訊號與系統的期末報告

In [1]:

```
# Get thinkdsp.py

import os

if not os.path.exists('thinkdsp.py'):
    !wget https://github.com/AllenDowney/ThinkDSP/raw/master/code/thinkdsp.py
```

我學到了許多有關python從庫裡面import程式的方法，不論是從外部的庫，或是內部的庫，抑或是網路上giyhub上的庫，也學到了許多訊號處理的使用，並進行分析。

但最令我感到興趣的是，從歷史資料(csv檔案)中，抓取數據並做成波型再進行訊號上的處理

像是有一次的作業中，抓取比特幣數據去分析他是屬於甚麼程度的噪音訊號，或是另一次分析自相關程度

那麼，我也想嘗試看看對加密貨幣做個訊號的處理。我將會使用BTC ETH作為我的數據

時間皆從2017/11/09~2022/06/17

In [2]:

```
import numpy as np
import matplotlib.pyplot as plt

from thinkdsp import decorate
```

In [3]:

```
import pandas as pd
```

In [4]:

```
from thinkdsp import Spectrum
```

導入兩者的數據

In [5]:

```
btc = pd.read_csv('BTC-USD.csv',
                  parse_dates=[0])
```

In [6]:

```
btc
```

Out[6]:

	Date	Open	High	Low	Close	Adj Close	Volum
0	2017-11-09	7446.830078	7446.830078	7101.520020	7143.580078	7143.580078	322624998
1	2017-11-10	7173.729980	7312.000000	6436.870117	6618.140137	6618.140137	520824985
2	2017-11-11	6618.609863	6873.149902	6204.220215	6357.600098	6357.600098	490868019
3	2017-11-12	6295.450195	6625.049805	5519.009766	5950.069824	5950.069824	895734988
4	2017-11-13	5938.250000	6811.189941	5844.290039	6559.490234	6559.490234	626324992
...	...	...	...	...	...	...	
1676	2022-06-12	28373.513672	28502.685547	26762.648438	26762.648438	26762.648438	3416322027
1677	2022-06-13	26737.578125	26795.589844	22141.257813	22487.388672	22487.388672	6820455644
1678	2022-06-14	22487.986328	23018.951172	20950.818359	22206.792969	22206.792969	5091357524
1679	2022-06-15	22196.730469	22642.671875	20178.376953	22572.839844	22572.839844	5491200701
1680	2022-06-16	22632.828125	22805.945313	20869.099609	21065.482422	21065.482422	338316718(

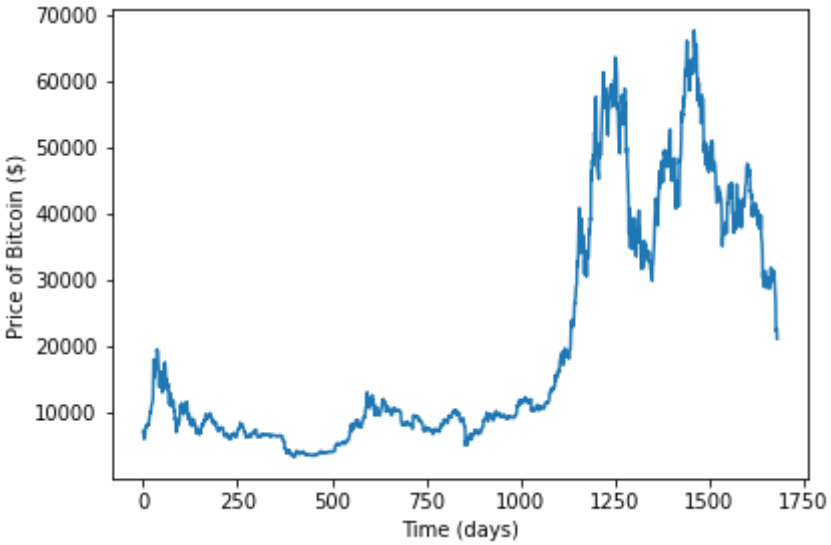
1681 rows × 7 columns

In [7]:

```
btc_ys = btc['Close']
btc_ts = btc.index
```

In [8]:

```
from thinkdsp import Wave

btc_wave = Wave(btc_ys, btc_ts, framerate=1)
btc_wave.plot()
decorate(xlabel='Time (days)',
         ylabel='Price of Bitcoin ($)')
```



In [9]:

```
eth = pd.read_csv('ETH-USD.csv',
                  parse_dates=[0])
```

In [10]:

```
eth
```

Out[10]:

	Date	Open	High	Low	Close	Adj Close	Volume
0	2017-11-09	308.644989	329.451996	307.056000	320.884003	320.884003	893249984
1	2017-11-10	320.670990	324.717987	294.541992	299.252991	299.252991	885985984
2	2017-11-11	298.585999	319.453003	298.191986	314.681000	314.681000	842300992
3	2017-11-12	314.690002	319.153015	298.513000	307.907990	307.907990	1613479936
4	2017-11-13	307.024994	328.415009	307.024994	316.716003	316.716003	1041889984
...	...	...	...	...	...	...	...
1676	2022-06-12	1530.189697	1539.705078	1436.183960	1445.216553	1445.216553	23465074882
1677	2022-06-13	1443.835449	1448.738037	1181.948242	1204.582764	1204.582764	45162788786
1678	2022-06-14	1204.555298	1252.471802	1094.701904	1211.662842	1211.662842	33327826525
1679	2022-06-15	1211.365967	1236.627563	1025.684204	1233.206421	1233.206421	37539999450
1680	2022-06-16	1239.099243	1245.067261	1092.004883	1104.854858	1104.854858	22591854592

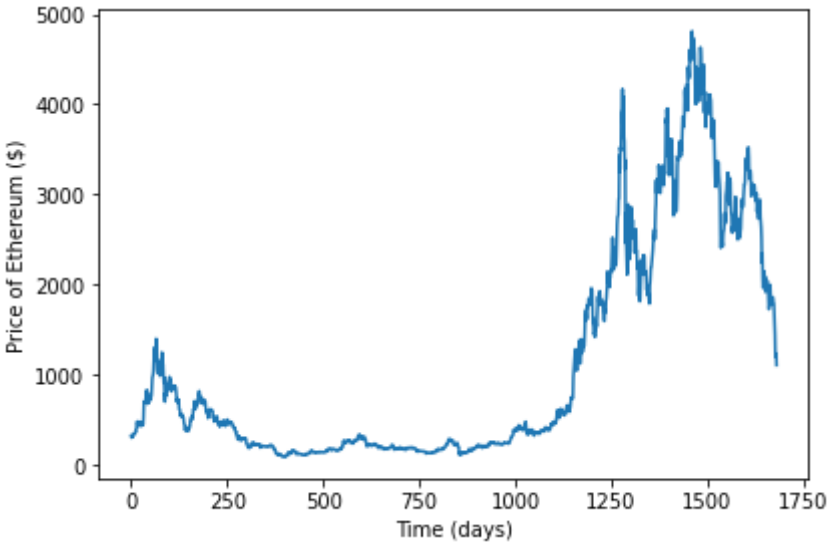
1681 rows × 7 columns

In [11]:

```
eth_ys = eth['Close']
eth_ts = eth.index
```

In [12]:

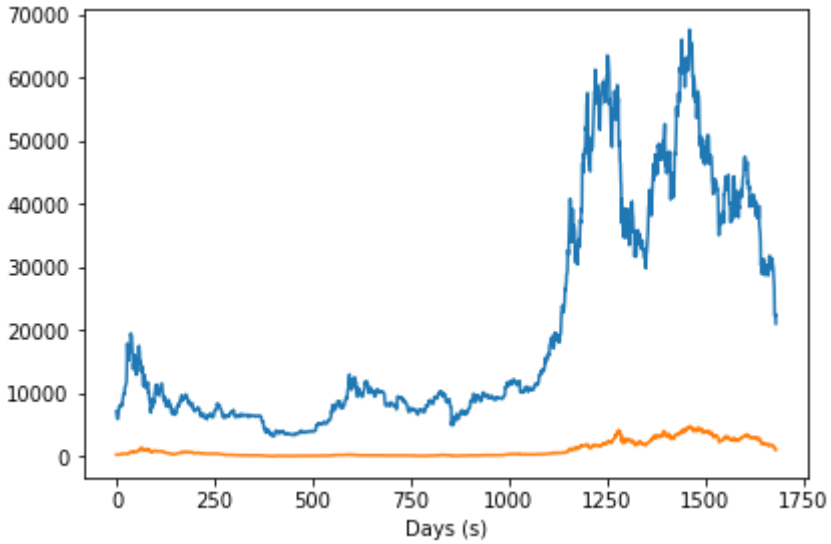
```
from thinkdsp import Wave

eth_wave = Wave(eth_ys, eth_ts, framerate=1)
eth_wave.plot()
decorate(xlabel='Time (days)',
         ylabel='Price of Ethereum ($)')
```



兩個放在一起看一下

In [13]:

```
btc_wave.plot()
eth_wave.plot()
decorate(xlabel='Days (s)')
```



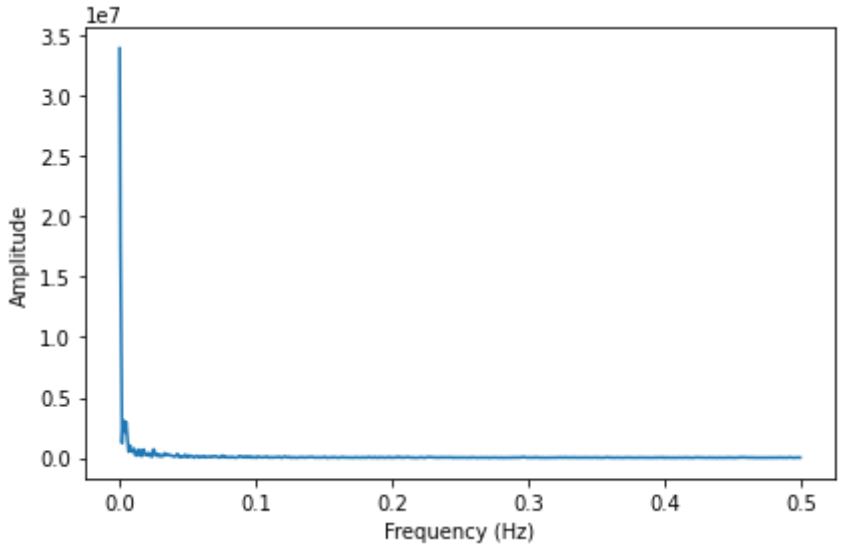
來看看頻譜的感覺

In [14]:

```
loglog = dict(xscale='log', yscale='log')
```

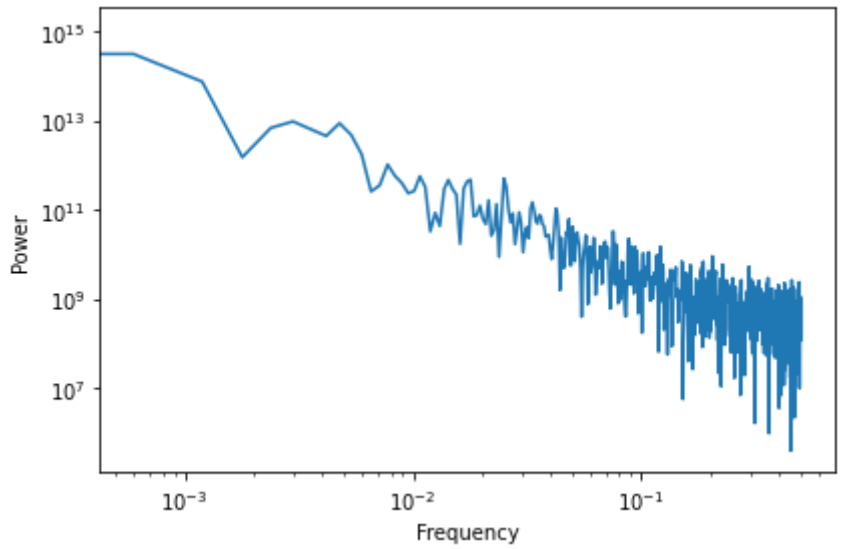
In [15]:

```
btc_spectrum = btc_wave.make_spectrum()
btc_spectrum.plot()
decorate(xlabel='Frequency (Hz)', ylabel='Amplitude')
```



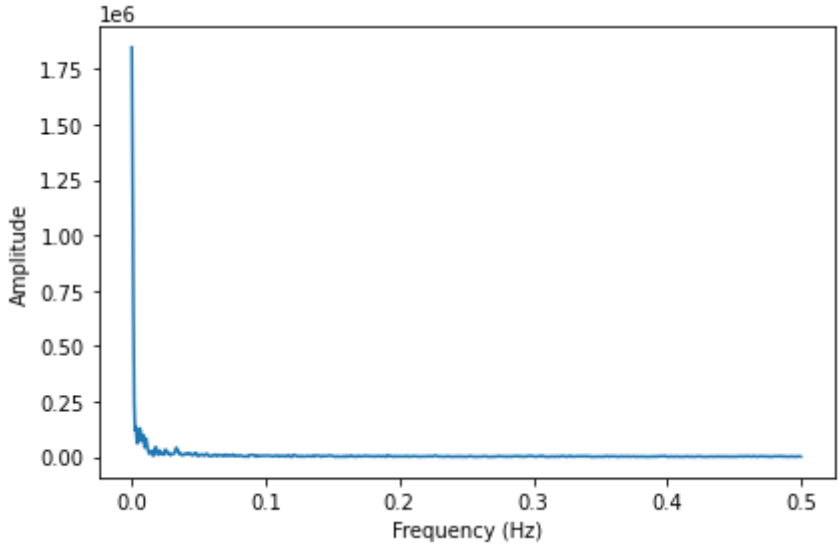
In [16]:

```
btc_spectrum.plot_power()
decorate(xlabel='Frequency',
         ylabel='Power',
         **loglog)
```



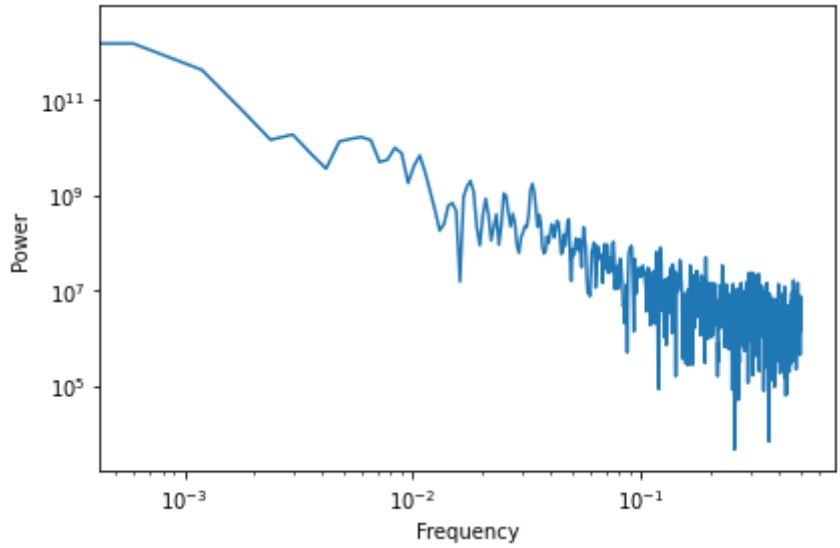
In [17]:

```
eth_spectrum = eth_wave.make_spectrum()
eth_spectrum.plot()
decorate(xlabel='Frequency (Hz)', ylabel='Amplitude')
```



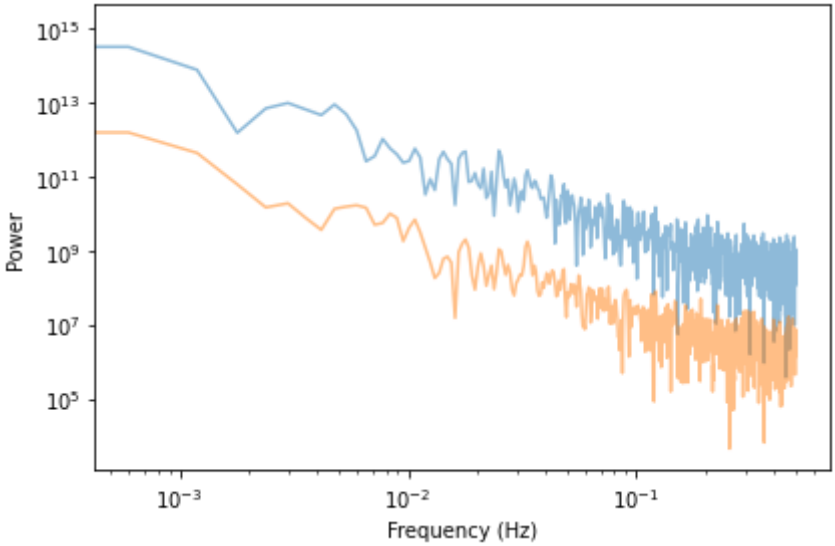
In [18]:

```
eth_spectrum.plot_power()
decorate(xlabel='Frequency',
         ylabel='Power',
         **loglog)
```



In [19]:

```
btc_spectrum.plot_power(alpha=0.5)
eth_spectrum.plot_power(alpha=0.5)
decorate(xlabel='Frequency (Hz)',
         ylabel='Power',
         **loglog)
```



看起來是很相像的，斜率都接近於-1

接著使用了chap05的相關係數python檔案

In [20]:

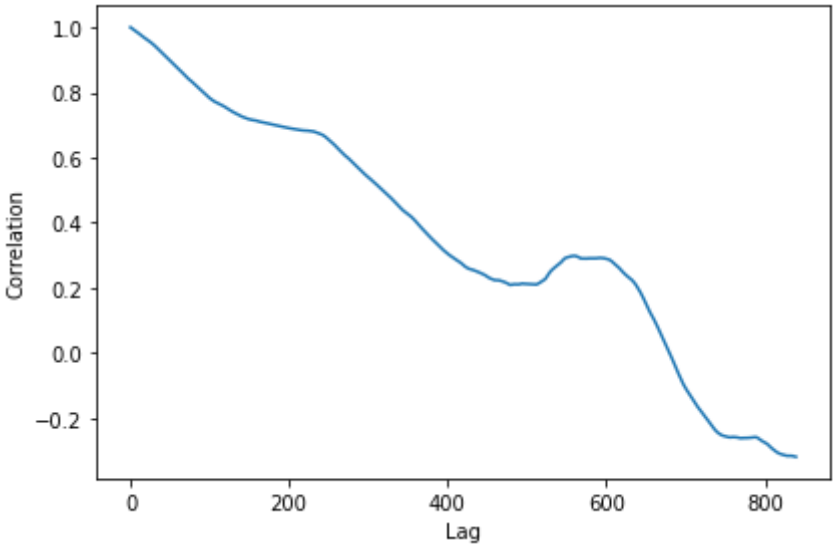
```
def autocorr(wave):
    """Computes and plots the autocorrelation function.

    wave: Wave
    """
    lags = np.arange(len(wave.ys)//2)
    corrs = [serial_corr(wave, lag) for lag in lags]
    return lags, corrs
```

In [21]:

```
def serial_corr(wave, lag=1):
    """Computes serial correlation with given lag.

    wave: Wave
    lag: integer, how much to shift the wave

    returns: float correlation coefficient
    """
    n = len(wave)
    y1 = wave.ys[lag:]
    y2 = wave.ys[:n-lag]
    corr_mat = np.corrcoef(y1, y2)
    return corr_mat[0, 1]
```

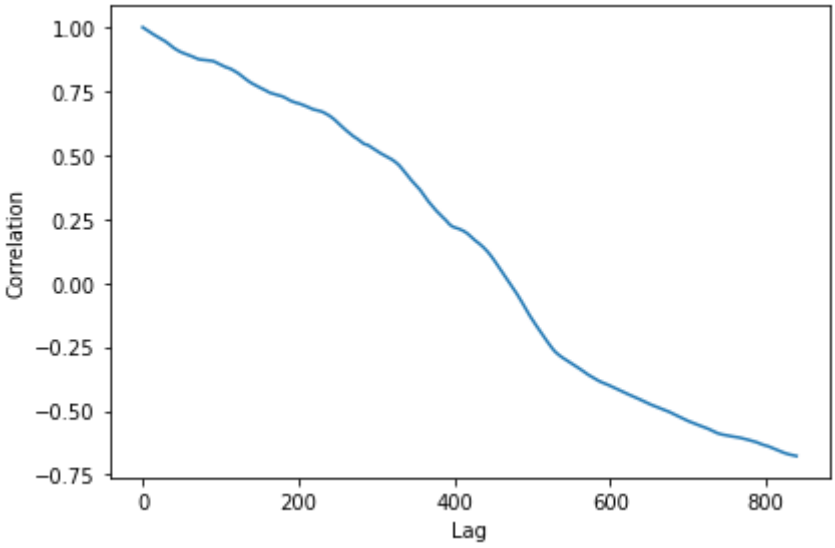
In [22]:

```
btc_lags, btc_corrs = autocorr(btc_wave)
plt.plot(btc_lags, btc_corrs)
decorate(xlabel='Lag',
         ylabel='Correlation')
```



In [23]:

```
eth_lags, eth_corrs = autocorr(eth_wave)
plt.plot(eth_lags, eth_corrs)
decorate(xlabel='Lag',
         ylabel='Correlation')
```



這些都是拿數據的部分當訊號

但我同時也學了些python工具，來弄點技術分析的東西

In [24]:

```
from scipy.stats import linregress
```

先弄個btc

In [25]:

```
btc_reg_up=linregress(x=btc_ts,y=btc_ys)
btc_reg_up
```

Out[25]:

```
LinregressResult(slope=28.291489603548364, intercept=-3607.0841068027585, rv
alue=0.7733433930182585, pvalue=0.0, stderr=0.5660290217150834, intercept_st
derr=549.1006667087629)
```

In [26]:

```
btc_reg_up.slope
```

Out[26]:

```
28.291489603548364
```

In [27]:

```
btc_reg_up[0]
```

Out[27]:

```
28.291489603548364
```

In [28]:

```
btc_up_line=btc_reg_up[1]+btc_reg_up[0]*btc.index
btc_up_line
```

Out[28]:

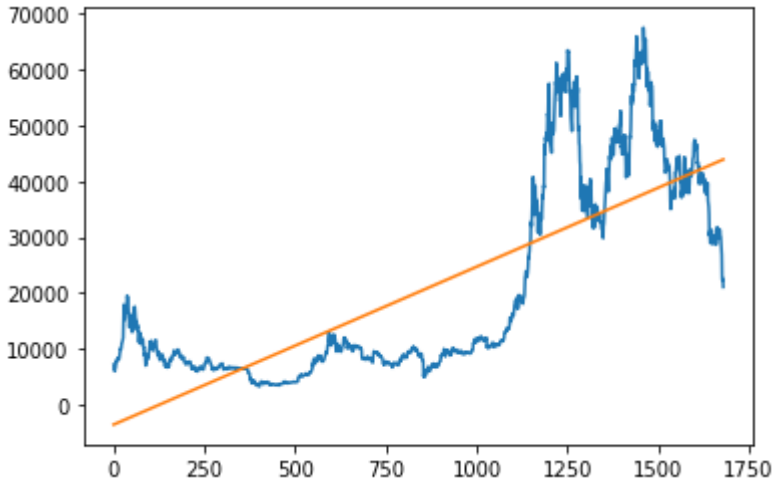
```
Float64Index([-3607.0841068027585, -3578.7926171992103, -3550.5011275956617,
              -3522.2096379921136,  -3493.918148388565,  -3465.626658785017,
              -3437.3351691814682,   -3409.04367957792, -3380.7521899743715,
              -3352.4607003708234,
              ...
                43667.99502072656,   43696.28651033011,   43724.57799993365,
                 43752.8694895372,   43781.16097914075,    43809.4524687443,
                43837.74395834785,    43866.0354479514,   43894.32693755494,
                43922.61842715849],
             dtype='float64', length=1681)
```

In [29]:

```
btc_wave.plot()
plt.plot(btc_up_line)
```

Out[29]:

```
[<matplotlib.lines.Line2D at 0x20d867ff7c0>]
```



如此即可劃出趨勢線，參考來源
https://github.com/pyinvest/quant_basic_toturial/blob/master/quant/11_Trend_line.ipynb
(https://github.com/pyinvest/quant_basic_toturial/blob/master/quant/11_Trend_line.ipynb)