06_analyzing_data_from_multiple_files

August 12, 2021

1 Analyzing Data From Multiple Files

If you look in the data folder, you'll notice that we have dozens of datasets that we plan on evaluating. Going through one by one is possible, but not practical.

If we want to perform the same action on several items, python allows us to do so by using `for` loops.

1.1 Questions

How can I do the same operations on many different files?

1.2 Objectives

1) Use a library function to get a list of filenames that match a wildcard pattern.
2) Write a `for` loop to process multiple files.

1.3 Credit:

things here are a mix of the really excellent Software Carpentry tutorial on Python: http://swcarpentry.github.io/python-novice-inflammation/

I've made some slight adaptations here and there, but the credit goes to those organizations. I hope I am using this correctly under the licences:

https://earth-env-data-science.github.io/LICENSE.html https://swcarpentry.github.io/python-novice-inflammation/LICENSE.html

---

We now have almost everything we need to process all our data files. The only thing that's missing is a library with an unfortunate name: `glob` . lets import `glob`, and `matplotlib` and `numpy` (we are going to use all of them)

[1]:
```python
# import glob, matplotlib.pyplot and numpy
%matplotlib inline
import glob
import matplotlib.pyplot as plt
import numpy as np
```

The `glob` library contains a function, also called `glob`, that finds files and directories whose names match a pattern. We provide those patterns as strings: the character `*` matches zero or more characters, while `?` matches any one character.

The useage looks something like `glob.glob(pattern)`

We can use this to get the names of all the CSV files in the data directory:

[2]:
```
glob.glob('../data/inflammation*.csv')
```

[2]:
```
['../data/inflammation-01.csv',
 '../data/inflammation-02.csv',
 '../data/inflammation-03.csv',
 '../data/inflammation-04.csv',
 '../data/inflammation-05.csv',
 '../data/inflammation-06.csv',
 '../data/inflammation-07.csv',
 '../data/inflammation-08.csv',
 '../data/inflammation-09.csv',
 '../data/inflammation-10.csv',
 '../data/inflammation-11.csv',
 '../data/inflammation-12.csv']
```

As these examples show, `glob.glob`'s result is a list of file and directory paths in arbitrary order. This means we can loop over it to do "something" with each filename in turn. In our case, the "something" we want to do is generate a set of plots for each file in our inflammation dataset. If we want to start by analyzing just the first three files in alphabetical order, we can use a built-in function called `sorted` to generate a new sorted list from the `glob.glob` output:

[3]:
```
sorted(glob.glob('../data/inflammation*.csv'))
```

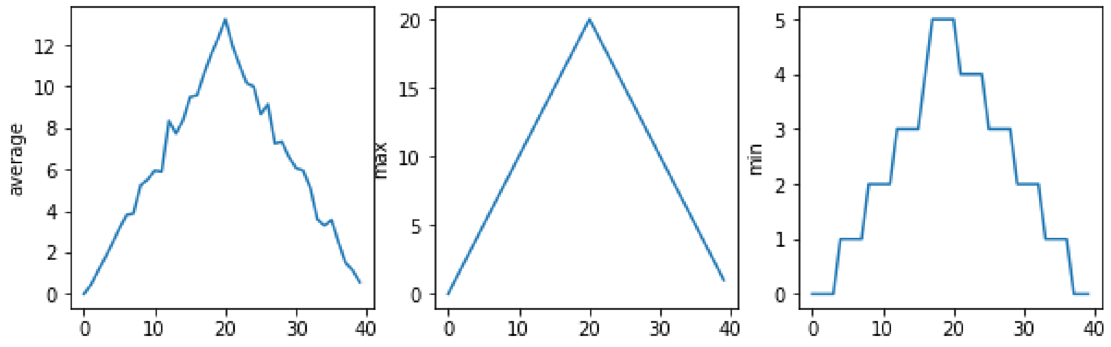
[3]:
```
['../data/inflammation-01.csv',
 '../data/inflammation-02.csv',
 '../data/inflammation-03.csv',
 '../data/inflammation-04.csv',
 '../data/inflammation-05.csv',
 '../data/inflammation-06.csv',
 '../data/inflammation-07.csv',
 '../data/inflammation-08.csv',
 '../data/inflammation-09.csv',
 '../data/inflammation-10.csv',
 '../data/inflammation-11.csv',
 '../data/inflammation-12.csv']
```

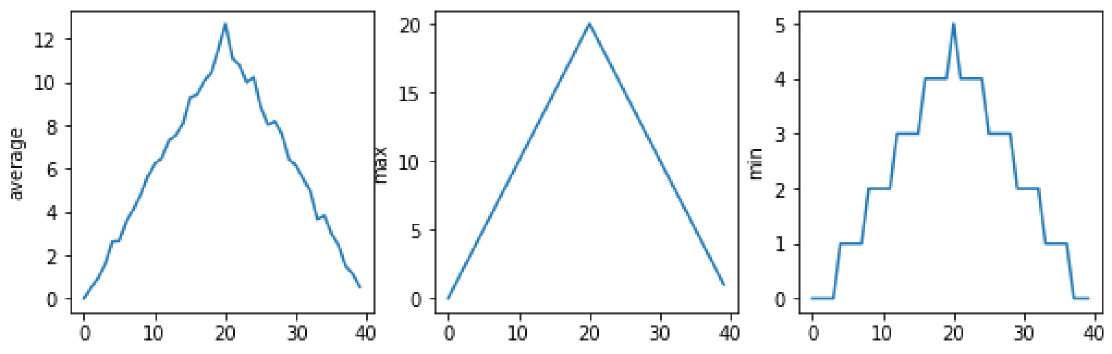
Now, lets loop through this list of files, and plot some data from each one.

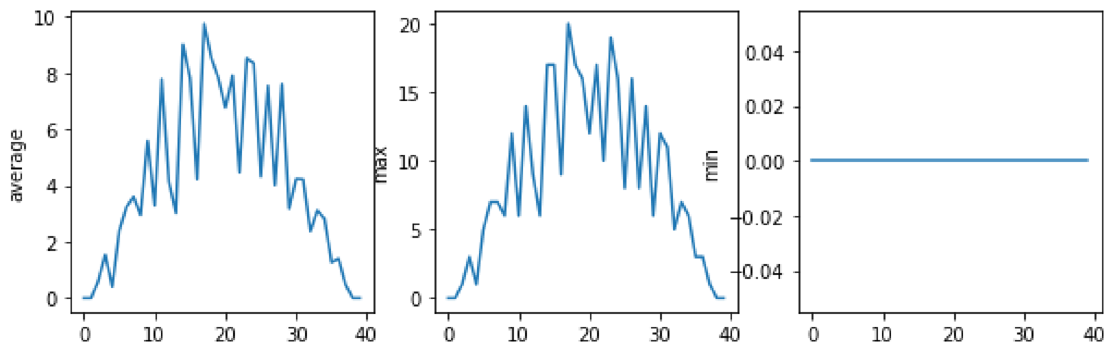
Let's make the same statistics plots we made before, but for the first three data files. This is a pretty long but of code, so bear with me. Remember to think about the structure of loops

[4]:
```python
# use glob to get the list of filenames:

filenames = sorted(glob.glob('../data/inflammation*.csv'))

# grab just the first three filenames:

somefilenames = filenames[0:3]

# loop on the files and make some plots

for f in somefilenames:
    print(f)

    data = np.loadtxt(f, delimiter=',')

    fig = plt.figure(figsize=(10.0, 3.0))

    axes1 = fig.add_subplot(1, 3, 1)
    axes2 = fig.add_subplot(1, 3, 2)
    axes3 = fig.add_subplot(1, 3, 3)

    axes1.set_ylabel('average')
    axes1.plot(np.mean(data, axis=0))

    axes2.set_ylabel('max')
    axes2.plot(np.max(data, axis=0))

    axes3.set_ylabel('min')
    axes3.plot(np.min(data, axis=0))
```

```
../data/inflammation-01.csv
../data/inflammation-02.csv
../data/inflammation-03.csv
```







2 questions?

that was a lot of code, let's pause. Everyone get what we did? that was our most complex loop yet

The end...

3 Exercise 06 / Breakout