Scott Anderwald

Case Study 4

Introduction:

For this case study time series analysis will be used to determine if certain stocks are viable candidates for investing. Data obtained from points in time can be analyzed by time series analysis. With the data obtained certain forecasting methods can be used to determine the trend of the data. For this case study momentum strategy will be used to determine the selected stock viability for investment. For the study the variable used for the analysis will be adj close since it gives a better indicator of the price.

Since the author is familar with the petroluem industry, three stocks which are believed to be a cross section of the industry will be used. Exxon (XOM) is the largest of the three with a wider portfolio including both international and domestic assets. Exxon has within it's system both upstream, midstream and downstream assets. Anadarko (APC) is similar to Exxon in regards to both international and domestic assets. Unlike Exxon, Anadarko has no downstream assets (refinement capability). The third stock being considered is Oasis Petroluem (OAS). Unlike both Exxon and Anadarko Oasis is primarly operates in Continental U.S. Oasis only has upstream operations.

Import of libraries and data retrieval:

Libraries for the case study were imported to aid in the project. Data for analysis was obtained from yahoo via pandas_datareader.

code for study was obtained from: https://www.datacamp.com/community/tutorials/finance-python-trading (https://www.datacamp.com/community/tutorials/finance-python-trading)

In [48]:
```
import pandas_datareader as pdr
import datetime
import matplotlib.pyplot as plt
import numpy as np
%matplotlib inline
```

In [49]:
```
import pandas as pd
def get(tickers, startdate, enddate):
  def data(ticker):
    return (pdr.get_data_yahoo(ticker, start=startdate, end=enddate))
  datas = map (data, tickers)
  return(pd.concat(datas, keys=tickers, names=['Ticker', 'Date']))

tickers = ['XOM', 'APC', 'OAS']
all_data = get(tickers, datetime.datetime(2015, 1, 4),
datetime.datetime(2017, 9, 25))
```

Data quality check:

When checking the quality of the data there is several steps involved. The first step will be checking the top and bottom of the dataframe by use of the .head() or .tail() method. With this step a determination can be made to see if the needed variables are present in the data.

The second step in checking data quality will be the determination of missing values which present potential issues with the alogrithms. A determination of missing values will be performed by using the .isnull() method.

If there is a missing value then the boolean logic will return true. If the values are missing and return response of true the potential cause could be the nature of time series data. In this case missing values could potentially be caused by the market closed on weekends and U.S. holidays.

Post analysis of the data quality checks shows that there appears to be no missing values. The variable of interest is present with the data.

In [50]:
```
all_data.head(n=5)
```

Out[50]:

		Open	High	Low	Close	Adj Close	Volume
Ticker	**Date**						
XOM	**2015-01-05**	92.099998	92.400002	89.500000	90.290001	81.976997	18502400
	2015-01-06	90.239998	91.410004	89.019997	89.809998	81.541191	16670700
	2015-01-07	90.650002	91.480003	90.000000	90.720001	82.367409	13590700
	2015-01-08	91.250000	92.269997	91.000000	92.230003	83.738388	15487500
	2015-01-09	92.300003	92.779999	91.370003	92.099998	83.620361	14449800

In [51]:
```
all_data.tail(n=5)
```

Out[51]:

		Open	High	Low	Close	Adj Close	Volume
Ticker	**Date**						
OAS	**2017-09-20**	8.70	9.10	8.64	8.91	8.91	9716900
	2017-09-21	8.89	8.90	8.65	8.73	8.73	5685100
	2017-09-22	8.68	8.81	8.56	8.62	8.62	4498500
	2017-09-25	8.84	9.14	8.77	9.13	9.13	8704700
	2017-09-26	9.04	9.21	8.87	9.11	9.11	9424300

Exploratory Data Analysis:

After the determination of data quality the next step is the exploratory data analysis. This step allows a quick look into the data prior to the start of the analysis.

By using the describe function and concentrating on the adj close variable the mean value of all stock prices is 50.61 dollars. The range for the stock spans from 4.29 to 92.80 dollars. The minimum price of 4.29 could potentially be from the OAS stock since it is the smallest of the three and has a greater exposure to volatility in the market.

This volatility will be explored in the next step and should confirm which stock has the greatest exposure to the swing of the market

In [52]:
```
all_data.describe()
```

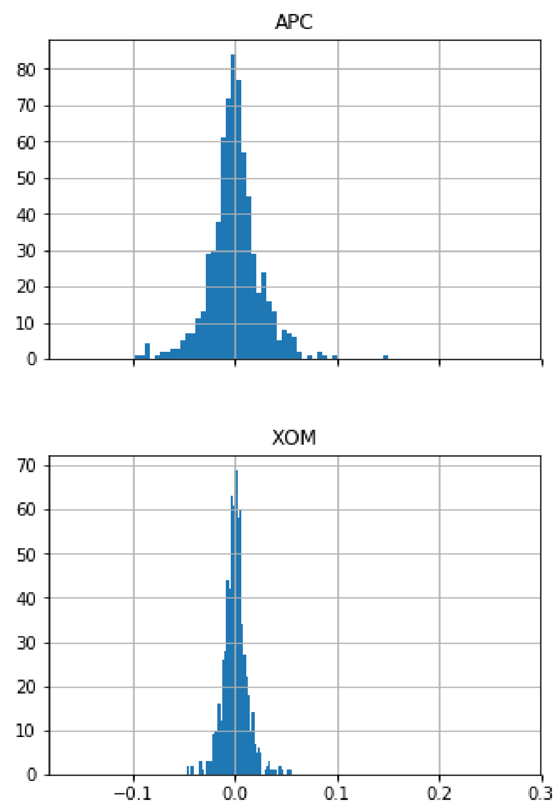
Out[52]:

	Open	High	Low	Close	Adj Close	Volume
count	2064.000000	2064.000000	2064.000000	2064.000000	2064.000000	2.064000e+03
mean	52.149283	52.762955	51.497209	52.132074	50.615332	9.489860e+06
std	31.698848	31.837824	31.532395	31.692216	30.255919	5.444934e+06
min	4.100000	4.430000	3.400000	4.290000	4.290000	1.515200e+06
25%	13.887500	14.237500	13.452500	13.882500	13.882500	5.484825e+06
50%	60.880001	61.750000	60.020001	61.024999	60.542700	8.695000e+06
75%	82.092497	82.632497	81.442501	82.019997	79.363852	1.212260e+07
max	95.440002	95.940002	94.639999	95.120003	92.801155	7.551530e+07

By taking the adj close and createing a daily percentage change displays the amount of varibility in the daily closing price of the stock. Per the study Oasis Petroluem appears have the greatest amount of variabilty with Exxon having the least. Also note both Anadarko and Exxon seemed to be normally distributed will Oasis appears to potentially bimodel.

This should be seen when a volatility index is calcucalted and graphed.

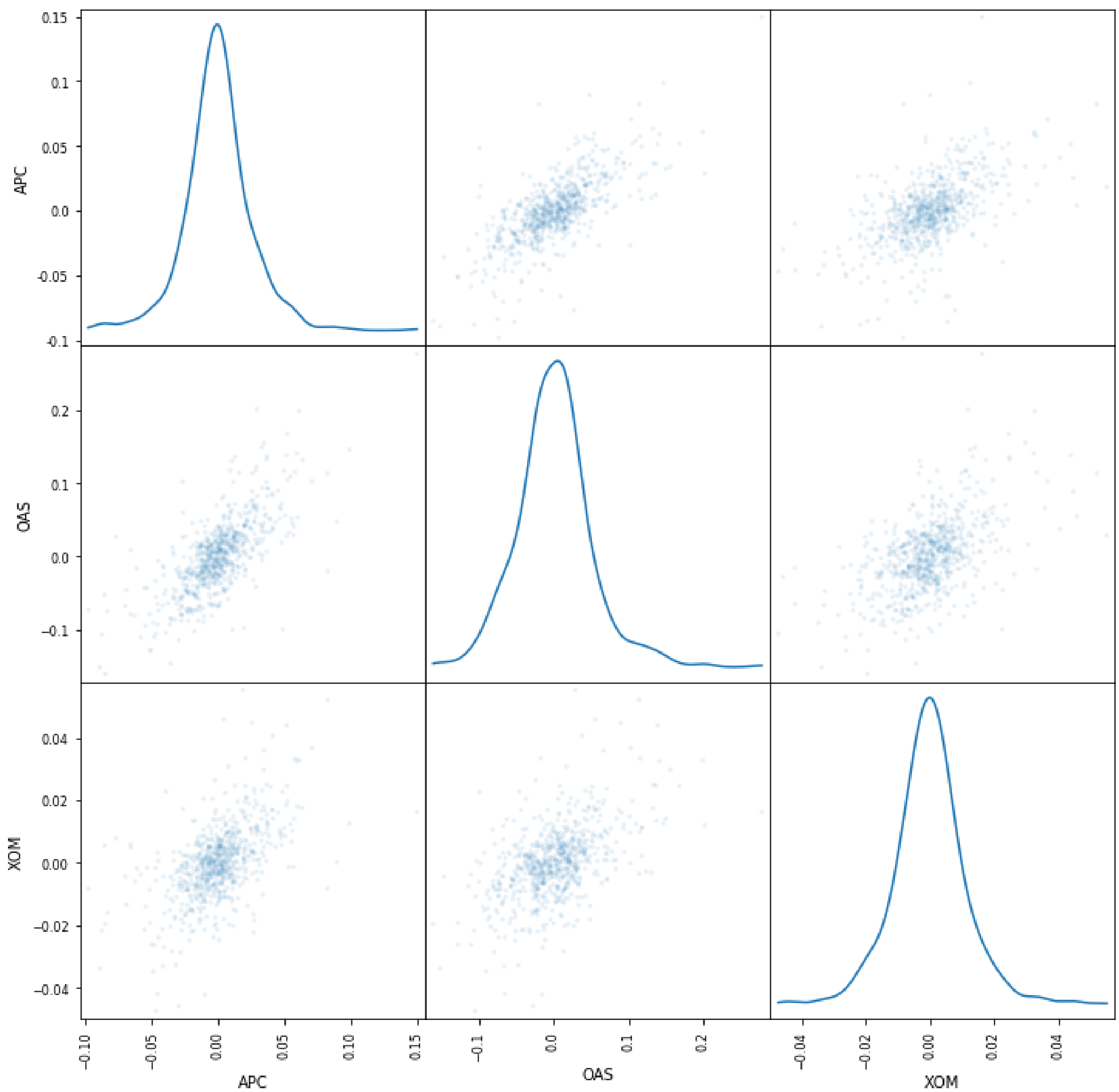
In [53]:
```
# Isolate the `Adj Close` values and transform the DataFrame
daily_close_px = all_data[['Adj Close']].reset_index().pivot('Date', 'Ti
cker', 'Adj Close')

# Calculate the daily percentage change for `daily_close_px`
daily_pct_change = daily_close_px.pct_change()

# Plot the distributions
daily_pct_change.hist(bins=50, sharex=True, figsize=(12,8))

plt.show()
```



To see if any correlation existed between the stocks a scatter matrix was plotted. Per the study correlation does exist. With Exxon and Anadarko with the greatest amount of correlation. It appears since with Oasis Petroluem has a greater exposure to market trends there is a lesser degree of correlation with both Exxon and Anadarko.

In [54]:
```
# Plot a scatter matrix with the `daily_pct_change` data
pd.scatter_matrix(daily_pct_change, diagonal='kde', alpha=0.1,figsize=(1
2,12))

# Show the plot
plt.show()
```

```
/Users/scott/anaconda/lib/python3.6/site-packages/ipykernel_launcher.p
y:2: FutureWarning: pandas.scatter_matrix is deprecated. Use pandas.plo
tting.scatter_matrix instead
```

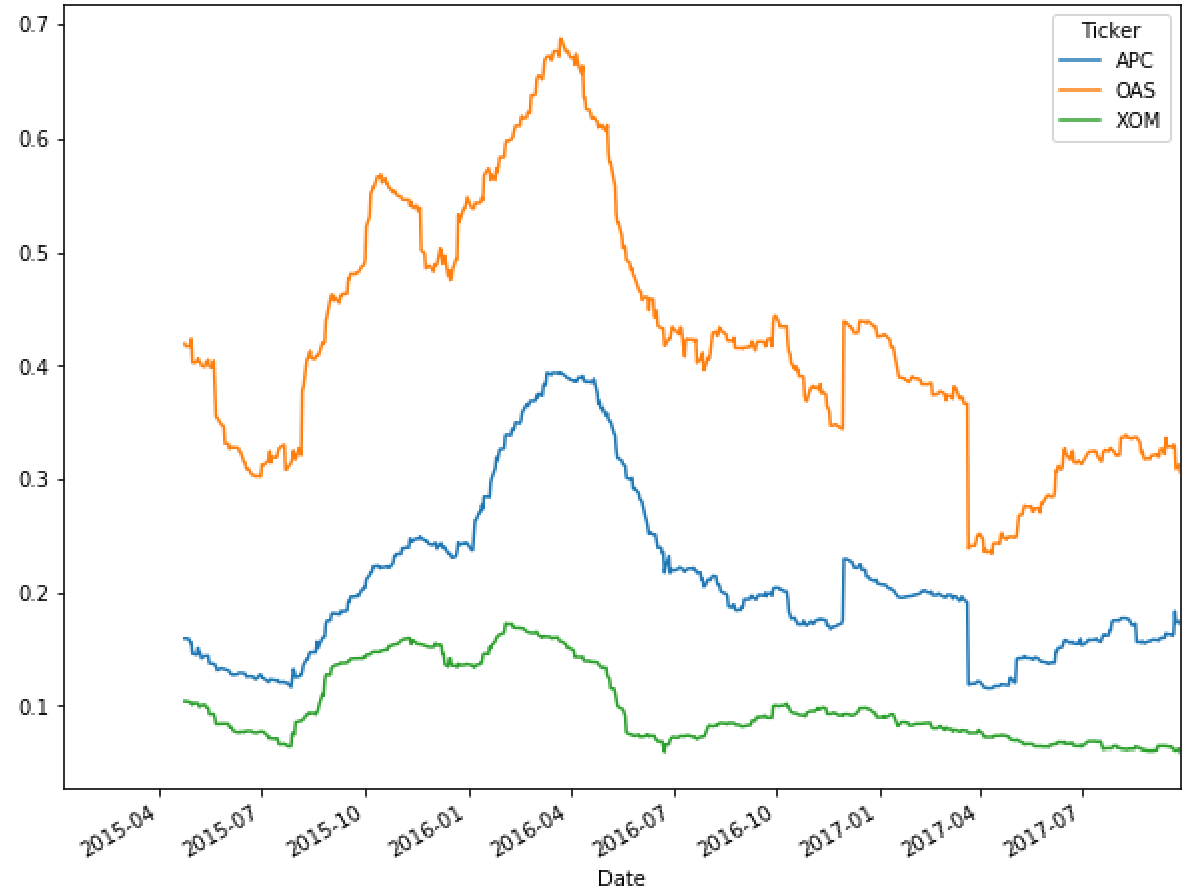


Next a volatility index was created to determine which stock could be a potential for investment. The author wanted to choose a stock that had the potential of a high return without total exposure to the volatility. With this index the greater the number indicates a higher exposure to market trends. The first step in the equation is to determine the periods, in this case 75 is typically used. Then a rolling average is calculated with the min periods.

From the graph it appears that Oasis Petroleum has the highest volatility index. The lower volatility index is assigned to Exxon which makes sense due to the business model of the company.

In [55]:
```
#define the minimum of periods to consider
min_periods = 75

#calculate the volatility
vol = daily_pct_change.rolling(min_periods).std()*np.sqrt(min_periods)
#plot the volatility
vol.plot(figsize=(10,8))
```

Out[55]: <matplotlib.axes._subplots.AxesSubplot at 0x117a98160>



Anadarko Stock:

For the remainder of the case study only one stock will be utilized. By looking at both the volatility gragh and scatter plot of the daily percentage change it appears that Anadarko might seem to be a viable stock for investment. Since APC stock was between the low volatile stock of Exxon and below the high volatile of Oasis it seem to be a good fit. This fit would allow an investor to have a fair return with some amount of volatility that could potentially have better than average returns. While staying away from the high volatile stock of Oasis where the investment potential could be considered high risk.

Data Retrieval:

Similar to the first portion of the case the data was gathered utlizing the services of yahoo.com via pandas data reader.

In [56]:
```
apc = pdr.get_data_yahoo('APC', start=datetime.date(2013,1,1),end=dateti
me.datetime(2017,9,24))
```

Data Quality Check:

Once again both the head and tail of the data will inspected via the .head() and .tail() method. Following that the .isnull() method will determine if there is missing values.

In [57]:
```
apc.head(n=5)
```

Out[57]:

	Open	High	Low	Close	Adj Close	Volume
Date						
2013-01-02	75.750000	76.080002	74.730003	76.059998	73.177605	4314700
2013-01-03	75.900002	77.349998	75.160004	76.330002	73.437386	3098500
2013-01-04	76.589996	78.480003	76.459999	78.269997	75.303841	3842200
2013-01-07	77.730003	78.930000	77.349998	78.250000	75.284599	2743700
2013-01-08	77.830002	78.860001	77.250000	78.349998	75.380814	3231000

In [58]:
```
apc.tail(n=5)
```

Out[58]:

	Open	High	Low	Close	Adj Close	Volume
Date						
2017-09-19	43.740002	44.119999	43.410000	43.759998	43.759998	2635700
2017-09-20	44.000000	45.240002	43.970001	44.810001	44.810001	4644300
2017-09-21	48.200001	48.560001	47.040001	48.490002	48.490002	15965500
2017-09-22	48.470001	49.450001	47.950001	48.830002	48.830002	7701300
2017-09-25	49.459999	50.150002	49.180000	49.930000	49.930000	6266900

In [59]:
```
pd.isnull(apc)
```

Out[59]:

	Open	High	Low	Close	Adj Close	Volume
Date						
2013-01-02	False	False	False	False	False	False
2013-01-03	False	False	False	False	False	False
2013-01-04	False	False	False	False	False	False
2013-01-07	False	False	False	False	False	False
2013-01-08	False	False	False	False	False	False
2013-01-09	False	False	False	False	False	False
2013-01-10	False	False	False	False	False	False
2013-01-11	False	False	False	False	False	False
2013-01-14	False	False	False	False	False	False
2013-01-15	False	False	False	False	False	False
2013-01-16	False	False	False	False	False	False
2013-01-17	False	False	False	False	False	False
2013-01-18	False	False	False	False	False	False
2013-01-22	False	False	False	False	False	False
2013-01-23	False	False	False	False	False	False
2013-01-24	False	False	False	False	False	False
2013-01-25	False	False	False	False	False	False
2013-01-28	False	False	False	False	False	False
2013-01-29	False	False	False	False	False	False
2013-01-30	False	False	False	False	False	False
2013-01-31	False	False	False	False	False	False
2013-02-01	False	False	False	False	False	False
2013-02-04	False	False	False	False	False	False
2013-02-05	False	False	False	False	False	False
2013-02-06	False	False	False	False	False	False
2013-02-07	False	False	False	False	False	False
2013-02-08	False	False	False	False	False	False
2013-02-11	False	False	False	False	False	False
2013-02-12	False	False	False	False	False	False
2013-02-13	False	False	False	False	False	False
...	...	...	...	...	...	...

	Open	High	Low	Close	Adj Close	Volume
Date						
2017-08-14	False	False	False	False	False	False
2017-08-15	False	False	False	False	False	False
2017-08-16	False	False	False	False	False	False
2017-08-17	False	False	False	False	False	False
2017-08-18	False	False	False	False	False	False
2017-08-21	False	False	False	False	False	False
2017-08-22	False	False	False	False	False	False
2017-08-23	False	False	False	False	False	False
2017-08-24	False	False	False	False	False	False
2017-08-25	False	False	False	False	False	False
2017-08-28	False	False	False	False	False	False
2017-08-29	False	False	False	False	False	False
2017-08-30	False	False	False	False	False	False
2017-08-31	False	False	False	False	False	False
2017-09-01	False	False	False	False	False	False
2017-09-05	False	False	False	False	False	False
2017-09-06	False	False	False	False	False	False
2017-09-07	False	False	False	False	False	False
2017-09-08	False	False	False	False	False	False
2017-09-11	False	False	False	False	False	False
2017-09-12	False	False	False	False	False	False
2017-09-13	False	False	False	False	False	False
2017-09-14	False	False	False	False	False	False
2017-09-15	False	False	False	False	False	False
2017-09-18	False	False	False	False	False	False
2017-09-19	False	False	False	False	False	False
2017-09-20	False	False	False	False	False	False
2017-09-21	False	False	False	False	False	False
2017-09-22	False	False	False	False	False	False
2017-09-25	False	False	False	False	False	False

1192 rows × 6 columns

Explanatory Data Analysis:

From the output the mean was 57.21 dollars with a max of 72.47 dollars and mean of 40.47 dollars. It appears that this follows the thought that Anadarko appears the be the best stock for investment.

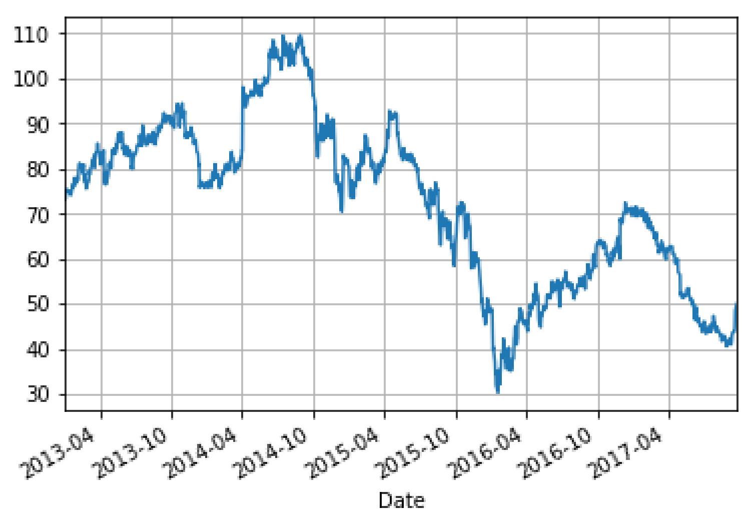
In [60]:
```
apc.describe()
```

Out[60]:

	Open	High	Low	Close	Adj Close	Volume
count	1192.000000	1192.000000	1192.000000	1192.000000	1192.000000	1.192000e+03
mean	74.022022	74.946519	73.022173	73.990990	72.449374	4.874276e+06
std	19.383755	19.426970	19.296307	19.371820	18.297147	3.153861e+06
min	30.240000	31.180000	28.160000	30.540001	30.335852	1.259200e+06
25%	57.500000	58.425002	56.767499	57.494999	57.266648	3.129275e+06
50%	78.235000	78.989998	77.140004	78.324997	75.877728	4.112000e+06
75%	88.447500	89.265000	87.412502	88.512501	85.448691	5.527850e+06
max	112.480003	113.510002	111.919998	112.690002	109.605408	4.479930e+07

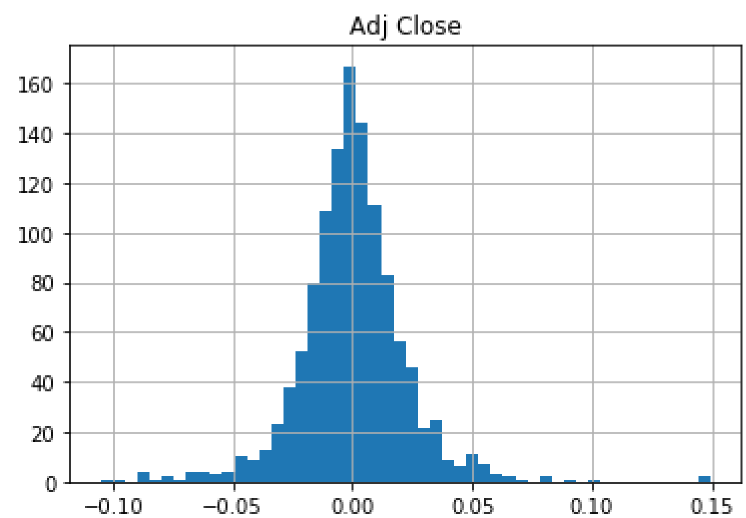
To better judge the vaibility of the investment the daily change in price will be analyzed. While the spread of the price is not as large of Oasis petroluem it is greater than that of Exxon.

In [61]:
```
apc['Adj Close'].plot(grid=True)
plt.show()
```



Calculation will now be made for a daily percentage change. This does not take into account dividends and other factors which represent change in value of a stock in a single day. Also note it appears to be normally distributed in nature.

In [62]:
```
#daily percentage change
# Assign `Adj Close` to `daily_close`
daily_close = apc[['Adj Close']]

# Daily returns
daily_pct_change = daily_close.pct_change()

# Replace NA values with 0
daily_pct_change.fillna(0, inplace=True)


# Daily log returns
daily_log_returns = np.log(daily_close.pct_change()+1)
# Plot the distribution of `daily_pct_c`
daily_pct_change.hist(bins=50)

# Show the plot
plt.show()

# Pull up summary statistics
print(daily_pct_change.describe())
```

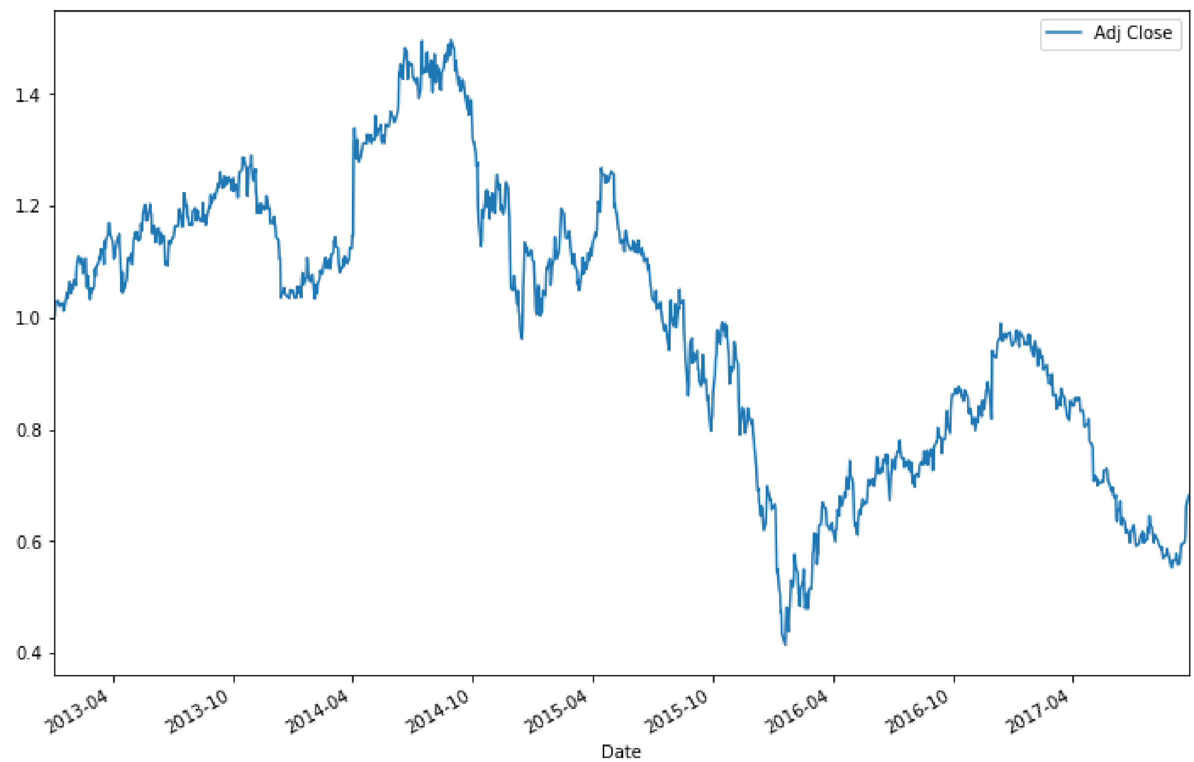



```
         Adj Close
count  1192.000000
mean     -0.000071
std       0.022382
min      -0.105245
25%      -0.011017
50%       0.000000
75%       0.010882
max       0.149435
```

Cumulative daily rate of return is used to determine value of an investment at regular intervals. The daily returns seems to be in determing the viability of an investment.

In [63]:
```
# Calculate the cumulative daily returns
cum_daily_return = (1 + daily_pct_change).cumprod()

# Print `cum_daily_return`
print(cum_daily_return.describe())

# Plot the cumulative daily returns
cum_daily_return.plot(figsize=(12,8))

# Show the plot
plt.show()
```

```
          Adj Close
count   1192.000000
mean       0.990048
std        0.250038
min        0.414551
25%        0.782571
50%        1.036898
75%        1.167689
max        1.497800
```



In [64]:
```
# Isolate the adjusted closing prices
adj_close_px = apc['Adj Close']

# Calculate the moving average
moving_avg = adj_close_px.rolling(window=50).mean()

# Inspect the result
print(moving_avg[-10:])
```

```
Date
2017-09-12    43.385671
2017-09-13    43.327395
2017-09-14    43.302888
2017-09-15    43.302551
2017-09-18    43.311608
2017-09-19    43.316071
2017-09-20    43.340534
2017-09-21    43.435003
2017-09-22    43.523686
2017-09-25    43.628777
Name: Adj Close, dtype: float64
```

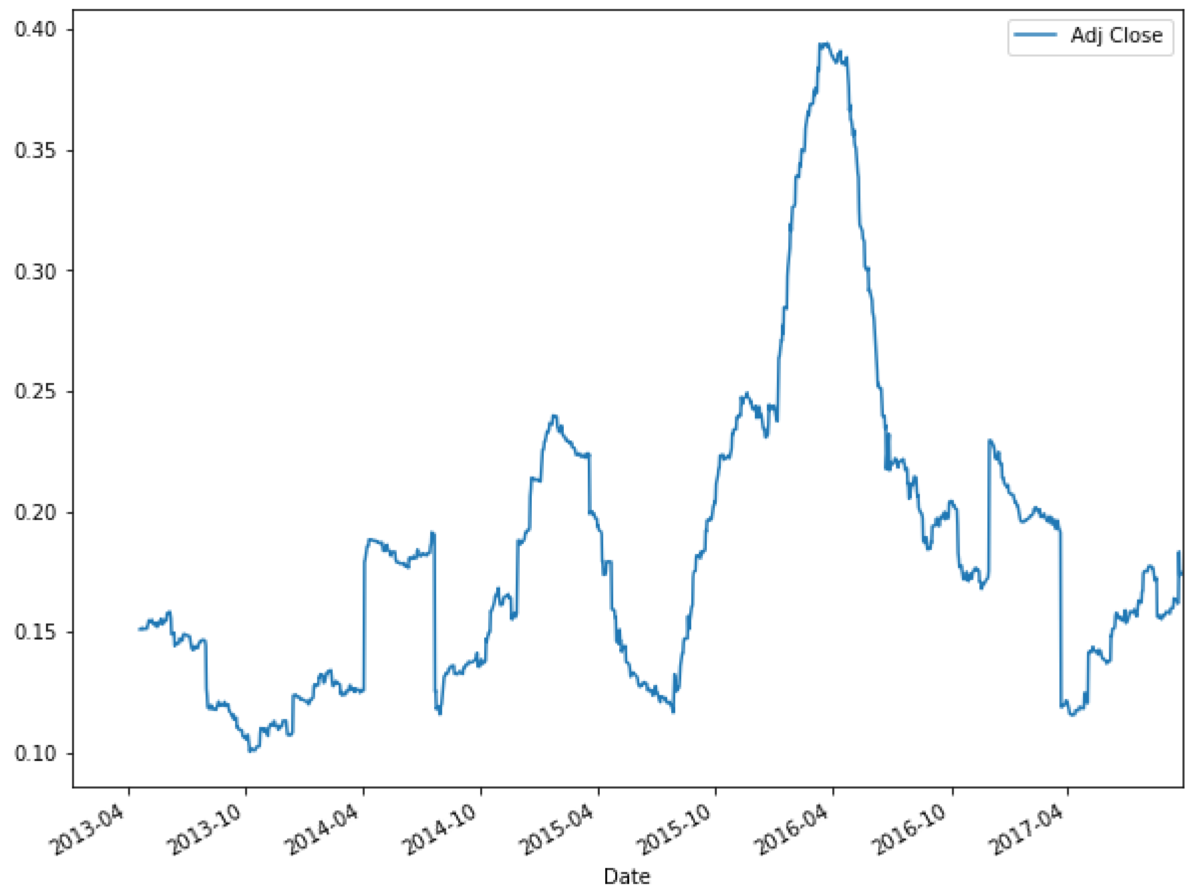
Analysis:

A better understanding of the overall trend of the stock can be obtained by utilizing rolling means averages. The graph below shows that the 50 day moving average tends to track the stock price better. While the trend for the 200 day rolling average tends to track slower in time.

In [65]:
```
# Short moving window rolling mean
apc['50'] = adj_close_px.rolling(window=50).mean()

# Long moving window rolling mean
apc['200'] = adj_close_px.rolling(window=200).mean()

# Plot the adjusted closing price, the short and long windows of rolling
 means
apc[['Adj Close', '50', '200']].plot()

# Show plot

plt.show()
```

Again the volatility of the stocks needs to be calculated. Similar to the first section with all three stocks a 75 day period will be used

In [66]:
```
# Define the minumum of periods to consider
min_periods = 75

# Calculate the volatility
vol = daily_pct_change.rolling(min_periods).std() * np.sqrt(min_periods)


# Plot the volatility
vol.plot(figsize=(10, 8))

# Show the plot
plt.show()
```

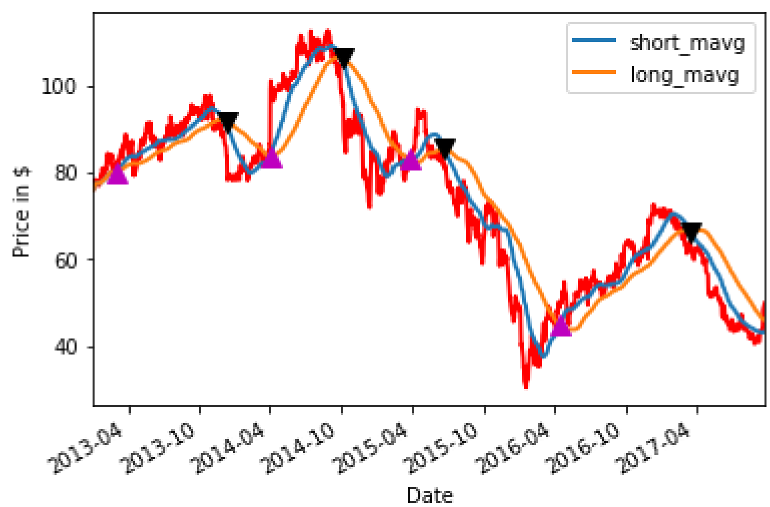


Momentum strategy:

Momentum strategy method, which is also called divergence or trend trading, will be used for the case study to develop a trading strategy. This method calculates both a short and long moving average. When these signals are graphed they signify when a trader needs to consider a long or short trading strategy.

In [67]:
```
# Initialize the short and long windows
short_window = 40
long_window = 100

# Initialize the `signals` DataFrame with the `signal` column
signals = pd.DataFrame(index=apc.index)
signals['signal'] = 0.0

# Create short simple moving average over the short window
signals['short_mavg'] = apc['Close'].rolling(window=short_window, min_pe
riods=1, center=False).mean()

# Create long simple moving average over the long window
signals['long_mavg'] = apc['Close'].rolling(window=long_window, min_peri
ods=1, center=False).mean()

# Create signals
signals['signal'][short_window:] = np.where(signals['short_mavg'][short_
window:]
                                            > signals['long_mavg'][short
_window:], 1.0, 0.0)

# Generate trading orders
signals['positions'] = signals['signal'].diff()
```

In [68]:
```
# Initialize the plot figure
fig = plt.figure()

# Add a subplot and label for y-axis
ax1 = fig.add_subplot(111,  ylabel='Price in $')

# Plot the closing price
apc['Close'].plot(ax=ax1, color='r', lw=2.)

# Plot the short and long moving averages
signals[['short_mavg', 'long_mavg']].plot(ax=ax1, lw=2.)

# Plot the buy signals
ax1.plot(signals.loc[signals.positions == 1.0].index,
         signals.short_mavg[signals.positions == 1.0],
         '^', markersize=10, color='m')

# Plot the sell signals
ax1.plot(signals.loc[signals.positions == -1.0].index,
         signals.short_mavg[signals.positions == -1.0],
         'v', markersize=10, color='k')

# Show the plot
plt.show()
```



Conclusion:

Time series can encompass many data types. For this case study the analysis of stock trading was utilized. Similar to all case studies certain steps need to be completed. These include data gathering, preperation and then analysis. For stock analysis data analysis can encompass many methods. These can include volatility analysis, moving averages and momentum strategy. Momentum strategy involves the computation of a short moving and long moving average. For this case study three individual stocks were picked to be analyzed and determined to be the best for further analysid. After looking at Exxon, Anadarko and Oasis Petroleum one stock was picked for further analysis. The analysis for the three included looking at both the daily adjusted close and the a calculated volatiltiy index. After the the primary analysis the Anadarko stock was picked for further investigation. Further analysis by means of momentum strategy revealed there were several periods where stocks could be profitable. These profibility would depend on either having a long or short position in the market.

```
In [ ]:
```