Copper example

June 18, 2021

1 (a) Calcula[illegible]turns of Copper

[9]:
```
!pip install pandas
```

```
Requirement already satisfied: pandas in c:\users\rluck\anaconda3\lib\site-
packages (1.2.4)
Requirement already satisfied: pytz>=2017.3 in
c:\users\rluck\anaconda3\lib\site-packages (from pandas) (2021.1)
Requirement already satisfied: numpy>=1.16.5 in
c:\users\rluck\anaconda3\lib\site-packages (from pandas) (1.20.1)
Requirement already satisfied: python-dateutil>=2.7.3 in
c:\users\rluck\anaconda3\lib\site-packages (from pandas) (2.8.1)
Requirement already satisfied: six>=1.5 in c:\users\rluck\anaconda3\lib\site-
packages (from python-dateutil>=2.7.3->pandas) (1.15.0)
```


[10]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import statsmodels.api as sm
```

[11]:
```
#Reading the data from Excel
data = pd.read_csv("C:\\Users\\rluck\\OneDrive\\commodity.csv")
data
```

[11]:

	OBS	COPPER	GOLD	LEAD	SILVER
0	5/01/1989	3039.1050	363.6	4823.0596	5.091
1	5/02/1989	2976.1080	361.8	4696.2984	5.110
2	5/03/1989	2924.9100	359.9	4592.1087	5.045
3	5/04/1989	3005.0100	360.6	4678.8006	5.076
4	5/05/1989	2969.1990	360.6	4623.0428	5.076
..	…	…	…	…	…
995	2/22/1993	1872.8360	368.6	2766.1788	4.256
996	2/23/1993	1872.1830	374.5	2757.7256	4.392
997	2/24/1993	1873.7095	375.1	2752.4793	4.389
998	2/25/1993	1883.5065	377.2	2785.7061	4.489
999	2/26/1993	1884.7230	375.2	2821.4303	4.448

[1000 rows x 5 columns]

#R = ln(P_t/P_t-1)

[12]:
```python
data['R_c']= 100*np.log(data['COPPER']/data['COPPER'].shift(1))
data['R_ca']= abs(data['R_c'])
data['R_c^2'] = data['R_c']*data['R_c']
```

[13]:
```python
print(data.head())
```

```
         OBS    CO[illegible]   LEAD  SILVER       R_c      R_ca     R_c^2
0  5/01/1989  3039[illegible]0596   5.091       NaN       NaN       NaN
1  5/02/1989  2976[illegible]2984   5.110 -2.094666  2.094666  4.387625
2  5/03/1989  2924[illegible]1087   5.045 -1.735270  1.735270  3.011160
3  5/04/1989  3005.010  360.6  4678.8006   5.076  2.701718  2.701718  7.299283
4  5/05/1989  2969.199  360.6  4623.0428   5.076 -1.198868  1.198868  1.437284
```

[14]:
```python
# Dropping NA's is required to use numpy's polyfit
data_s = data.dropna(subset=['R_c'])
print(data_s.head())
```

```
         OBS    COPPER   GOLD       LEAD  SILVER       R_c      R_ca     R_c^2
1  5/02/1989  2976.108  361.8  4696.2984   5.110 -2.094666  2.094666  4.387625
2  5/03/1989  2924.910  359.9  4592.1087   5.045 -1.735270  1.735270  3.011160
3  5/04/1989  3005.010  360.6  4678.8006   5.076  2.701718  2.701718  7.299283
4  5/05/1989  2969.199  360.6  4623.0428   5.076 -1.198868  1.198868  1.437284
5  5/08/1989  2926.000  359.6  4506.0400   5.067 -1.465592  1.465592  2.147959
```

2 (b) (i) Summary statistics

[15]:
```
!pip install scipy
```

```
Requirement already satisfied: scipy in c:\users\rluck\anaconda3\lib\site-
packages (1.6.2)
Requirement already satisfied: numpy<1.23.0,>=1.16.5 in
c:\users\rluck\anaconda3\lib\site-packages (from scipy) (1.20.1)
```

[16]:
```python
from scipy import stats
Rc= stats.describe(data_s['R_c'])
Rca=stats.describe(data_s['R_ca'])
Rc2= stats.describe(data_s['R_c^2'])
```

[17]:
```python
print(Rc)
```

```
DescribeResult(nobs=999, minmax=(-10.037918530762216, 6.633374744570145),
mean=-0.04782604641413218, variance=2.988453995572702,
skewness=-0.1364205630955574, kurtosis=2.6804296191644488)
```

[18]:
```python
print(Rca)
```

```
DescribeResult(nobs=999, minmax=(0.0, 10.037918530762216),
mean=1.253037876686397, variance=1.4190654473695909,
skewness=1.9654561151759231, kurtosis=6.1340532142037)
```


[19]:
```python
print(Rc2)
```

```
DescribeResult(nob                    ), 100.75980843021948),
mean=2.98774988084                    92050785896554, skewness=6.482381876231294,
kurtosis=67.573689
```

3 b(iii) Jacqu

[20]:
```python
stats.jarque_bera(data_s['R_c'])
```

[20]:
```
Jarque_beraResult(statistic=302.1619199254945, pvalue=0.0)
```


[21]:
```python
stats.jarque_bera(data_s['R_ca'])
```

[21]:
```
Jarque_beraResult(statistic=2209.4000465672993, pvalue=0.0)
```

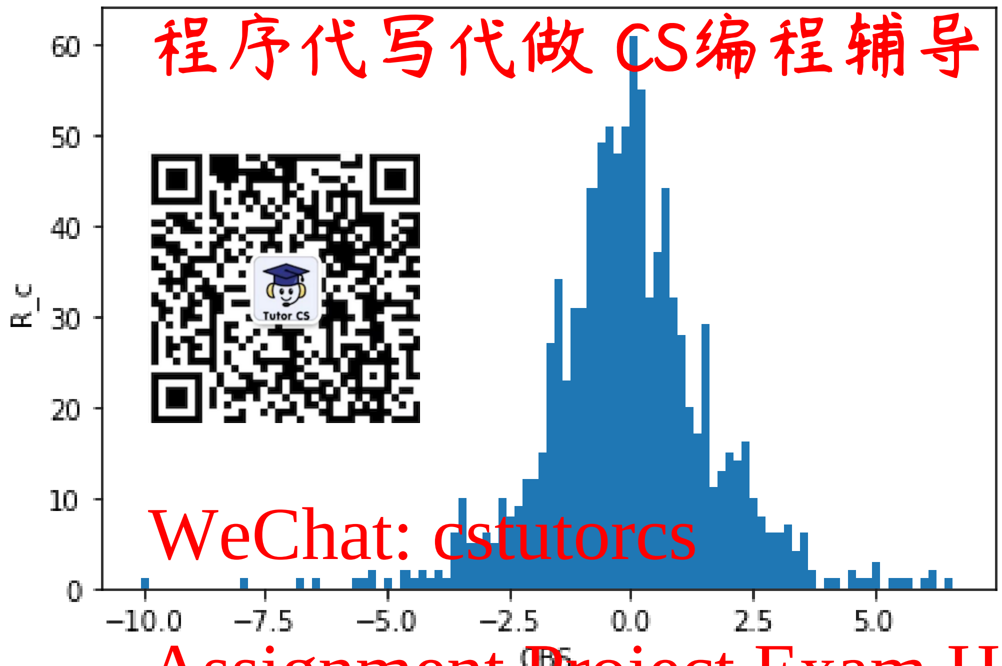

[22]:
```python
stats.jarque_bera(data_s['R_c^2'])
```

[22]:
```
Jarque_beraResult(statistic=197064.76373846942, pvalue=0.0)
```


Interpretation: The JB tests show there is a non-normal distribution (p-value <0.05), given the Copper's daily returns is negatively skewed and has an excess Kurtosis of 2.68.

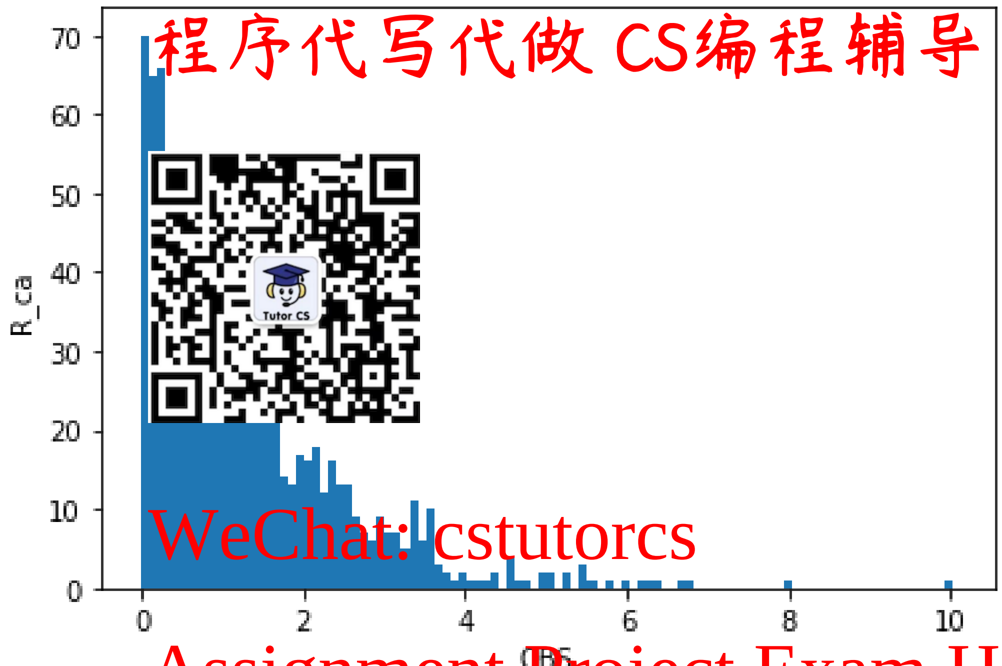
4 b(iv) Histogram

[23]:
```python
_ = plt.hist(data['R_c'],bins=100)
_ = plt.xlabel('OBS')
_ = plt.ylabel('R_c')
plt.show()
plt.savefig('C:\\Users\\rluck\\OneDrive\\myplot.png')
```

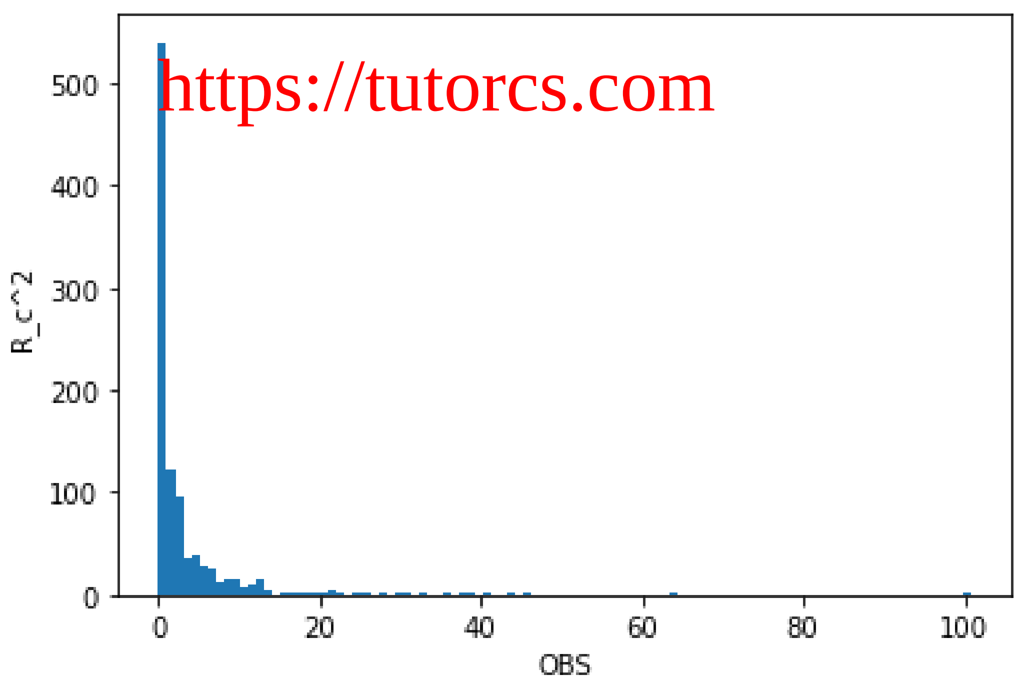


<Figure size 432x288 with 0 Axes>

[24]:
```
_ = plt.hist(data['R_ca'],bins=100)
_ = plt.xlabel('OBS')
_ = plt.ylabel('R_ca')
plt.show()
```



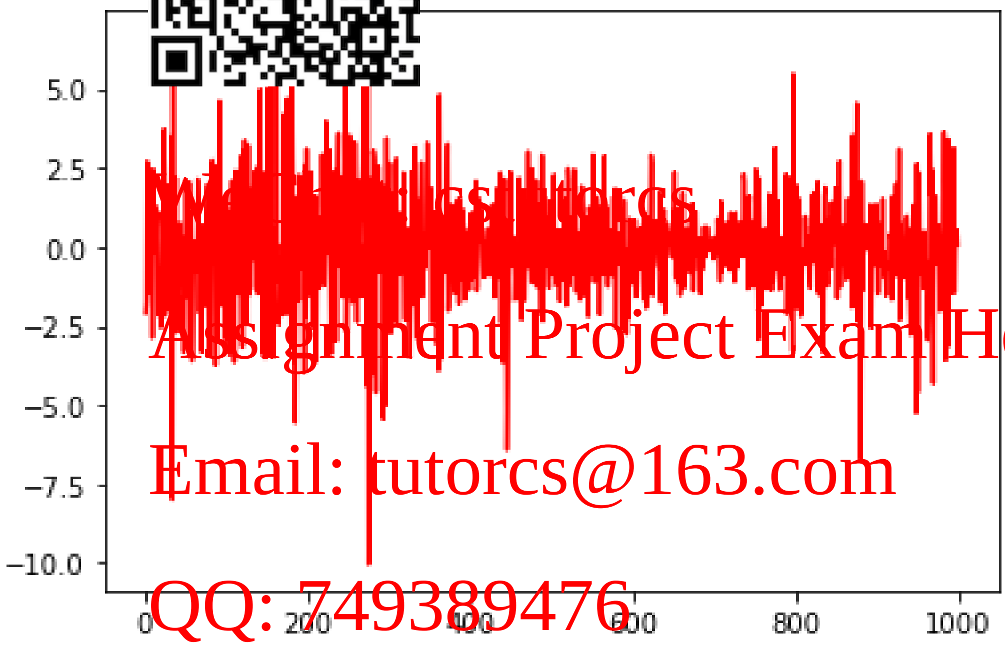
[25]:
```
_ = plt.hist(data['R_c^2'],bins=100)
_ = plt.xlabel('OBS')
_ = plt.ylabel('R_c^2')
plt.show()
```



5 Plotting charts

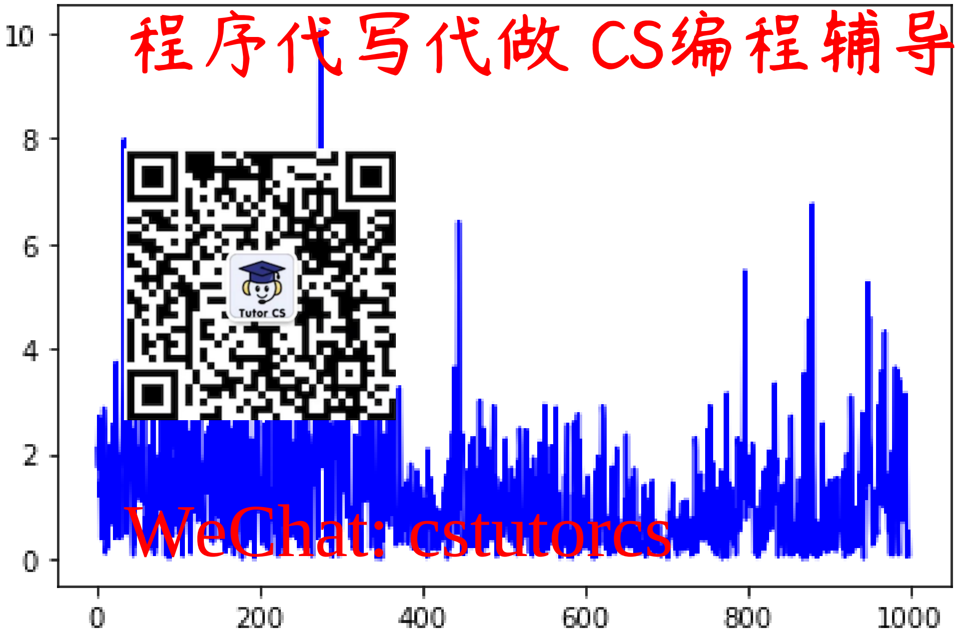
[26]:
```
plt.plot(data.R_c
```

[26]: [<matplotlib.line ca529a0>]



[27]:
```
plt.plot(data.R_ca,color='blue')
```

[27]: [<matplotlib.lines.Line2D at 0x2491cab0790>]



[28]:
```
dta= data_s['R_c']
```

6 Computing ACF and PACF

[29]:
```
>>> sm.graphics.tsa.plot_pacf(dta.values.squeeze(), lags=24)
sm.graphics.tsa.plot_acf(dta.values.squeeze(),lags=24)
>>> plt.show()
```


Partial Autocorrelation
1.0
0.8
0.6
0.4
0.2
0.0
−0.2
0
5
10
15
20
25


Autocorrelation
1.0
0.8
0.6
0.4
0.2
0.0
−0.2
0
5
10
15
20
25

```
[30]: r,q,p = sm.tsa.acf(dta.values.squeeze(), qstat=True)
      data = np.c_[range(1,41), r[1:], q, p]
      table = pd.DataFrame(data, columns=['lag', "AC", "Q", "Prob(>Q)"])
      print (table.set_index('lag'))
```

```
            AC          Q      Prob(>Q)
lag
1.0  -0.169258  [illegible]  [illegible]
2.0   0.013418  [illegible]  [illegible]
3.0  -0.034927  [illegible]  [illegible]
4.0  -0.042273  [illegible]  [illegible]
5.0   0.040801  [illegible]  [illegible]
6.0  -0.030759  34.534399  5.303518e-06
7.0  -0.048466  36.902332  4.894754e-06
8.0   0.008219  36.970498  1.165359e-05
9.0   0.023870  37.546022  2.102220e-05
10.0 -0.000546  37.546323  4.550870e-05
11.0  0.031456  38.547811  6.320925e-05
12.0 -0.019199  38.921279  1.084230e-04
13.0 -0.042012  40.711358  1.061207e-04
14.0 -0.010708  40.827757  1.893163e-04
15.0  0.050749  43.445094  1.342517e-04
16.0  0.012242  43.597563  2.270240e-04
17.0 -0.021301  44.059599  3.356420e-04
18.0  0.019068  44.430240  5.005839e-04
19.0  0.001336  44.432061  8.226156e-04
20.0  0.046795  46.668800  6.519565e-04
21.0 -0.017221  46.972047  9.472905e-04
22.0  0.060033  50.660881  4.770893e-04
23.0 -0.026672  51.389781  6.034195e-04
24.0  0.001243  51.391364  9.386307e-04
25.0 -0.062687  55.425856  4.326261e-04
26.0 -0.067126  60.056752  1.647766e-04
27.0  0.025273  60.713863  2.124971e-04
28.0  0.006739  60.760627  3.253423e-04
29.0  0.084367  68.098464  5.494580e-05
30.0  0.053573  71.060362  3.495973e-05
31.0 -0.068947  75.971216  1.206393e-05
32.0  0.011689  76.112502  1.858629e-05
33.0 -0.007772  76.175038  2.892027e-05
34.0 -0.074013  81.851574  7.997707e-06
35.0  0.068058  86.656475  2.858653e-06
36.0 -0.072387  92.097697  8.262371e-07
37.0  0.002966  92.106843  1.348620e-06
38.0 -0.004419  92.127165  2.164122e-06
39.0  0.003125  92.137336  3.438375e-06
40.0  0.014532  92.357529  5.063225e-06
```

```
C:\Users\rluck\anaconda3\lib\site-packages\statsmodels\tsa\stattools.py:657:
FutureWarning: The default number of lags is changing from 40 to min(int(10 *
np.log10(nobs)), nobs - 1) after 0.12is released. Set the number of lags to an
integer to  silence this warning.
  warnings.warn(
C:\Users\rluck\ana                   ckages\statsmodels\tsa\stattools.py:667:
FutureWarning: fft                   the default after the release of the 0.12
release of statsmo                   this warning, explicitly set fft=False.
  warnings.warn(
```


As per the above cor utocorrelation of -0.169 is statistically significant as it exceeds the bands (i. area) in the chart. We can infer that the log price of Copper does not follow a random walk and the efficient market hypothesis does not hold. For squared returns (R_c^2) and absolute returns (R_ca), the autocorrelations are even stronger, as demonstrated by the following ACF and PACF charts.

[31]:
```
dta_1= data_s['R_ca']
```

[32]:
```
>>> sm.graphics.tsa.plot_pacf(dta_1.values.squeeze(), lags=12)
sm.graphics.tsa.plot_acf(dta_1.values.squeeze(), lags=12)
>>> plt.show()
```


[33]:
```python
r,q,p = sm.tsa.acf(dta_1.values.squeeze(), qstat=True)
data = np.c_[range(1,41), r[1:], q, p]
table = pd.DataFrame(data, columns=['lag', "AC", "Q", "Prob(>Q)"])
print (table.set_index('lag'))
```

lag	AC	Q	Prob(>Q)
1.0	0.250814	63.038756	2.031943e-15
2.0	0.152524	86.367442	1.760144e-19
3.0	0.153909	110.150402	1.018515e-23
4.0	0.127490	126.485836	2.196721e-26
5.0	0.150763	149.352358	1.833912e-30
6.0	0.140662	169.277473	6.400788e-34
7.0	0.109180	181.293948	1.038331e-35
8.0	0.098477	191.079671	4.827095e-37
9.0	0.100862	201.355470	1.722055e-38
10.0	0.119461	215.785175	8.144212e-41
11.0	0.068930	220.594251	3.883007e-41
12.0	0.056851	223.868911	3.739222e-41
13.0	0.059601	227.471661	2.983310e-41
14.0	0.078392	233.710486	6.632136e-42
15.0	0.076650	239.681269	1.646911e-42
16.0	0.074328	245.301502	4.753518e-43
17.0	0.079118	251.675918	9.486837e-44
18.0	0.109714	263.946256	1.175163e-45

```
19.0  0.103477  274.872190  2.735358e-47
20.0  0.085096  282.268874  1.322181e-48
21.0  0.129912  299.525625  3.972767e-51
22.0  0.105942  311.013572  7.090149e-53
23.0  0.107406  [illegible]  [illegible]e-54
24.0  0.096674  33[illegible]  [illegible]e-56
25.0  0.108696  34[illegible]  [illegible]e-58
26.0  0.096625  35[illegible]  [illegible]e-59
27.0  0.110406  36[illegible]  [illegible]e-61
28.0  0.090840  37[illegible]  [illegible]e-62
29.0  0.088709  38[illegible]  [illegible]e-63
30.0  0.088184  39[illegible]  [illegible]e-64
31.0  0.133722  409.793997  1.092694e-67
32.0  0.066202  414.326274  4.907364e-68
33.0  0.129816  431.771517  5.489116e-71
34.0  0.089096  439.997554  1.430751e-72
35.0  0.102302  450.854107  1.054561e-73
36.0  0.064577  455.184466  5.153559e-74
37.0  0.076557  461.276941  1.108424e-74
38.0  0.080198  467.969607  1.800223e-75
39.0  0.108762  480.291745  2.170477e-77
40.0  0.049010  482.796366  2.423205e-77

C:\Users\rluck\anaconda3\lib\site-packages\statsmodels\tsa\stattools.py:657:
FutureWarning: The default number of lags is changing from 40 tomin(int(10 *
np.log10(nobs)), nobs - 1) after 0.12is released. Set the number of lags to an
integer to  silence this warning.
  warnings.warn(
C:\Users\rluck\anaconda3\lib\site-packages\statsmodels\tsa\stattools.py:667:
FutureWarning: fft=True will become the default after the release of the 0.12
release of statsmodels. To suppress this warning, explicitly set fft=False.
  warnings.warn(
```

[34]:

```
dta_2= data_s['R_c^2']
```

[35]:

```
>>> sm.graphics.tsa.plot_pacf(dta_2.values.squeeze(), lags=12)
sm.graphics.tsa.plot_acf(dta_2.values.squeeze(),lags=12)
>>> plt.show()
```


Partial Autocorrelation
1.0
0.8
0.6
0.4
0.2
0.0
0
2
4
6
8
10
12


Autocorrelation
1.0
0.8
0.6
0.4
0.2
0.0
0
2
4
6
8
10
12

[36]:

```
r,q,p = sm.tsa.acf(dta_2.values.squeeze(), qstat=True)
data = np.c_[range(1,41), r[1:], q, p]
table = pd.DataFrame(data, columns=['lag', "AC", "Q", "Prob(>Q)"])
print (table.set_index('lag'))
```


lag	AC	Q	Prob(>Q)
1.0	0.267265	[illegible]	[illegible]e-17
2.0	0.136738	[illegible]	[illegible]e-20
3.0	0.111404	[illegible]	[illegible]e-22
4.0	0.099766	[illegible]	[illegible]e-23
5.0	0.120752	[illegible]	[illegible]e-26
6.0	0.125132	143.228817	2.086198e-28
7.0	0.066961	147.748703	1.205533e-28
8.0	0.055213	150.824826	1.319382e-28
9.0	0.100740	161.075827	4.438458e-30
10.0	0.091719	169.581739	3.386147e-31
11.0	0.032893	170.676845	8.560979e-31
12.0	0.031368	171.673737	2.169953e-30
13.0	0.039585	173.262956	4.004580e-30
14.0	0.051436	175.948865	4.288407e-30
15.0	0.038930	177.489023	7.623657e-30
16.0	0.009632	177.583403	2.568328e-29
17.0	0.057397	180.938148	1.876983e-29
18.0	0.096342	190.399625	8.250680e-31
19.0	0.081927	197.248691	1.199337e-31
20.0	0.058761	200.775589	7.903797e-32
21.0	0.110171	213.186262	9.044721e-34
22.0	0.082405	220.136703	1.246595e-34
23.0	0.077137	226.233201	2.526215e-35
24.0	0.065301	230.606764	1.113264e-35
25.0	0.034623	231.837496	2.011860e-35
26.0	0.044748	233.895403	2.468183e-35
27.0	0.090672	242.353562	1.698376e-36
28.0	0.102785	253.233828	3.941894e-38
29.0	0.057602	256.654487	2.599329e-38
30.0	0.017237	256.961123	6.832310e-38
31.0	0.073249	262.503968	1.722169e-38
32.0	0.011442	262.639345	4.777961e-38
33.0	0.063586	266.824825	2.179022e-38
34.0	0.048306	269.242934	2.159314e-38
35.0	0.044036	271.254528	2.539467e-38
36.0	0.049462	273.795001	2.349134e-38
37.0	0.043425	275.755218	2.782275e-38
38.0	0.027704	276.553889	5.423483e-38
39.0	0.041901	278.382719	6.688394e-38
40.0	0.020043	278.801625	1.506677e-37

```
C:\Users\rluck\anaconda3\lib\site-packages\statsmodels\tsa\stattools.py:657:
FutureWarning: The default number of lags is changing from 40 to min(int(10 *
np.log10(nobs)), nobs - 1) after 0.12is released. Set the number of lags to an
integer to  silence this warning.
  warnings.warn(
C:\Users\rluck\ana[illegible]ckages\statsmodels\tsa\stattools.py:667:
FutureWarning: fft[illegible]the default after the release of the 0.12
release of statsmo[illegible]this warning, explicitly set fft=False.
  warnings.warn(
```