PERFORMANCE COMPARISON BETWEEN METAHEURISTIC BASED METHODS FOR THE PORTFOLIO OPTIMIZATION PROBLEM

Khurram Ayubi Butt

Department of Computer Science

Submitted in partial fulfillment

of the requirements of:

COSC 4F90 - Honors Bachelor of Science Thesis

Faculty of Mathematics and Science, Brock University

St. Catharines, Ontario

Dedication

This thesis is dedicated to my mother and father who
have always supported me and encouraged me
to follow my passion.

PERFORMANCE COMPARISON BETWEEN METAHEURISTIC BASED

METHODS FOR THE PORTFOLIO OPTIMIZATION PROBLEM

Abstract

by Khurram Ayubi Butt,
Brock University
April 2020

Portfolio management is one of the most studied problems in finance and refers to creating an optimal well diversified portfolio of assets that have multiple constraints on them depending on the organization's financial goal. Investors seek to find an optimal combination of their assets to maximize their reward-to-variability ratio. Construction of such a portfolio is a high dimensional constrained optimization problem. This thesis discusses the use of a genetic algorithm (GA) and particle swarm optimization (PSO) to solve the portfolio optimization problem and compare the performance between them. The results indicate that the GA is superior in comparison to the PSO and an effective tool for solving this optimization problem.

ACKNOWLEDGMENT

I wish to express my deepest gratitude to my supervisor, Professor Beatrice M. Ombuki-Berman for her guidance, time and help throughout the project and to allow me this opportunity to solidify my interest in artificial intelligence research. I would like to acknowledge the support of the Computer Science department at Brock University as well as the exceptional level of education they have provided me. I would like to thank my parents for their continued support, for without their trust and support I would not be where I am today.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

List of Algorithms

Chapter One

Introduction

Portfolio optimization is the process of finding the best portfolio among a set of all the portfolios being considered. A portfolio essentially represents a distribution of assets. An optimal portfolio is one which gives the best distribution of these assets according to some objective, which is typically geared towards maximizing factors such as financial gain and at the same time minimizing factors such as loss, costs and financial risks [6].

Selection of an optimal portfolio is a central problem in many financial decisions. In his 1952 essay, Henry Markwitz introduced the modern portfolio theory (MPT) based on his work on mean-variance portfolios [18]. MPT states that any rational investor will want to maximize his profits for a given level of risk or will want to minimize the risk for a given level of expected return [1]. This concept leads to a set of efficient portfolios whose risk-expected return relation can be visualized using a graph, called efficient frontier, in which the portfolios on the curve minimize the risk for a given level of expected return and vice versa. According to the MPT, the fundamental principle of financial investments is diversification of the assets that make up the portfolio [7]. Construction of an optimal portfolio is a high dimensional constrained optimization problem whereby investors want to achieve an optimal combination of the investments in order to achieve the best reward-risk ratio [7]. Among the many methodologies present, maximizing the Sharpe-Ratio is one of the most popular [7]. Sharpe-Ratio is a measure of the performance of an investment after adjusting its risk,

against a risk free asset. It represents the additional expected return an investor receives after accounting for the level of risk the investment faces.

Since its introduction, MPT has been widely accepted as a practical tool for portfolio optimization. However, in some cases, the characteristics of the problem such as its size, computational time and real-world constraints etc. make analytical methods less suitable for solving a large instance of the mean-variance model [11]. In these cases, we have to resort to heuristic based methods which can find high-quality solutions in a reasonable amount of time. Genetic algorithms (GA) and Particle Swarm optimization (PSO) are two meta-heuristic algorithms that have been applied to a variety of optimization problems. PSO and GA are both population based stochastic models which work to find the fittest individual (in GA) or particle (in PSO) by iteratively improving candidate solutions with regards to some measure of quality. Many outstanding studies have been published in the recent years making use of various heuristic methods to find good quality solutions to the asset allocation problem. Many meta-heuristic techniques such as GA, simulated annealing and tabu search have been employed to find a cardinality constrained optimal portfolio [11]. Hybrid techniques such as PSO with Artificial Neural Networks (ANN) have been used for the portfolio selection problem and have demonstrated effectiveness and superiority in forecasting performance compared to traditional econometric methodologies [11]. Fuzzy Analytic Hierarchy Process has been combined with the portfolio selection problem to model uncertain environments [11]. Ant colony optimization has also been shown as an effective solver for the portfolio selection problem. However, these approaches have their shortcomings [11]. Fuzzy analytic methodologies lack learning ability, ANN are prone to overfitting and can be trapped in local minima [11]. As the size of the problem instance increases, these approaches face an increased load and time to converge and find adequate solutions.

1.1 Objectives

There have been several studies on PSO and GA for the portfolio selection problem. The main purpose of this thesis is to employ a GA and PSO algorithm for the portfolio selection problem and compare the performance between the two.

The fulfil the proposed objectives, this work aims to achieve the following goals:

- Employ a PSO algorithm to find a high-quality solution for portfolio optimization problem
- Test the parameters for the PSO algorithm to fine tune and find the best parameters
- Employ a GA to find a high quality solution for the portfolio optimization problem
- Test the parameters for the GA to fine tune and find the best parameters
- Compare the performance of the GA and PSO algorithm to determine if one performed significantly better than the other in terms of finding the best solution

1.2 Thesis organization

The remainder of this thesis is organized as follows:

Chapter 2 contains all background information relevant to the study. This includes an overview of some models for portfolio optimization along with algorithms such as genetic algorithm and particle swarm optimization.

Chapter 3 describes the experimental setup used in this study. The parameters tested and optimized for the GA and PSO as well as the parameters compared between the two are listed, along with the data which was used to run the experiments. A brief description of the performance measure used is given. Lastly the statistical methods of analysis are listed and a brief description given for each.

Chapter 4 presents the results of all experiments performed for optimizing the GA and PSO, as well as the results for the experiments performed to compare the performance of tha GA versus the PSO.

Chapter 5 describes the conclusions drawn and guidance for future work.

Chapter Two

Background

The aim of portfolio optimization is to find a combination of assets that is optimal with respect to some performance measure. Often times this can be a two step process whereby investors first decide which type of assets to invest in and then decide how the money should be spread across all these chosen assets to maximize financial gain and minimize loss and costs. For the purposes of this thesis, we will only be focused on the second part to find an optimal allocation of assets. This section briefly discusses the various models used for portfolio optimization and then gives a brief overview of the two algorithms used in this thesis.

2.1 Models for portfolio optimization

Since diversification is key in many financial investments, investors look to diversify their investments into many different assets [11]. Portfolio diversification minimizes an investors risk while at the same time maximizing their expected return. Therefore, it can be considered a multi objective optimization problem [11]. Next, the thesis will discuss some of the models that have been developed over the years to solve this optimization problem.

2.1.1 Markowitz mean-variance model

In Markowitz mean-variance model, one function is selected to be optimized while the other objective functions are defined as constrained conditions. The security selection for the risky portfolio construction is set as the single function to be optimized and the mean return is defined as one of the constraints [11]. This model is described as:

$$MIN \sum_{i=1}^{N} \sum_{j=1}^{N} w_i w_j \sigma_{ij} \tag{2.1}$$

$$\text{Subject to } \sum_{i=1}^{N} w_i r_i = R^*, \tag{2.2}$$

$$\sum_{i=1}^{N} w_i = 1 \tag{2.3}$$

$$0 \leq w_i \leq 1, \text{ i} = \{1, .. , \text{N}\} \tag{2.4}$$

Where, N is the number of total assets in the portfolio, w_i is the weight of asset i, w_j is the weight of asset j, r_i is the. return of asset i, σ_{ij} is the covariance between the returns of asset i and asset j and R* is the desire mean return [11].

2.1.2 Efficient frontier

In the Efficient frontier model, one objective function is constructed for optimization by weighing the multiple objective functions. A new risk aversion parameter λ is introduced . With this new parameter, the model is described as a single objective function:

$$MIN\lambda[\sum_{i=1}^{N} \sum_{j=1}^{N} w_i w_j \sigma_{ij}] - (1-\lambda)[\sum_{i=1}^{N} w_i r_i] \tag{2.5}$$

$$\text{Subject to } \sum_{i=1}^{N} w_i = 1 \tag{2.6}$$

$$0 \leq w_i \leq 1, \text{ i} = \{1, .. , \text{N}\} \tag{2.7}$$

Here, when $\lambda = 1$, the risk of the portfolio is minimized regardless of the returns. In contrast, when $\lambda = 0$, the expected mean return is maximized regardless of the risk. Thus, as λ approaches 0, the sensitivity to risk is decreased whereas when λ approaches one, the sensitivity to risk is increased. [11]

Different values of λ will produce different objective function values. Plotting the intersection of mean returns and variance (risk), we can trace a curve which connects them and this curve is referred to as the efficient frontier. Each point on the curve represents an optimum i.e. a portfolio with the highest mean return for a given level of risk and vice versa [11].

2.1.3 Sharpe Ratio model

The Sharpe Ratio model, similar to the efficient frontier model has one objective function for optimization. Insead of looking at the mean-variance efficient frontier, we seek to maximize the Sharpe Ratio. The sharpe ratio is a measure of the risk-adjusted return of a portfolio and is often used to determine the quality of a portfolio. The sharpe ratio takes into account both the expected mean return and the variance of the portfolio and is given by the following formula:

$$SharpeRatio = \frac{R_p - R_f}{StdDev(p)} \tag{2.8}$$

Where R_P is the mean return of the portfolio, R_f is the risk free security usually the interest rate from the treasuary bill and StdDev(p) is the standard deviation of the returns of the portfolio p.

By adjusting the weights of various assets in the portfolio p, we can maximize the Sharpe ratio measure which in turn balances the tradeoff between maximizing expected returns and minimizing risk [11]. For the purposes of this thesis, the Sharpe Ratio model was used to find an optimal distribution of weights for the various stocks in a portfolio.

2.2 Particle Swarm Optimization

Particle swarm optimization (PSO) is a stochastic population based metaheuristic algorithm, inspired from the behaviour of fish schooling and flocking of birds. It was first developed by Kennedy and Eberhert in 1995, as a method to simulate social behaviour based on bird flocking and fish schooling [3]. Since then, it has been realized as an optimization algorithm which has been used in a variety of applications e.g., in various aspects of geotechnical engineering such as slope stability analysis, tunnelling and underground space design, rock and soil mechanics [14], in power system optimizations [9], in network security defences [17] and many more.

The term "particle" in PSO refers to a simple entity which represents a candidate solution and these particles are moved around the n-dimensional search space. Each particle has a position (in n-dimensions), velocity and is responsible for keeping track of its best position (solution) found so far. The movement of each particle is governed by its velocity, current position, their personal best as well as the global best i.e. the particle which has the best fitness value. The fitness value of particles is measured over some objective function value which represents how good or bad a particle is. The term "swarm" refers to the population of said particles, where each particle is a candidate solution. Since the movement is dependent on the personal best as well as the global best, as new and better positions are found, they guide the movement of the swarm. Typically, the particle's position represents the candidate solution. Thus, by moving the particles through the n-search space, the PSO algorithm attempts to iteratively improve the solution represented by each particle with an end goal of convergence or finding a high quality solution.

The following three factors primarily influence the movement of particles in the search space:

- The velocity of the particle
- The global best position

- The personal best position of the particle found so far

The effect of these 3 factors on the position update is controlled by a constant for each factor. The inertia, w, controls the impact of the velocity of a particle in finding a new position. The inertia in physics is defined as the resistance of an object to a change in its state of motion or rest. In PSO, the inertia allows to apply a portion of the previous velocity to the updated velocity and thus the particle will attempt to continue in the same direction it was previously heading in. This can help overcome local minima. The cognitive component c_1, controls how much the personal best position affects the movement of the particle. Lastly, there is a social component c_2 which controls the impact of the global best on the movement of the particle. There are also the stochastic components, r_1 and r_2, which are uniform random vectors with component values between [0,1]. These add a random element to enhance exploration and can help prevent premature convergence. The equations for the velocity and position update for a particle are given in equations (2.9)-(2.10). The values for the inertia weight, social and cognitive components need to be established empirically through testing as they can differ for each problem. However, there are some values which have been shown to provide high quality results for a variety of problems e.g., van den Bergh has shown that the parameters $w = 0.729844$ and $c_1 = c_2 = 1.496180$ perform empirically well and can lead to convergence [5].

The velocity update for each particle is done using the following equation:

$$v_i(t+1) = \omega v_i(t) + c_1 r_1(y(t) - x_i(t)) + c_2 r_2(\hat{y}(t) - x_i(t)) \tag{2.9}$$

Where, $v_i(t+1)$ is the updated velocity of particle i at the next time step, v_i is the velocity of particle i at time t, $y(t)$ is the personal best solution of particle i at time t, $\hat{y}$ is the global best solution at time t and x_i is the current position of particle i.

The position update then follows:

$$x_i(t+1) = x_i + x_i(t+1) \tag{2.10}$$

Where, $x_i(t+1)$ is the updated position of particle i at the next time step, x_i is the current position of particle i and $v_i(t+1)$ is the updated velocity for particle i calculated using equation (2.9).

Velocity clamping is often employed in PSO to provide a limit on the component sizes for the velocity vector, ensuring that the velocity vector values do not explode. Velocity clamping is defined in the equations below:

$$v_{ij}(t+1) = \begin{cases} -v_{max}, & \text{if } v_{ij}(t+1) < -v_{max} \\ v_{max}, & \text{if } v_{ij}(t+1) > v_{max} \\ v_{ij}(t+1), & \text{otherwise} \end{cases} \tag{2.11}$$

Where v_{max} is the largest allowable step size in any dimension.

The PSO algorithm [15] is shown in Algorithm 1.

Algorithm 1: Particle Swarm Optimization

Result: Optimal Solution

Initialize particles randomly;

while *current iteration < max iterations* **do**

  for *each particle in swarm* **do**

    Evaluate fitness;

    Update personal best position;

  end

  Update neighborhood best particle;

  for *each particle in swarm* **do**

    Calculate new velocity (Eq (2.9));

    Update particle position (Eq (2.10));

  end

end

2.3 Genetic Algorithm

Similar to PSO, Genetic algorithms (GA) are stochastic population based meta-heursitic algorithms. The natural inspiration for GA comes from Darwin's theory of evolution and natural selection, whereby the concept of "survival of the fittest" is ensued. GA's were first introduced by John Holland in 1960 [4]. Holland and his students and colleagues worked to develop genetic algorithms in the University of Michingan in 1960 and 1970's [4]. Since then, GA's have improved drastically with the introduction of new genetic operators and have been applied to a variety of problems such as Cryptanalysis of various cipher problems [8], computation creativity [21], feature selection for artificial neural networks [13] and many more.

Genetic algorithms generate high quality solutions to optimization problems by utilizing biologically inspired genetic operators such as selection, crossover and mutations. Similar to PSO, in GA there is a population of candidate solutions (referred to as phenotype or individuals) where each individual has a set of characteristics (chromosome or genotype) which define that individual and it is this genetic makeup of the individuals that typically represents a complete (or partial) solution to the problem at hand. Through an iterative process, the GA seeks to alter this genotype of each individual in hopes of reaching convergence or a high quality solution.

The GA starts by randomly creating a fixed size population of new individuals which represent candidate solutions. Then, through the iterative process of applying the genetic operators, these individuals are improved. At each generation, parents are selected using a selection scheme. The selected parents are then mated using a *crossover* technique to produce offspring. The crossover is a method to combine two parents, swap genetic material between them to create one or more offspring. The offspring are then *mutated* to introduce some variation from their parents. The mutation operator allows the offspring to evolve in new directions so as to not fix the population traits and bring some variety into the gene

pool. These individuals replace the population from the previous generation to create a new population. The process is repeated until convergence is reached or a high quality solution is found. The fitness value of each individual is determined through an objective function value which shows how good or bad the solution represented by an individual is. The mutation and crossover both have a rate which determines the probability with which an individual will undergo crossover and mutation. Algorithm 2 presents the complete genetic algorithm.

One additional operator used in GA is called *elitism*. Elitism refers to allowing a certain number of the fittest individuals in a generation to get carried over to the next generation without alteration. This allows to ensure that the fittest chromosomes from the population are retained and not lost during the crossover and mutation phase.

Algorithm 2: Genetic Algorithm

Result: Optimal Solution

Initialize population of candidate solutions;

while *current generation* $<$ *max generations* **do**

    Evaluate fitness of individuals;

    Select individuals from population for crossover;

    Apply crossover on the selected parents;

    Apply mutation on the offsprings produced;

    Replace individuals with the elites;

    Replace population with the new generated population;

end

Chapter Three

Experimental Setup

This section goes over all the necessary information regarding the experiments performed, the performance measured used and the data used for the experiments.

3.1 Performance Measure - Sharpe Ratio

The performance measure derived from the fitness function is critical to GA and PSO. Each particle in PSO and each chromosome in GA has a fitness value. This fitness value allows comparison between the candidate solutions to rank them and see which solution is the best. In this thesis, Sharpe Ratio was used as a single objective function. Sharpe ratio is defined as ,

$$f_i = SharpeRatio = \frac{\sum_{i=1}^{N} w_i r_i - R_f}{\sum_{j=1}^{N} w_i w_j \sigma_{ij}} \tag{3.1}$$

Where f_i is the fitness value of particle i in PSO and chromosome i in the GA. The objective function seeked to maximize the Sharpe Ratio, thus higher the fitness value, the better the solution. R_f which denotes the risk free rate was set to 0.25% as obtained from official U.S. government website for the U.S. department of treasury [20].

3.2 Problem Definition

For the purpose of this thesis, only the restricted portfolio optimization problem was considered and Sharpe Ratio was used as the single objective maximization function. This can be defined as:

$$\text{Max Sharpe Ratio} = MAX \frac{\sum_{i=1}^{N} w_i r_i - R_f}{\sum_{j=1}^{N} w_i w_j \sigma_{ij}} \tag{3.2}$$

$$\text{Subject to } \sum_{i=1}^{N} w_i = 1, \tag{3.3}$$

$$0 \leq w_i \leq 1, \text{ i} = \{1, .. , \text{N}\} \tag{3.4}$$

The reason it is called restricted is because short selling of assets is not allowed. In a restricted problem instance, the investors are allowed to sell an asset they do not own, under the condition that they will buy the asset later at hopefully a lower price point, thus allowing asset weights in a portfolio to be negative [11]. Comparatively, as stated above, in a restricted instance, the weights all have to satisfy constraint (3.4).

3.3 PSO

This section describes the setup of the PSO algorithm used in this thesis. It also goes over the experimental setup for PSO, listing the parameters which were tested and fine tuned.

3.3.1 PSO Setup

In the PSO implemented, the position vector for each particle represented a candidate solution. The position vector was N-spaced, where N is the number of assets being considered in the portfolio. To initialize all particles, their velocities were set to 0 and the position vectors were randomly initialized to contain values between [0,1] which sum up to 1, to satisfy constraints (3.3)-(3.4). Velocity clamping was not used in our implementation.

3.3.2 PSO Parameters

The parameters tested to optimize the PSO algorithm were the cognitive constant c_1, social constant c_2 and the inertia weight constant w.

3.4 GA

This section describes the setup of the genetic algorithm used in this thesis. It also goes over the experimental setup for GA, listing the parameters which were tested and fine tuned.

3.4.1 GA Setup

This section goes over the genetic operators and the chromosome representation used for the GA in this thesis.

Chromosome Representation

For the portfolio optimization problem, the chromosome represents the weights of the different assets/stocks in the portfolio. If the portfolio being optimized is made up of N assets then the chromosome is represented by an array of N positive real number values, each of which represents a percentage of wealth invested in the asset. The weight of all the assets sums up to 1 to satisfy (3.3)-(3.4). Let c be a chromosome, then:

$$c = \{w_1, w_2, ..., w_n\} \tag{3.5}$$

where each w_i is a gene in the chromosome and represents the percentage of wealth invested in asset i.

Selection

The parents selected for producing offspring are chosen using the k-tournament selection scheme. In tournament selection, k individuals are randomly drawn from the population

and the best among them is chosen. This process is repeated to choose more individuals until the population size is reached. These new individuals act as the parents that will mate to produce offspring. For the purposes of this thesis k=3, unless otherwise stated.

Crossover

The crossover method used is the BLX-α crossover. This crossover introduced in [2], generates offsprings by choosing values between an interval which contains both parents. The upper and lower bound of the interval is calculated using the following equations:

$$UpperBound = Max(w_{i,parentA}, w_{i,parentB}) + \alpha * d_i \tag{3.6}$$

$$LowerBound = Min(w_{i,parentA}, w_{i,parentB}) - \alpha * d_i \tag{3.7}$$

$$d_i = |w_{i,parentA} - w_{i,parentB}| \tag{3.8}$$

Where w_i is the weight of asset i in the respective parent chromosome and α coefficient is a constant between [0,1]. α was set as 0.7 for the purposes of this thesis. This value was chosen because [18] reported success using this value in a GA for portfolio optimization. It is important to note that in order to satisfy the constraints (3.3)-(3.4), the offsprings were normalized.

Mutation

The mutation operator used is the simple displacement mutation. In this mutation, a subsection of the chromosome is taken and inserted into a random place in the chromosome, shifting the other alleles accordingly.

Elitism

At most top 10% of the population's fittest individuals were carried over to the next generation without any alteration. To implement elitism, the top 10% of the populations fittest

individuals were saved before the crossover and mutations. Once the population had gone through the mutation stage, the elites randomly replaced other individuals of the population, hence keeping the total population size the same. The population was then copied over to replace the old population and the process was repeated.

3.4.2 GA Parameters

The parameters tested to optimize the GA were the mutation rate, crossover rate and the population size.

3.5 Tests between GA and PSO

Portfolios of size 8, 16, 24 and 32 were compared between GA and PSO. Matched Wilcoxon test (also called Mann Whitney U-tests) were performed between the highest Sharpe Ratio achieved for the given portfolio using both algorithms. 30 portfolios for each size were tested. The portfolios were generated randomly from the list of stocks presented in Section 3.6.

3.6 Dataset

All the experiments were carried out using historical prices for 2 years (01/01/2010 - 31/12/2012). The adjusted prices for the time period specified, were taken from Yahoo Finance.

The stocks were randomly chosen from the following list: ['TXT', 'AKAM', 'TGT', 'D', 'DTE', 'FDX', 'XOM', 'GD', 'FMC', 'MYL', 'PXD', 'PRU', 'PNR', 'CSX', 'VZ', 'MTB', 'TAP', 'AMG', 'OKE', 'O', 'WDC', 'LEG', 'NSC', 'ADP', 'INTU', 'MCK', 'CLX', 'EXC', 'PWR', 'IBM', 'ETN', 'K', 'BA', 'JPM', 'PH', 'BAC', 'JNJ', 'EA', 'BBBY', 'APH', 'IPG', 'ESS', 'CELG', 'KEY', 'LMT', 'SYK', 'DOV', 'EFX', 'CMS', 'SYY'].

It must be noted that when performing the experiments to tune the parameters for PSO and GA, a fixed porfolio of size 8 was used ['AAPL', 'AMZN', 'NDAQ', 'WEC', 'TGNA',

'IBM', 'JPM', 'MSFT']. For all the other experiments, the stocks were randomly chosen from the list to create portfolios of the given size.

3.7 Statistical Methods

3.7.1 Shapiro Wilk test

The Shapiro Wilk test is a test for normality i.e. it checks whether the sample comes from a population with normal distribution [19]. The test was first introduced by Samuel Shapiro and Martin Wilk in 1965. The hypothesis tested is as follows:

$\boldsymbol{H_0}$: The samples come from a normally distributed population

$\boldsymbol{H_1}$: The samples come from a population that is not normally distributed

On a 95% confidence interval, a p-value of less than 0.05 suggests that we reject the null hypothesis and therefore cannot assume that the underlying population is normally distributed.

3.7.2 Analysis of Variance (ANOVA) and Kruskal-Wallis test

Analysis of variance (ANOVA) is a statistical method to test the equality of three or more population means i.e. 3 or more samples come from a population with equal means [19]. Kruskal-Wallis test is the non-parametric equivalent of the ANOVA test and it tests the equality of three or more population medians i.e. 3 or more samples come from a population with equal medians [19]. In cases where data is not normally distributed, we use the Kruskal-Wallis test. The hypothesis tested by this method are as follows:

$\boldsymbol{H_0}$: There is no significant difference between any of the group means (or medians for Kruskal-Wallis test)

$\boldsymbol{H_1}$: There is a significant difference between at least two groups means (or medians for Kruskal-Wallis test)

On a 95% confidence interval, if the p-value is less than 0.05, we reject the null hypothesis of equal means (or median).

3.7.3 Pairwise t-tests and pairwise wilcoxon rank-sum tests

Since ANOVA and Kruskal-Wallis only tell whether or not there is a significant difference between the groups, they cannot tell which of these groups are significantly different from one another. Thus, we must perform pairwise t-tests and pairwise wilcox tests (the nonparametric equivalent of t-tests), to determine which of the groups have a significant difference between them. Since the Wilcoxon rank-sum test is a nonparametric test, it does not require a normal or any other particular distribution. When doing multiple pairwise tests, we run into the problem of having a high risk of type 1 error - finding a difference by chance alone i.e. finding a difference in groups when a difference does not really exist [19]. Therefore, when pairwise tests are performed, the p values are adjusted using the Benjamini and Hochberg method. This method controls the rate of false discoveries among the rejected null hypothesis [22]. The hypothesis tested by these pairwise methods are as follows:

$\boldsymbol{H_0}$: The two independent samples have values with equal means (or medians for wilcoxon rank-sum test)

$\boldsymbol{H_1}$: The two independent samples have values with means (or medians for wilcoxon rank-sum test) that are not equal.

On a 95% confidence interval, if the p-value is less than 0.05, we reject the null hypothesis of equal means (or median).

Chapter Four

Experimental Results and Discussion

This chapter presents and discusses the results obtained from the experiments performed. The experiments were split into three main sections. The first section and second section, 4.1 and 4.2, go over the tests performed on the PSO algorithm and GA, respectively, to find suitable parameters for these algorithms for the portfolio optimization problem. These two sections go over the parameters tested and the conditions they were tested under. Section 4.3 covers the experiments performed for comparing PSO with GA for the problem at hand and presents an analysis of a typical run for both algorithms when solving this optimization problem.

4.1 Optimizing PSO Parameters

This section covers the performance of PSO with regards to its parameter setup to determine the best set of parameters for the portfolio optimization problem.

4.1.1 Experimental Setup

All experiments were performed with 10 particles in the swarm, 500 iterations and on the 8 stocks listed above. 10 runs were performed for each parameter configuration.

Pairwise Wilcoxon Rank Sum Test P-values for c1 values tested						
c1 values	**0.2**	**0.5**	**0.8**	**1.2**	**1.5**	**1.7**
0.5	0.4500	-	-	-	-	-
0.8	0.4500	0.9487	-	-	-	-
1.2	0.3042	0.5114	0.5262	-	-	-
1.5	0.3530	0.5903	0.7730	0.8368	-	-
1.7	0.0147	0.1679	0.1679	0.3688	0.3056	-
2	0.0058	0.1948	0.1632	0.3688	0.4500	0.6555

Table 4.1 Reported p-values from the wilcoxon rank sum test for optimizing the cognitive component values

4.1.2 Cognitive Component

7 values were tested for the cognitive component {0.2, 0.5, 0.8, 1.2, 1.5, 1.7, 2}. While testing values for the cognitive component, the social component, c_2 was set to 1.7 and ω was set at 0.9. The resulting p-values from the pairwise wilcoxon rank sum test are reported in Table 4.1. The only statistical difference observed was that of between 0.2 and 1,7 and 2. 0.2 performed significantly worse than 1.7 and 2. However, there was no significant difference observed between 1.7 and 2. 1.7 was chosen as the best value since it had smaller standard error and a higher mean compared to the other values (Figure 4.1), albeit the difference was not statistically significant. The author of [16] concluded that out of the 3 parameters being tuned, c_1 is usually the least relevant control parameter. Looking at the boxplot in A.1 and 4.1, it can be seen that the results of the cognitive component values do not deviate much from one another.

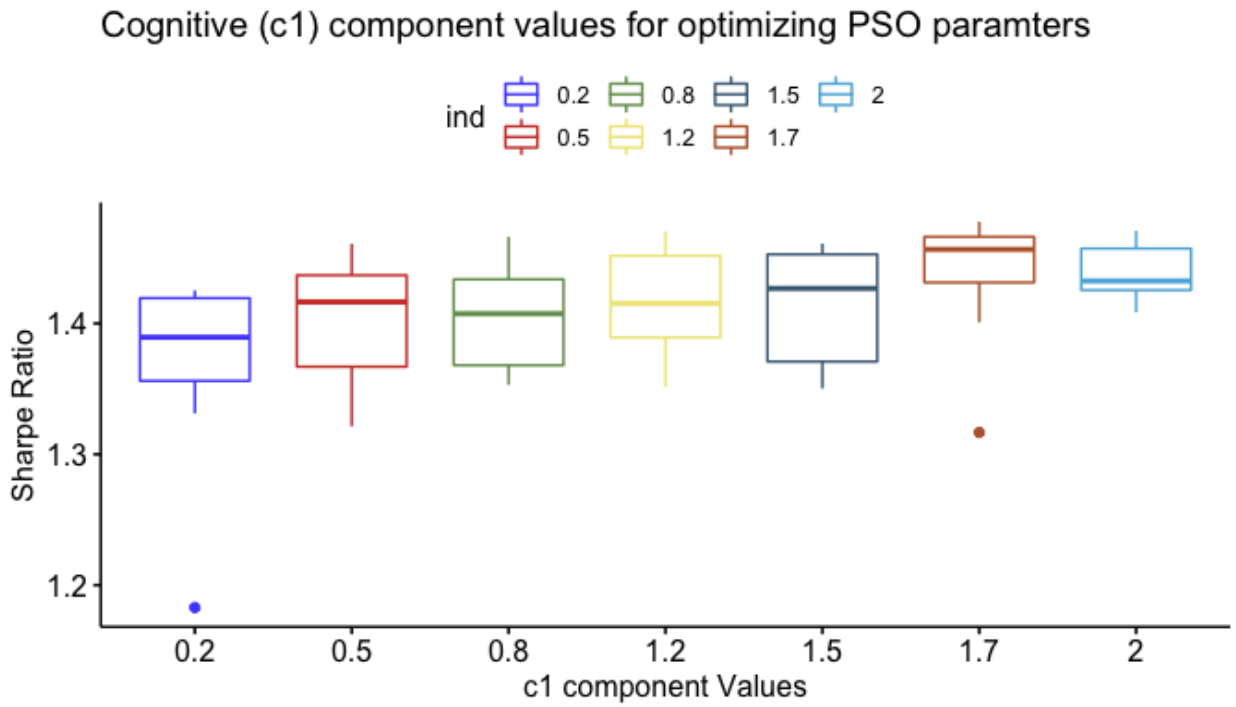

Figure 4.1 Boxplots for cognitive component values against the Sharpe Ratios achieved over the 10 runs.

4.1.3 Social Component

7 values were tested for the social component {0.2, 0.5, 0.8, 1.2, 1.5, 1.7, 2}. While testing values for the social component, the cognitive component, c_1 was set to 1.7 and ω was set at 0.9. The resulting p-values from the pairwise wilcoxon rank sum test are reported in Table 4.2. There was no statistically significant difference between any of the c_2 values tested. However, in Figure A.2 it can be seen that 2 has a smaller standard error and higher mean compared to the other values tested.Therefore, $c_2 = 2$ was chosen as the value for the social component constant.

4.1.4 Momentum

Momentum is the most important control parameter to tune as concluded in [16]. As seen in Figure. 4.3, most of the values introduced significant variation in the resulting Sharpe

Pairwise Wilcoxon Rank Sum Test P-values for c2 values tested						
c2 values	**0.2**	**0.5**	**0.8**	**1.2**	**1.5**	**1.7**
0.5	0.4055	-	-	-	-	-
0.8	0.4055	0.6412	-	-	-	-
1.2	0.4055	0.4055	0.6555	-	-	-
1.5	0.1632	0.1632	0.4055	0.4055	-	-
1.7	0.3984	0.4055	0.7730	0.9962	0.4055	-
2	0.6272	1.0000	0.6272	0.4055	0.1632	0.4055

Table 4.2 Reported p-values from the wilcoxon rank sum test for optimizing the social component values

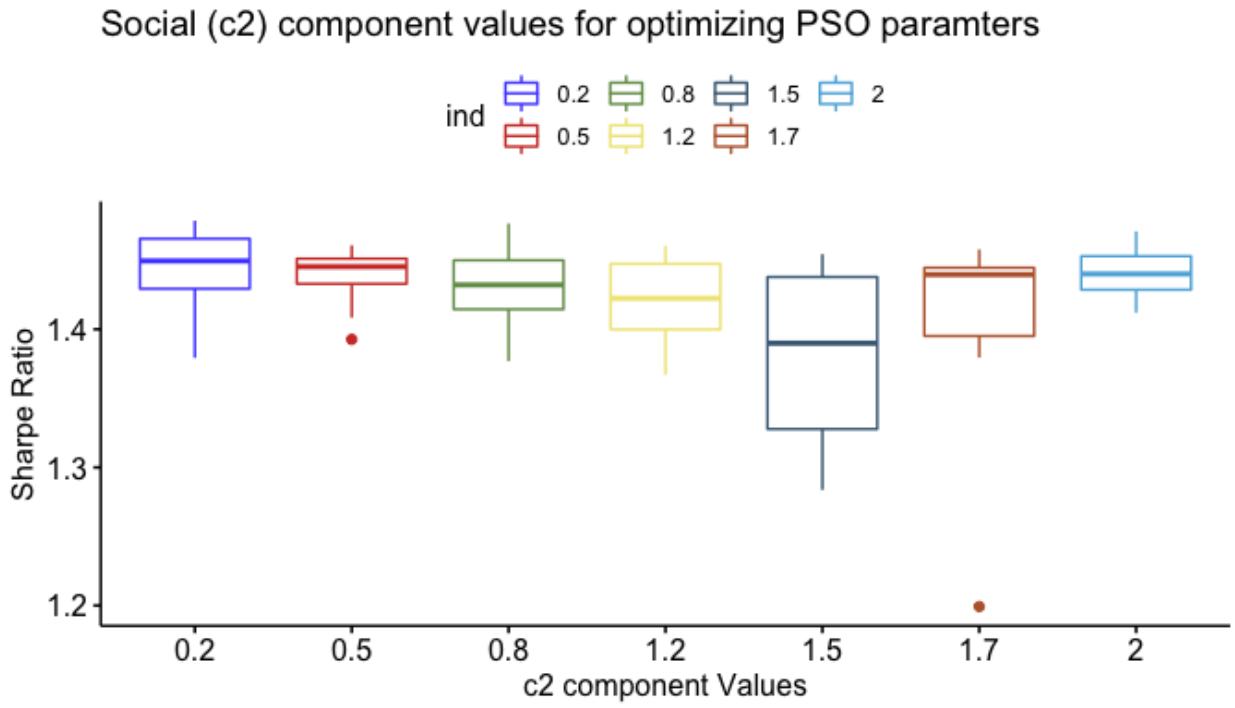

Figure 4.2 Boxplots for social component values against the Sharpe Ratios achieved over the 10 runs.

Pairwise Wilcoxon Rank Sum Test P-values for Momentum values tested										
Momentum Values	**-1**	**-0.8**	**-0.6**	**-0.4**	**-0.2**	**0**	**0.2**	**0.4**	**0.6**	**0.8**
-0.8	0.06959	-	-	-	-	-	-	-	-	-
-0.6	0.00159	0.03205	-	-	-	-	-	-	-	-
-0.4	0.00001	0.00001	0.00159	-	-	-	-	-	-	-
-0.2	0.00001	0.00001	0.02712	0.55974	-	-	-	-	-	-
0	0.00006	0.00001	0.68981	0.00001	0.02282	-	-	-	-	-
0.2	0.00006	0.00001	0.72574	0.00001	0.03675	0.94873	-	-	-	-
0.4	0.00001	0.00001	0.00014	0.20047	0.20047	0.00004	0.00004	-	-	-
0.6	0.00001	0.00001	0.00001	0.00001	0.00001	0.00001	0.00001	0.00001	-	-
0.8	0.00001	0.00001	0.00001	0.03675	0.06959	0.00001	0.00001	0.52564	0.00064	-
1	0.00001	0.00001	0.00778	0.94873	0.38001	0.00014	0.00004	0.49215	0.00004	0.16255

Table 4.3 Reported p-values from the wilcoxon rank sum test for optimizing the momentum value.

Ratios obtained. Since the momentum constant is of more importance, 10 values were tested for the momentum starting from -1 to 1, in increments of 0.2. While testing values for the momentum, the cognitive component, c_1 was set to 1.7 and the social component c_2 was set at 1.7 as well. The resulting p-values from the pairwise wilcoxon rank sum test are reported in Table 4.3. There was a statistically significant difference between 0.6 and all other tested values and $\omega = 0.6$ performed better compared to all other values. As seen in Figure 4.3 $\omega = 0.6$ had a tighter range and a higher mean compared to all the other values. Therefore, $\omega = 0.6$ was chosen as the momentum constant.

4.2 Optimizing GA parameters

This section covers the performance of GA with regards to its parameter setup to determine the best set of parameters for the portfolio optimization problem.

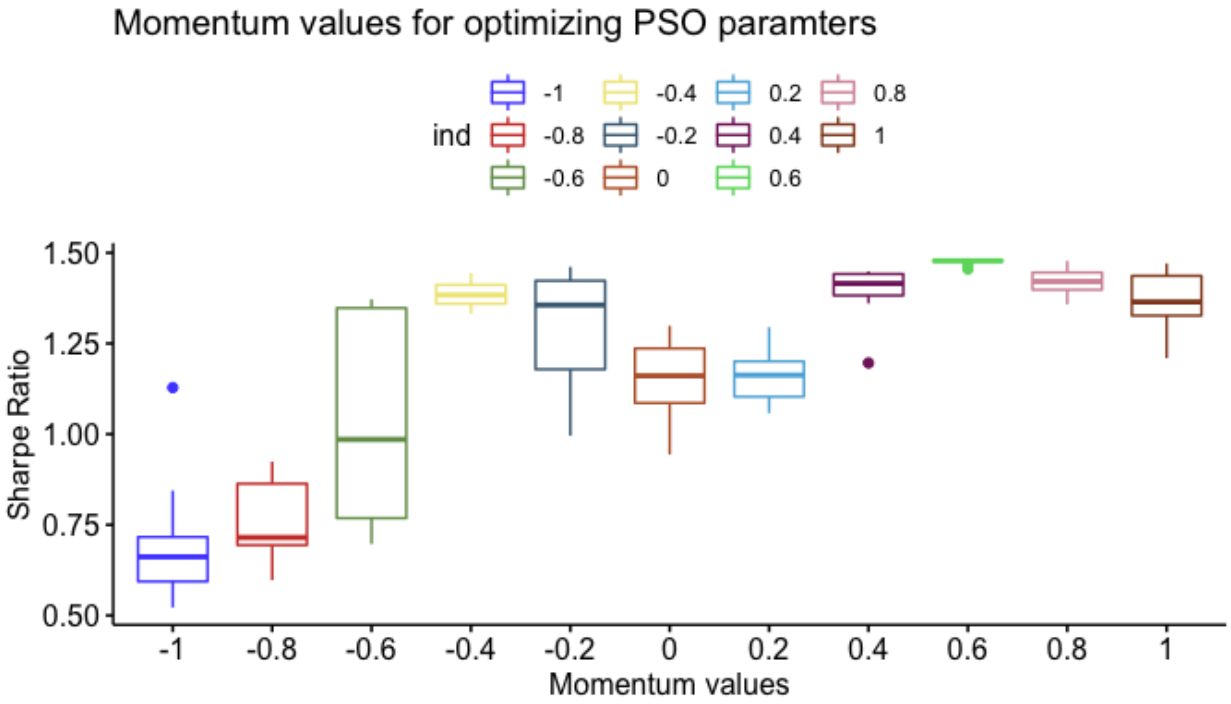

Figure 4.3 Boxplots for Momentum values against the Sharpe Ratios achieved over the 10 runs.

4.2.1 Experimental Setup

All experiments were performed over 80 generations and on the 8 stocks listed above. 10 runs were performed for each parameter configuration.

4.2.2 Crossover Rate

6 values were tested for the crossover rate staring from 0% to 100%, in increments of 20%. While testing different values for the crossover rate, the mutation rate was set to a constant of 80% and the population size was set to 100. The resulting p-values from the pairwise wilcoxon rank sum test are reported in Table 4.4. The values of 80% and 100% performed significantly better than all other values except 60%. There was no statistical difference between 80% and 100%, however, the chosen value for crossover rate was 80% instead of 100% to allow the population to converge in the later generations. With 100%, the crossover

Pairwise Wilcoxon Rank Sum Test P-values for Crossover rate values tested					
Crossover rate (%)	**0**	**20**	**40**	**60**	**80**
20	0.000006	-	-	-	-
40	0.000006	0.002101	-	-	-
60	0.000006	0.000037	0.214138	-	-
80	0.000006	0.000006	0.005594	0.321414	-
100	0.000006	0.000006	0.002101	0.222908	0.948729

Table 4.4 Reported p-values from the wilcoxon rank sum test for optimizing the crossover rate.

will always occur and thus not allow any convergence at all. An accompanyhing boxplot of the different crossover rate values tested and their resulting Sharpe Ratio is given in Figure 4.4.

4.2.3 Mutation Rate

Similar to the crossover rate, 6 values were tested for the mutation rate starting from 0% to 100%, in increments of 20%. While testing different values for the mutation rate, the crossover rate was set to a constant of 80% and the population size was set to 100. The resulting p-values from the pairwise wilcoxon rank sum test are reported in Table 4.5. A mutation rate of 80% and 100% both performed statistically significantly different from all other tested values. 0% and 20% also had a significant difference compared to all other tested, however they performed better than all other values. The difference between 0% and 20% was almsot negligible (p-value 0.047). We chose 20% for the mutation rate as 0% would not introduce any variability in the gene pool and will cause the population to converge too quickly. An accompanying box-plot of the different mutation rate values tested and their resulting Sharpe Ratio is given in Figure 4.5.

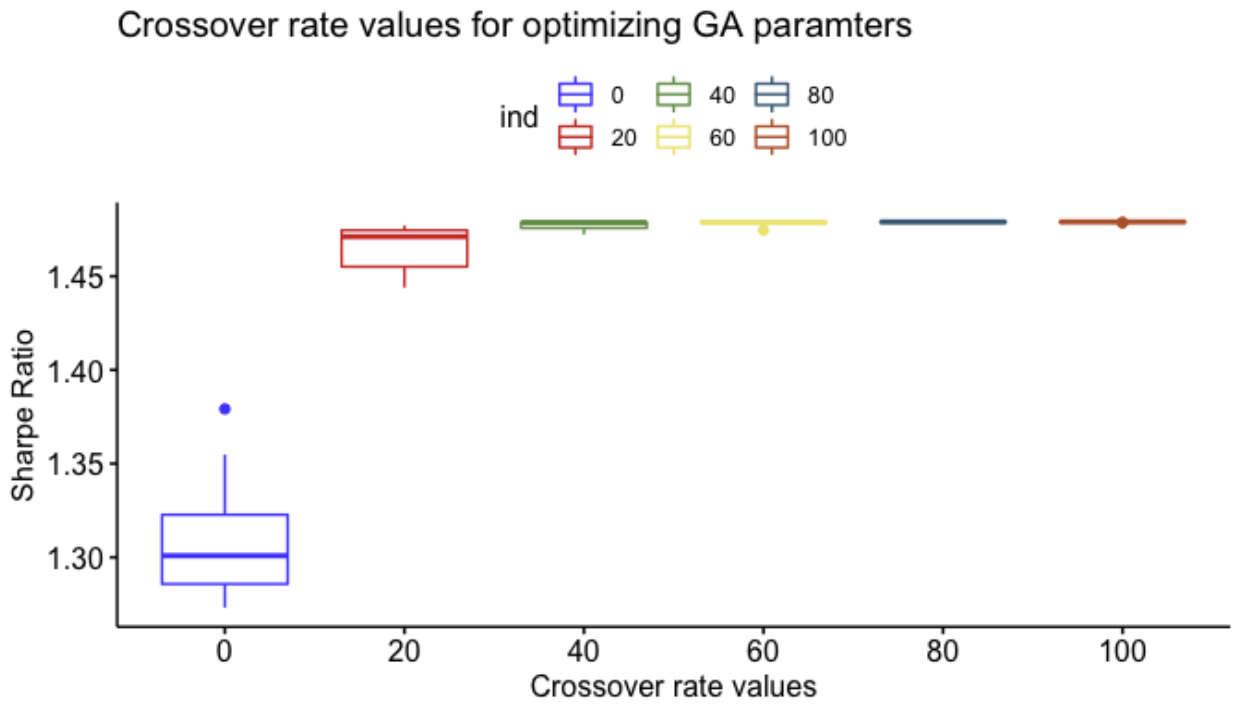

Figure 4.4 Boxplots for Crossover rates against the Sharpe Ratios achieved over the 10 runs.

Pairwise Wilcoxon Rank Sum Test P-values for Mutation rate values tested					
Mutation Rate (%)	**0**	**20**	**40**	**60**	**80**
20	0.047307	-	-	-	-
40	0.000657	0.009611	-	-	-
60	0.000012	0.000012	0.002325	-	-
80	0.000009	0.000009	0.000101	0.047307	-
100	0.000009	0.000009	0.000009	0.000754	0.000754

Table 4.5 Reported p-values from the wilcoxon rank sum test for optimizing the mutation rate.

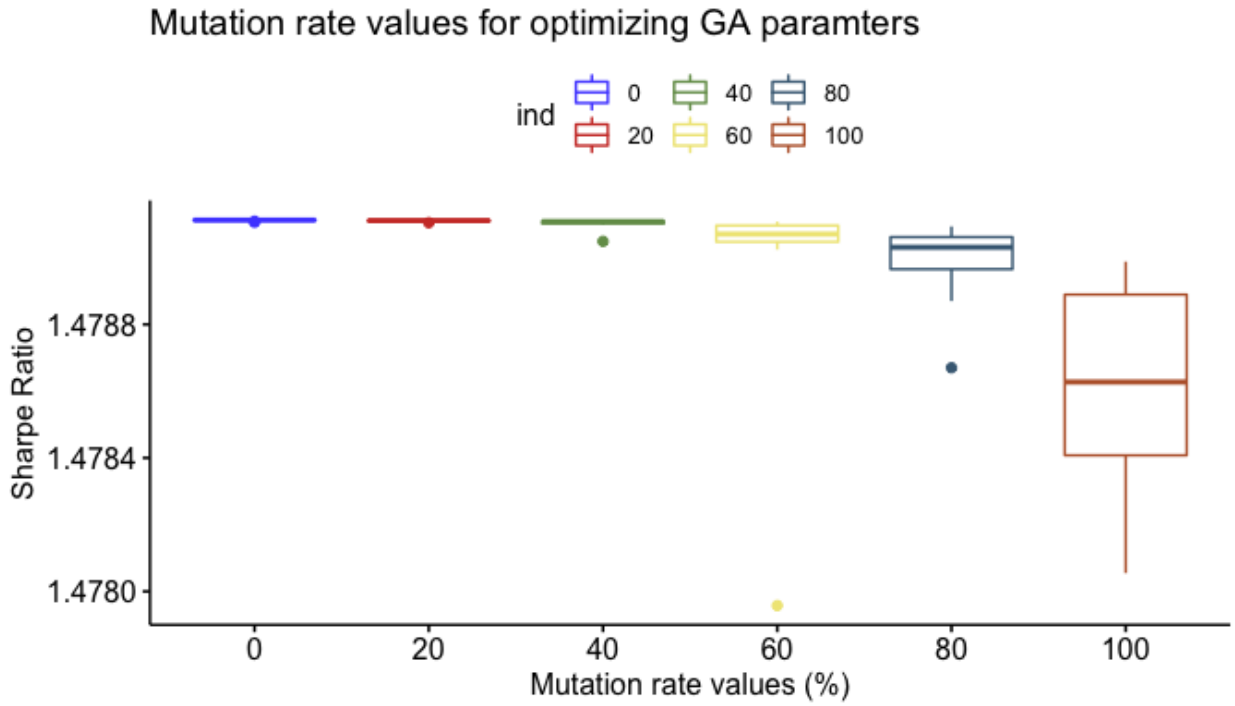

Figure 4.5 Boxplots for Mutation rates against the Sharpe Ratios achieved over the 10 runs.

4.2.4 Population Size

4 values were tested for the population size {50, 100, 150, 200}. The only significant difference was between 50 and the other values and a population size of 50 performed worst compared to all others. Since there was no statistically significant difference between population sizes of 100, 150 and 200, we chose a size of 100 for the next set of experiments as our preliminary results showed that the population converges to a solution rather quickly anyway.

Pairwise Wilcoxon Rank Sum Test P-values for Population size values tested			
Population Size	**50**	**100**	**150**
100	5.67E-06	-	-
150	5.67E-06	0.15959242	-
200	5.67E-06	0.15959242	0.699353021

Table 4.6 Reported p-values from the wilcoxon rank sum test for optimizing the population size.

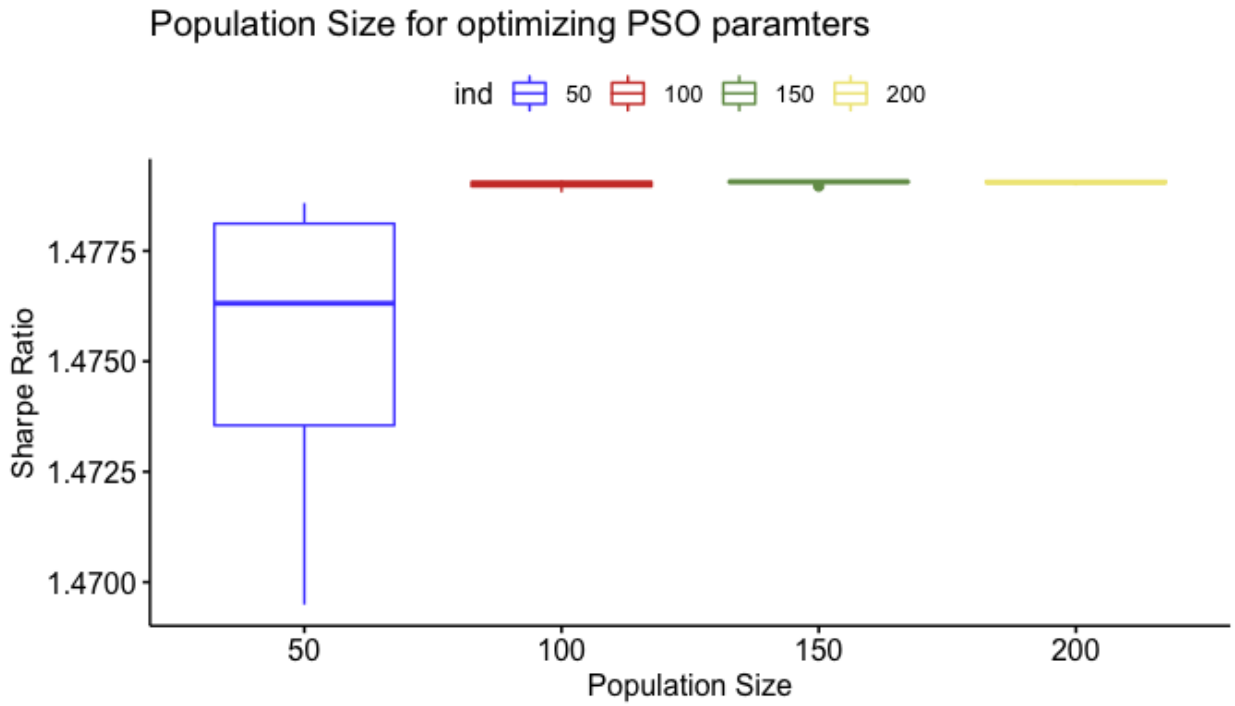

Figure 4.6 Boxplots for Population sizes against the Sharpe Ratios achieved over the 10 runs.

4.3 Genetic Algorithm versus Particle Swarm Optimizing for the portfolio optimization problem

4.3.1 Experimental Setup

The PSO and GA were run for these experiments using the best tested parameters obtained from the previous two sections (Sections 4.1 & 4.2). Portfolios of 8, 16, 24 and 32 stocks were used for the experiments. Random stocks were drawn from the list of stocks to create the portfolios for the number of stocks being tested and then the two algorithms were run on all those portfolios. 30 portfolios were generated for each portfolio size for comparison between the two algorithms and thus 30 runs were performed to ensure statistical significance. For each portfolio created, the two algorithms were run on each portfolio created and therefore the data was analyzed as paired data i.e. for each Sharpe Ratio obtained for a portfolio analyzed using the GA there was an associated Sharpe Ratio obtained using PSO.

4.3.2 Comparison between GA and PSO

Wilcoxon pair wise tests showed that there was a significant difference between the results of GA and PSO for all portfolio sizes tested. GA performed better than PSO in all cases. In the boxplots 4.7-4.10, it can be seen that as the number of stocks in the portfolio increases, the difference between GA and PSO increases as well, with GA performing best for all portfolios. Since the parameters for both algorithms were tested and optimized only using portfolios of size 8, it is possible that PSO requires a change in the parameters as the number of stocks in a portfolio increases. A possible explanation can be that for the portfolio optimization problem, GA parameters have a lesser influence on the performance compared to the PSO parameters, however additional parameter testing will be needed to empirically establish that. Additionally, this suggests that a self adaptive PSO which can tune its parameters values over time may be worthwhile to investigate with the portfolio selection problem.

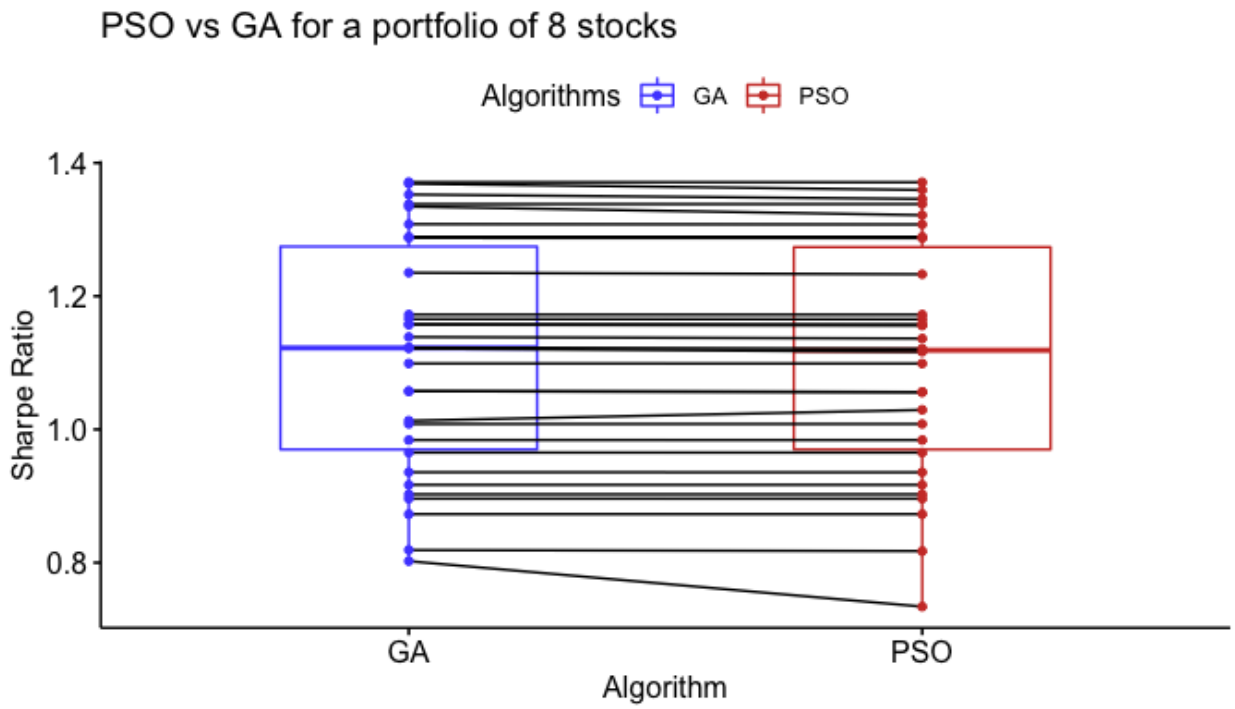

Figure 4.7 Boxplots of achieved Sharpe Ratio for performance comparison between GA and PSO using a portfolio of 8 stocks.

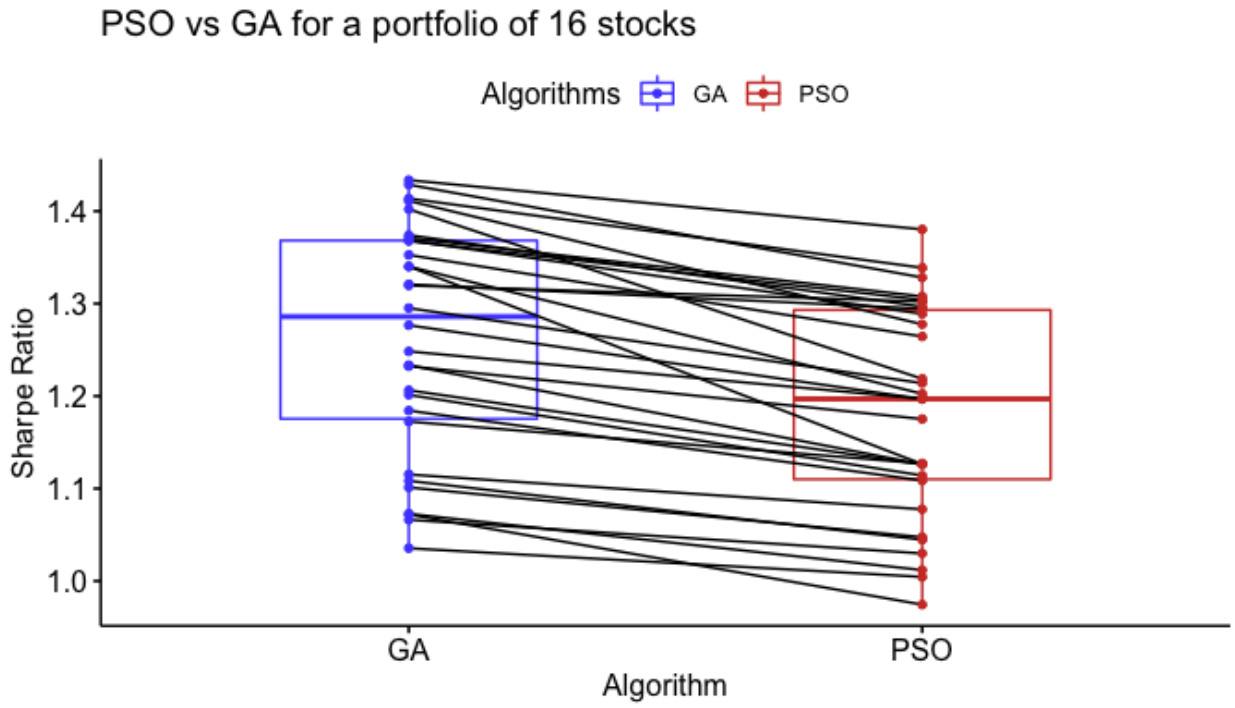

Figure 4.8 Boxplots of achieved Sharpe Ratio for performance comparison between GA and PSO using a portfolio of 16 stocks.

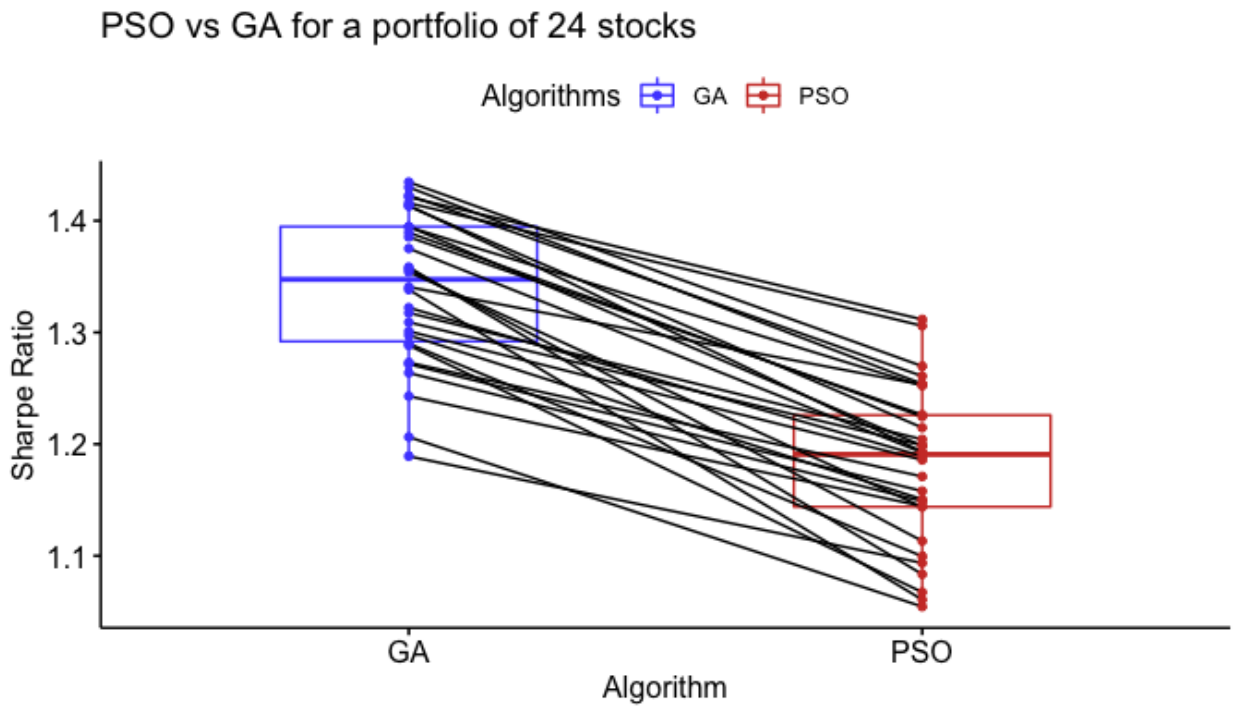

Figure 4.9 Boxplots of achieved Sharpe Ratio for performance comparison between GA and PSO using a portfolio of 24 stocks.

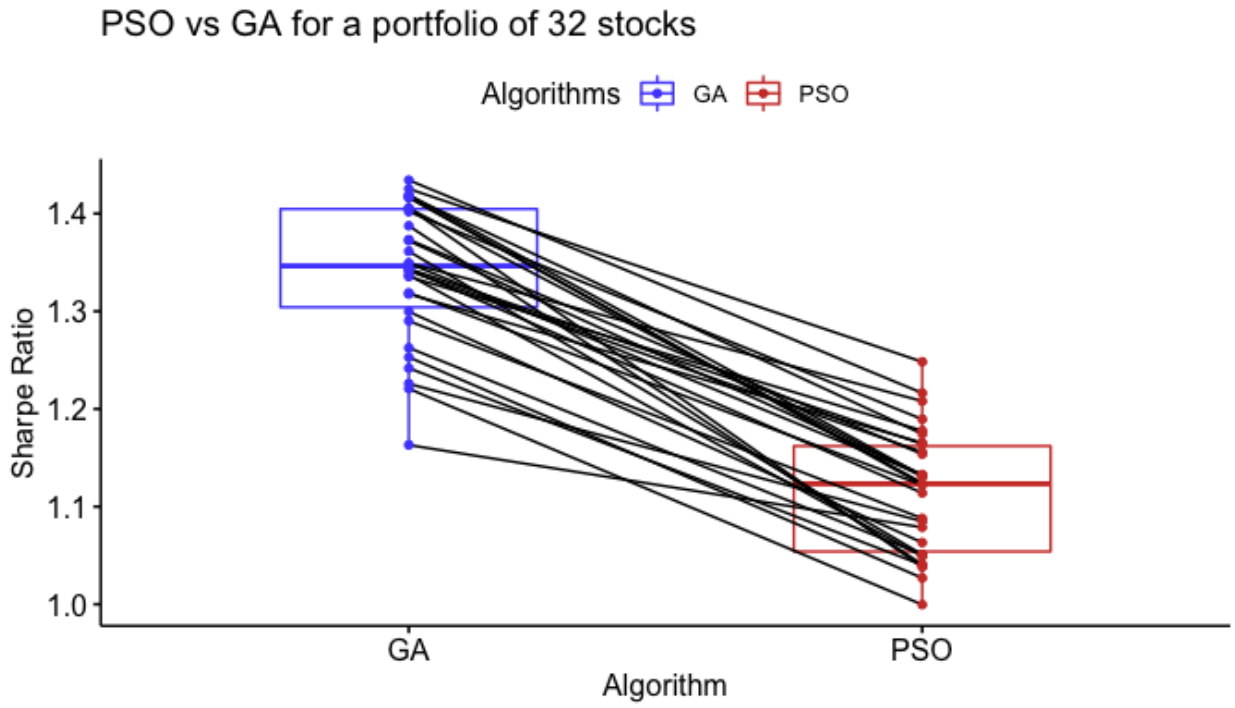

Figure 4.10 Boxplots of achieved Sharpe Ratio for performance comparison between GA and PSO using a portfolio of 32 stocks.

Since momentum was the variable which introduced the most variability in results, the self adaptive PSO should focus more on tuning the momentum value as suggested in [16]. These results do not agree with the results obtained in [11] where the authors concluded that PSO was superior to GA for solving the restricted portfolio optimization problem. They used similar parameters for GA except the mutation rate {Crossover rate: 0.98, Mutation Rate: 0.01, Population size: 100, max generations: 100}. Its possible that in their implementation, such a low mutation rate caused the population to not introduce any additional diversity other than the crossover, leading to a poor performance of the GA. Additionally, [11] does not mention the specific genetic operators used in the GA e.g., crossover and mutation technique. [18] concluded that GA was an effective tool for solving the portfolio optimization problem albeit they did not do a comparison between PSO and GA. However, they used BLX-α crossover as the crossover technique and concluded that through out the evolutionary phase, this crossover played the most significant role in arriving at the optimal solution. Also, [10] which did a comparison between GA and PSO, showed that GA was superior to PSO for the portfolio problem and they used roulette selection for the selection technique. Thus, it can be seen that the selection technique is of immense value to finding a high quality solution for the portfolio optimization problem. The authors of [12] concluded that PSO performed better than GA and also that PSO was able to find the best solution in a quicker time than the GA. Despite not having done any statistical analysis on the computational time, we can confirm that in the experiments of this thesis, the PSO experiments were significantly quicker than the GA counterparts. As the number of assets in a portfolio increased, the computational time difference between GA and PSO increased as well.

Comparing figure 4.11 with figure 4.12, it can be seen that GA typically converges quickly in a smaller number of generations whereas PSO takes much longer to converge. Additionally, the average population fitness in the PSO has a lot of variability as the iterations go on where as in GA, there is a steady increase in the average population fitness along with the best chromosome fitness and a lot less variability compared to a typical run of PSO. During a

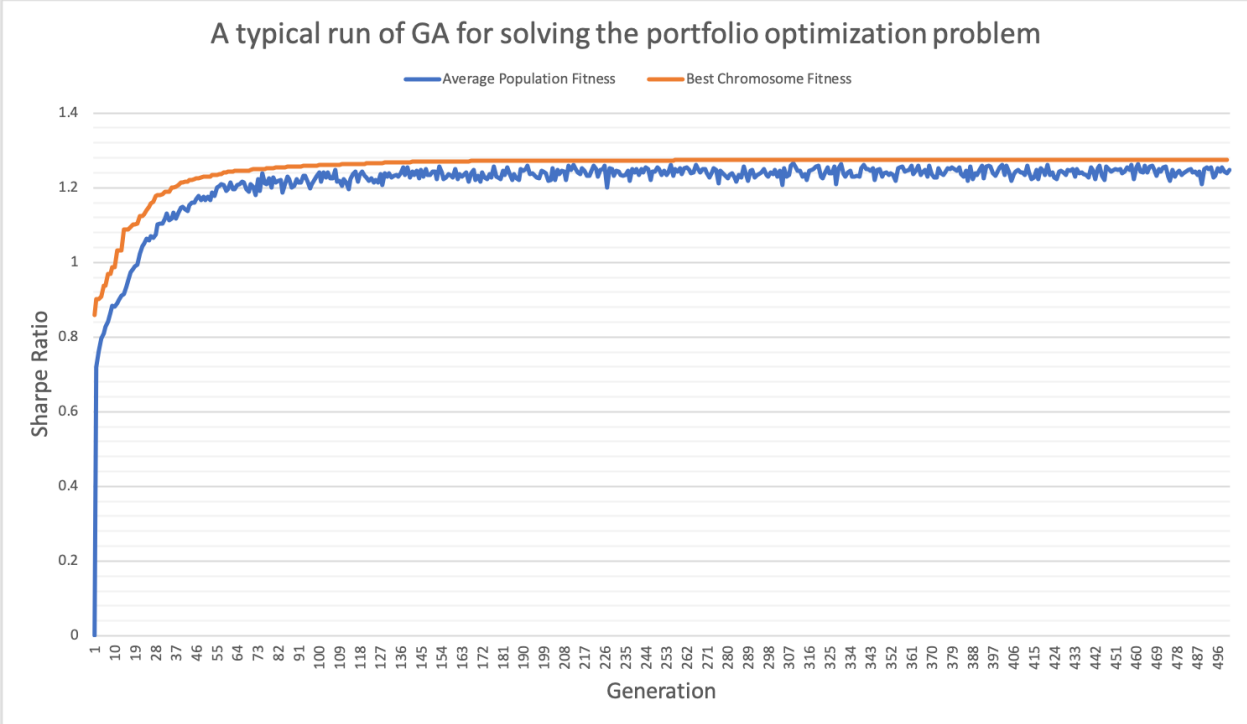

Figure 4.11 A typical run of the GA for solving the restricted portfolio optimization problem, using a portfolio of 16 stocks over 500 generations.

typical run of PSO, it appears that the best position often gets stuck in a local minima and plateaus for a few iterations before a short burst of improvement. In comparison, in figure 4.11 , using GA the Sharpe Ratio increase readily for the first 50 generations before rapidly approaching convergence.

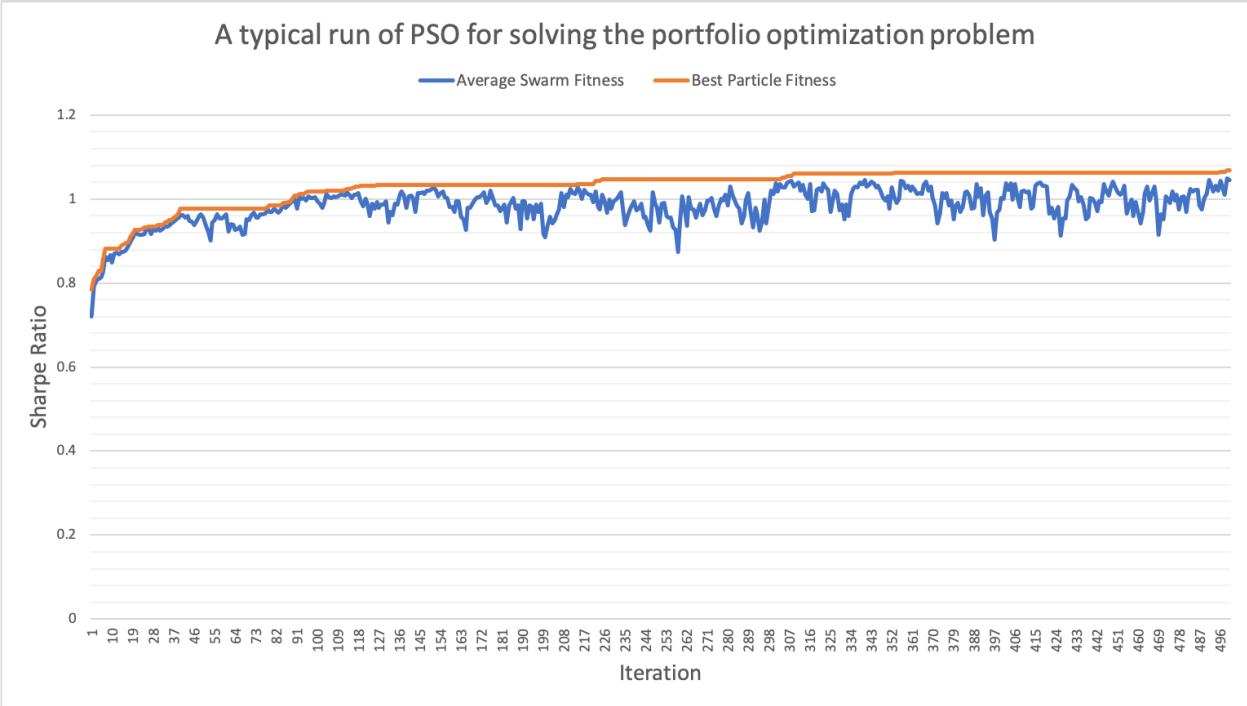

Figure 4.12 A typical run of the PSO for solving the restricted portfolio optimization problem, using a portfolio of 16 stocks over 500 iterations.

Chapter Five

Conclusions and Future Work

Portfolio optimization deals with assisting an investor to make a decision regarding the investment in assets and how the weights should be distributed amongst those assets. The procedure usually involves satisfying multiple constrains and these constrains tend to change depending on the organization's investment decisions and the domain in which the problem is being considered.

This thesis set out to employ a genetic algorithm and particle swarm optimization algorithm to find a high quality solution for the restricted portfolio optimization problem. Sharpe Ratio which is a common measure of portfolio performance and used throughout the literature was used as the fitness function. The two algorithms were independently implemented and tested with various parameter setups to find the best suitable parameters to solve the problem. The best configuration of parameters was then used to compare the two algorithms with each other and see which had a superior performance. Statistical tests were performed on the results to compare the different parameters and compare the two algorithms to ensure that the difference was not from sheer luck but rather there existed a statistically significant difference.

The results indicated that GA performed better than PSO in all our experiments, for all portfolio sizes considered. This result however was not supported by some previous research done in the field [12] [11]. However, the thesis discusses the potential reasons for these

discrepancies for example, the uncertainty of genetic operators used in their research.

There are many fascinating opportunities for further research in the area of portfolio optimization using meta-heuristic based methods. An immediate avenue of further research can be to compare more meta heuristic based methods such as simulated annealing, ant colony optimization etc. and compare their performance with GA and PSO to see which algorithm if any is superior to the rest. Additionally, hybridization techniques incorporating artificial neural networks can be added to the comparisons to determine if they offer any significant benefits compared to the stand alone meta-heuristic methods. Another future investigation may be to develop and test methods to improve the computation efficiency of GA and PSO as the portfolio size increases. Additional genetic operators can be tried for the GA to determine which type of selection, crossover and mutation techniques perform the best for optimizing a portfolio. Regarding PSO, more testing can be done on the parameter configuration to determine how the parameters impact the performance as the iterations go on such as using a dampening inertia constant that decreases as the swarms starts to converge. The different variations of PSO can also be used to compare their performance and see if any one of them performs significantly better for the problem of portfolio optimization.

REFERENCES

[1] Harry Markowitz. "PORTFOLIO SELECTION*". In: *The Journal of Finance* 7.1 (1952), pp. 77–91. DOI: 10.1111/j.1540-6261.1952.tb01525.x. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1111/j.1540-6261.1952.tb01525.x. URL: https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1540-6261.1952.tb01525.x.

[2] Larry J. Eshelman and J. David Schaffer. "Real-Coded Genetic Algorithms and Interval-Schemata". In: *Foundations of Genetic Algorithms* (1993), pp. 187–202. DOI: 10.1016/b978-0-08-094832-4.50018-0.

[3] J. Kennedy and R. Eberhart. "Particle swarm optimization". In: *Proceedings of ICNN'95 - International Conference on Neural Networks*. Vol. 4. 1995, 1942–1948 vol.4.

[4] Melanie Mitchell. *An introduction to genetic algorithms*. MIT, 1998.

[5] Frans Bergh. "An Analysis of Particle Swarm Optimizers". In: (Jan. 2002).

[6] Chieh-Yow Chianglin. "Applications of Genetic Algorithm to Portfolio Optimization with Practical Transaction Constraints". In: *Proceedings of the 9th Joint Conference on Information Sciences (JCIS)* (2006). DOI: 10.2991/jcis.2006.273.

[7] M. A. Dashti et al. "Implementation of particle swarm optimization in construction of optimal risky portfolios". In: *2007 IEEE International Conference on Industrial Engineering and Engineering Management* (2007). DOI: 10.1109/ieem.2007.4419303.

[8] Karel P. Bergmann, Renate Scheidler, and Christian Jacob. "Cryptanalysis using genetic algorithms". In: *Proceedings of the 10th annual conference on Genetic and evolutionary computation - GECCO 08* (2008). DOI: 10.1145/1389095.1389297.

[9] M. R. AlRashidi et al. "Particle Swarm Optimization and Its Applications in Power Systems". In: *Computational Intelligence in Power Engineering*. Ed. by Bijaya Ketan Panigrahi, Ajith Abraham, and Swagatam Das. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 295–324. ISBN: 978-3-642-14013-6. DOI: 10.1007/978-3-642-14013-6_10. URL: https://doi.org/10.1007/978-3-642-14013-6_10.

[10] A. Talebi, M. A. Molaei, and M. J. Sheikh. "Performance investigation and comparison of two evolutionary algorithms in portfolio optimization: Genetic and particle swarm

optimization". In: *2010 2nd IEEE International Conference on Information and Financial Engineering*. 2010, pp. 430–437.

[11] Hanhong Zhu et al. "Particle Swarm Optimization (PSO) for the constrained portfolio optimization problem". In: *Expert Systems with Applications* 38.8 (2011), pp. 10161–10169. DOI: 10.1016/j.eswa.2011.02.075.

[12] Samira Kamali. "Portfolio Optimization Using Particle Swarm Optimization And Genetic Algorithm". In: *Journal of Mathematics and Computer Science* 10 (May 2014), pp. 85–90. DOI: 10.22436/jmcs.010.02.01.

[13] Mariela Cerrada et al. "Multi-Stage Feature Selection by Using Genetic Algorithms for Fault Diagnosis in Gearboxes Based on Vibration Signal". In: *Sensors* 15.9 (2015), pp. 23903–23926. DOI: 10.3390/s150923903.

[14] M. Hajihassani, D. Jahed Armaghani, and R. Kalatehjari. "Applications of Particle Swarm Optimization in Geotechnical Engineering: A Comprehensive Review". In: *Geotechnical and Geological Engineering* 36.2 (Apr. 2017), pp. 705–722. DOI: 10.1007/s10706-017-0356-z.

[15] Kyle Harrison. *COSC 3P71 Lecture, PSO: Fundamentals and Extensions*. Oct. 2018.

[16] Kyle Robert Harrison, Beatrice M. Ombuki-Berman, and Andries P. Engelbrecht. "An Analysis of Control Parameter Importance in the Particle Swarm Optimization Algorithm". In: *Lecture Notes in Computer Science Advances in Swarm Intelligence* (2019), pp. 93–105. DOI: 10.1007/978-3-030-26369-0_9.

[17] Hsiao-Chung Lin, Ping Wang, and Wen-Hui Lin. "Implementation of a PSO-Based Security Defense Mechanism for Tracing the Sources of DDoS Attacks". In: *Computers* 8.4 (Dec. 2019), p. 88. ISSN: 2073-431X. DOI: 10.3390/computers8040088. URL: http://dx.doi.org/10.3390/computers8040088.

[18] Emanuele Stomeo et al. "Genetic Algorithm Application to Portfolio Optimisation". In: 2019.

[19] Mario F. Triola and Jason Roy. *Biostatistics for the biological and health sciences*. Pearson, 2019.

[20] *U.S. Department of the Treasury*. Mar. 2020. URL: https://home.treasury.gov/.

[21] David Iclănzan. "The Creativity Potential Within Evolutionary Algorithms". In: *Advances in Artificial Life Lecture Notes in Computer Science* (), pp. 845–854. DOI: 10.1007/978-3-540-74913-4_85.

[22] *R documentation*. URL: https://www.rdocumentation.org/packages/stats/versions/3.6.2/topics/p.adjust.

APPENDIX

Appendix A

Additional Figures From The Experiments Performed

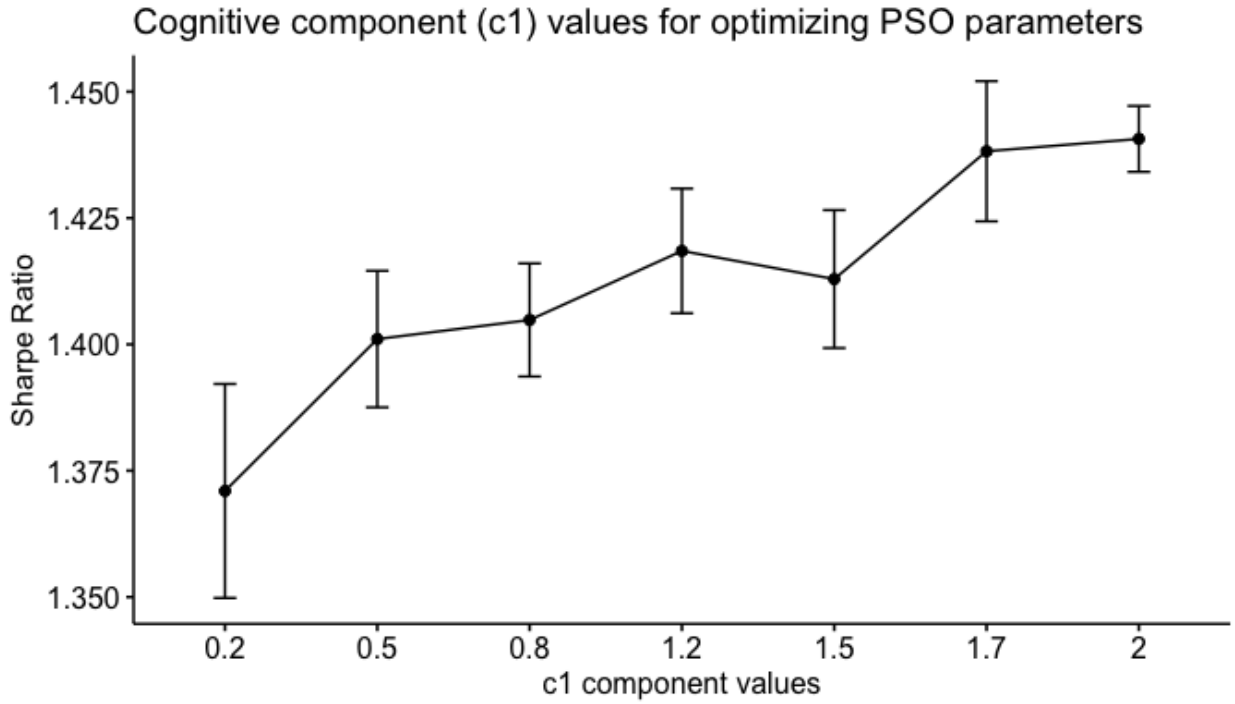

Figure A.1 Line charts with mean and standard error for cognitive component values against the Sharpe Ratios achieved over the 10 runs of the PSO.

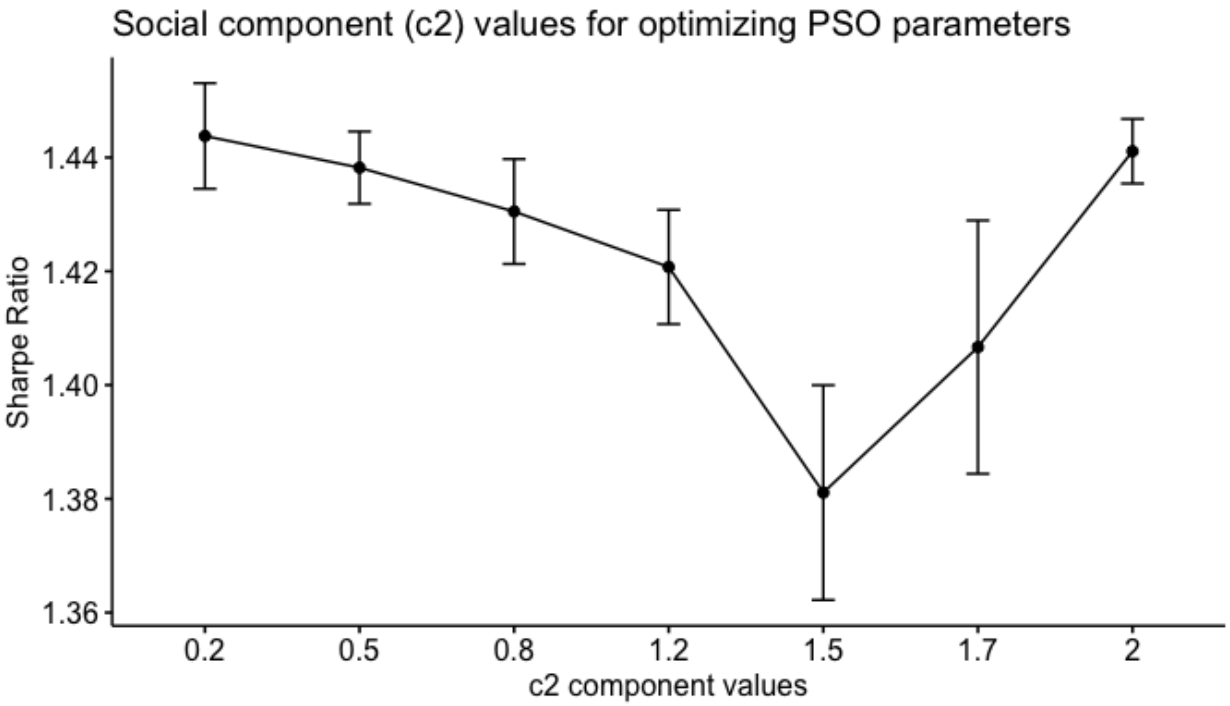

Figure A.2 Line charts with mean and standard error for social component values against the Sharpe Ratios achieved over the 10 runs of the PSO.

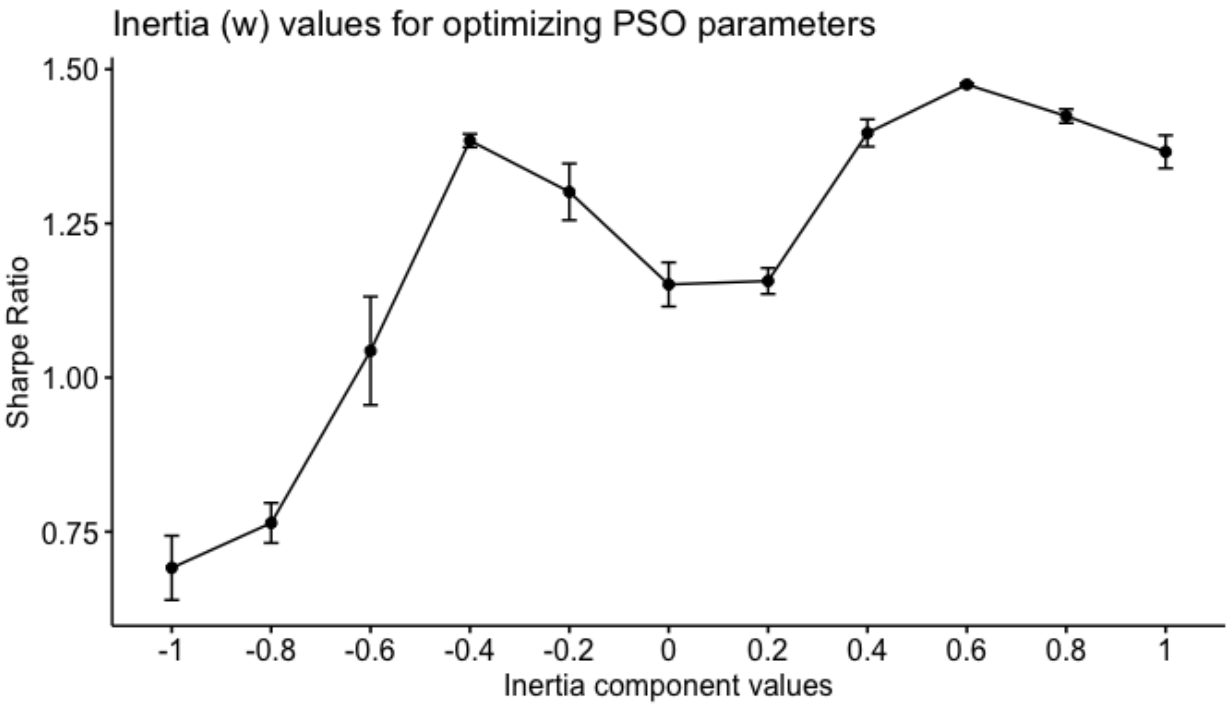

Figure A.3 Line charts with mean and standard error for momentum values against the Sharpe Ratios achieved over the 10 runs of the PSO.

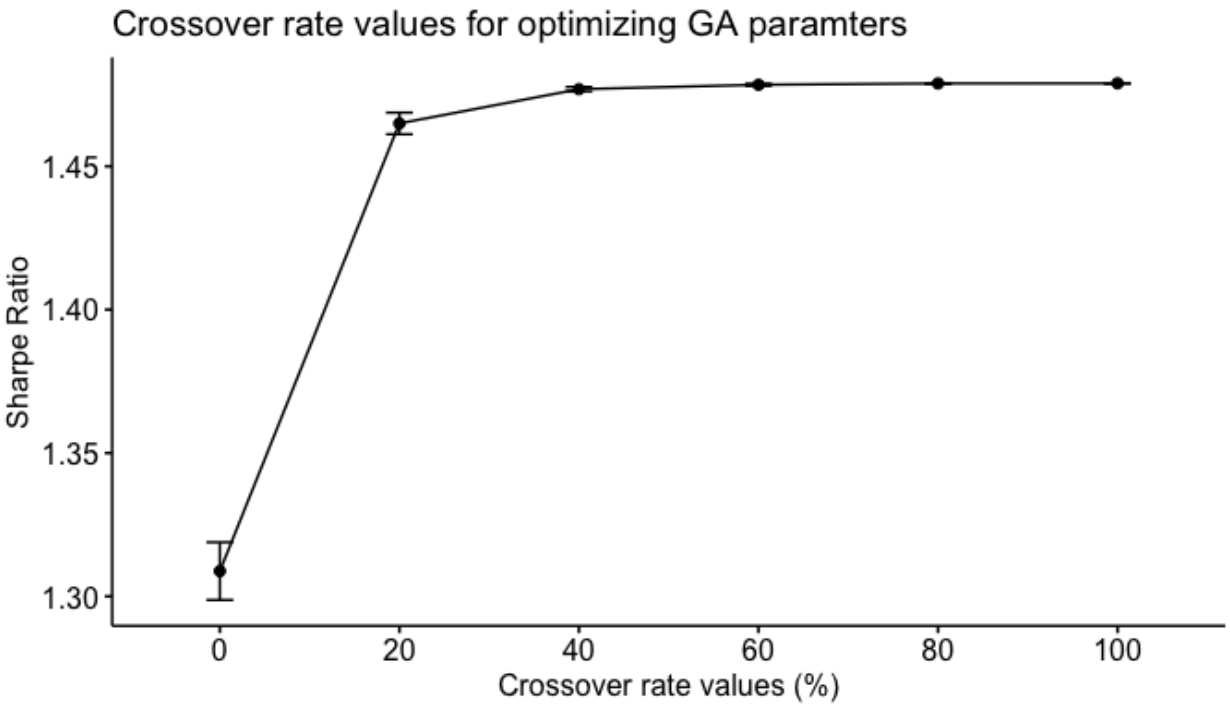

Figure A.4 Line charts with mean and standard error for crossover rate values against the Sharpe Ratios achieved over the 10 runs of the GA.

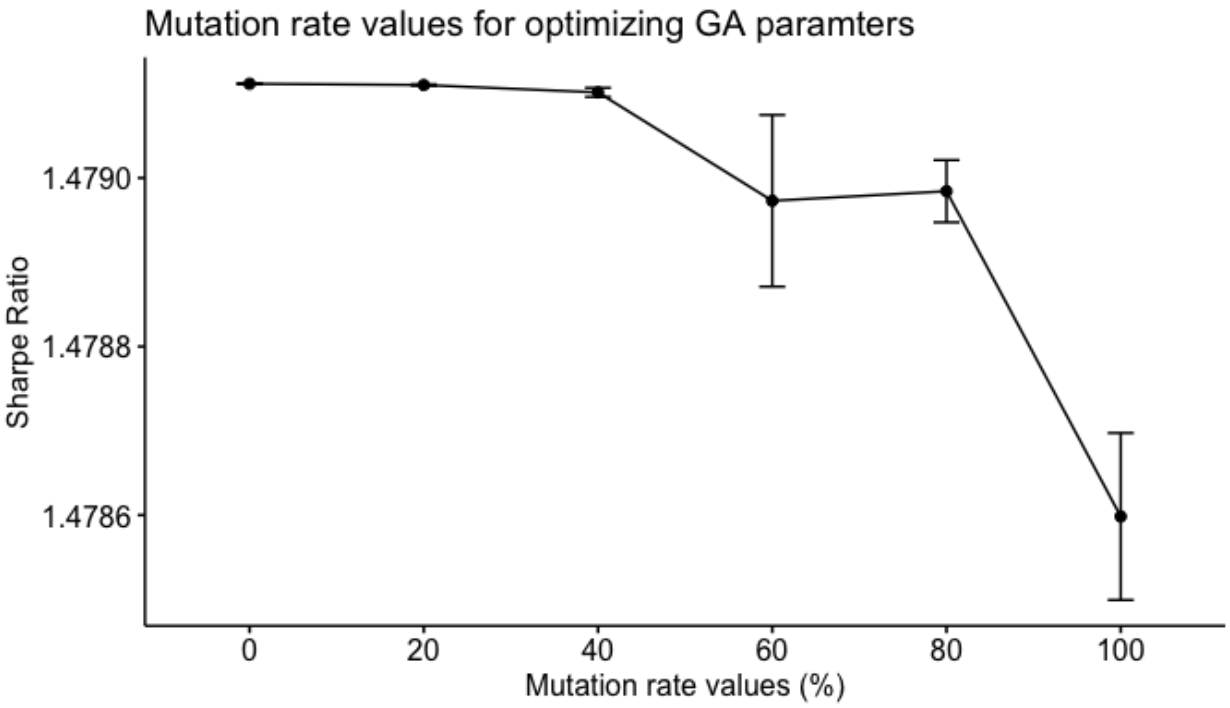

Figure A.5 Line charts with mean and standard error for mutation rate values against the Sharpe Ratios achieved over the 10 runs of the GA.

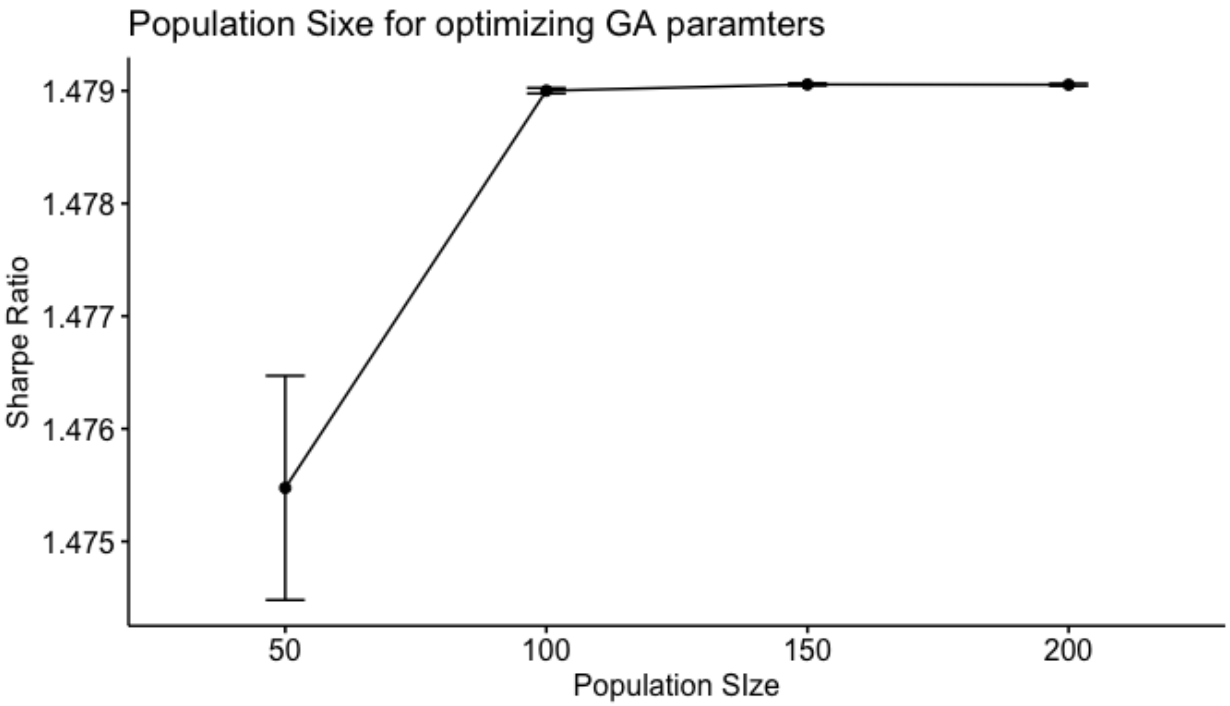

Figure A.6 Line charts with mean and standard error for the population size values against the Sharpe Ratios achieved over the 10 runs of the GA.