1

October 3, 2017

0.1 Evgenii Safronov, Mikhail Kurenkov, Taras Melnik

In [1]:
```
import numpy as np
import scipy as sp
from matplotlib import pyplot as plt
from numpy.linalg import inv
%matplotlib inline
```

0.2 Reading the data

In [2]:
```
year, month, monthly_flux, monthly_number = [],[],[],[]
```

In [3]:
```
with open('data_group6.txt') as f:
    for line in f:
        y,m,mf,mn =  map(float, line.split())
        year.append( y )
        month.append( m )
        monthly_flux.append( mf )
        monthly_number.append( mn )
```

In [4]:
```
year = np.array(year)
month = np.array(month)
monthly_flux = np.array(monthly_flux)
monthly_number = np.array(monthly_number)
```

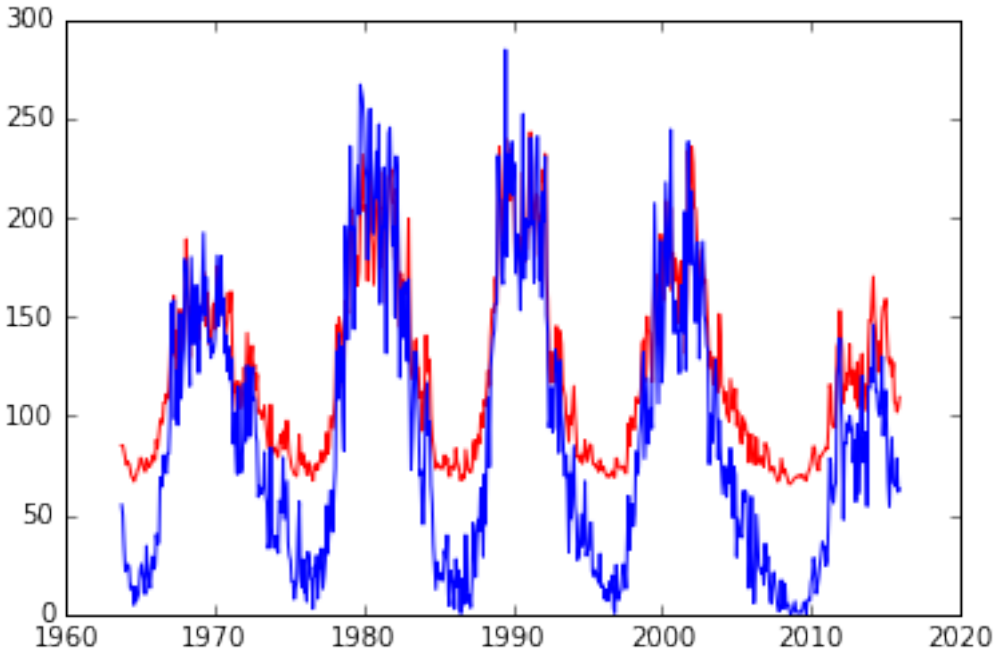
0.3 Alternative

In [5]:
```
year, month, monthly_flux, monthly_number = np.loadtxt('data_group6.txt', unpack = True)
```

0.4 Plotting

In [6]:
```
plt.plot((year + month/12), monthly_flux,'r')
plt.plot((year + month/12), monthly_number,'b')
```

Out[6]:
```
[<matplotlib.lines.Line2D at 0x7fc5f1bc6518>]
```

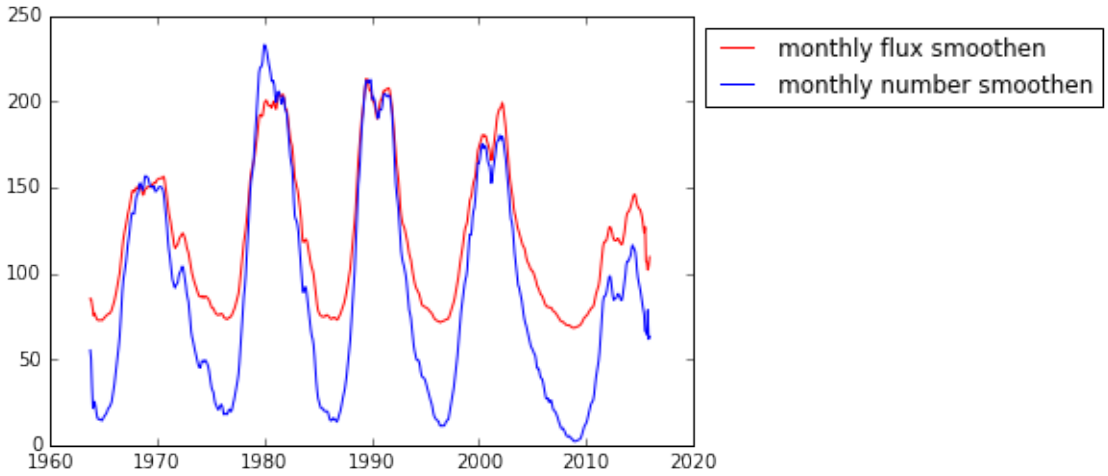


0.5 Smoothing

```
In [7]: averager = np.ones(13)
        averager[0] = averager[-1] = 0.5
        averager/=12

In [8]: mf_smooth = np.convolve(monthly_flux,  averager, mode = 'same')
        mf_smooth[:6] = monthly_flux[:6]
        mf_smooth[-6:] = monthly_flux[-6:]
        mn_smooth = np.convolve(monthly_number,averager, mode = 'same')
        mn_smooth[:6] = monthly_number[:6]
        mn_smooth[-6:] = monthly_number[-6:]

In [9]: plt.plot((year + month/12), mf_smooth,'r', label = 'monthly flux smoothen')
        plt.plot((year + month/12), mn_smooth,'b', label = 'monthly number smoothen')
        plt.legend(loc = 'upper left', bbox_to_anchor =[1,1])
        plt.show()
```

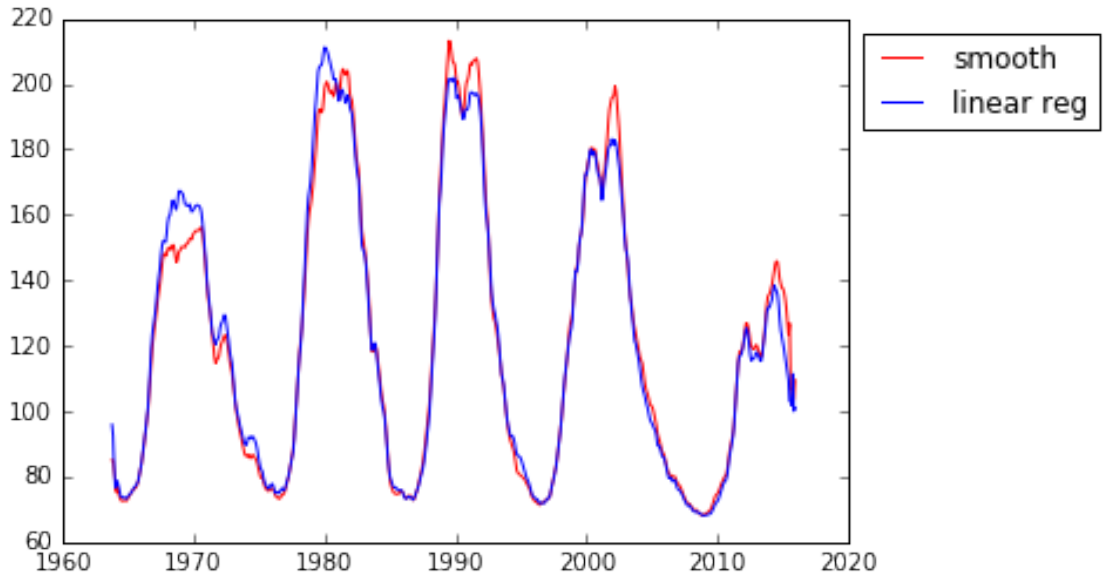


0.6 Linear regression

```
In [10]: F = mf_smooth
         sp = mn_smooth
         R = np.array([np.ones_like(sp), sp, sp**2, sp**3])
         R = R.transpose()

In [11]: beta = (inv(R.transpose().dot(R))).dot(R.transpose()).dot(F)

In [12]: plt.plot((year + month/12), mf_smooth,'r',label = 'smooth')
         plt.plot((year + month/12), R.dot(beta) ,'b', label = 'linear reg')
         plt.legend(loc = 'upper left', bbox_to_anchor =[1,1])
         plt.show()
```



0.7 Dispersion

```
In [13]: sigma_sq = np.sum( (F - R.dot(beta))**2 ) / (len(F) - 1)

In [14]: print('sigma_squared = %.2f' % (sigma_sq))

sigma_squared = 30.18
```

0.8 Today we learned how to use linear regression in matrix form via Python.