In [1]:
```
import numpy as np
import matplotlib.pyplot as plt
import scipy.stats as sp
import pandas as pd
```

In [2]:
```
data = pd.read_csv('IBM1.csv')
data.tail(5)
```

Out[2]:

	Date	Open	High	Low	Close	Adj Close	Volume
327	2020-04-21	114.000000	117.150002	112.059998	116.760002	116.760002	14349000
328	2020-04-22	119.870003	120.330002	117.550003	119.309998	119.309998	7087900
329	2020-04-23	119.570000	123.029999	119.120003	121.349998	121.349998	6881600
330	2020-04-24	122.410004	125.000000	120.760002	124.720001	124.720001	4987300
331	2020-04-27	125.559998	126.989998	125.470001	125.919998	125.919998	4923800

In [3]:
```
data.columns
```

Out[3]:
```
Index(['Date', 'Open', 'High', 'Low', 'Close', 'Adj Close', 'Volum
e'], dtype='object')
```

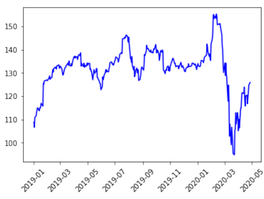
In [4]:
```
price = data['Adj Close']
price.head(5)
```

Out[4]:
```
0    108.800987
1    106.628937
2    110.793617
3    111.577446
4    113.163979
Name: Adj Close, dtype: float64
```

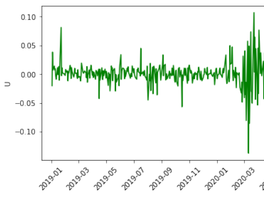
In [5]:
```
date_ymd = data['Date']
print(date_ymd[1])
from datetime import datetime
date_out = [datetime.strptime(x, '%Y-%m-%d') for x in date_ymd]
print(date_out[1])
```

```
2019-01-03
2019-01-03 00:00:00
```

In [6]:
```
plt.plot_date(date_out,price,'b-')
plt.xticks(rotation=45)
plt.show()
```



In [7]:
```
N = len(price)
M = N-1
U = np.zeros(M)
for j in range(M):
    U[j] = np.log(price[j+1]/price[j])
plt.plot_date(date_out[:-1],U,'g-')
plt.xticks(rotation=45)
plt.ylabel('U')
plt.show()
```



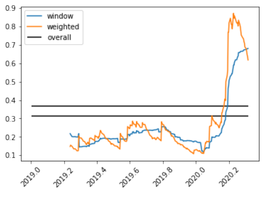
In [8]:
```
t=np.zeros(N)
for i in range(N):
    t[i] = date_out[i].year + date_out[i].timetuple().tm_yday/365.2
5
print('start = ',date_out[0],' = ',t[0])
print('end = ',date_out[-1],' = ',t[-1])
T = t[-1] - t[0]
dt = T/M
print('M = ',M)
print('T = ',T)
print('dt = ',dt)
```

```
start =  2019-01-02 00:00:00  =  2019.0054757015744
end =  2020-04-27 00:00:00  =  2020.3230663928816
M =  331
T =  1.3175906913072595
dt =  0.0039806636529629183
```

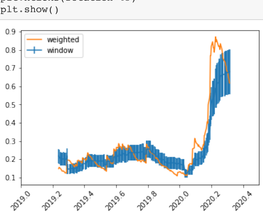
In [9]:
```
am = np.mean(U)
bm2 = np.var(U,ddof=1)
bm = np.sqrt(bm2)
print('am    = ',am)
print('bm    = ',bm)
nu = am/dt
sigma = bm/np.sqrt(dt)
nu_err = 1.96*sigma/np.sqrt(M)
sig_err = 1.96*sigma/np.sqrt(2*M)
print('nu = ',nu,' +/- ',nu_err)
print('sigma = ',sigma,'+/- ',sig_err)

def sumsq(U):
    sum = 0
    M = len(U)
    for i in range(M):
        sum = sum + U[i]**2
    return sum

sigma0 = np.sqrt(sumsq(U)/(dt*M))
print('sigma0= ',sigma0)
```

```
am    =  0.0004414693737586753
bm    =  0.021525073323290716
nu =  0.11090421606511226  +/-  0.036754494315818999
sigma =  0.3411680690359327 +/-  0.025989352169798027
sigma0=  0.34072417532526866
```

In [10]:
```
# (a) A moving window using the L = 60 most recent values of U.
L = 60
s = np.zeros(M-L+1)
tshift = t[L-1:-1]
for k in range(M-L+1):
    V = U[k:L+k]
    s[k]=np.sqrt(np.var(V,ddof=1)/dt)

w = 0.94
sw = np.zeros(M-L+1)
for k in range(M-L+1):
    sum1=0
    sum2=0
    for i in range(L):
        sum1 = sum1 + w**(i+1)*U[L+k-1-i]**2
        sum2 = sum2 + w**(i+1)
    sw[k] = np.sqrt(sum1/(sum2*dt))

plt.plot(tshift,s,label='window')
plt.plot(tshift,sw,label='weighted')
plt.hlines([sigma+sig_err,sigma-sig_err],t[0],t[-1-1],label='overal
l')
plt.legend()
plt.xticks(rotation=45)
plt.show()
```



In [11]:
```
plt.errorbar(tshift,s,yerr=1.96*s/np.sqrt(2*(L-1)),label='window')
plt.plot(tshift,sw,label='weighted')
plt.legend()
plt.xlim((2019,2020.5))
plt.xticks(rotation=45)
plt.show()
```



In [ ]:
```
```