Algorithmic-Trading — Fall 2018
Alex Thomas
Colgate University

**Algorithmic-Trading - Simple Moving Average - Application: Dual Moving Average**

# Introduction

Simple Moving Average (SMA) is an elementary AT strategy. Because of this, it is also the most commonly available tutorial online. The most common application of SMA is found in dual moving average [1], which utilizes two different averages that indicate momentum swings in the market.

# Motivations and Measures

Simple moving average looks at a rolling average of a specified window. Mathematically this can be defined as: SMA($t$)= $1/n \sum_{i=t-n}^{t} x(i)$ [1] In words, this gets the average price over a specified window of time for a specified function. In context to the stock market, SMA can be applied for both short term and long term averages, with the former under an hour while the latter can be over hundreds of days. In application to the stock market and closing prices, this creates an average closing price for a specified amount of time, which gives an indicator of price swings in that period of time.

# Key Techniques

The dual moving average technique is one of the most widely used techniques. It works by taking two different SMA's - a short window and a long window. In our implementation, a 40 day window and 200 day window were chosen, giving insight into a strategy that uses longer averages. The short window crossing below the long window gives a buy signal reinforced by high trading volumes while the opposite is considered bearish and gives a sell signal. Figure 1 demonstrates this strategy graphically. The purple triangles represent a buy signal, while the black triangles represent sell signals.

# Analysis

## Effectiveness

Due to the nature of SMA, which generally looks at long periods of time to highlight trends, this can induce a lagged effect to the buy and sell orders. A lagged effect in the context of moving averages is that the current moving average doesn't react to the current trend because of this longer observation window, often making this method ineffective. This lagged effect can be fixed by looking at an Exponential Moving Average analysis [2]. In our implementation, the short window looks over a period of 40 days while the long window is 200 days. For a meaningful change to register it requires multiple days of continued poor or good performance for a sizable change to register. For example, given a recent downtrend in a particular stock over the last 7 days, the shorter window wouldn't feel that effect yet Therefore, this can cause a lagged effect, making this algorithm not very effective. Figure 1's sell signal in 2016-03 demonstrates this effect.

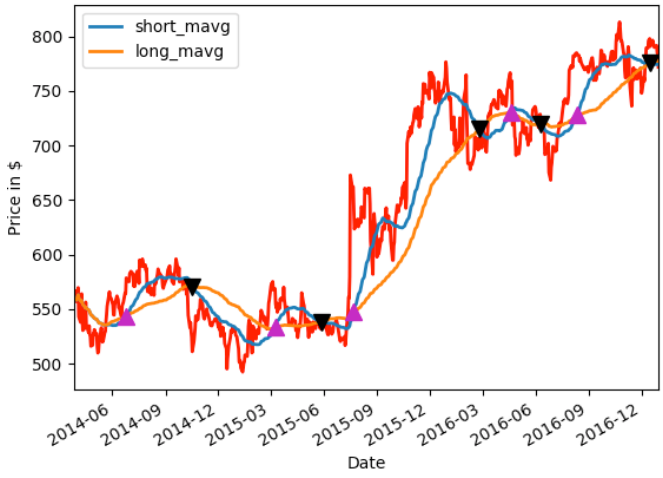


Figure 1: Dual Moving Average Strategy applied to GOOG

### Runtime

The components to consider for this strategy include pulling stock data into a Pandas data frame and calculating rolling means from the data-frame and marking their differences. Pandas is a python data analysis package and is perhaps the most powerful open source data analysis or manipulation tool available. For each calculation, it has to scrape through the entire data-frame at a rolling window, giving a linear runtime.

### Quality Metric

The strategy is tested against a baseline long position of 100 shares of a given stock. Therefore, we compare the strategy against simply buying 100 shares of stock right at the opening date and then sell at the final date. For our implementation, stocks AAPL, GOOG, DIS, FB, INTC were chosen. None of the chosen stocks outperformed the baseline performance. Facebook had by far and away the closest performance with only a -0.64% performance against the baseline. Google had the worst performance and ended up with a -30.25% performance with the strategy actually going negative profit.

### Space / Memory implications

The only space for SMA required is the data-frame, which is very reasonable.

## Conclusion

SMA is a simple AT strategy that looks at rolling window averages of closing prices for a particular stock. SMA is often applied in this dual moving average implementation where both short and long windows crossing generates buy and sell orders. However, it's performance against a baseline long position is not good, as the baseline long position for the stocks chosen always outperforms the strategy.

## Implementation

```
def execute(stock, start_date, end_date):
    stock = stockDataRetriever(stock, start_date, end_date).getStock()

    # Initialize the short and long windows and buy sell in df
    short_window = 40
    long_window = 100

    # set Pandas df with stock data
    df = pd.DataFrame(index=stock.index)

    # Create short and long simple moving average
    df['short_moving_average'] = df.movingWindow(short_window)
    df['long_moving_average'] = df.movingWindow(long_window)

    # mark signal based on comparison of the two averages
    df['signal'] = df.compare(short_moving_average, long_moving_average,1,0)

    # when signal changes from 1 to 0 or 0 to 1 - is a buy or sell
    df['positions'] = df['signal'].diff()
```

## References

[1] Lamartine Almeida Teixeira, Adriano Lorena, and Inácio De Oliveira. A method for automatic stock trading combining technical analysis and nearest neighbor classification.

[2] John Ehlers. SIGNAL ANALYSIS CONCEPTS. Technical report.