FM320 Summative Project

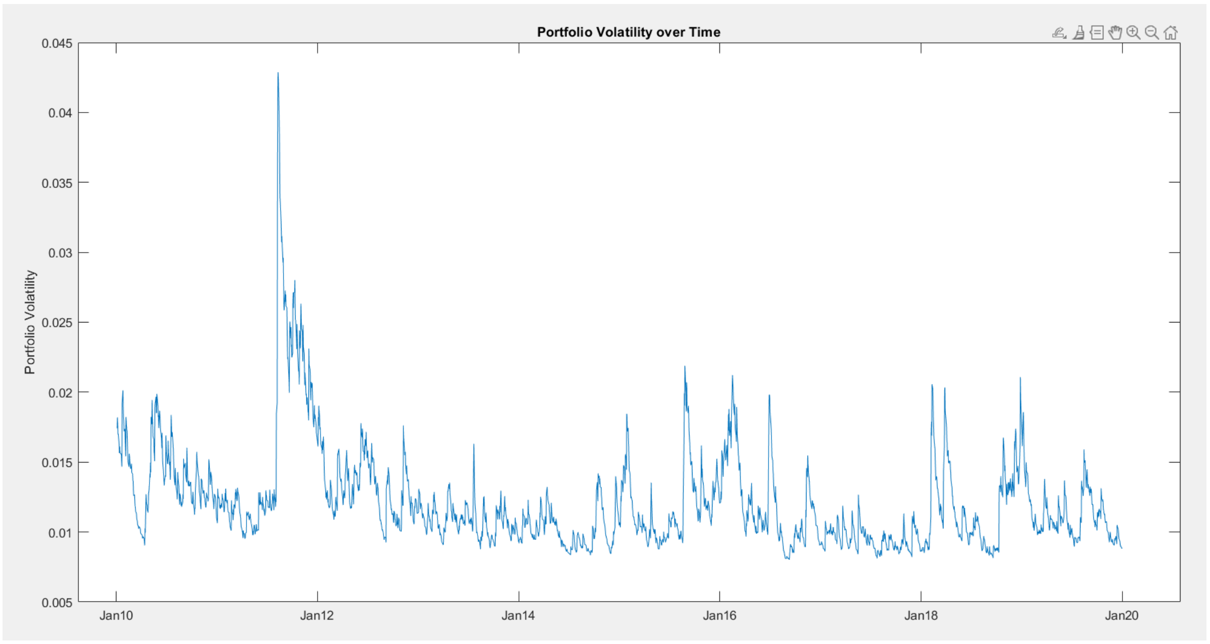
Portfolio Selection

I have selected **MSFT**, **JPM**, **BAC**, and **UNH** for my portfolio for several reasons. I anticipate that **JPM**, and **BAC** to have one of the highest pairwise correlations among the options which will be interesting to investigate. In hindsight, we also know that the technology sector has performed extraordinary well in the past few years therefore I selected **MSFT** but no other tech stocks. Finally, I decided that the last stock should be as little correlated with the others as possible which is **UNH**, in my opinion.

Model Choice

$$\Sigma_t = D_t C_t D_t$$

I decided to use DCC with GARCH(1,1) as the univariate model since it does not face severe dimensionality problems compared to other models. Dimensionality is an important factor when selecting models since higher dimensionality means we need to estimate more parameters hence larger inaccuracies in results. I decided to use GARCH(1,1) over EWMA since GARCH is mean reverting. In fact, GARCH(1,1) is the most recognised in the industry.



This graph shows that the volatility returns to a mean after volatile periods. When estimating the conditional correlations matrix C_t, we standardize our returns otherwise it would rely too heavily on volatile periods. We want the different periods to be equally informative about the correlations between the stocks.

Methodology

I subtracted the mean of returns of each stock from all the returns so that we can calculate the conditional covariance. The reason why we de-mean our returns is like the reason why we subtract the mean when calculating sample variance, and sample covariance. We are interested in how the returns deviate from the mean rather than how they deviate from 0.

I then applied the ***dcc*** function from the MFE-toolbox. The 2516th entry (last entry) of hDCC is the conditional covariance matrix for day 2517 (2nd January 2020) because we do not have an estimate for day 1 since there is no past data. I calculated the conditional variance of the portfolio using:

$$\mathbf{w}\Sigma\mathbf{w}'$$

where $\mathbf{w}$ is a 1 x 4 vector containing the portfolio weights. To get the conditional volatility, I calculated the square root of the conditional variance.

Since we do not have the portfolio value, we are unable to calculate the absolute VaR hence I have left it in terms of return. To calculate the absolute VaR, you would simply multiply it by the portfolio value of the day before the day of interest.

For the back testing, I decided to use a window length of 100 days because we ideally want to see at least 3 violations of the VaR at 5%. Therefore, we want a window length of at least 3/0.05 = 60 but I decided to use 100 days to allow for more leniencies hence obtain better estimates.

To calculate the multivariate conditional covariance matrix using only the previous 100 days, I extracted the returns for the last 100 days and carried out ***dcc*** on this smaller sample as above. Then I calculated the portfolio returns for the following day by multiplying the corresponding row by the weightings. If the portfolio returns were lower than -VaR, then we have violation which we assign the value 1 to otherwise we set it to 0. After looping through everyday up to day 2515, we obtain a hit sequence of 1s and 0s. We stop calculating the VaR at day 2515 because day 2516 is the very last day which we know the returns for.

To carry out the unconditional coverage test, I first needed to find the restricted and unrestricted log likelihood functions. Since the hit rate follows a Bernoulli distribution, this was very easy to do. I used the MLE to estimate ***p***. The test statistic follows a chi-squared distribution with 1 degree of freedom.

$$-2(\log L_R - \log L_U) \sim \chi^2$$

Results and Conclusions

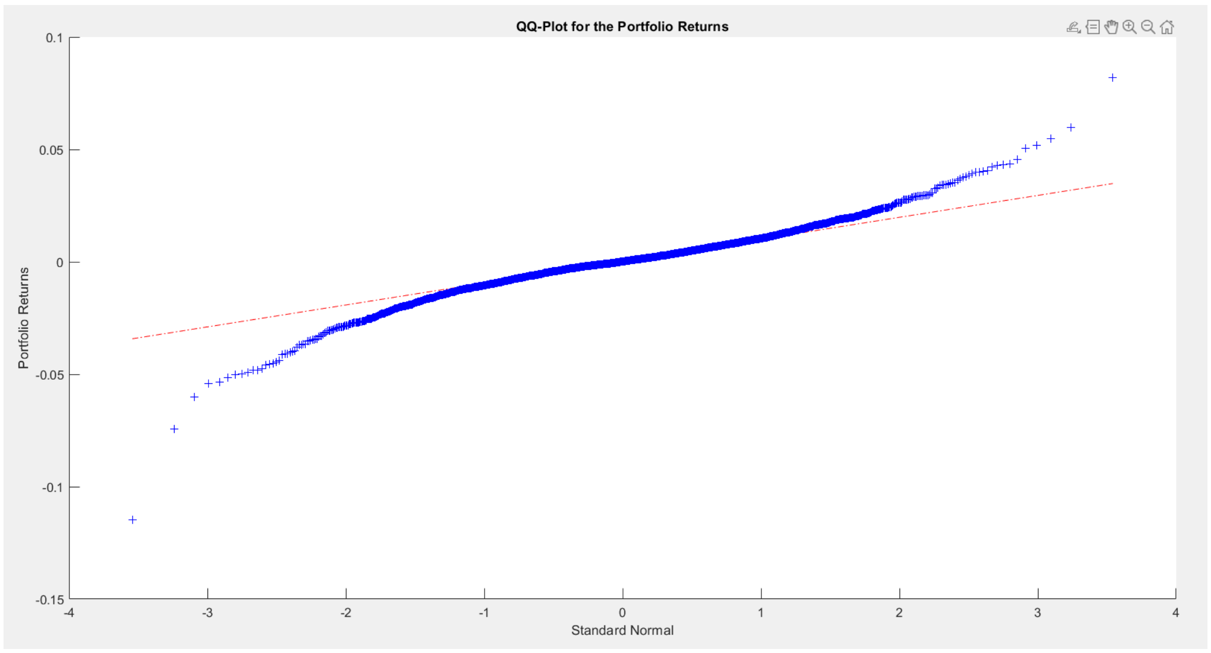
```
last_Cov =

   1.0e-03 *

    0.1341    0.0516    0.0549    0.0494
    0.0516    0.1084    0.1064    0.0472
    0.0549    0.1064    0.1450    0.0543
    0.0494    0.0472    0.0543    0.1436
```

As anticipated, JPM and BAC have the highest conditional correlation since their conditional covariance is the greatest. However, the correlation between all the other combinations are very similar which may be due to the market factor. This can be investigated further using PCA.

To calculate the estimated Value-at-Risk, I multiplied the inverse normal for the p-value (5%) by the conditional volatility for the first day of 2020. I have assumed that the returns have followed a normal distribution however this is often not the case. We have seen that the S&P500 have fatter tails with the majority of returns very close to zero. This can be shown by the QQ-plot below.



An alternative to using the normal distribution would be the t-distribution. This accounts for the fatter tails however it also has fixed variance. For example, t(4) has a variance of 2 whilst our returns may have a different variance hence each distribution has their own advantages.

After backtesting for 2415 different windows, I obtained the following results:

p_hat = 0.0518

h = 0

p-value = 0.6931

test statistic = 0.1557

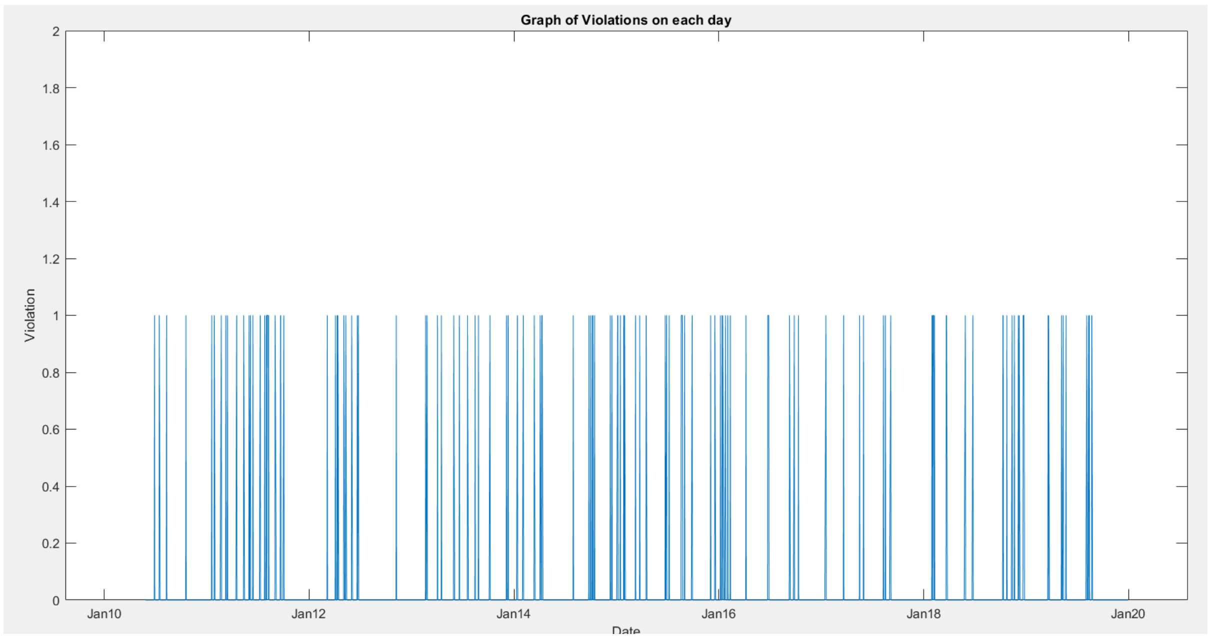
critical value = 3.8415

The MLE for ***p*** is slightly higher than 5% implying we have more than 5% violations. Based on the unconditional coverage test, we reject the null hypothesis since the test statistic is significantly less than the critical value. We conclude that the violations are most likely IID with a probability of occurrence of 5%.

Critiques and Improvements

Although VaR can be relatively easy to compute, it is not a coherent risk measure since it often violates sub-additivity when there are very fat tails. An alternative to VaR is Expected Shortfall which is harder to estimate since it requires the distribution of the returns. Expected Shortfall captures information regarding the entire tail of the distribution hence it is more informative when it can be accurately estimated.

To investigate whether violations cluster, I have plotted a graph of the hit sequence.



Looking at the graph, you can identify that there are periods with many violations. From this we can see that violations may depend on whether a violation occurred on the day before. To test this, we can use the conditional coverage test.

Appendix

The following section calculates the multivariate conditional covariance for the returns on the 02/01/2020 then uses this to compute the VaR.

```
clear
clc


% Reading dataset
[data, dates] = xlsread('Project Data MT 2021.xlsx', 'D9:Z2524');


% Extracting returns for MSFT, JPM, BAC, and UNH respectively
returns = data(:,[1,10,12,19])


% I extracted the dates then converted them to MATLAB date serial numbers
dnr = dates(:,23)
dnr2 = datenum(dnr)


% De-meaning the returns
returns = returns - mean(returns)


% Using dcc function from MFE-Toolbox to calculate the conditional covariance
matrix
[parDCC,~,hDCC] = dcc(returns,[],1,0,1);
```

```
% Extracting conditional covariance matrix for 02/01/2020
last_Cov = hDCC(:,:,2516)


% Defining the portfolio weights
w = [0.25, 0.25, 0.25, 0.25]


% Calculating portfolio volatility for the last day
last_Vol = sqrt(w*last_Cov*w')


% By assuming normality of returns, I have calculated the 5% VaR in returns
last_VaR = -norminv(0.05)*last_Vol
```

Plotting Portfolio Volatility

```
portfolio_volatility =[]
% Extracting portfolio conditional volatility for each day
for i=1:2516
portfolio_volatility(i,:) = sqrt(w*hDCC(:,:,i)*w')
end


% Plotting portfolio volatility against date
figure()
plot(dnr2,portfolio_volatility)
datetick('x',12,'keeplimits')
ylabel('Portfolio Volatility')
title('Portfolio Volatility over Time')
```

QQ-plot for Normality of Portfolio Returns

I calculated the portfolio returns by multiplying the returns by the portfolio weights. I then created a QQ-plot of the portfolio returns against the normal.

```
% Calculating Portfolio returns for each day
portfolio_returns = returns * w'


figure()
qqplot(portfolio_returns)
title('QQ-Plot for the Portfolio Returns')
xlabel('Standard Normal')
ylabel('Portfolio Returns')
```

Backtesting VaR on a 100 day window

```
% Allocating empty vector to store the hit sequence for VaR violations
hit_seq = []


% I am using a window length of 100 days
window = 100


% Creating for loop to calculate VaR for each day from day 101 onwards
for i = 101:2515
    % Extracting the returns for the window of interest
    returns2 = returns((i-window):(i-1),:)
    [parDCC2,~,hDCC2] = dcc(returns2,[],1,0,1)


    Cov = hDCC2(:,:,100)
    Vol = sqrt(w*Cov*w')


    % Calculating VaR at day i
    VaR = -norminv(0.05)*Vol


    % Calculating the portfolio returns for day 'i+1'
    portfolio_returns = returns(i+1,:)*w'

    % Checking if returns on day i+1 violate the VaR calculated from day i
    if portfolio_returns < -VaR
        hit_seq(i-100) = 1
    else
        hit_seq(i-100) = 0
    end
end


count= sum(hit_seq == 1)
miss = sum(hit_seq == 0)


% Calculating MLE of p for bernoulli
p_hat = count/(2516-101)


% Calculating restricted and unrestricted likelihood
uLogL = log(((1-p_hat)^miss)*(p_hat^count))
rLogL = log(((1-0.05)^miss)*(0.05^count))


% Carrying out log likelihood test
```

```
[h,pValue,stat,cValue] = lratiotest(uLogL,rLogL,1)


% Extracting dates for which we have tested for violations
dnr3 = dnr2(102:2516,:)


% Plotting the hit sequence to check for clusters
figure()
plot(dnr3, hit_seq)
datetick('x',12,'keeplimits')
xlabel('Date')
ylabel('Violation')
title('Graph of Violations on each day')
ylim([0, 2])
```