Continental_US_monthly_temperatures

March 28, 2018

```
In [1]: import numpy as np
        import matplotlib.pyplot as plt
        from collections import namedtuple
```

0.1 U.S. Historical Climatology Network (USHCN)

The U.S. Historical Climatology Network (USHCN) data are used to quantify national- and regional-scale temperature changes in the contiguous United States (CONUS). The USHCN is a designated subset of the NOAA Cooperative Observer Program (COOP) Network with sites selected according to their spatial coverage, record length, data completeness, and historical stability.

- raw (`*raw*`) - raw monthly average temperature from weather stations
- curated (`*.FLs.52j.*`) - USHCN Version 2.5.5 homogenized data are associated with the naming convention "52j" to reflect the version of the Pairwise Homogenization Algorithm used.

```
In [2]: DATADIR='../data/ushcn.v2.5.5.20180322'
        STATIONS='../data/ushcn-v2-stations.txt'
```

0.1.1 Stations file format

```
011084  31.0581  -87.0547   25.9 AL BREWTON 3 SSE                  ------ ------ ------ +6
0----+----1----+----2----+----3----+----4----+----5----+----6----+----7----+----8----+----
statio      lat       lon  heigh st stationname_____________________ alias1 alias2 alias3 tz
```

```
In [3]: Station=namedtuple('Station',['id','lat','lon','h','state','name','aliases','tz'])
```

```
In [4]: def read_stations_data(filename):
            stations=[]
            for line in open(filename, "r"):
                s = Station(
                    id=line[0:6],
                    lat=float(line[8:15]),
                    lon=float(line[16:25]),
                    h=float(line[26:32]),
                    state=line[33:35],
                    name=line[36:66].strip(),
```

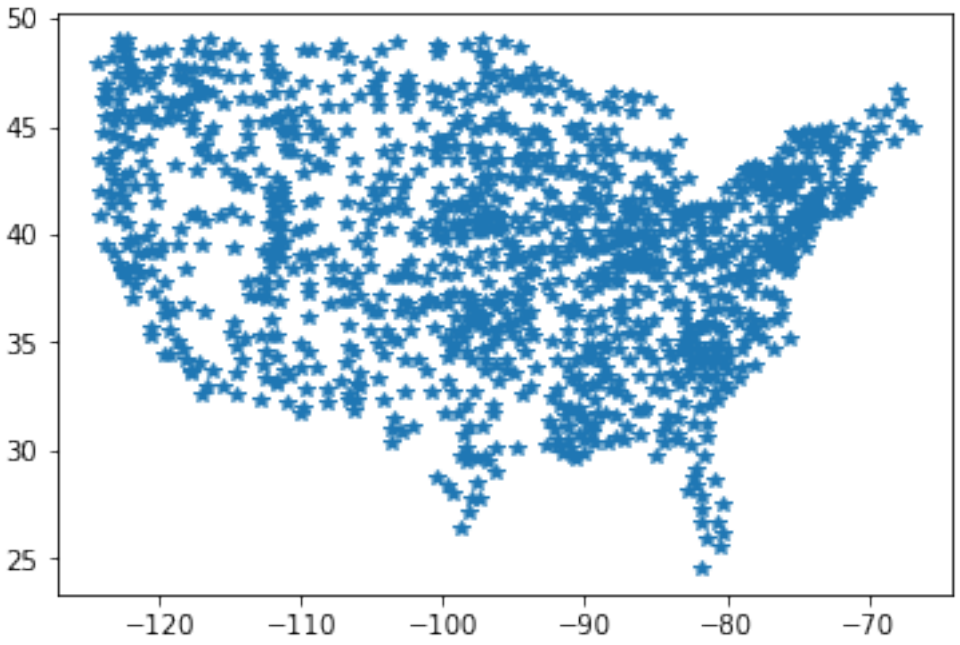
```
                    aliases=[l for l in [line[67:73], line[74:80], line[81:87]] if l!='------']
                    tz=line[88:90])
                stations.append(s)
            return stations
```

```
In [5]: stations = read_stations_data(STATIONS)
        len(stations)
```

```
Out[5]: 1218
```

```
In [6]: plt.plot([sta.lon for sta in stations],[sta.lat for sta in stations], '*')
```

```
Out[6]: [<matplotlib.lines.Line2D at 0x7f85a6efd080>]
```



0.1.2 Average temperature file format

```
USH00011084 1928   758b    1108     1571     1696     2207     2595     2834     2821     2442
USH00011084 1928   800b    1135     1614     1711     2218     2596     2829     2817    -9999
0----+----1----+----2----+----3----+----4----+----5----+----6----+----7----+----8----+----9----
?????statio year   janF     febF ....
```

- All the temperatures are in řC
- F - flags ("E" means estimated, there are some other flags)
- -9999 means unknown

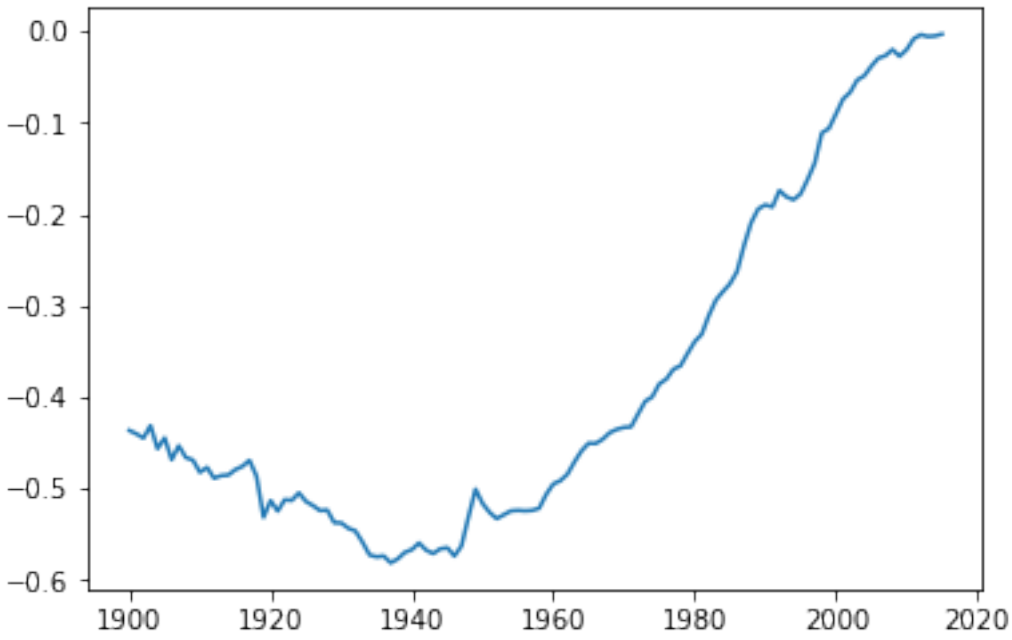
```
In [7]: MonthlyData = namedtuple('MonthlyData', ['id', 'first_year', 'last_year', 'values', 'f
```

```
In [8]: def read_monthly_file(filename):
            lines = open(filename).readlines()
            y1=int(lines[0][12:16])
            y2=int(lines[-1][12:16])
            values = np.full((y2-y1+1, 12), float('NaN'), dtype=np.float32)
            flags = np.zeros((y2-y1+1, 12), dtype=np.byte)
            for l in lines:
                y = int(l[12:16])-y1
                for m in range(12):
                    v=int(l[17+m*9:22+m*9])
                    f=l[23+m*9]
                    if v!=-9999:
                        values[y,m] = v/100.0
                    flags[y,m] = ord(f)
            return MonthlyData(id=lines[0][5:11], first_year=y1, last_year=y2, values=values,

In [9]: def read_all_data(y1=1900, y2=2010):
            raw = np.full((len(stations), y2-y1+1, 12), float('NaN'), dtype=np.float32)
            curated = np.full((len(stations), y2-y1+1, 12), float('NaN'), dtype=np.float32)
            for i,s in enumerate(stations):
                mraw=read_monthly_file(DATADIR + "/USH00"+s.id+".raw.tavg")
                mcurated=read_monthly_file(DATADIR + "/USH00"+s.id+".FLs.52j.tavg")
                ry1,ry2 = mraw.first_year,mraw.last_year
                iy1,iy2=max(y1,ry1),min(y2,ry2)
                raw[i,iy1-y1:iy2-y1+1,:] = mraw.values[iy1-ry1:iy2-ry1+1,:]
                cy1,cy2 = mcurated.first_year,mcurated.last_year
                iy1,iy2=max(y1,cy1),min(y2,cy2)
                curated[i,iy1-y1:iy2-y1+1,:] = mcurated.values[iy1-cy1:iy2-cy1+1,:]
            return raw,curated

In [10]: first_year,last_year=1900,2015
         years=np.arange(first_year,last_year+1)
         raw,curated=read_all_data(first_year,last_year)

In [11]: raw.shape,curated.shape

Out[11]: ((1218, 116, 12), (1218, 116, 12))
```

0.1.3 Difference between raw and curated data as function of year

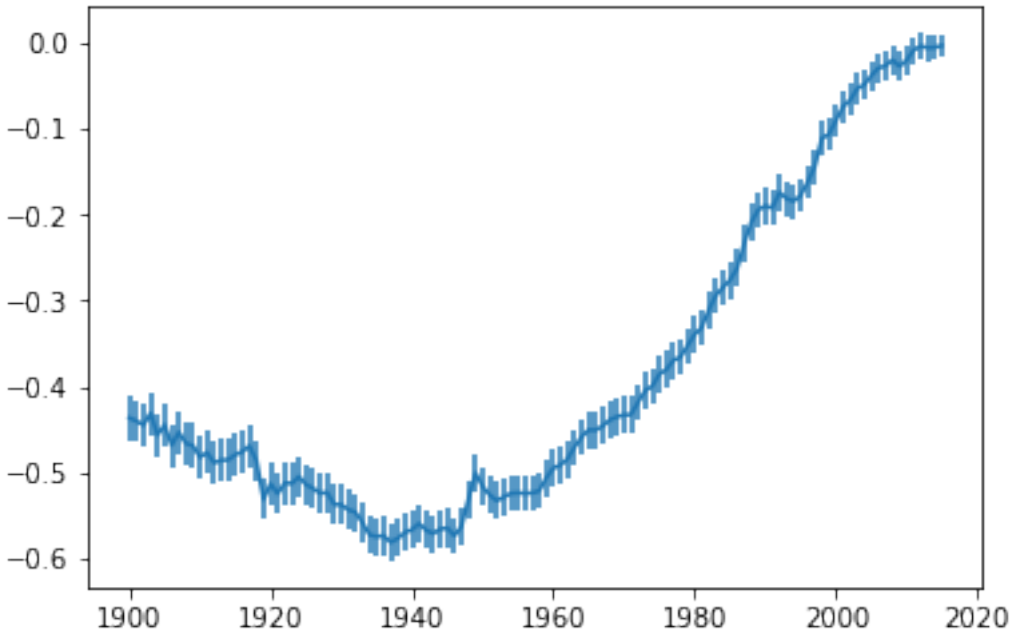
For cells where we have both raw and curated data, find the difference, and average per year.

```
In [12]: yearly_diff=np.nanmean(curated-raw,axis=(0,2))

In [13]: plt.plot(years,yearly_diff)

Out[13]: [<matplotlib.lines.Line2D at 0x7f85a6102c18>]
```



In [14]:
```
yearly_std=np.nanstd(curated-raw,axis=(0,2))/np.sqrt(len(stations))
plt.errorbar(years,yearly_diff,yearly_std)
```

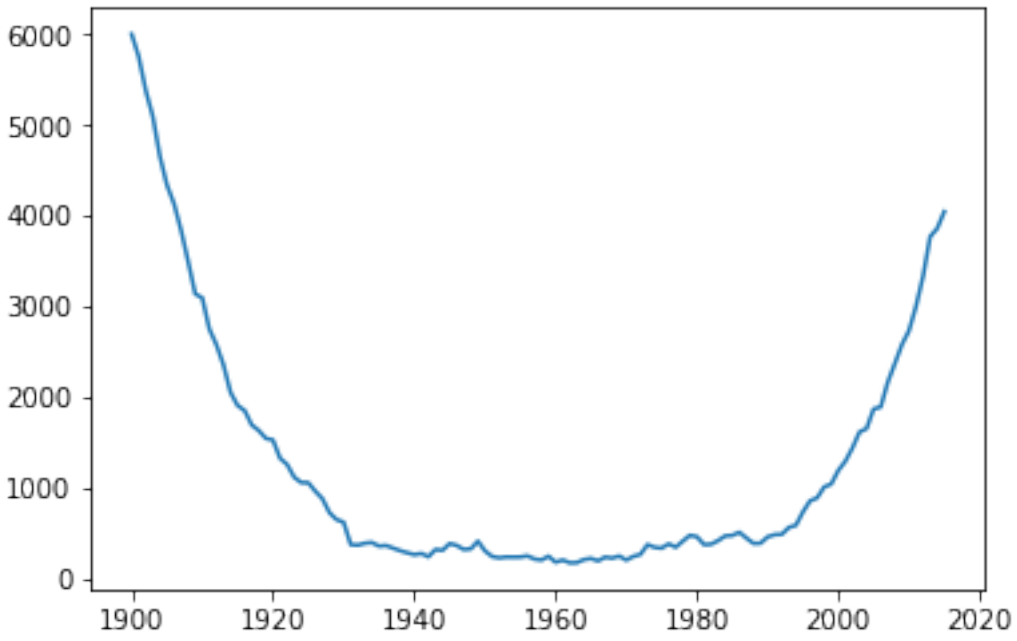
Out[14]: <Container object of 3 artists>



0.1.4 Number of holes in the raw data, as function of year

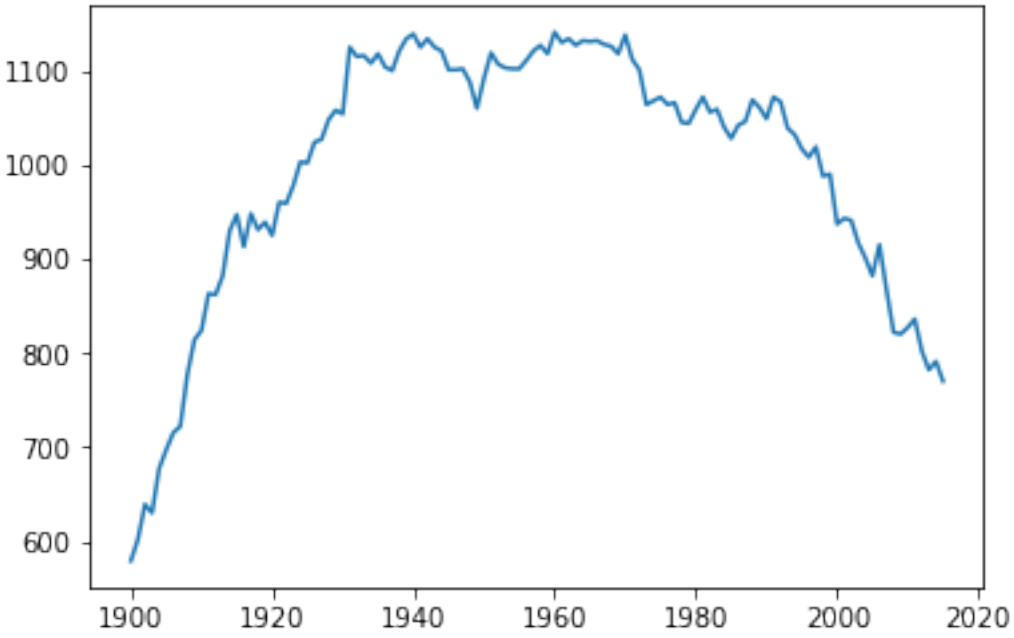
In the beginning, the number of holes is going down as expected, probably because new stations are built. Starting from ~1990 the stations start to disappear (which coincides with GW)

```
In [15]: yearly_holes=np.sum(np.isnan(raw), axis=(0,2))

In [16]: plt.plot(years,yearly_holes)

Out[16]: [<matplotlib.lines.Line2D at 0x7f85a600bb70>]
```



Number of the stations alive for the whole year

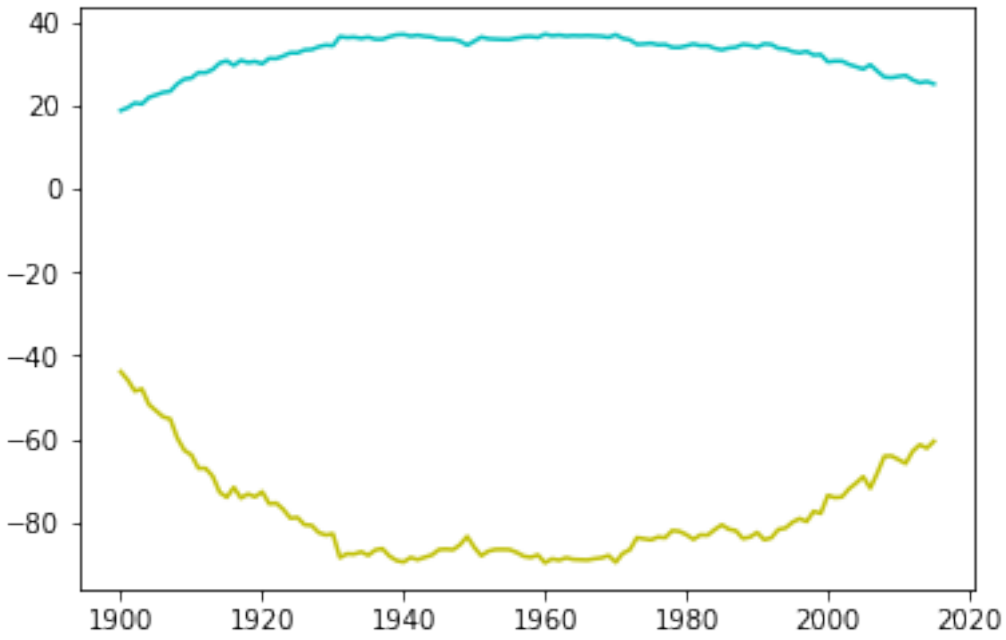
```
In [17]: alive_stations = np.sum(np.isnan(raw),axis=2)==0
         num_alive_stations = np.sum(alive_stations, axis=0)

In [18]: plt.plot(years,num_alive_stations)

Out[18]: [<matplotlib.lines.Line2D at 0x7f85a5f856d8>]
```



0.1.5 Geographic distribution (average latitude and longitude) of alive stations per year

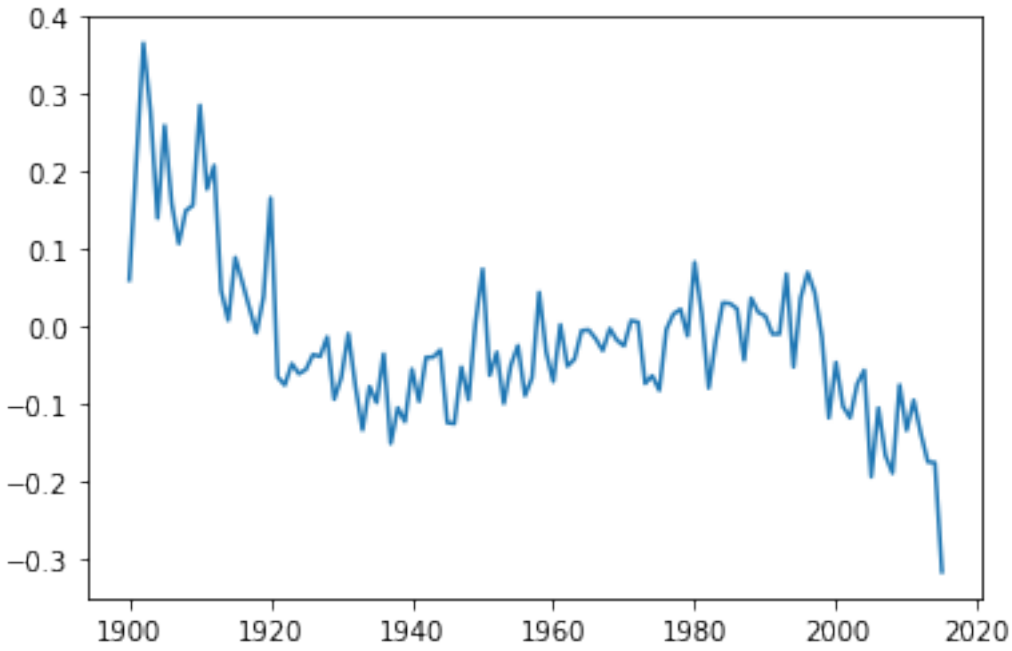
The average latitude is going up and then down. The average longitude is going left and then right.

```
In [19]: all_lat = np.array([s.lat for s in stations])
         all_lon = np.array([s.lon for s in stations])
         s_lat = np.dot(np.transpose(alive_stations), all_lat) / len(stations)
         s_lon = np.dot(np.transpose(alive_stations), all_lon) / len(stations)
         plt.plot(years,s_lat,'c',years,s_lon,'y')

Out[19]: [<matplotlib.lines.Line2D at 0x7f85a5f6d710>,
          <matplotlib.lines.Line2D at 0x7f85a5f6d9e8>]
```



1950 temperature on the station vs its life status Early stations are mostly located in warm regions; after 1990, "warm" stations are getting closed.

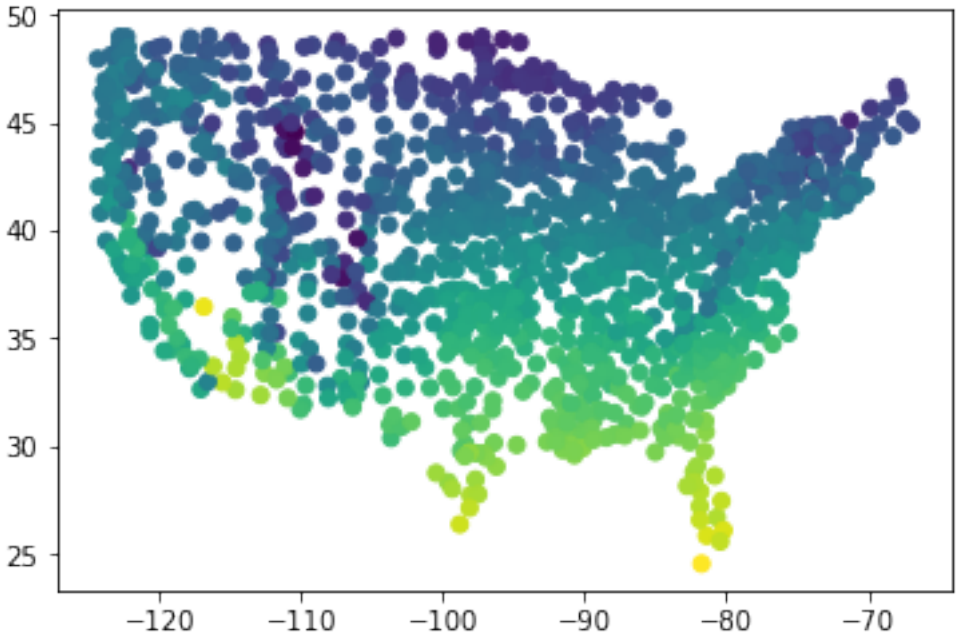
```
In [20]: # 1950ś10 average temp per station
         avg_1950=np.nanmean(raw[:,1940-first_year:1960-first_year,:], axis=(1,2))
         avgavg = np.mean(avg_1950)
         fate_by_avg_1950=np.dot(avg_1950-avgavg,alive_stations)/np.sum(alive_stations,axis=0)
         plt.plot(years,fate_by_avg_1950)

Out[20]: [<matplotlib.lines.Line2D at 0x7f85a5edea90>]
```



Just for fun - scatter plot of stations with average temperatures

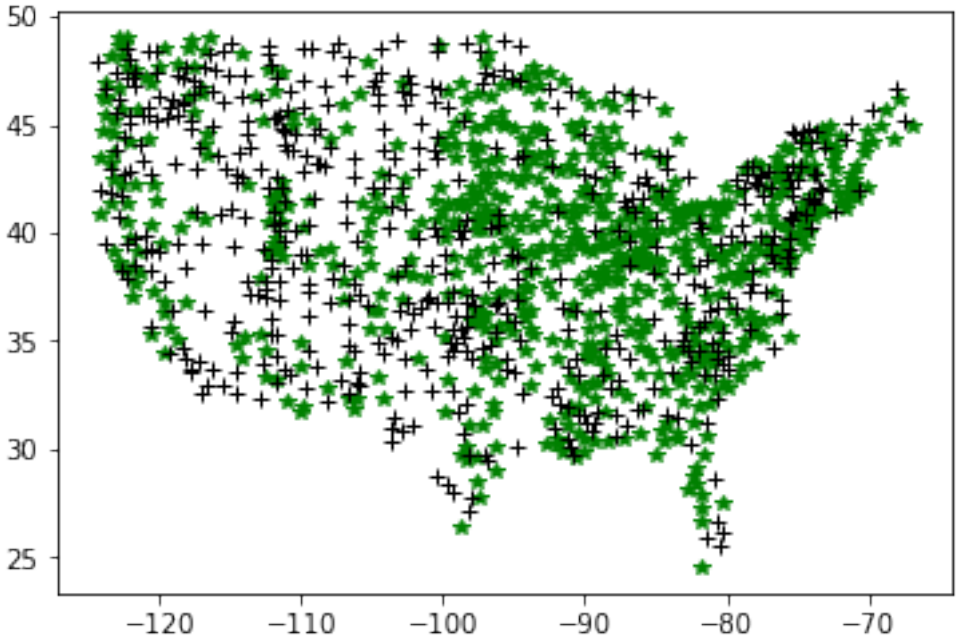
```
In [21]: plt.scatter([sta.lon for sta in stations],[sta.lat for sta in stations], c=avg_1950)
```

```
Out[21]: <matplotlib.collections.PathCollection at 0x7f85a5e54748>
```

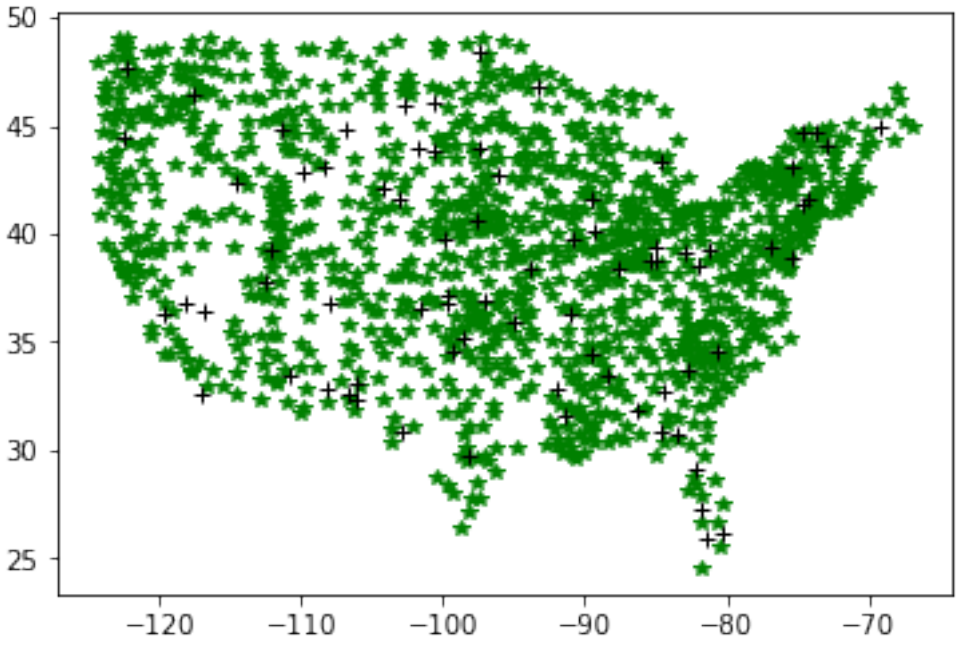


Mapping the dead and alive stations per year

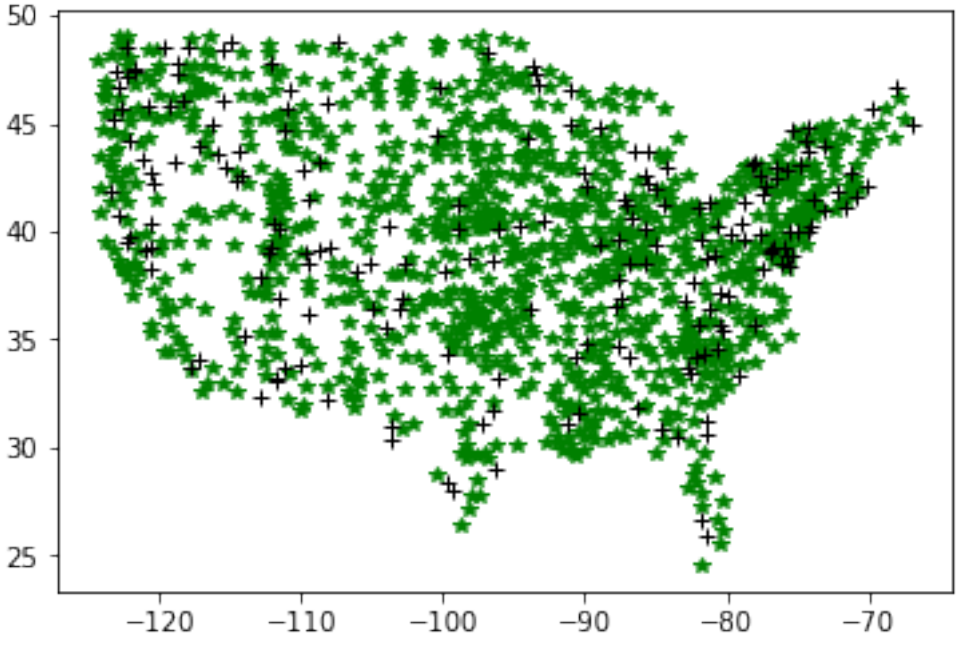
```
In [22]: alive_stations.shape

Out[22]: (1218, 116)

In [23]: def map_dead_alive(year):
             yi = year-first_year
             alive_s = [s for i,s in enumerate(stations) if alive_stations[i,yi]]
             dead_s = [s for i,s in enumerate(stations) if not alive_stations[i,yi]]
             plt.plot([s.lon for s in alive_s],[s.lat for s in alive_s], 'g*',
                      [s.lon for s in dead_s],[s.lat for s in dead_s], 'k+')

In [24]: map_dead_alive(1905)
```



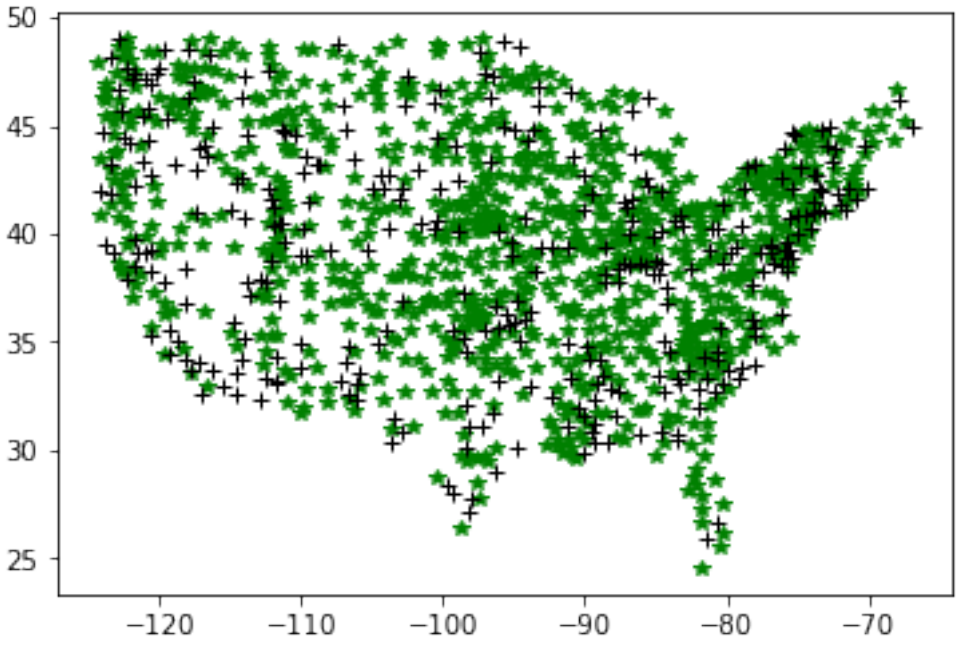

```
In [25]: map_dead_alive(1960)
```



In [26]: `map_dead_alive(1995)`



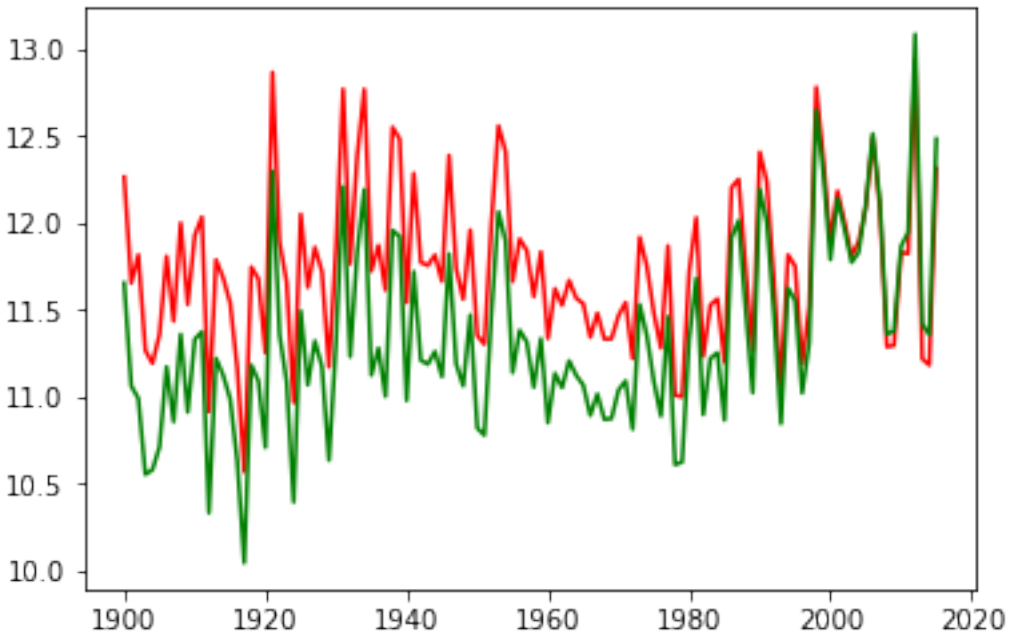
In [27]: `map_dead_alive(2005)`



0.1.6 Average raw vs curated yearly temperature

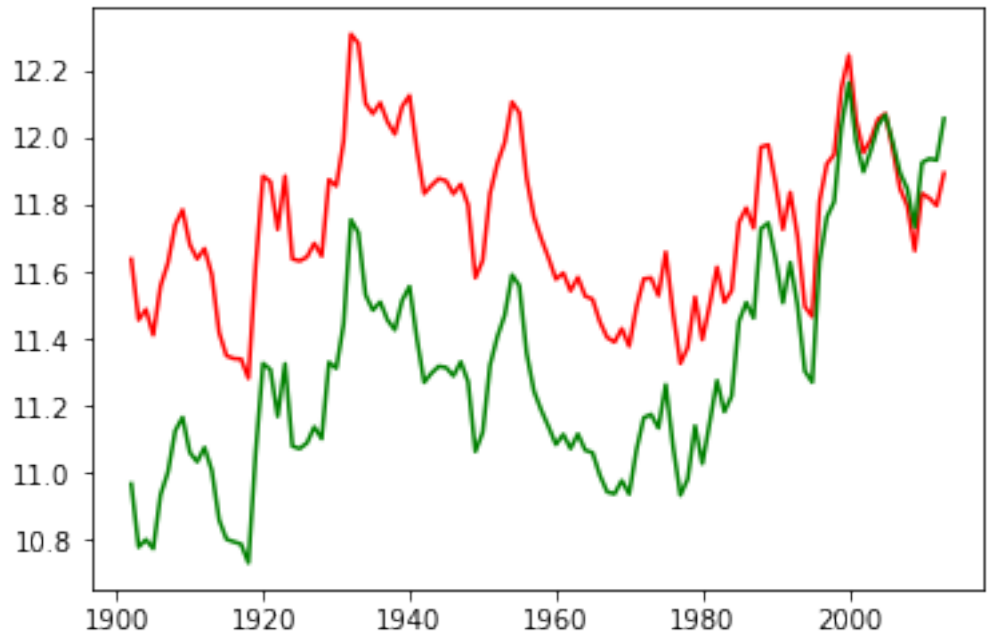
This is a naïve approach, just average of all known data. Raw is red, curated is green.

```
In [28]: naive_avg_raw = np.nanmean(raw, axis=(0,2))
         naive_avg_curated = np.nanmean(curated, axis=(0,2))

In [29]: plt.plot(years,naive_avg_raw,'r', years,naive_avg_curated,'g')

Out[29]: [<matplotlib.lines.Line2D at 0x7f85a4495b00>,
          <matplotlib.lines.Line2D at 0x7f85a4495dd8>]
```



Five year moving average of the above

```
In [30]: def moving_average(a, n=5):
             ret = np.cumsum(a, dtype=float)
             ret[n:] = ret[n:] - ret[:-n]
             return ret[n-1:] / n
         ma_raw=moving_average(naive_avg_raw)
         ma_curated=moving_average(naive_avg_curated)

In [31]: plt.plot(years[2:-2], ma_raw,'r', years[2:-2], ma_curated, 'g')

Out[31]: [<matplotlib.lines.Line2D at 0x7f85a44119e8>,
          <matplotlib.lines.Line2D at 0x7f85a4411cc0>]
```



Adding a correction for alive stations distribution

```
In [32]: plt.plot(years[2:-2], moving_average(naive_avg_raw - fate_by_avg_1950), 'r', years[2:-
```

```
Out[32]: [<matplotlib.lines.Line2D at 0x7f85a437cba8>,
          <matplotlib.lines.Line2D at 0x7f85a437ce80>]
```