Table of Contents


```
% stocks =
 hist_stock_data(now-365,now,'SPY','T','VZ','GOOG','NFLX','FB','EBAY','SBUX','TSLA
```

My portfolio consists of 55 stocks from different sectors within S&P 500. Let's download the data from Yahoo Finance and organize it for our analysis.

```
s = struct(stocks);
Date = stocks.Date;
Ticker = {stocks.Ticker};
AdjClose = [stocks.AdjClose];
colNames =
 { 'SPY','T' 'VZ' 'GOOG' 'NFLX' 'FB' 'EBAY' 'SBUX' 'TSLA' 'NKE' 'AMZN' 'KO' 'CL' '
Table = array2table(AdjClose,'VariableNames',colNames);
Final_Table = [Date Table];
Final_Table.Var1 = datetime(Final_Table.Var1);
Final_Table.Properties.VariableNames{1} = 'Date';

head(Final_Table,5)
```

```
ans =

  5×57 table

       Date           SPY          T          VZ         GOOG        NFLX
  FB          EBAY        SBUX        TSLA        NKE         AMZN        KO
   CL          PEP          PG         WMT         XOM         EOG         KMI
```

CVX WMB MS JPM WFC BAC GS
CVS PFE JNJ UNH ANTM BA CAT
HON FDX NOC MSFT CSCO MU ADBE
NVDA SHW FMC MOS CF LYB CBRE
SPG KIM BXP AVB FE NI PPL
AEP CMS

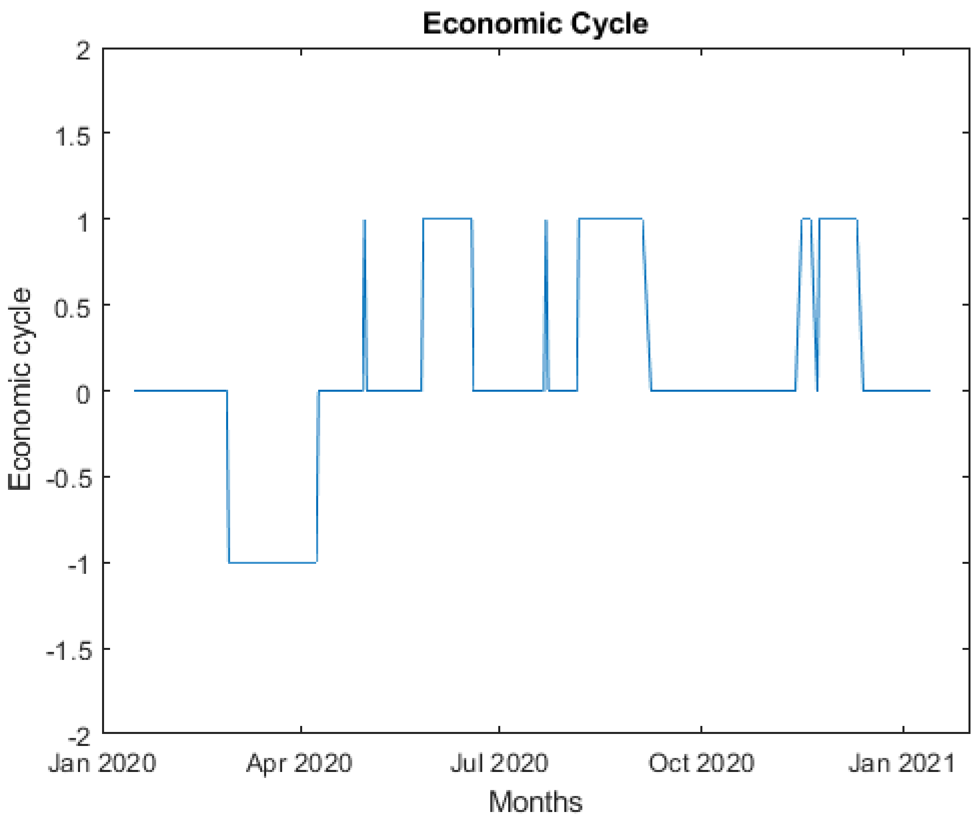
___________ _______ _______ _______ _______ _______
_______ _______ _______ _______ _______ _______ _______
_______ _______ _______ _______ _______ _______ _______
_______ _______ _______ _______ _______ _______ _______
_______ _______ _______ _______ _______ _______ _______
_______ _______ _______ _______ _______ ______ _______
_______ _______ _______ _______ _______ _______ ______
_______ _______ _______ _______ _______ _______ _______
_______ _______

16-Jan-2020 324.85 35.444 57.153 1451.7 338.62
221.77 35.411 90.709 102.7 102.39 1877.9 54.94
69.107 135.66 123.03 114 63.73 84.188 19.932
110.62 22.135 54.792 132.34 47.347 33.813 243.85
73.707 37.017 144.28 295.87 300.99 330.04 143.55
178.11 157.18 377.42 164.43 47.385 57.68 345.38
248.52 579.86 97.741 21.76 44.173 86.492 60.8
138.59 19.382 133.25 206.21 46.679 27.917 33.999
94.227 63.675
17-Jan-2020 325.86 35.77 57.594 1480.4 339.67
222.14 35.312 91.728 102.1 103.54 1864.7 55.056
69.527 137.26 123.36 113.07 63.49 82.533 19.923
109.38 21.969 55.831 133.25 47.279 33.803 243.59
73.678 36.926 145.22 293.64 301.33 322.23 144.46
179.05 157.61 373.68 165.35 47.356 57.66 349.74
248.87 589.27 98.143 21.612 44.414 86.955 61.03
137.88 19.316 133.52 207.34 46.947 28.11 34.121
95.437 63.938
21-Jan-2020 325.23 35.901 57.776 1484.4 338.11
221.44 35.214 90.66 109.44 103.59 1892 55.278
69.635 137.84 123.05 113.69 62.582 81.305 19.521
107.23 21.186 54.287 131.94 47.049 33.365 239.92
72.292 36.771 145.32 295.66 301.92 311.52 142.58
176.81 155.2 371.87 164.75 47.143 58.97 350
247.53 587.87 96.327 20.9 42.594 84.488 61.1
139.13 19.449 134.28 210.44 47.177 28.255 34.065
96.193 64.483
22-Jan-2020 325.27 36.386 57.929 1485.9 326
221.32 35.381 90.66 113.91 103.5 1887.5 55.704
69.31 139.32 123.27 114.19 62.221 80.385 19.549
106.85 20.846 54.229 131.79 46.683 33.463 241.24
72.214 36.634 144.33 295.72 299.47 307.17 139.43
175.95 153.24 369.1 163.96 47.404 59.17 350.06
249.7 587.74 95.788 19.793 41.112 81.889 60.91
136.23 19.382 133.84 207.89 47.741 28.13 34.197
96.832 64.814
23-Jan-2020 325.64 36.003 57.958 1486.7 349.6
219.76 35.105 91.856 114.44 102.05 1884.6 55.917

```
  69.389     139.58      122.7     113.91     61.832     79.166     20.296
   107.04     20.661     53.909     131.65     46.356     33.229      239.8
   71.746     37.108      144.6     294.61     300.35     315.91     139.55
   175.49     152.59     375.18     164.97     47.336       59.2     351.76
  252.44     591.32     95.032     19.714      40.91     81.511     60.84
 137.39     19.487     135.98     209.66     48.182     28.255     34.708
 97.771      65.36
```

In almost every case in investing, we need to evaluate the state of the economy to figure out which stocks will perfom better or worse during different periods. One simple way to do this is to calculate MACD. If you are interested in how we make different economic projections and modeling then check out Modeling the US economy. Let's find the state of the economy Calculate the leading and lagging moving averages and then calculate the MACD.

```
SPY = Final_Table.SPY;
movAvgShort = movavg(SPY,'exponential',12);      %lead of 3 samples
movAvgLong = movavg(SPY,'exponential',26);       % lead of 5 samples
MACD = movAvgShort - movAvgLong;

% If MACD >= 5 then momentum of the economy is up, if MACD <= 5,
 economy
% momentum is down and if MACK <5 and MACD >-5 then the economy is
 flat.
up = nnz(MACD >= 5);
up2020 = nnz( (year(Final_Table.Date) == 2020) & (MACD >= 5) );

% Let us create a column vector named econPerformance whose elements
% are 1 when the economy is up, -1 when the economy is down, and 0
 otherwise.
econPerformance = zeros(length(MACD),1);
econPerformance(MACD >= 5) = 1;
econPerformance(MACD <= -5) = -1;

%Let's plot econPerformance and Date
plot(Final_Table.Date,econPerformance)
ylim([-2 2])
title("Economic Cycle")
xlabel("Months")
ylabel("Economic cycle")

% We can see from the graph that market experienced a lot of ups and
 downs
% during coronavirus pandemic.
```




```
Final_Table.econPerformance = econPerformance;
```

Import data from text file

Script for importing data from the following text file:

```
filename: C:\Users\kushk\OneDrive\Documents\FinancialEngineering\StockIn
```

Auto-generated by MATLAB on 01-Jan-2021 08:21:28

Set up the Import Options and import the data

```
opts = delimitedTextImportOptions("NumVariables", 3);

% Specify range and delimiter
opts.DataLines = [2, Inf];
opts.Delimiter = ",";

% Specify column names and types
opts.VariableNames = ["StockTicker", "Sector", "Classification"];
opts.VariableTypes = ["string", "categorical", "string"];

% Specify file level properties
opts.ExtraColumnsRule = "ignore";
opts.EmptyLineRule = "read";
```

```
% Specify variable properties
opts = setvaropts(opts, "StockTicker", "WhitespaceRule", "preserve");
opts = setvaropts(opts,
 ["StockTicker", "Sector", "Classification"], "EmptyFieldRule", "auto");

% Import the data
StockInfo = readtable("C:\Users\kushk\OneDrive\Documents
\FinancialEngineering\StockInfo.csv", opts);
```

Clear temporary variables

```
clear opts
```

Examine the variables in StockInfo - the momentum on Feb-27-2020 is -1 i.e., 'down' Calculate the economy's performance for a specific date

```
dateOfInterest = datetime([2020 02 28]);
performanceAllDates = Final_Table.econPerformance;
performance = performanceAllDates(Final_Table.Date == dateOfInterest);

% If performance is equal to -1, extract the stock tickers of all the
 'downcycle' tickers
% If performance is 1 and if it is, extract the stock tickers of all
 the 'upcycle' tickers.
if performance == 1
   upcycleIdx = strcmp(stockInfo.Classification,'upcycle');
   stocksToBuy = stockInfo{upcycleIdx,'StockTicker'};
elseif performance == -1
   downcycleIdx = strcmp(StockInfo.Classification,'downcycle');
   stocksToBuy = StockInfo{downcycleIdx,'StockTicker'};
end

% Calculate the VaR by using function calculateVaR of each of the
 stocks
for i=1:length(stocksToBuy);
   stockTicker = stocksToBuy{i};
   var5(i) = calculateVaR(Final_Table{:,stockTicker});
end

% Find the stock with minimum VaR
[minValue,idx] = min(-var5);
minVaRStock = stocksToBuy{idx};
```

```
        Our analysis suggests us to buy WMT. Now, let us visualize the change
        in correlation over time. between SPY & WMT.
```

```
SP = Final_Table.SPY;
WM = Final_Table.WMT;

% Concatenate the vectors CM and SP horizontally.
% Use the function tick2ret with the matrix index as input
% to convert the index values to returns.
```

```
index = [WM SP];
indexRetns = tick2ret(index);


% Compute the correlation of the 2 columns of the matrix indexRetns
% using the function corr. The corr function outputs a matrix.
% Extract the correlation coefficient from the output matrix,
cm = corr(indexRetns);
c = cm(1,2);

% Write a 'for loop' to compute the correlation of 15 elements at a
 time.
% Store the correlation coefficients in a vector called rollingCorr.

windowSize = 15;
numRecords = size(indexRetns,1);
for k = 1:(numRecords - windowSize + 1);
   rollingCorrMatrix = corr(indexRetns(k:k+windowSize-1,:));
   rollingCorr(k) = rollingCorrMatrix(1,2);
end
```

Let us plot VaR at 5% for CMS

Calculate Returns

Using GBM formula, we can predict multiple paths of the future stock prices of WMT. The formula for GBM is

$$S(T + deltaT) = S(T) * exp((mu - sigma^2/2) * deltaT + sigma * epsilon * sqrt(deltaT))$$

;

```
data = Final_Table(:,16);
data = data{:,:};       %Converting table to array
logPrices = log(data);

% Calculate CMS returns
returns = diff(logPrices);
```

Calculating factors of GBM

Calculate the descriptive statistics

```
% Compute mu
mu = mean(returns);

% compute sigma
sigma = std(returns);

% Assign deltaT = 1
deltaT = 1;
```

Calculate the future prices

Now create a vector of normally distributed numbers for epsilon. Simulating 100 future paths of stock price for WMT.

```
epsilon = randn(22,100);
```

Create factors for GBM

```
factors = exp((mu-sigma^2/2)*deltaT + sigma*epsilon*sqrt(deltaT));
```

To predict the future values of the stock price, first we need to extract the price of WMT on the last trading day and store the results.

```
S0 = data(end,1);
```

Create a row vector of size 1 by 100 such that each element of the vector has the value equal to the last recorded stock price S0. Note:If you change epsilon to 200, you need to change this vector too.

```
lastPriceVector = ones(1,100)*S0;
```

Now concatenate lastPriceVector and factors

```
factors2 = [lastPriceVector;factors];
```

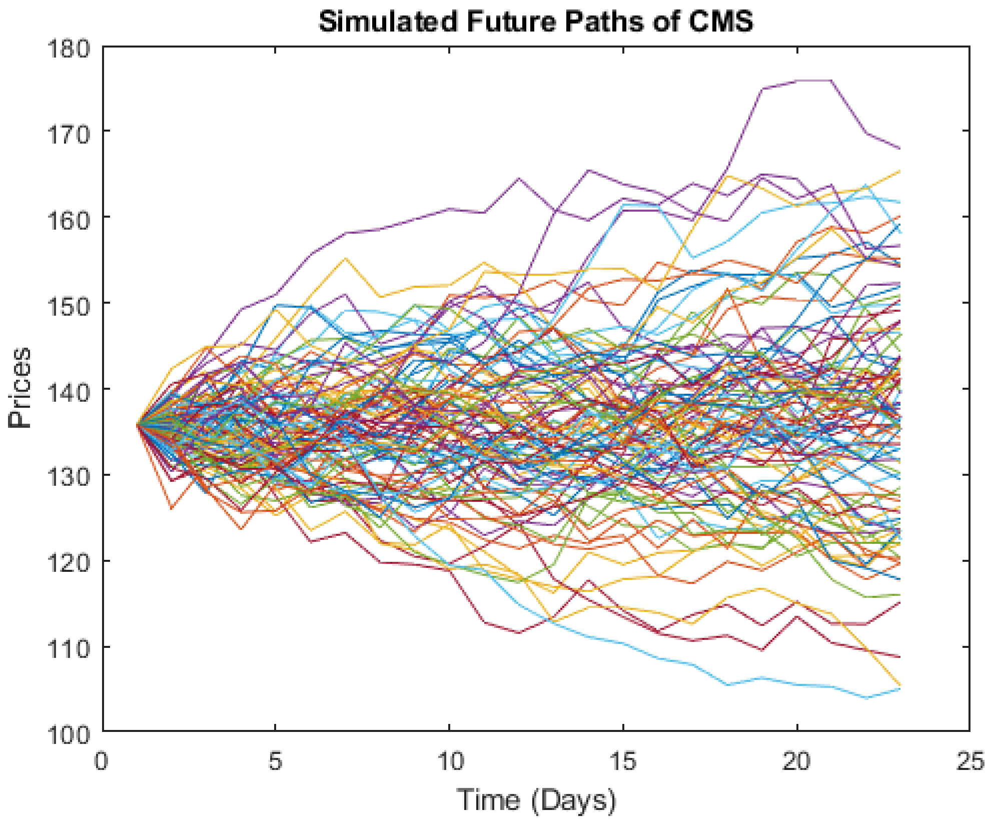
Functions like cumprod are applied to each column of the input matrix Use the function cumprod with factors 2 as the input to compute the prices at future time instants

```
paths = cumprod(factors2);
```

Plot the paths

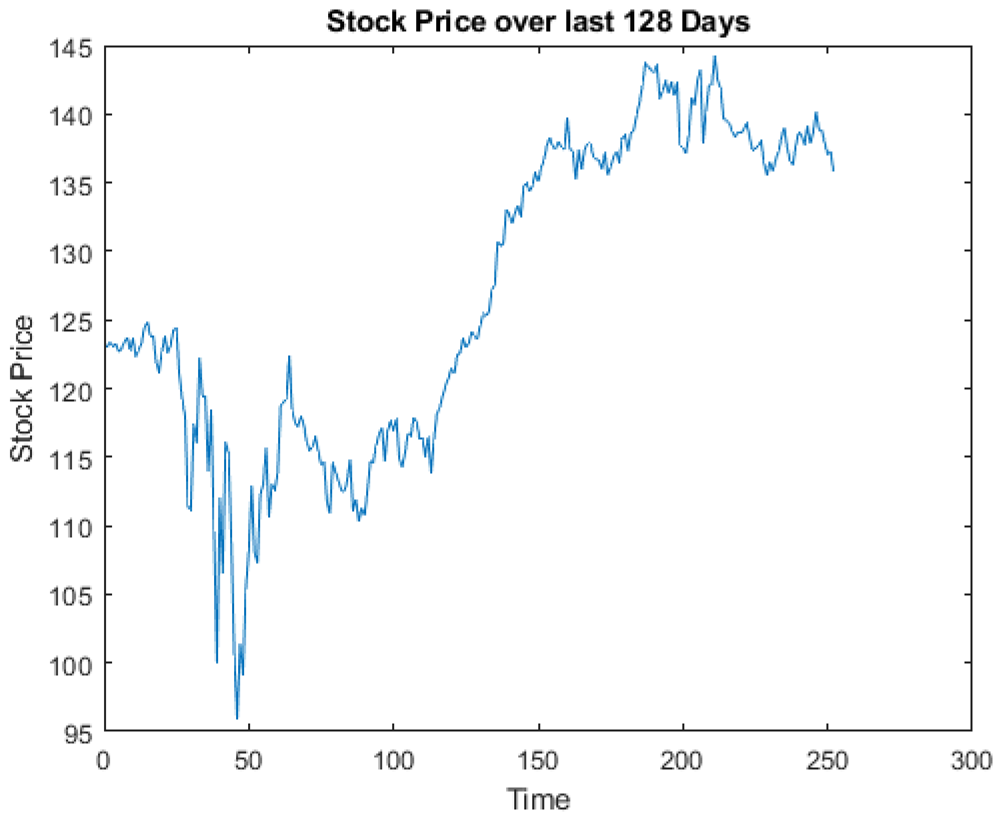
Plot paths to see the predicted prices of WMT

```
plot(paths)
xlabel("Time (Days)")
ylabel("Prices")
title("Simulated Future Paths of CMS")
```



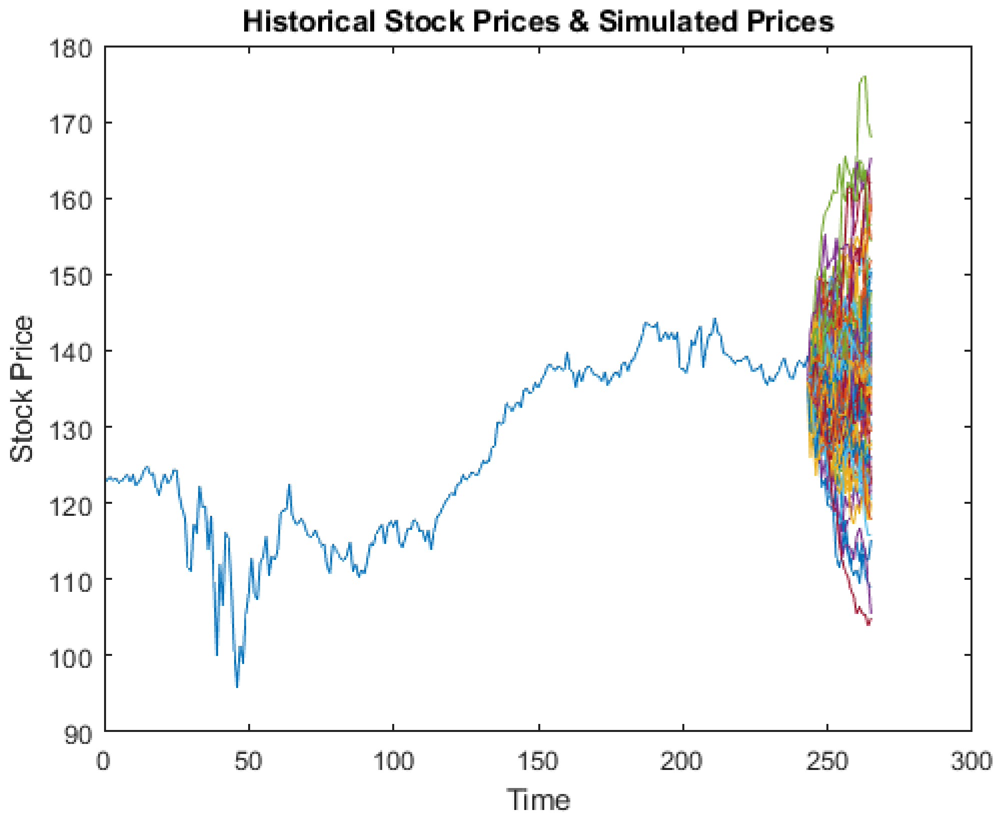
Combine stock price and predicted path together

```
plot(data)
xlabel("Time")
ylabel("Stock Price")
title("Stock Price over last 128 Days")
hold on
```



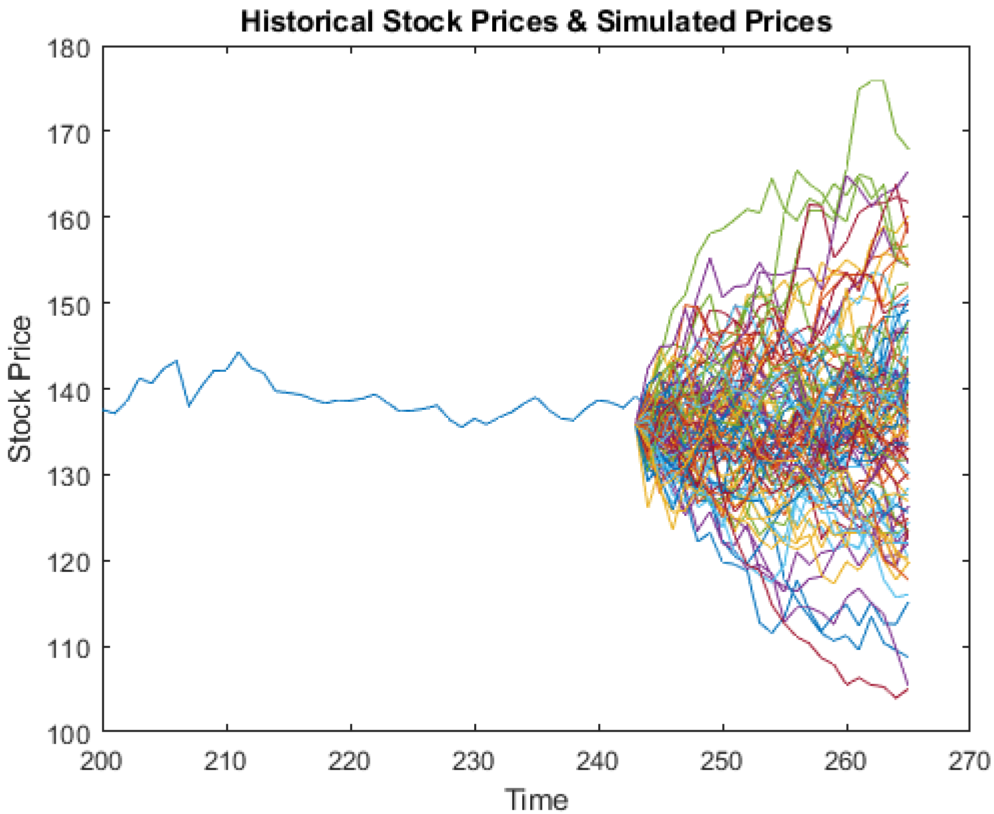
The future prices should have x-data that represents the future time points. Since the historical data contains 252 records, the future value should be plotted against the index(x-data)

```
plot(243:265,paths)
xlabel("Time")
ylabel("Stock Price")
title("Historical Stock Prices & Simulated Prices")
hold off
```



Zoom in the graph by changing the axis limit

```
xlim([200 270])
```



Extract all rows and columns from paths

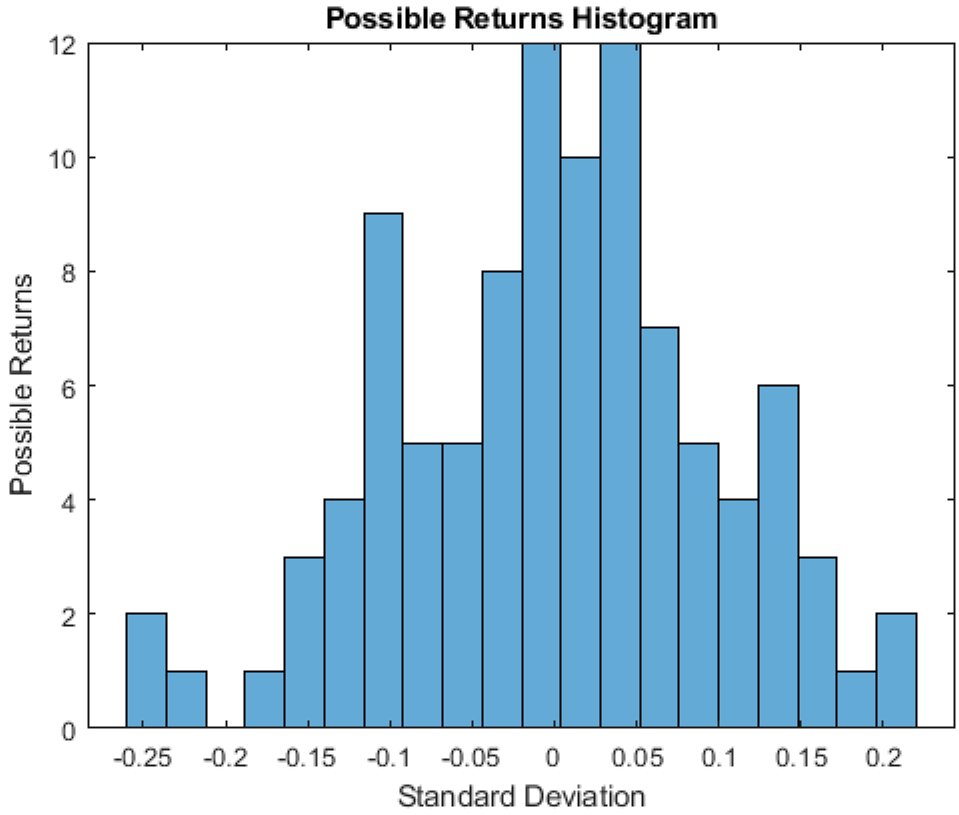
```
finalPrices = paths(end,:);
```

Calculate possible returns

```
possibleReturns = log(finalPrices) - log(S0);
```

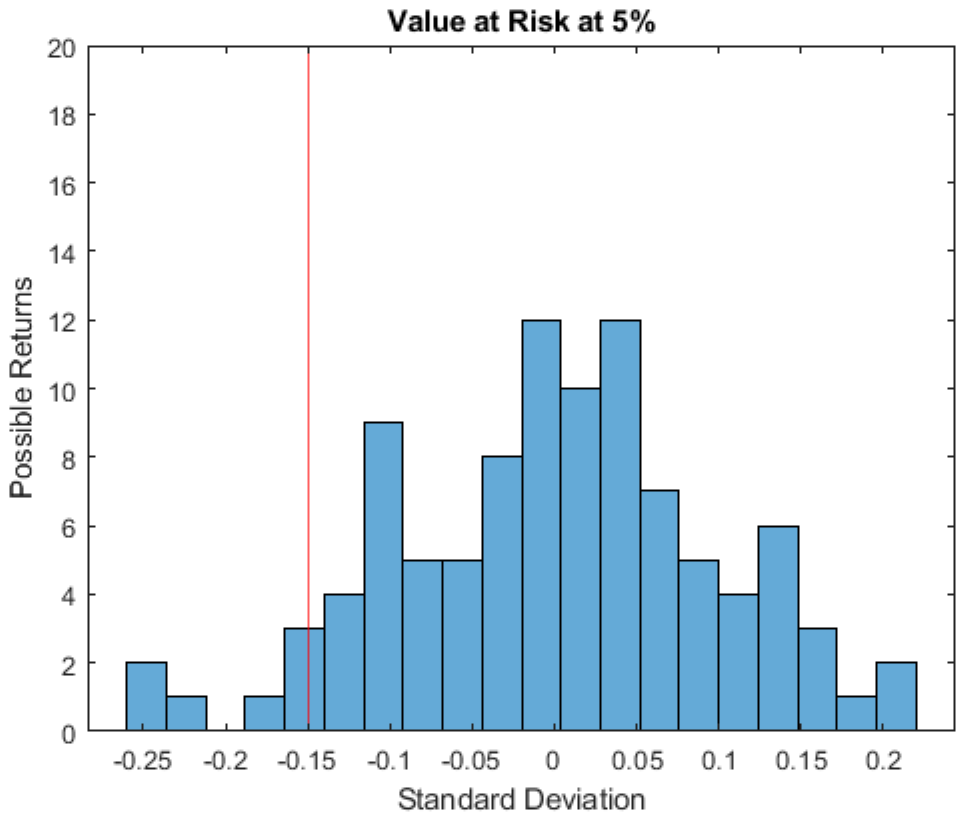
Plot a histogram of the possible returns with 20 bins.

```
histogram(possibleReturns,20)
xlabel("Standard Deviation")
ylabel("Possible Returns")
title("Possible Returns Histogram")
```



Calculate the VaR

```
var5 = prctile(possibleReturns,5);
hold on
plot([var5 var5],[0 20], 'r')
title("Value at Risk at 5%")
hold off
```



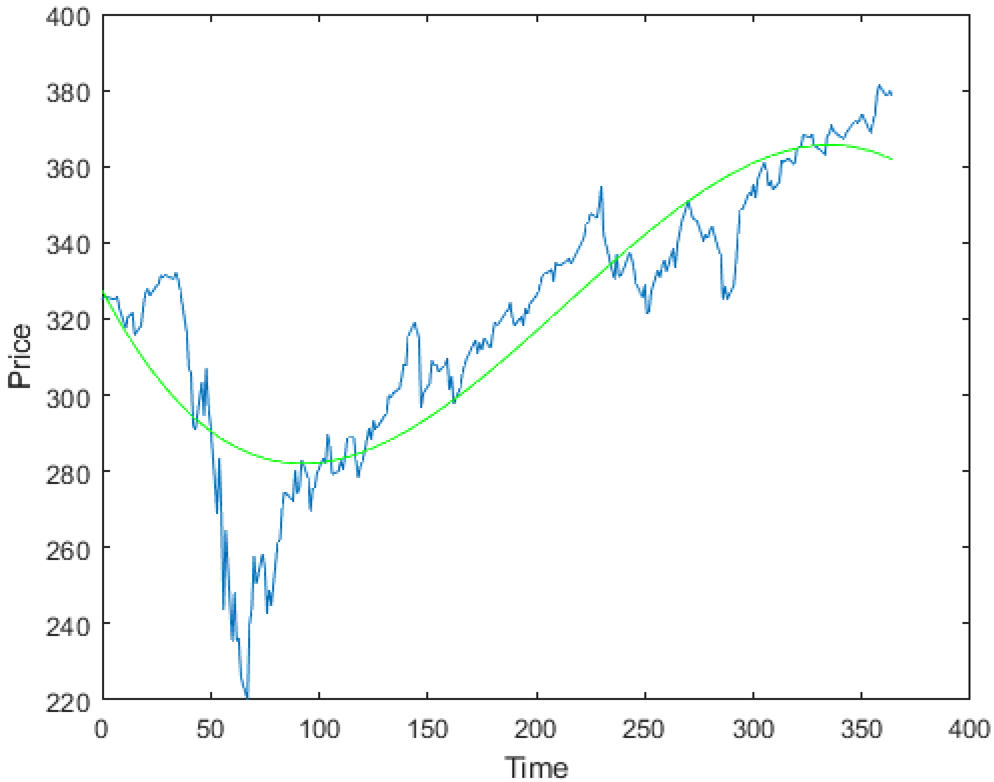
% Let's plot a regression line against SPY

```
Final_Table.days = days(datetime(Final_Table.Date)-
datetime(Final_Table.Date(1)));
SPYfit = fitlm(Final_Table,'PredictorVars','days','ResponseVar','SPY')
d = Final_Table.days;
SPYfit1 = fitlm([d d.^2 d.^3],SPY)
properties(SPYfit1)
c = SPYfit1.Coefficients
R_squared = SPYfit1.Rsquared
formula = SPYfit1.Formula
```

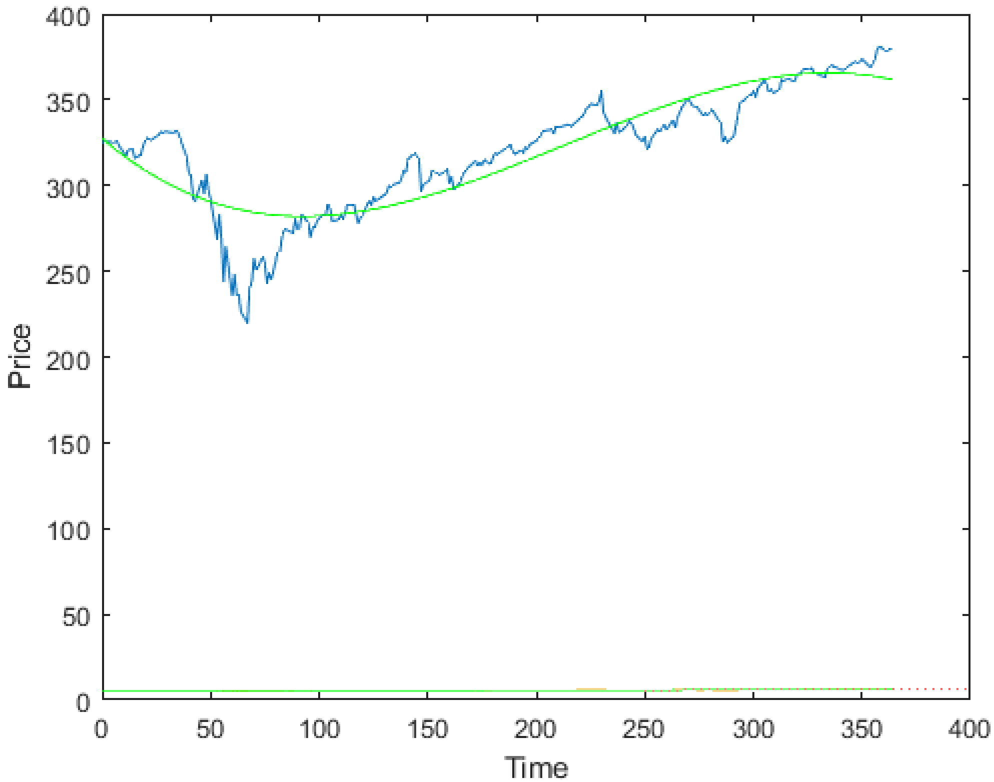
```
SPYfit =


Linear regression model:
    SPY ~ 1 + days

Estimated Coefficients:
                   Estimate       SE        tStat       pValue
                   ________    ________    ______    ___________

    (Intercept)      274.5       2.8306    96.977    2.2894e-200
    days           0.25008     0.013464    18.574     5.6056e-49
```

```
Number of observations: 252, Error degrees of freedom: 250
Root Mean Squared Error: 22.4
R-squared: 0.58,  Adjusted R-Squared: 0.578
F-statistic vs. constant model: 345, p-value = 5.61e-49

SPYfit1 =


Linear regression model:
    y ~ 1 + x1 + x2 + x3

Estimated Coefficients:
                    Estimate          SE           tStat        pValue
                   ___________    __________    ________    ___________

    (Intercept)         327.58        4.1901       78.18    9.8985e-177
    x1                 -1.0857       0.09897      -10.97      4.381e-23
    x2               0.0074973    0.00062842       11.93     3.1196e-26
    x3             -1.1693e-05    1.1311e-06     -10.338     4.5958e-21


Number of observations: 252, Error degrees of freedom: 248
Root Mean Squared Error: 16.4
R-squared: 0.775,  Adjusted R-Squared: 0.772
F-statistic vs. constant model: 284, p-value = 5.82e-80

Properties for class LinearModel:

    Residuals
    Fitted
    Diagnostics
    MSE
    Robust
    RMSE
    Formula
    LogLikelihood
    DFE
    SSE
    SST
    SSR
    CoefficientCovariance
    CoefficientNames
    NumCoefficients
    NumEstimatedCoefficients
    Coefficients
    Rsquared
    ModelCriterion
    VariableInfo
    NumVariables
    VariableNames
    NumPredictors
    PredictorNames
    ResponseName
    NumObservations
```

```
    Steps
    ObservationInfo
    Variables
    ObservationNames


c = 

  4×4 table

                       Estimate         SE         tStat       pValue   
                     ___________    __________    _______    ___________

    (Intercept)           327.58        4.1901      78.18    9.8985e-177
    x1                   -1.0857       0.09897     -10.97      4.381e-23
    x2                 0.0074973    0.00062842      11.93     3.1196e-26
    x3               -1.1693e-05    1.1311e-06    -10.338     4.5958e-21


R_squared = 

  struct with fields:

    Ordinary: 0.7748
    Adjusted: 0.7721


formula = 

y ~ 1 + x1 + x2 + x3

%Plotting
plot(d,SPY)
hold on
plot(d,SPYfit1.Fitted,'g')
xlabel("Time")
ylabel("Price")
```

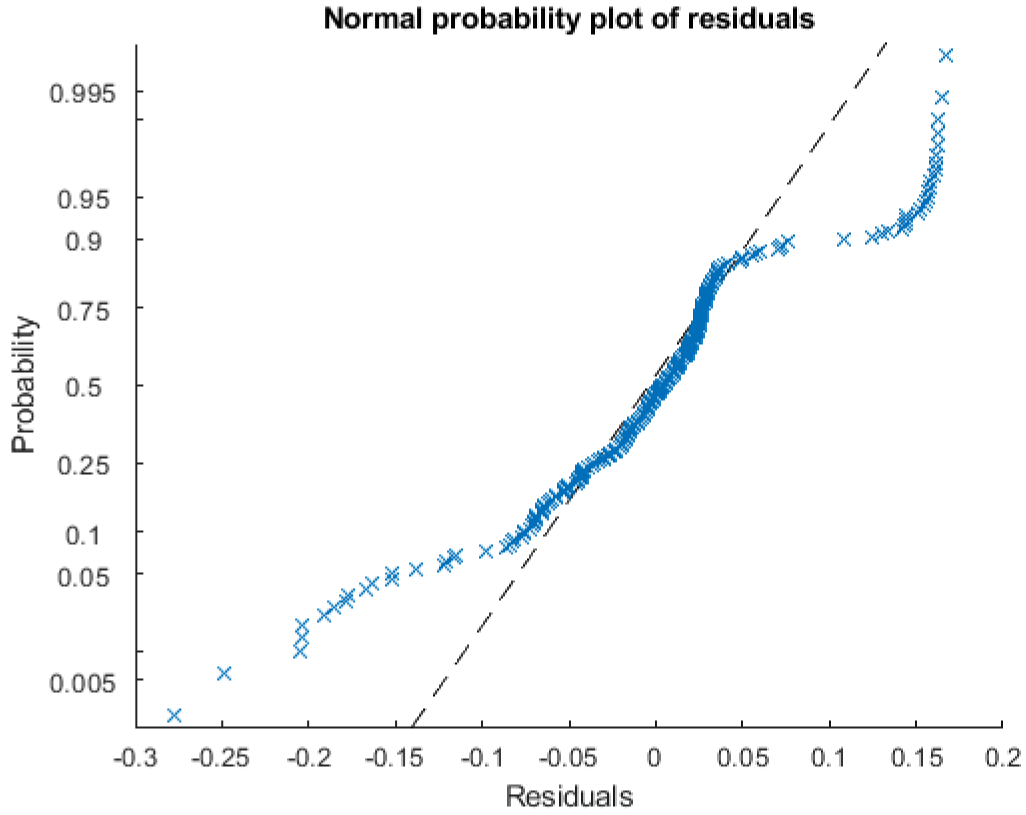



```
%Lets find the predicted SPY index values for the first 200 days using
%SPYfit.
SPY = log(Final_Table.SPY);
day = Final_Table.days;
dates = Final_Table.Date;
SPYfit3 = fitlm(day,SPY);
plot(day,SPY)
hold on
plot(d,SPYfit3.Fitted,'g')
y = predict(SPYfit3,(250:400)');
plot(250:400,y,'r:')
```



REsiduals

```
plotResiduals(SPYfit3)
plotResiduals(SPYfit3,'fitted')
plotResiduals(SPYfit3,'probability')
```



Make residual table

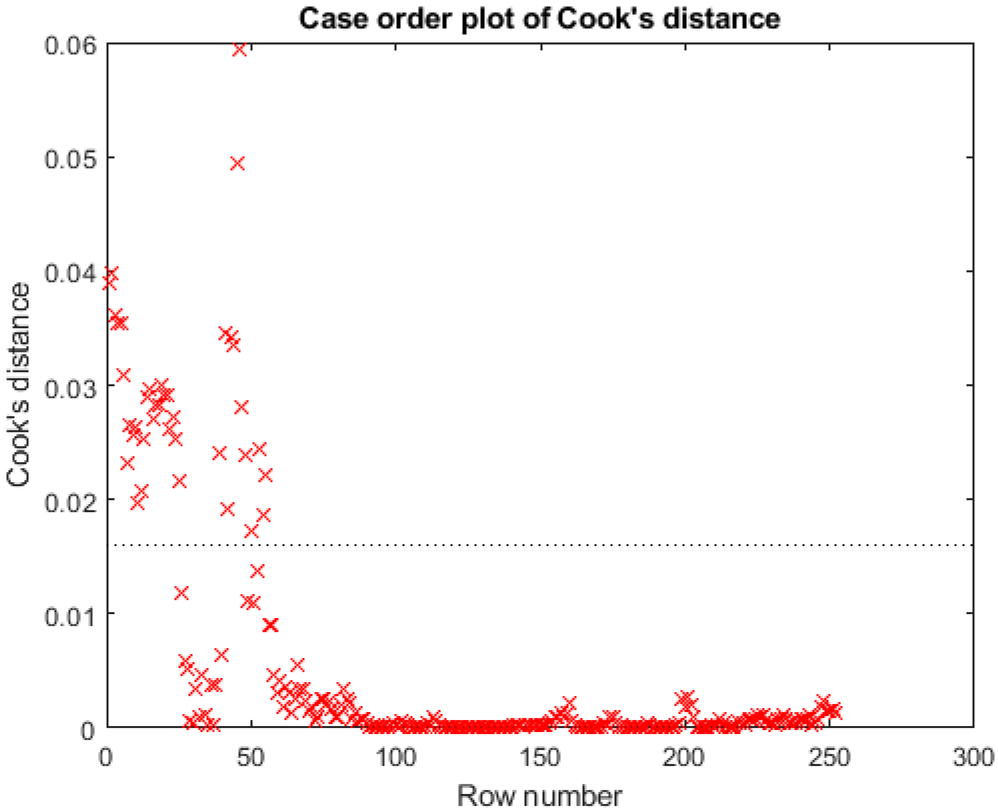
```
resTable = SPYfit3.Residuals;
% Create a numeric array containing the raw data from residual table
res = resTable.Raw;
% Let's apply Lilliefors normality test on raw residual data.
[hLil,pLil] = lillietest(res)

% Since hLil value is 1, test rejects null hypothesis that the data
 comes
% from the normal distribution. Smaller p-value also supports the
 rejection
% of the null hypothesis

% Le'ts try Jarque-Bera normality test on the raw residuals.
[hJB,pLil] = jbtest(res)

% We are confident from these tests that residuals do not have normal
% distribution.
```

```
Warning: P is less than the smallest tabulated value, returning
 0.001.

hLil =

     1
```

```
pLil =

   1.0000e-03

Warning: P is less than the smallest tabulated value, returning
 0.001.

hJB =

     1


pLil =

   1.0000e-03
```

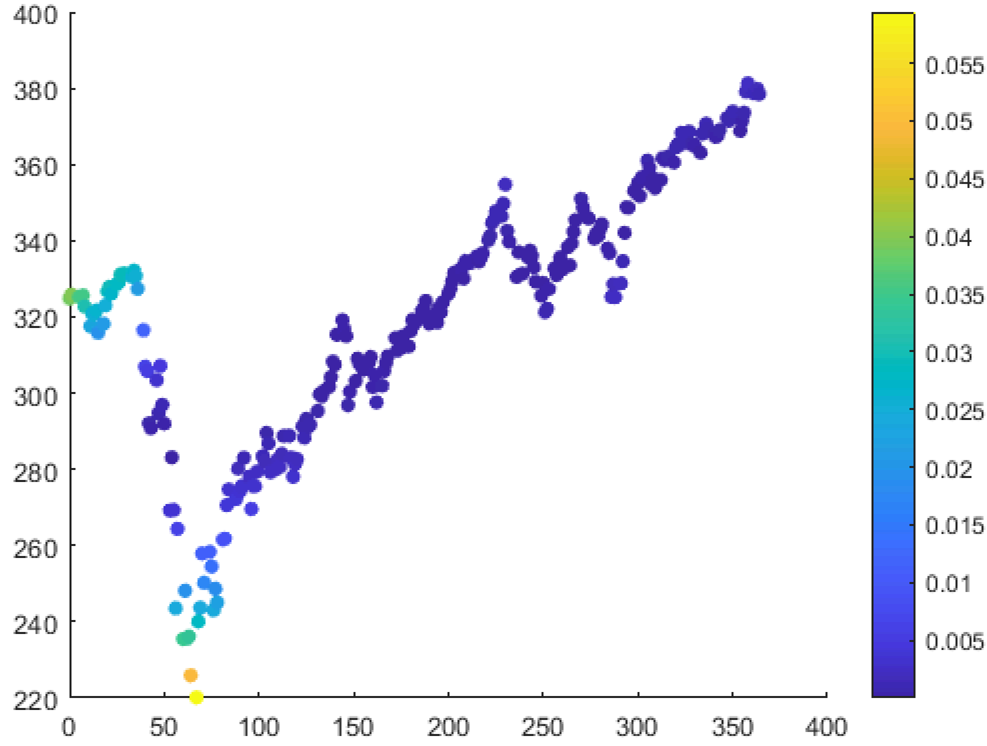
Let's evaluate the goodness of fit Diagnostics plots helps us identify outliers and different problems with out fit.

```
plotDiagnostics(SPYfit3)
plotDiagnostics(SPYfit3,'covratio')

%Cook's distance is a common way to determine outliers.
plotDiagnostics(SPYfit3,'cookd')

% Let's create diagnostic table
diagTable = SPYfit3.Diagnostics;

% Lets create numeric array containing CooksDistance
ckd = diagTable.CooksDistance;
```



Let's visualize the raw data with variables color coded according to Cook's distance.

```
scatter(Final_Table.days,Final_Table.SPY,30,ckd,'filled');
colorbar
```



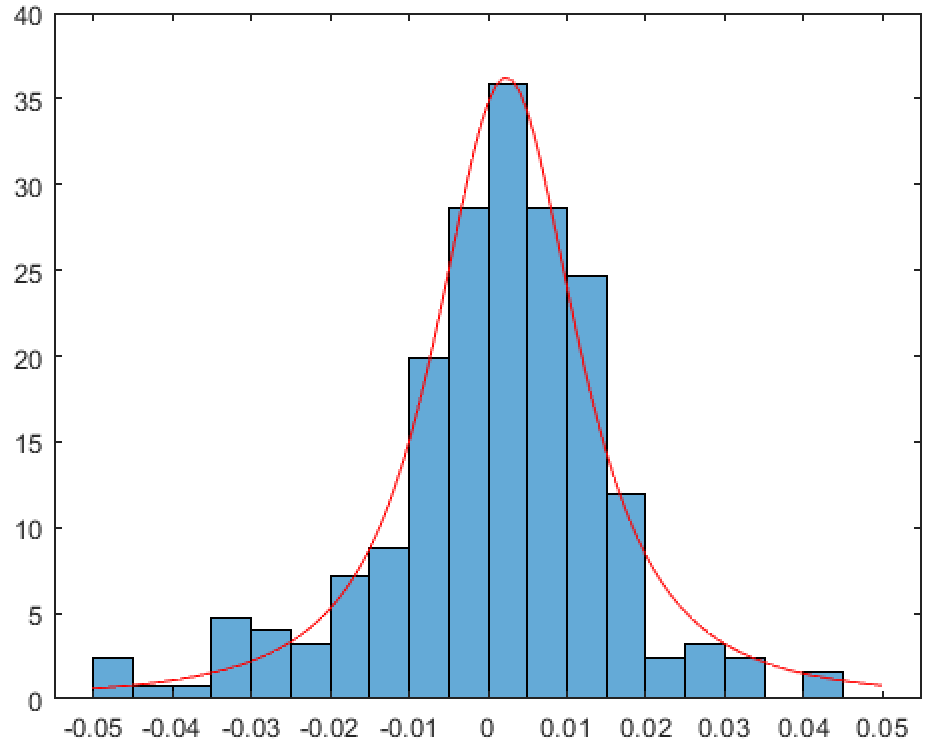
Parametric Fitting

Fit Distribution

```
SPYreturns = diff(SPY);
ndo = fitdist(SPYreturns,'Normal');
tFit = fitdist(SPYreturns,'tLocationScale')

% Hold the parameter values property of t Location-Scale probability
% distribution object
msn = tFit.ParameterValues;

% Determine the inverse cdf value of the fit.
paramVaR99 = icdf(tFit,0.01)
```

```
tFit =

  tLocationScaleDistribution

  t Location-Scale distribution
       mu = 0.00230176   [0.000735917, 0.00386761]
    sigma = 0.00973263   [0.00816485, 0.0116014]
       nu =    1.96424   [1.43557, 2.68759]
```

```
paramVaR99 =

   -0.0672


%Plotting Distributions
binEdges = -0.05:0.005:0.05;
histogram(SPYreturns,binEdges,'Normalization','pdf')
% Assigning the property of Normalization to the values of pdf will
% normalize the height of each var so the sum of bar areas equals 1

%Let's create a vector that starts at -0.05 and goes to 0.05
rq = -0.05:0.001:0.05;

% Let's use pdf function to determine pdf value of the fit
p = pdf(tFit,rq);

%Plot the pdf against rq.
hold on
plot(rq,p,'r')
hold off
```




```
% Let's use SPY returns to create t Location-Scale probability
 distribution
% object and using this fit, generate the random numbers.
```

Import data and compute returns

```
Final_Table.Date = datetime(Final_Table.Date);
dys = days(datetime(Final_Table.Date) -
 datetime(Final_Table.Date(1)));
SPYreturns = tick2ret(Final_Table.SPY,dys);
```

Fit a t scale-location distribution

```
tFit = fitdist(SPYreturns,'tlocationscale');
```

Monte-Carlo Simulations for a t distribution

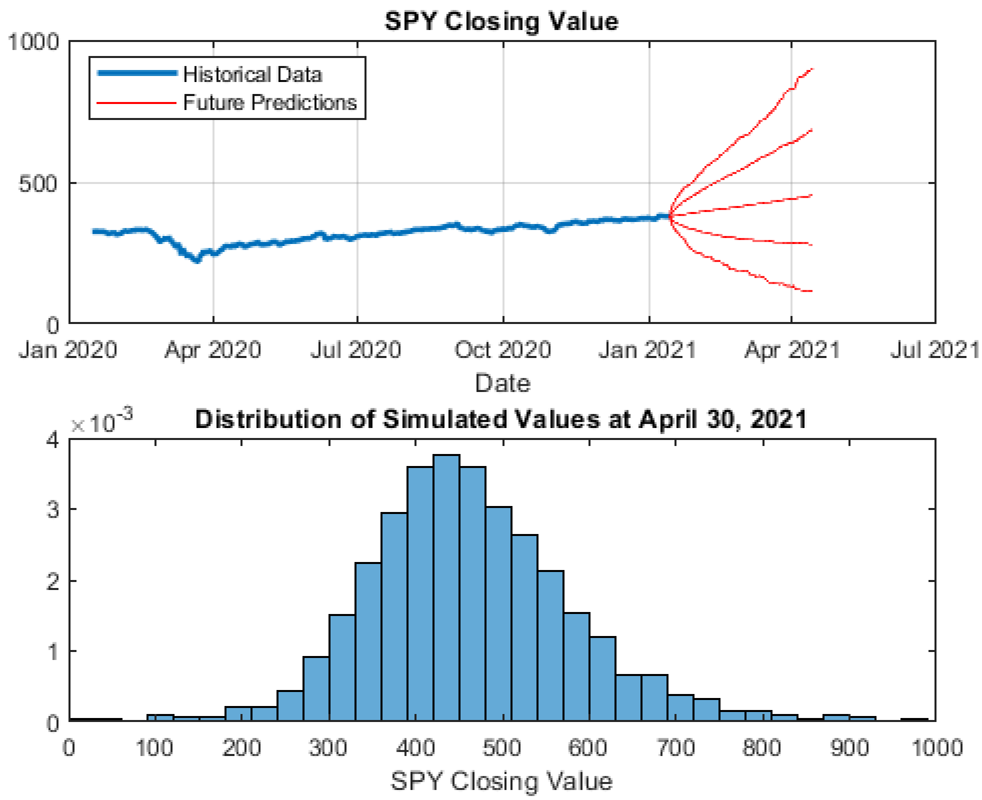
```
nSteps = 90;        % Number of steps into the future
nExp = 5e3;         % Number of random experiments to run
```

Modify simReturns to generate random numbers from the fit

```
simReturns = random(tFit,nSteps,nExp);

predictions = ret2tick(simReturns,Final_Table.SPY(end));
quantileCurves = quantile(predictions,[0.01 0.05 0.5 0.95 0.99],2);
```

Plot the returns

```
figure
subplot(2,1,1)
plot(Final_Table.Date,Final_Table.SPY,'LineWidth',2)
title('SPY Closing Value')
xlabel('Date')
grid on
hold on
plot(Final_Table.Date(end) + (0:nSteps),quantileCurves,'r')
legend('Historical Data','Future Predictions','Location','NW')
hold off

subplot(2,1,2)
histogram(predictions(end,:),'Normalization','pdf')
xlabel('SPY Closing Value')
title('Distribution of Simulated Values at April 30, 2021')
xlim([0 1000])
```



marketModel

Predicting Market Movement based on stocks on our portfolio

Import the data

```
ReturnTable = tick2ret(Final_Table(:,[2:57]));
dates = Final_Table.Date(1:end-1,:);
```

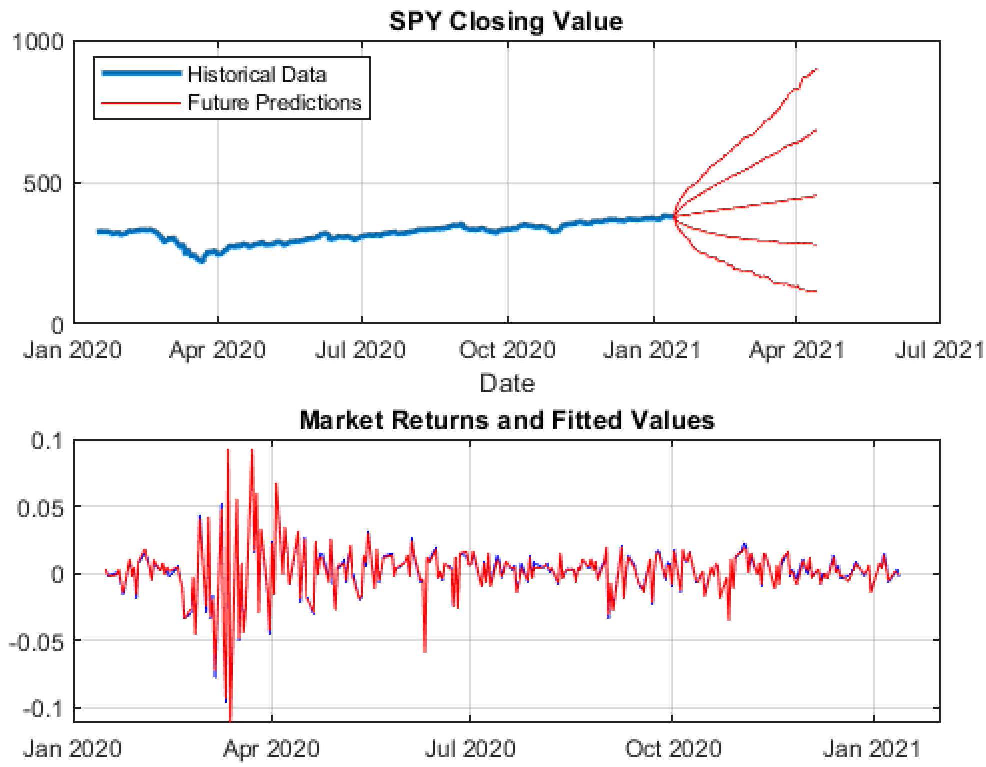
Create a matrix of factors and a vector with the market returns

```
factors = ReturnTable{:,2:end};
market = ReturnTable{:,1};
marketModelObject = fitlm(factors,market);
```

Plot the data

```
plot(dates,market,'b')
hold on
plot(dates,marketModelObject.Fitted,'r')
grid
```

```
title('Market Returns and Fitted Values')
```



Published with MATLAB® R2020b