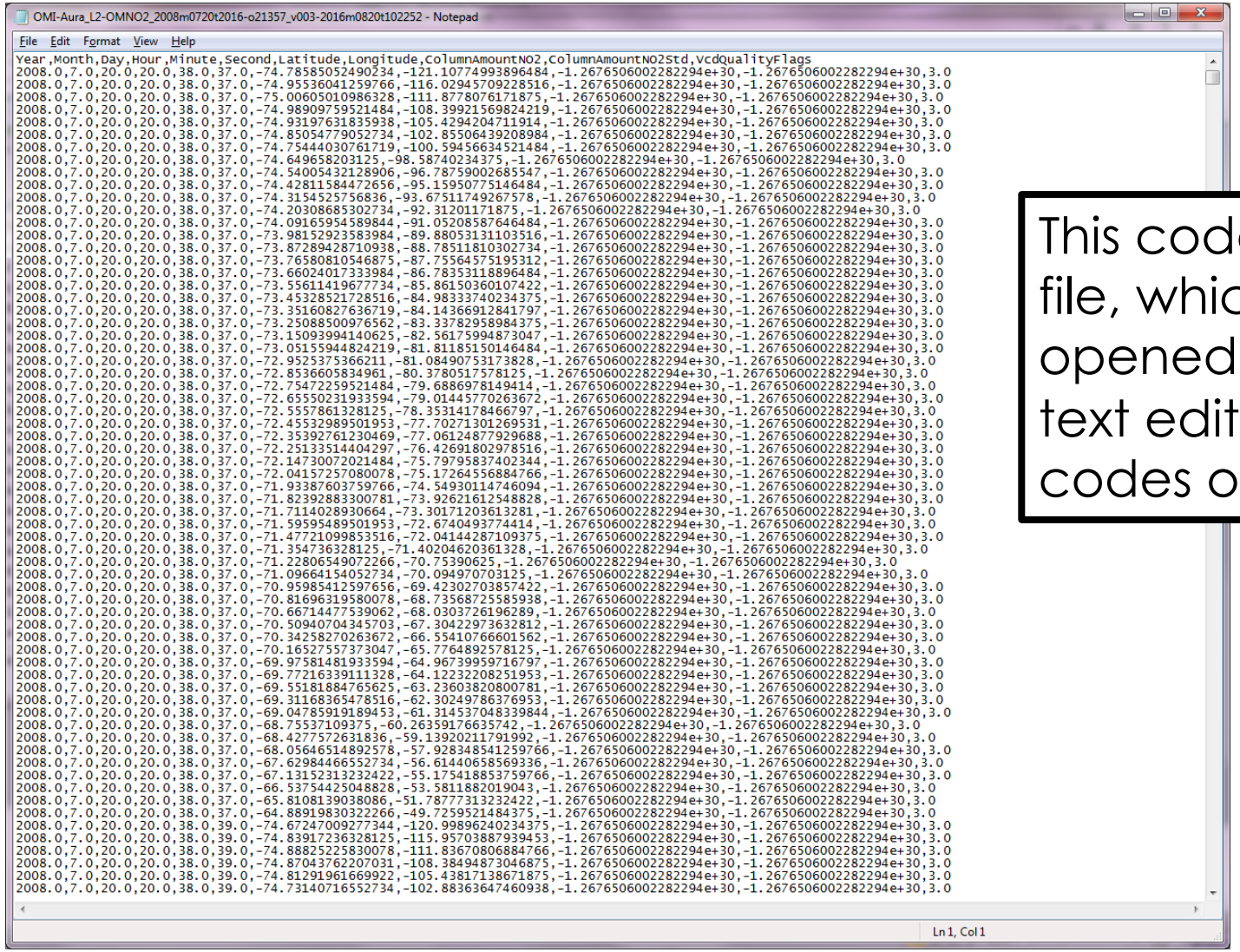
Output OMI NO2/SO2 HDF Variables to a CSV File

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



Output



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

Editing the Code

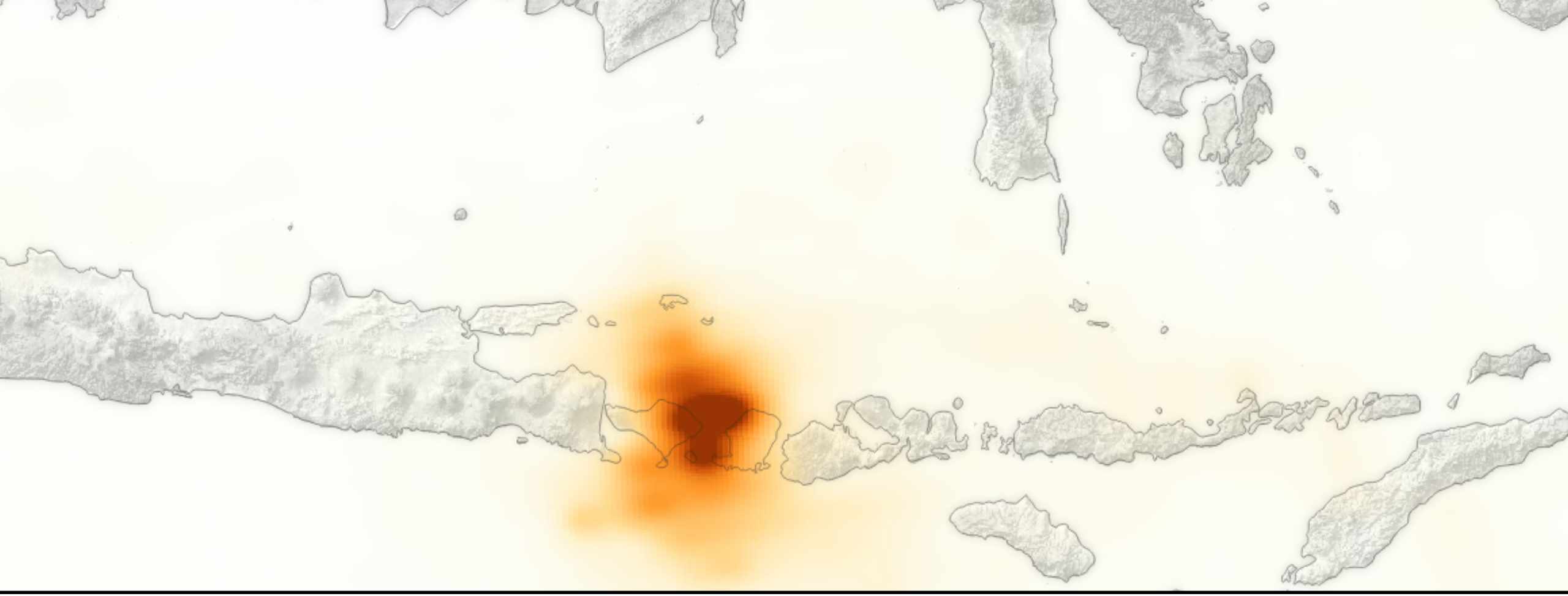
Change the SDS to be written as output

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



Questions & Answers