FlamMap pre-processing tool for wind and initial fuel moisture data (Python)

Author: Dennis W. Hallema (mailto:dwhallem@ncsu.edu)

Description: Tool for preparing wind (.wnd) and initial fuel moisture (.fms) input files for the FlamMap (https://www.firelab.org/project/flammap) fire analysis application.

Date created: 4/7/2020

Depends: See `environment.yml` .

Data:

- FireFamily Plus (https://www.firelab.org/project/firefamilyplus) Fire Risk Export file for FlamMap

In [1]:

```
# Import modules
import numpy as np
import pandas as pd
import os

# Set variables
filepaths = ['data/PY_' + str(x+1).zfill(3) + ".txt" for x in range(127 + 1)]
resultpath = 'results/'

for i, filepath in enumerate(filepaths):
#     i =1
#     filepath = filepaths[i]
    print(filepath)

    # Import file
    with open(filepath, 'r') as file:
        lines = file.readlines()

    pyrome_id = i + 1
    days = int(lines[3].split()[0])

    ## Prepare initial fuel moisture files

    # Extract initial fuel moisture percentile data
    colnames = lines[days + 5].split()
    fms = pd.DataFrame([l.split() for l in lines[days+6:days+106]], columns = col
    fms = fms.apply(pd.to_numeric)

    # Create result objects
    fuel_type = pd.DataFrame(range(1,257), columns = ["Fuel type"])

    # Fuel moisture 80th percentile
    col4 = pd.DataFrame([90]*256)
    col5 = pd.DataFrame([110]*256)
    values = fms.loc[[80-1],["FM1", "FM10", "FM100"]]
    fmsi = pd.concat([values]*256).reset_index(drop=True)
    fms80 = pd.concat([fuel_type, fmsi, col4, col5], axis=1)
    fms80.columns = ["Fuel type", "80th percentile pyrome", "","","",""]
    fms80.to_csv(resultpath + os.path.split(filepath)[1] + "_fms80.fms", header=T

    # Fuel moisture 90th percentile
    col4 = pd.DataFrame([60]*256)
    col5 = pd.DataFrame([80]*256)
    values = fms.loc[[90-1],["FM1", "FM10", "FM100"]]
    fmsi = pd.concat([values]*256).reset_index(drop=True)
    fms90 = pd.concat([fuel_type, fmsi, col4, col5], axis=1)
    fms90.columns = ["Fuel type", "90th percentile pyrome", "","","",""]
    fms90.to_csv(resultpath + os.path.split(filepath)[1] + "_fms90.fms", header=T

    # Fuel moisture 97th percentile
    col4 = pd.DataFrame([40]*256)
    col5 = pd.DataFrame([60]*256)
    values = fms.loc[[97-1],["FM1", "FM10", "FM100"]]
    fmsi = pd.concat([values]*256).reset_index(drop=True)
    fms97 = pd.concat([fuel_type, fmsi, col4, col5], axis=1)
    fms97.columns = ["Fuel type", "97th percentile pyrome", "","","",""]
```

```
    fms97.to_csv(resultpath + os.path.split(filepath)[1] + "_fms97.fms", header=1

    ## Prepare wind files

    # Get energy release component (ERC) data
    erc = pd.DataFrame([l.split() for l in lines[4:4+days]])
    colnames = ['erc_avg', 'erc_stdev', 'erc_curr', 'date'] + ['erc_yr' + str(x +
    erc.columns = colnames[0:len(erc.columns)]
    erc["erc_avg"] = pd.to_numeric(erc["erc_avg"])
    erc["date"] = pd.to_datetime(erc["date"], format = '%m/%d/%Y')

    # Aggregate ERC by month, find month with highest ERC
    erc["month"] = erc["date"].dt.month
    erc_m = erc.resample('m', on='date').mean()
    erc_m.sort_values(by=['erc_avg'], inplace=True, ascending=False)
    ercmax = erc_m.loc[erc_m.index[0], 'erc_avg']
    ercmax_month = erc_m.loc[erc_m.index[0], 'month']

    # Get wind percentile distribution data for month with highest ERC
    colnames = lines[days + 117].split()
    k = days + 118 + 9 * (ercmax_month - 1)
    wnd = pd.DataFrame([lines[l].split() for l in range(k,k+6)], columns = colnam
    wnd.set_index(['speed'], inplace=True)

    # Get predominant wind direction by wind speed for month with highest ERC
    wdir = wnd.loc[:, wnd.columns != 'speed'].idxmax(axis=1)[:]
    wdirpc = wnd.loc[:, wnd.columns != 'speed'].max(axis=1)[:]
    colnames = ['wdir5','wdir10','wdir15','wdir20','wdir25','wdir30',
                'wdirpc5','wdirpc10','wdirpc15','wdirpc20','wdirpc25','wdirpc30']
    wdir = pd.DataFrame(pd.concat([wdir, wdirpc], axis=0)).set_index([colnames]).

    # Get overall predominant wind direction and speed for month with highest ERC
    wmod_dir = [c for c in wnd.columns if any(wnd[c] == wnd.values.max())][0]
    wnd_t = wnd.transpose()
    wmod_spd = [c for c in wnd_t.columns if any(wnd_t[c] == wnd_t.values.max())][

    # Create result object
    pyrome_ids = pd.Series(pyrome_id, name = 'pyrome_id')
    ercmaxs = pd.Series(ercmax, name = 'ercmax')
    ercmax_months = pd.Series(ercmax_month, name = 'ercmax_month')
    wmod_dir = pd.Series(wmod_dir, name = 'wmod_dir')
    wmod_spd = pd.Series(wmod_spd, name = 'wmod_spd')
    wdir_out = pd.concat([pyrome_ids, ercmaxs, ercmax_months, wmod_dir, wmod_spd,
    wdir_out.to_csv(resultpath + os.path.split(filepath)[1] + "_ercmax_wdir.csv",
```

```
data/PY_001.txt
data/PY_002.txt
data/PY_003.txt
data/PY_004.txt
data/PY_005.txt
data/PY_006.txt
data/PY_007.txt
data/PY_008.txt
data/PY_009.txt
data/PY_010.txt
data/PY_011.txt
```

```
data/PY_012.txt
data/PY_013.txt
data/PY_014.txt
data/PY_015.txt
data/PY_016.txt
data/PY_017.txt
data/PY_018.txt
data/PY 019.txt
```