Practical Application of Python for Air Quality Data Analysis and Modeling

Byeong-Uk Kim[1,*], Barron Henderson[2], Marcus Trail[1], Di Tian[1], and James Boylan[1]
[1]Georgia Environmental Protection Division, Atlanta, GA, USA, [2]Office of Air Quality Planning and Standards, US EPA, RTP, NC, USA, *Byeong.Kim@dnr.ga.gov

GEORGIA DEPARTMENT OF NATURAL RESOURCES
ENVIRONMENTAL PROTECTION DIVISION

Goal

- ❑ The goal of this presentation is to illustrate how air quality scientists and engineers can utilize Python for their daily air quality modeling and data analysis tasks.

Why Python?

- ❑ Now, Python is everywhere. Check out TIOBE index: "https://www.tiobe.com/tiobe-index/".
- ❑ Python has very diverse software ecosystem; scientific community is one of active groups. Many free and community-supported libraries are available for high-performance computing, publication quality graphics, and large-scale data analyses.
- ❑ Most libraries are cross-platform; they work for Windows, Linux, and OSX. To highlight this, examples in this presentation were made on a Windows Machine.

Air Quality Data Analysis

- ❑ Getting a pre-generated AQS data file, extracting Georgia data, saving it as a CSV file, and plotting a histogram

```
import zipfile, requests, os
import numpy as np
import pandas as pd

#https://aqs.epa.gov/aqsweb/airdata/hourly_44201_2002.zip
file_key = 'hourly_44201_2002'
aqs_file_url =
'https://aqs.epa.gov/aqsweb/airdata/{0}.zip'.format(file_key)
if not os.path.exists('{0}.zip'.format(file_key)):
    print("You don't have the specified zip file. I am downloading
the file.")
    requested = requests.get(aqs_file_url, verify=False)
    open('{0}.zip'.format(file_key), 'wb').write(requested.content)
    zf = zipfile.ZipFile('{0}.zip'.format(file_key)).extractall()
df1 = pd.read_csv("{0}.csv".format(file_key), dtype=object)
ofile_cols = ['State Code', 'County Code', 'Site Num', 'Parameter
Code', 'POC', 'Parameter Name', 'Date Local', 'Time Local', 'Sample
Measurement']
out_df = df1.loc[df1["State Code"]=="13",ofile_cols]
out_df.to_csv("reformatted_ga_{0}.csv".format(file_key), index=False)
out_df['o3'] = out_df['Sample Measurement'].astype(np.float64) # to
make numeric dataset

out_df['o3'].hist(bins=10).get_figure().savefig("hist.png", dpi=600)

out_df.pivot_table(values="o3", index=["County Code"],
aggfunc=max).plot(kind="bar").get_figure().savefig("bar.png",
dpi=600)
```

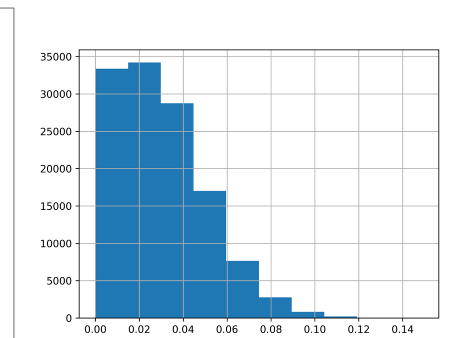


Fig. 1. Histogram of hourly ozone concentrations in 2002 at all Georgia monitors

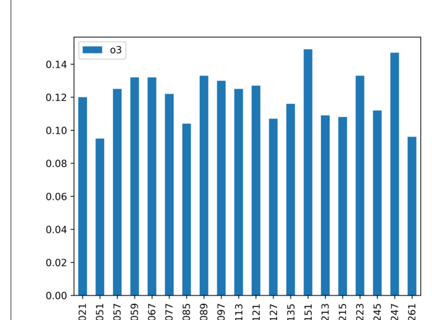


Fig. 2. Annual maximum 1-hour ozone concentrations in Georgia counties during 2002

Air Quality Modeling Analysis 1

- ❑ Reading a native CAMx file, adding a derived variable, and saving the file as an IOAPI compliant NETCDF file

```
import PseudoNetCDF as pnc

pncf =
pnc.pncopen(r"CAMx.v6.50.midwest.36.12.noMPI.2002060
3.avrg.grd01").copy()

# Computing a derived variable
pncf.eval("NOX=NO+NO2", inplace=True)

# To make outfile VERDI-compatible...
del pncf.Conventions
del pncf.variables['x']
del pncf.variables['y']
del pncf.variables['latitude']
del pncf.variables['longitude']
pncf.save("test_out.nc")
```

PseudoNetCDF (based on python-netCDF4 library) can read/write various air quality modeling input/output files (and more). Consider visiting the "PseudoNetCDF v3" poster booth.

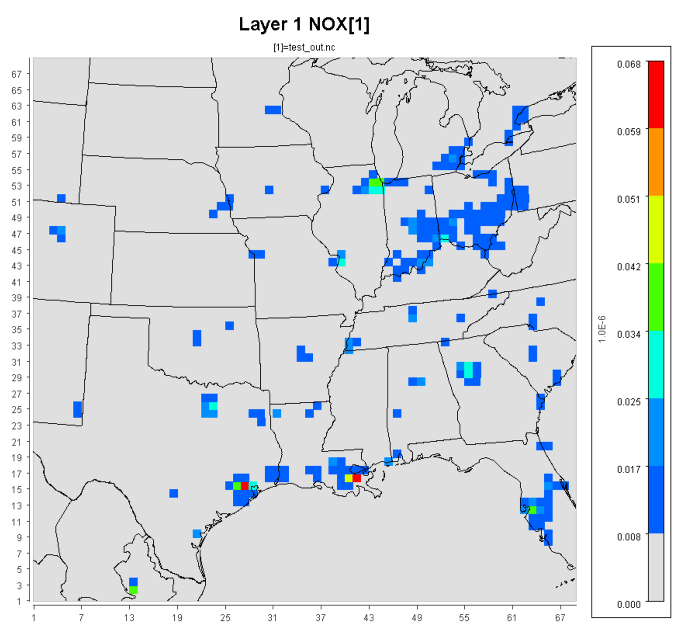


Fig. 3. NOx concentrations in an IOAPI compliant netCDF file converted from a native CAMx output file by PseudoNetCDF (visualized with VERDI)

Air Quality Modeling Analysis 2

- ❑ Extracting hourly data at monitor location from an IOAPI compliant NETCDF file, saving them as a CSV or an Excel file, and comparing with observed data

```
from datetime import timedelta, datetime
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import PseudoNetCDF as pnc

pncf = pnc.pncopen("test_out.nc")

#AIRS,LAT,LON,Name
atl_o3_mon_df = pd.read_csv("ga_o3_monitors.csv"
lon  = atl_o3_mon_df["LON"].tolist()
lat  = atl_o3_mon_df["LAT"].tolist()
i, j = pncf.ll2ij(lon, lat)

#    i, j : indices (0-based) for variables
pnc_at_mon_loc = pncf.sliceDimensions(ROW=j, COL=i)
o3_at_mon      = pnc_at_mon_loc.variables["O3"].array().astype(np.float64)
o3_at_mon_df          = pd.DataFrame(o3_at_mon[:,0,:])
o3_at_mon_df.columns = atl_o3_mon_df["Name"].values
o3_at_mon_df["utc"]  = pncf.getTimes()
o3_at_mon_df["utc"]  = o3_at_mon_df["utc"].apply(lambda x : x.replace(tzinfo=None))
o3_at_mon_df["est"]  = o3_at_mon_df["utc"] + timedelta(hours=-5)
o3_at_mon_df.to_csv("mod_extract_at_mon.csv", index=False)
o3_at_mon_df.to_excel("mod_extract_at_mon.xlsx", index=False)

#Reading monitored O3 data
ga_o3_data_df = pd.read_csv("reformatted_ga_hourly_44201_2002.csv", dtype=object)
ga_o3_data_df['Sample Measurement'] = ga_o3_data_df['Sample Measurement'].astype(np.float64)
ga_o3_data_df["AIRS"]= ga_o3_data_df["State Code"]+ga_o3_data_df["County
Code"]+ga_o3_data_df["Site Num"]
#AISID for Confederate Ave. : 131210055
conf_ave_o3_df = ga_o3_data_df.loc[ga_o3_data_df["AIRS"]=="131210055",:]
conf_ave_o3_df["datetime_str"] = conf_ave_o3_df["Date Local"]+conf_ave_o3_df["Time Local"]
conf_ave_o3_df["est"] = conf_ave_o3_df["datetime_str"].apply(lambda x : datetime.strptime(x, "%Y-
%m-%d%H:%M"))

mod_df = o3_at_mon_df[["est", "Confederate Ave."]].rename(columns={"Confederate Ave." : "MODEL"})
obs_df = conf_ave_o3_df[["est", "Sample Measurement"]].rename(columns={"Sample Measurement" :
"OBSERVATION"})
mod_obs_df = pd.merge(mod_df, obs_df, on="est", how="left")

mod_obs_df.plot(x="est", y=["OBSERVATION", "MODEL"], style=[".","-"], figsize=(8,6), xlim
=[mod_obs_df["est"][mod_obs_df.index[0]], mod_obs_df["est"][mod_obs_df.index[-1]]])
plt.savefig('ts.png', dpi=600)

mod_obs_df.plot(x=["OBSERVATION"],y=["MODEL"],kind="scatter")
plt.savefig('scatter.png', dpi=600)
```

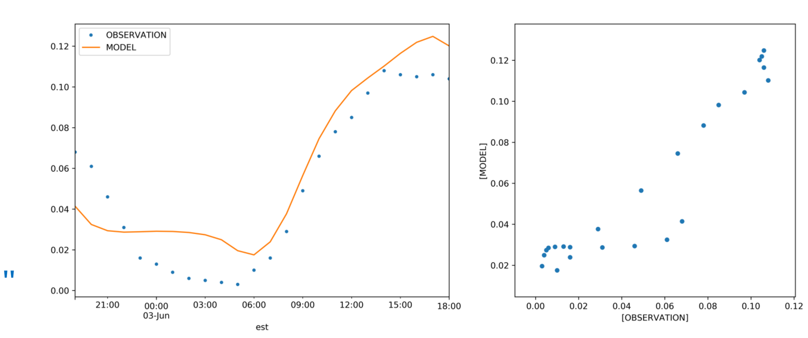


Fig. 4. Example time series (left) and scatter plots (right) comparing model results and observation using PseudoNetCDF and Pandas

Emission Analysis

- ❑ Using PANDAS to compute statistics, make pivot tables, and perform "outer" join!

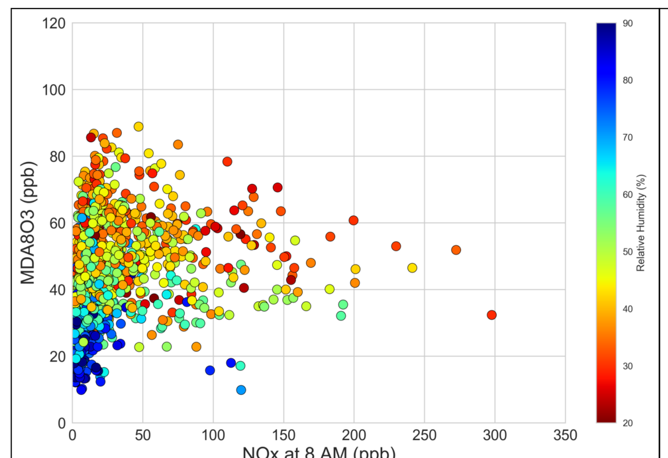
```
...
s = st_cnty_df0.sum(axis=0).reset_index()
...
temp_df3 = pd.pivot_table(temp_df2, values="TotalEmissions",
index=["StateAndCountyFIPSCode","SourceClassificationCode"],columns="PollutantCode")
...
df4 = pd.merge(df3, np_pt_reconciliation_df, how="outer", suffixes=('_1', '_2'), on=u'SCC')
df4.to_excel("final_np_sccs_cap_only_pnt_sub_summary_df.xlsx", index=False)
...
```

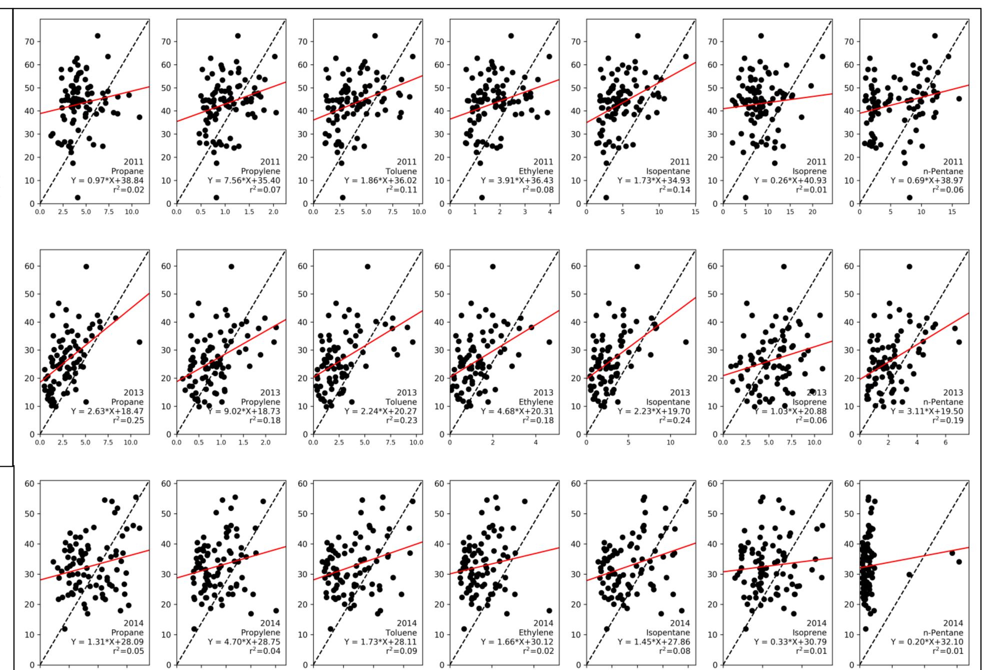
- ❑ Developing, documenting, and performing QAs for emission inventories with Python

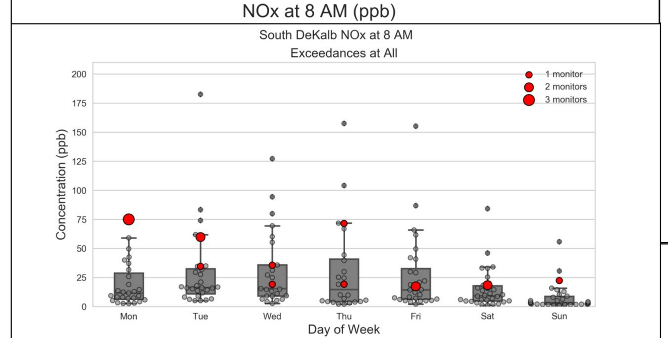
```
...
### Excluding of Ag Burn (2801500000) and Land Clearing (2610000500) since Tao submitted them
tblEmissions.loc[tblEmissions["SourceClassificationCode"]==2801500000,"TotalEmissions"] = pd.np.nan
tblEmissions.loc[tblEmissions["SourceClassificationCode"]==2610000500,"TotalEmissions"] = pd.np.nan

### Revising data based on the feedback report and appendix_3_scc_code_retirements_updated.xlsx
# 2461800000 is an old code mapped to 2461850000 or  2460800000, Solvent - Consumer & Commercial
Solvent Use
# Pesticide Application: All Processes
# need to remove because EPA estimates will be used
tblEmissions.loc[tblEmissions["SourceClassificationCode"]==2461800000,"TotalEmissions"] = pd.np.nan

# Mapping to a new SCC
tblEmissions.loc[tblEmissions["SourceClassificationCode"]==2805001000,"SourceClassificationCode"] =
2805001100

# Mapping to a new SCC
tblEmissions.loc[tblEmissions["SourceClassificationCode"]==2501070000,"SourceClassificationCode"] =
2501070100
...
```

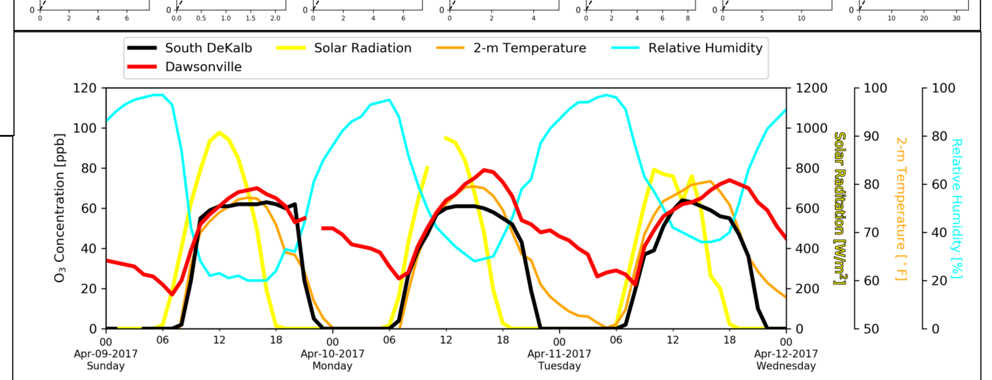
Illustrative Examples

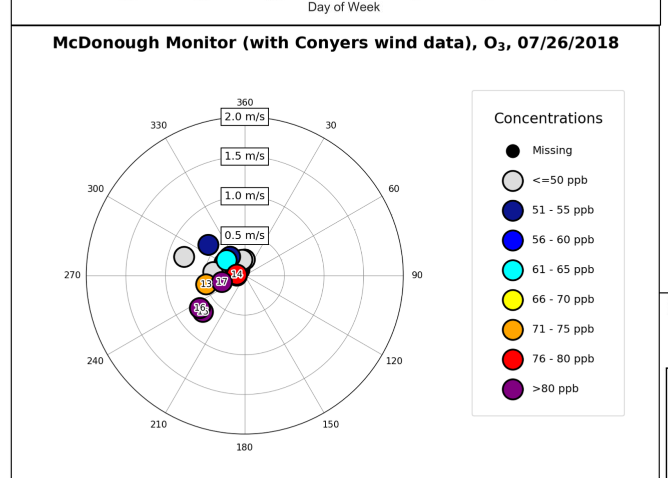
- ❑ Code examples are not *optimized* because the purpose was to introduce various common operations I used in my daily tasks.
- ❑ Example plots below were generated with a set of frequently used libraries on both Linux and Windows operating computers.
  - Python Distribution: Anaconda
  - Python Standard Libraries: os, sys, and datetime
  - Data Analysis: Pandas
  - Data Visualization: Matplotlib (with Basemap and Cartopy) and Seaborn (which is baesd on Matplotlib)
  - Modeling Data Read/Write: PseudoNetCDF (based on NetCDF4 and Xarray)
  - PYSPLIT

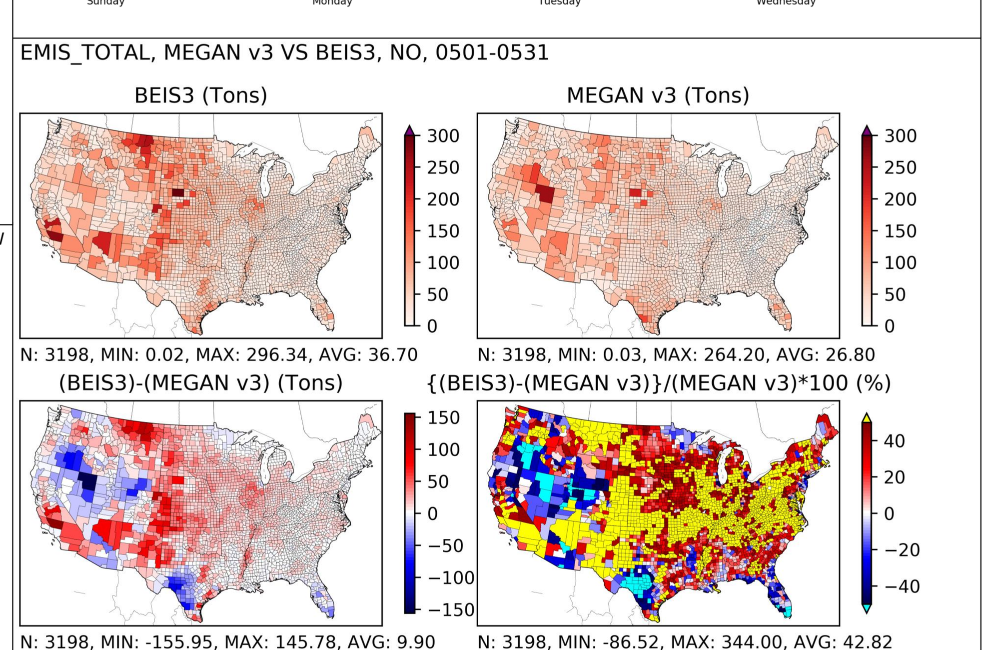


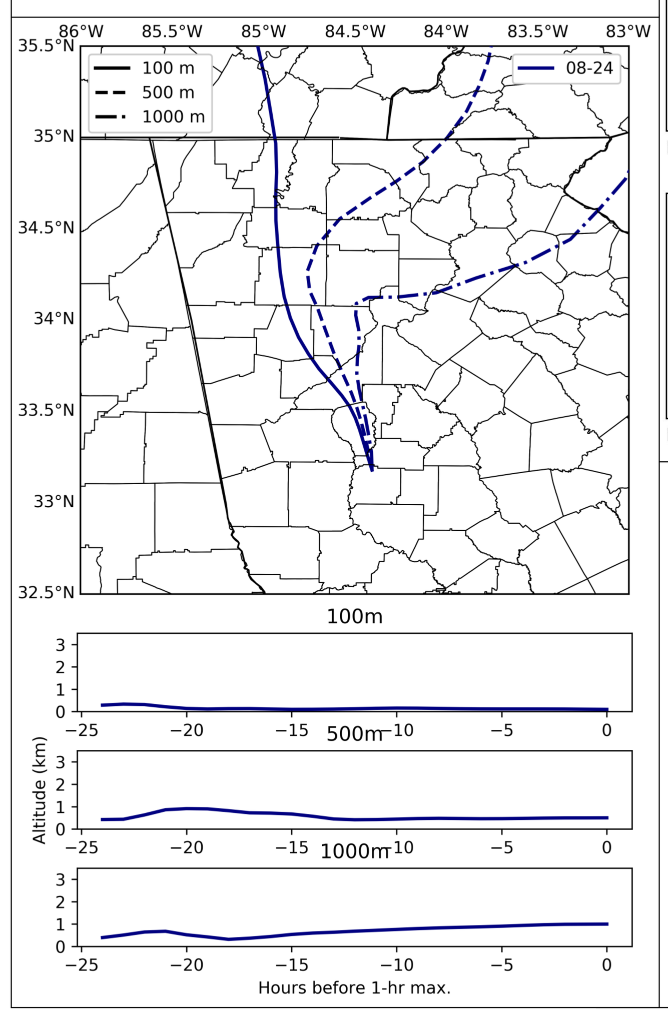




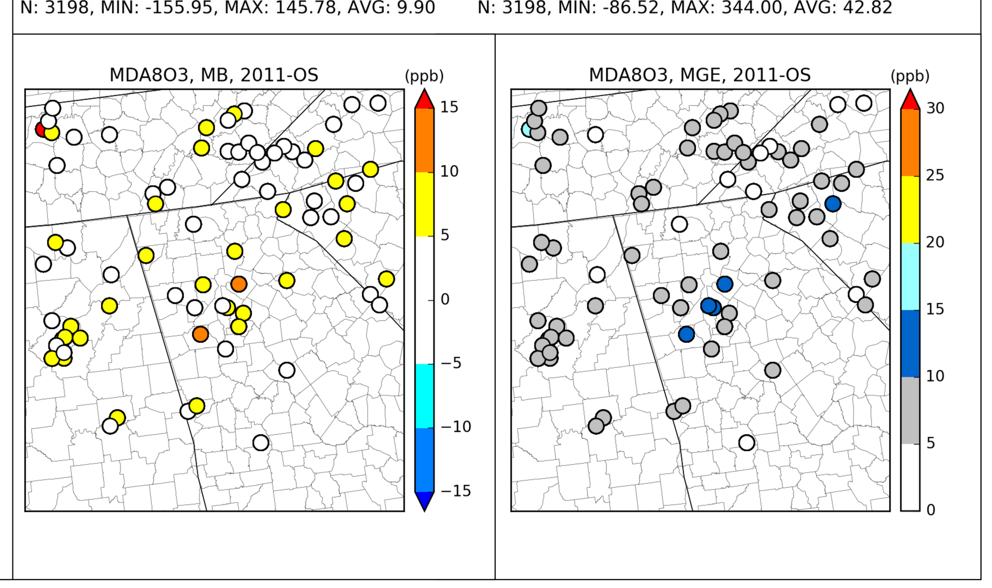








pySPLIT: https://github.com/mscross/pysplit



Ready for More? Search names of libraries in Google!
You can find tons of examples.