Sharpe Ratio Optimization via Monte Carlo Simulation

DATA 604 Simulation and Modeling Techniques

Mael Illien ~ Summer 2020

Introduction

Problem and Significance

This goal of this project is to optimize the Sharpe ratio of a portfolio via simulation.

The value is found in determining an allocation of stocks in a portfolio that maximizes risk adjusted returns and answering questions like:

Are a portfolio’s returns the result of a good investment or the result of excess risk?

Since this simulation is backwards looking, it is important to note that the optimal allocation for the previous does not guarantee consistent results if applied in the current year.

Introduction

Sharpe Ratio

- Rp = Return of portfolio
- Rf = Risk free rate (assumed to be 0)
- σ_p = Standard deviation of portfolio returns

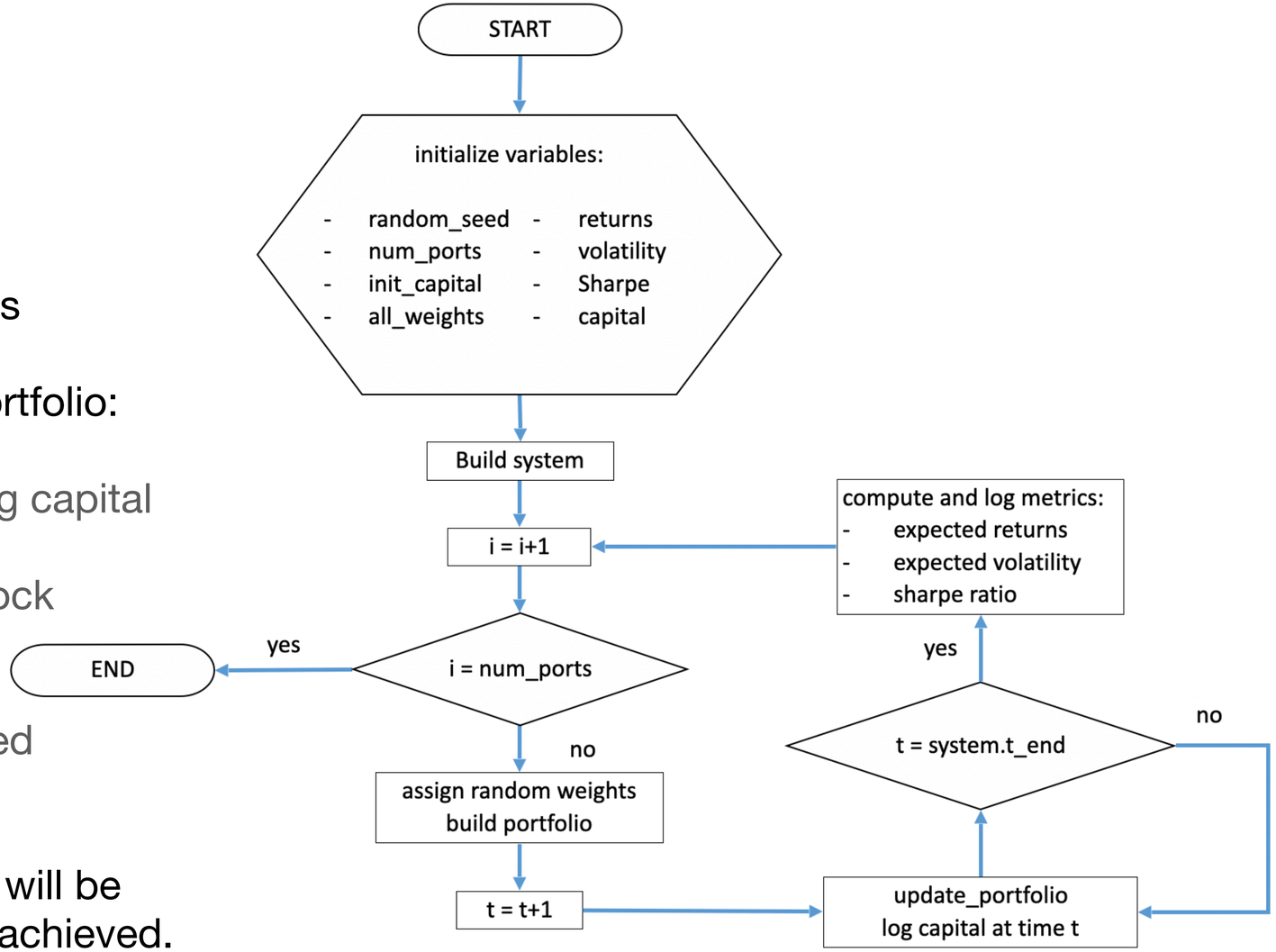
$$SR = \frac{R_p - R_f}{\sigma_p}$$

Methodology

Flowchart

- Initialize simulation parameters
- Initialize arrays to store the simulation results
- Build System object
- Simulate num_ports portfolios. For every portfolio:
  - Assign random initial weights
  - Build Portfolio object initialized to starting capital
  - For every time step t of System:
    - Update Portfolio holdings via daily stock returns
    - Record capital
  - Compute and record metrics for simulated portfolio

Once the simulation is run, the 'best portfolio' will be selected based on the maximum sharpe ratio achieved.



Data

price_date	V	BAC	NKE	AAPL	DLTR
2019-01-02	131.6611	24.2500	73.0811	154.7950	91.20
2019-01-03	126.9165	23.8614	71.7884	139.3763	90.76
2019-01-04	132.3842	24.8524	73.6633	145.3261	92.89
2019-01-07	134.7713	24.8330	74.7192	145.0027	97.96
2019-01-08	135.5043	24.7844	75.7158	147.7669	98.60
...	...	...	...	...	...
2019-12-24	186.9857	34.9967	99.8629	282.8313	93.23
2019-12-26	188.5707	35.2948	100.4314	288.4428	92.08
2019-12-27	188.8000	35.1259	101.2890	288.3333	92.84
2019-12-30	187.2448	34.9271	100.5211	290.0446	93.49
2019-12-31	187.3146	34.9967	101.0297	292.1638	94.05

252 rows × 5 columns

price_date	V	BAC	NKE	AAPL	DLTR
2019-01-02	NaN	NaN	NaN	NaN	NaN
2019-01-03	-0.036036	-0.016025	-0.017689	-0.099607	-0.004825
2019-01-04	0.043081	0.041532	0.026117	0.042689	0.023468
2019-01-07	0.018032	-0.000781	0.014334	-0.002225	0.054581
2019-01-08	0.005439	-0.001957	0.013338	0.019063	0.006533
...	...	...	...	...	...
2019-12-24	0.002619	0.001422	0.000999	0.000951	0.008546
2019-12-26	0.008477	0.008518	0.005693	0.019840	-0.012335
2019-12-27	0.001216	-0.004785	0.008539	-0.000380	0.008254
2019-12-30	-0.008237	-0.005660	-0.007581	0.005935	0.007001
2019-12-31	0.000373	0.001993	0.005060	0.007306	0.005990

252 rows × 5 columns

CSV → **Daily stock prices** → **Daily returns**

Simulation Code

Objects

```
class System():
    def __init__(self, stock_returns, init_capital):
        self.t0 = 0
        self.t_end = len(stock_returns)
        self.stock_returns = stock_returns
        self.stock_names = list(self.stock_returns.columns)
        self.init_capital = init_capital
```

```
class Portfolio():
    def __init__(self, stock_names, init_weights, capital):
        self.init_weights = init_weights
        self.capital = capital
        self.holdings = pd.DataFrame([self.init_weights], columns=stock_names) * capital

    def update_holdings(self, daily_returns):
        "Multiply latest holdings by daily_returns at system.t"
        updated_holdings = np.multiply(self.latest_holdings(), [ret+1 for ret in daily_returns])
        self.holdings = self.holdings.append(updated_holdings)

    def latest_holdings(self):
        return port.holdings.iloc[[-1]]

    def calculate_port_value(self):
        return sum(self.holdings.iloc[-1])
```

Simulation Code

Initialization

```
# initialize
random.seed(604)
num_ports = 500
init_capital = 1e6
all_init_weights = np.zeros((num_ports, len(stocks.columns)))
returns = np.zeros(num_ports)
volatility = np.zeros(num_ports)
sharpe = np.zeros(num_ports)

# Build system
system = System(stock_returns, init_capital)

capital = np.zeros((system.t_end, num_ports))
```

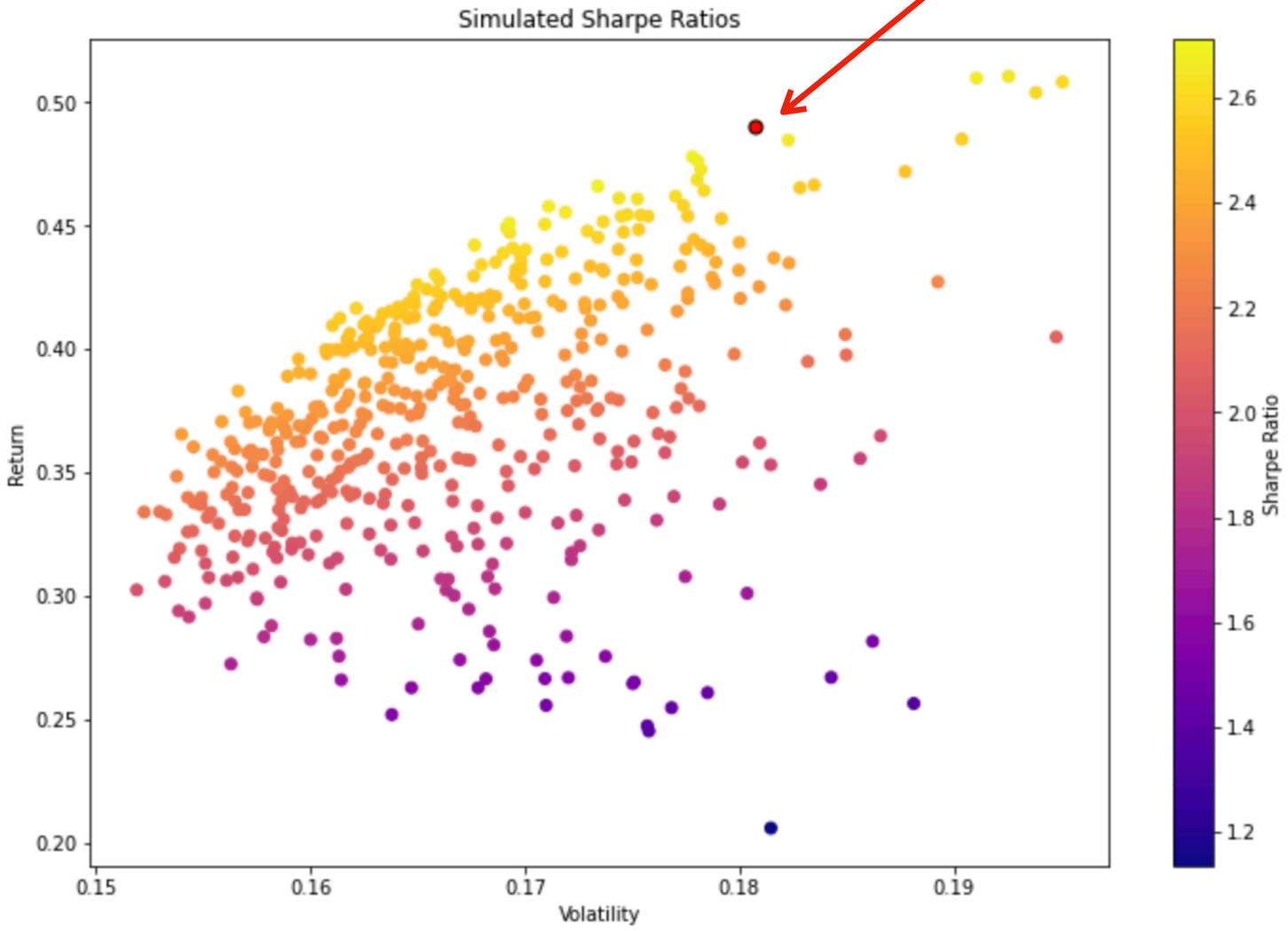
Simulation

```
# Simulate Portfolio
for i in range(num_ports):

    # Assign random weights normalized to sum to 1
    weights = np.array(np.random.random(len(system.stock_names)))
    init_weights = weights/np.sum(weights)
    all_init_weights[i,:] = init_weights

    # Build portfolio and initialize series with the original capital
    port = Portfolio(system.stock_names, init_weights, system.init_capital)

    # Initialize capital series with the original capital
    capital[0,i] = port.calculate_port_value()

    # Update portfolio holdings daily and record capital at time t
    for t in range(system.t0+1, system.t_end):

        daily_returns = system.stock_returns.iloc[t].values.tolist()
        port.update_holdings(daily_returns)
        capital[t,i] = port.calculate_port_value()

    # Compute and record metrics

    # Record expected average returns (annualized)
    ret = pd.Series(capital[:,i]).pct_change(1)
    returns[i] = ret.mean() * 252

    # Record expected volatility (annualized)
    volatility[i] = ret.std() * (252**0.5)

    # Record Sharpe Ratio (annualized)
    sharpe[i] = returns[i]/volatility[i]
```

Results & Findings

Optimized Portfolio

```
Best Portofolio: 401
Max Sharpe Ratio achieved: 2.713
Stocks: ['V', 'BAC', 'NKE', 'AAPL', 'DLTR']
Inital Weight Allocation: [0.311, 0.202, 0.05, 0.408, 0.03]
Returns: 0.49
Volatility: 0.181
Final Portfolio Value: $1,602,591.85
```



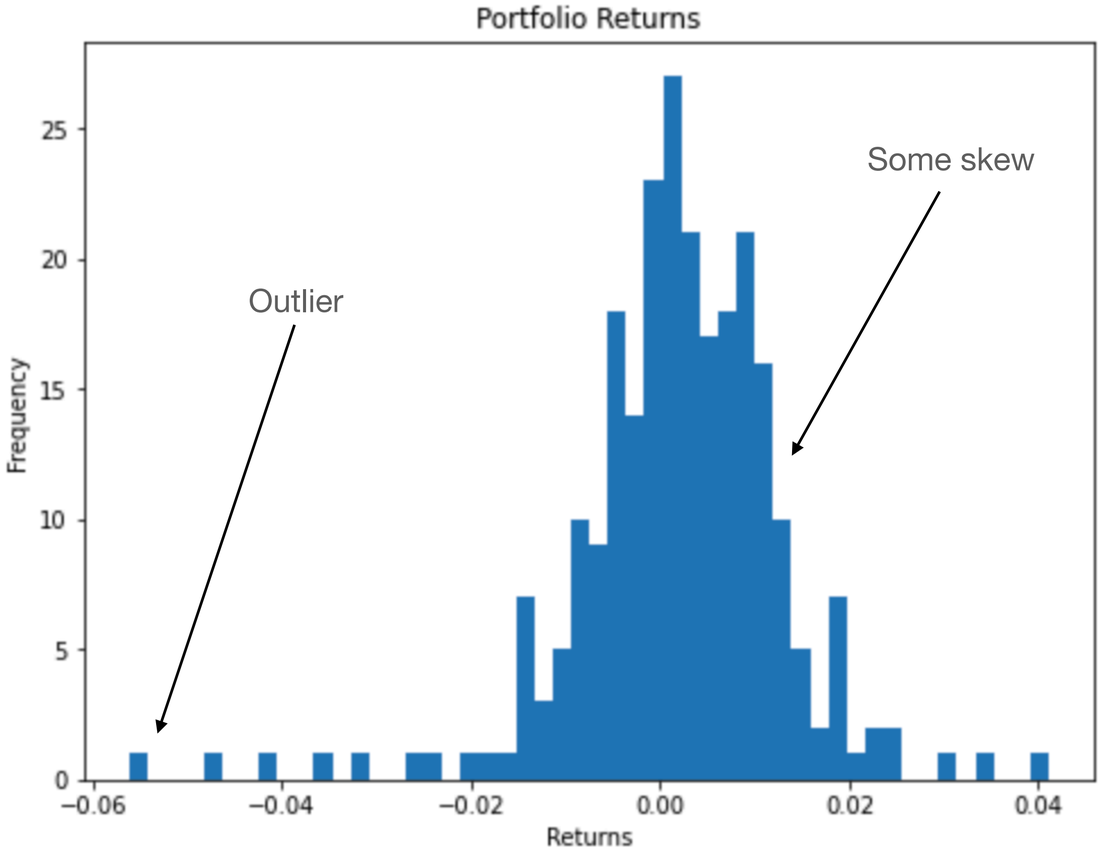
- SR > 1: acceptable
- SR > 2: very good
- SR > 3: excellent

- Note that there are no negative returns regardless of the allocation during the period under study
- 2019 stocks: V, AAPL, BAC, DLTR, NKE

Results & Findings

Verification

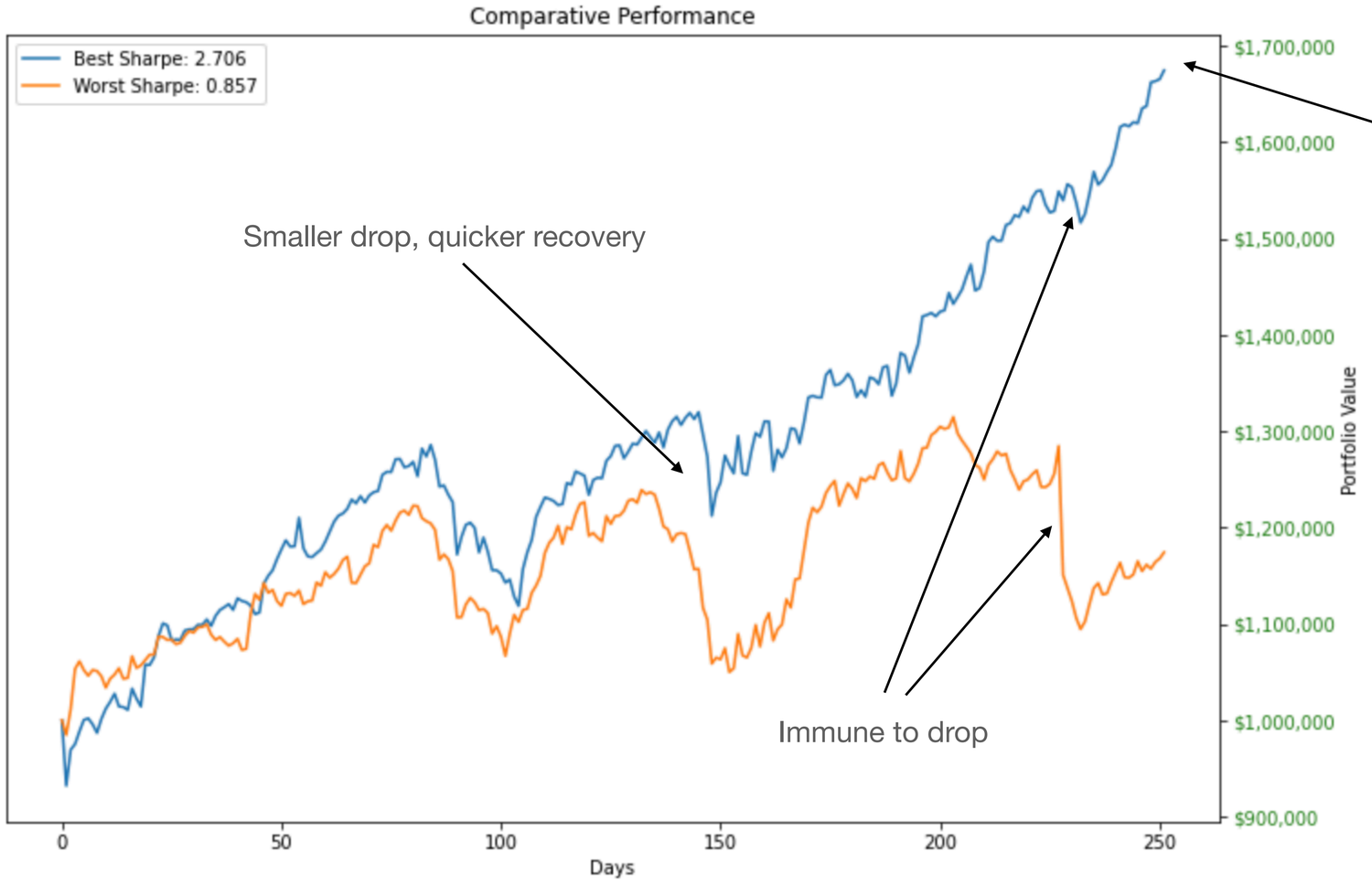
- SR assumes normality of returns
- Shapiro-Wilk Normality test
  - p-value: 0.00042
- Reject null hypothesis, and accept alternate that the returns do not come from the normal distribution
- Caution



Best portfolio returns

Results & Findings

Verification



Conclusion

Best Allocation for 2019

```
Best Portofolio: 401
Max Sharpe Ratio achieved: 2.713
Stocks: ['V', 'BAC', 'NKE', 'AAPL', 'DLTR']
Inital Weight Allocation: [0.311, 0.202, 0.05, 0.408, 0.03]
Returns: 0.49
Volatility: 0.181
Final Portfolio Value: $1,602,591.85
```

While this allocation returned the highest return and shape ratio for the year 2019, there is no guarantee that using this allocation for the year 2020 would yield the same. This could be verified further by backtesting the portfolio performance for the year 2020.

Further Study

Additionally, another simulation could be run where the "buy and hold" strategy would be developed to include monthly rebalancing (20 trading days) in order to keep the same initial weights.

Finally, further exploration would include the use of built-in optimizing functions from SimPy.

Sharpe Ratio Optimization via Monte Carlo Simulation

DATA 604 Simulation and Modeling Techniques

Mael Illien

Thank You