TEMPO via the Python Interface to Remote Sensing Information Gateway

Presented: Barron H. Henderson

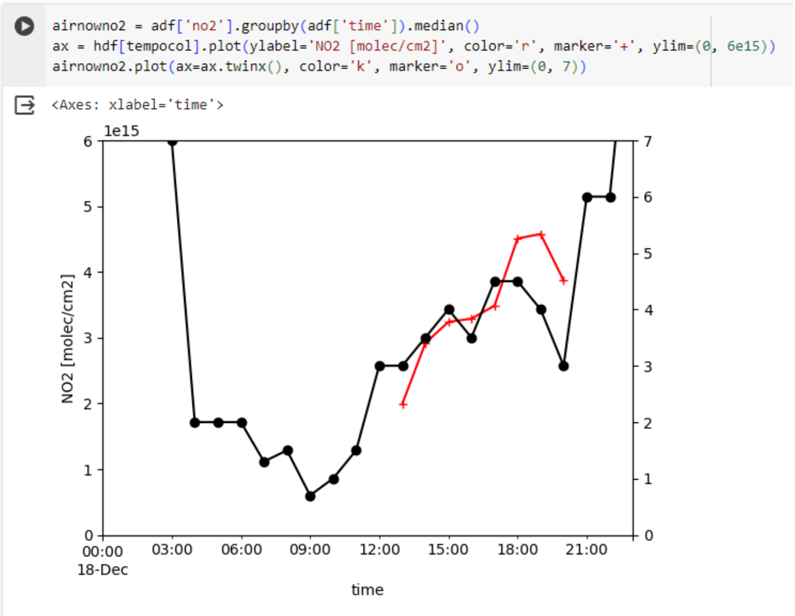
Date: 2024-05-09

RSIG Team: Jim J. Szykman, Luke Valin, Todd Plessel, Matt Freeman

***Disclaimer**: The views expressed in this presentation are those of the authors and do not necessarily reflect the views or policies of the U.S. Environmental Protection Agency.*

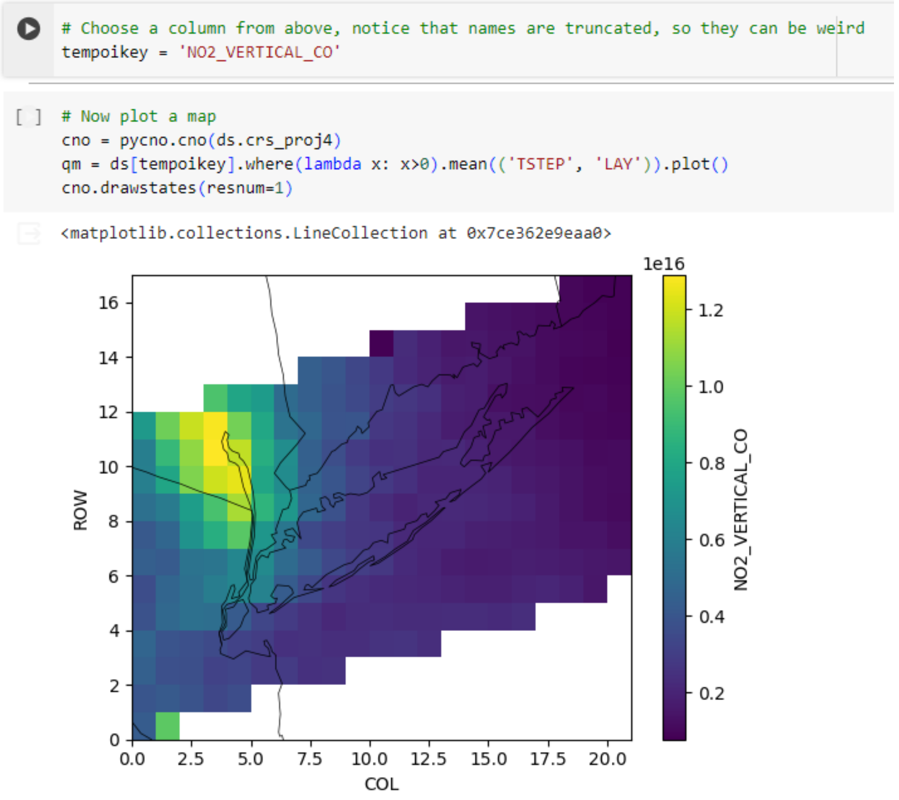
Overview

- RSIG make satellite data easier;
- When data is easy to use, we all win.
- Today, I want to make it easy for you to process TEMPO data.
- Skills covered
  - Compare TEMPO to observations (AirNow and Pandora)
  - Create a TEMPO NO2 map.
  - Create a TEMPO Surface NO2 estimate.
  - Adapt other tutorials.



```
airnowno2 = adf['no2'].groupby(adf['time']).median()
ax = hdf[tempocol].plot(ylabel='NO2 [molec/cm2]', color='r', marker='+', ylim=(0, 6e15))
airnowno2.plot(ax=ax.twinx(), color='k', marker='o', ylim=(0, 7))
```

```
<Axes: xlabel='time'>
```



```
# Choose a column from above, notice that names are truncated, so they can be weird
tempoikey = 'NO2_VERTICAL_CO'
```

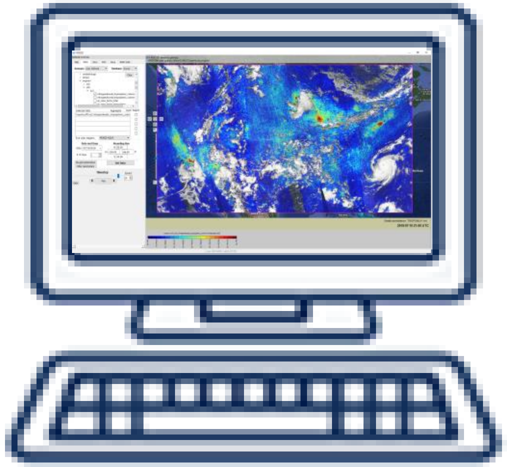
```
# Now plot a map
cno = pycno.cno(ds.crs_proj4)
qm = ds[tempoikey].where(lambda x: x>0).mean(('TSTEP', 'LAY')).plot()
cno.drawstates(resnum=1)
```

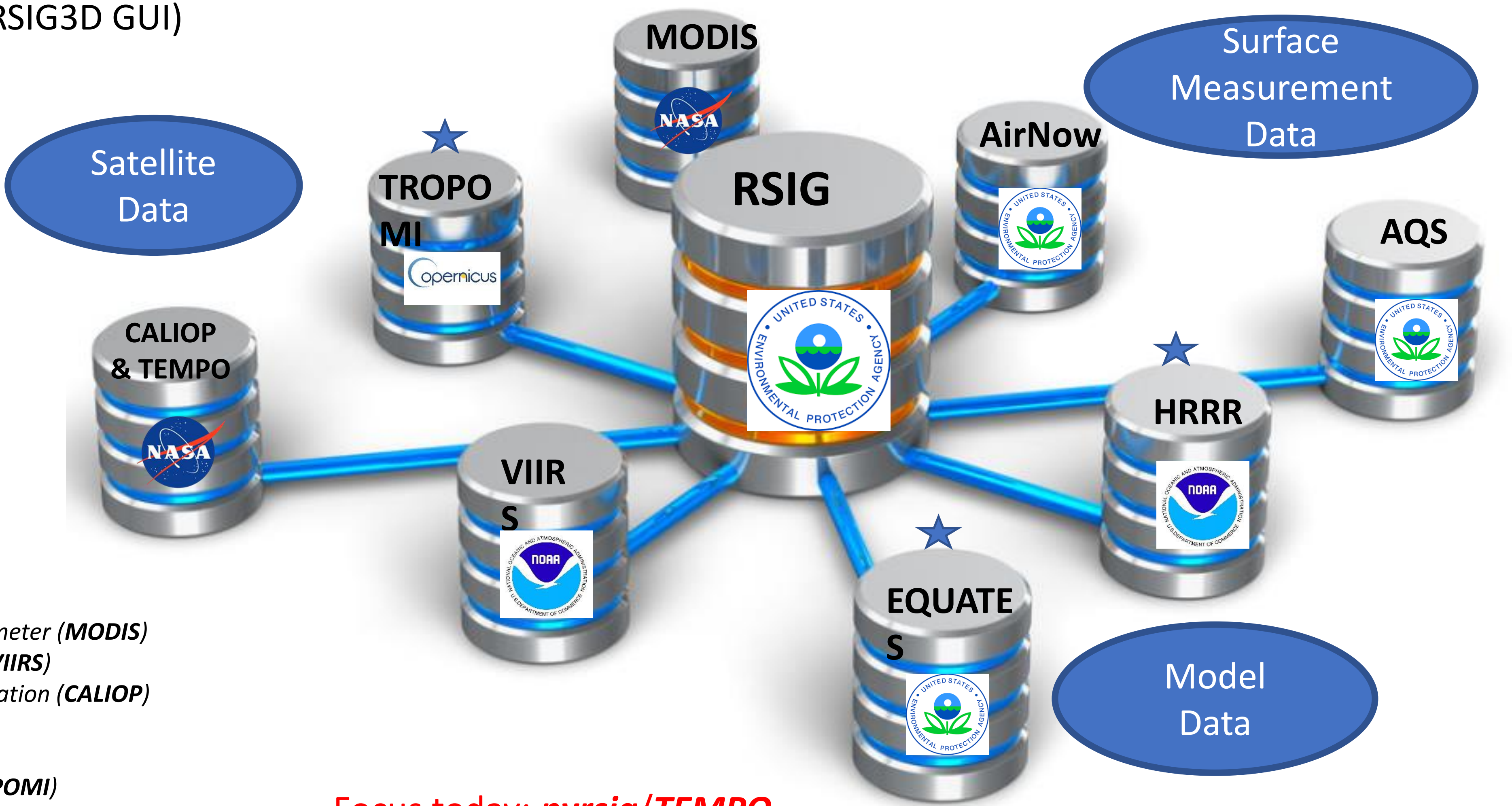
```
<matplotlib.collections.LineCollection at 0x7ce362e9eaa0>
```

What is the Remote Sensing Information Gateway?

- ***Free*** multi-platform, ***scriptable*** access to terabytes (TB) of air quality model, measurement, and satellite data from EPA, NOAA, NASA, ESA, etc.

- Multiple access methods
  - Graphical User Interface (RSIG3D GUI)
  - Python Interface (pyrsig)
  - Custom API shell scripts





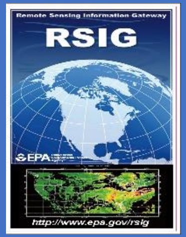
Popular Air Quality Data Sets:

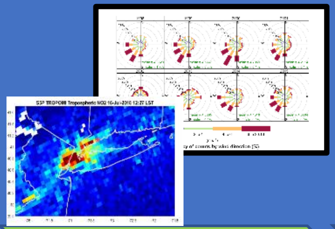
- *Moderate Resolution Imaging Spectroradiometer (**MODIS**)*
- *Visible Infrared Imaging Radiometer Suite (**VIIRS**)*
- *Cloud-Aerosol Lidar with Orthogonal Polarization (**CALIOP**)*
- *Air Quality System (**AQS**)*
- *EPA's Air QUAlity TimE Series (**EQUATES**)*
- *TROPOspheric Monitoring Instrument (**TROPOMI**)*
- *Tropospheric Emissions: Monitoring of POllution (**TEMPO**)*
- *High-Resolution Rapid Refresh (**HRRR**)*

- Focus today: ***pyrsig/TEMPO***

Enabling Improved Data Access

Typical Data Analytics Flow





Problem Definition → Data Preparation & Exploration → Advance Data Analysis → Report Results

*

RSIG's Function

Input Data Sets from Distributed Archives → Time, Date, and Domain, with pre-select Data Quality Flags → Output Enriched Data Sets in Analysis Ready Formats

Input Data Sets from Distributed Archives → Data Set Visualization

RSIG Used to Support (point and click)

Air Quality Modelers
- ➢ Model Evaluation
- ➢ Case Study Analysis

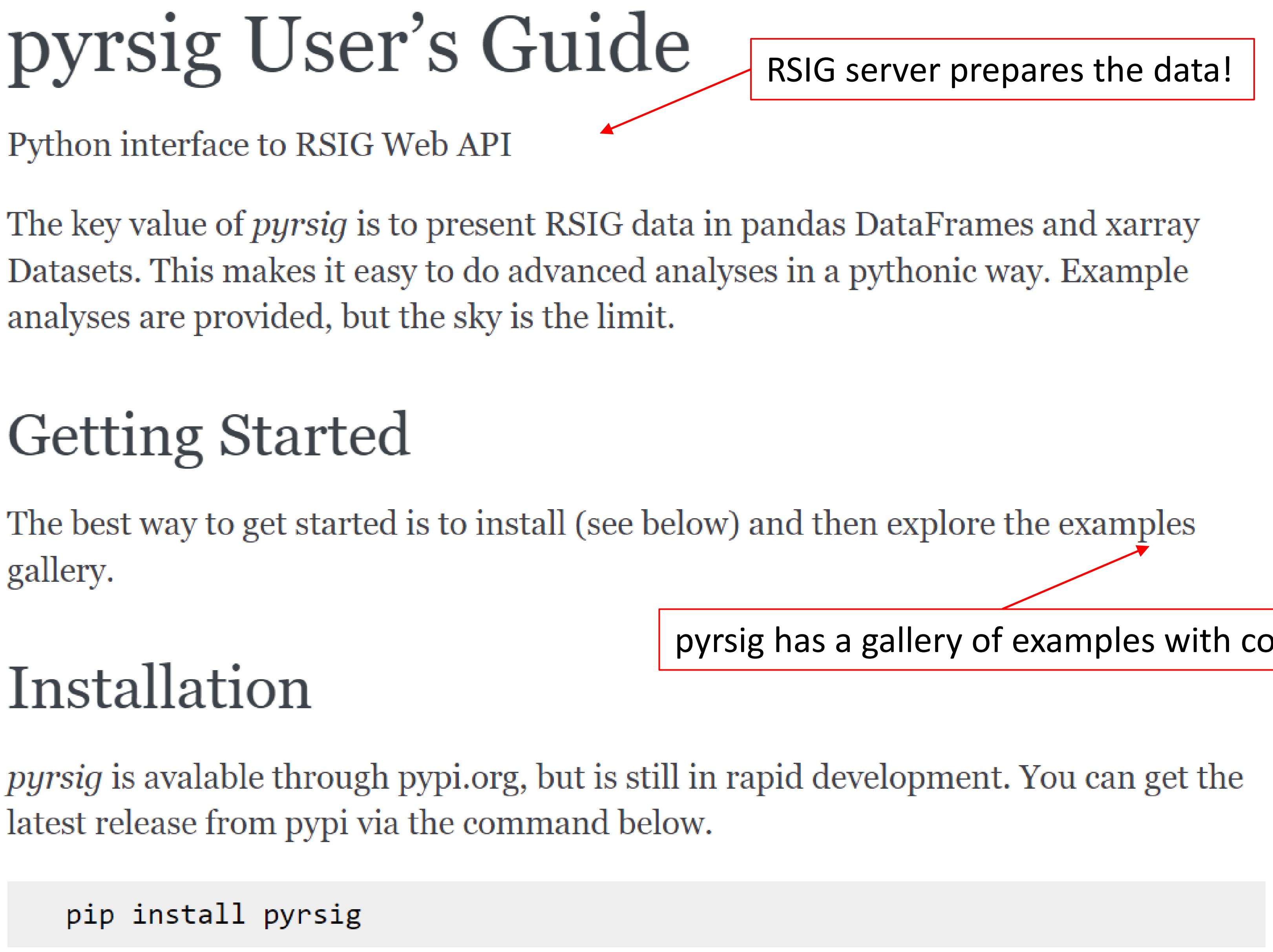
Air Quality Forecasters
- ➢ Retrospective Analysis

Scientists & Researchers
- ➢ Exposure studies
- ➢ Inverse modeling studies
- ➢ Improving air quality models

Students
- ➢ Thesis/Dissertation Research

New** ***pyrsig
- ➢ Custom analyses including GIS
- ➢ Runs on laptop, server, or cloud
- ➢ Cloud apps even run on phones!

https://barronh.github.io/pyrsig/



pyrsig User's Guide

Python interface to RSIG Web API

The key value of *pyrsig* is to present RSIG data in pandas DataFrames and xarray Datasets. This makes it easy to do advanced analyses in a pythonic way. Example analyses are provided, but the sky is the limit.

Getting Started

The best way to get started is to install (see below) and then explore the examples gallery.

Installation

pyrsig is avalable through pypi.org, but is still in rapid development. You can get the latest release from pypi via the command below.

```
pip install pyrsig
```

RSIG server prepares the data!

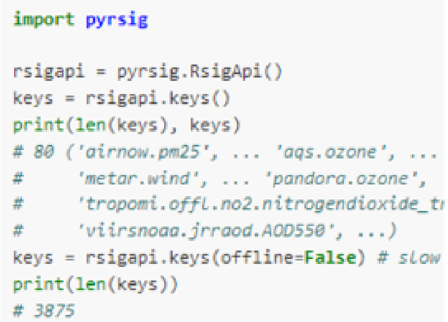
pyrsig has a gallery of examples with code

Get data examples

Download Python source code: get_2_tropomidf.py

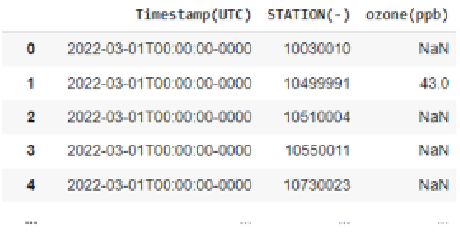
Download Jupyter notebook: get_2_tropomidf.ipynb

Examples showing how to get data.



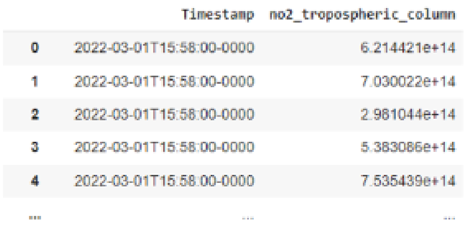
```
import pyrsig

rsigapi = pyrsig.RsigApi()
keys = rsigapi.keys()
print(len(keys), keys)
# 80 ('airnow.pm25', ... 'aqs.ozone', ...
#      'metar.wind', ... 'pandora.ozone',
#      'tropomi.offl.no2.nitrogendioxide_tr
#      'viirsnoaa.jrraod.AOD550', ...)
keys = rsigapi.keys(offline=False) # slow
print(len(keys))
# 3875
```

Get List of Possible Coverages



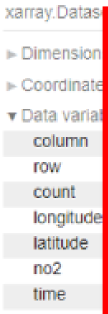
	Timestamp(UTC)	STATION(-)	ozone(ppb)
0	2022-03-01T00:00:00-0000	10030010	NaN
1	2022-03-01T00:00:00-0000	10499991	43.0
2	2022-03-01T00:00:00-0000	10510004	NaN
3	2022-03-01T00:00:00-0000	10550011	NaN
4	2022-03-01T00:00:00-0000	10730023	NaN
...	...	...	...

Get DataFrame for AQS ozone

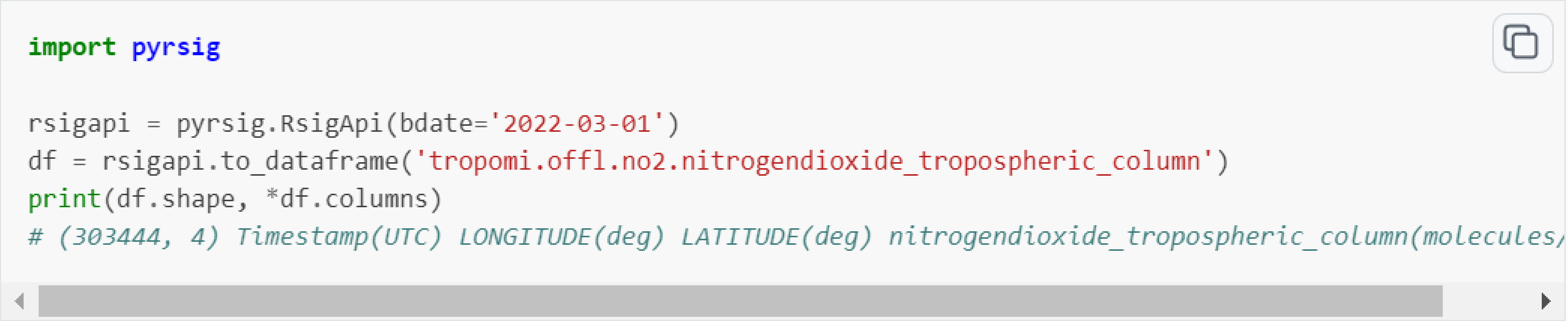


	Timestamp	no2_tropospheric_column
0	2022-03-01T15:58:00-0000	6.214421e+14
1	2022-03-01T15:58:00-0000	7.030022e+14
2	2022-03-01T15:58:00-0000	2.981044e+14
3	2022-03-01T15:58:00-0000	5.383086e+14
4	2022-03-01T15:58:00-0000	7.535439e+14
...	...	...

Get DataFrame for TropOMI NO2



Get COA
NetCDF



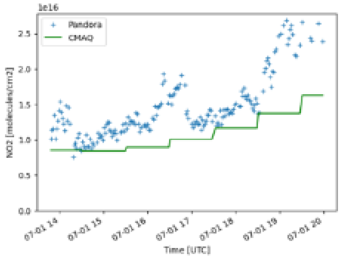
```
import pyrsig

rsigapi = pyrsig.RsigApi(bdate='2022-03-01')
df = rsigapi.to_dataframe('tropomi.offl.no2.nitrogendioxide_tropospheric_column')
print(df.shape, *df.columns)
# (303444, 4) Timestamp(UTC) LONGITUDE(deg) LATITUDE(deg) nitrogendioxide_tropospheric_column(molecules/
```

Timeseries examples

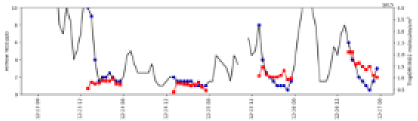
Examples showing timeseries analyses that illustrate the power of pyrsig.

Download Python source code: plot_elpaso.py

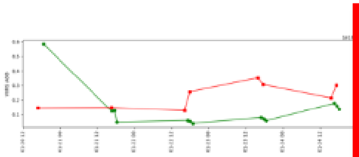
Download Jupyter notebook: plot_elpaso.ipynb



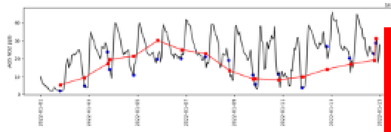
CMAQ vs Pandora



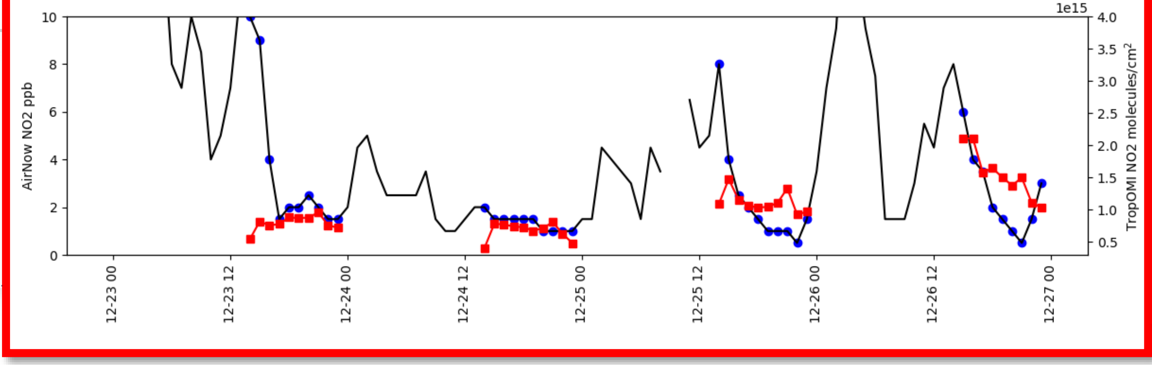
El Paso AirNow vs TEMPO

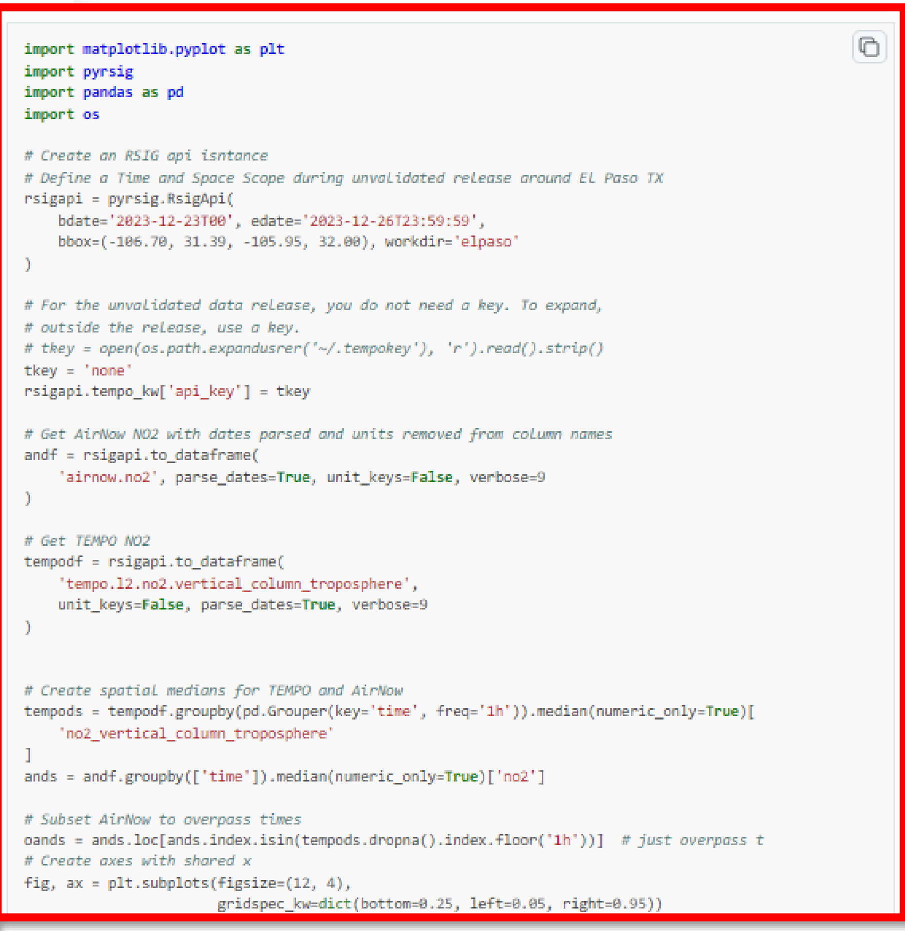


NYC VIIRS AOD vs TropOMI NO2



Phoenix AQS vs TropOMI

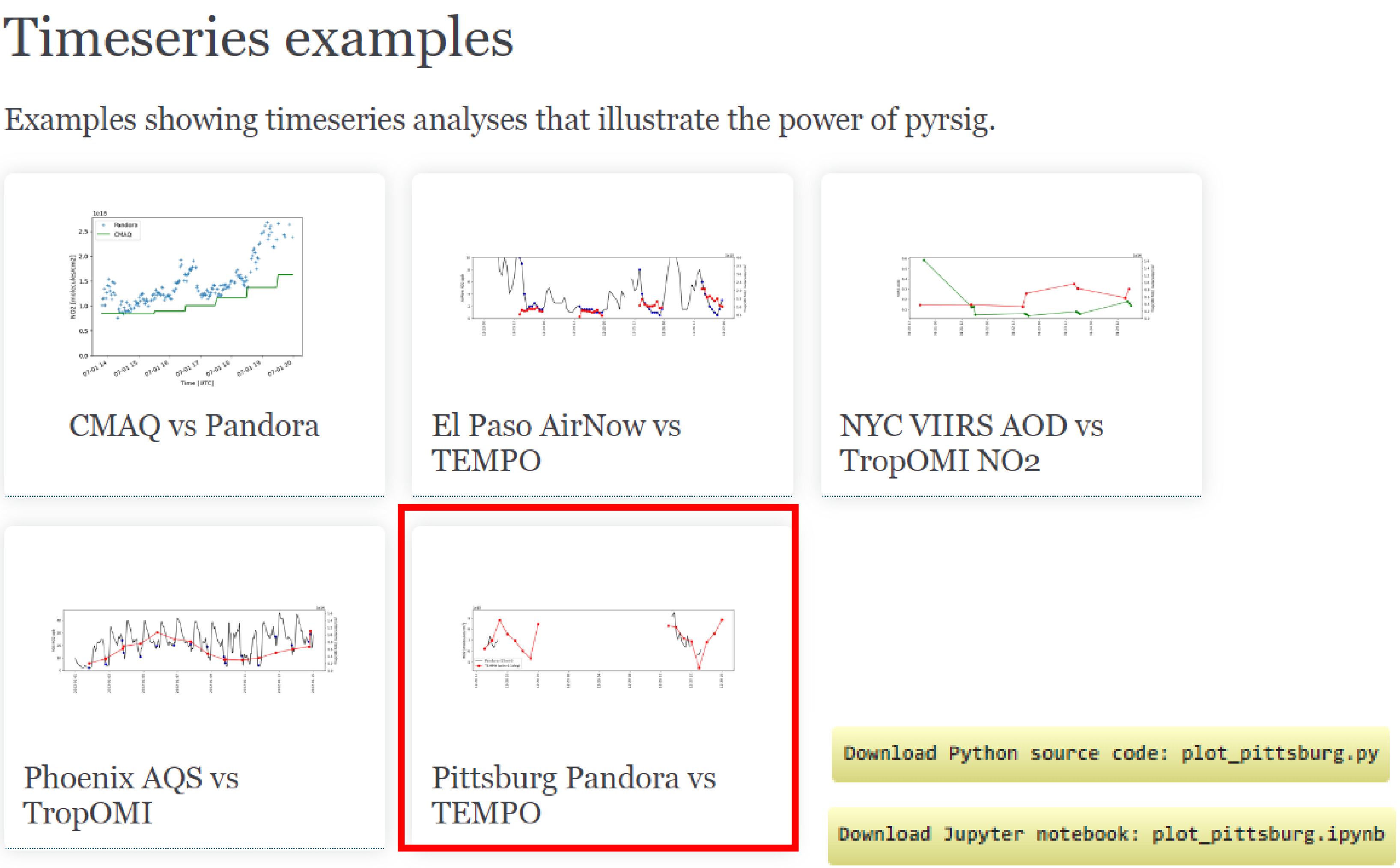




```
import matplotlib.pyplot as plt
import pyrsig
import pandas as pd
import os

# Create an RSIG api isntance
# Define a Time and Space Scope during unvalidated release around El Paso TX
rsigapi = pyrsig.RsigApi(
    bdate='2023-12-23T00', edate='2023-12-26T23:59:59',
    bbox=(-106.70, 31.39, -105.95, 32.00), workdir='elpaso'
)

# For the unvalidated data release, you do not need a key. To expand,
# outside the release, use a key.
# tkey = open(os.path.expandusrer('~/.tempokey'), 'r').read().strip()
tkey = 'none'
rsigapi.tempo_kw['api_key'] = tkey

# Get AirNow NO2 with dates parsed and units removed from column names
andf = rsigapi.to_dataframe(
    'airnow.no2', parse_dates=True, unit_keys=False, verbose=9
)

# Get TEMPO NO2
tempodf = rsigapi.to_dataframe(
    'tempo.l2.no2.vertical_column_troposphere',
    unit_keys=False, parse_dates=True, verbose=9
)


# Create spatial medians for TEMPO and AirNow
tempods = tempodf.groupby(pd.Grouper(key='time', freq='1h')).median(numeric_only=True)[
    'no2_vertical_column_troposphere'
]
ands = andf.groupby(['time']).median(numeric_only=True)['no2']

# Subset AirNow to overpass times
oands = ands.loc[ands.index.isin(tempods.dropna().index.floor('1h'))]  # just overpass t
# Create axes with shared x
fig, ax = plt.subplots(figsize=(12, 4),
                       gridspec_kw=dict(bottom=0.25, left=0.05, right=0.95))
```

Example Gallery
https://barronh.github.io/pyrsig

pyrsig has a gallery of examples with code



Timeseries examples

Examples showing timeseries analyses that illustrate the power of pyrsig.

CMAQ vs Pandora

El Paso AirNow vs TEMPO

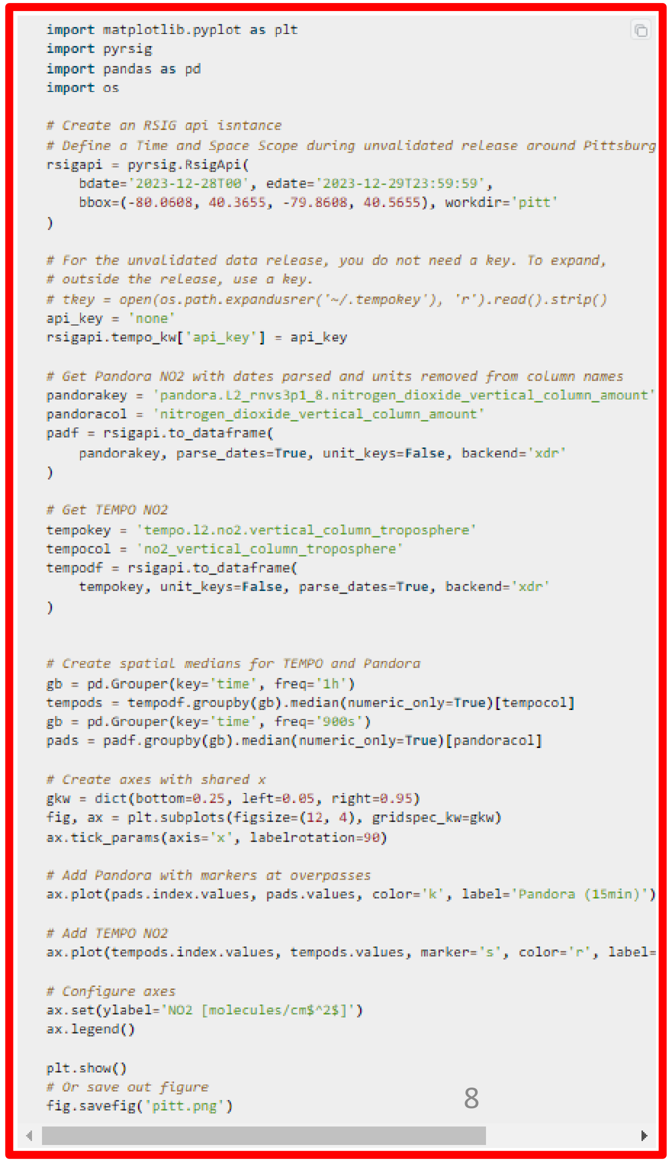
NYC VIIRS AOD vs TropOMI NO2

Phoenix AQS vs TropOMI

Pittsburg Pandora vs TEMPO

Download Python source code: plot_pittsburg.py

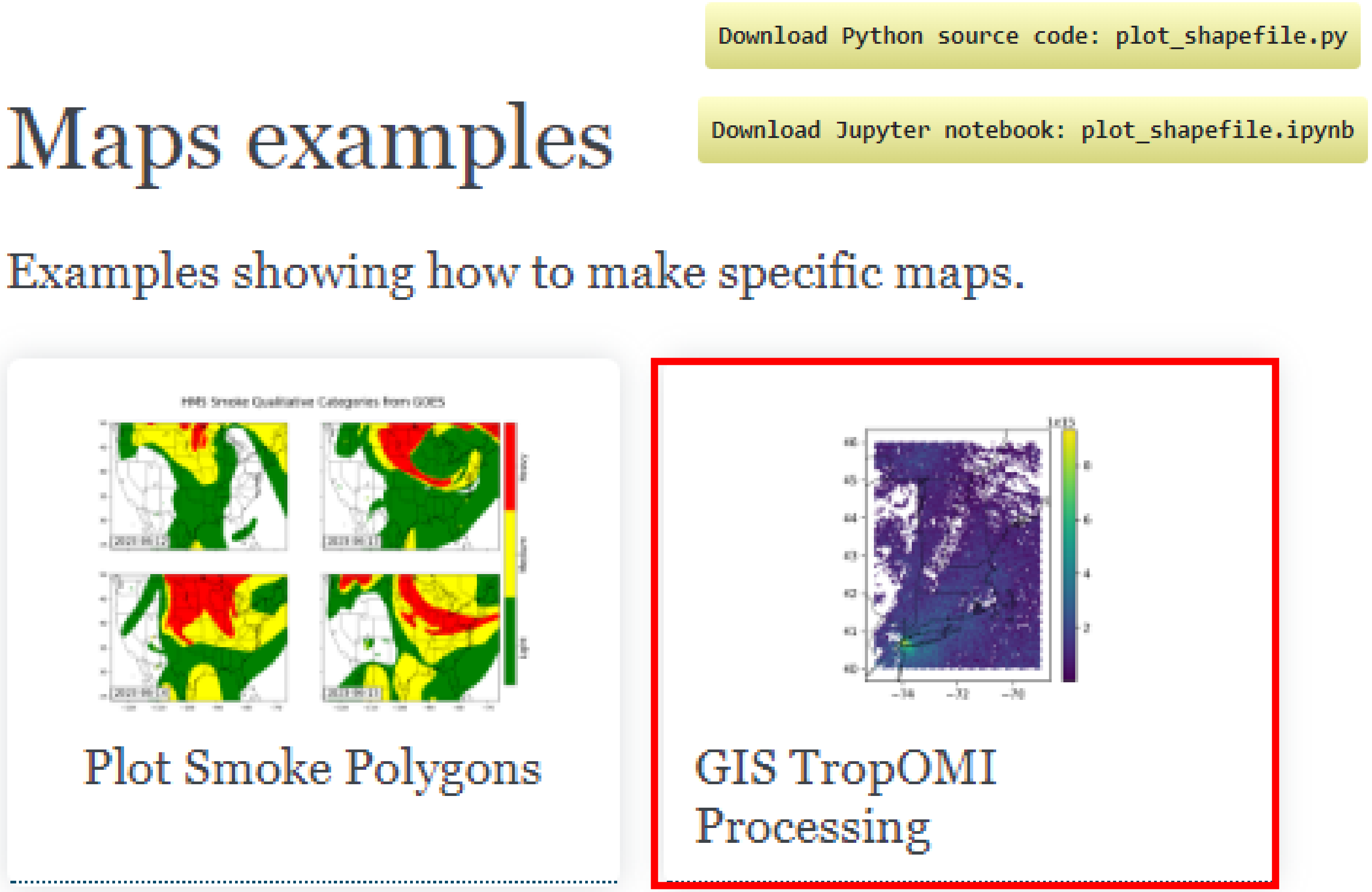
Download Jupyter notebook: plot_pittsburg.ipynb



```
import matplotlib.pyplot as plt
import pyrsig
import pandas as pd
import os

# Create an RSIG api isntance
# Define a Time and Space Scope during unvalidated release around Pittsburg
rsigapi = pyrsig.RsigApi(
    bdate='2023-12-28T00', edate='2023-12-29T23:59:59',
    bbox=(-80.0608, 40.3655, -79.8608, 40.5655), workdir='pitt'
)

# For the unvalidated data release, you do not need a key. To expand,
# outside the release, use a key.
# tkey = open(os.path.expandusrer('~/.tempokey'), 'r').read().strip()
api_key = 'none'
rsigapi.tempo_kw['api_key'] = api_key

# Get Pandora NO2 with dates parsed and units removed from column names
pandorakey = 'pandora.L2_rnvs3p1_8.nitrogen_dioxide_vertical_column_amount'
pandoracol = 'nitrogen_dioxide_vertical_column_amount'
padf = rsigapi.to_dataframe(
    pandorakey, parse_dates=True, unit_keys=False, backend='xdr'
)

# Get TEMPO NO2
tempokey = 'tempo.l2.no2.vertical_column_troposphere'
tempocol = 'no2_vertical_column_troposphere'
tempodf = rsigapi.to_dataframe(
    tempokey, unit_keys=False, parse_dates=True, backend='xdr'
)


# Create spatial medians for TEMPO and Pandora
gb = pd.Grouper(key='time', freq='1h')
tempods = tempodf.groupby(gb).median(numeric_only=True)[tempocol]
gb = pd.Grouper(key='time', freq='900s')
pads = padf.groupby(gb).median(numeric_only=True)[pandoracol]

# Create axes with shared x
gkw = dict(bottom=0.25, left=0.05, right=0.95)
fig, ax = plt.subplots(figsize=(12, 4), gridspec_kw=gkw)
ax.tick_params(axis='x', labelrotation=90)

# Add Pandora with markers at overpasses
ax.plot(pads.index.values, pads.values, color='k', label='Pandora (15min)')

# Add TEMPO NO2
ax.plot(tempods.index.values, tempods.values, marker='s', color='r', label=

# Configure axes
ax.set(ylabel='NO2 [molecules/cm$^2$]')
ax.legend()

plt.show()
# Or save out figure
fig.savefig('pitt.png')
```

Example Gallery
https://barronh.github.io/pyrsig

pyrsig connects satellite data to GIS



Download Python source code: plot_shapefile.py

Download Jupyter notebook: plot_shapefile.ipynb

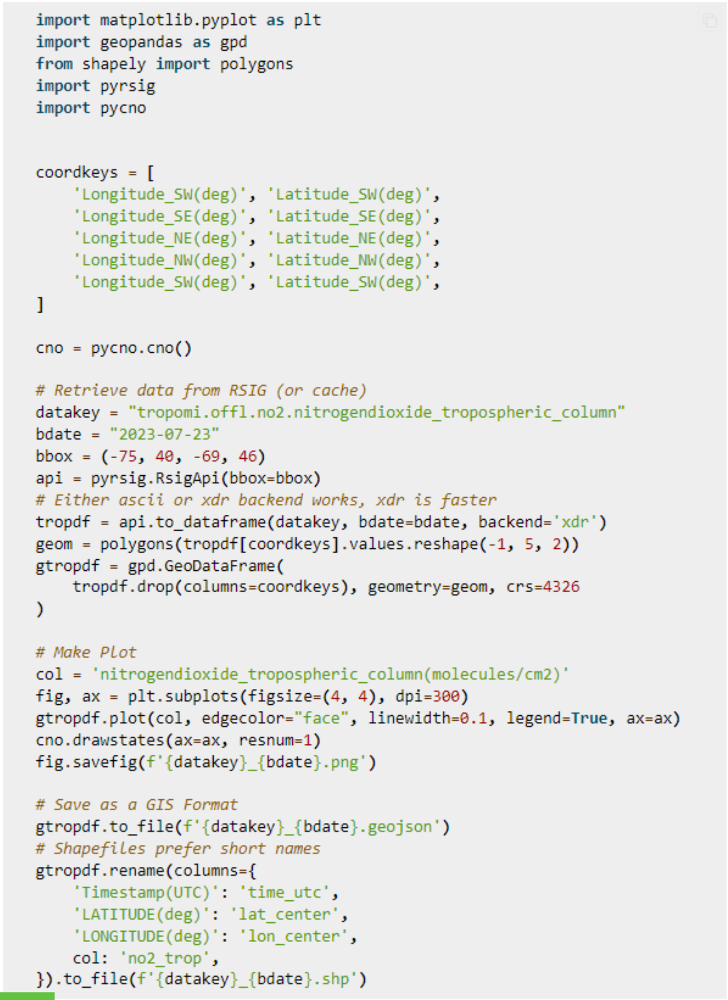
Maps examples

Examples showing how to make specific maps.

Plot Smoke Polygons

GIS TropOMI Processing

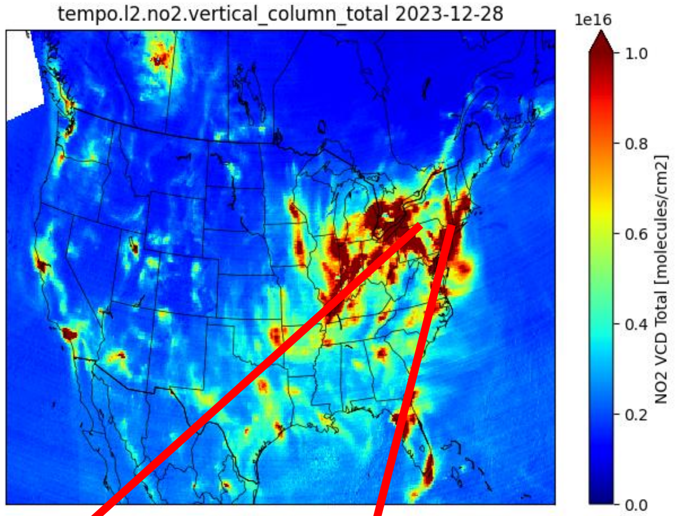
Convert GIS-Compatible Vector Files

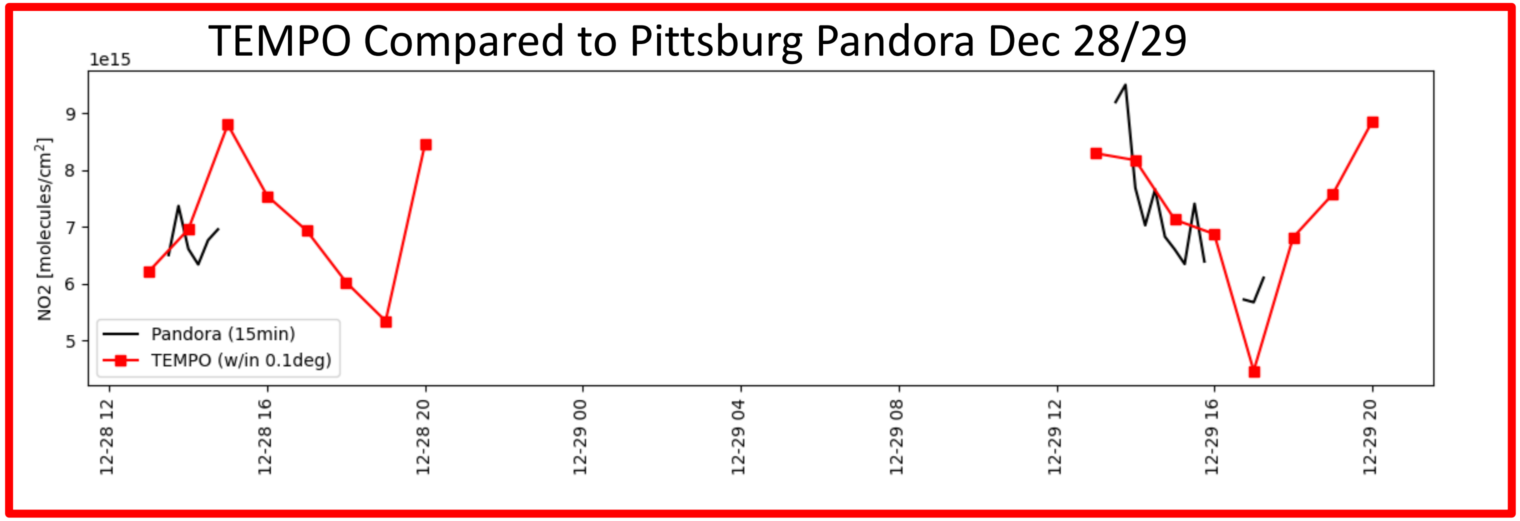


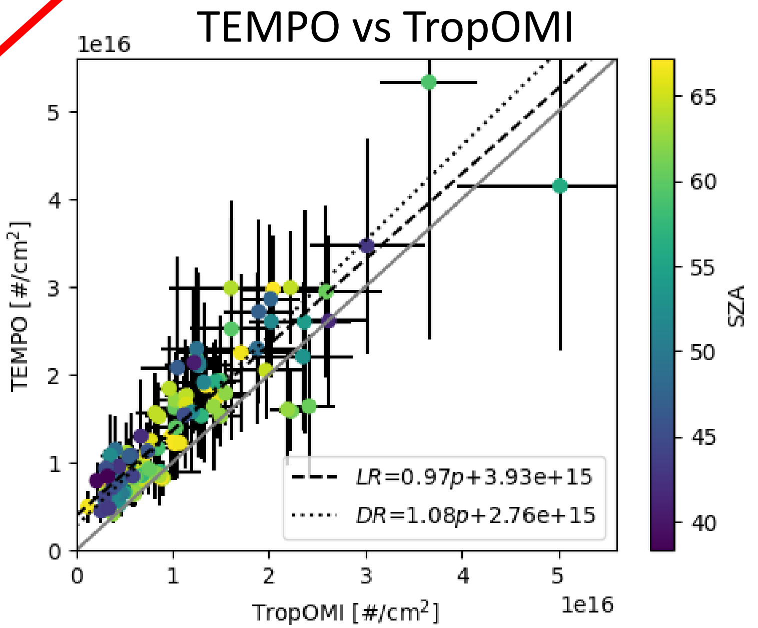
```python
import matplotlib.pyplot as plt
import geopandas as gpd
from shapely import polygons
import pyrsig
import pycno


coordkeys = [
    'Longitude_SW(deg)', 'Latitude_SW(deg)',
    'Longitude_SE(deg)', 'Latitude_SE(deg)',
    'Longitude_NE(deg)', 'Latitude_NE(deg)',
    'Longitude_NW(deg)', 'Latitude_NW(deg)',
    'Longitude_SW(deg)', 'Latitude_SW(deg)',
]

cno = pycno.cno()

# Retrieve data from RSIG (or cache)
datakey = "tropomi.offl.no2.nitrogendioxide_tropospheric_column"
bdate = "2023-07-23"
bbox = (-75, 40, -69, 46)
api = pyrsig.RsigApi(bbox=bbox)
# Either ascii or xdr backend works, xdr is faster
tropdf = api.to_dataframe(datakey, bdate=bdate, backend='xdr')
geom = polygons(tropdf[coordkeys].values.reshape(-1, 5, 2))
gtropdf = gpd.GeoDataFrame(
    tropdf.drop(columns=coordkeys), geometry=geom, crs=4326
)

# Make Plot
col = 'nitrogendioxide_tropospheric_column(molecules/cm2)'
fig, ax = plt.subplots(figsize=(4, 4), dpi=300)
gtropdf.plot(col, edgecolor="face", linewidth=0.1, legend=True, ax=ax)
cno.drawstates(ax=ax, resnum=1)
fig.savefig(f'{datakey}_{bdate}.png')

# Save as a GIS Format
gtropdf.to_file(f'{datakey}_{bdate}.geojson')
# Shapefiles prefer short names
gtropdf.rename(columns={
    'Timestamp(UTC)': 'time_utc',
    'LATITUDE(deg)': 'lat_center',
    'LONGITUDE(deg)': 'lon_center',
    col: 'no2_trop',
}).to_file(f'{datakey}_{bdate}.shp')
```

pyrsig - Tropospheric Emissions: Monitoring of Pollution (TEMPO)

- pyrsig connects you to the same data ***and*** allows you to perform custom analyses.
- Map uses TEMPO data that is already public (Dec 1-29)
- Scatterplot for long-term comparison between TEMPO and TropOMI.
- Time series on the bottom-right shows comparison to a Pandora ground-based remote sensor to evaluate TEMPO.


tempo.l2.no2.vertical_column_total 2023-12-28
1e16
1.0
0.8
0.6
0.4
0.2
0.0
NO2 VCD Total [molecules/cm2]


TEMPO Compared to Pittsburg Pandora Dec 28/29


TEMPO vs TropOMI

TEMPO Tutorial

- https://gaftp.epa.gov/Air/aqmg/bhenders/tutorials.html
  - Click on “TEMPO pyrsig training notebook (May 2024 meeting)
  - The link takes you to a github gist.
- To run on Google Colab, click the “Open in Colab” badge
  - Click play on each cell
  - Google will warn you that they didn’t make this… answer yes.
  - The first code cell will tell you to restart.
  - Choose “runtime”, then “restart runtime”.
  - Then run the rest of the cells.

We’ll do this together

Questions?

pyrsig: henderson.barron@epa.gov

RSIG: rsig@epa.gov