# DATA 618 Spring 2017 MiniProject 3

*James Topor*

*May 15, 2017*

This document provides a summary of the Quantopian/Python code found at the following Github link:

- https://github.com/jtopor/CUNY-MSDA-618/blob/master/P3/618-MP3-Signal-Processing.py

The algorithm makes use of a Kalman filter in an attempt to predict the movement of a security based on the value of a second independent security that is similar in nature to that of the first (i.e., from the same industry or sector and having a reasonably similar business model or purpose).

The Kalman filter calculations are based upon the sample Kalman filter code provided to us on Blackboard. However, the specifics of both the implementation of the filter and the associated trading algorithm are distinct in several aspects:

1. Contrary to the sample Blackboard code which attempted to implement a pairs trading algorithm, the algorithm used here makes use of a more aggressive "one way" trading approach, wherein the "dependent" security whose price is being estimated is traded either "long" or "short" with no offsetting "short/long" executed for "independent" security whose price is being used to estimate that of the "dependent" security.
2. The algorithm used here does not permit a trade to occur if the filter has not yielded a non-zero estimate of the price of the security to be traded. By contrast, the code provided on Blackboard allowed a trade to occur even if the estimate of the price of the dependent variable (**yhat**) was zero. As such, the sample blackboard code allowed at least one unbalanced pairs trade to occur at the start of its execution. Checking the value of the **yhat** variable to ensure that it is non-zero eliminates such behavior and ensures that the variables within the Kalman filter have attained at least semi-reasonable values prior to their being used as the basis of a trade.
3. Unlike the sample Blackboard code, the Kalman filter used here is periodically "reset" (or "re-initialized") to remove the influence of relatively outdated time series data from estimates of a security's value. The reset is controlled via a pair of user-settable global variables (**context.max__filter__iter** and **context.filter__iter**). For the results discussed below, the maximum number of filter iterations allowed prior to a filter reset was 120, which within the algorithm equates to 120 trading days due to the once daily execution of the algorithm. Once 120 filter iterations have been executed, the **filter__reset()** function is invoked to reset the values of each of the component filter variables to their original values, thereby enabling the filter to be free of the influence of relatively outdated time series data. **Repeated testing of the Kalman filter as implemented herein indicated that periodic resetting of the filter yielded signficant improvements in the algorithm's investment returns**. More research would be needed to determine whether periodic resetting should be required within other contexts as well.
4. The algorithm is invoked once daily via a scheduled function as defined within the **initialize()** function.
5. The amount of a security to be longed/shorted is determined by calculating the magnitude of the amount the estimate of the security's price exceeds its actual value: The larger the deviation, the more capital is allowed to be used for a trade. As with the sample Blackboard code, no trade is executed if that difference is within 1 standard deviation of the actual value of the security. However, if that difference exceeds 1 standard deviation, capital is allocated to a trade differently. That rule is quantified within Python as shown in the code snippet provided below. A long or short trade is then executed as appropriate. By contrast, the sample Blackboard code executed the purchase or sale of 1000 shares

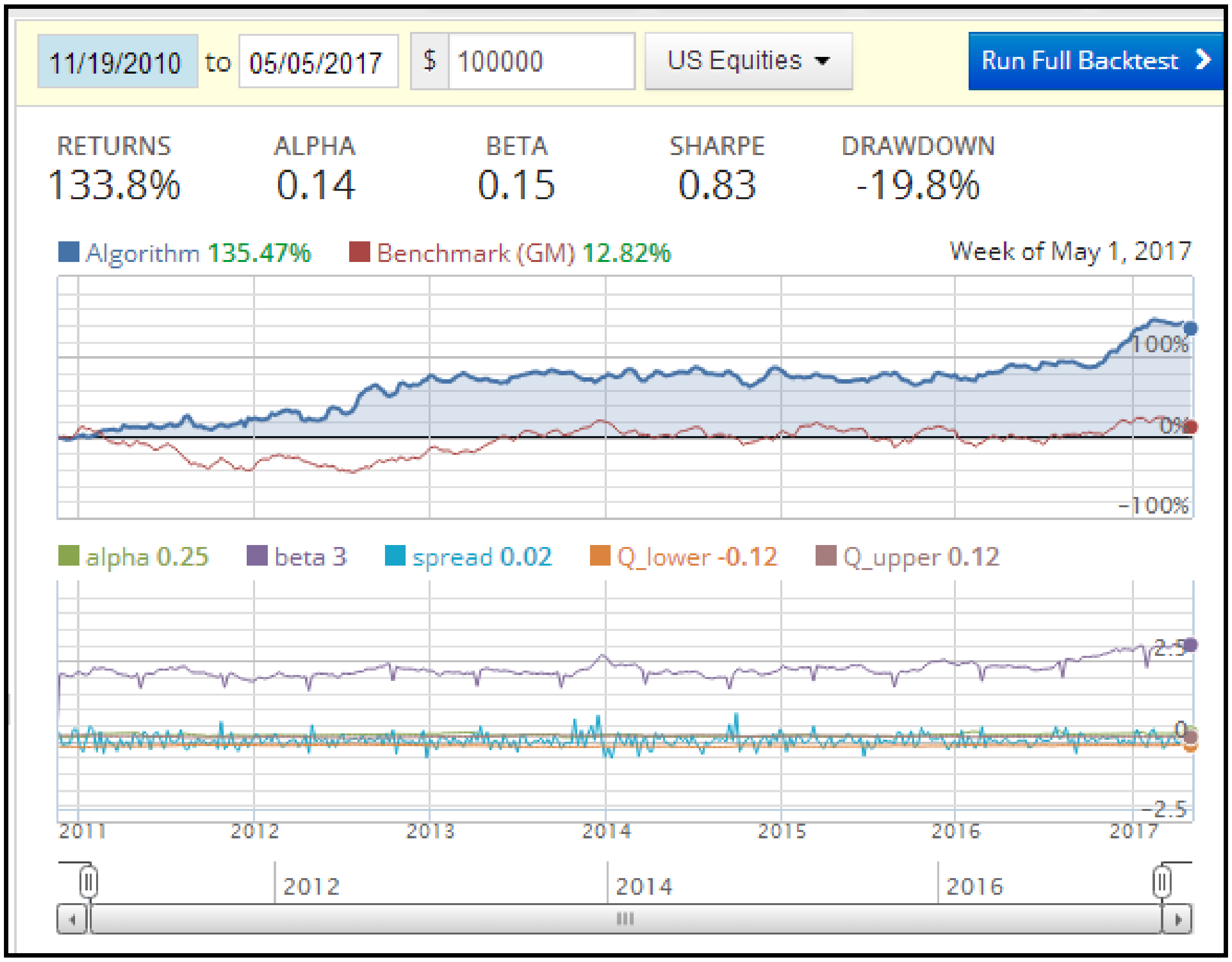
whenever the estimate of the price of the security was more than 1 standard deviation away from the actual price of the security: As such, larger deviations resulted in no additional capital being applied to a trade.

```python
# calculate how much the price estimate exceeds 1 standard deviation from actual price
trade_mag = (abs(e)/sqrt_Q) - 1

# allocate more cash to a trade based on amount estimate exceeds 1 std dev of actual price
if trade_mag <= 0.5:
    weight = .3
elif trade_mag > 0.5 and trade_mag <= 1:
    weight = 0.5
elif trade_mag > 1 and trade_mag <= 1.5:
    weight = 0.7
else: # else if difference > 2.5 standard deviations, trade 90% of cash
    weight = 0.9

# calculate the total number of shares to long or short based on weight + available cash
s2_shares = (cash * weight) / data.current(context.s2, 'price')
```

Backtesting this code as-is with $100,000 in initial capital for the period 1/4/2010 - 5/5/2017 with **Ford** as the independent security and **GM** as the security whose trades were to be determined produced (return = 133.8%, alpha = 0.14, sharpe = 0.83), an approximately 10-fold improvement over simply buying and holding GM stock, as shown in the graphic below.



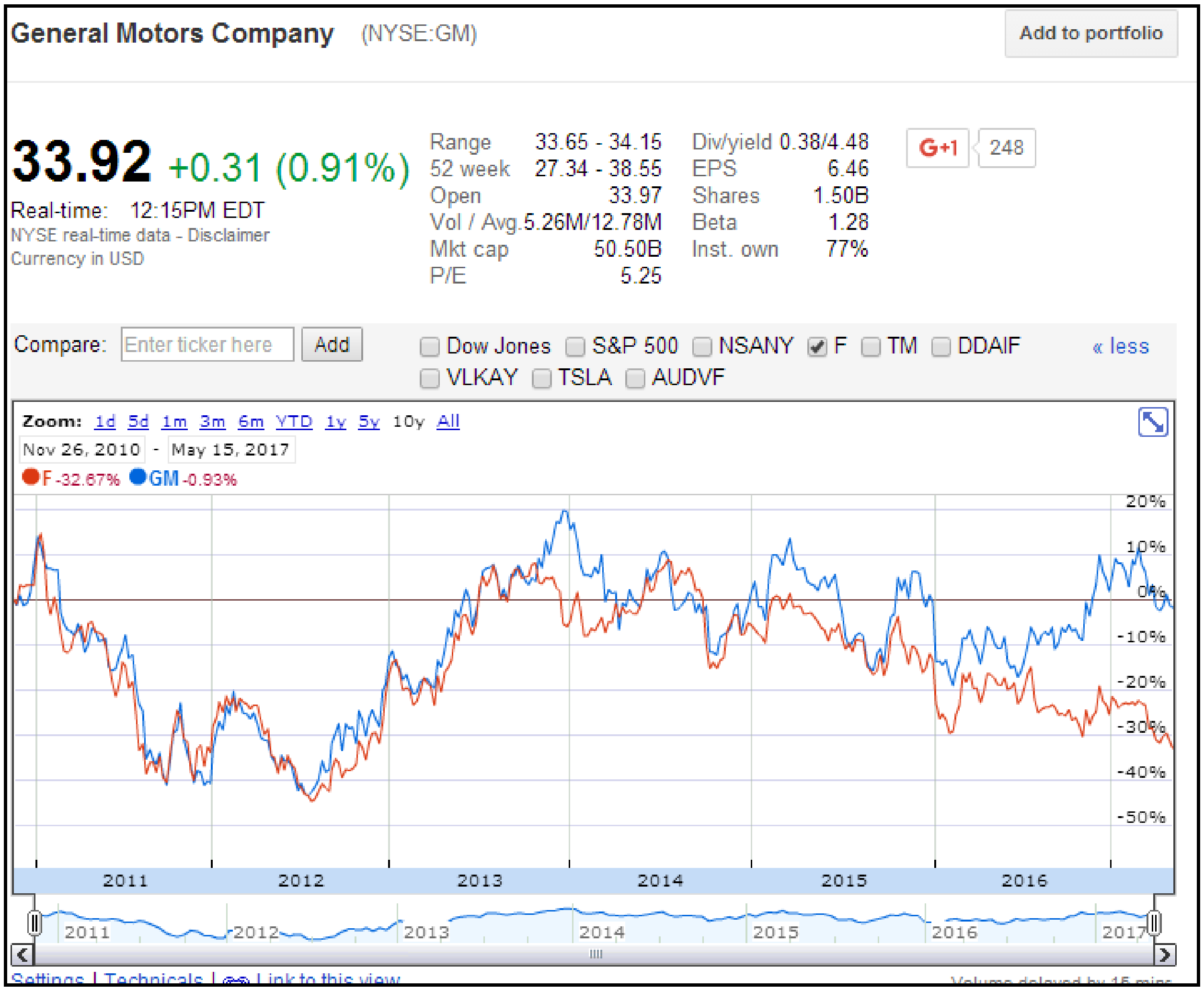
11/19/2010 to 05/05/2017 $ 100000 US Equities

| RETURNS | ALPHA | BETA | SHARPE | DRAWDOWN |
|---|---|---|---|---|
| 133.8% | 0.14 | 0.15 | 0.83 | -19.8% |

🟦 Algorithm 135.47% 🟥 Benchmark (GM) 12.82%

🟩 alpha 0.25 🟪 beta 3 🟦 spread 0.02 🟧 Q_lower -0.12 🟫 Q_upper 0.12

A plot of the historical relationship between the securities shows how closely correlated the two appear to be:



General Motors Company (NYSE:GM) Add to portfolio

33.92 +0.31 (0.91%)

Real-time: 12:15PM EDT
NYSE real-time data - Disclaimer
Currency in USD

| | | | |
|---|---|---|---|
| Range | 33.65 - 34.15 | Div/yield | 0.38/4.48 |
| 52 week | 27.34 - 38.55 | EPS | 6.46 |
| Open | 33.97 | Shares | 1.50B |
| Vol / Avg. | 5.26M/12.78M | Beta | 1.28 |
| Mkt cap | 50.50B | Inst. own | 77% |
| P/E | 5.25 | | |

Compare: Enter ticker here Add ☐ Dow Jones ☐ S&P 500 ☐ NSANY ✓ F ☐ TM ☐ DDAIF « less ☐ VLKAY ☐ TSLA ☐ AUDVF

Zoom: 1d 5d 1m 3m 6m YTD 1y 5y 10y All

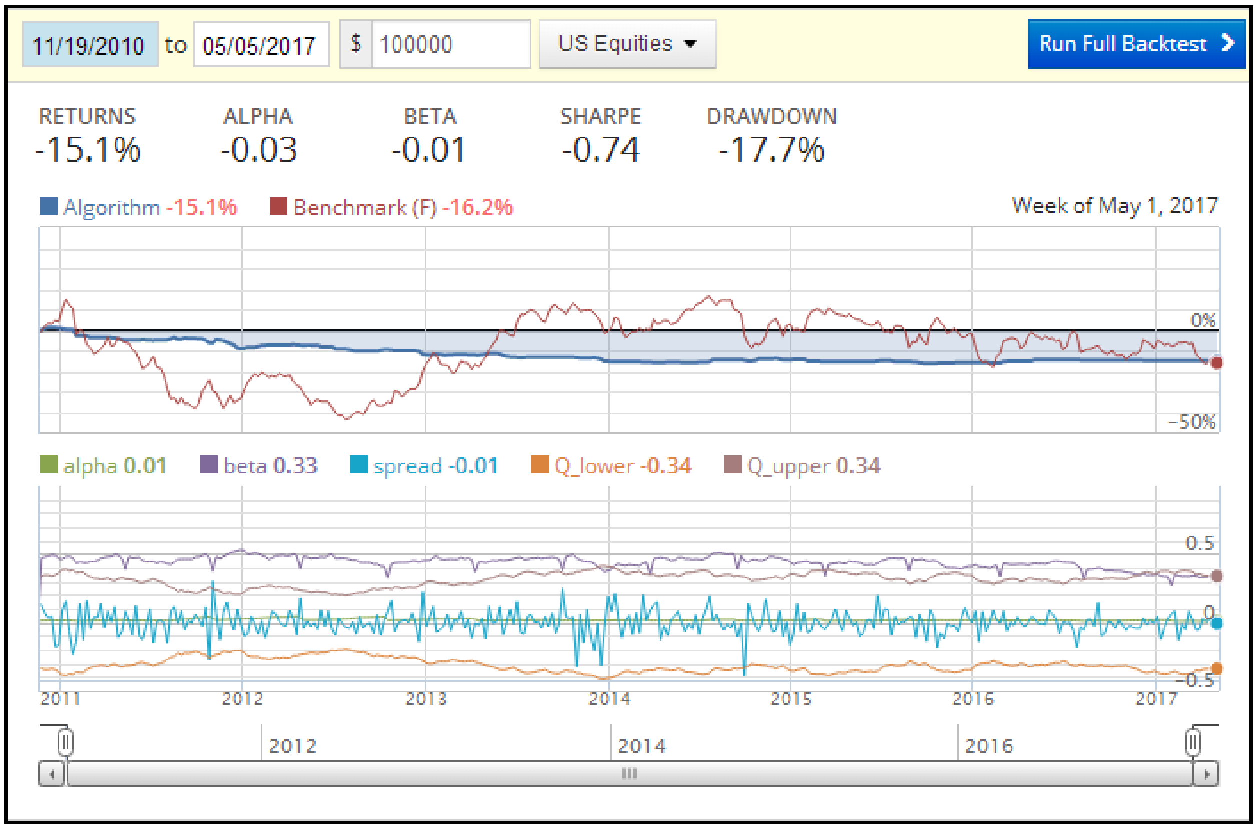
Nov 26, 2010 - May 15, 2017

● F -32.67% ● GM -0.93%

Settings | Technicals | Link to this view

Volume delayed by 15 mins

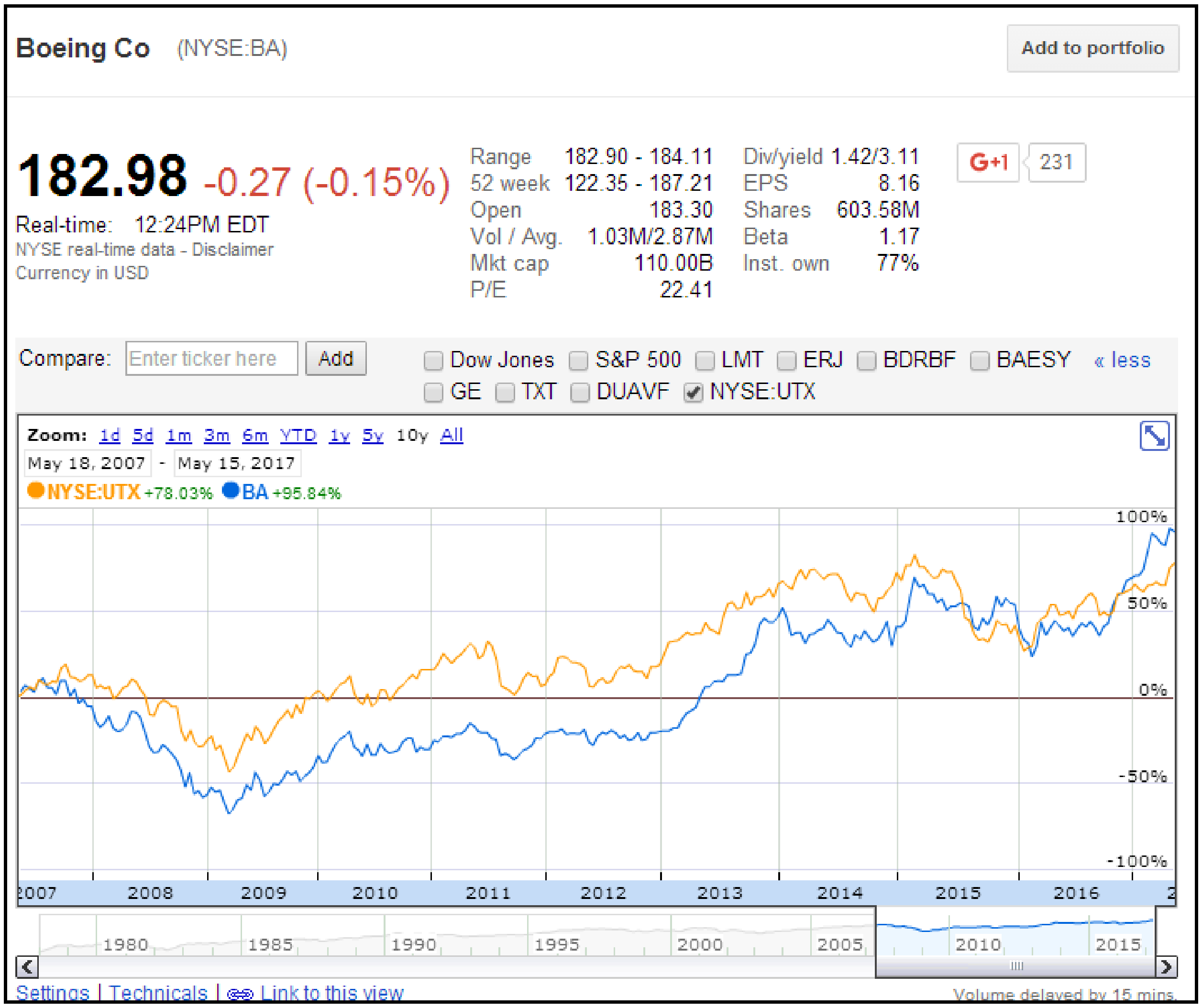
However, the trading results we obtained are not easily replicable. For example, if we instead attempt to use **GM** as the independent security and **Ford** as the security whose trade are to be determined, we don't greatly improve upon simply buying and holding Ford's stock.


11/19/2010 to 05/05/2017 $ 100000 US Equities

| RETURNS | ALPHA | BETA | SHARPE | DRAWDOWN |
|---|---|---|---|---|
| -15.1% | -0.03 | -0.01 | -0.74 | -17.7% |

**These results highlight the extreme sensitivity of the Kalman filter to how the dependent and independent variables are chosen.** Selecting "Security A" as the dependent variable and "Security B" as the independent variable can lead to drastically different results than might selecting "Security B" and "Security A" as the dependent and independent variables, respectively.

Additional testing with other pairs of securities yielded similar results. For example, if we examine the relationship between the stocks of **United Technologies (UTX)** and **Boeing (BA)**, the two appear to be fairly correlated:



**Boeing Co** (NYSE:BA)

**182.98** -0.27 (-0.15%)

Real-time: 12:24PM EDT
NYSE real-time data - Disclaimer
Currency in USD

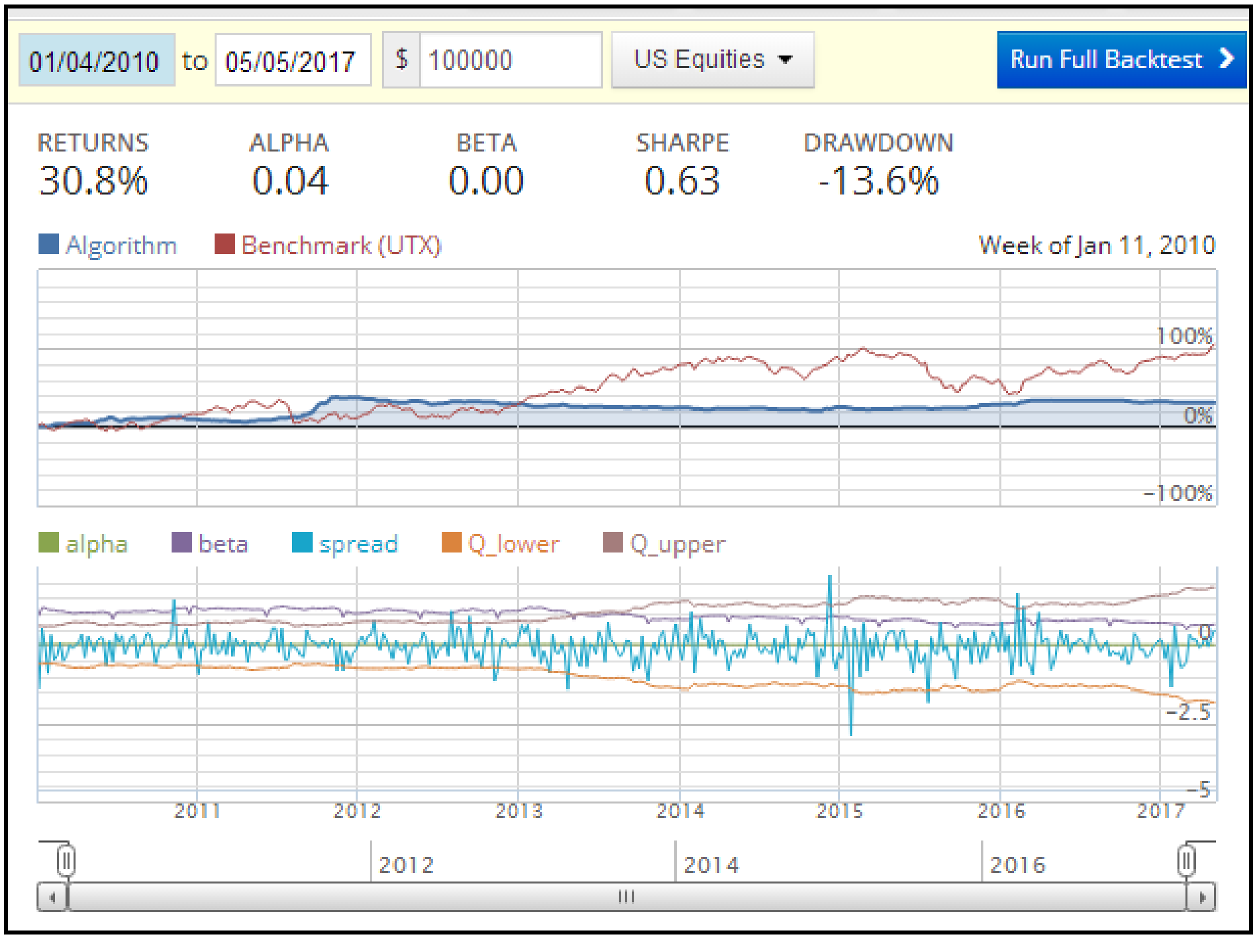
| | | | |
|---|---|---|---|
| Range | 182.90 - 184.11 | Div/yield | 1.42/3.11 |
| 52 week | 122.35 - 187.21 | EPS | 8.16 |
| Open | 183.30 | Shares | 603.58M |
| Vol / Avg. | 1.03M/2.87M | Beta | 1.17 |
| Mkt cap | 110.00B | Inst. own | 77% |
| P/E | 22.41 | | |

☐ Dow Jones ☐ S&P 500 ☐ LMT ☐ ERJ ☐ BDRBF ☐ BAESY
☐ GE ☐ TXT ☐ DUAVF ☑ NYSE:UTX

May 18, 2007 - May 15, 2017

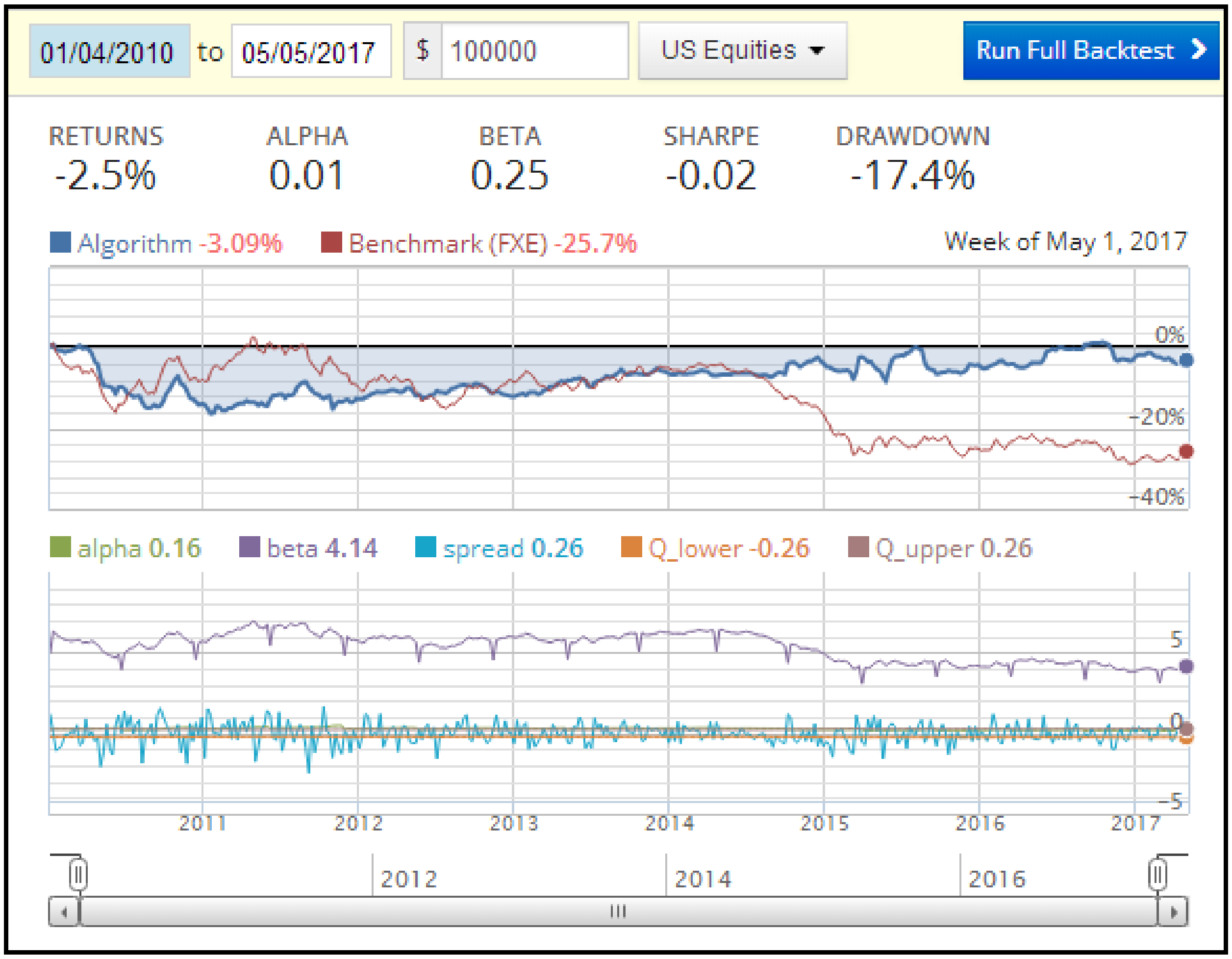
● NYSE:UTX +78.03% ● BA +95.84%

However, an attempt at predicting movements in **UTX** using **BA** did not improve upon simply buying and holding **UTX**, as shown below:


01/04/2010 to 05/05/2017 $ 100000 US Equities

| RETURNS | ALPHA | BETA | SHARPE | DRAWDOWN |
|---|---|---|---|---|
| 30.8% | 0.04 | 0.00 | 0.63 | -13.6% |

By contrast, using a \$USD ETF to predict movements of a Euro ETF did appear to offer a significant improvement over simply buying and holding the Euro ETF:


01/04/2010 to 05/05/2017 $ 100000 US Equities

RETURNS -2.5% ALPHA 0.01 BETA 0.25 SHARPE -0.02 DRAWDOWN -17.4%

Algorithm -3.09% Benchmark (FXE) -25.7% Week of May 1, 2017

alpha 0.16 beta 4.14 spread 0.26 Q_lower -0.26 Q_upper 0.26

In summary, the use of Kalman filters for predicting movements of stock prices seems to be at best an empirical endeavour. Given the sensitivity of the filter to the selection of the independent and dependent variables, great care must be taken when attempting to construct an Kalman filter algorithm to be used for purposes of buying and selling securities.