Table of Contents

FIN279_MIDTERM

Description for the entire folder.

1. The files required to submit are the follows. * *FIN279_MIDTERM_WFENG.m* * *FIN279_MIDTERM_WFENG.pdf* * *Workspace.mat* * *fin279_midterm.xlsx*

2. The self-created functions files are as follows. *myimport.m*, *logReturn.m*, *volumeChange.m*, *MAmean.m*, *EWMAvolatility.m*, *myBootstrap.m*, *myplot.m* and *myBootstrapPlot.m*.

3. Additional files for references. * For convnenience, I also integrate the function in the *AllFunctionFils.m* and *AllFunctionFiles.pdf*.

Description for this script.

1. This script is the one can analyze all stock of the sheet in excel and includes all the user defined functions I created.

2. About the raw data I used in this script. I have some problem on the raw data downloaded on Oct. 27th, so I re-download the data on Nov. 3rd. so the data might not be total the same, but that doesn't influence the function of this script, because it can do the same analysis for any excel files.

Loop for several stocks

Built-in function input() can be used here to replace the following three lines to ask user input the filename and sheetname and columnsum (the the total columns with data in your sheet). As solving specific file, I do it as follows for convenience so that you do not need to input again.

```
filename = 'fin279_midterm.xlsx';
sheetname = 'Sheet1';
columnsum = 12;
[~,text] = xlsread(filename,sheetname);
```

```
% This for-loop is the most outside one. You can find the end line around
% line 427.
for n = 1:3:(columnsum-2)

    % ii is used to mark the ordinal of fiugres.
    % columnsum=12;stocknum=4; [1:4]=(n+2)/3; [1:12:37]=12*([1:4]-1)+1=ii;
    ii = 2*n-1;
    stock = text(1,n);
```

PART 1) Volatiility and Volume Analysis

In this part, I used the functions created by myself. They are the following.

- myimport(), which is used to import data.
- logReturn(), which is used to compute the log returns.
- volumeChange(), which is used to compute the change in daily trading volume.
- MAmean(), which is used to compute the moving average mean in two ways.
- EWMAvolatility(), which is used to compute the EWMA volatility.

a) clean raw data.

- In this step, we have to go to excel and clean the sheet that we use.
- In the sheet, each stock should has 3 columns in specific order i.e. [dates, price and volume, and the stock ticker should be right above the first comlumn, which is dates.
- We should not leave any columns of blanks between each stocks.

b) Import data into Matlab.

```
[dates,price,volume] = myimport(stock,filename,sheetname,columnsum);
```

c) Calculate daily log returns.

- The following is the solution for part1-c.
- *log_returns* is the log returns of the stock.
- *vlm_change* is the daily change in trading volume.

```
log_returns = logReturn(price);
vlm_change = volumeChange(volume);
```

To compute the third variable.

The third varable required to build is the price in rate of change. V-ROC is the third variable I want to calculate. Refer to investopedia.com, Rate of change (ROC) stands alone as an important indicator used by many technicians interested in market momentum. It shows the speed at which a variable changes over a specific period of time, and can give traders a sense of how two variables change in relation to each other

and at what speed. Long-term views of the market or a specific sector or stock will use perhaps a 26- to 52-week time period for Yx and a shorter view would use 10 days to around six months.

Here I will use 252/2 datys time period for Yx, which mean the closing price x days ago.

```
pROC = NaN(fix(length(price)/126),1);
for i = 2:fix(length(price)/126)
    pROC(i) = price(126*i)./price(126*(i-1));
end
```

d) Using the EWMA volatility estimation function.

This part will use self-created function to do the calculations.

To compute the normal mean of log return and the exponentially weighted average volatility. The following is the solution for part1-d-1. *UWMAmu_LogRet* is normal mean of log return. *sigma1* is Exponentially weighted average volatility of log return using normal mean.

```
UWMAmu_LogRet = MAmean(log_returns,90);
sigma1 = EWMAvolatility(log_returns,UWMAmu_LogRet,90,0.94);
```

To compute the exponentially weighted average volatility. The following is the solution for part1-d-2. *EWMAmu_LogRet* is weighted average mean. *sigma2* is exponentially weighted average volatility of log return using weighted average mean.

```
EWMAmu_LogRet = MAmean(log_returns,90,0.94);
sigma2 = EWMAvolatility(log_returns,EWMAmu_LogRet,90,0.94);
```

To compute the mean of daily trading volume. The following is the solution for part1-d-3. *mu_vlm3* is the mean of daily trading volume.

```
mu_vlm3 = MAmean(volume,200,0.94);
```

To compute the mean of the daily change in trading volume. The following is the solution for part1-d-4. *mu_vlm3* is the mean of the daily change in trading volume.

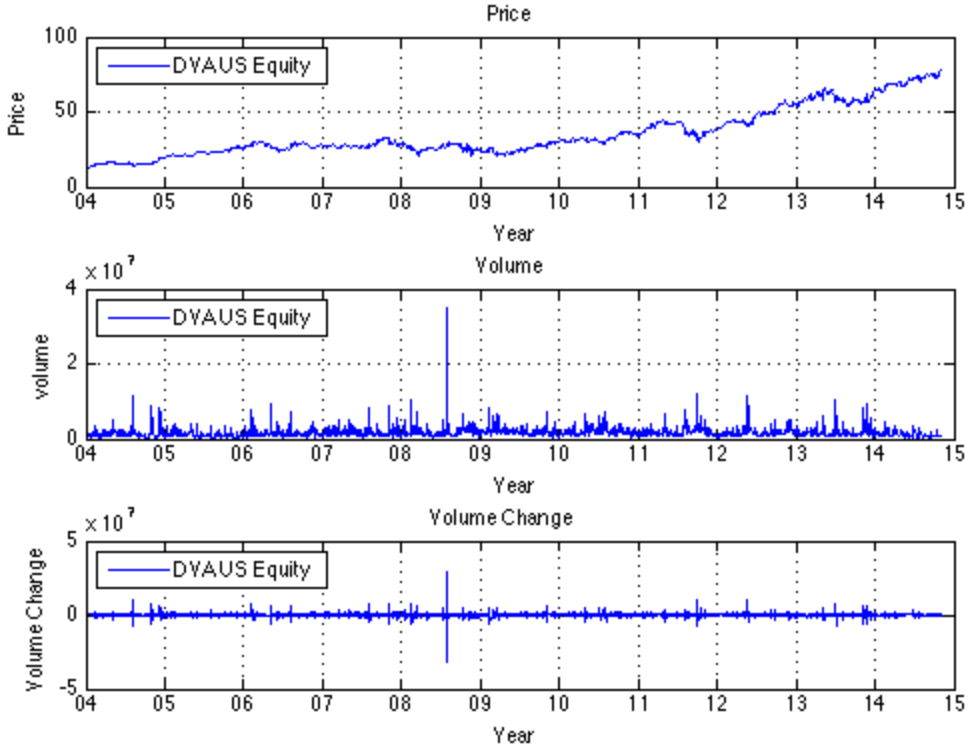
```
mu_vlm_change4 = MAmean(vlm_change,200,0.94);
```

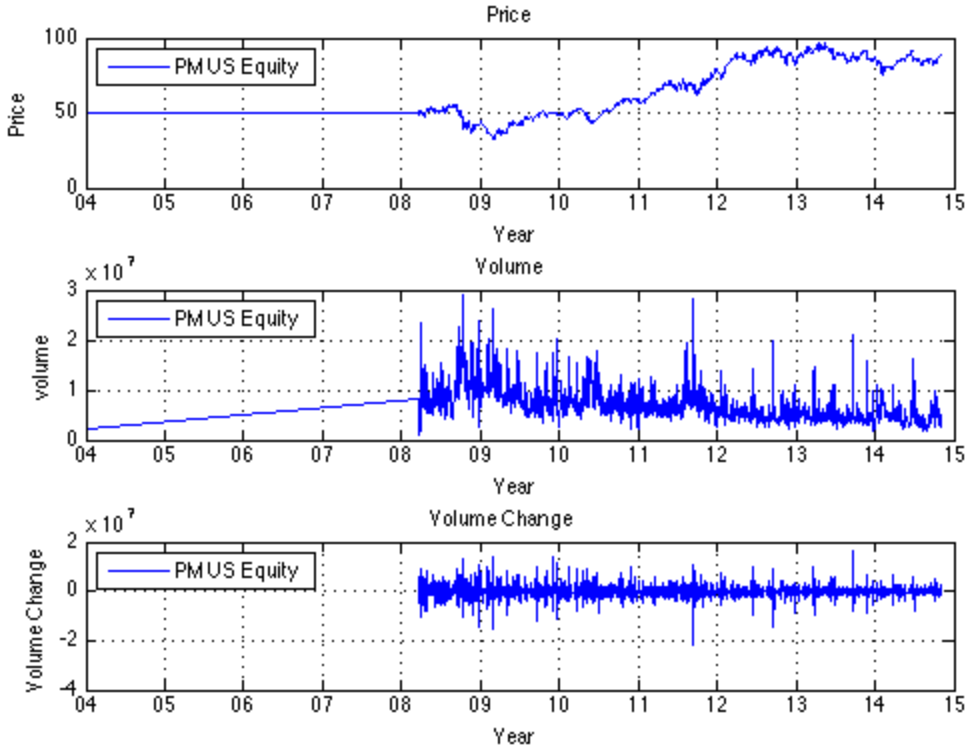
e) Plot and Explain.

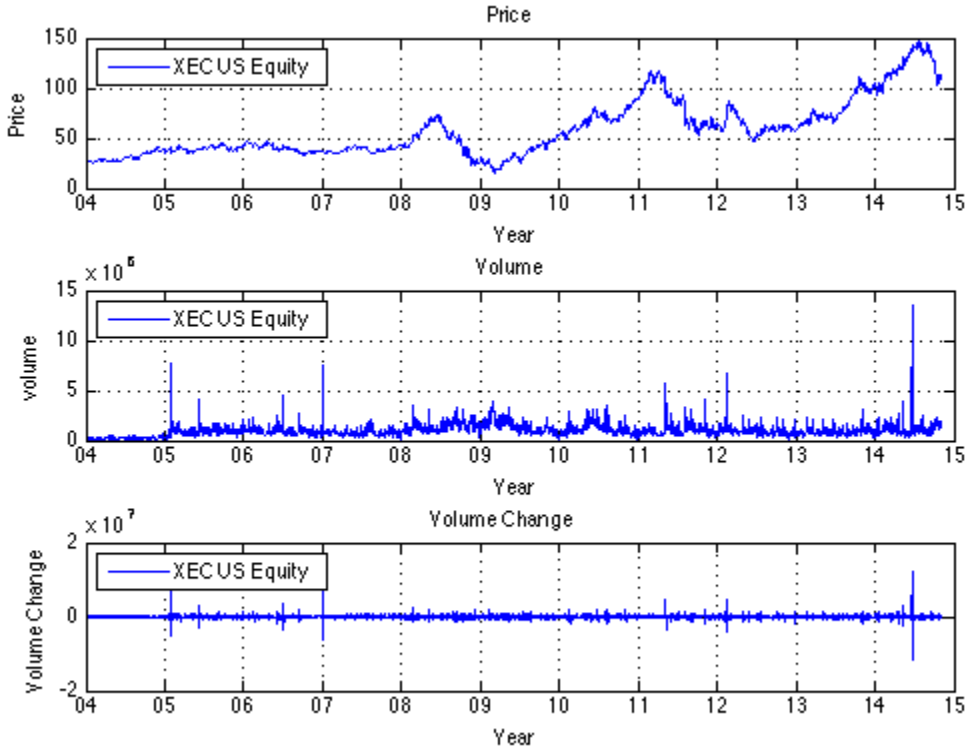
Plot daily stock price, the daily trading volume and the change in daily trading volume.

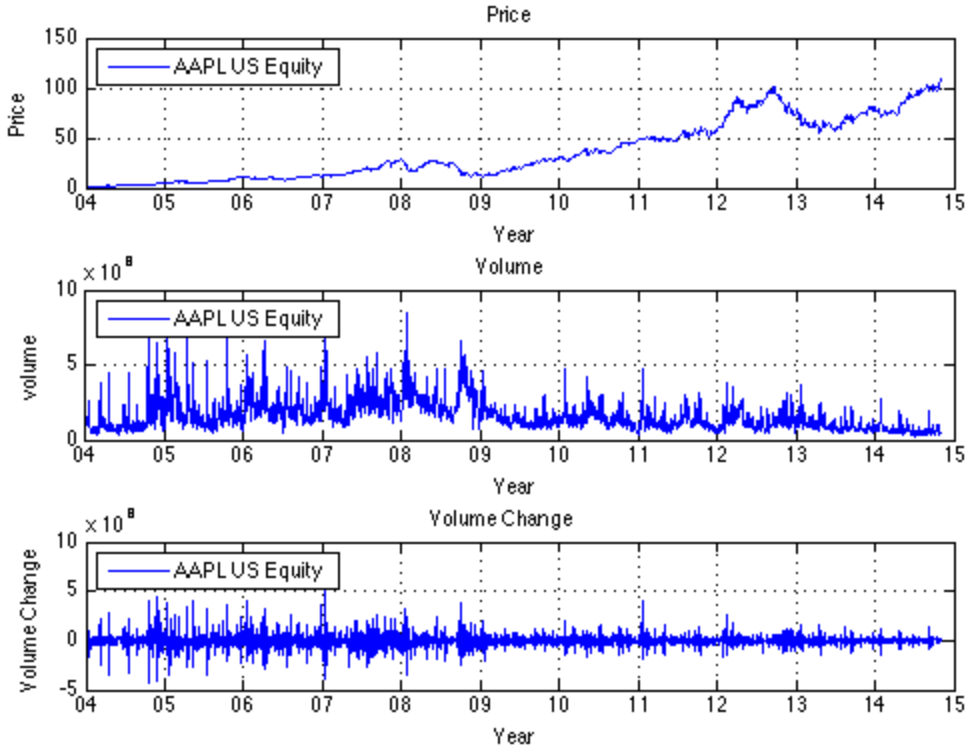
```
figure(ii)

subplot(3,1,1)
plot(datenum(dates), price);
grid on
datetick('x','yy')
xlabel('Year')
ylabel('Price')
title('{\bf Price}')
legend(stock,'Location','NW')

subplot(3,1,2)
```

```
plot(datenum(dates), volume);
grid on
datetick('x','yy')
xlabel('Year')
ylabel('volume')
title('{\bf Volume}')
legend(stock,'Location','NW')

subplot(3,1,3)
plot(datenum(dates), vlm_change);
grid on
datetick('x','yy')
xlabel('Year')
ylabel('Volume Change')
title('{\bf Volume Change}')
legend(stock,'Location','NW')
```



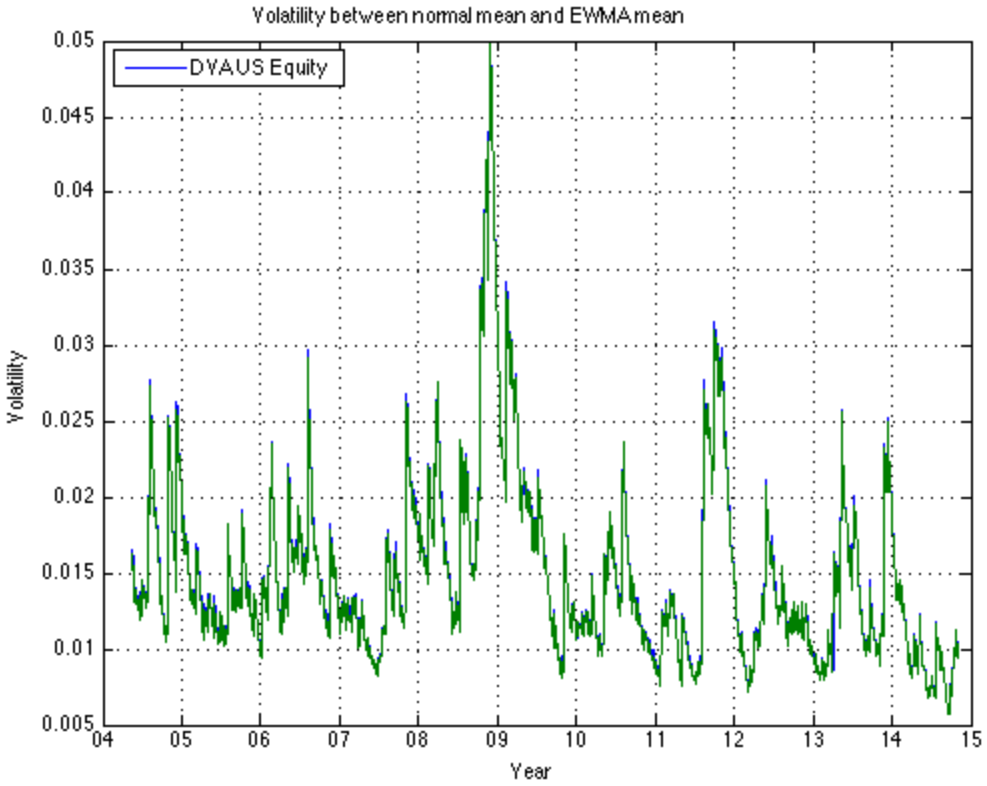

Price
PM US Equity
Price
Year
Volume
x 10^7
PM US Equity
volume
Year
Volume Change
x 10^7
PM US Equity
Volume Change
Year

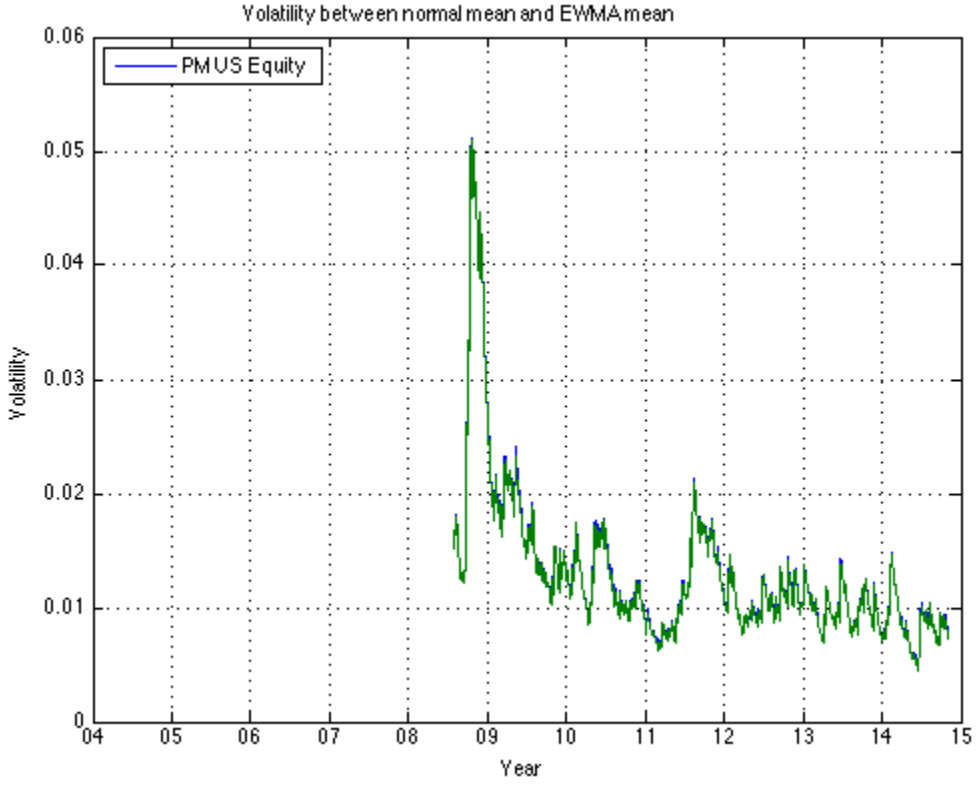

Price
XEC US Equity
Price
Year
Volume
x 10^6
XEC US Equity
volume
Year
Volume Change
x 10^7
XEC US Equity
Volume Change
Year

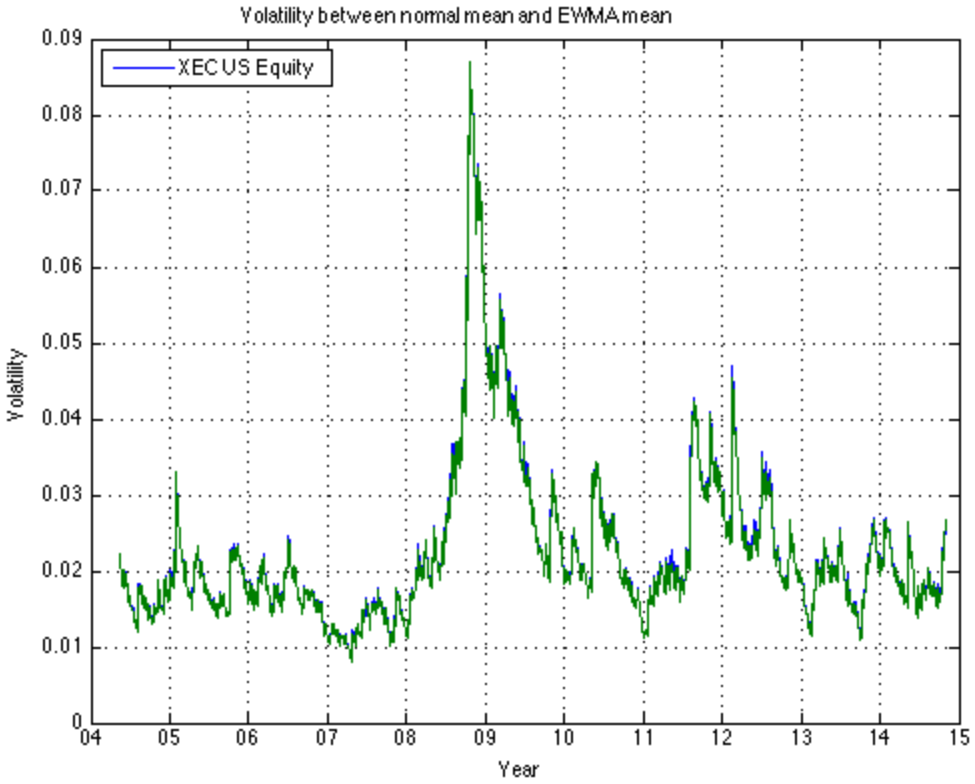


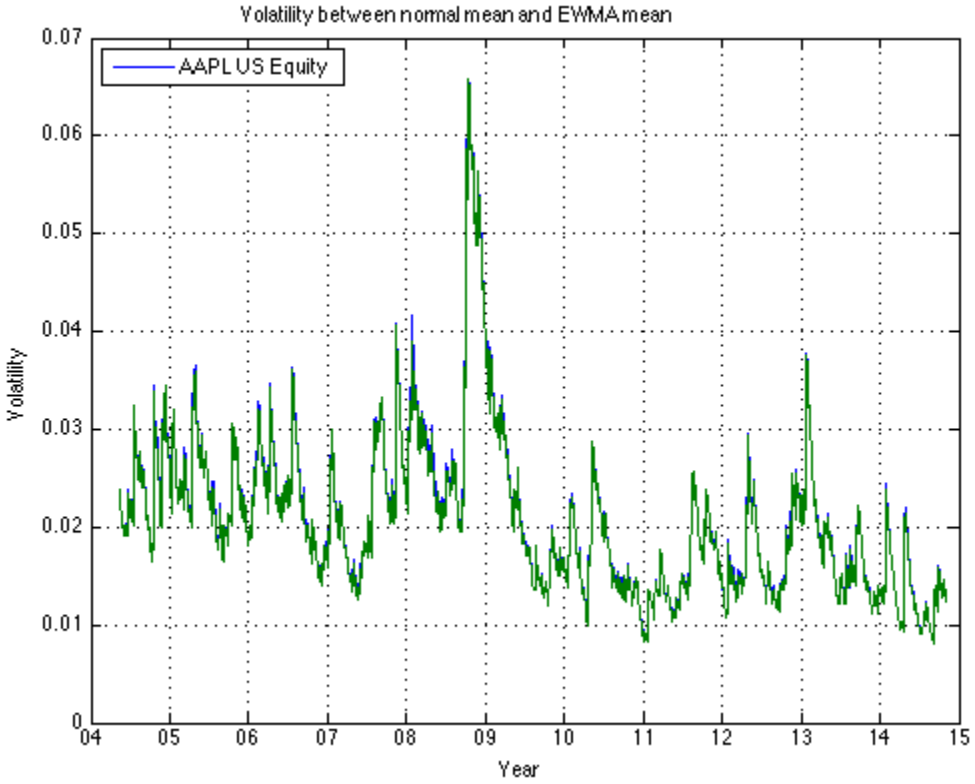
Plot and compare the Volatility between normal mean and exponentially weighted average mean.

```
figure(ii+1)
plot(datenum(dates), [sigma1,sigma2]);
grid on
datetick('x','yy')
xlabel('Year')
ylabel('Volatility')
title('{\bf Volatility between normal mean and EWMA mean}')
legend(stock,'Location','NW')
```


Volatility between normal mean and EWMA mean
DVA US Equity
Volatility
Year
0.05
0.045
0.04
0.035
0.03
0.025
0.02
0.015
0.01
0.005
04
05
06
07
08
09
10
11
12
13
14
15


Volatility between normal mean and EWMA mean
PM US Equity
Volatility
Year
0.06
0.05
0.04
0.03
0.02
0.01
0
04
05
06
07
08
09
10
11
12
13
14
15


Volatility between normal mean and EWMA mean
XEC US Equity
Volatility
Year
0.09
0.08
0.07
0.06
0.05
0.04
0.03
0.02
0.01
0
04
05
06
07
08
09
10
11
12
13
14
15


Volatility between normal mean and EWMA mean
AAPL US Equity
Volatility
Year
0.07
0.06
0.05
0.04
0.03
0.02
0.01
0
04
05
06
07
08
09
10
11
12
13
14
15

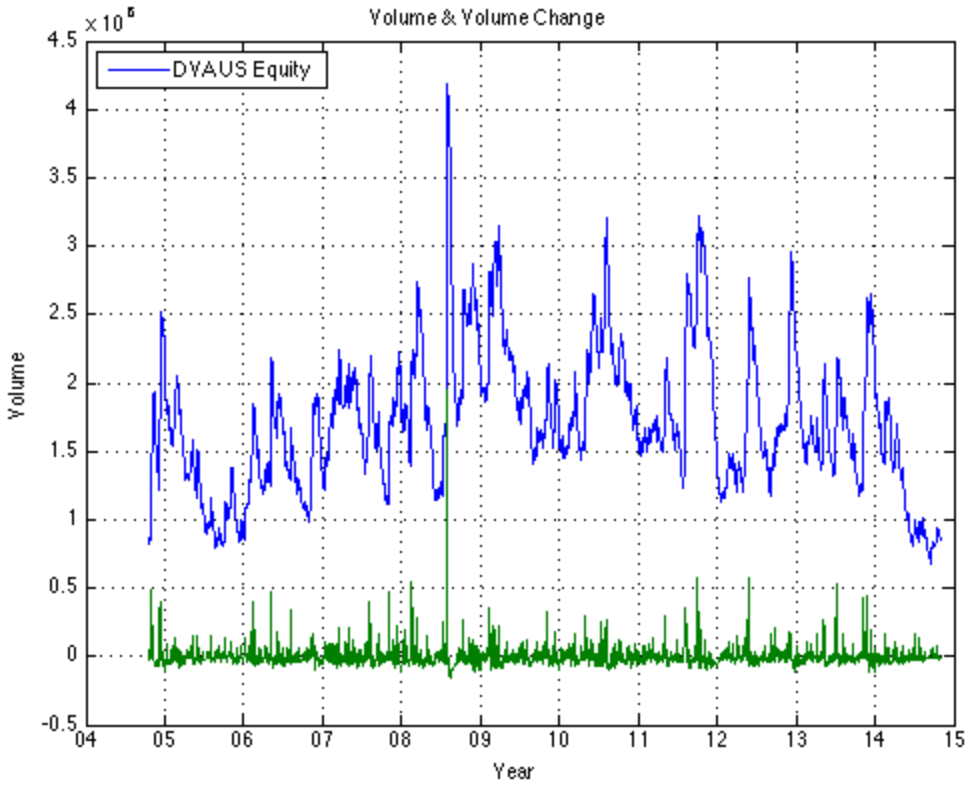
After observation of the volatility between normal mean and EWMA mean, I find the following.

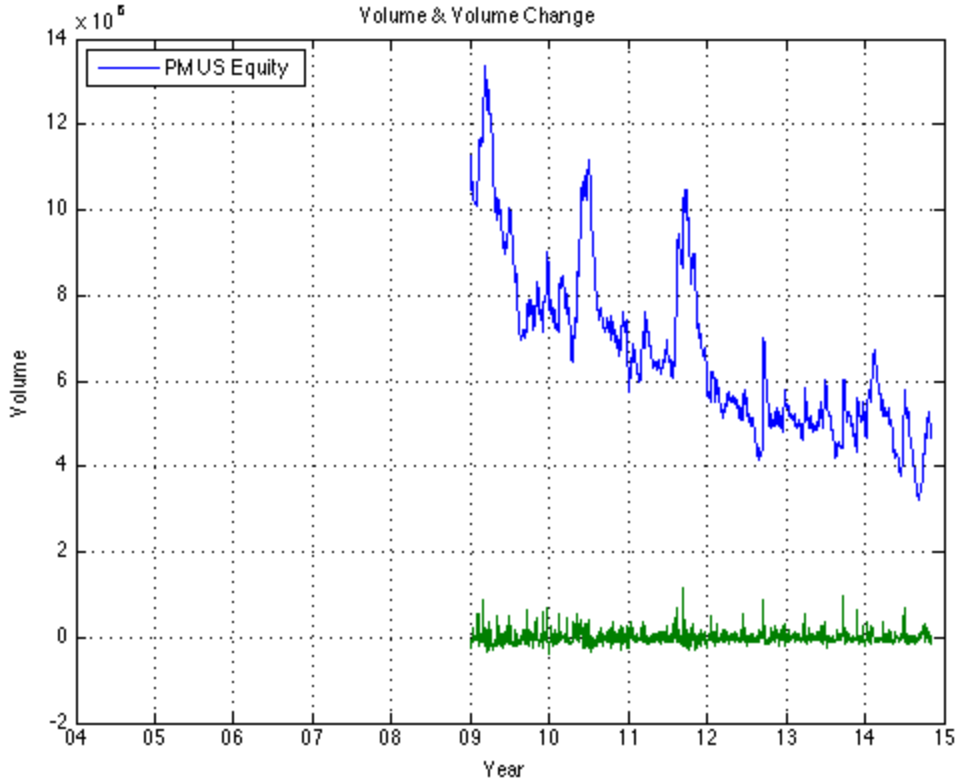
1. The two line are nearly stay in the same position as the other at each point, which means they are very close.

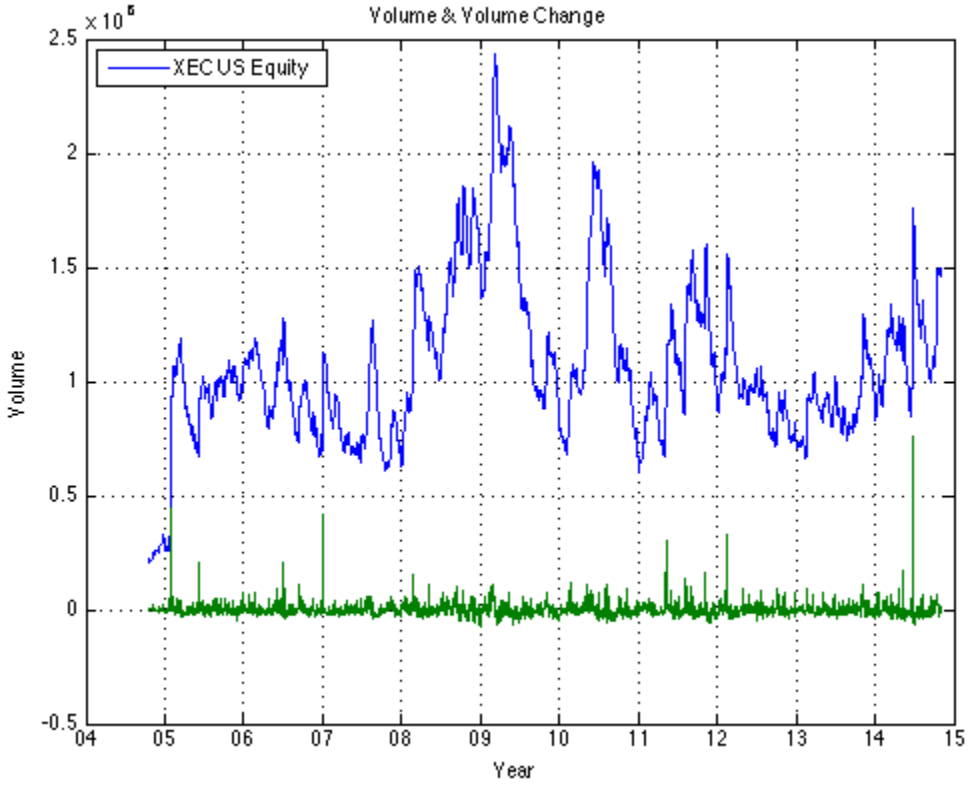
2. According to the graph, we can easily conclude that the results of computing EWMA volatility with normal mean or weighted mean are almost the same.

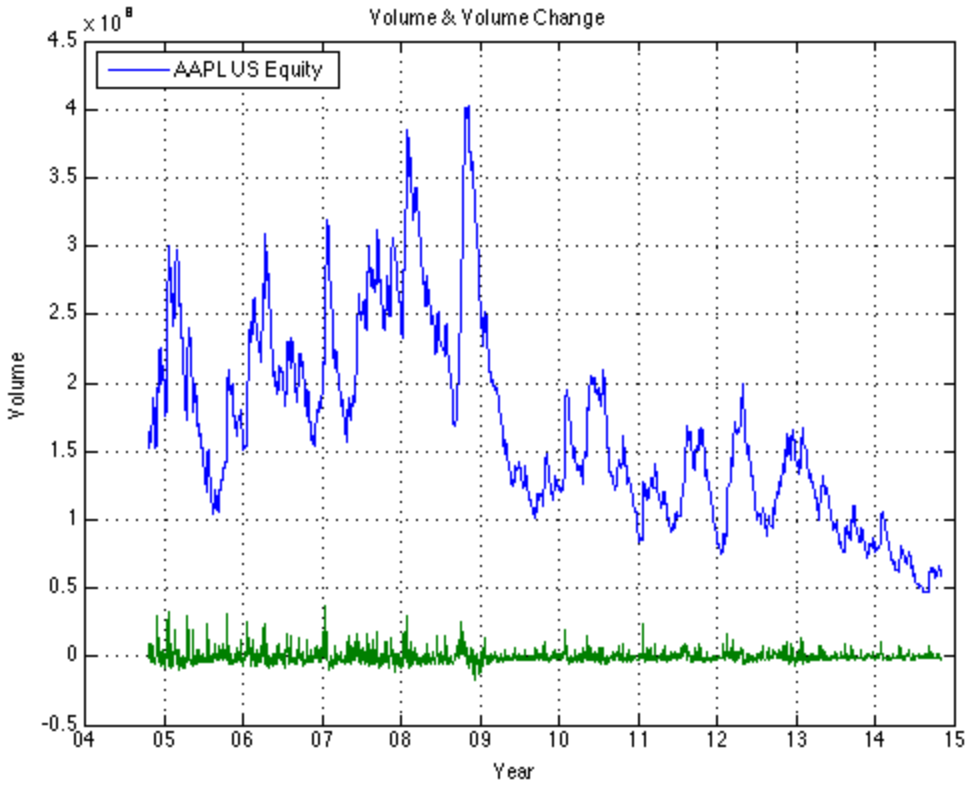
Plot and compare the volume and the change in volume.

```
figure(ii+2)
plot(datenum(dates), [mu_vlm3,mu_vlm_change4]);
grid on
datetick('x','yy')
xlabel('Year')
ylabel('Volume')
title('{\bf Volume & Volume Change}')
legend(stock,'Location','NW')
```




Volume & Volume Change
x 10^6
PM US Equity
Volume
Year
14
12
10
8
6
4
2
0
-2
04
05
06
07
08
09
10
11
12
13
14
15


Volume & Volume Change
x 10^6
XEC US Equity
Volume
Year
2.5
2
1.5
1
0.5
0
-0.5
04
05
06
07
08
09
10
11
12
13
14
15



After observation of the volume and the change of volume, I find the following.

1. The EWMA mean of daily trading volume is more smooth than the EWMA mean of the daily change in trading volume. (for convenience, I will describe one as upper line, the other as lower line.

2. The move of the lower one is followed by the move of the upper one.

3. As time went by, the EWMA mean of daily trading volume gradually becomes comparatively stable and the EWMA mean of the daily change in trading volume keep nearly constant as well.

4. Basically, volume is volatile around 2009 for each stock because of the financial crises.

PART 2) Bootstraping for Volatility.

In this part, I used the functions created by myself. They are myBootstrap() and myplot(). I also create a myBootstrapPlot function, which can do both of myBootstrap() and myplot(). And that function is simple as well.

a) to d). Bootstraping, confidence interval and timer.

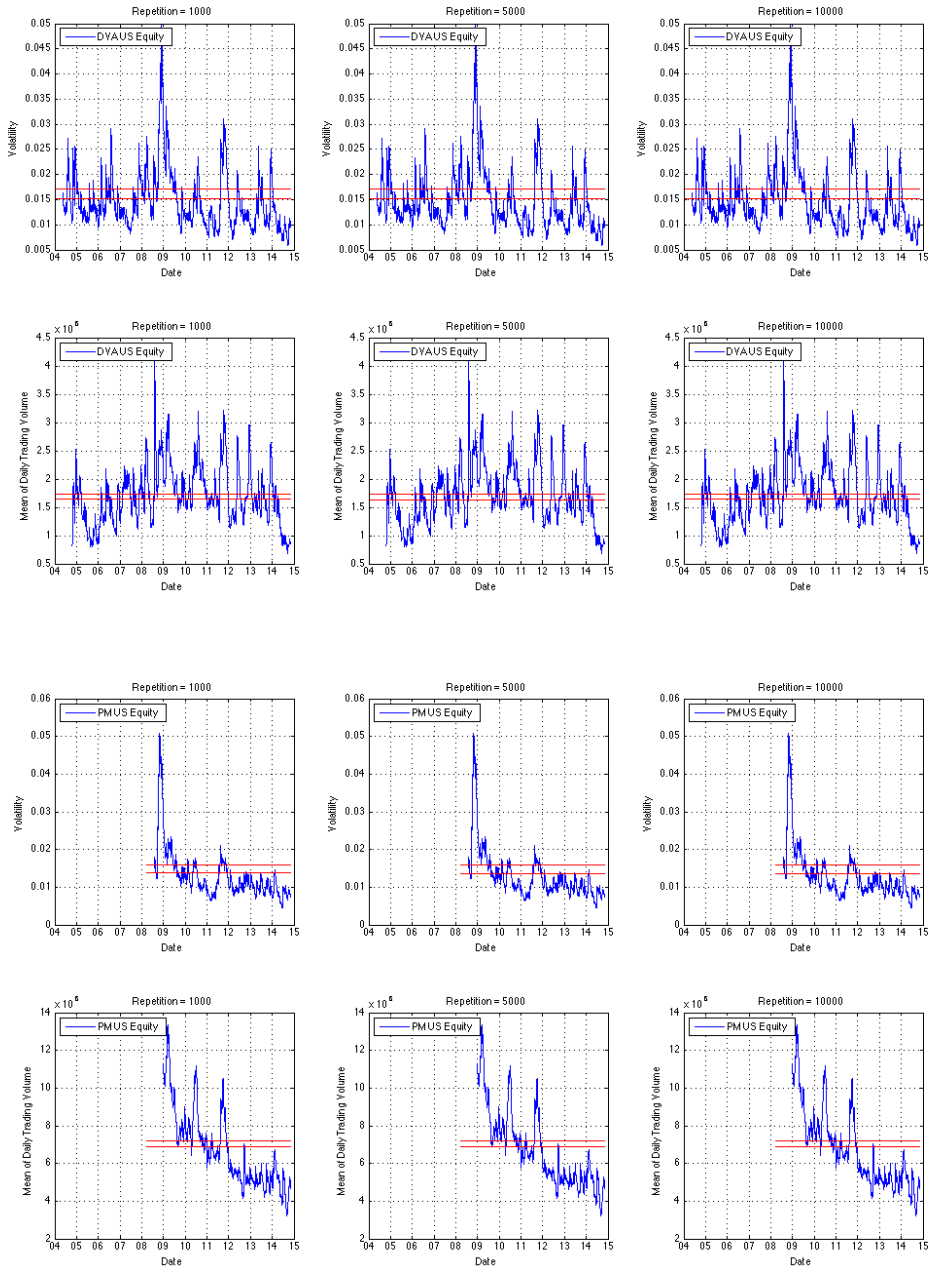
Here I use for-loop to do the bootstrap, compute its confidence interval bounds and also do the time in the same time.

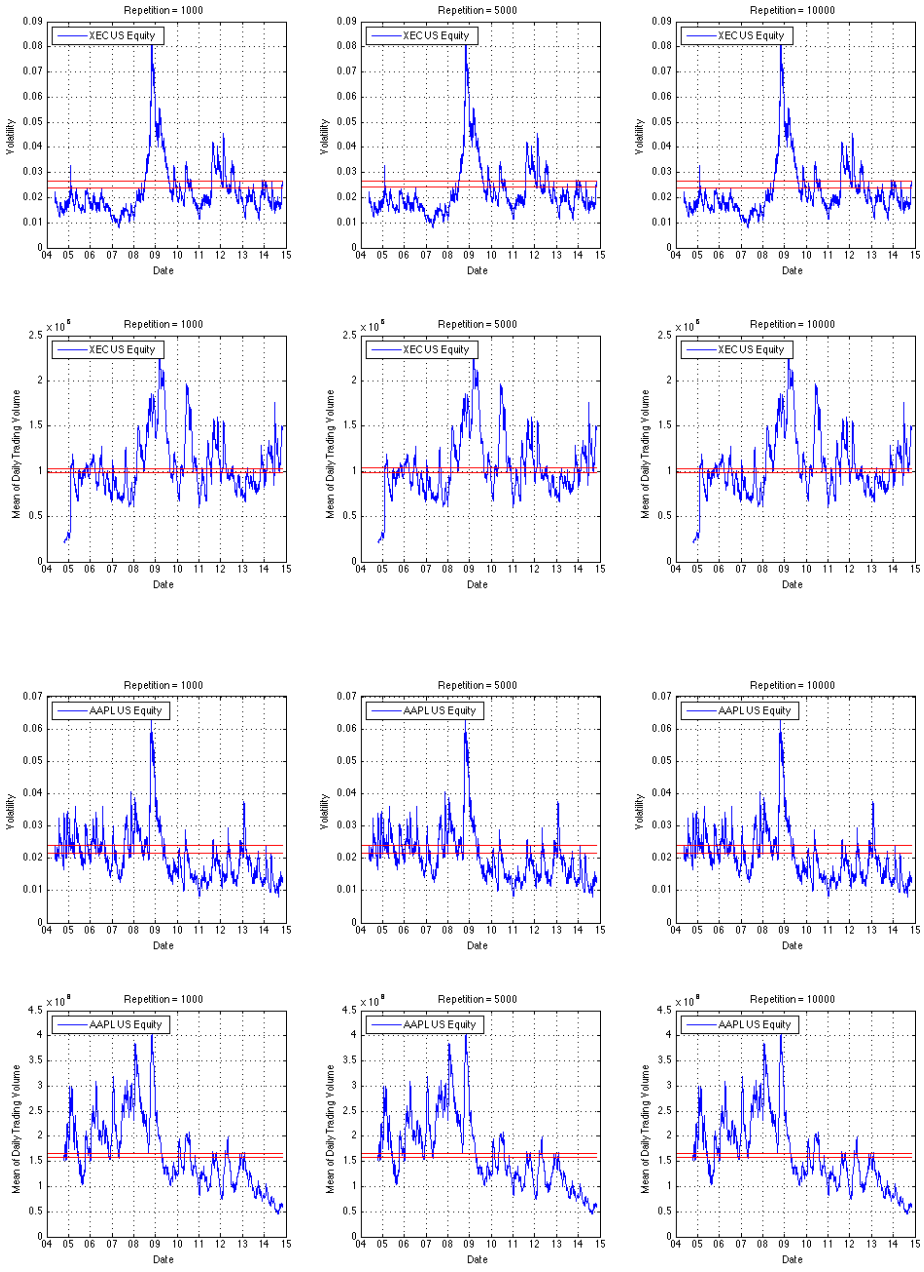
```
% rep concludes three different repetitions for the bootstrap.
rep = [1000;5000;10000];
for i = [1:3]
```

```
    tic
    temp1 = myBootstrap(log_returns,rep(i),0.95,2);
% eval() is used to evaluates the MATLAB code in the string expression.
    eval(['Volci',num2str(rep(i)),'=temp1',';']);
    temp2 = myBootstrap(volume,rep(i),0.95,1);
    eval(['Vlmci',num2str(rep(i)),'=temp2',';'])
    time(i)=toc;
end
 clearvars temp1 temp2
```

e) Plot the volatility and mean of trading volume with different confidence interval.

```
% Firstly, do the volatility plot.
figure(ii+3)
h = figure(ii+3);
set(h,'Units','Normalized','Position',[0.05,0.02,0.9,0.85]);
for i = [1:3]
    subplot(2,3,i)
    eval(['ci=Volci',num2str(rep(i)),';']);
    myplot(dates,sigma2,ci,1);
    grid on
    datetick('x','yy')
    xlabel('Date')
    ylabel('Volatility')
    title(['Repetition = ',num2str(rep(i))]);
    legend(stock,'Location','NW')
end
% Secondly, do the mean of volume plot.
for i = [1:3]
    subplot(2,3,i+3)
    eval(['ci=Vlmci',num2str(rep(i)),';']);
    myplot(dates,mu_vlm3,ci,1);
    grid on
    datetick('x','yy')
    xlabel('Date')
    ylabel('Mean of Daily Trading Volume')
    title(['Repetition = ',num2str(rep(i))]);
    legend(stock,'Location','NW')
end

clearvars h
```


Repetition = 1000
Repetition = 5000
Repetition = 10000
DYAUS Equity
PMUS Equity
Volatility
Mean of Daily Trading Volume
Date



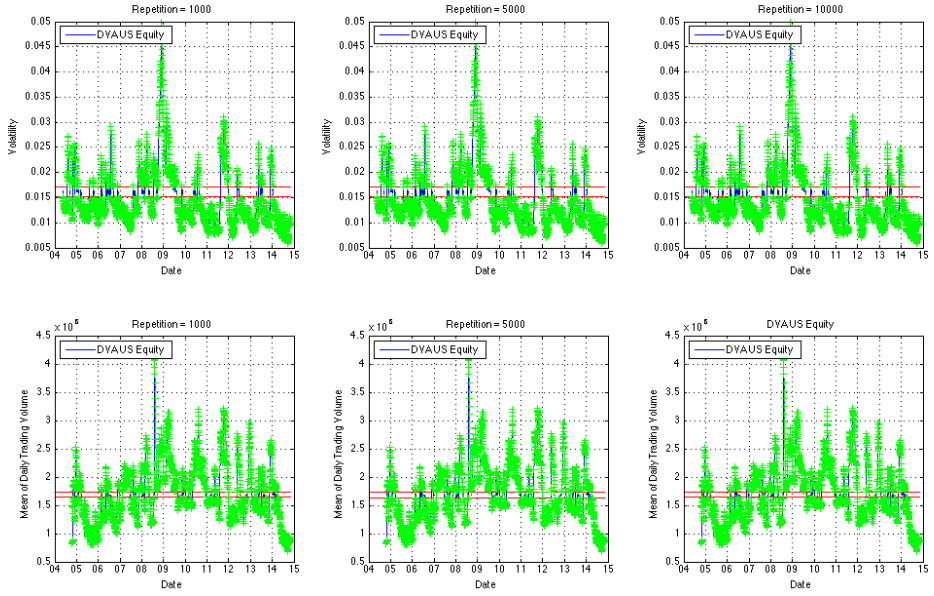
PART 3) Analysis of Volume and Volatility

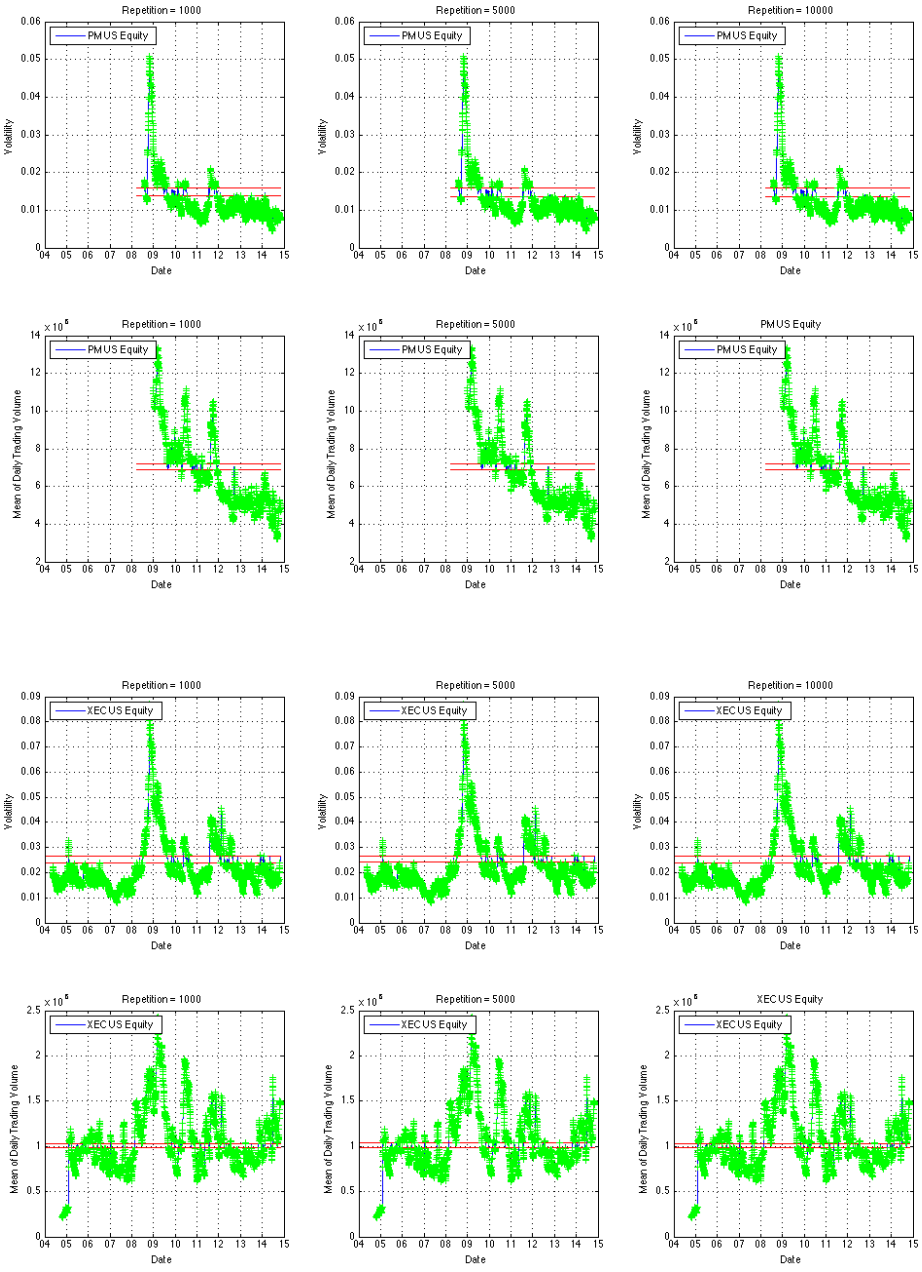
In this part, I used the functions created by myself. They are *myplot()*.

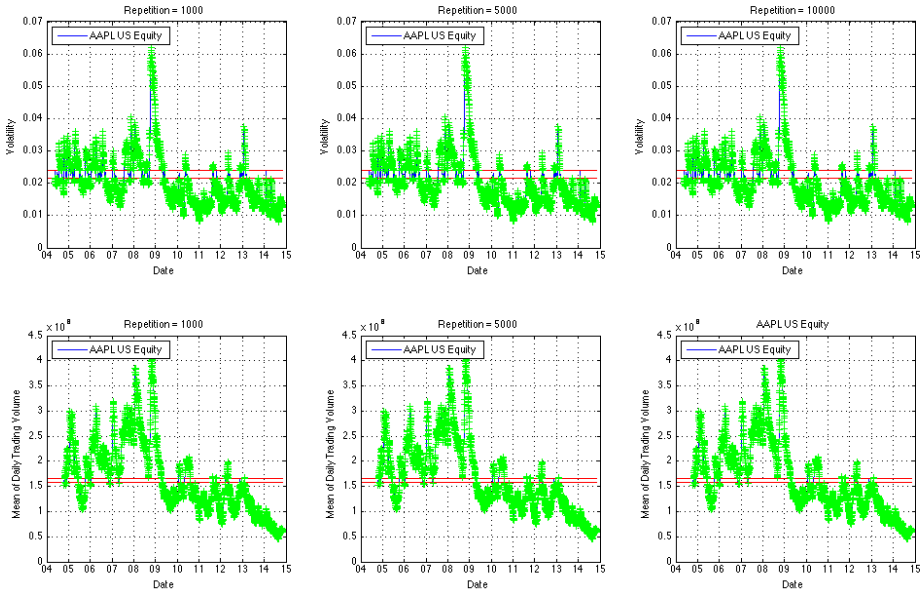
a) and b) volatility for each of 3 boot bounds.

```
figure(ii+3)
title(stock)
% Add the points outside bounds to the former figure respectively.
% This for-loop is for volatility each of 3 boot bounds.
```

```
for i = 1:3
    subplot(2,3,i,'align')
    % The following line is nearly equal to 'ci=Volci1000;'.
    eval(['ci=Volci',num2str(rep(i)),';']);
    eval(['[odateVol',num2str(i),',oVol',...
        num2str(i),']=myplot(dates,sigma2,ci,2);'])
end
% This one is for mean of volume for each of 3 boot bounds.
for i = 1:3
    subplot(2,3,i+3,'align')
    eval(['ci=Vlmci',num2str(rep(i)),';']);
    eval(['[odateVlm',num2str(i),',oVlm',...
        num2str(i),']=myplot(dates,mu_vlm3,ci,2);'])
end
```




Repetition = 1000
Repetition = 5000
Repetition = 10000
PM US Equity
XEC US Equity
Volatility
Mean of Daily Trading Volume
Date



c) Analyze the dates outside of confidence interval bounds.

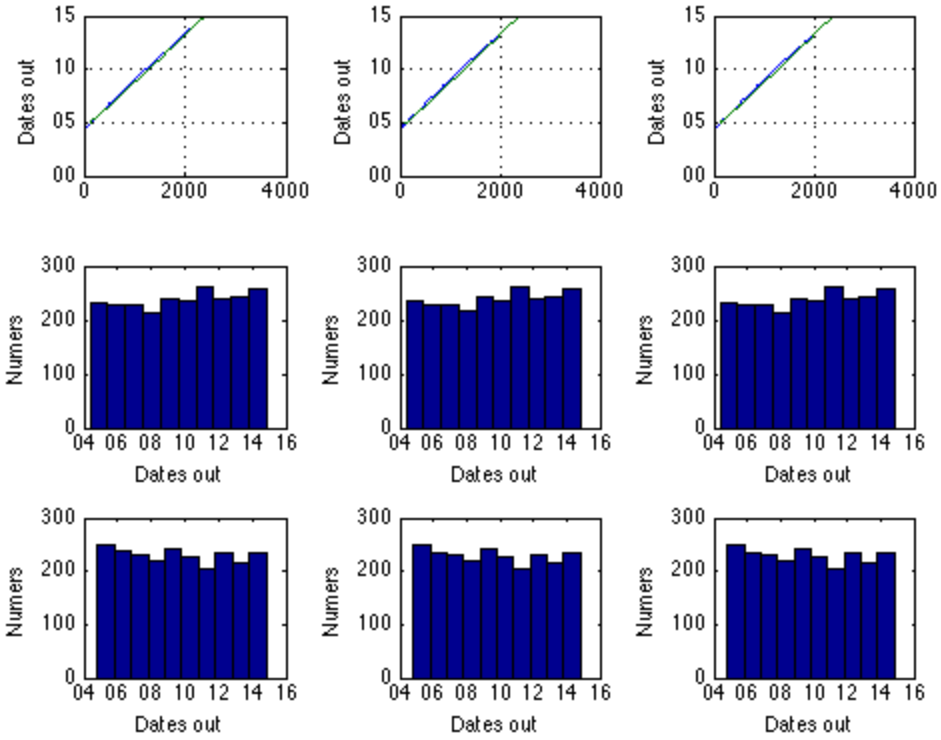
First of all, based on the graph we get in part3-a)b), we found that the datapoints were outside of the bounds on most of dates. That means either of the followings.

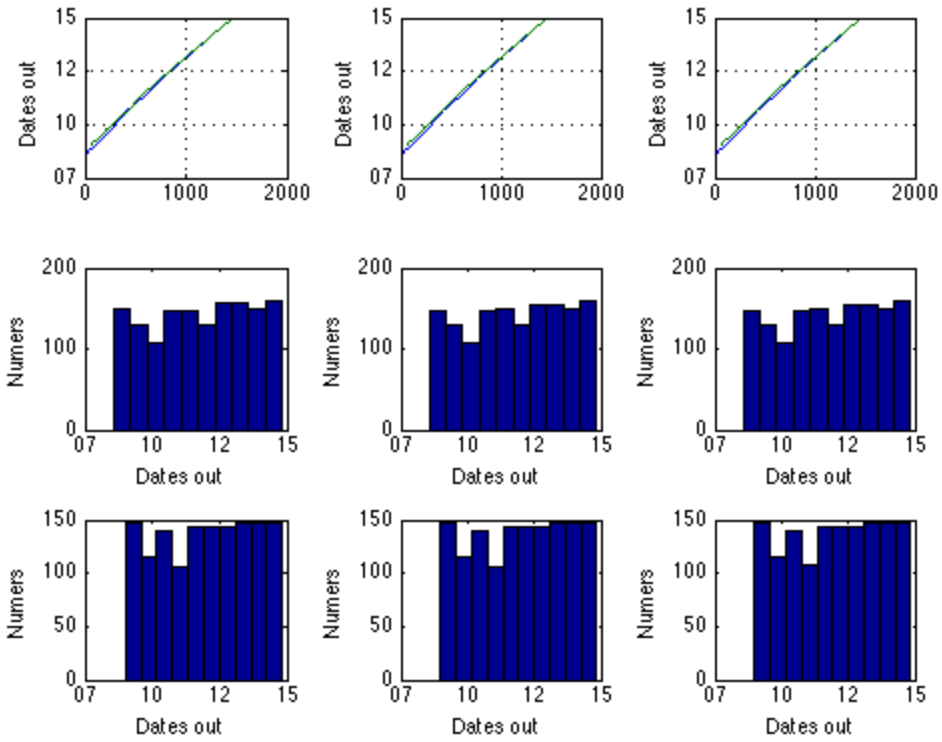
- Stock price was abnormal to some extent.
- I was wrong among the steps I computed those needed datapoints.

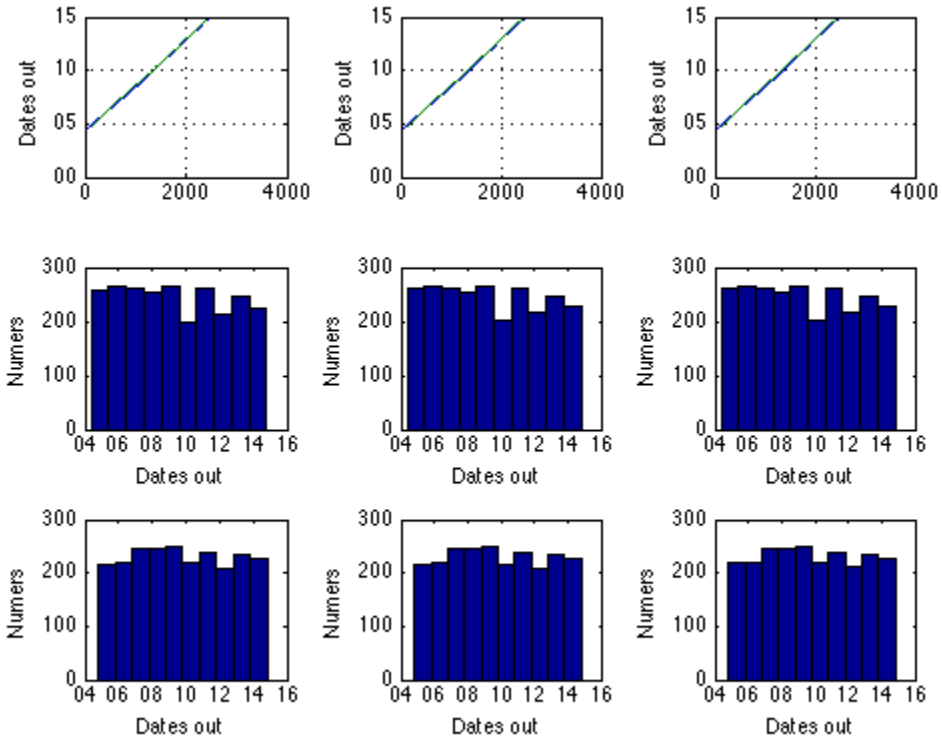
Assuming everything is right, I analyze the dates outsides of bounds for each repetition by graph simple linear plot and histogram and compare their correlation coefficients

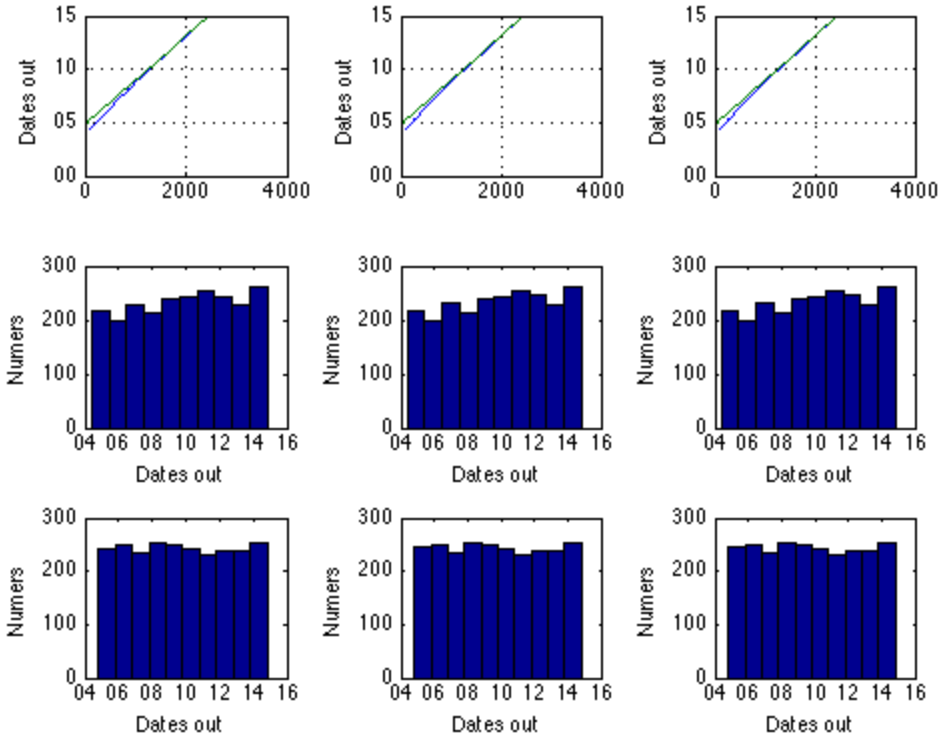
```
% Add up the size of each odate column until they have same size.
for i=1:3
    eval(['tempsize1(i) = length(odateVol',num2str(i),');'])
    eval(['tempsize1(i+3) = length(odateVlm',num2str(i),');'])
end
tempN = max(tempsize1);
addN = tempN - tempsize1;
for i=1:3
    eval(['odateVol',num2str(i),' = [NaN(addN(i),1); odateVol',...
        num2str(i),'];'])
    eval(['odateVlm',num2str(i),' = [NaN(addN(i+3),1); odateVlm',...
        num2str(i),'];'])
end

% Do the simple linear plots and the histograms.
figure(ii+4)
for i = 1:3
    subplot(3,3,i)
    eval(['plot([1:tempN],[odateVol',num2str(i),',odateVlm',...
        num2str(i),'])'])
```

```
    grid on
    datetick('y','yy')
    ylabel('Dates out')
end
clear iia

figure(ii+4)
for i = 1:3
    subplot(3,3,i+3)
    eval(['hist(odateVol',num2str(i),')'])
    xlabel('Dates out')
    ylabel('Numers')
    datetick('x','yy')
    subplot(3,3,i+6)
    eval(['hist(odateVlm',num2str(i),')'])
    xlabel('Dates out')
    ylabel('Numers')
    datetick('x','yy')
end
```




Dates out
Numers
0
1000
2000
07
10
12
15
50
100
150
200


Dates out
Numers
00
05
10
15
0
2000
4000
04 06 08 10 12 14 16
100
200
300



Compute the covariance coefficient for each two comparable data.

```
% Group them up.
odatesVolVlm = [odateVol1,odateVlm1,odateVol2,odateVlm2,...
    odateVol3,odateVlm3];
% To fix up the size.
for i = 1:6
    tempsize2(1,i) = length(find(isnan(odatesVolVlm(:,i))));
end
size = max(tempsize2);
odatesVolVlm1 = odatesVolVlm(size+1:end,:);
% To compute the correlation coefficient for each repetition between
% volatility and mean of trading volume.
rho1 = corrcoef([odatesVolVlm1(:,1),odatesVolVlm1(:,2)]);
rho2 = corrcoef([odatesVolVlm1(:,3),odatesVolVlm1(:,4)]);
rho3 = corrcoef([odatesVolVlm1(:,5),odatesVolVlm1(:,6)]);
```

After analyzing , I have the following conclusions.

1. The dates out of confidence interval of both volatility and mean of volume are almost in the same position by the linear plot.

2. By the histogram, we can also find each of them are similar to each others.

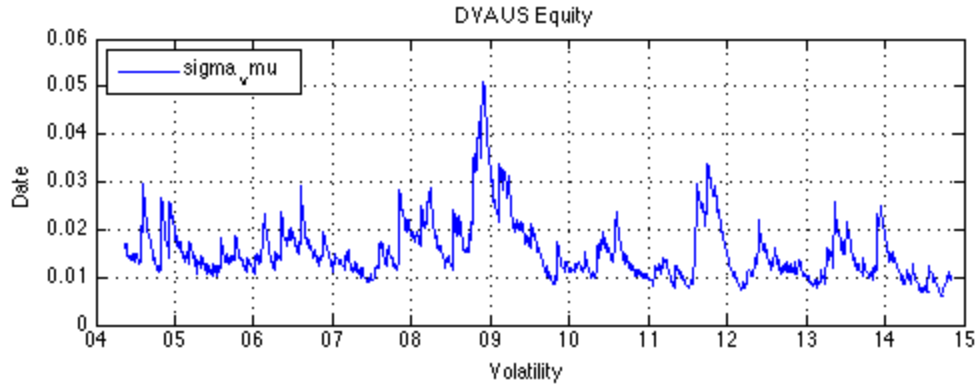
3. As times went by, two lines between dates of volatility and dates of mean of trading volume are converging.

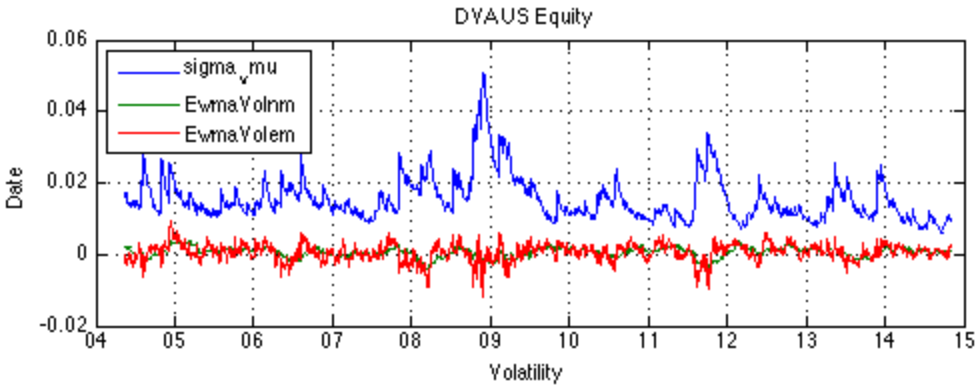
4. Since the correlation coefficient for each pair is larger than 0.99, we easily come into the conclusion that those dates out of bounds from each repetition between volatility and mean of trading volume are highly correlative.

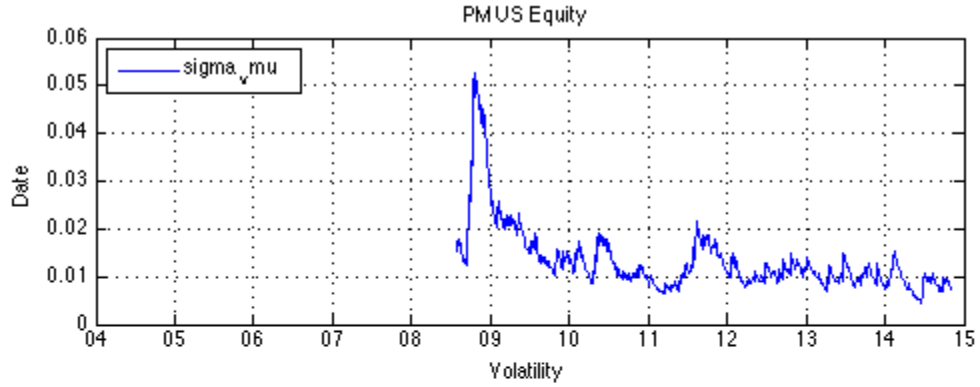
Extra credit

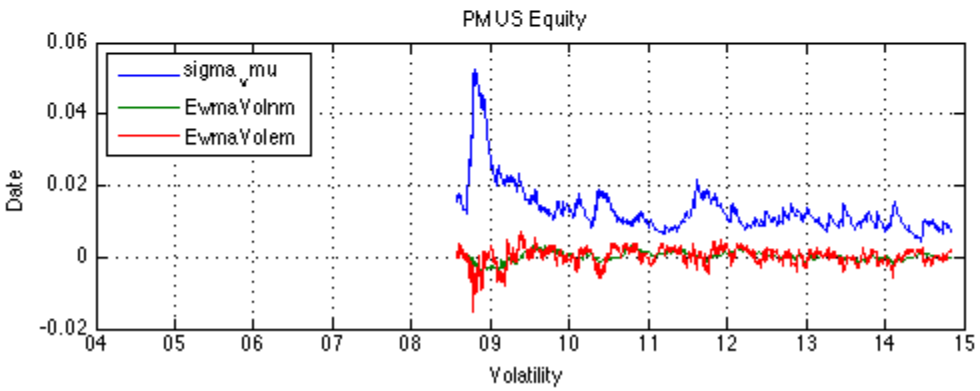
In the following, I will take the percentage change of volume as the weighted factor.

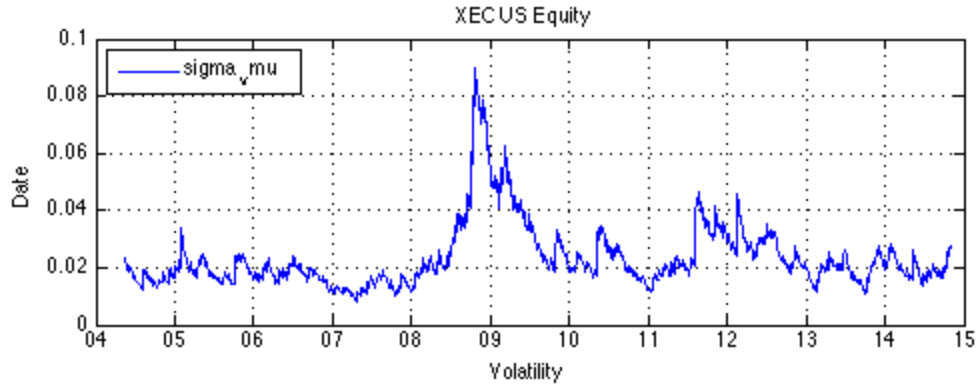
```
% pvlm is the percentage change of volume.
pvlm = vlm_change./volume;
% use the percentage change of volume as the weighted factor.
wLogRet = abs(pvlm .* log_returns);
% compute the moving average with weighted.
vmu = MAmean(wLogRet,90);
% compute the volatility
sigma_vmu = EWMAvolatility(log_returns,vmu,90,0.94);
% graph and compare
figure(ii+5)
subplot(2,1,1)
plot(dates,sigma_vmu)
grid on
datetick('x','yy')
xlabel('Volatility')
ylabel('Date')
legend('sigma_vmu','Location','NW')
title(stock)
subplot(2,1,2)
plot(dates,[sigma_vmu UWMAmu_LogRet EWMAmu_LogRet])
grid on
datetick('x','yy')
xlabel('Volatility')
ylabel('Date')
legend('sigma_vmu','EwmaVolnm','EwmaVolem','Location','NW')
title(stock)
```

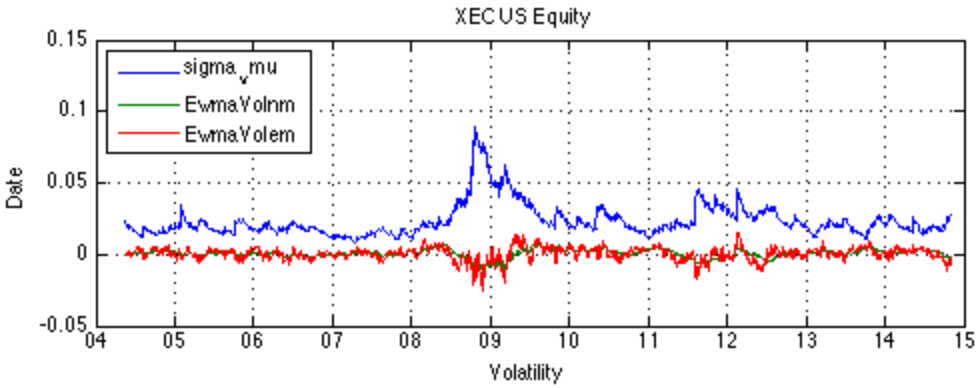

DVAUS Equity
sigma_mu
Date
Volatility

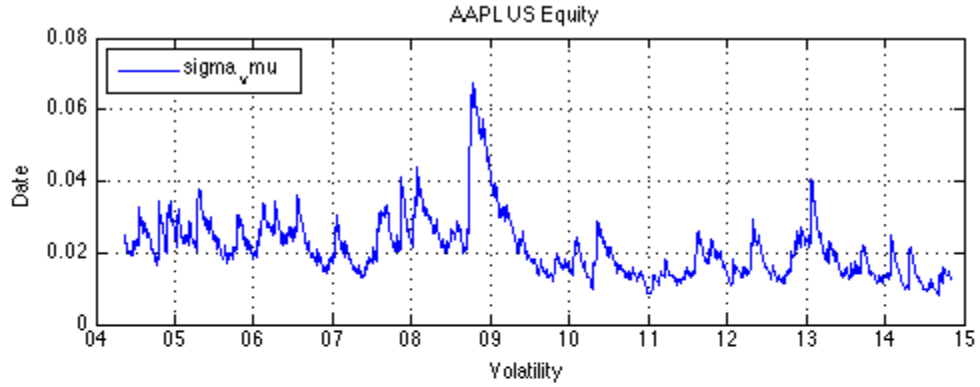

DVAUS Equity
sigma_mu
EwmaVolnm
EwmaVolem
Date
Volatility

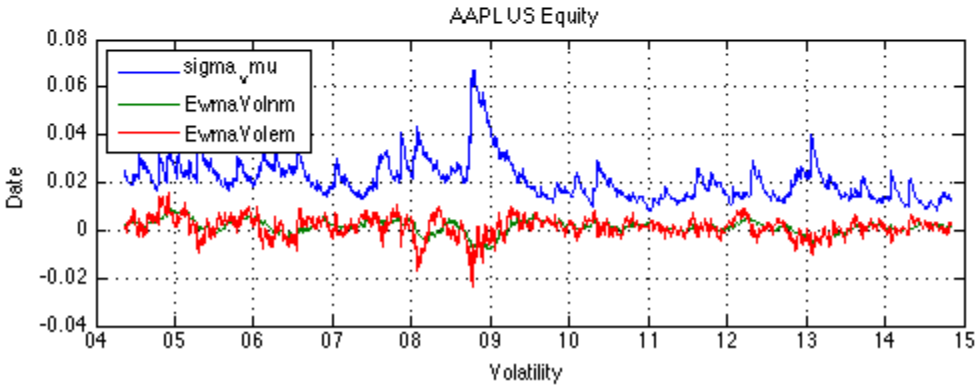

PMUS Equity
sigma_mu
Date
Volatility


PMUS Equity
sigma_mu
EwmaVolnm
EwmaVolem
Date
Volatility


XEC US Equity
sigma_v mu
Date
Volatility


XEC US Equity
sigma_v mu
EwmaVolnm
EwmaVolem
Date
Volatility


AAPL US Equity
sigma_v mu
Date
Volatility


AAPL US Equity
sigma_v mu
EwmaVolnm
EwmaVolem
Date
Volatility

Group up datas.

The following is going to group up eact price by specific datapoints.

```
    num1 = (n+2)/3;
    PriceE(:,num1) = price;
    VolumeE(:,num1) = volume;
    LogReturnE(:,num1) = log_returns;
    VolumeChangeE(:,num1) = vlm_change;
    pROCE(:,num1) = pROC;
    EwmaVolnmE(:,num1) = sigma1;
    EwmaVolemE(:,num1) = sigma2;
    MeanVlmE(:,num1) = mu_vlm3;
    MeanVlmChgE(:,num1) = mu_vlm_change4;
    TimerForEach(:,num1) = time;

    % EwmaVolpvcE are for the extra credit question.
    EwmaVolpvcE(:,num1) = sigma_vmu;

end

PriceE = [dates,PriceE];
VolumeE = [dates,VolumeE];
LogReturnE = [dates,LogReturnE];
VolumeChangeE = [dates,VolumeChangeE];
EwmaVolnmE = [dates,EwmaVolnmE];
EwmaVolemE = [dates,EwmaVolemE];
MeanVlmE = [dates,MeanVlmE];
MeanVlmChgE = [dates,MeanVlmChgE];
EwmaVolpvcE = [dates,EwmaVolpvcE];
```

To clear temporary variable and make my workspace tidy.

```
clearvars n i num1 stock sigma2 time UWMAmu_LogRet vlm_change Vlmci1000...
    volume ci log_returns EWMAmu_LogRet mu_vlm3 mu_vlm_change4 price...
    pROC sigma1 Vlmci5000 Vlmci1000 Vlmci10000 Volci1000 Volci5000...
    Volci10000 text sheetname tempN tempsize ii filename columnsum...
    addN rep iia iib size tempsize1 tempsize2 odatesVolVlm odatesVolVlm1
```

All Self-created Functions.

myimport

```
function [dates,price,volume] = myimport(stock,filename,sheetname,...
columnsum)
    % myimport(stock,filename,sheetname,columnsum) imports the stock data in
    % the sheet of excel. And the sheet has columnsum, which is required to
    % input in this function.
    % stock refers to the complete name of stock in excel.
    % filename refers to the name of excel file.
    % sheetname refers to the name of sheet in the excel file you want to use.
    % columnsum refers to the total columns in the sheet.

%import the rawdata.
```

```
[rawdata,text] = xlsread(filename,sheetname);
%process the rawdata.
for i = 1:3:columnsum;
     tf = strcmp(text(1,i), stock); % Compare text(1,i) and stock.
     if tf == true;      % loop until we find the stock.
         mydata = rawdata(:,i:i+2);
     end
 end
dates = mydata(:,1);
price = mydata(:,2);
volume = mydata(:,3);
end
```

logReturn

```
function logReturn = logReturn(price)
%logReturn(price) computes the log return of price.
%price refer to the stock volume you want to analyze.

logReturn = log(price(2:end,:)./price(1:end-1,:));
%concatenate NaN to the result to make it have same row and column as price.
row = size(price,1)-size(logReturn,1);
col = size(price,2);
logReturn = [NaN(row,col);logReturn];
end
```

volumeChange

```
function volumeChange = volumeChange(volume)
    % volumeChange(volume) computes the change in volume.
    % volume refer to the stock volume you want to analyze.
    volumeChange = volume(2:end,:)-volume(1:end-1,:);
    %concatenate NaNs to the result to make it have same row and column
    % as price.
    row = size(volume,1)-size(volumeChange,1);
    col = size(volume,2);
    volumeChange = [NaN(row,col);volumeChange];
end
```

MAmean

```
function mu = MAmean(data,n,lambda)
    % UWMAmean(data,n) computes the normal moving average mean of data
    % keeping n numbers.
    % data can be the one like original prices, original volume, log
    % returns, etc.
    row = size(data,1);
    col = size(data,2);
    % creat a matirx mu with same rows an columns as data.
    mu = NaN(row,col);
    if nargin == 2
        % fill out mu with the following means.
        for t = n+2:row
            mu(t,:) = mean(data(t-n:t-1,:));
        end
```

```
    end
    % EWMAmean(data,n) computes the exponentially weighted moving
    % average mean of data keeping n numbers.
    % data can be the one like original prices, original volume, log
    % returns, etc.
    if nargin == 3
        lambdaN = lambda.^(0:n-1)';
        lambdaN = repmat(lambdaN,1,col);
        % fill out mu with the following means.
        for t = n+1:row
            mu(t,:) = (1-lambda) .* sum(lambdaN .* data(t-1:-1:t-n,:));
        end
    end
end
```

EWMAvolatility

```
function sigma = EWMAvolatility(data,mu,n,lambda)
    % EWMAvolatility(data,mu,n,lambda) computes the exponentially
    % weighted moving average volatility of data keeping n numbers.
    % here the decay factor is lambda.
    % data can be the one like original prices, original volume, log
    % returns, etc.
    % mu is supposed to the mean used here, like normal mean and
    % weighted average mean.
    row = size(data,1);
    col = size(data,2);
    % to transform lambda in vector to be in matrix form in case that
    % we use it in bootstrap situation.
    lambdaN = lambda.^[0:n-1]';
    lambdaN = repmat(lambdaN,1,col);
    % creat a matirx mu with same rows an columns as data.
    var = NaN(row,col);
    % fill out mu with the following means.
    for t = n+2:row
        var(t,:) = (1-lambda) .* sum(lambdaN.*(data(t-1:-1:t-n,:)-mu...
        (t,:)).^2);
    end
    sigma = sqrt(var);
end
```

myBootstrap

```
function bounds = myBootstrap(data,rep,ci,flag)
    % [BootData,cibounds] = myBootstrapB(data,rep,ci) do the bootstrap
        data here.
    % it returns the bootstrap dataset for rep times for the input data
    as BootData.
    % it also returns the bounds of ci confidence interval ci as cibounds.
    % data decides what value you need to bootstrap.
    % rep is the bootstrap repetition.
    % ci is the confidence interval.

% delete the NaN elements in data.
    data(isnan(data)) = [];
```

```
% bootstraping data
    N = size(data,1);
    %INDICES = fix(1+N*rand(rep*N,1)); % Random integers between 1 and N
    INDICES = randi([1,N],rep*N,1);
    BootData = data(INDICES,:);
    BootData = reshape(BootData, N, rep);
    % computing the critical points.
    alpha = 1-ci;
    bounds = norminv([alpha/2 1-alpha/2],0,1)';
    switch flag
        case 1
        % computing the bounds with ci for mean.
            BootDataMu = mean(BootData);
            MuMu = mean(BootDataMu);
            MuStd = std(BootDataMu);
            bounds = MuMu + bounds .* MuStd;
        case 2
            BootDataStd = std(BootData);
            StdMu = mean(BootDataStd);
            StdStd = std(BootDataStd);
            bounds = StdMu+ bounds .* StdStd;
    end
end
% QUESTION:
%1) what is the difference for the following ways to create indices
    for bootstrap.
% a. INDICES = fix(1+N*rand(rep*N,1)); % Random integers between 1 and N
% b. INDICES = randi([1,N],rep*N,1);
% 2) I know there arer some other way which might computing the bounds
%   for specific confidence interval, but I don't know how to use it correctly.
% a. norfit()
%   b. norminv()
% c. paramci()
%  d. prctile()
```

myplot

```
function [OutBoundsDates,OutData] = myplot(dates,data2,ci,plottype)
    switch plottype
        case 1
            hold off
            plot(dates,data2);
            hold on
            %plot(dates,repmat(ci(1),1,size(dates,1)),'r-')
            plot(dates,ci(1),'r--')
            %plot(dates,repmat(ci(2),1,size(dates,1)),'r-')
            plot(dates,ci(2),'r--')
            datetick('x','yy')
            xlabel('Date')
        case 2
            indices = find(data2>ci(2)|data2<ci(1));
            OutBoundsDates = dates(indices);
            OutData = data2(indices);
            hold on
```

```
            plot(OutBoundsDates,OutData,'g+')
    end
end
```

Addition: myBootstrapPlot

```
%%%%%%%%%Instruction%%%%%%%%%
% [Muci,Stdci] = myBootstrapPlot(data1,rep,ci,dates,data2,type,
% PlotOutOfBoundsData)
% 1) this function do the bootstrap here
% 2) it returns the mean(Muci) and standard deviation(Stdci) confidence
% bounds of bootstrap samples.
% 3) it can also plot the rawdata with bounds of assigned confident
% interval.
% 4) it can also point out the data which are out of the bounds of
%  assigned confident interval.
%
%%%%%%%%%%Usage%%%%%%%%%%%%
%%%% Basic parameters %%%%
% 1) data1: the raw data you want to use to bootstrap. it only can be
%   a vector.
% 2) rep: repetition times for bootstraping.
% 3) ci: percentage of confidence interval.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%% Other parameters %%%%
% You can skip the following if you do not need to do the plot. If
%   you do, please read the following.
% 1) dates: the dates correspond to the raw data.
% 2) data2: the data you want to compare with the bounds of assigned
%   confidence interval.
% 3) type: the type of your raw data. i.e. 'Volatility' or 'Mean of Daily
% Trading Volume'
% 4) PlotOutOfBoundsData: just input 'y' to plot the points out of
%   bounds of assigned confidence interval.

function [Muci,Stdci] = myBootstrapPlot(data1,rep,ci,dates,data2,...
type,PlotOutOfBoundsData)
    % delete the NaN elements in data
    data1(isnan(data1)) = [];
    % bootstraping data
    N = size(data1,1);
    %INDICES = fix(1+N*rand(rep*N,1)); % Random integers between 1 and N
    INDICES = randi([1,N],rep*N,1);
    BootData = data1(INDICES,:);
    BootData = reshape(BootData, N, rep);
    % computing the critical points.
    alpha = 1-ci;
    bounds = norminv([alpha/2 1-alpha/2],0,1)';
    % computing the bounds with ci for mean.
    BootDataMu = mean(BootData);
    MuMu = mean(BootDataMu);
    MuStd = std(BootDataMu);
    Muci = MuMu + bounds .* MuStd;
    % computing the bounds with ci for standard deviation.
```

```
BootDataStd = std(BootData);
StdMu = mean(BootDataStd);
StdStd = std(BootDataStd);
Stdci = StdMu+ bounds .* StdStd;

        if nargin >= 6
            switch type
                case 'Volatility'
                    tempci = Stdci;
                    ylname = 'Volatility';
                case 'mVolume'
                    tempci = Muci;
                    ylname = 'Mean of Daily Trading Volume';
            end
        end

        if nargin == 6
            hold off
            plot(dates,data2);
            hold on
            plot(dates,tempci(1),'r-')
            plot(dates,tempci(2),'r-')
            datetick('x','yy')
            xlabel('Date')
            ylabel(ylname)
            title(['Repetition = ',num2str(rep)]);
        end
        if nargin == 7
            if PlotOutOfBoundsData == 'y'
                indices = find(data2>tempci(2)|data2<tempci(1));
                OutBoundsDates = dates(indices);
                OutData = data2(indices);
                hold on
                plot(OutBoundsDates,OutData,'g+')
            end
        end
    end
```

Published with MATLAB® R2014a