NUS FINTECH

SIMPLE MOVING AVERAGE CROSSOVER (S.M.A.C.O)

Team lead: Daniel
Researchers: Hong Rui, Jin Kai
Developers: Chuan Zhe, Jeron, Jeffery, Raj

CONTENTS

01 Introduction

02 Methodology

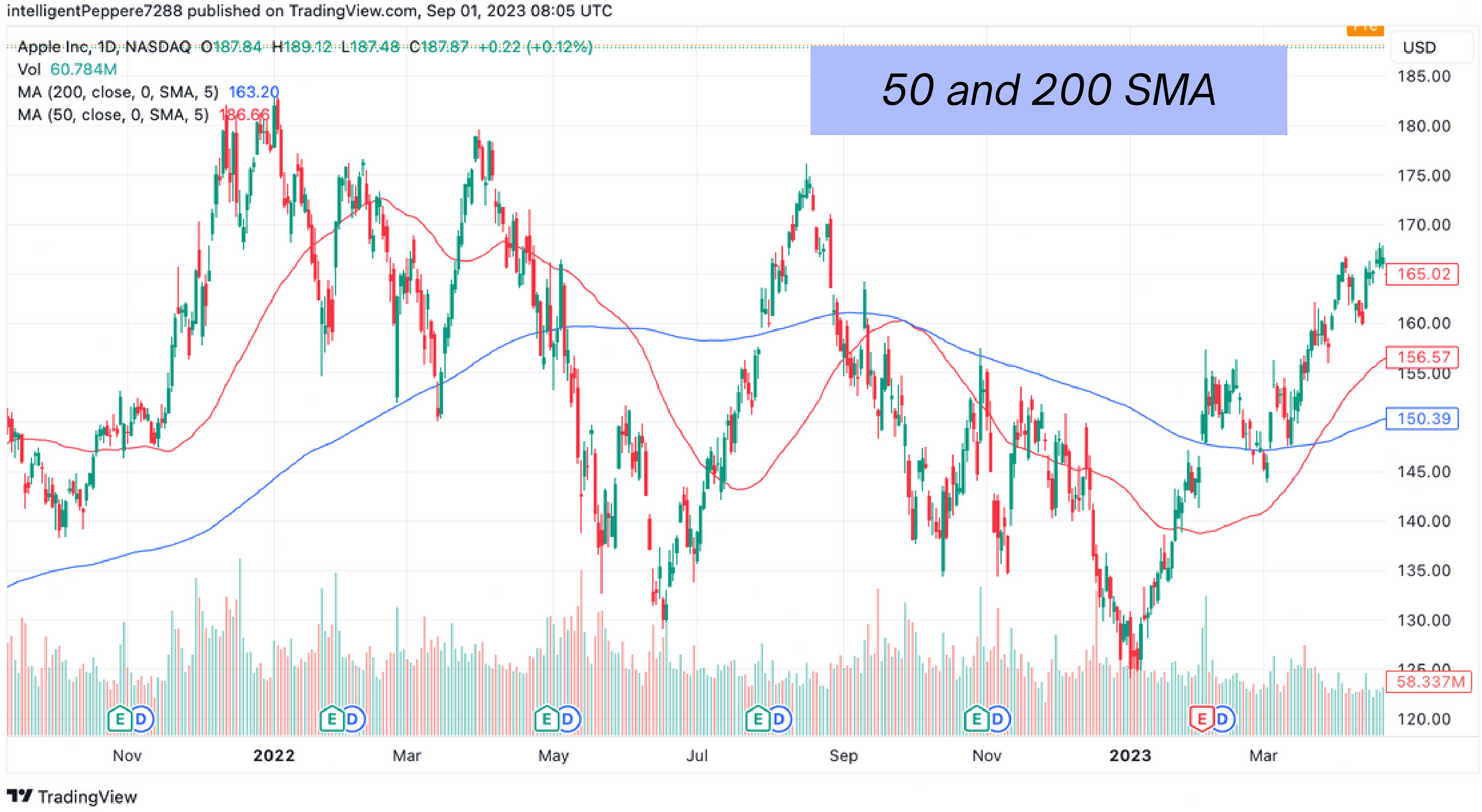
03 Results

04 Conclusion

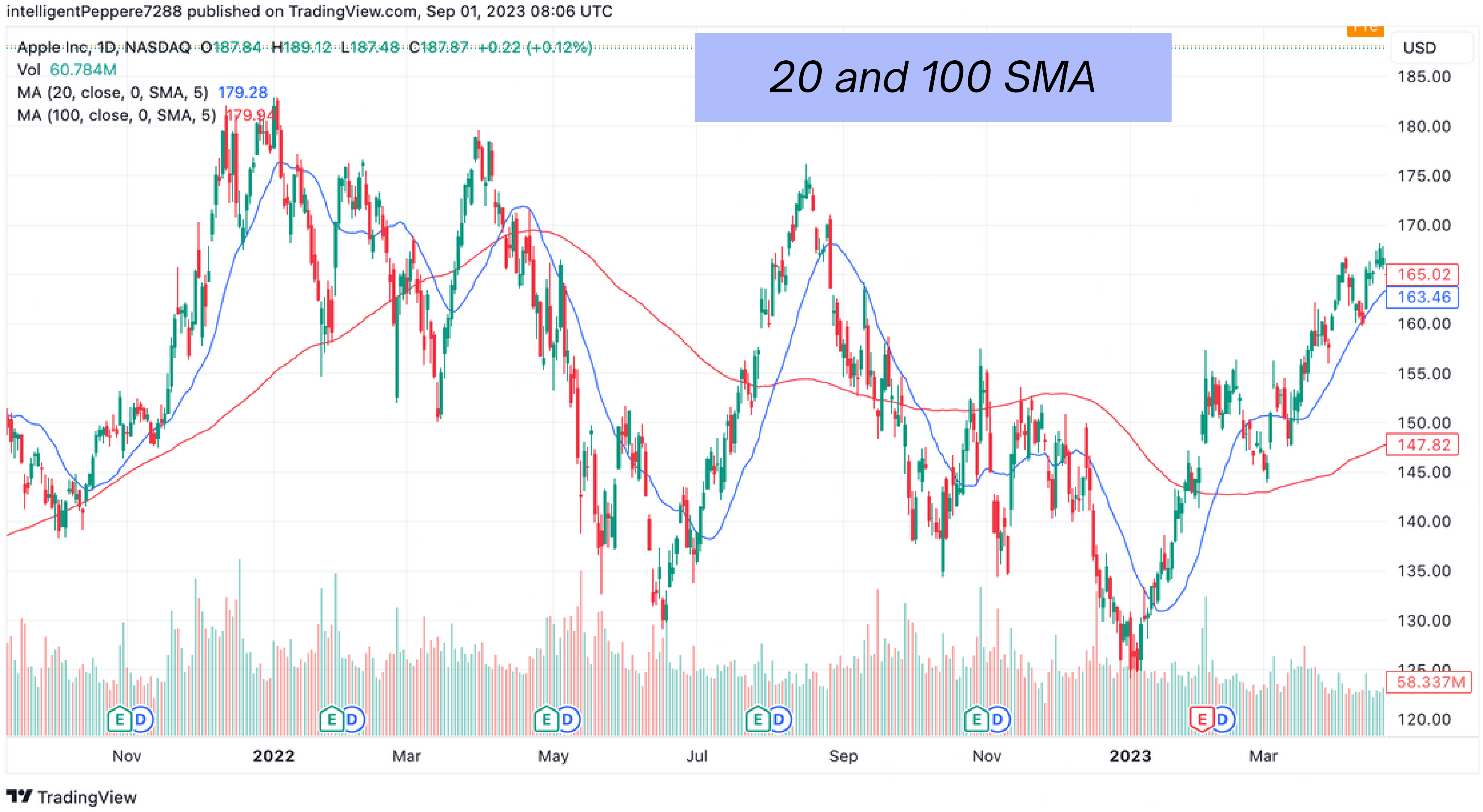
05 QNA

INTRODUCTION (RECAP)

Assume you employ the moving average crossover strategy and are currently trying to short AAPL. Which moving average pair values would you choose?



50 and 200 SMA

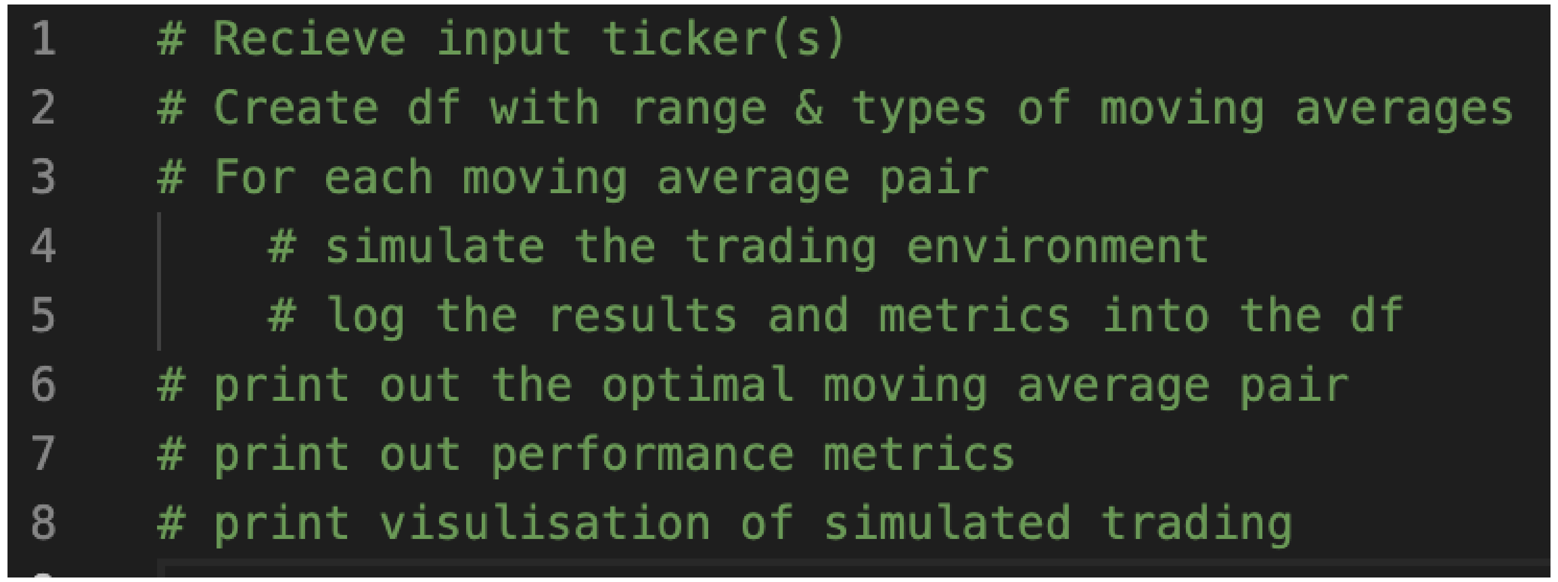


20 and 100 SMA

Observing the charts, we can see using different pairs of moving average values to produce buy and sell signals provide different returns, despite being applied on the same asset. *How can we find the optimal pairs of values for the moving average strategy?*

PROPOSED PROJECT METHODOLOGY

Pseudo-code



```
# Recieve input ticker(s)
# Create df with range & types of moving averages
# For each moving average pair
    # simulate the trading environment
    # log the results and metrics into the df
# print out the optimal moving average pair
# print out performance metrics
# print visulisation of simulated trading
```

Considerations / Discussion pointers

- How do we minimize the time required by the for loop?
    - Minimising the range of values by having a clearly defined investment strategy.
    - Any progamming methodologies to explore?
- How do we prevent the algorithm from overfitting?
    - Partly based on understanding of technical analysis.
    - How does the algorithm perform in black swan events?
- What metrics do we care about?
- How can we make the algorithm more useful?
    - Include recommended stop losses?

OUR BRAINSTORMS

Ideation 1: Brute Force Method (Using triple for loops)

- Time Complexity: O(n^3)
- Average run time: 12255.121 ms

Ideation 2: Caching using Python's lru_cache decorator (sliding window)

- the current state of a strategy portfolio can be stored in a cache until it is rebalanced, such that the list doesn't need to be regenerated upon each loop of the trading algorithm
- The lru_cache decorator evicts existing entries only when there's no more space to store new listings. With sufficient space, entries in the cache will live forever and never get refreshed. Could implement an expiration time into a new decorator that extends lru_cache

WINNING IDEATION

Ideation 3: Parellelisation (w some caching)

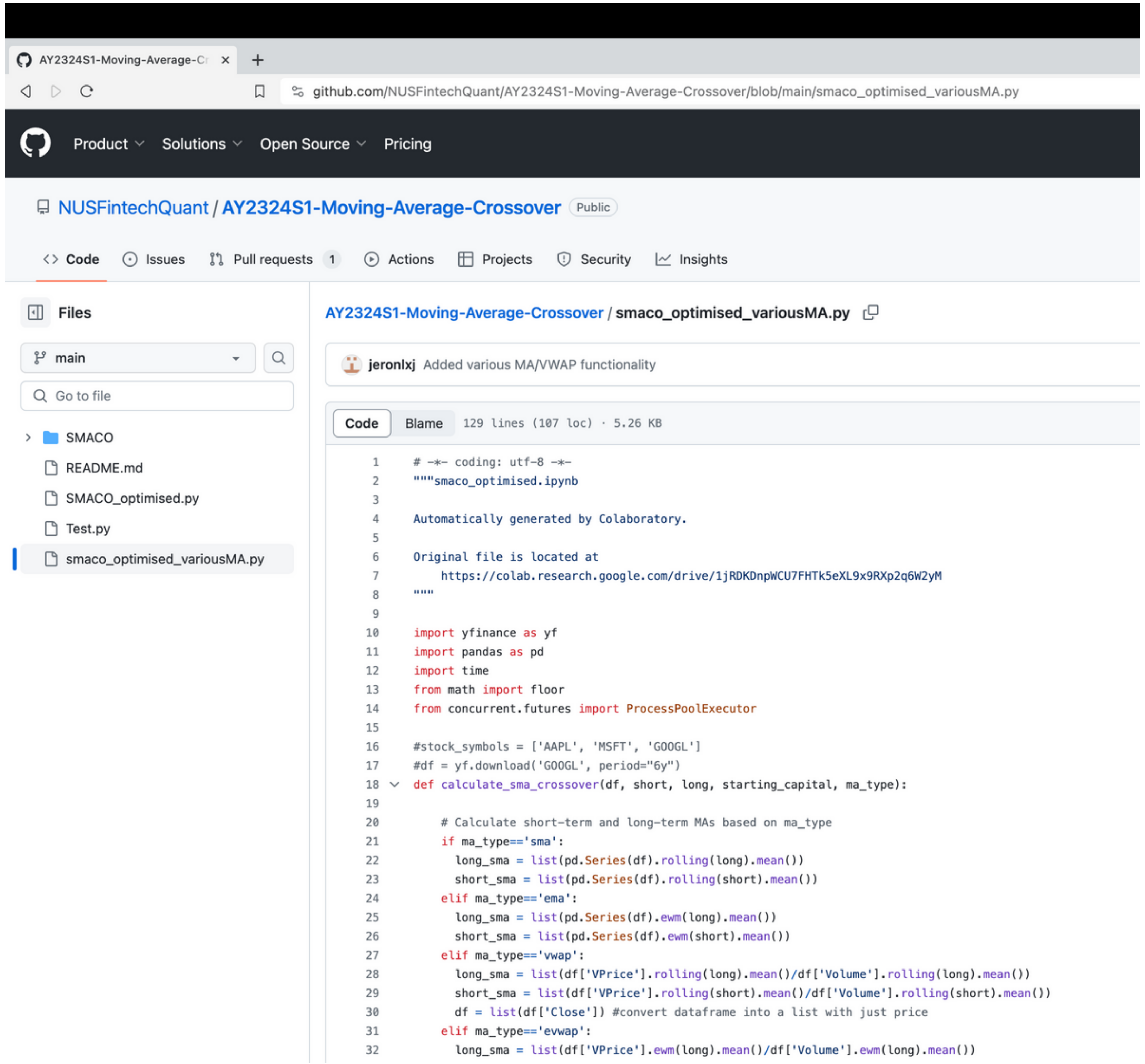
- Parallelisation - with Python concurrent.futures
- Many operations in algorithmic trading systems can be done concurrently. So-called "embarrassingly parallel" algorithms include steps that can be computed fully independently of other steps
- Same Time Complexity, with performance gains of:

$$\frac{1}{F + \frac{1-F}{P}}$$

F is the fraction of the program that is serial
P is the number of processors

OUR CODE

Our implementation ... check it out on Github!


AY2324S1-Moving-Average-Cr
github.com/NUSFintechQuant/AY2324S1-Moving-Average-Crossover/blob/main/smaco_optimised_variousMA.py
Product
Solutions
Open Source
Pricing
NUSFintechQuant / AY2324S1-Moving-Average-Crossover
Public
Code
Issues
Pull requests 1
Actions
Projects
Security
Insights
Files
main
Go to file
SMACO
README.md
SMACO_optimised.py
Test.py
smaco_optimised_variousMA.py
AY2324S1-Moving-Average-Crossover / smaco_optimised_variousMA.py
jeronlxj Added various MA/VWAP functionality
Code
Blame
129 lines (107 loc) · 5.26 KB

```python
# -*- coding: utf-8 -*-
"""smaco_optimised.ipynb

Automatically generated by Colaboratory.

Original file is located at
    https://colab.research.google.com/drive/1jRDKDnpWCU7FHTk5eXL9x9RXp2q6W2yM
"""

import yfinance as yf
import pandas as pd
import time
from math import floor
from concurrent.futures import ProcessPoolExecutor

#stock_symbols = ['AAPL', 'MSFT', 'GOOGL']
#df = yf.download('GOOGL', period="6y")
def calculate_sma_crossover(df, short, long, starting_capital, ma_type):

    # Calculate short-term and long-term MAs based on ma_type
    if ma_type=='sma':
      long_sma = list(pd.Series(df).rolling(long).mean())
      short_sma = list(pd.Series(df).rolling(short).mean())
    elif ma_type=='ema':
      long_sma = list(pd.Series(df).ewm(long).mean())
      short_sma = list(pd.Series(df).ewm(short).mean())
    elif ma_type=='vwap':
      long_sma = list(df['VPrice'].rolling(long).mean()/df['Volume'].rolling(long).mean())
      short_sma = list(df['VPrice'].rolling(short).mean()/df['Volume'].rolling(short).mean())
      df = list(df['Close']) #convert dataframe into a list with just price
    elif ma_type=='evwap':
      long_sma = list(df['VPrice'].ewm(long).mean()/df['Volume'].ewm(long).mean())
```

Function Descriptions

main(ticker, period, interval, starting_capital)

- The range of values is initialized from 50 to 200, in line with the trading strategy developed by our researchers.

best_pair_for_all()

- Master function that takes in chosen ticker, interval and starting capital that outputs best SMA pair for each interval based on greatest (positive) change to starting capital

calculate_sma_crossover(df, short, long,starting_capital)

- calculates net capital after executing short/long trade based on direction of crossover

optimise_sma()

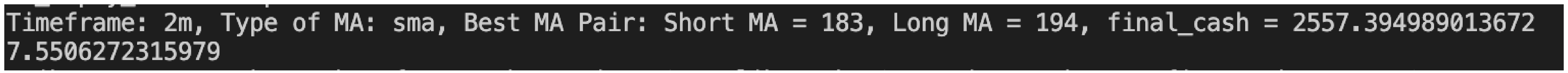
- Bridging function between main and calculatesmacrossover()

ALGO RESULTS

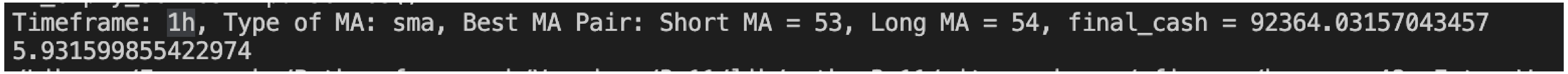
Stock: NVDA, Starting Capital: $10,000

Timeframe	Type of MA	(Smaller MA, Larger MA)	Time (ms)	Profit (dollars)	Profit (%)
2m	SMA	(183, 194)	7550.63	2557.39	-74.42 %
5m	SMA	(108, 178)	5387.70	3260.57	-67.39 %
15m	SMA	(54, 134)	2882.51	4132.67	-58.67 %
30m	SMA	(105, 134)	2372.49	4028.07	-59.72 %
1h	SMA	(53, 54)	5931.60	92364.03	823.64 %

Sample outputs



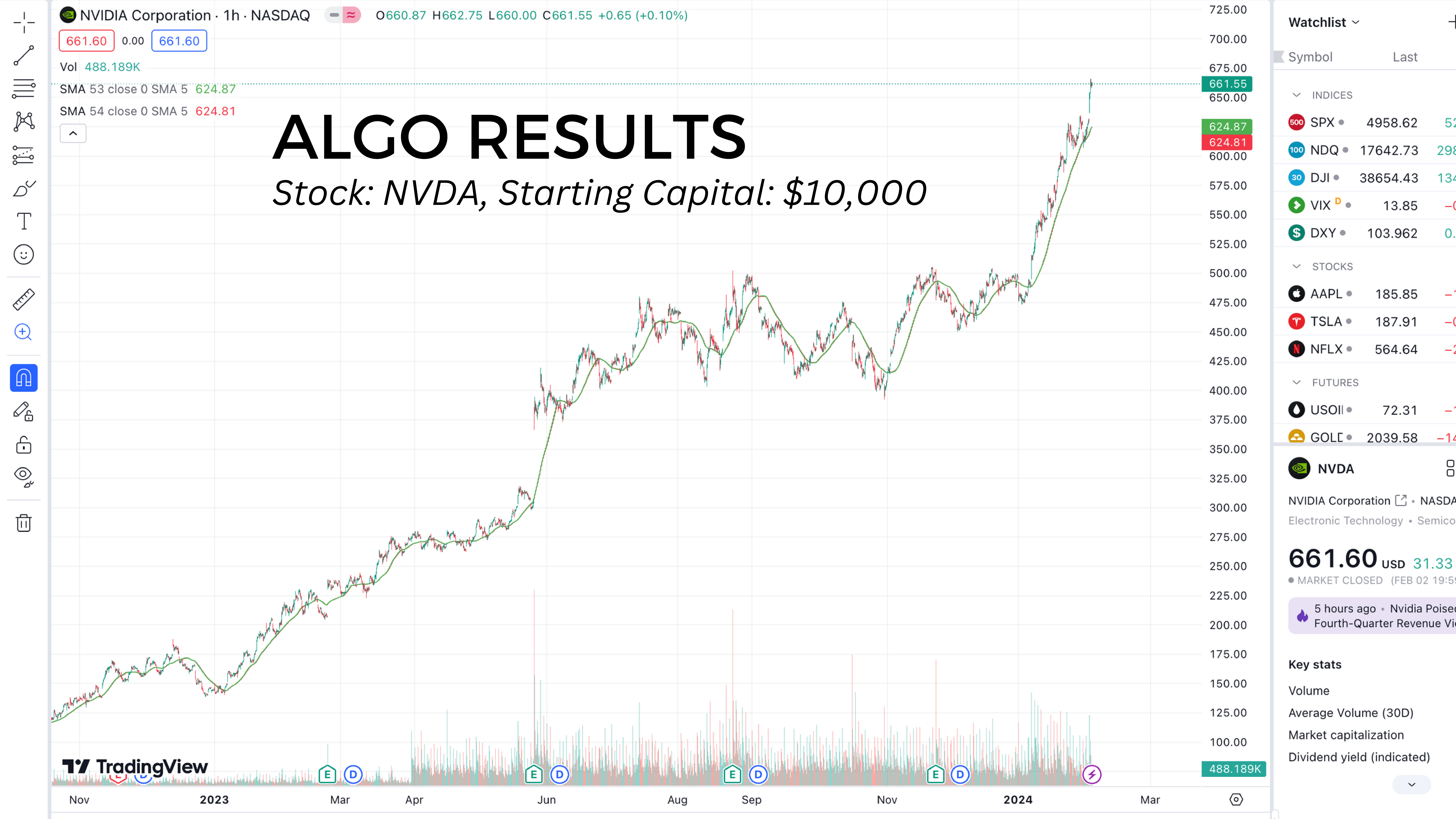
```
Timeframe: 2m, Type of MA: sma, Best MA Pair: Short MA = 183, Long MA = 194, final_cash = 2557.394989013672
7.5506272315979
```



```
Timeframe: 1h, Type of MA: sma, Best MA Pair: Short MA = 53, Long MA = 54, final_cash = 92364.03157043457
5.931599855422974
```

ALGO RESULTS

Stock: NVDA, Starting Capital: $10,000



NVIDIA Corporation · 1h · NASDAQ O660.87 H662.75 L660.00 C661.55 +0.65 (+0.10%)

661.60 0.00 661.60

Vol 488.189K

SMA 53 close 0 SMA 5 624.87

SMA 54 close 0 SMA 5 624.81

Watchlist

Symbol	Last	
INDICES		
SPX	4958.62	52
NDQ	17642.73	298
DJI	38654.43	134
VIX	13.85	−0
DXY	103.962	0.
STOCKS		
AAPL	185.85	−1
TSLA	187.91	−0
NFLX	564.64	−2
FUTURES		
USOIL	72.31	−1
GOLD	2039.58	−14

NVDA

NVIDIA Corporation • NASDA

Electronic Technology • Semico

661.60 USD 31.33

MARKET CLOSED (FEB 02 19:5

5 hours ago • Nvidia Poised Fourth-Quarter Revenue Vi

Key stats

Volume

Average Volume (30D)

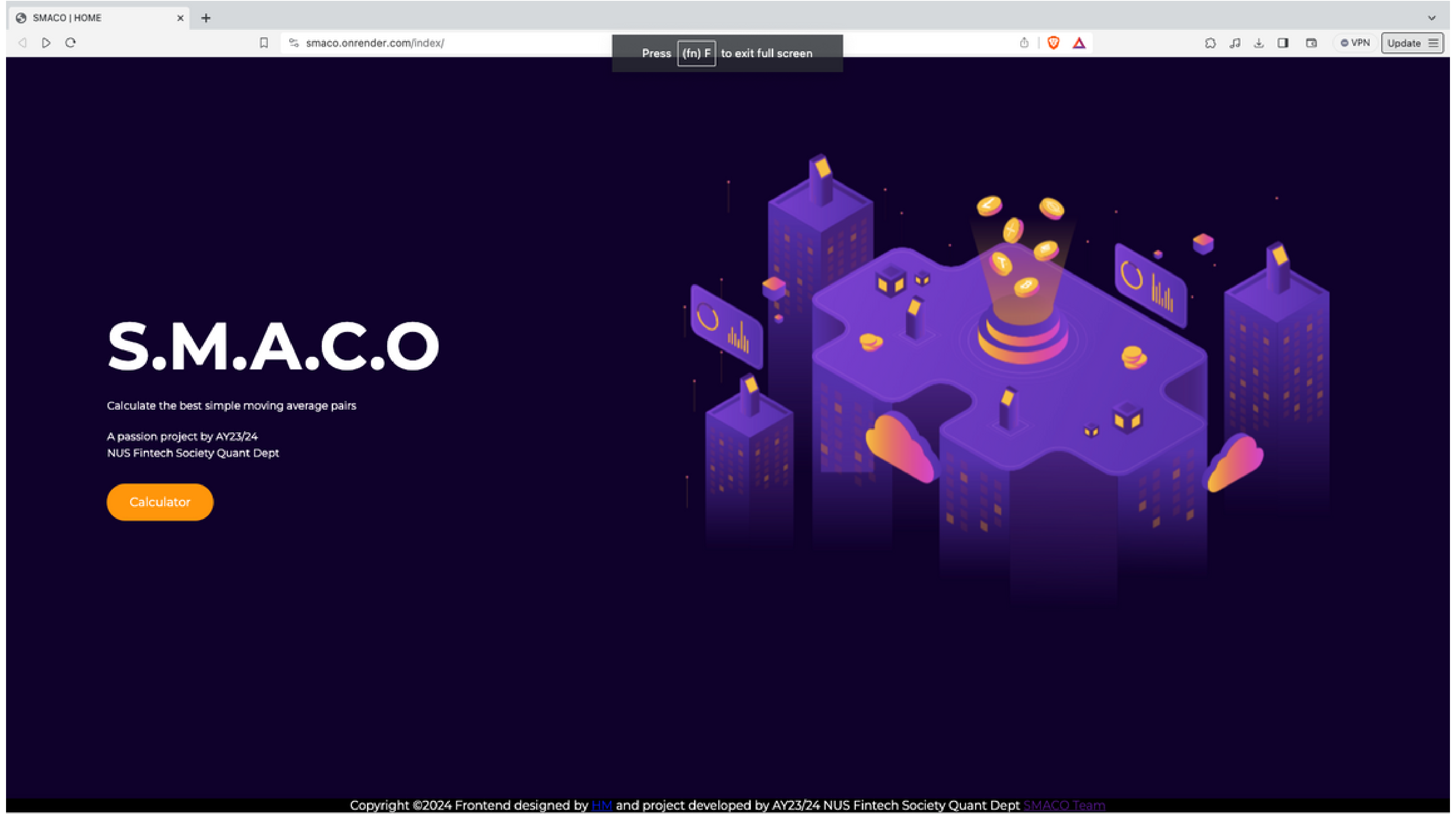
Market capitalization

Dividend yield (indicated)

Cheeky intermission …. Presenting;

FRONTEND BY RAJ

https://smaco.onrender.com/index/

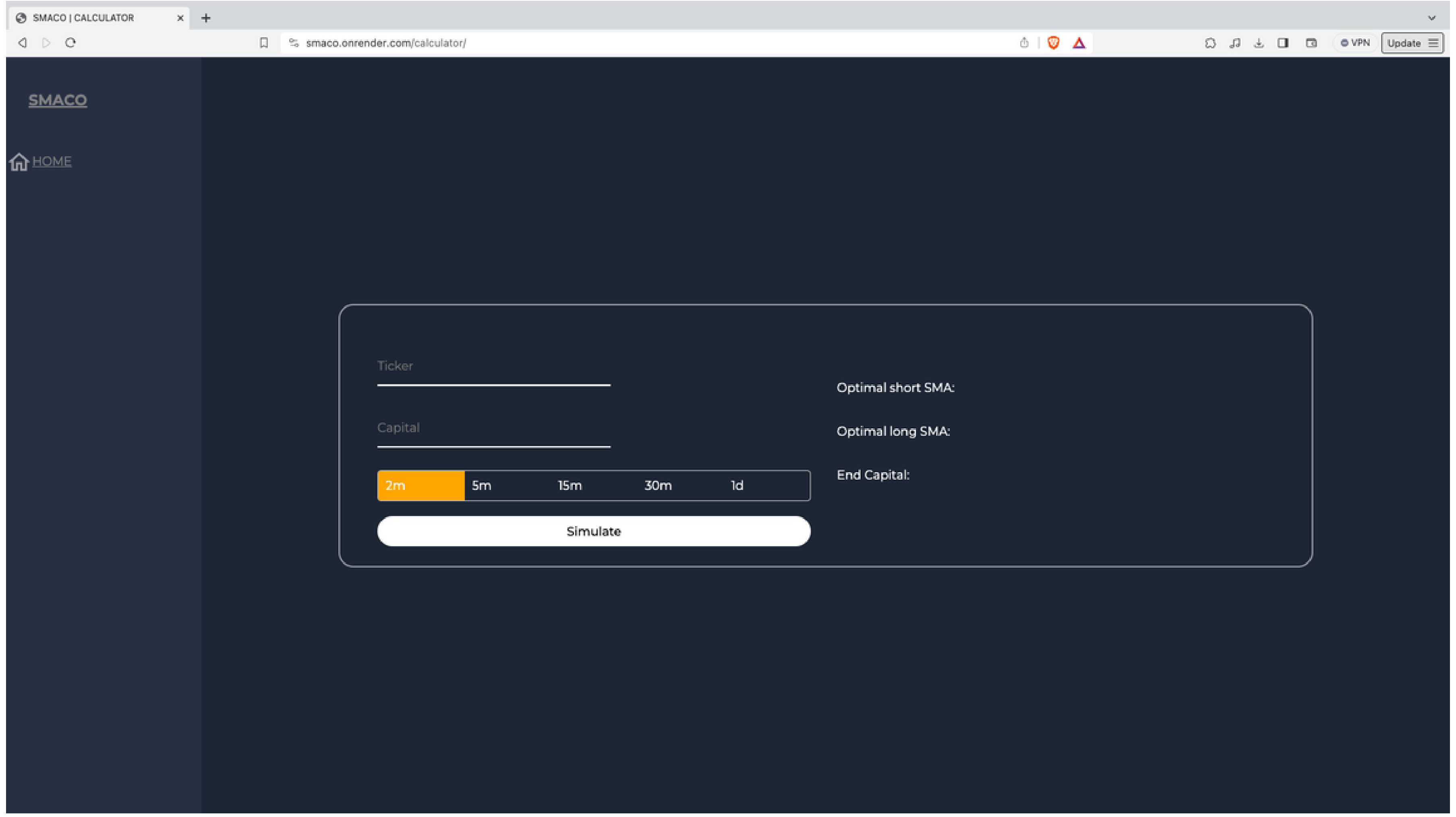


S.M.A.C.O

Calculate the best simple moving average pairs

A passion project by AY23/24
NUS Fintech Society Quant Dept

Copyright ©2024 Frontend designed by HM and project developed by AY23/24 NUS Fintech Society Quant Dept SMACO Team



Ticker

Capital

Optimal short SMA:

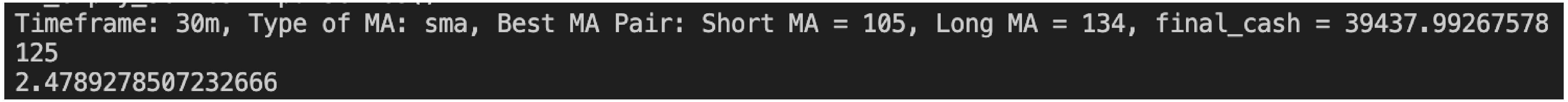
Optimal long SMA:

End Capital:

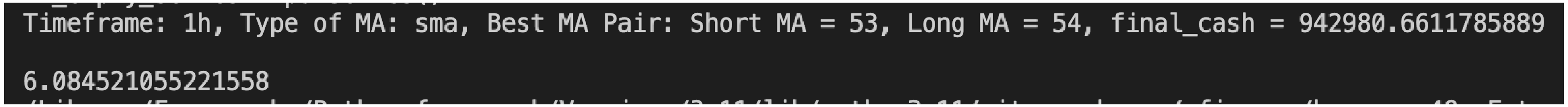
REFLECTIONS

- This strategy can only work on strongly trending assets.
- In certain cases, it is more profitable to buy and hold the stock.
- At smaller timeframes, the frequency of crossovers outweight the profits.

Stock: NVDA, Starting Capital: $100,000



```
Timeframe: 30m, Type of MA: sma, Best MA Pair: Short MA = 105, Long MA = 134, final_cash = 39437.99267578
125
2.4789278507232666
```



```
Timeframe: 1h, Type of MA: sma, Best MA Pair: Short MA = 53, Long MA = 54, final_cash = 942980.6611785889

6.084521055221558
```

- Our team explored live development with both Metatrader5 and Interactive brokers.
  - Metatrader5 had changed its syntax since our last contact with it and redeveloping the application was outside the time contraint we had, bounded by exams.
  - Interactive brokers utilized python syntax and was more user-friendly, however we did not have enough time to explore its utility in depth.

CONCLUSION

- **Increased adaptability**: The best moving average pair based on performance can be found using our code, which may or may not be the traditional pair values (e.g. the Golden and Death cross pair).

- **Increased efficiency**: Used parallelisation to speed up testing of the various MA pairs

- **Utility**: NUS FinTech Society members can reference the information from the algorithm to derive / support investment decisions (NOT FINANCIAL ADVICE, DO YOUR OWN RESEARCH)

- **Documentation**: Thorough documentation for internal understanding has been done, and this allows future readers the design considerations and technical analysis assumptions involved in the design.

- **Future plans**: To incorporate the algorithm into an automated trading strategy. (Tried on MT5 but need to pay unfortunately)

THANK YOU