

Parallel organization

The project has been (again) restructured. Now, most analysis is launched from the main directory via `analyze.py`. Other scripts are in the `build` and `maps` directories.

There is some in progress stuff like `simulate`.

The rest is utilities, in `myutil`, and `test`.

The database is at main level, and it comes in in two sizes, one for as many days back as there are files at main level in the source, and the other with previous files stashed by month.

The average script starts like this:

```
import sys, os, subprocess
base = os.environ.get('covid_base')
sys.path.insert(0,base)
```

Thus, you must set `covid_base` correctly. Everything is specified as a path from `covid_base`.

Command line arguments

These can be viewed with `-h` or `--help` with any script.

Features that are currently supported are given by the `--help` flag:

```

> p3 analyze.py -h

flags
-h  --help      help
-n  <int>       display the last -n values, default: 7
-N  <int>       display -N rows of data, default: 50
-c  <int>       --delta, change from x days ago, default: 1
-L  <int>       depth of search for keys and subkeys, default: 0

-a  --all       use the complete db, starting 2020-03-22
-d  --deaths    display deaths rather than cases (default)
-f  --csv       format output as csv
-g  --graph     plot a graph of the data (not yet)
-m  --map       make a choropleth map (not yet)
-o  --only      do not descend from say, US to states
-p  --pop       normalize to population (this disables totals)
-q  --quiet     silence output (for tests)
-r  --rate      compute statistics (over last 7 days)
-s  --sort
-t  --totals    (only)
-v  --verbose   debugging mode

to do:
-u  <int>       data slice ends this many days before yesterday

example:
python analyze.py -h -n 10 -sdr

>

```

I did not use the built-in Python module for parsing the command line arguments, but rolled my own, see `uinit.py`

The statistic is the slope of a linear regression, divided by the mean of the values.

So, for example, if a 10-day series goes smoothly from 100 to 110, then the slope is about $10/10 = 1$ and the statistic is a bit less than 0.01. If the series goes from 1000 to 1100, then the slope is about $100/10 = 10$, but the statistic is still approximately 0.01.

Approach

The idea is to use the main part of the script to assemble the correct keys in order. This list is passed to `ucalc` and then to `ufmt` along with the `conf` dictionary.

All the trimming, sorting and stats happens in `ucalc` , and all the output formatting happens in `ufmt` .

The code about keys does not know which database we're using. I found that too complicated to maintain since I added the option of building a `max` database.

So now the database is passed to `ukeys` functions as an argument.

Examples (as of 2020-07-09)

Let's go through the flags one by one.

```
> p3 analyze.py -N 2
      07/03 07/04 07/05 07/06 07/07 07/08 07/09
Afghanistan 32022 32324 32672 32951 33190 33384 33594
Albania      2662   2752   2819   2893   2964   3038   3106
```

The `-N` flag takes an integer modifier, and it cuts the number of rows to that value. Since we did not provide a search term, we get the world.

```
> p3 analyze.py SC -N 4 -n 4
      07/06 07/07 07/08 07/09
Abbeville   124    135    134    137
Aiken       507    516    530    545
Allendale   61     64     64     64
Anderson    777    798    824    886
>
```

We search for `SC` (South Carolina), and we change the default number of columns from 7 to 4.

```
> p3 analyze.py SC -N 4 -n 4 -f
,07/06,07/07,07/08,07/09
Abbeville,124,135,134,137
Aiken,507,516,530,545
Allendale,61,64,64,64
Anderson,777,798,824,886
>
```

The `-f` flag asks to format as csv. This is most useful for plotting programs.

Here's the US states:

```
> p3 analyze.py US -N 5 -n 5
```

	07/05	07/06	07/07	07/08	07/09
Alabama	42359	43450	44375	45263	46424
Alaska	1107	1134	1162	1180	1222
Arizona	94567	98103	101455	105094	108614
Arkansas	22322	22907	23288	23598	24301
California	252895	264681	271035	284012	292560

We can add totals (for the whole US) with `-t` :

```
> p3 analyze.py US -N 5 -n 5 -t
```

	07/05	07/06	07/07	07/08	07/09
Alabama	42359	43450	44375	45263	46424
Alaska	1107	1134	1162	1180	1222
Arizona	94567	98103	101455	105094	108614
Arkansas	22322	22907	23288	23598	24301
California	252895	264681	271035	284012	292560
total	2820368	2868846	2916232	2974609	3032316

```
>
```

The `-o` flag limits the output to just the US

```
> p3 analyze.py US -N 5 -n 5 -o
```

	07/05	07/06	07/07	07/08	07/09
US	2820368	2868846	2916232	2974609	3032316

```
>
```

The `-p` flag normalizes to population. Not all locations have the population entered so this may fail.

```
> p3 analyze.py US -N 5 -n 5 -p
```

	07/05	07/06	07/07	07/08	07/09
Alabama	863	886	905	923	946
Alaska	151	155	158	161	167
Arizona	1299	1347	1393	1443	1492
Arkansas	739	759	771	781	805
California	640	669	685	718	740

```
>
```

The `-r` flag computes a statistic and is most useful combined with `-s` for sort:

```
> p3 analyze.py US -N 5 -n 5 -rs
      07/05  07/06  07/07  07/08  07/09  stat
Idaho      7369   7732   8051   8538   8968  0.049
Texas     192153 194932 205642 216026 224929 0.042
Montana     1167   1212   1249   1327   1371  0.041
Florida    189851 199885 206217 213563 223532 0.039
California 252895 264681 271035 284012 292560 0.036
>
```

But sort will work without `-r`

```
> p3 analyze.py US -N 5 -n 5 -s
      07/05  07/06  07/07  07/08  07/09
New York   396598 397131 397649 398237 398929
California 252895 264681 271035 284012 292560
Texas     192153 194932 205642 216026 224929
Florida    189851 199885 206217 213563 223532
New Jersey 172354 172717 172916 173196 173383
>
```

We can use the `-c` flag to show the change from a previous time. Most often, we would show the day-over-day change, but `-c 10` can give an estimate of active cases.

```
> p3 analyze.py US -N 5 -n 5 -s -c 10
      07/05  07/06  07/07  07/08  07/09
Florida    80967  85997  83391  81157  82600
California 56970  63569  64191  73428  77264
Texas      64021  60374  65469  70132  74777
Arizona    34377  34822  34796  34972  34695
Georgia    21230  21651  21705  23054  24958
>
```

And then finally, we might choose to look at deaths:

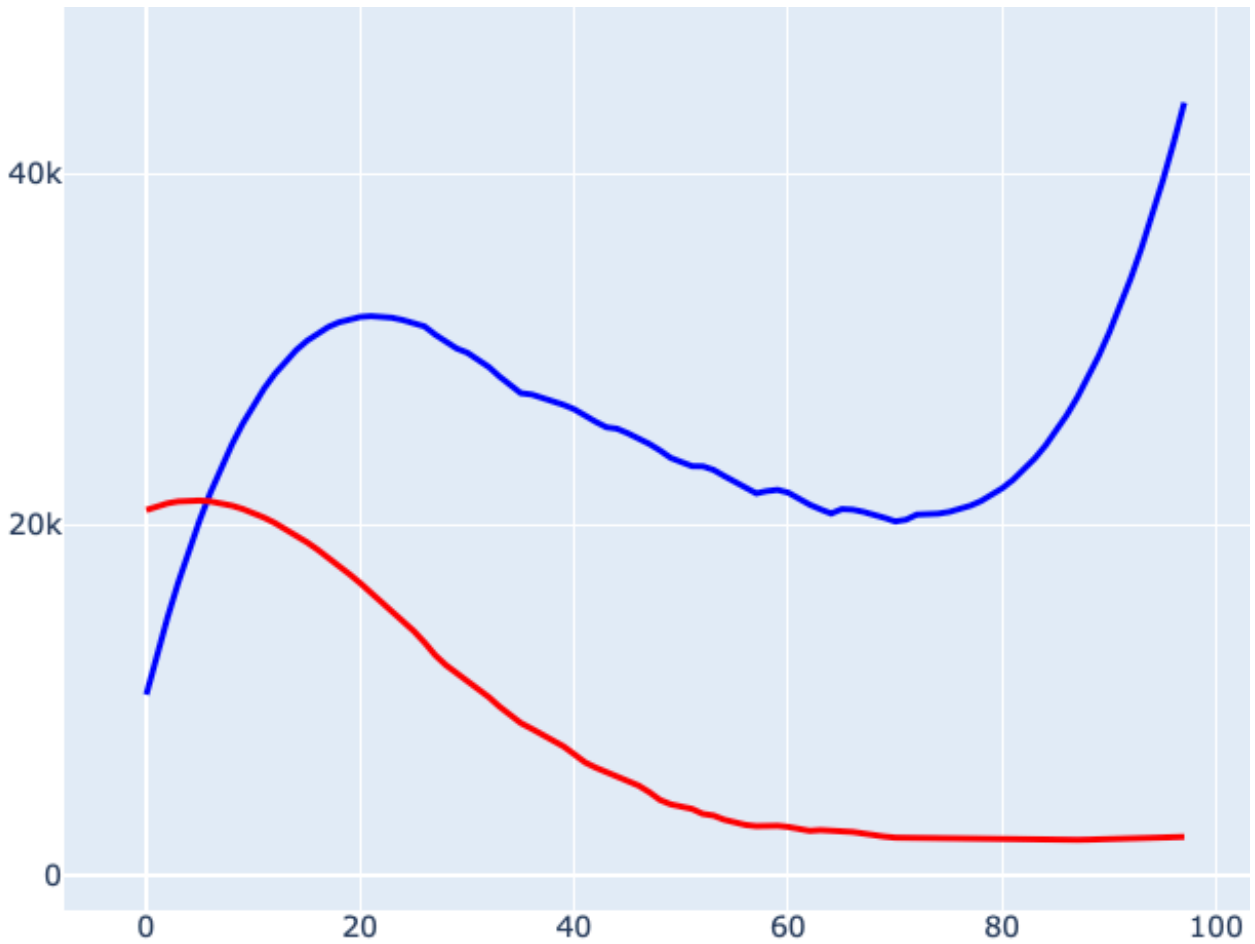
```
> p3 analyze.py US -o -dc
      07/03  07/04  07/05  07/06  07/07  07/08  07/09
US       676    697    242    268    345    959    824
>
```

There's more. You can look at counties by running `us_by_counties.py`. Maybe I will fold that into the main script soon.

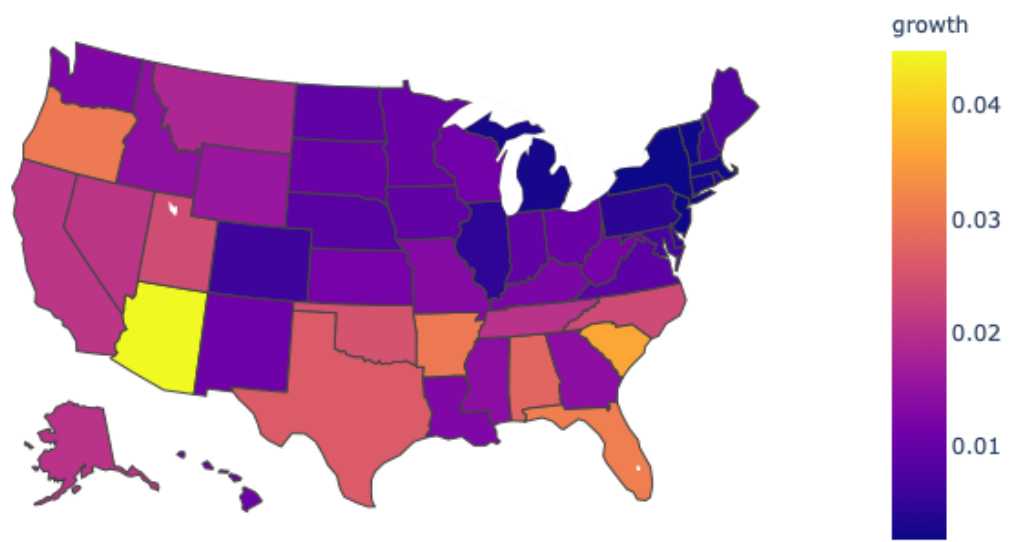
Plots and maps

Results from `plot_eu_us.py`

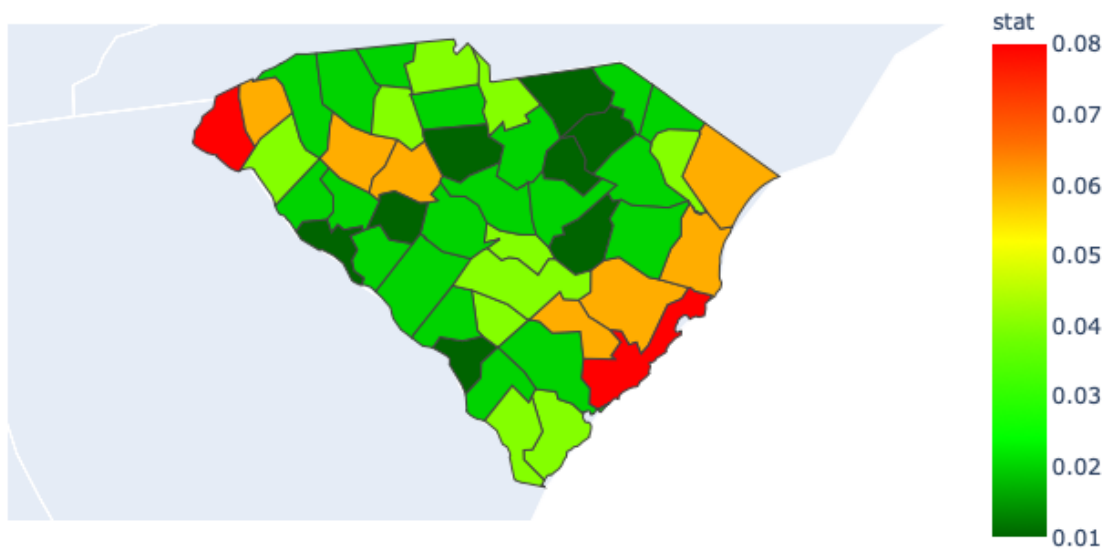
US v. EU new cases:



Choropleth 2020-06-19



and 2020-06-27



```
python3 geo/one_state_map.py CA MN SC TX WY KY
```

