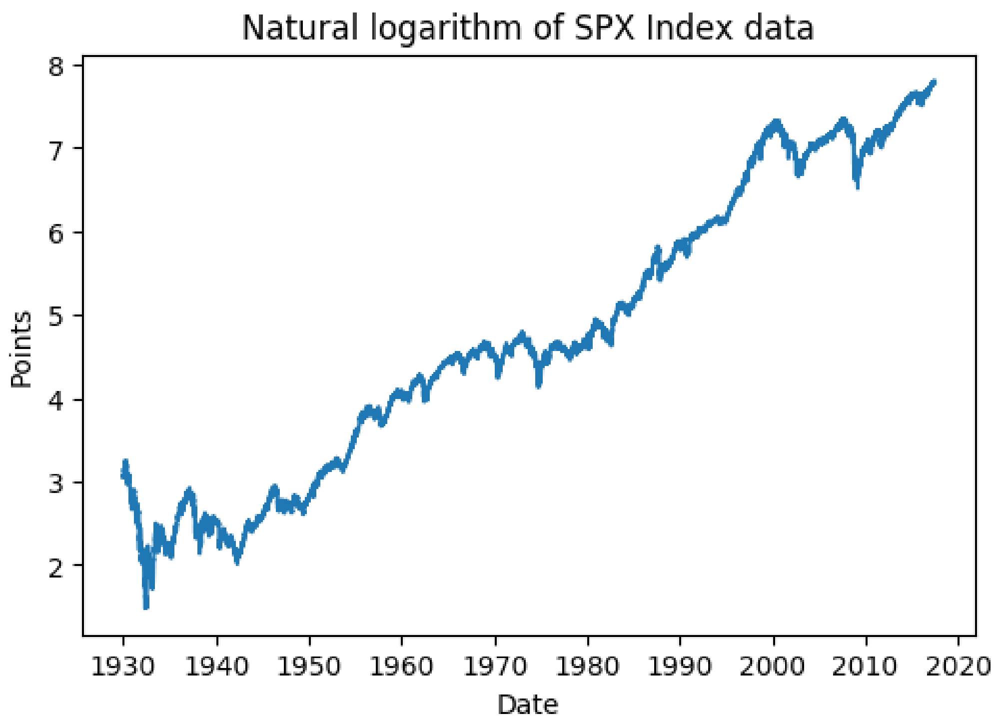
Contents

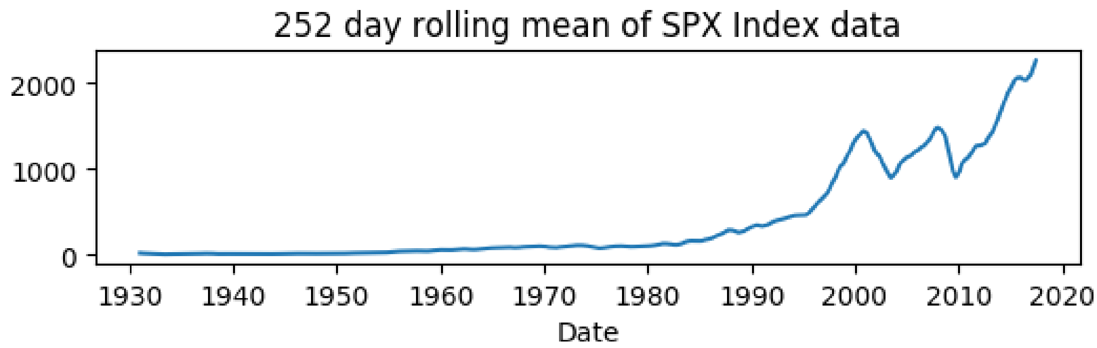
1 Q1 Regression Methods

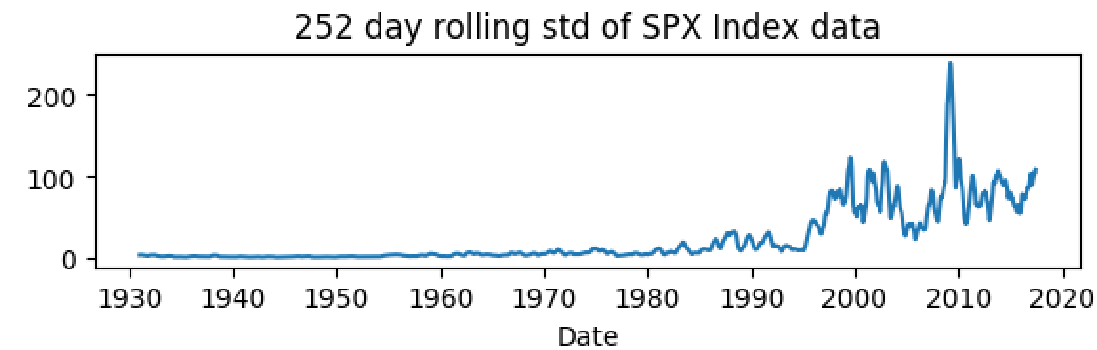
1.1 Processing stock price data in Python

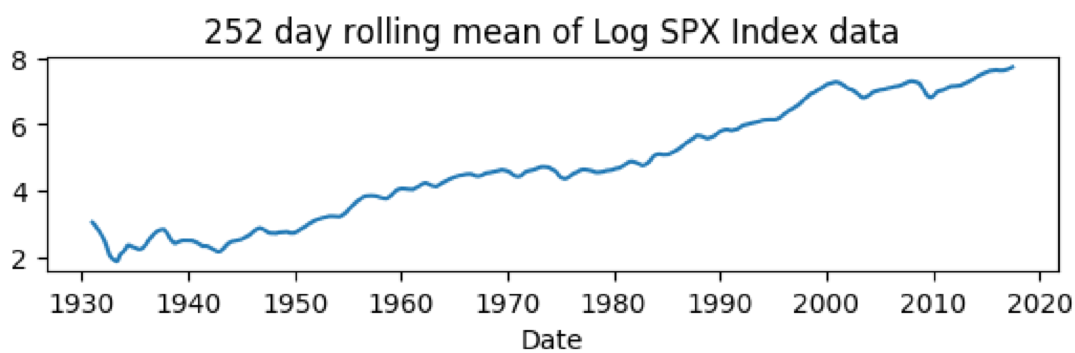
1.1.1 Natural-Log

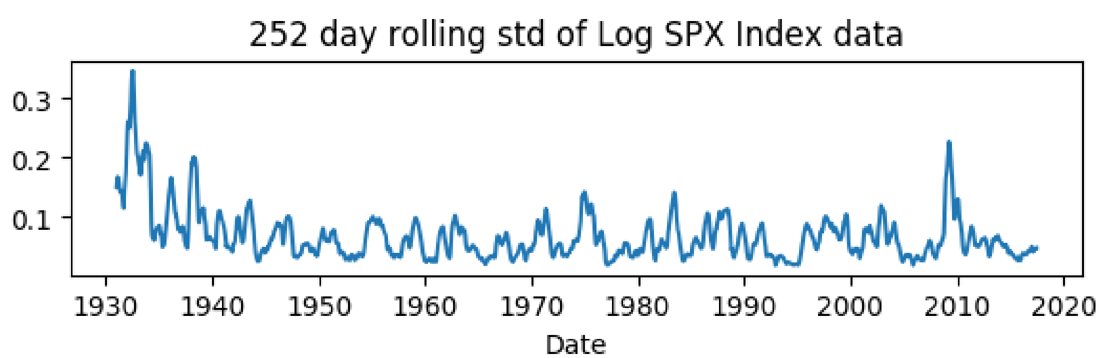


1.1.2 Stationarity









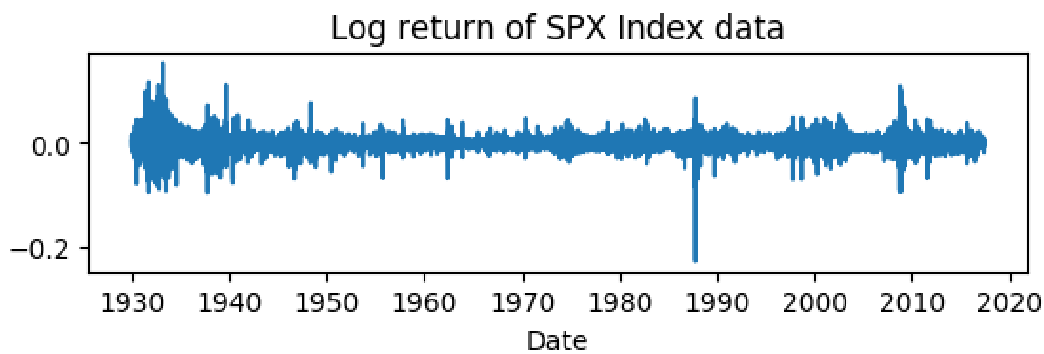
Stationarity of price time-series The rolling mean for both price and log price shows that the mean price trends upwards. Neither of these time-series are stationary.

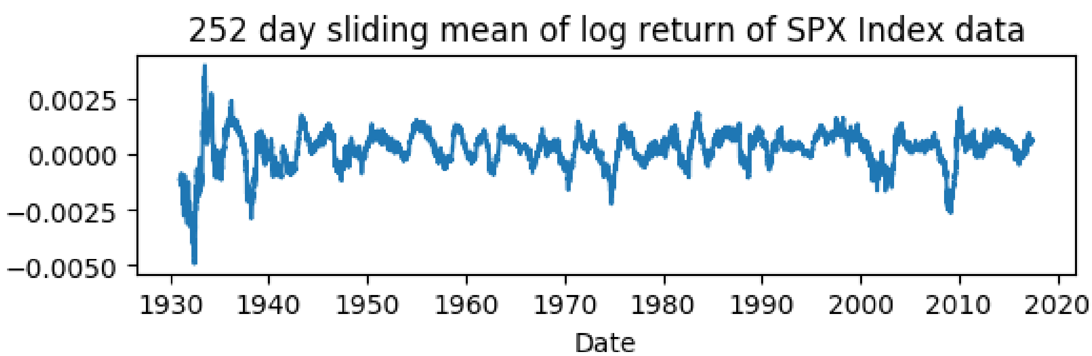
The rolling std for price also trends upwards, but the log price std does not exhibit a clear, positive trend. The simple price rolling std is susceptible to the exponential growth of the price. The same percentage change in 2019, would produce a larger variance value than that same percentage change in 1940. So the rolling std of the price is not stationary.

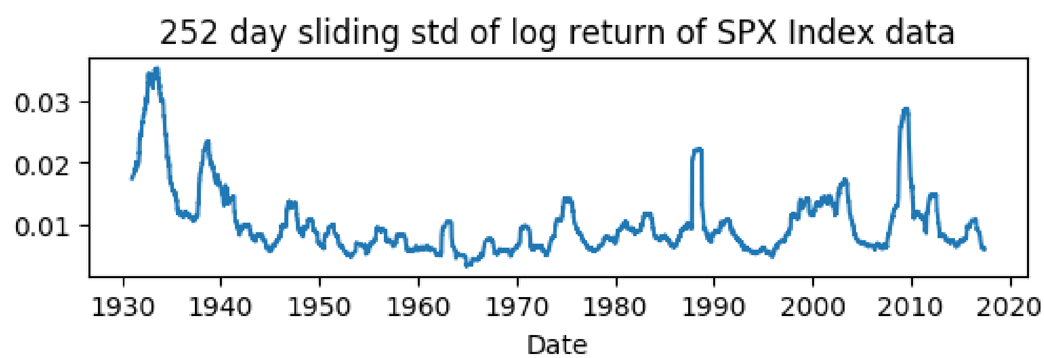
The rolling std of the log price is obtained from the linearly increasing log price. This makes is less susceptible to the trend of the log price time-series, and is stationary.

1.1.3 Returns

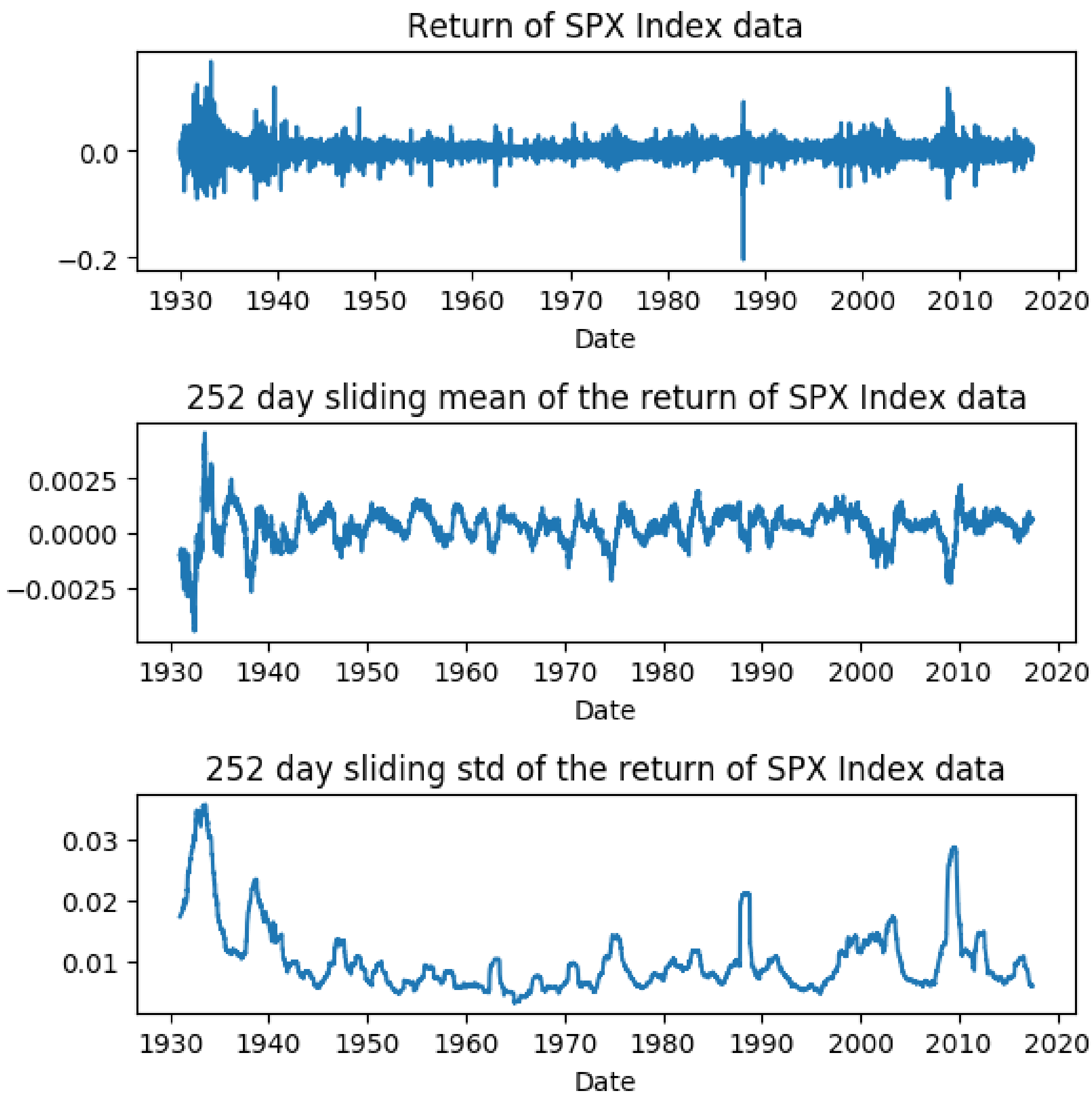
Sliding statistics of log returns







Sliding statistics of simple returns



The rolling mean of both the log return and simple return are stationary, unlike the rolling means of the log price and price.

This makes the time-series more suitable for analysis as it removes trends and allows for better comparison to other time-series.

1.1.4 Log versus Simple returns (i)

Suitability of log returns over simple returns for SP purposes: If we assume that prices are log normally distributed, then the log returns would be normally distributed. This property is useful for when normality is assumed in statistics.

When returns are very small, the log returns are very close in value to the simple returns. Small returns are common for trades holding the asset for a short duration, e.g. one day / daily returns.

Log returns can be decomposed into the difference between two logs. So the compound return over n time periods can be done in O(1) time. The sum of normal values is normal, but the product is not. So the compound return obtained via log returns is also normally distributed.

Data	JB p-value
log price	0
log returns	0
simple returns	0

The Jarque-Bera test returned 0 for the log returns, thus rejected the null hypothesis. This implies that the original price data was not log normally distributed. However, over shorter period of time, the prices should be log normally distributed. So the theoretical advantages of using simple returns would not apply to this SPX time-series.

1.1.5 Log versus Simple returns (ii)

Day	Simple Return
1	0
2	1
3	-0.5

```
The simple returns give a total return of: 0.5
```

Day	Simple Return
1	0.000
2	0.693
3	-0.693

```
The log returns give a total return of: 0.0
```

This example shows that log returns are symmetric, and so the compound log returns can be calculated with just the initial and final prices.

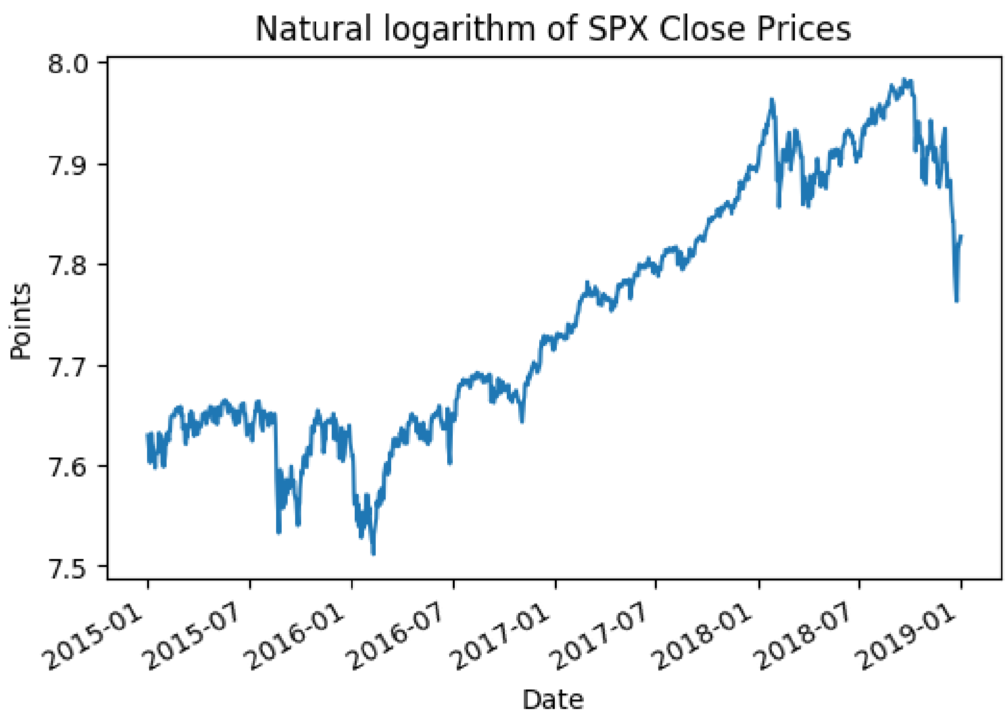
1.1.6 Log versus Simple returns (iii)

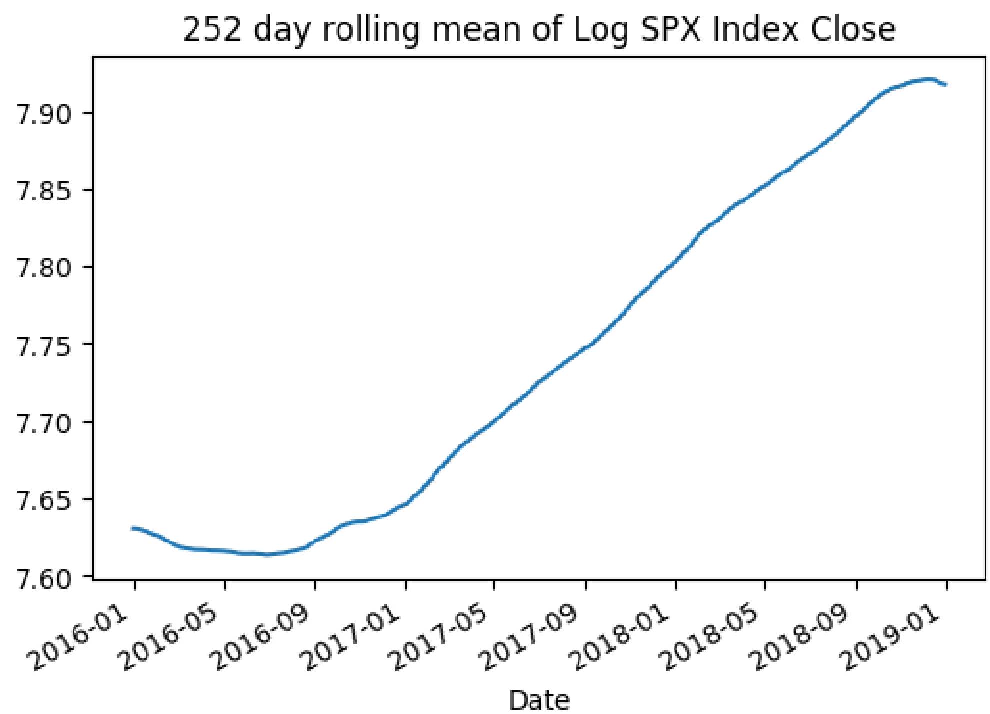
Log returns should not be used over long periods of times, as the assumption of log-normallity is unrealistic.

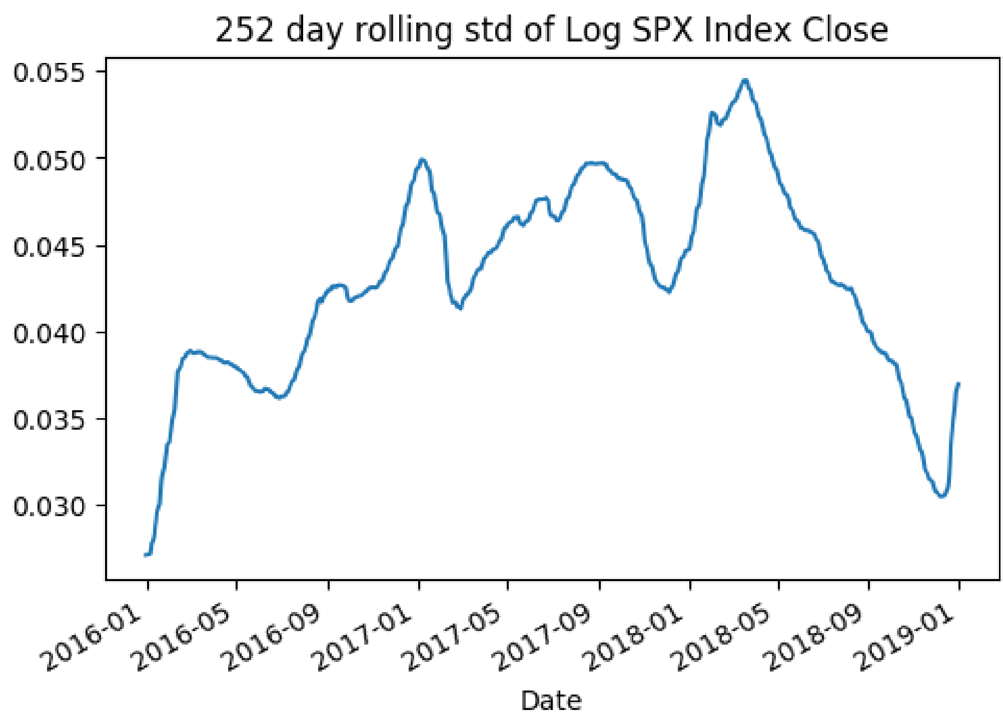
Log returns are not linearly additive across assets, and so should not be used when dealing with multi-asset portfolios.

1.2 ARMA vs ARIMA Models for Financial Applications

1.2.1 Suitability of ARMA and ARIMA

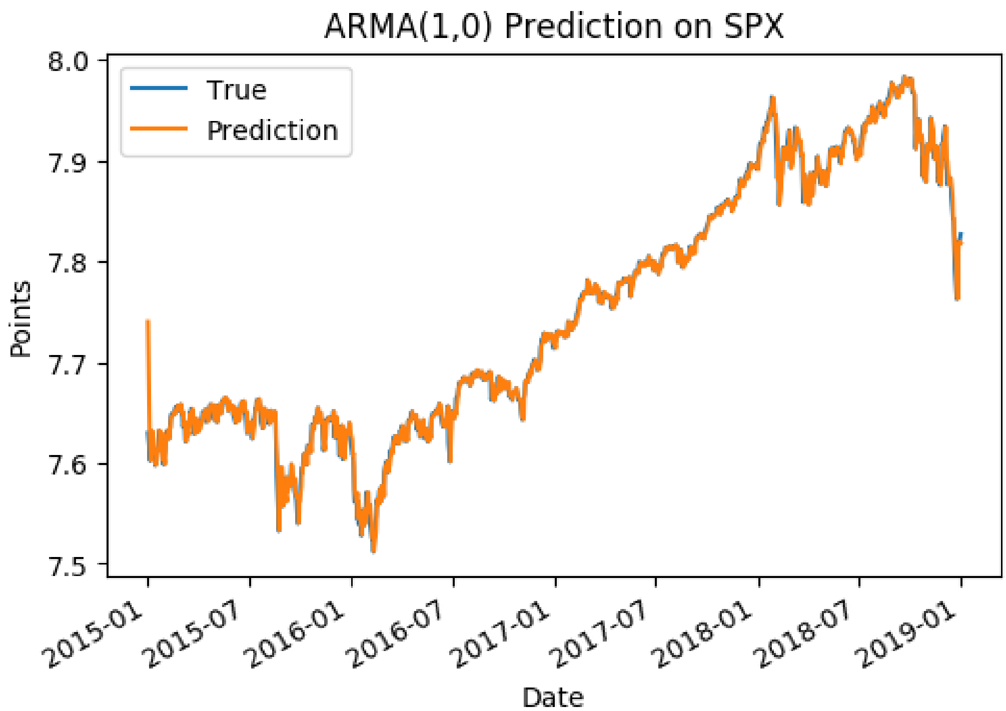



252 day rolling mean of Log SPX Index Close
7.60
7.65
7.70
7.75
7.80
7.85
7.90
2016-01
2016-05
2016-09
2017-01
2017-05
2017-09
2018-01
2018-05
2018-09
2019-01
Date


252 day rolling std of Log SPX Index Close
0.030
0.035
0.040
0.045
0.050
0.055
2016-01
2016-05
2016-09
2017-01
2017-05
2017-09
2018-01
2018-05
2018-09
2019-01
Date

The rolling mean shows that the time-series is not stationary, so ARMA not suitable. and so ARIMA is more appropriate.

1.2.2 ARMA




```
SSE of ARMA(1,0) Predictions: 0.08675908609244501
```

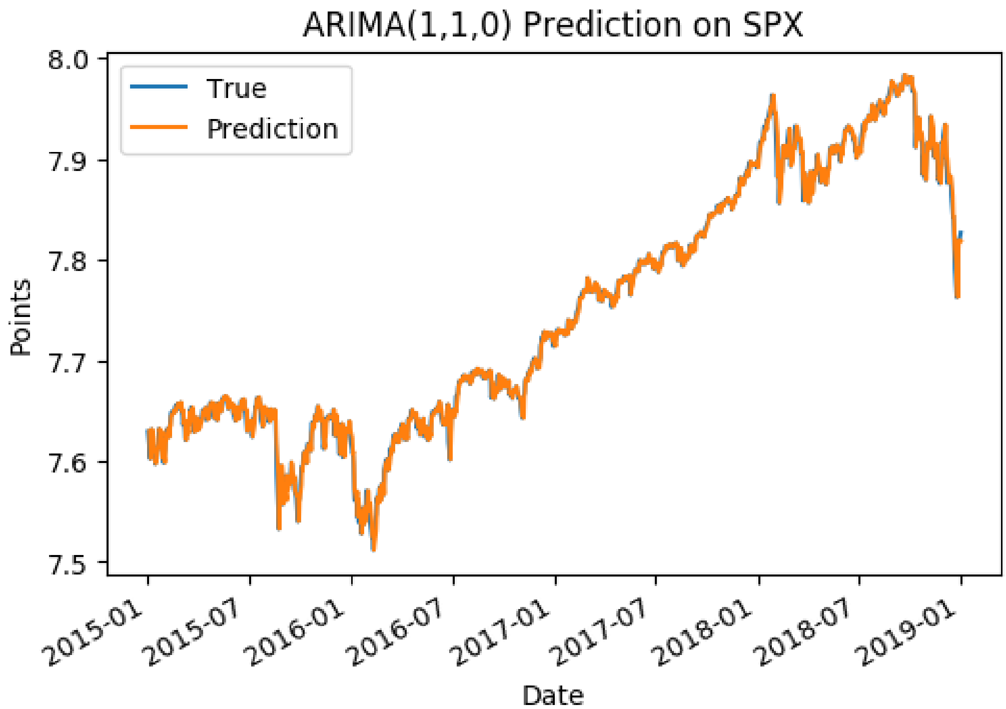
The ARMA(1,0) model appears to perform well at predicting this time-series.

The model predicts the next days prices as the current days price, with an additional amount of small, random noise.

This is effectively a random walk, which can be viable for very short term predictions, shown by the low SSE of the model.

In practice, this is not that useful unless the stock is only held for a day. Since the model heavily relies on the trend, predictions for prices further than one day away will have this trend bias along with compounding errors.

1.2.3 ARIMA




```
SSE of ARIMA(1,1,0) Predictions: 0.0746545241081268
```

The ARIMA(1,1,0) model produced predictions with a lower SSE.

ARIMA 'detrends' the data, and the reliance on trends is minimal compared to ARMA(1,0). This makes the ARIMA more suitable for predicting returns further than a day out.

1.2.4 Log-Prices for ARIMA

The log prices have the symmetric property. This is important for the initial differencing step in ARIMA to correctly remove the trends. Otherwise, the prices are exponential and the initial differencing step would fail.

1.3 Vector Autoregressive (VAR) Models

1.3.1 Concise VAR

VAR can be represented as

$$\mathbf{Y} = \mathbf{BZ} + \mathbf{U} \quad (1.3.11)$$

where:

$$(1.3.12)$$

$$\mathbf{Y} = [\mathbf{y}[p]][\mathbf{y}[p+1]]...[\mathbf{y}[T]]$$

$$\mathbf{B} = [\mathbf{c}\mathbf{A}_1\mathbf{A}_2 ... \mathbf{A}_p]$$

$$\mathbf{Z} = \begin{bmatrix} 1 & 1 & \cdots & 1 \\ \mathbf{y}[p-1] & \mathbf{y}[p] & \cdots & \mathbf{y}[T-1] \\ \mathbf{y}[p-2] & \mathbf{y}[p-1] & \cdots & \mathbf{y}[T-2] \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{y}[0] & \mathbf{y}[1] & \cdots & \mathbf{y}[T-p] \end{bmatrix}$$

$$\mathbf{U} = [\mathbf{e}[p][\mathbf{e}[p+1]...[\mathbf{e}[T]]$$

1.3.2 Optimal VAR Coefficients

Using the LSE estimator on the model, the cost function is given by:

$$J(B) = (Y - BZ)^T (Y - BZ) \quad (1.3.21) \tag{1}$$

when expanded, it turns into

$$J(B) = YY^T - 2Y^T BZ + Z^T B^T BZ \quad (1.3.22) \tag{2}$$

The aim is to minimize the cost function:

$$\frac{J(B)}{dB} = 0 \quad (1.3.23) \tag{3}$$

This results in:

$$-2YZ^T + B_{opt}(ZZ^T + ZZ^T) = 0 \quad (1.3.24) \tag{4}$$

$$-2YZ^T + 2B_{opt}ZZ^T = 0 \quad (1.3.25) \tag{5}$$

Finally, the optimal parameters are given by:

$$B_{opt} = YZ^T(ZZ^T)^{-1} \quad (1.3.26) \tag{6}$$

1.3.3 Eigenvalues of VAR

For the system to be stable the eigenvalues of the matrix A must be smaller than one in absolute value. This is proven by taking the Z-transform:

$$Y(z) = AY(z)z^{-1} + E(z) \quad (1.3.31) \tag{7}$$

$$Y(z)(I - Az^{-1}) = E(z) \tag{8}$$

$$H(z) = (I - Az^{-1})^{-1} \tag{9}$$

$$A = QDQ^{-1} \Rightarrow H(z) = (I - QDQ^{-1}z^{-1})^{-1} \quad (1.3.32) \tag{10}$$

$$\text{where } A \text{ has been diagonalized.} \tag{11}$$

$$\text{For the system to be stable, there must be no poles. This happens when} \tag{12}$$

$$|\lambda| < 1 \tag{13}$$

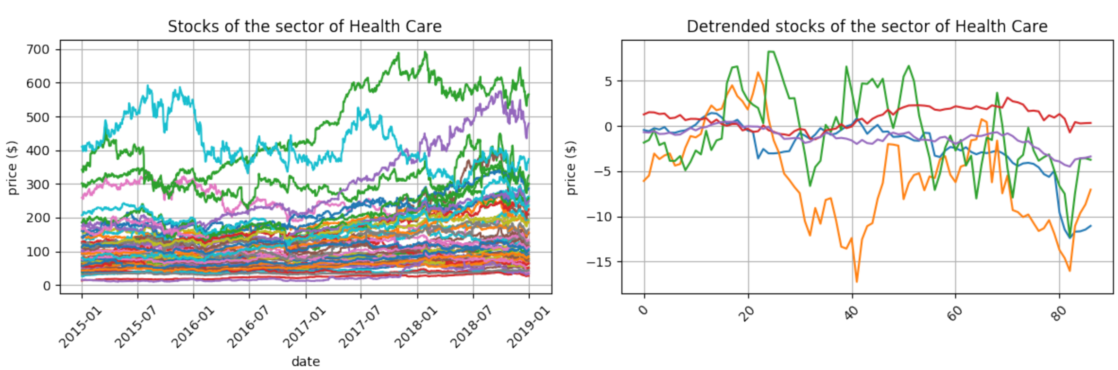
1.3.4 Portfolio Analysis with VAR (i)

```
eigenvalue magnitude

               0.726

               0.726

               1.006

               0.861

               0.911
```

It would not make sense to construct a portfolio using these stocks as the third eigenvalue has a magnitude greater than 1. This means that the VAR(1) process is not stable here.

1.3.5 Portfolio Analysis with VAR (ii)

```
Chosen Sector: Health Care
```




```
Largest eigenvalue Magnitude of VAR(1) in health sector: 0.9941531205433555
```

```
 sector                          largest eigenvalue magnitude
```

Industrials	0.992
Health Care	0.994
Information Technology	0.993
Communication Services	0.982
Consumer Discretionary	0.991
Utilities	0.986
Financials	1.004
Materials	0.992
Real Estate	0.983
Consumer Staples	0.992
Energy	0.986

Constructing a portfolio that consists of stocks from just one sector isn't advisable unless the investor is OK with the additional risk carried. Any negative effects to the industry would cause the majority of stocks to lose value. However, the advantage of investing in the sector instead of a single company is that is is less risky. A single company can lose value whilst the overall sector retains a positive or neutral outlook.

It is useful to analyse sectors as it can show stocks that outperform the rest of the industry. Also the sector wide analysis is less sensitive to individual stock fluctuations, which can make analysis more insightful.

Additionally, the largest eigenvalue magnitude for the Health sector was under 1. All sectors, apart from the financials sector also had a largest eigenvalue magnitude of under 1. This supports that a portfolio could constructed from the sector as the VAR(1) of that portfolio would be stable.

2 Q2 Bond Pricing

2.1 2.1 Examples of bond pricing

2.1.1 Effective Rates

An investor receives USD 1,100 in one year in return for an investment of USD 1,000 now. Calculate the percentage return per annum with: a) Annual compounding, b) Semiannual compounding, c) Monthly compounding, d) Continuous compounding

compounding type	return per annum (%)
annual	10.000
semiannual	9.762
monthly	9.569
continuous	9.531

2.1.2 Equivalent Rates (i)

What rate of interest with continuous compounding is equivalent to 15% per annum with monthly compounding?

```
14.91% continuous compounding is equivalent to 15% per annum with monthly
compounding
```

2.1.3 Equivalent Rates (ii)

A deposit account pays 12% per annum with continuous compounding, but interest is actually paid quarterly. How much interest will be paid each quarter on a USD 10,000 deposit?

We calculate the continuous compounding interest rate:

[38]:
```
cont_rate = math.log(1.12)
```

```
continuous compounding rate : 11.333%
```

[39]:
```
init_deposit = 10000
# calc account value after each quarter
first_quarter  = init_deposit   * math.exp(cont_rate/4)
second_quarter = first_quarter  * math.exp(cont_rate/4)
third_quarter  = second_quarter * math.exp(cont_rate/4)
fourth_quarter = third_quarter  * math.exp(cont_rate/4)
```

quarter	interest paid ($)

```
first                  287.37

second                 295.63

third                  304.13

fourth                 312.87
```

2.2 Forward rates

Suppose that the one–year interest rate, r1 is 5%, and the two–year interest rate, r2 is 7%. If you invest USD 100 for one year, your investment grows to 100×1.05 = USD105; if you invest for two years, it grows to 100×1.072 = USD114.49. The extra return that you earn for that second year is 1.072/1.05 - 1 = 0.090, or 9.0 %

- The decision to invest for two years instead of one depends on how risk-averse the investor is and how long they can comfortably invest for. If a two year period is suitable and the bond has a low risk (i.e. developed country Government bond), then the two year investment seems suitable, especially if it beats inflation.
- The 5% strategy pays out the quickest, allowing the investor to re-evaluate after one year. This is suitable if the investor would like to invest in other assets after one year. The 7% strategy has the largest return over the two year period, but requires capital to be locked away the longest. If the investor has no desire to hold other assets over the period, this is the ideal strategy, assuming the bond has very low risk. The 9% strategy, like the 5%, only requires capital to be tied down for one year, but it is tied down in the second year.
- Whilst providing a higher return compared to 5%, the investor must have the agreed initial capital available after one year to loan. So while free to invest elsewhere in the first year, there is an additional risk of not having the funds when required. This risk is created when investing in other assets in the first year, with the aim of beating 5% for the year.
- If investing for one year at 5%, then a further 9% is required if reinvesting over the second year, to match the two year investment. The forward rate implies that after one year, the new one year rate would be 9%. However this is unlikely to hold after one year as the one-year and two-year rates are likely to change before then.

2.3 Duration of a coupon-bearing bond

2.3.1 Duration

The duration of the 1% bond in Table is found by summing the (Year x Fraction of PV) terms

```
The duration is 6.76 years
```

2.3.2 Modified Duration

```
The modified duration is 6.44
```

This measures how senstitive the bond price is with respect to the yield, while the duration is the weighted average of the times to each of the cash payments.

2.3.3 Sensitivity Analysis

Using the duration allows for immunization (sensitivity analyses). A first order immunization using two bonds can allow a pension plan to meet a price P in time D using the following:

$$P = x_1 P_1 + x_2 P_2 \quad (2.1.31)$$

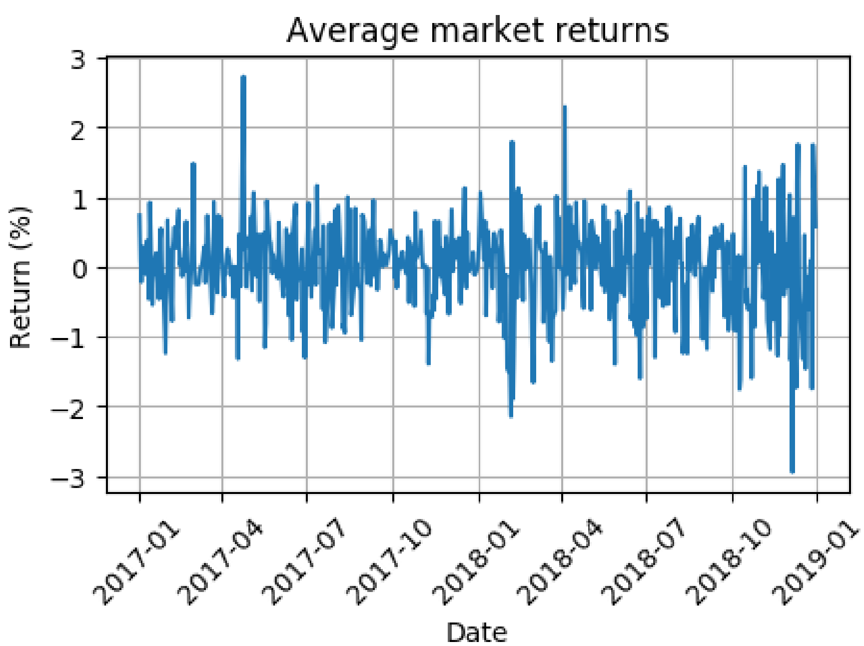
$$D = \frac{x_1 P_1}{D_1} + \frac{x_2 P_2}{D_2} \quad (2.1.32)$$

Where $x_Y, P_Y and D_Y$ are the portfolio weighting, bond price and bond duration respectively, for bond Y.

This helps to reduce the impact on the pension portfolio against unexpected changes in interest rates (which would affect the prices of the two individual bonds).

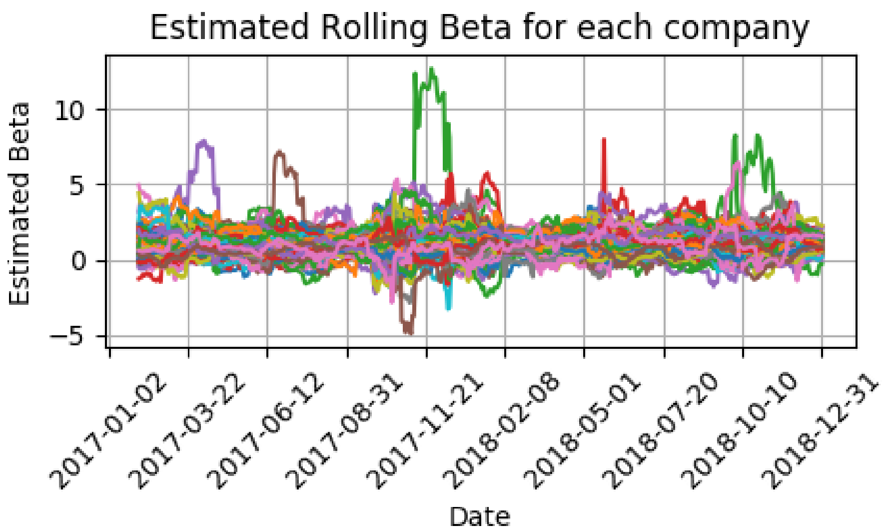
2.4 Capital Asset Pricing Model (CAPM) and Arbitrage Pricing Theory (APT)

2.4.1 Market returns per day




```
The average market return was 0.0000460% with a std of 0.00658%
```

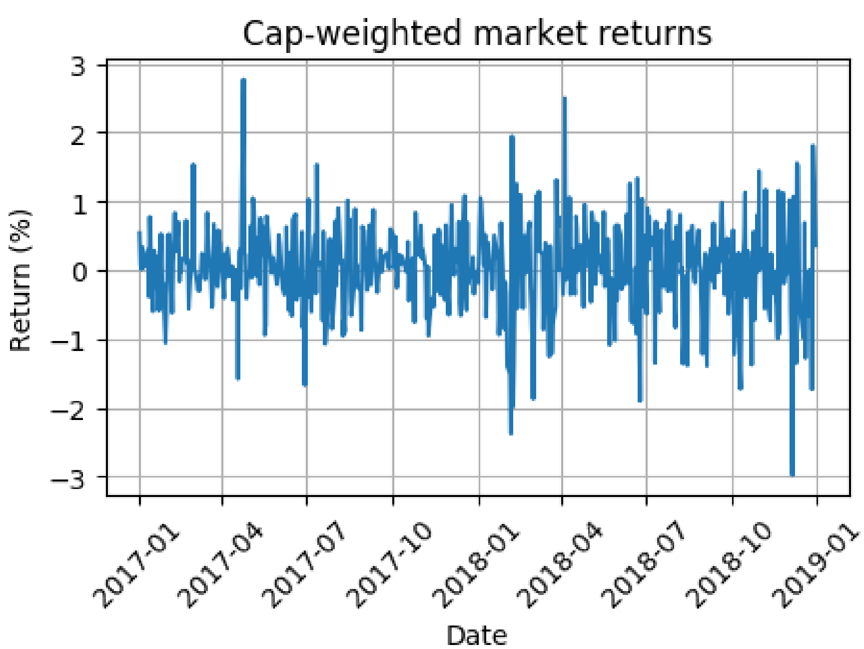
2.4.2 Rolling beta




```
The average Beta for all companies was 0.983 with a an average std of 0.550
```

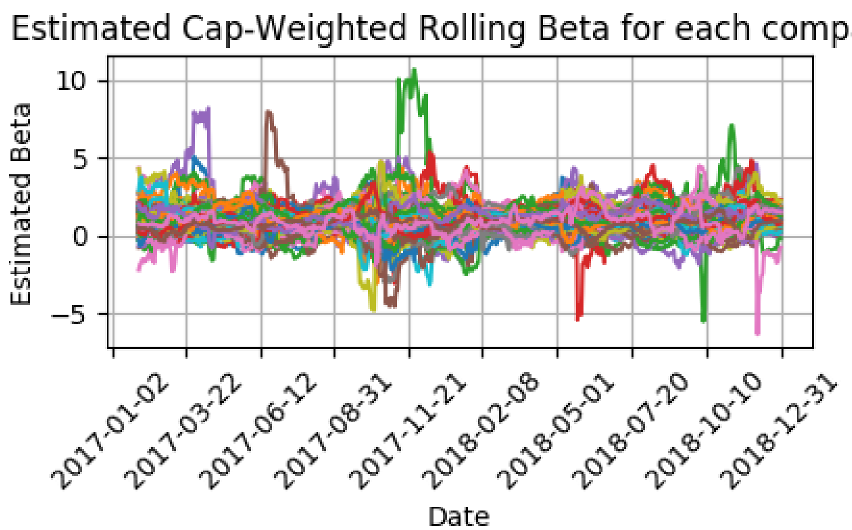
Volatility of Rolling Beta Most rolling beta can be seen above to be volatile, and so cannot be assumed to be constant. This is supported by the std of the mean Beta per company, being close in value to the mean of mean Beta per company.

2.4.3 Market-cap weighted returns



Weighting coefficient This coefficient is the total market capitalization during that day, assuming the stocks used are all stocks in the market being studied.

2.4.4 Cap-weighted Beta




```
The average cap-weighted Beta for all companies was 0.938 with a an average std
of 0.562
```

Comparison to equally-weighted Betas The cap-weighted Betas are also volatile, but have a lower magnitude compared to the equally-weighted, (0.938 mean vs 0.983)

High beta values have decreased in magnitude, which can be observed from the peaks of both plots. This suggests the corresponding companies were being over weighted in the equal weighting. Large Beta suggests that company drives/correlates with the market. If the Beta decreases then that company was contributing more to the market return than it's market capitalization would allow in cap-weighted. This highlights the importance of using cap-weighted Betas when assessing individual companies Betas for investment decisions.

2.4.5 Arbitrage Pricing Theory (APT) for a two factor model

For this exercise these factors are the cap-weighted market returns and the small minus big (SMB). The sensitivity to SMB is taken to be the natural logarithm of the market value of the asset. The sensitivity to the market returns are the cap-weighted Betas.

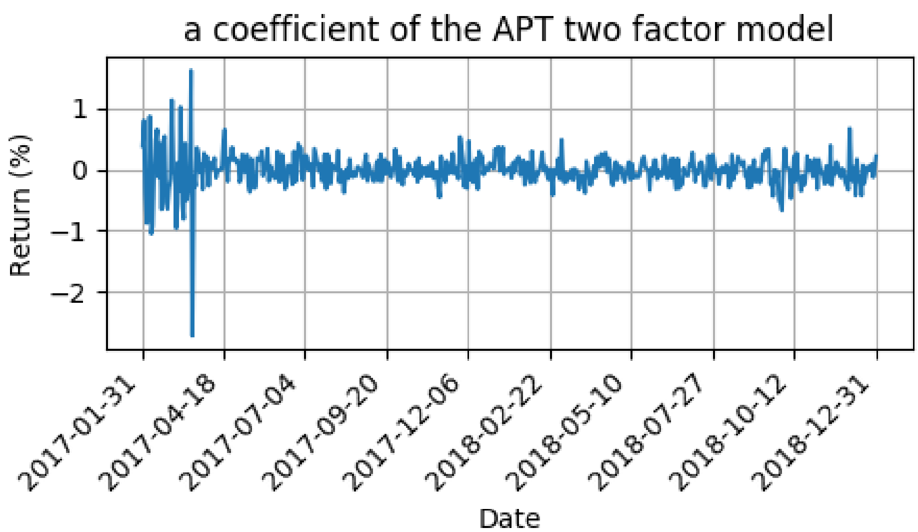
Model: $r_i = a + b_{m_i} R_m + b_{s_i} R_s + \epsilon_i$

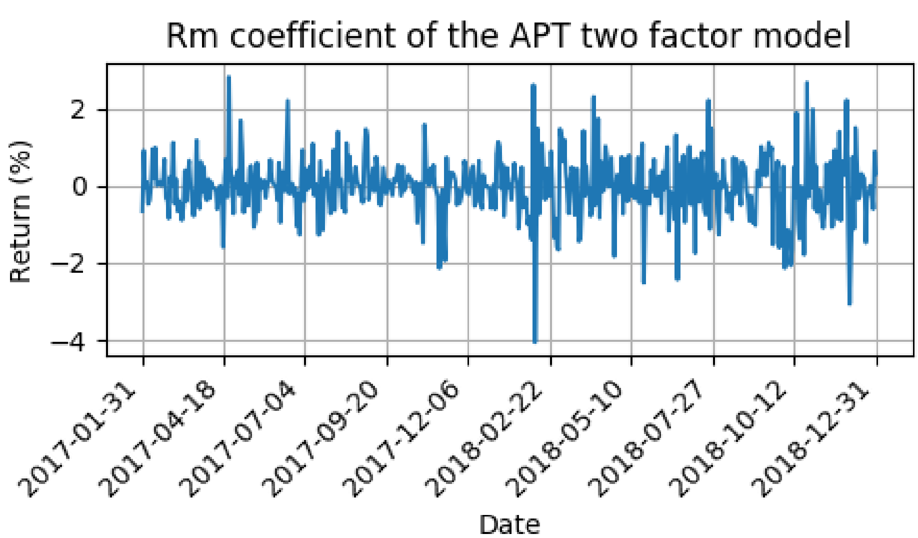
Q2.4.5 a)

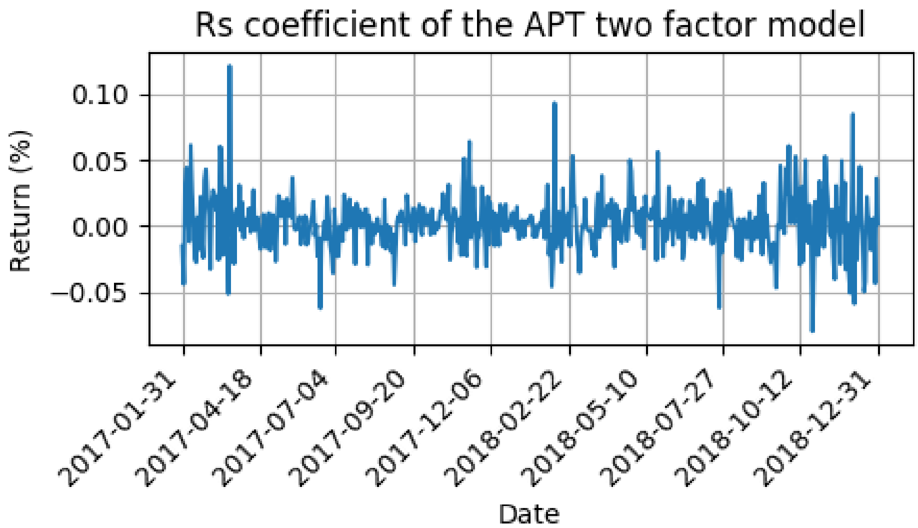
[55]:

```
num_days = len(betas) - WINDOW + 1
num_stocks = 157

# specific return
e  = np.zeros([num_days, num_stocks])
# coeffecients - per day
rs = np.zeros([num_days])
rm = np.zeros([num_days])

# sensitivities
bs = np.zeros([num_stocks, num_days])
bm = np.zeros([num_stocks, num_days])

# independent term
a  = np.zeros([num_days])

# Y
ri = np.zeros([num_stocks, num_days])

# Fill above arrays
df1 = df.replace([np.inf, -np.inf], np.nan)
df1 = df.fillna(0)
i = 0
for date, stock in df1['mcap'].iteritems():
    stock_ = stock.replace([0], 1) # log(1) = 0
    bs[i] = np.log(stock_)[21:]
    #bs[i][bs[i] == -np.inf] = 0
    i += 1
i = 0
betas_ = betas.replace([np.inf, -np.inf], np.nan)
betas_ = betas_.fillna(0)
for date, stock in betas_.iteritems():
    bm[i] = stock[21:]
    i += 1
i = 0
for date, stock in df1['ret'].iteritems():
    ri[i] = stock[21:]
    i += 1

# Prep for Linear Regression
X = np.zeros([num_days, num_stocks, 2])
for t in range(num_days):
    for i in range(num_stocks):
        X[t,i,0] = bm[i,t]
        X[t,i,1] = bs[i,t]
Y = ri.T.reshape([num_days, num_stocks])
```

```
for t in range(num_days):
    reg = LinearRegression().fit(X[t,:,:],Y[t,:])
    y_pred = reg.predict(X[t,:,:])
    e[t,:] = Y[t,:] - y_pred
    a[t] = reg.intercept_
    rm[t] = reg.coef_[0]
    rs[t] = reg.coef_[1]
```







Q2.4.5 b)

parameter	mean (%)	stddev
a	-0.02311	0.27715
Rm	-0.03306	0.78403
Rs	0.00254	0.02126
|a|	0.17948	0.21245
|Rm|	0.56310	0.54655
|Rs|	0.01517	0.01511

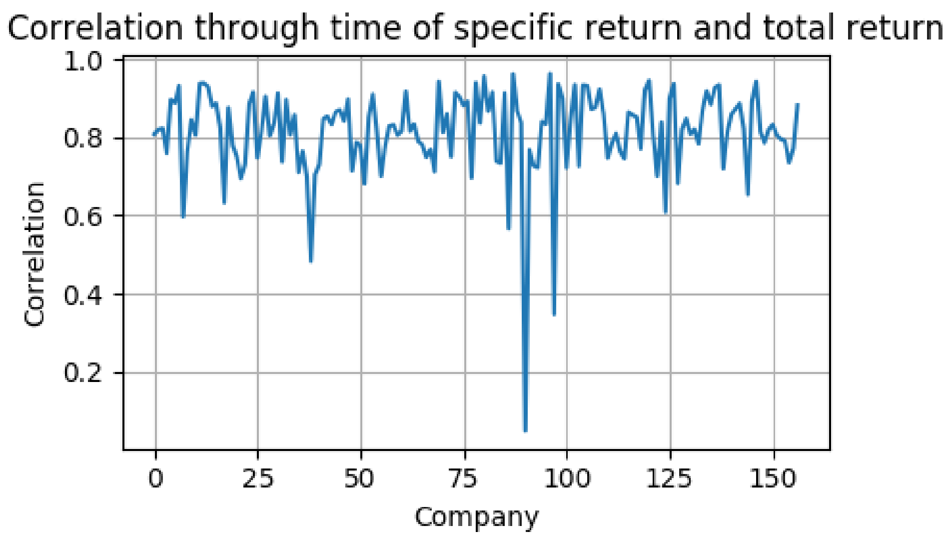
The mean magnitude of Rm is about 0.6%, whereas the mean magnitude of Rs is around 0.015%, or over 30 times smaller. This means that the market return is more significant than the exposure to size for daily returns. A also has a significantly larger mean magnitude compared to Rs, just over 10 times larger.

Rm also has a very large variance. This can be explained by there being days where prices change significantly and there being many days where prices remain at similar levels. Essentially, the market is volatile inter-day.

Rs has a vary low variance. By taking the log of the market cap, this factor has been made less sensitive to changes and so the inter-day variation is low.

The value of 'a' mainly varies between 1% and -1%, with a mean of approximately 0 and a stddev of 0.277. This term can be considered as the risk-free rate for that day. The negative values can be viewed as the market pricing the rate as negative, as seen with German government bonds. If analysing data from the USA, over the same period, 'a' should have a larger positive mean, due to US government bonds having a higher yield.

Q2.4.5 c)



Q2.4.5 d)

```
The covariance matrix of R is
[[ 6.15936602e-05 -8.05737998e-07]
 [-8.05737998e-07  4.52932279e-08]]
```

```
The eigen values of the covariance matrix are
[6.16042064e-05 3.47470098e-08]
```

The covariance between Rm and Rs is $-8.06e^{-7}$.

This is small, approximately zero. This could be due to the small magnitude of Rs, but Rs can be said to have no significant impact on Rm.

This matrix is also stable as the magnitude of the eigen values are less than 1.

Q2.4.5 e)

```
Percentage of the variance explained by the first ten principal components -
Specific Returns Covariance
```

component	perc. variance explained (%)
1	25.744
2	11.505
3	9.949
4	5.978
5	4.391
6	3.799
7	3.148
8	2.709
9	2.498
10	2.115

The first 10 components, out of 157, account for explaining 71.84% of the variance.

The specific return used, deducts the market return and market size factors already. Therefore, the covariance between these returns can show if there exists other factors not already accounted for. By performing PCA, these factors can be extracted out. The first component represents a potential factor that could explain about 26% of the specific returns. This is a significant amount, and so a third factor derived from this, by assessing the covariance has the potential to increase the prediction accuracy of a new, 3 factor APT.

```
Percentage of the variance explained by the first ten principal components -
Daily Returns Covariance
```

component	perc. variance explained (%)
1	35.439
2	19.105
3	7.855
4	6.900
5	3.509
6	2.889
7	2.301
8	1.732
9	1.709
10	1.462

The table above looks at the first few components, when performing PCA on the covariance of the original daily returns. The first two components explain a higher percentage of variance. This supports the idea that the factors used in the APT model account for taking away variance in the specific returns. Namely, the market return and market size factors reduced the variance of the specific returns, leading to the first component of PCA explaining a lower ratio of the variance.

3 Q3 Portfolio Optimization

3.1 Adaptive minimum-variance portfolio optimization

3.1.1 Minimum Variance Portfolio Optimization

By allowing negative weights on assets, or, allowing short selling, the minimum-variance portfolio is given by:

$$min \quad \frac{1}{2}\mathbf{w}^T\mathbf{C}\mathbf{w} \qquad (3.1.11)$$

$$subjectto \quad \mathbf{w}^T\mathbf{1} = 1$$

The minimum-variance portfolio optimization problem can be solved by finding the optimal conditions for the associated Lagrangian function:

$$L(\mathbf{w}, \lambda, \mu) = \frac{1}{2}\mathbf{w}^T\mathbf{C}\mathbf{w} - \lambda(\mathbf{w}\mathbf{1} - 1) \quad (3.1.12)$$

By differentiating w.r.t. $\mathbf{w}$ and λ and assuming that $\mathbf{C}$ is invertible, the following optimality conditions are obtained:

$$\frac{dL}{d\mathbf{w}} = \mathbf{C}\mathbf{w} - \lambda\mathbf{1} = 0 \text{ -> } \mathbf{w} = \lambda\mathbf{C}^{-1}\mathbf{1} \quad (3.1.13)$$

$$\frac{dL}{d\lambda} = \mathbf{w}^T\mathbf{1} - 1 = 0 \text{ -> } \mathbf{w}^T\mathbf{1} = 1 \quad (3.1.14)$$

The set of optimal weights $\mathbf{w}_{opt}$ can be found by solving for λ and substituting:

$$\mathbf{w}^T\mathbf{1} = (\lambda\mathbf{C}^{-1}\mathbf{1})^T\mathbf{1} = \lambda\mathbf{1}^T\mathbf{C}^{-1^T}\mathbf{1} = 1 \quad (3.1.15)$$

$$\text{-> } \lambda = \frac{1}{\mathbf{1}^T\mathbf{C}^{-1^T}\mathbf{1}} \quad (3.1.16)$$

$$\mathbf{w}_{opt} = \lambda\mathbf{C}^{-1}\mathbf{1} = \frac{1}{\mathbf{1}^T\mathbf{C}^{-1^T}\mathbf{1}}\mathbf{C}^{-1}\mathbf{1} \quad (3.1.17)$$

The theoretical variance of the returns using $\mathbf{w}_{opt}$ is given by:

$$\sigma_p^2 = \mathbf{w}_{opt}^T\mathbf{C}\mathbf{w}_{opt} = \frac{1}{\mathbf{1}^T\mathbf{C}^{-1^T}\mathbf{1}}(\mathbf{C}^{-1}\mathbf{1})^T\mathbf{C}\frac{1}{\mathbf{1}^T\mathbf{C}^{-1^T}\mathbf{1}}\mathbf{C}^{-1}\mathbf{1} \quad (3.1.18)$$

$$\sigma_p^2 = \frac{1}{(\mathbf{1}^T\mathbf{C}^{-1^T}\mathbf{1})^2}(\mathbf{C}^{-1}\mathbf{1})^T\mathbf{C}(\mathbf{C}^{-1}\mathbf{1}) = \frac{1}{(\mathbf{1}^T\mathbf{C}^{-1^T}\mathbf{1})^2}\mathbf{1}^T\mathbf{C}^{-1^T}\mathbf{1} = \frac{1}{\mathbf{1}^T\mathbf{C}^{-1^T}\mathbf{1}} \quad (3.1.19)$$

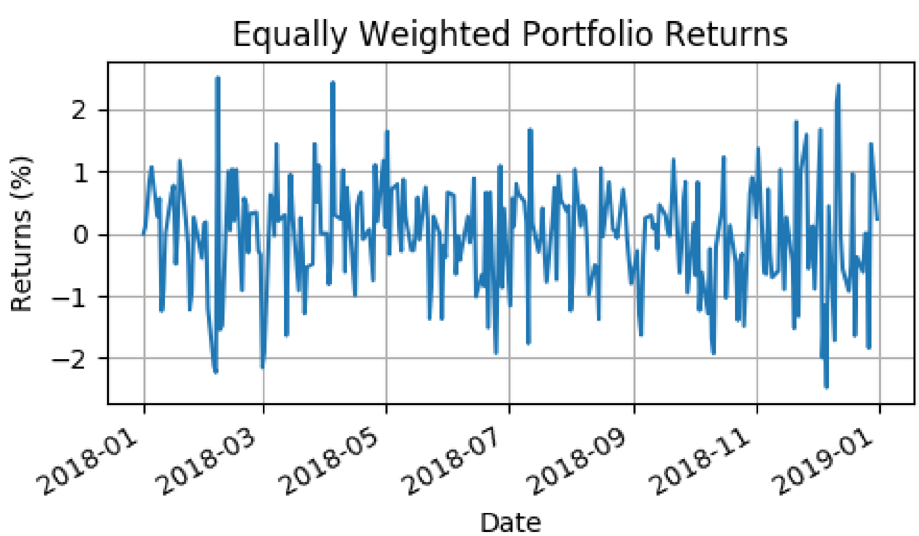
3.1.2 Static Minimum Variance Portfolio

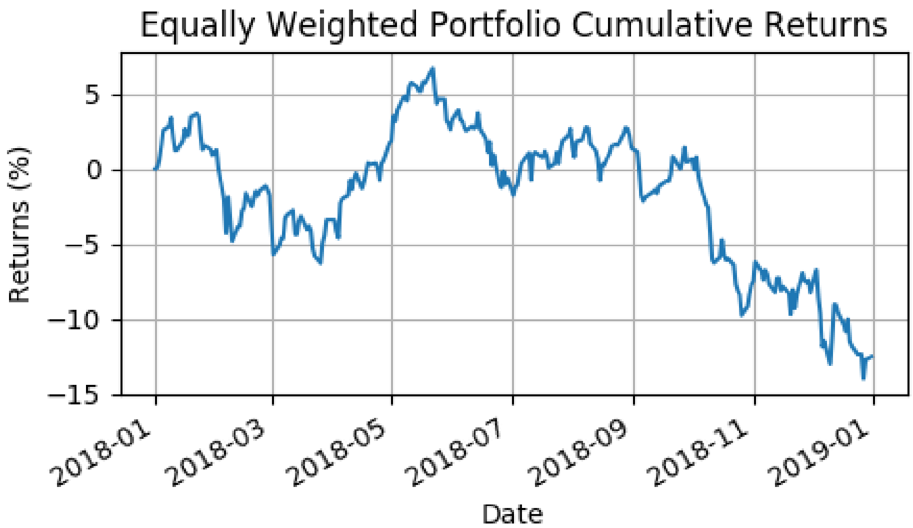
[93]:
```
def optimal_weights(returns):
    C = np.cov(returns, rowvar=False)
    C_inv = np.linalg.inv(C)
    ones = np.ones(shape=[C.shape[0]])

    w_opt = ( np.dot(C_inv,ones) ) / ( np.dot(np.dot(ones.T,C_inv.T), ones) )

    theo_var = 1 / ( np.dot(np.dot(ones.T,C_inv.T), ones) )

    return w_opt, theo_var
```

[94]:
```
def test_portfolio(weights, returns):
    port_returns = (returns*weights).sum(axis=1)

    var = np.var(port_returns)

    cum_returns = np.log1p(port_returns)
    cum_returns = np.expm1(cum_returns.cumsum())
    return port_returns, var, cum_returns
```

Equally Weighted Portfolio

[95]:
```
weights = [1/10 for _ in range(10)]
ret, var_ew, cum_ret = test_portfolio(weights, df_test)
```

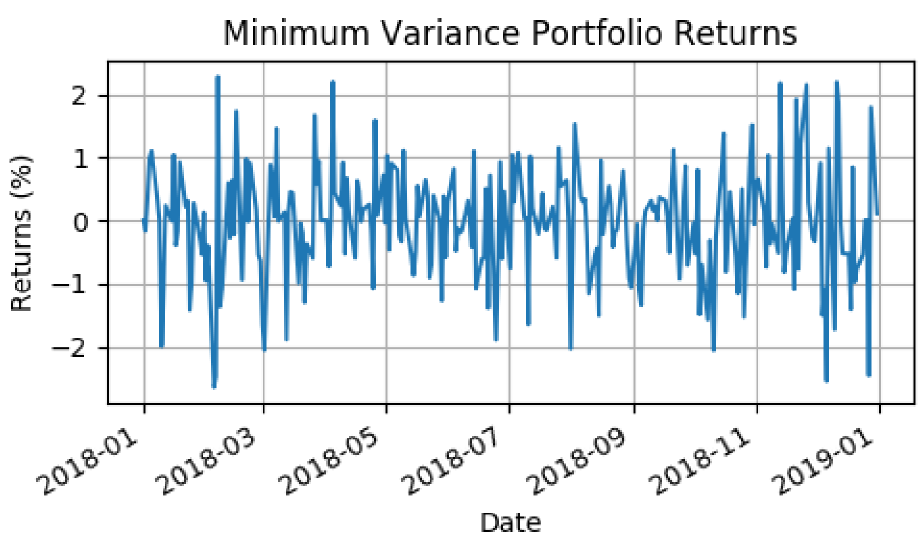


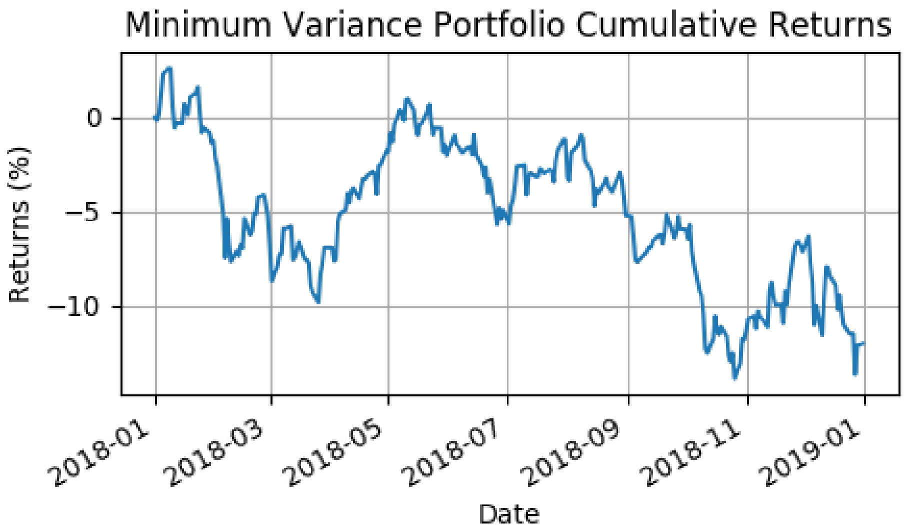


Min-Var Portfolio

[97]:
```
weights, theo_var = optimal_weights(df_train)
ret_mv, var_mv, cum_ret_mv = test_portfolio(weights, df_test)
```

```
Theoretical variance of this portfolio: 0.0000286
```





Strategy	Mean Return (%)	Final Cumulative Ret (%)	Variance
Equal Weights	-0.0473190	-12.5290991	0.0000790
Min-Var	-0.0450505	-12.0391415	0.0000816

The min-var portfolio had a slightly lower mean daily return than the equally weighted portfolio, but had a better final cumulative return.

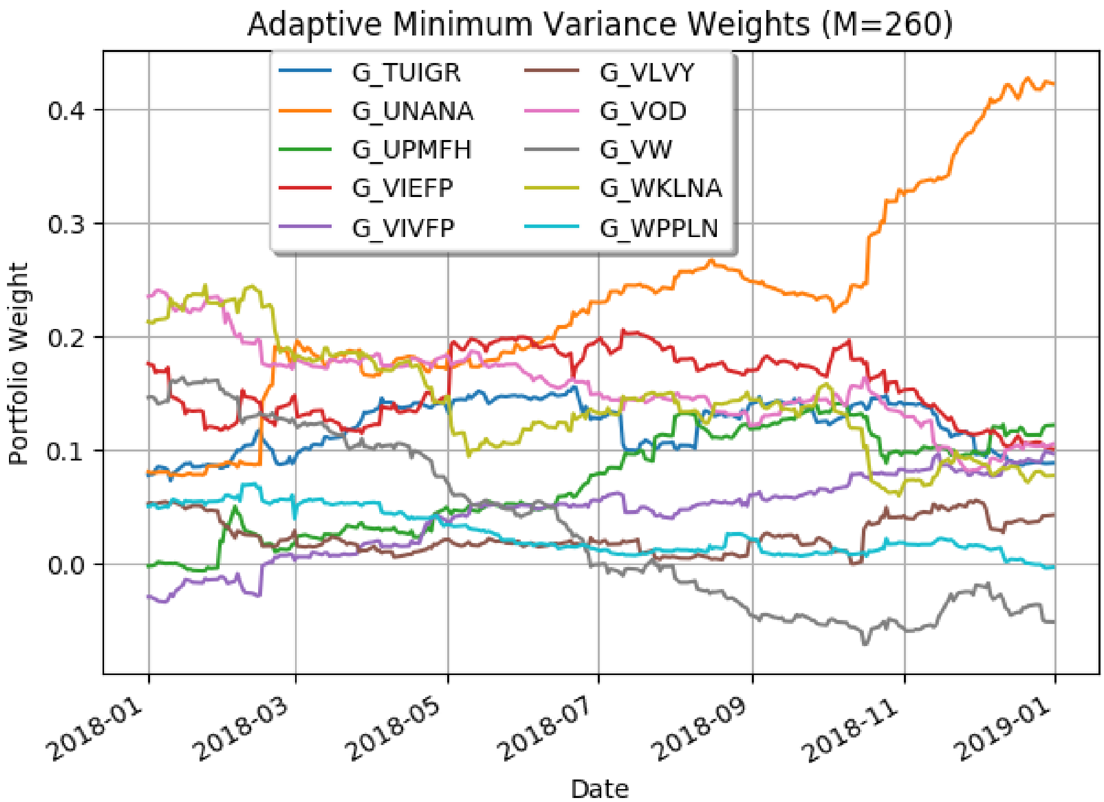
The variance of the portfolio returns, $8.16e-5$ was greater than the theoretical variance of $2.86e-5$. It was also greater than the variance of the equally weighted portfolio ($7.90e-5$). This shows that a minimum variance portfolio over one time period, is not guaranteed to also be a minimum variance portfolio over a different time period.

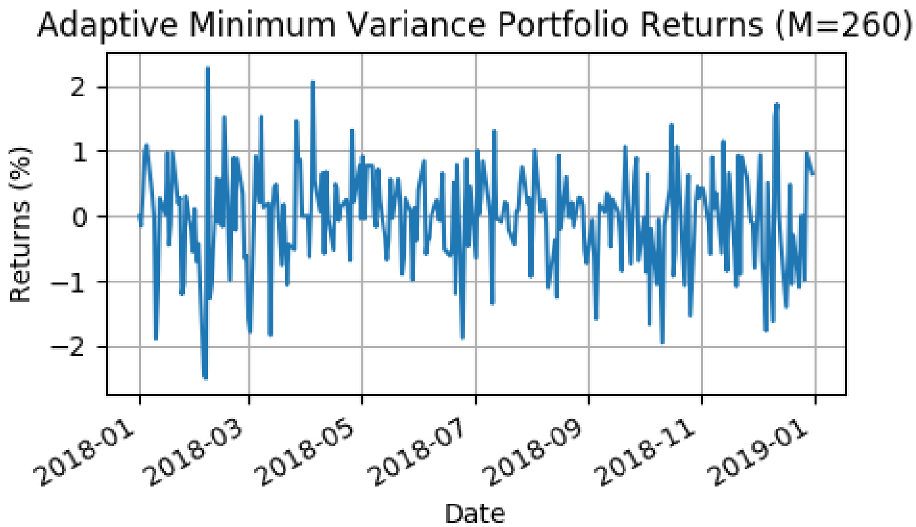
3.1.3 Adaptive Minimum Variance Portfolio

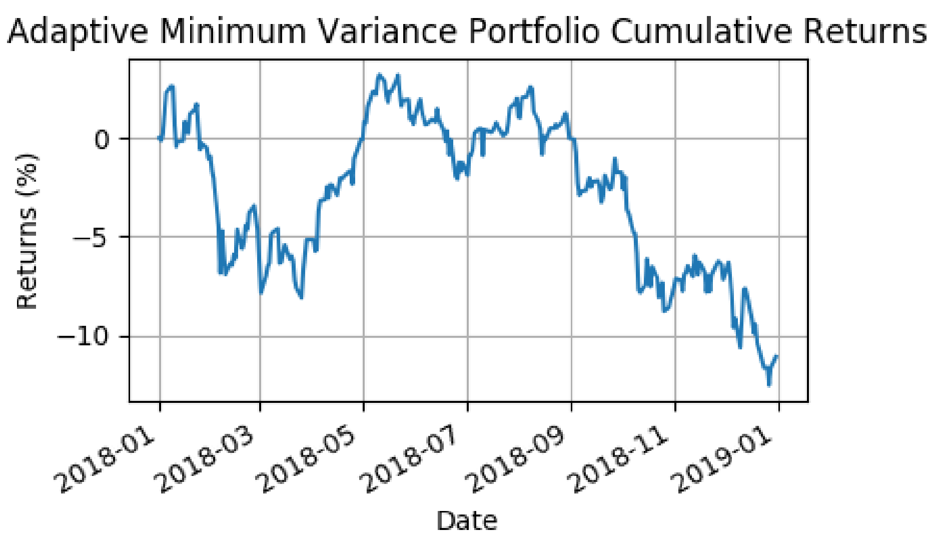
[102]:
```
def optimal_weights_rolling(returns, M):
    Cs = df_t.rolling(M).cov().to_numpy()[M*10:,:].reshape([261,10,10])
    C_invs = np.linalg.inv(Cs)
    ones = np.ones(shape=[10])

    num = np.dot(C_invs,ones)
    div = np.dot(np.dot(ones.T,C_invs.transpose((0,2,1))), ones)

    w_opt = num
    for i, d in enumerate(div):
        w_opt[i] /= d
```

```
    return w_opt
```

[103]:
```
def test_portfolio_rolling(returns, weights, M):

    port_returns = (returns[M:]*weights).sum(axis=1)

    var = port_returns.var()

    cum_returns = np.log1p(port_returns)
    cum_returns = np.expm1(cum_returns.cumsum())
    return port_returns, var, cum_returns
```



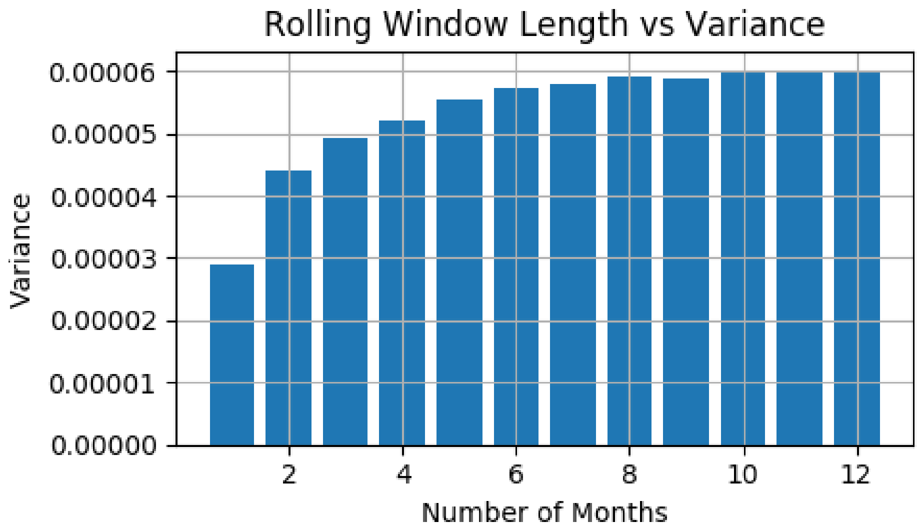


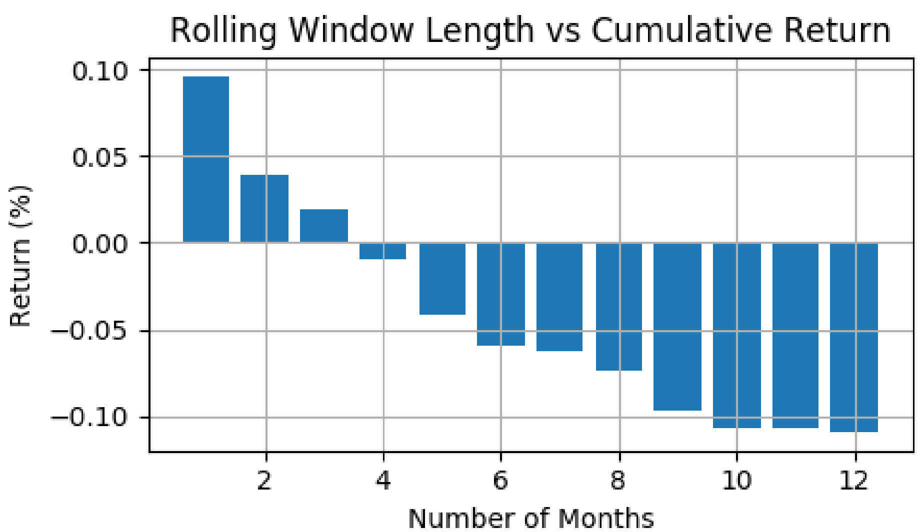


Strategy	Mean Return (%)	Final Cumulative Ret (%)	Variance
Equal Weights	-0.0473190	-12.5290991	0.0000790
Static Min-Var	-0.0450505	-12.0391415	0.0000816
Adaptive Min-Var	-0.0419880	-11.0838185	0.0000603

The adaptive portfolio showed better performance vs the previous two strategies in terms of variance, when using $M = 260$ having the lowest at $6.03e - 5$. It also resulted in a better final cumulative return, but still lost value. However, in practice, updating portfolio weights daily substantially increases transactions costs, and would result in lower returns than shown.

Varying Roliing Window, M





Taking 21 days as one trading month, the number of months the minimum-variance portfolio weights was trained on was varied from 1 to 12. As the number of months increased, so did the variance of the returns, and the cumulative return decreased. Using shorter rolling windows of 1 to 3 months, resulted in positive cumulative returns and significantly smaller variances. This shows that the market was changing enough at least every quarter, that a re-balancing would be required to produce good performance. It is important to note that transactions costs would be independent of the rolling window length, as the portfolio weights are updated daily for each M tested.

The covariance matrix used equal weightings to each daily return in the rolling window. However, there are other methods to assign the weightings. For example, it is possible to place a greater reliance on more recent daily returns, by using an exponential weight with the daily returns when calculating the rolling covariance matrices. So, more recent returns have a larger weighting than older returns. This introduces additional parameters to investigate, such as the decay rate of the weightings. This would be more useful when looking at windows of one quarter or greater, as it allows historical data to have some influence whilst also heavily capturing the more recent trends.

4 Q4 Robust Statistics and Non Linear Methods

4.1 Exploratory Data Analysis Models

4.1.1 Key Descriptive Statistics

For each of AAPL, IBM, JPM and DJI, the tables below show the mean, median, standard deviation, median absolute deviation, interquartile range, skew and kurtosis. For all assets, the mean and median are quite similar apart from the returns, where the median is higher. The measure of dispersion differ more, namely the std.dev and MAD.

AAPL

Statistic	Open	High	Low	Close	Adj. Close	Volume	Return
Mean	187.687	189.562	185.824	187.712	186.174	3.27048e+07	0.000425548
Median	186.29	187.4	184.94	186.12	184.352	2.9184e+07	0.00161136
StdDev	22.1456	22.2816	22.0088	22.1607	21.9047	1.41797e+07	0.019323
MAD	18.1422	18.2413	18.0637	18.1758	17.9117	1.03785e+07	0.0133526
IQR	36	36.34	36.06	36.755	35.6854	1.63117e+07	0.0181045
Skew	0.259917	0.300385	0.220489	0.263849	0.29077	1.74332	-0.41564
Kurtosis	-0.912594	-0.924602	-0.917632	-0.932425	-0.928017	4.35318	4.24306

IBM

Statistic	Open	High	Low	Close	Adj. Close	Volume	Return
Mean	138.454	139.492	137.329	138.363	134.903	5.19894e+06	-0.000251616
Median	142.81	143.99	142.06	142.71	138.566	4.2379e+06	0.000409482
StdDev	12.1143	11.9131	12.2046	12.0281	10.6716	3.32896e+06	0.0155616
MAD	9.99034	9.81456	10.0753	9.9129	8.68543	2.02542e+06	0.010278
IQR	15.38	14.72	16.34	15.505	14.1039	1.95295e+06	0.0132241
Skew	-0.676024	-0.622707	-0.713446	-0.682246	-0.811222	3.1929	-0.309538
Kurtosis	-0.585272	-0.623607	-0.561975	-0.584037	-0.420852	11.7969	7.23577

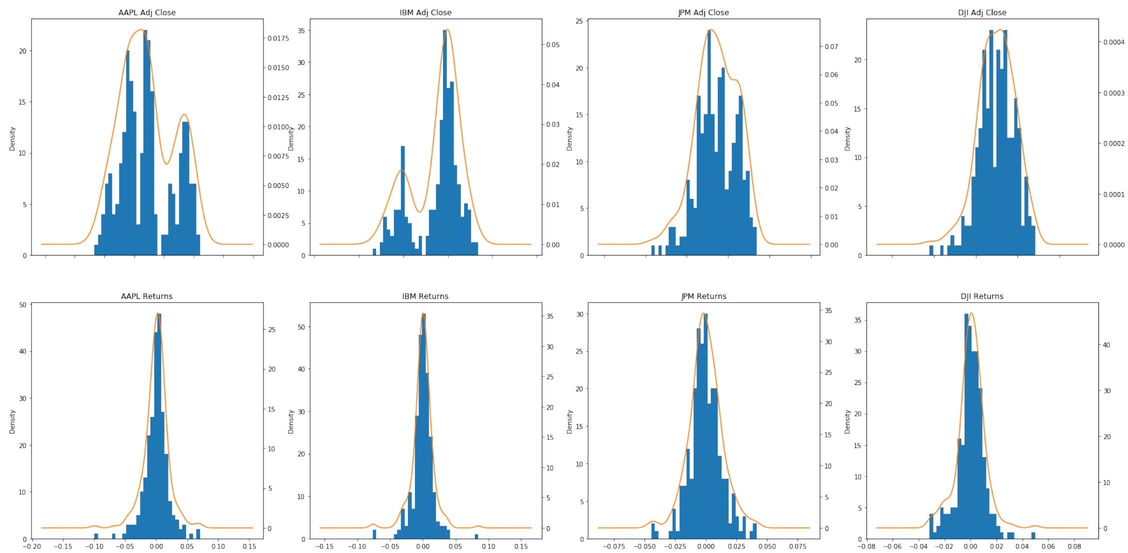
JPM

Statistic	Open	High	Low	Close	Adj. Close	Volume	Return
Mean	108.708	109.652	107.683	108.607	107.263	1.47007e+07	-0.00013302
Median	109.18	110.53	107.79	109.02	107.219	1.3633e+07	-0.000602616
StdDev	5.35908	5.20287	5.43254	5.30048	4.83332	5.34977e+06	0.0130878
MAD	4.41151	4.33669	4.43555	4.37808	3.92939	3.945e+06	0.00975647
IQR	8.81001	8.845	8.845	8.835	7.22244	6.2336e+06	0.01497
Skew	-0.420811	-0.376221	-0.377517	-0.374853	-0.344491	1.69346	0.0228698
Kurtosis	-0.322536	-0.544163	-0.2657	-0.396579	-0.105437	4.4302	1.31859

DJI

Statistic	Open	High	Low	Close	Adj. Close	Volume	Return
Mean	25001.3	25142	24846	24999.2	24999.2	3.32889e+08	0.00019681
Median	25025.6	25124.1	24883	25044.3	25044.3	3.1379e+08	0.000374537
StdDev	858.835	815.204	903.302	859.132	859.132	9.4078e+07	0.0104764
MAD	682.034	658.96	712.026	686.324	686.324	6.87602e+07	0.0074519
IQR	1109.43	1077.82	1204.42	1158.16	1158.16	1.0893e+08	0.00988984
Skew	-0.372127	-0.239367	-0.456447	-0.380147	-0.380147	1.73956	-0.015718
Kurtosis	0.485736	0.118153	0.557592	0.400668	0.400668	5.85758	2.77395

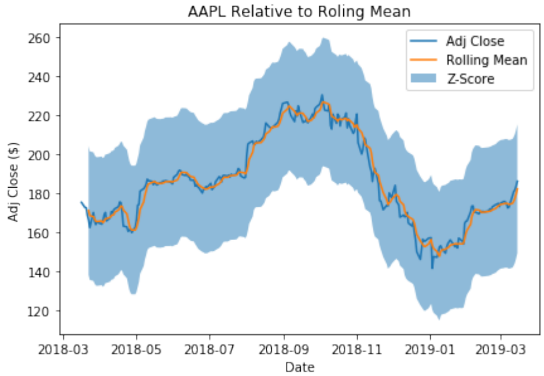
4.1.2 Histograms and Probability Density Functions

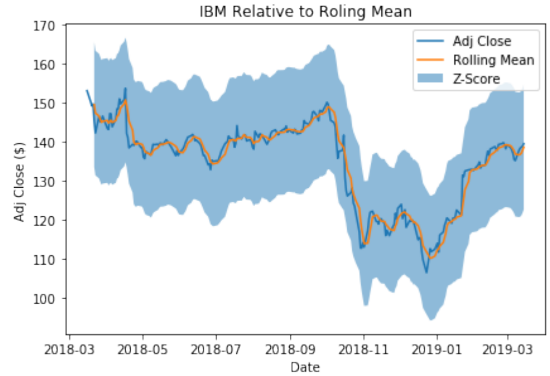


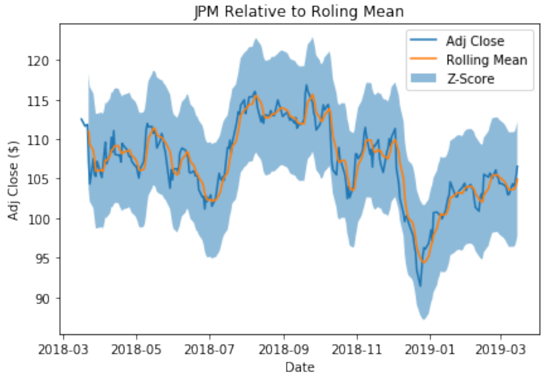
The pdfs of the returns appear to be a normal distribution, making the use of mean and standard deviation viable for them. However, the pdfs of the prices do not to appear to be normal distributions. So the classical statistics of mean and std should not be used with classic methods that assume normality of the underlying distribution, when using the price data.

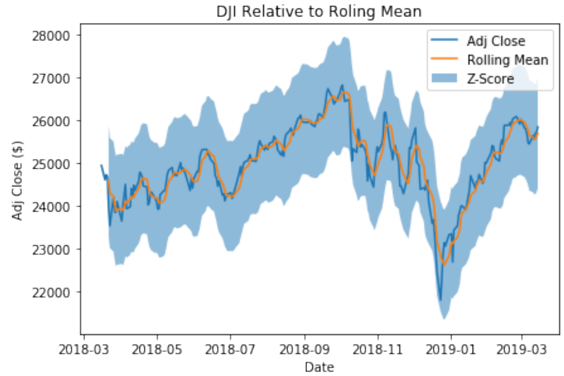
4.1.3 Mean vs Median Outlier Detection

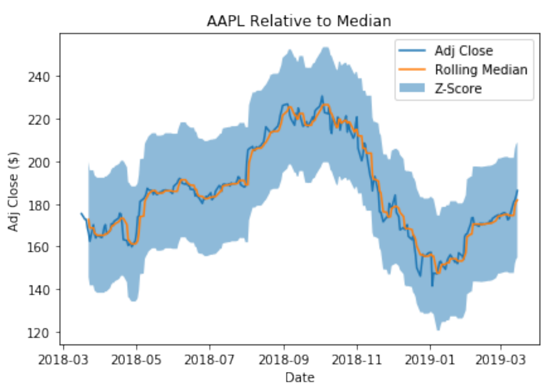
Using the static dispersion measurements:

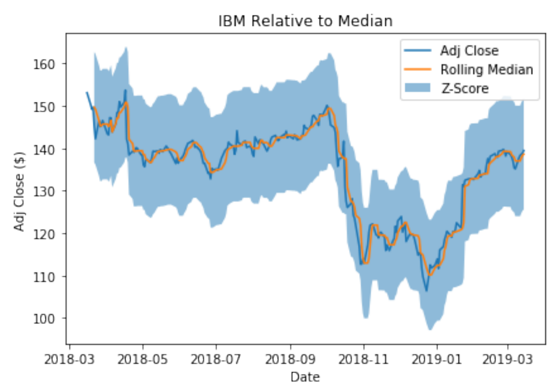

AAPL Relative to Roling Mean
Adj Close
Rolling Mean
Z-Score
Adj Close ($)
260
240
220
200
180
160
140
120
2018-03
2018-05
2018-07
2018-09
2018-11
2019-01
2019-03
Date

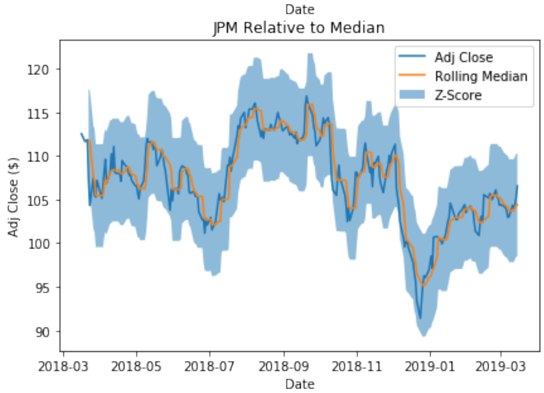

IBM Relative to Roling Mean
Adj Close
Rolling Mean
Z-Score
Adj Close ($)
170
160
150
140
130
120
110
100
2018-03
2018-05
2018-07
2018-09
2018-11
2019-01
2019-03
Date

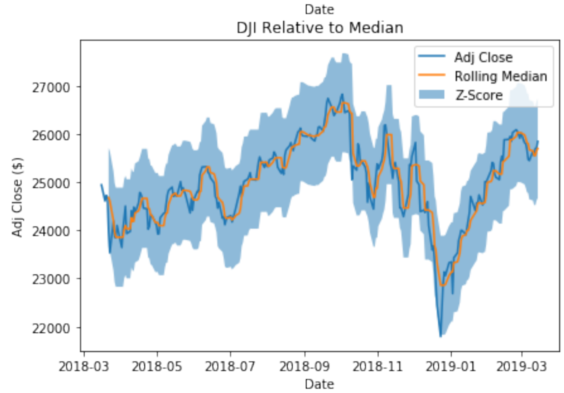

JPM Relative to Rolling Mean
Adj Close
Rolling Mean
Z-Score
Adj Close ($)
120
115
110
105
100
95
90
2018-03
2018-05
2018-07
2018-09
2018-11
2019-01
2019-03
Date


DJI Relative to Rolling Mean
Adj Close
Rolling Mean
Z-Score
Adj Close ($)
28000
27000
26000
25000
24000
23000
22000
2018-03
2018-05
2018-07
2018-09
2018-11
2019-01
2019-03
Date

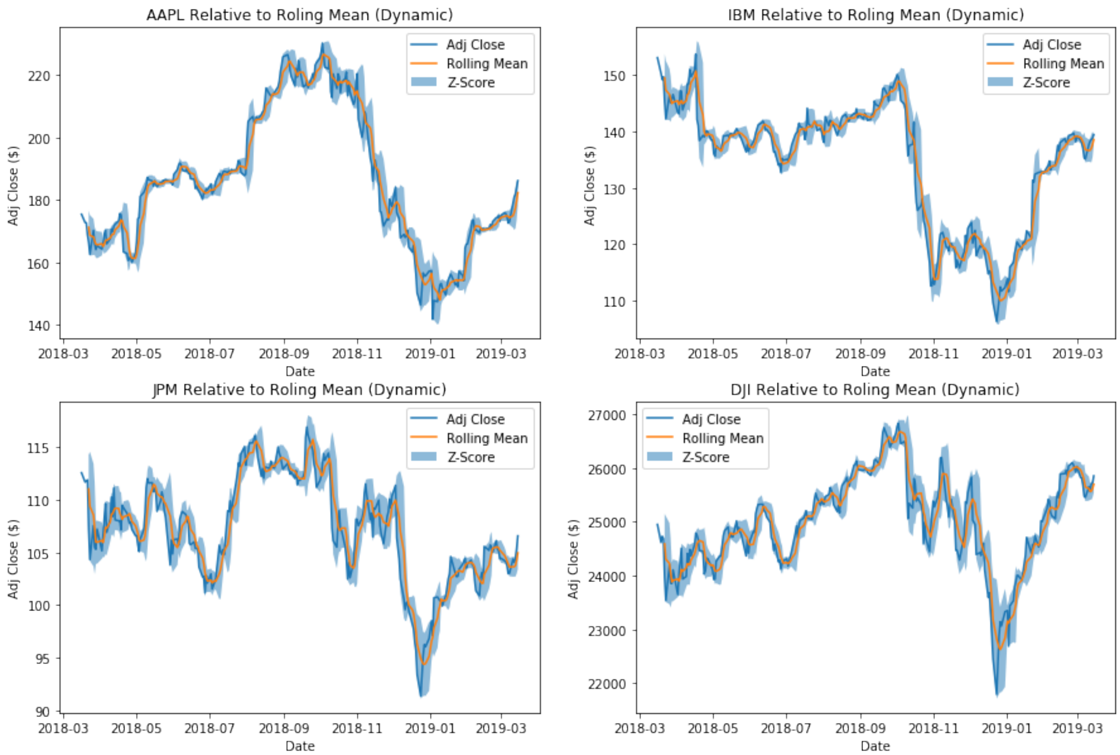

AAPL Relative to Median
Adj Close
Rolling Median
Z-Score
Adj Close ($)
240
220
200
180
160
140
120
2018-03
2018-05
2018-07
2018-09
2018-11
2019-01
2019-03
Date

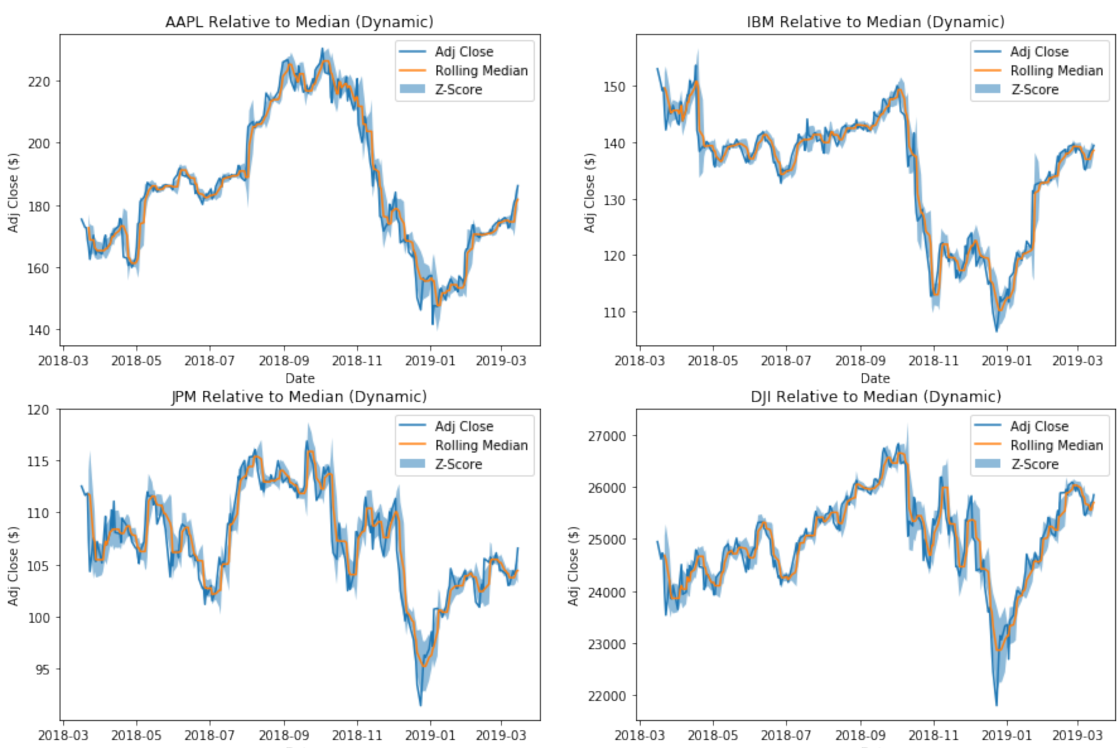

IBM Relative to Median
Adj Close
Rolling Median
Z-Score
Adj Close ($)
160
150
140
130
120
110
100
2018-03
2018-05
2018-07
2018-09
2018-11
2019-01
2019-03
Date


JPM Relative to Median
Adj Close
Rolling Median
Z-Score
Adj Close ($)
120
115
110
105
100
95
90
2018-03
2018-05
2018-07
2018-09
2018-11
2019-01
2019-03
Date


DJI Relative to Median
Adj Close
Rolling Median
Z-Score
Adj Close ($)
27000
26000
25000
24000
23000
22000
2018-03
2018-05
2018-07
2018-09
2018-11
2019-01
2019-03
Date

Using dynamic dispersion measurements (rolling):





Outliers from each Detection Method

Stock	Mean, Static Std	Median, Static MAD	Mean, Dynamic Std	Median, Dynamic MAD
AAPL	0	0	30	98
IBM	0	0	31	98
JPM	0	2	33	97
DJI	0	3	30	95

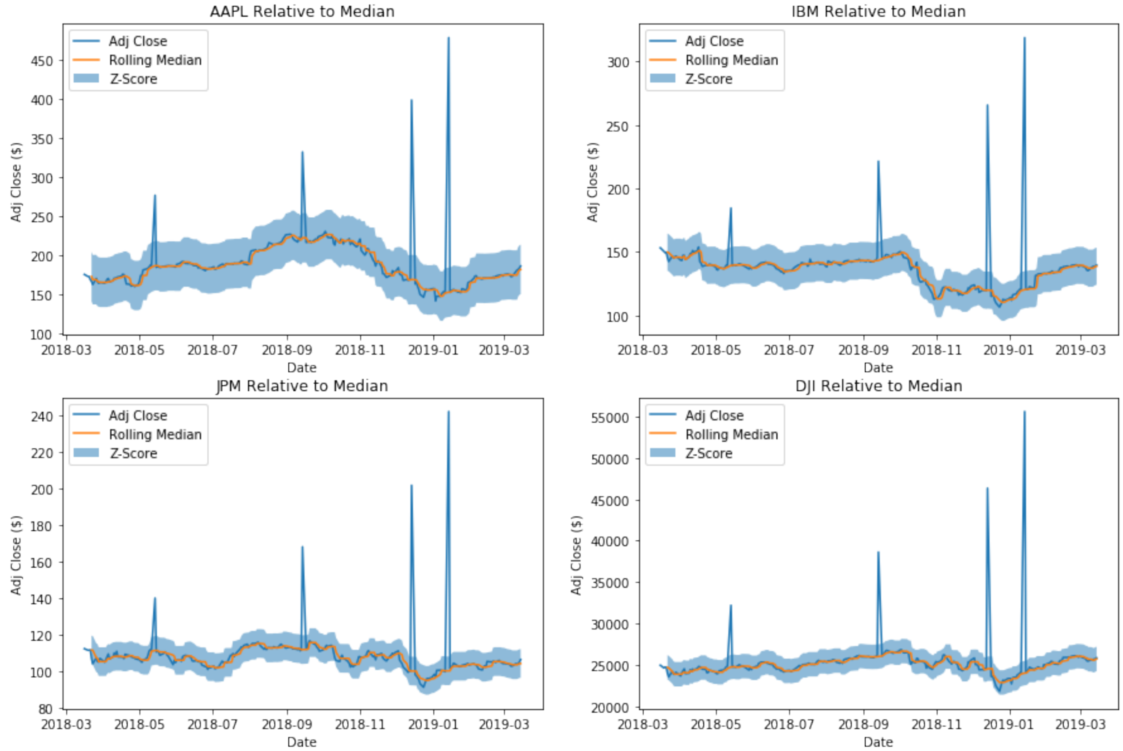
In practice, the static dispersion measures cannot be used as they require future data. The dynamic approach, calculates the std. and MAD from the same rolling window used for the mean and median.

The mean method results in less outliers being detected. This is due to an outlier effecting the std. greatly, whereas it has minimal impact on the MAD. As the std. increases more with outliers, they are less likely to be considered out of the range compared to the mean.

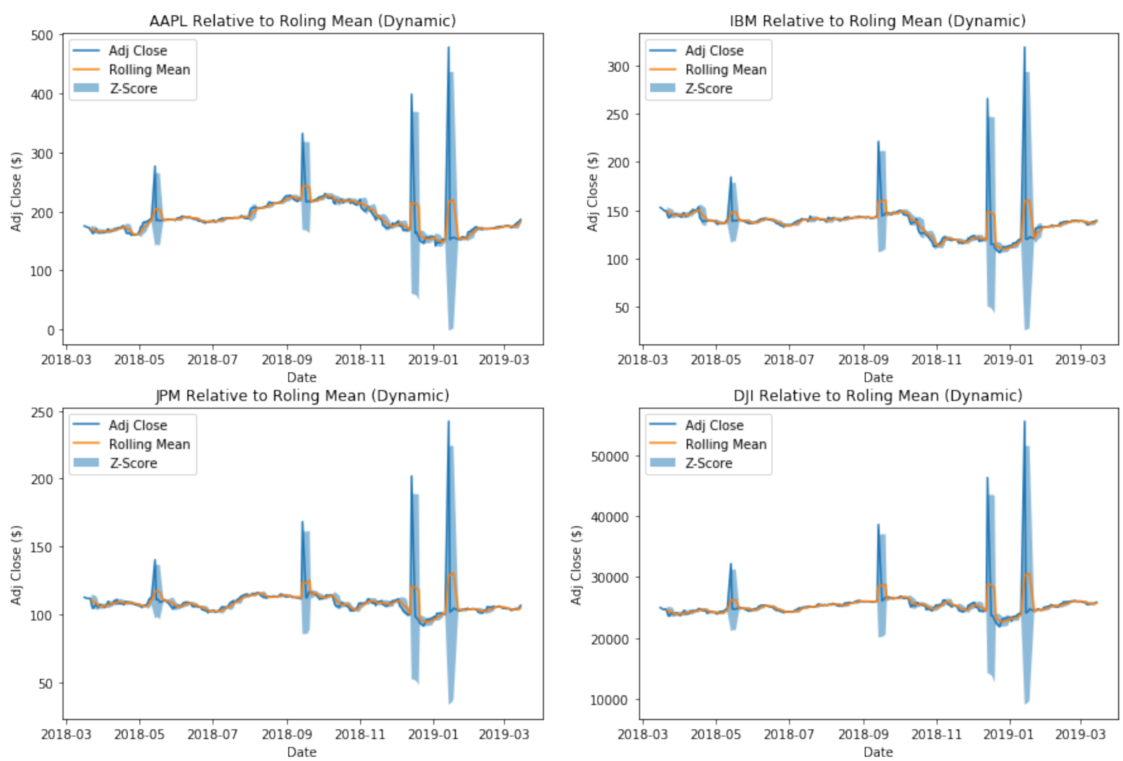
Overall, the dynamic method detects more outliers than the static method, which is expected as the price range has a tighter bound / window.

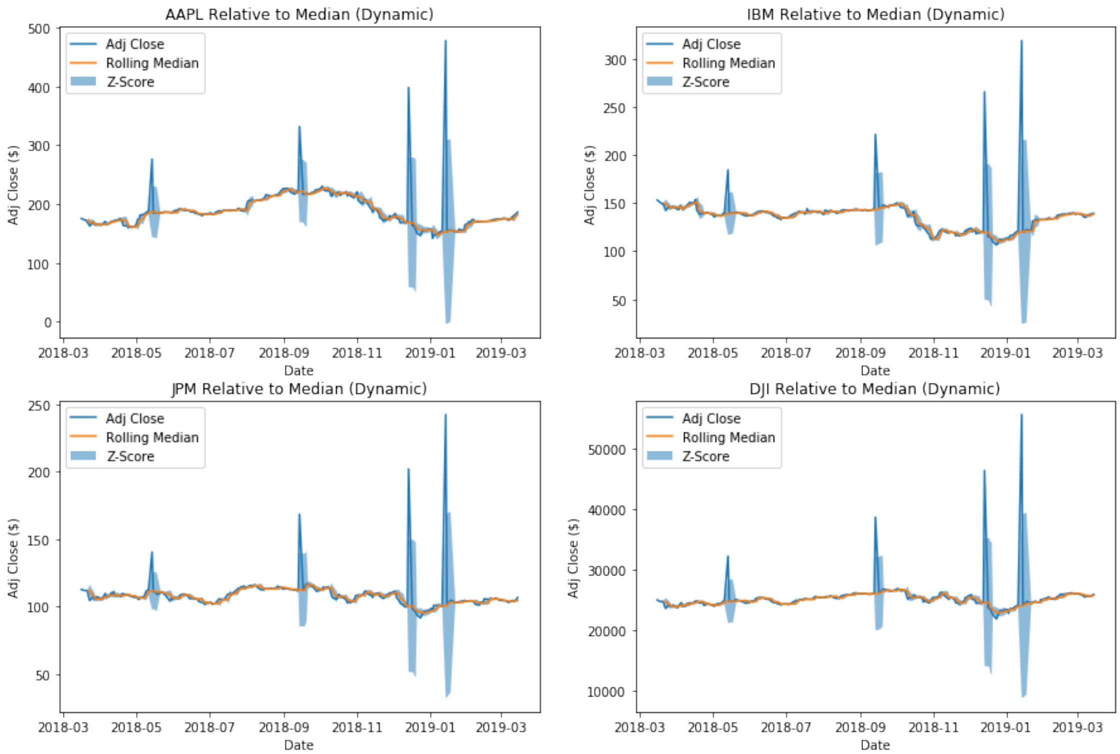
4.1.4 Impact of Artificial Outliers

Using the static dispersion measurements:



Using dynamic dispersion measurements (rolling):





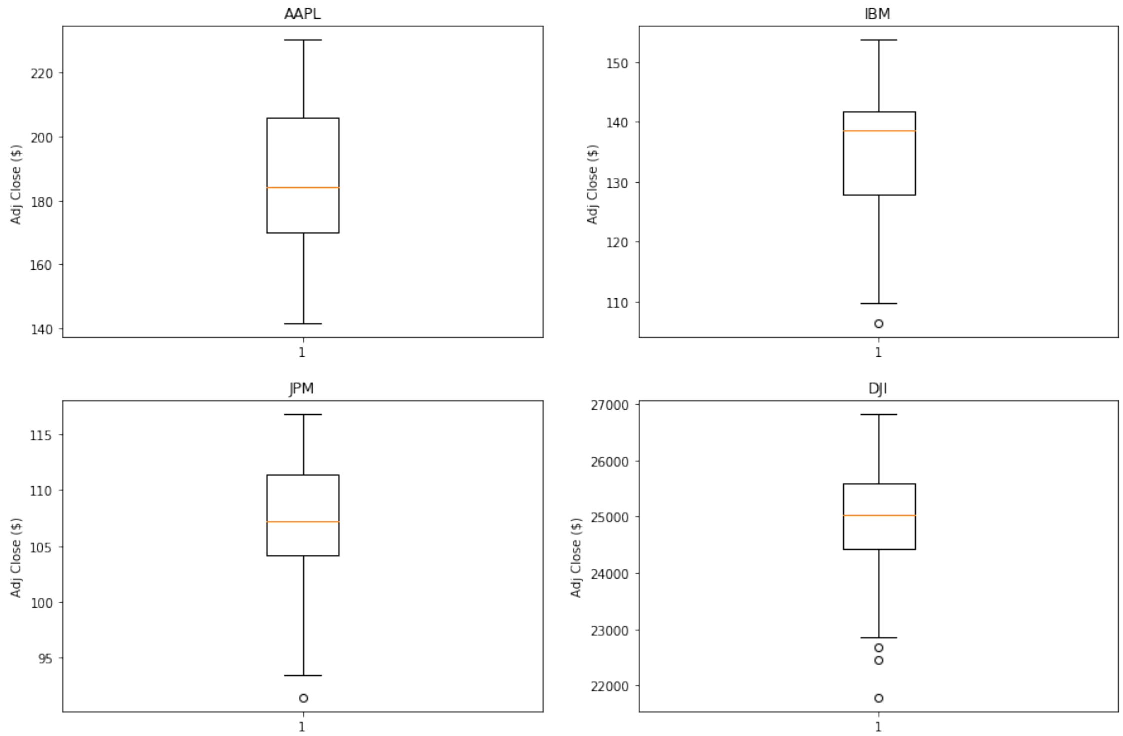
Outliers from each Detection Method

Stock	Mean, Static Std	Median, Static MAD	Mean, Dynamic Std	Median, Dynamic MAD
AAPL	11	4	32	93
IBM	12	4	31	94
JPM	12	4	33	88
DJI	12	4	29	90

As expected, the dynamic methods were not impacted that much by the artificial outliers.

The introduced outliers impacted the mean based, static std. detection method the most, making the average number of detected outliers 12, instead of the previous total of 0. The outliers increased the mean, and skewed the data to be short-tailed. This lead to the lower values being less likely to fall within the window around the rolling mean, and hence be detected as outliers.

4.1.5 Box Plots



The box plot provides: - median - the orange line - interquartile rang (IQR) - length of the box - 25% quartile - bottom line of box - 75% quartile - top line of box - min value excl. outliers - bottom whisker - max value excl. outliers - top whisker - points outside 1.5x the IQR - points on the plot

Each of the plots shown either have one or more of: - asymmetric quartiles - asymmetric distance to median from min vs max - outlier points

This suggests that normal / Gaussian distribution of the data cannot be assumed.

4.2 Robust Estimators

4.2.1 Implementation

[21]:
```
def median(s):
    _sorted = s.sort_values()
    return _sorted[int(len(_sorted)/2)]

def IQR(s):
    _sorted = s.sort_values()
    quarter = len(_sorted)/4
    lo = _sorted[int(quarter)]
    hi = _sorted[int(3*quarter)]
    return hi-lo

def MAD(s):
    med = median(s)
    devs = abs(s-med)
    return median(devs)
```

4.2.2 Complexity Analysis

For series length N.

Mean: - Requires a sort - $O(nlogn)$ - Array lookup for middle term - $O(1)$ - Overall - $O(nlogn)$

IQR: - Requires a sort - $O(nlogn)$ - Index calculation, array lookups, difference - $O(1)$ - Overall - $O(nlogn)$ - Same complexity as median, 1 more lookup, 3 extra calculations - For large N, same performance as median

MAD: - Two median calculations - $O(nlogn)$ - Absolute difference calculation - $O(n)$ - Overall - $O(nlogn)$ - Same complexity as both median and IQR, but over twice the amount of operations

All three estimators are less computationally efficient than both mean and stddev ($O(n)$).

4.2.3 Breakdown Points

Breakdown point of an estimator - a measure of its robustness.

To expand, if the sample data were to change, then it indicates the level above which the estimator can deteriorate. Specifically, the sample breakdown point is the fraction of the data that can be changed, without making the estimator 'bad' for a sample of data of size N. As n tends to infinity, the breakdown point instead refers usually to the asymptotic breakdown point.

Median: - breakdown point - 0.5 - up tp $\frac{0.5n-1}{n}$ can change without effecting the middle value of a sorted array

IQR: - breakdown point - 0.25 - Split the data array into 4 parts, either one can change

MAD: - breakdown point - 0.5 - computes the median to then compute the median of absolute deviation from the median

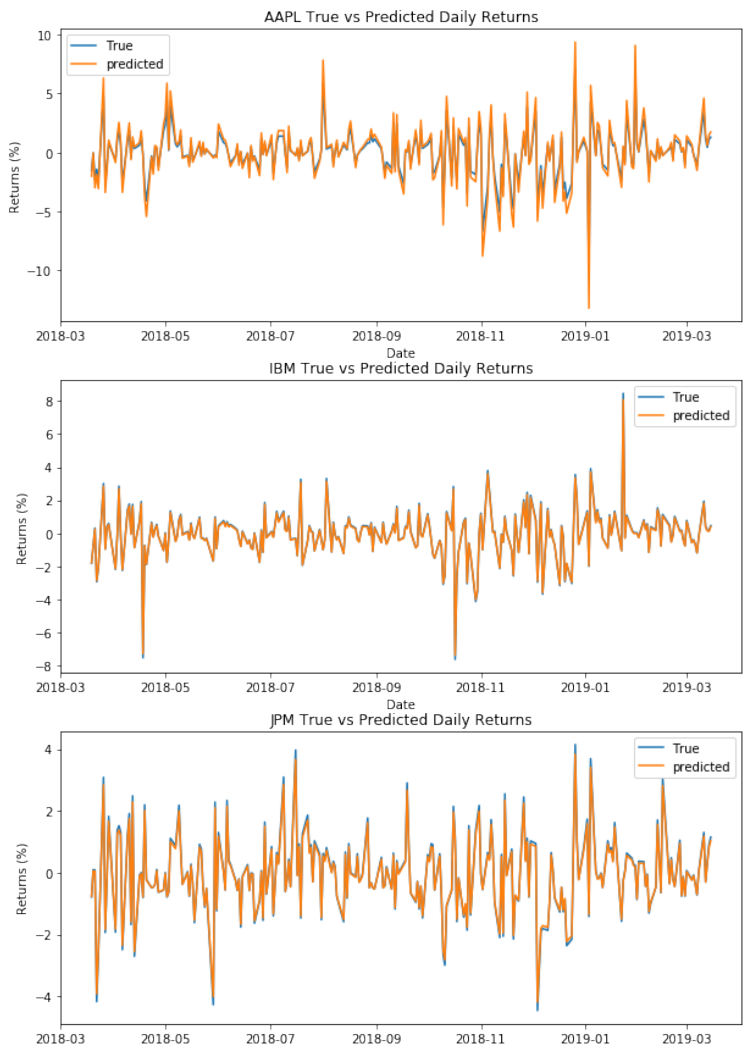
4.3 Robust and OLS regression

4.3.1 OLS Regression

For each stock: - $\mathbf{r} = \mathbf{X}\mathbf{b} + \mathbf{e}$ - $\mathbf{r}$ is the 1-day return if the stock - $\mathbf{X}$ is the 1-day DJI returns - $\mathbf{b}$ are the OLS coefficients - $\mathbf{e}$ is the regression residual

- OLS aims to minimize the error, $\mathbf{e}^2 = (\mathbf{r} - \mathbf{a} - \mathbf{X}\mathbf{b})^2$
  - This has the solution $\hat{\mathbf{b}} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{r}$.

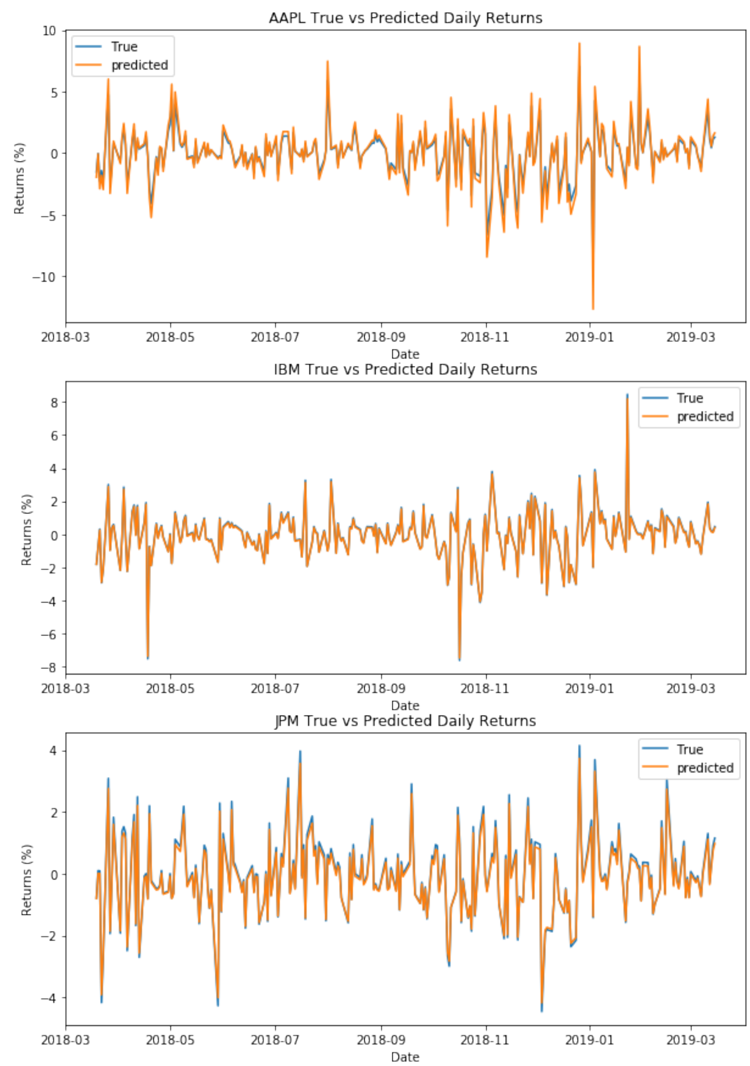
	AAPL	IBM	JPM
alpha	0.00016466	-0.000440572	-0.000316331
beta	1.32558	0.960092	0.931408
SSE	0.00987816	0.000142372	0.000224262



4.3.2 Huber Regression

For each stock: - minimize the squared error for $\frac{||r-Xb||}{\sigma} < e$ and the absolute error for $\frac{||r-Xb||}{\sigma} > e$

	AAPL	IBM	JPM
alpha	-0.000130371	-0.000509435	-0.000800961
beta	1.27021	0.973562	0.919662
SSE	0.00678835	0.000105344	0.000431414



4.3.3 Comparison

The Huber Regressions resulted in 31.2% lower SSE for AAPL and 26.1% lower SSE for IBM. However, for JPM, it resulted in a 92.4% increase in SSE.

The OLS regression method is highly sensitive to outliers in the data as it aims to minimize the squared errors which is a measure of squared deviations. Therefore, the OLS model assumes that the underlying distribution has consistent variance, which is not true for data with outliers.

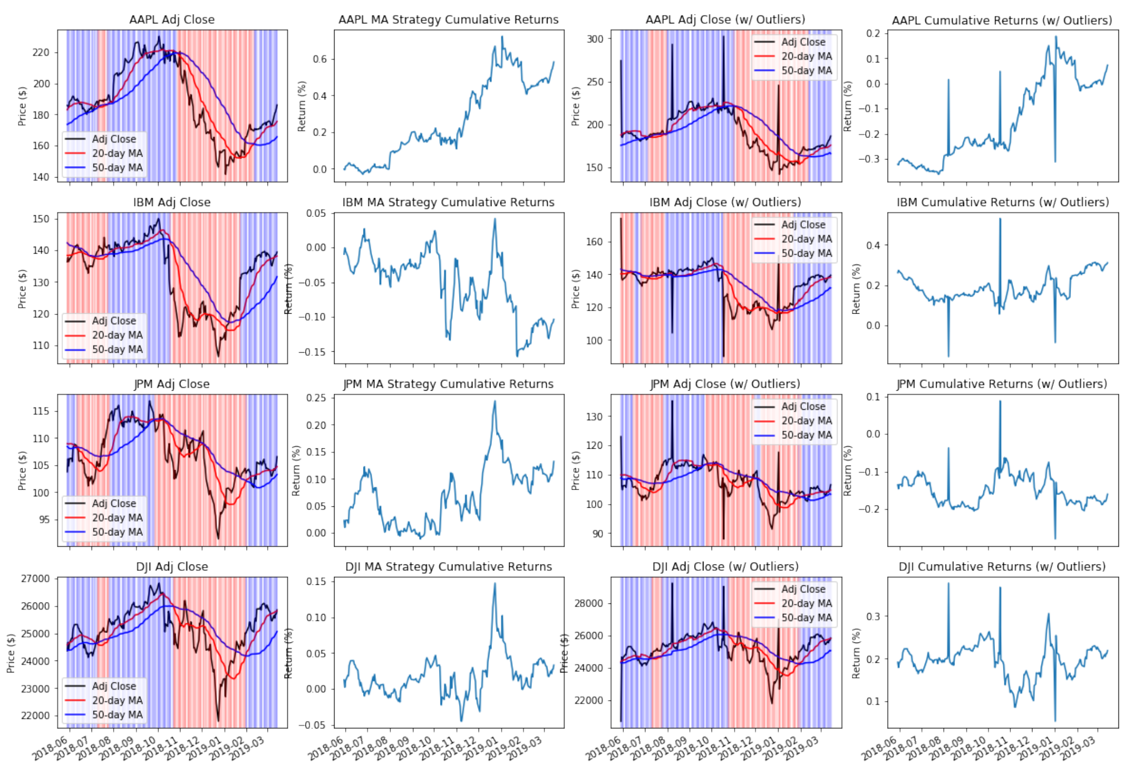
Huber regression makes the error asymmetric such that it is quadratic for small values but linear for bigger ones, it effectively reduces the impact of outliers on the estimation of the parameters.

4.4 Robust Trading Strategies

4.4.1 Moving Mean Crossover Strategy

Overlap between MA strategy with and without Outliers

Ticker	Overlap (%)	Cumulative Returns Correlation
AAPL	93.0348	0.288576
IBM	94.5274	0.37606
JPM	93.5323	0.0190856
DJI	97.0149	0.549375



The figures to the left show the MA strategy and its performance on the original data, and the figures on the right show them for a corrupted version of the stock.

The blue vertical lines represent buy decisions and red represent sell decisions.

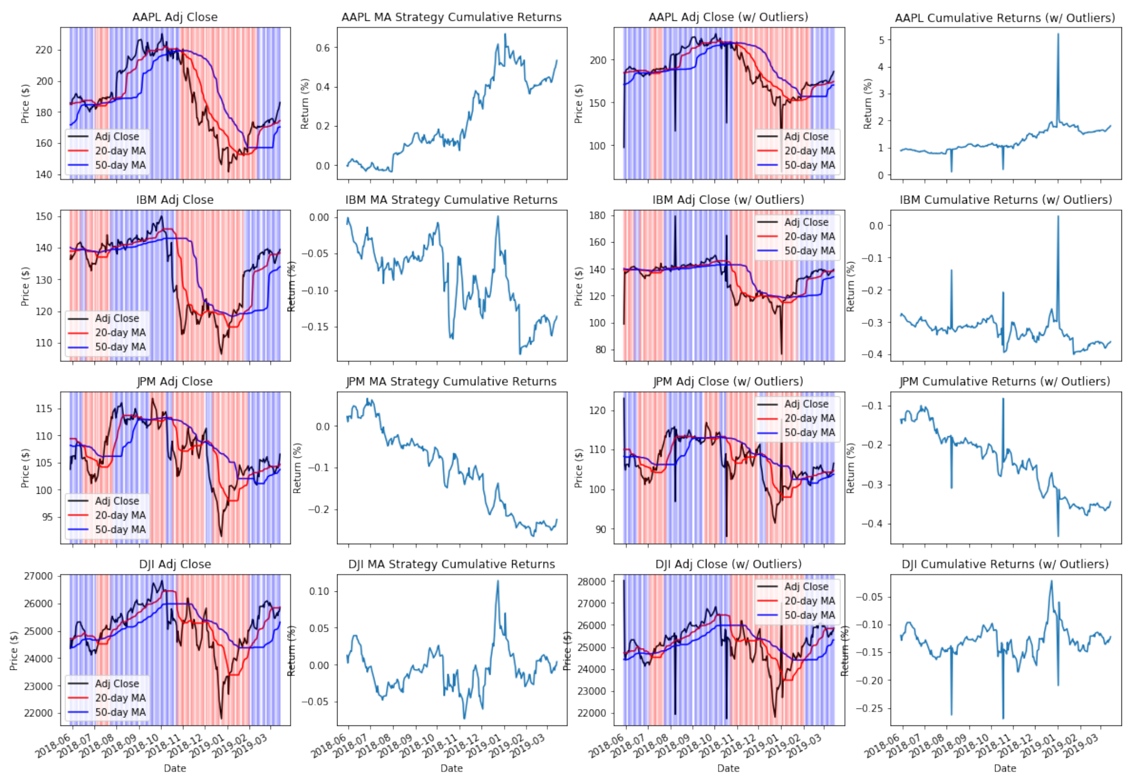
For the individual stock, the presence of the outliers reduce the overlap of the regions by over 5%, but less than 3% for the index.

The performance, measured by cumulative return, was worse for all tickers when the strategy was applied to the corrupted data. The correlation between the returns was under 0.5 for the stocks, and around 0.5 for the index.

4.4.2 Moving Median Crossover Strategy

Overlap between MA strategy with and without Outliers

Ticker	Overlap (%)	Cumulative Returns Correlation
AAPL	99.5025	0.831634
IBM	99.5025	0.717987
JPM	99.005	0.978919
DJI	99.005	0.838572



When using the rolling median, the overlap between long and short regions with and without the outliers, is close to 100%, unlike the rolling mean where the overlap reduced by over 5% for the stocks and 3% for the index.

The cumulative returns follow a similar trend with the outliers, shown by the average correlation of approx. 0.8. This is much larger than for the rolling mean version (0.5 and under). This shows that the rolling median MA strategy is more robust against outliers in the data.

5 Q5 Graphs in Finance

5.1 Choice of assets

The SP500s largest sector is technology. Within this sector, two sub groups have been chosen:

- Large Market-Cap
  - Alphabet Inc
  - Amazon
  - Apple
  - Facebook
  - Netflix
  - Microsoft
- Semiconductor
  - Intel
  - AMD
  - NVIDIA
  - Xilinx

The first group contains the most well known companies, who also have some of the largest market-caps in the index.

The second group contain the 3 largest datacentre players, hardware wise. Recent cloud services have also offered FPGA solutions, and so Xilinx was included as alongside Intel, they dominate the FPGA market share.

The motivation behind this selection is to look at the connections between the largest users of the hardware produced by the semiconductor companies.

```
Correlations (average excludes self-correlation)
```

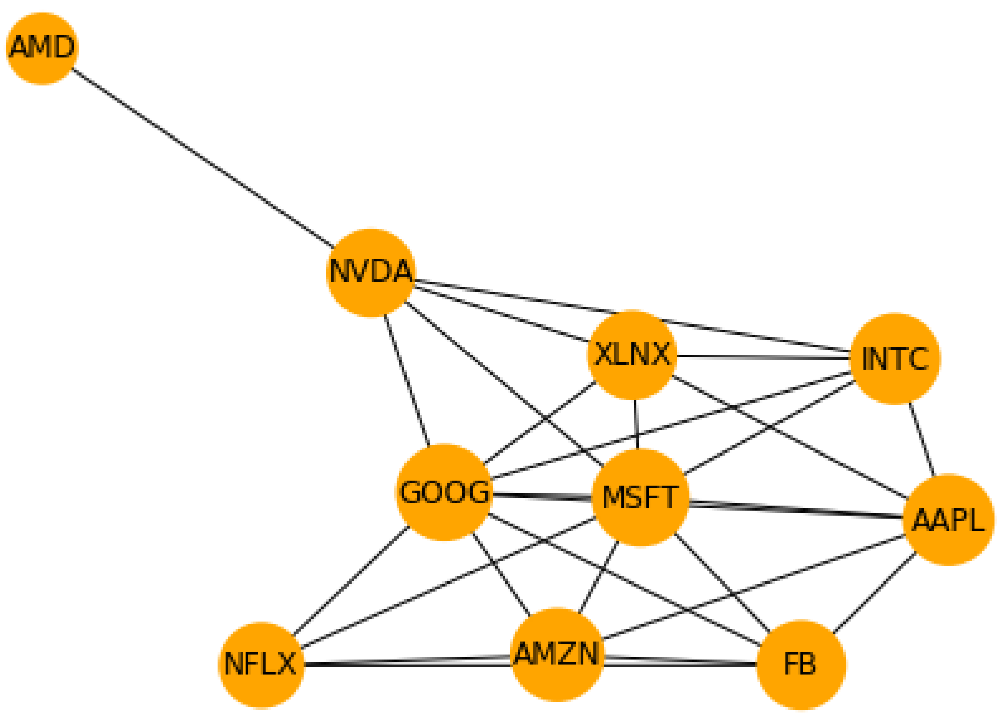
[6]:

	GOOG	AMZN	AAPL	FB	NFLX	MSFT	AMD \
GOOG	1.000000	0.658465	0.516988	0.605307	0.482377	0.673407	0.206115
AMZN	0.658465	1.000000	0.485394	0.564983	0.486907	0.613155	0.241828
AAPL	0.516988	0.485394	1.000000	0.455673	0.377391	0.565830	0.254863
FB	0.605307	0.564983	0.455673	1.000000	0.408820	0.512326	0.193364
NFLX	0.482377	0.486907	0.377391	0.408820	1.000000	0.439069	0.238924
MSFT	0.673407	0.613155	0.565830	0.512326	0.439069	1.000000	0.215910
AMD	0.206115	0.241828	0.254863	0.193364	0.238924	0.215910	1.000000
INTC	0.454353	0.380928	0.465652	0.386259	0.360302	0.583873	0.288890

NVDA	0.417002	0.385172	0.399208	0.367231	0.345852	0.460514	0.415791
XLNX	0.458464	0.381782	0.454684	0.290227	0.322659	0.487455	0.343273

	INTC	NVDA	XLNX	average
GOOG	0.454353	0.417002	0.458464	0.447248
AMZN	0.380928	0.385172	0.381782	0.419861
AAPL	0.465652	0.399208	0.454684	0.397568
FB	0.386259	0.367231	0.290227	0.378419
NFLX	0.360302	0.345852	0.322659	0.346230
MSFT	0.583873	0.460514	0.487455	0.455154
AMD	0.288890	0.415791	0.343273	0.239896
INTC	1.000000	0.453693	0.553593	0.392754
NVDA	0.453693	1.000000	0.477436	0.372190
XLNX	0.553593	0.477436	1.000000	0.376957

5.2 Constructing a Graph



Role of the correlation matrix The correlation matrix is the bases for the graph network. Each entry (not including the diagonal) represents a connection to another node. The value in the entry represents the weight for the edge in the graph, and correlation of log-returns is used as the weight of the edge. To visualize the larger weights, higher values result in nodes being placed closer together, as apposed to drawing thicker edges. Smaller weights result in the respective nodes being spaced further apart. The nodes with the largest sum of weights are scaled to be larger than nodes with a lower sum of weights.

5.3 Analysis of Correlation Graph

The threshold for and edge was set to 0.4 correlation. As this graph was built over two sub sectors, a lower threshold was required.

Results Microsoft was connected to all of the companies apart from AMD. MSFT has a very diverse portfolio of products and services, and so it was expected to see it have many visible connections. Xilinx was closest to MSFT. An explanation for this is MSFT large cloud service, Azure, and the use of Xilinx's products in the rapidly growing cloud computing service. Alphabet Inc also provide cloud services and also have an edge to Xilinx. Amazon has its cloud service AWS, but doesn't correlate as much with Xilinx as the other main providers do. This could suggest AWS is still a small driver of Amazons returns.

No semiconductors have edges with Amazon, which further shows the lower significance of AWS for Amazon returns. The large market-caps that have significant hardware interests, (cloud of consumer products) connect well with the semiconductors. This provides support for the motivation of investigating large market-cap tech and the semiconductor industry.

AMD has only one connection, to Nvidia, and it very distant. The connection can be derived from the fact that both companies product enterprise and consumer graphics products. While AMD does produce x86 CPUs like Intel, only very recently has it gained noticeable market share. Data over a much more recent period would be expected to result in a edge between the two companies.

Netflix and Apple are at opposite end of the large market-cap section of the graph. These two stocks have low correlation, which is to be expected given the very different nature of their businesses. Apple mainly produces consumer electronics whereas Netflix solely focuses on entertainment.

Re-Ordering Re-ordering the time-series data, i.e. swapping returns between days, if applied to all series, would not change the correlation between the series. Therefore the results would not be affected by data re-ordering. However, re-ordering the series or the graph vertices can effect the results due to the construction process.

The graph is derived from the correlation matrix, but the initial starting point is random. The process then iterates over the matrix to build the graph. As long as the correlation values are the same, then the size of each node should remain the same. However, the order of the time-series and order of vertices, along with the starting position, do effect the distance between the vertices. This is due to the dynamic nature of the process as well as the non-deterministic start.

5.4 An alternative distance metric

The distance metric chosen aims to capture similarity between the volatility of the log-returns. This is done by computing the rolling StdDev (21 days) of the log-returns, and then finding the correlation between each of the assets.

The aim is to find a relationship around assets that increase and decrease in volatility together, regardless of the direction of the price.

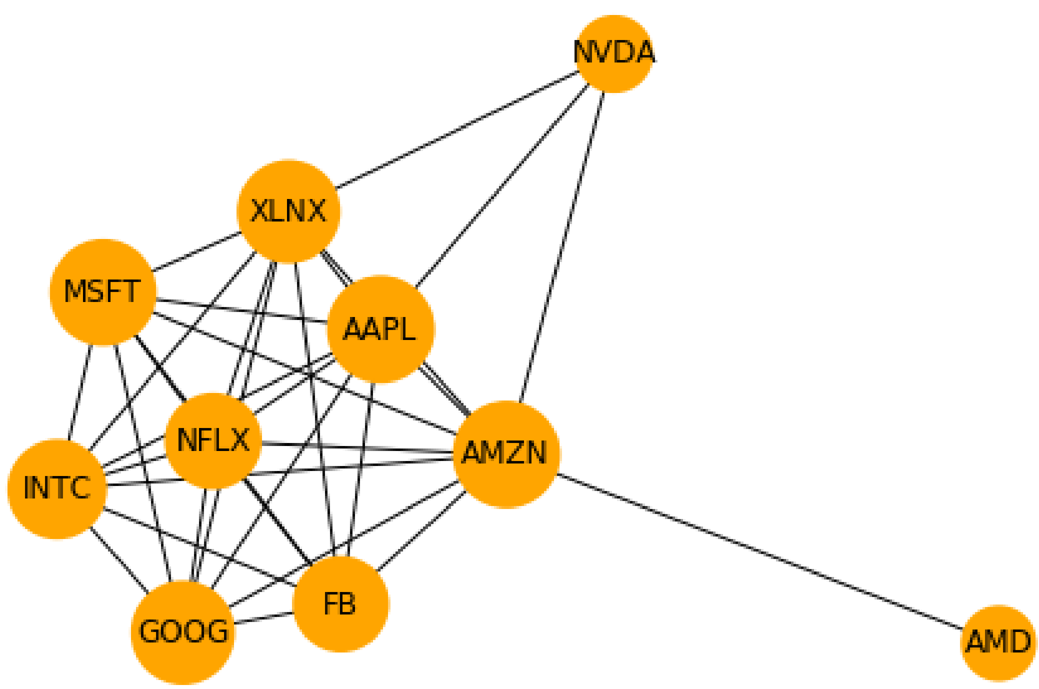
Volatility Correlations (average excludes self-correlation)

[10]:

	GOOG	AMZN	AAPL	FB	NFLX	MSFT	AMD
GOOG	1.000000	0.714537	0.741024	0.572596	0.682515	0.710255	0.304546
AMZN	0.714537	1.000000	0.779052	0.569243	0.598572	0.840682	0.394923
AAPL	0.741024	0.779052	1.000000	0.600365	0.613508	0.768164	0.381921
FB	0.572596	0.569243	0.600365	1.000000	0.435825	0.536743	0.205624
NFLX	0.682515	0.598572	0.613508	0.435825	1.000000	0.666182	0.340922
MSFT	0.710255	0.840682	0.768164	0.536743	0.666182	1.000000	0.305784
AMD	0.304546	0.394923	0.381921	0.205624	0.340922	0.305784	1.000000
INTC	0.578917	0.615752	0.618152	0.660769	0.520810	0.684378	0.108327
NVDA	0.336742	0.414943	0.420414	0.224103	0.134205	0.338858	0.241030
XLNX	0.581071	0.618871	0.645758	0.588869	0.467038	0.618842	0.382492

\

	INTC	NVDA	XLNX	average
GOOG	0.578917	0.336742	0.581071	0.522220
AMZN	0.615752	0.414943	0.618871	0.554658
AAPL	0.618152	0.420414	0.645758	0.556836
FB	0.660769	0.224103	0.588869	0.439414
NFLX	0.520810	0.134205	0.467038	0.445958
MSFT	0.684378	0.338858	0.618842	0.546989
AMD	0.108327	0.241030	0.382492	0.266557
INTC	1.000000	0.246496	0.721005	0.475461
NVDA	0.246496	1.000000	0.471256	0.282805
XLNX	0.721005	0.471256	1.000000	0.509520



Results The first observation is that AMDs only connection is now with Amazon, not Nvidia. As AMD and Nvidia are direct competitors, when one asset is unstable, this graph suggests that the other is stable. To conclude, while they compete, uncertainty in one does not imply it for the other.

Every large market-cap asset has a connection to every other large market-cap asset in this graph, unlike the previous. The volatility in the US market can be said to be driven by the volatility in its largest components, which includes these very large tech assets. The complete sub graph is then explained by the common factor being general market volatility.

Xilinx is now connected to every large cap, compared to three in the previous graph. This shows that XLNX is volatile when the market is volatile, and stable when the market is stable.

All the assets are part of the SP500, so the resulting graph could have been expected. It is interesting to see that both AMD and NVIDIA have fewer connections than the other assets. The next step would be to investigate non SP500 constituents with this volatility-correlation graph method. For other assets that correlate a lot with the large-cap techs, the VIX could be used as a volatility indicator for the small-cap asset, to purchase an options spread that profits when prices go either above a strike or below another strike. The idea being that the small-cap asset would also become volatile and fall into and in-the-money position, but delayed in volatility enough for the options premium to be on discount. A lead-lag on rolling StdDev could be run between the SP500 and the small-cap asset once identified by the graph method. If the SP500 leads by a significant enough amount for trades to be made, then a trading opportunity would be discovered. (An alternate would be to use the same strike price, with the aim of the sum of intrinsic value netting to 0 after the move, but the increase in volatility giving gains to the sum of the extrinsic values).

Re-Ordering Re-ordering the time-series data would effect the values in the correlation matrix, as the rolling StdDev could change differently for each asset. Thus the correlation between the assets is not guaranteed to be the same, thus the weights of the edges in the graph would be different.

Re-ordering the vertices would change the results for the same reason as previously discussed.

5.5 Considering Raw Prices

Raw prices are not zero-mean and cannot be assumed to be normal. In fact, the prices will contain trends that lead to short/long tail distributions, depending on the trend. Correlation measures how one variable moves against its average, as the other moves against its average.

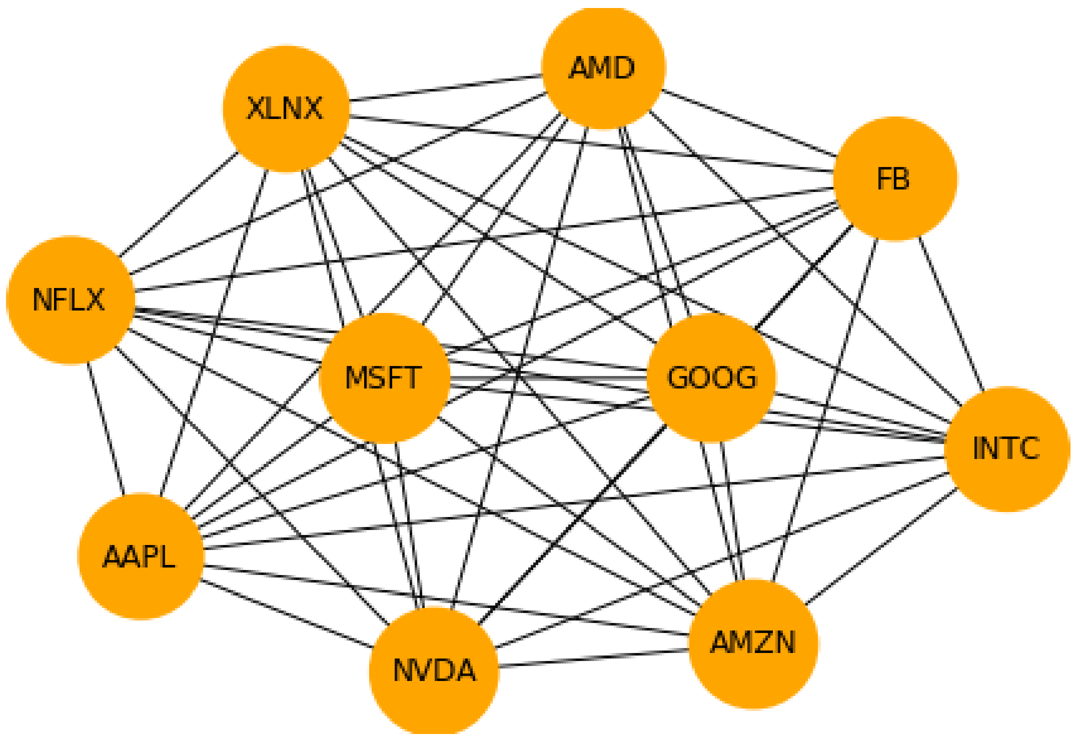
So raw prices cannot be used. With an example, if both assets A dand B rise with the same trend, then their prices would have positive correlation. However, while A increases, B can decrease, yet both increase by the same amount overall in the time period. This would produce negative correlation of returns, yet postive correlation of prices. Taking returns can be considered to remove the misleading trend, when considered for correlation.

The graph resulting from the correlation matrix of raw prices, would have more edges and the vertices would be closer together. This is due to the high price trend of the SP500 over the time period.

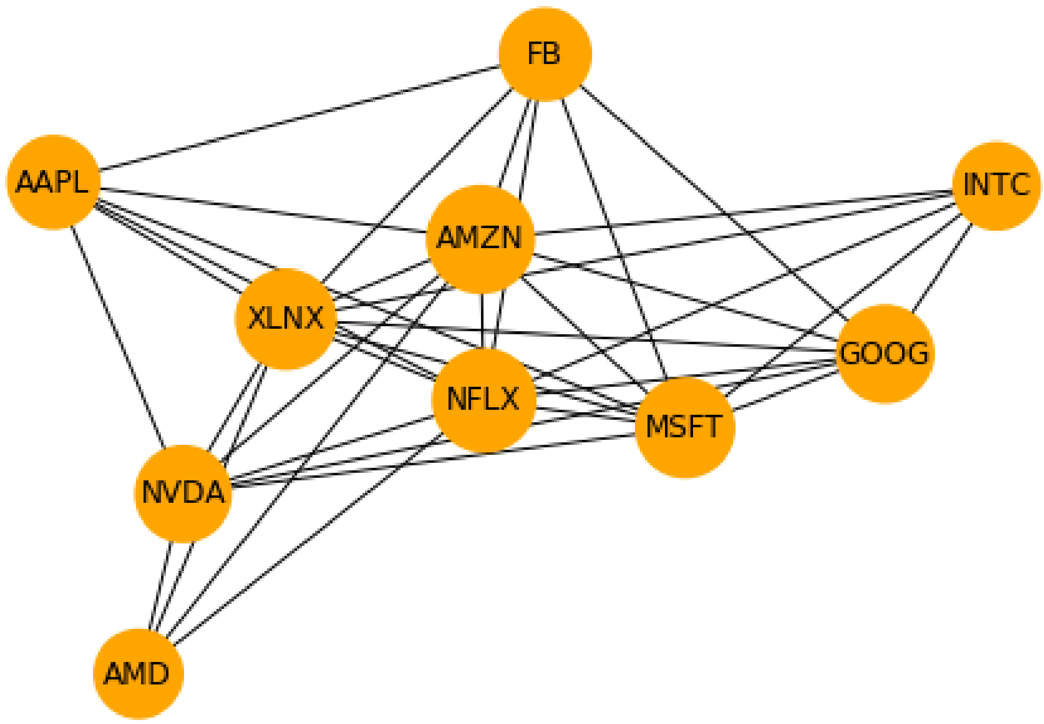
The graph resulting from the volatility correlation matrix of raw prices would also have more connections and closer vertices. As the price would deviate less from the average price, compared to the return deviating from the mean return. So all the stocks would have lower rolling volatility. However, this graph should be less connected than the correlation of raw prices, as the underlying volatility should still come show in the results.

Both of the graphs have been plotted below, confirming the above expectations.

Graph of Raw Prices Correlation



Graph of Raw Prices Volatility Correlation


FB
AAPL
INTC
AMZN
XLNX
GOOG
NFLX
MSFT
NVDA
AMD