MultiObjective Genetic Programming for Financial Portfolio Management in Dynamic Environments

Ghada Nasr Aly Hassan

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
of
University College London.

Department of Computer Science
University College London

I, Ghada Nasr Aly Hassan, confirm that the work presented in this thesis is my own. Where information has been derived from other sources, I confirm that this has been indicated in the thesis.

Abstract

Multiobjective (MO) optimisation is a useful technique for evolving portfolio optimisation solutions that span a range from high-return/high-risk to low-return/low-risk. The resulting Pareto front would approximate the risk/reward Efficient Frontier [Mar52], and simplifies the choice of investment model for a given client's attitude to risk.

However, the financial market is continuously changing and it is essential to ensure that MO solutions are capturing true relationships between financial factors and not merely over fitting the training data. Research on evolutionary algorithms in dynamic environments has been directed towards adapting the algorithm to improve its suitability for retraining whenever a change is detected. Little research focused on how to assess and quantify the success of multiobjective solutions in unseen environments. The multiobjective nature of the problem adds a unique feature to be satisfied to judge robustness of solutions. That is, in addition to examining whether solutions remain optimal in the new environment, we need to ensure that the solutions' relative positions previously identified on the Pareto front are not altered.

This thesis investigates the performance of Multiobjective Genetic Programming (MOGP) in the dynamic real world problem of portfolio optimisation. The thesis provides new definitions and statistical metrics based on phenotypic cluster analysis to quantify robustness of both the solutions and the Pareto front. Focusing on the critical period between an environment change and when retraining occurs, four techniques to improve the robustness of solutions are examined. Namely, the use of a validation data set; diversity preservation; a novel variation on mating restriction; and a combination of both diversity enhancement and mating restriction. In addition, preliminary investigation of using the robustness metrics to quantify the severity of change for optimum tracking in a dynamic portfolio optimisation problem is carried out.

Results show that the techniques used offer statistically significant improvement on the solutions' robustness, although not on all the robustness criteria simultaneously. Combining the mating restriction with diversity enhancement provided the best robustness results while also greatly enhancing the quality of solutions.

To the ones I love the most: Khaled, Omar and Kareem

Acknowledgements

I am grateful to my two supervisors, Christopher Clack and Philip Treleaven. This work would not have been possible without their guidance and encouragement. In particular, I have benefited a great deal from working closely with Chris. I believe I was able to improve my research skills through learning from his: reading critically; spotting what is important; and working thoroughly. I am also pretty sure my English has improved through the regular correction of my written texts!

I would like to thank many members of my family, who have given me unconditional support; both emotional and financial. My dear husband has understood the importance of this journey for me, was always there to lift me up, and has willingly made many sacrifices that are only justified by love. My mother took the time and the effort to be in London several times to give very much needed help. My parents in law were tremendously supportive at every step of the way. They all had faith in me, and at some points, more faith than I had in myself. I would not have been able to do it without them, and I will always be indebted to them for this, as well as everything else.

I am very lucky to have come to know many wonderful friends in the 8.11 lab and beyond throughout my stay in London. They made the day-to-day life much more fun and much less lonely. Thank you all for being who you are. I would especially like to acknowledge my dear friend Chi-Chun Chen, who is just an awesome person, my deepest thanks to her for the lovely times, the interesting chats, and the proof-reading!

Last, but certainly not least, my research is funded by a scholarship from the Egyptian government and the Missions program. I am thankful to them for giving me this opportunity. I am also thankful to the Egyptian Cultural Bureau in London for their on-the-ground support, especially the Counsellors Alla El-Gindy, and Amre Abu-Ghazala. I would also like to acknowledge Reuters who, through an agreement with my supervisors and UCL, have provided the stock market data used in this research.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

In the field of computational finance, Machine Learning (ML) and Artificial Intelligence (AI) algorithms and techniques are often used to investigate problems in the area of finance and economics. Machine Learning algorithms are distinguished by their capability for mechanical self improvement through experience, as evident in the Neural Network (NN) learning algorithms, Evolutionary Algorithms (EA), Decision Trees and Reinforcement Learning. The flourishing computational finance field owes its success to the current advances in computing in both algorithms and hardware; more data on financial systems is becoming available; algorithms suitable for deeper analysis of data are being developed; the processing power of computer hardware is accelerating; and memory storage capacity is of much less concern now than it used to be. These factors are enabling researchers to analyse the complex financial and economical models better, helping them to gain more insight into the dynamics governing the financial market and assist the decision-making process based on the new knowledge acquired. There are a number of financial areas in which machine learning algorithms have been applied, such as [TMJ04]: forecasting financial time series, exploiting arbitrage opportunities, challenging the fundamentals of finance and economics, and portfolio selection and management.

This thesis focuses on evolving rules for stock selection in a portfolio management problem. In the early conventional view of investment and trading, optimisation methods and trading strategies often assumed return maximisation as the sole objective of the investor, and as the only performance measure. The introduction of the Markowitz mean-variance approach [Mar52], brought to light 'risk' (to be minimised) as another objective of the rational investor and emphasised the concept of selecting a portfolio of investment assets which collectively have lower risk than any single individual asset therein. Markowitz described the portfolio optimisation problem as a problem of optimising two contradicting objectives of risk and return, and the problem came to be recognised as a multiobjective (MO) one. Recently, it is increasingly acknowledged that investors may actually have, in addition, more objectives that they would like their investment to fulfil. Examples of such objectives are portfolio liquidity, various measures of risk, dividend return, growth in sales, and amount invested in R&D, as shown in [CAS04] and [SQ05].

The recognition of multiobjective optimization problems in finance, and using MOEA to solve them is a recent development. Examples of applying MOEA in financial applications [1] include: risk management [SMS05], bank loan management [MBDM02] as well as management of financial portfolios [AL05, LT06, SBE+05, Lau05, MERTS06, SKN07].

Existing research reveals successes as well as problems and gaps. Some are specific to the financial domain but others are inherent in the multiobjective algorithms. One of these problems is the performance and applicability of MO algorithms when the environment is changing. We call this problem 'robustness in dynamic environments' and it deals with two aspects of multiobjective optimization. The first of these is how to examine and improve the performance of the multiobjective solutions on out-of-sample environments. Secondly, it is concerned with how to measure the degree of environment change beyond which the algorithm will need to be modified or retrained to adapt to the changing environment. Robustness is *not a requirement* in a stationary environment, where the problem at hand has its environment fixed by definition; that is to say optimisation problems in which the range and behaviour of the input data is known and fixed, and the optimization algorithm task is to find solutions that will yield the best tradeoffs between objectives once and for all. Robustness is *desirable* when the change in the environment is small due to noise in the input decision variables or the uncertainty of objectives. However, when the optimization problem has a continuously changing environment by nature, then investigating robustness becomes an *important* requirement.

What concerns this research in particular is the correlation between the time series data that are available to the stock-picking model in training and the future performance of the rules evolved based on available data. Our stock-picking models do not have to predict precise future stock prices, but they are required to rank the stocks correctly based on establishing a relationship between their historical fundamental/technical factors and the stock's future performance. If the correlation between time series and future stock performance changes, then a given model (equation) may become less effective in ranking stocks. Previous work on multiobjective GP [LT06] and single objective GP [YC06] have shown that the performance of MOGP and GP evolved stock-picking equations can vary substantially when used in an out-of-sample-environment. However, all previous studies have only considered optimality (as measured by one or another financial related measure for investment success). In the case of a single objective GP, the output is one solution, and the optimality of this solution on out-of-sample environments is often used as measure of the robustness of the algorithm [Kab00, PSV04, YC06]. In the case of multiobjective GP, and since the output is a set of solutions, the average performance of the solution set is sometimes taken as the measure of performance on the out-of-sample environment as in [Bin07, BFF07, BJ08], and hence the robustness of the algorithm. In practice, however, investment will be carried out using individual rules from the solution set. The individual solutions will be selected based on the client's appetite for risk,

[1] For a review of applications of MO Evolutionary algorithms in finance and economics see [Coe06] [TGC07]

and in return his expectation of a certain degree of profit. Hence, in addition to optimality on the out-of-sample environment, a practitioner will be equally interested in ensuring that the perceived relative positions of solutions on the Pareto front do not switch (for example a situation where a "lowest-risk" portfolio becomes "highest-risk").

1.1 Motivation

Robustness of MOGP solutions is extremely important for the real-world problem of stock-picking for a monthly investment portfolio. It is therefore essential for MOGP solutions to be analysed in *unseen* environments, not just in training, and although it may be difficult to define an absolute measure of solution robustness we must be able to determine which of two solutions is more robust, and which of two Pareto fronts is more robust.

The motivation for this research is the investigation of the effectiveness of a multiobjective genetic programming approach, to evolve robust factor models for stock ranking, in a real world portfolio management problem. The MOGP evolved Pareto front accurately depicts the tradeoff between risk and return in training (although the real Pareto front is generally not known). To ensure the practical utility of such an algorithm, the robustness of solutions in subsequent unseen (test) environment is examined.

In volatile environments such as the financial markets, robustness is of major importance. If robustness is not achieved, the solutions will exhibit unstable performance that may render them unfit in subsequent environments, and the algorithm is only useful as an analysis tool of historical data. Investors ideally prefer a solution whose specific risk and return never changes. Given that this is highly unlikely ever to be achieved in a volatile market, the next best solution is one that sustains the characteristics of its objectives. For example, given a solution with the lowest risk relative to the available alternative solutions, it should continue to give the lowest relative risk as the environment changes (even though the precise objective value of risk may change). This aspect of robustness performance of the MO algorithms has not previously been identified. We believe that this particular aspect is as important as optimality when it comes to the practicality of using MO solutions in real life optimization problems.

1.2 Approach

So how should the system respond to market instability? One obvious response is to employ dynamic adaptation via retraining, using new training data drawn from the new environments. However, in the context of monthly investment this is problematic:

- The most pressing problem is the lack of new data, because the time series comprises only monthly data — it is not feasible to train on just a handful of data points, so the system must wait many months before sufficient new data has been gathered to permit retraining (and by that time, the market may have changed again);
- Rather than waiting many months, the system may employ a "sliding window" method

where it continuously retrains on the most recent (say) twelve months of data — the disadvantage with this approach is that for the first (say) six months following a change the training data will predominantly come from the old environment, and so it will still take considerable time before more suitable equations can be evolved;

- How often should the system retrain? Too frequently, and too little data will be available; too infrequently, and the retraining may be ineffective because it happens at the wrong time (for example, just before a change in the market);

- Should the system only retrain when a change in the market trend is detected? This would appear to be a better solution, but turns out to be difficult to achieve, as explained below.

Certain gross behaviour of the financial markets (for example, a "bull", "bear", or "volatile" market) can be identified by inspection of the behaviour of a benchmark portfolio (or "index" portfolio) which invests in all stocks equally (alternatively, investing in all stocks using a standard weighting such as capitalisation). The index can therefore be used to identify a change in market environment. However, it turns out to be very difficult to detect the point at which a market changes — it is relatively easy to identify a "bull" or a "bear" market once it is established, but at the turning point it can be difficult to know for certain whether the market is really changing, and difficult to determine the nature of the new market (is it changing from "bull" to "bear" or from "bull" to "volatile"?).

1.3 Problem Statement

Despite the broad range of research in MO algorithms, most of this work has focused on static optimisation and hence on generating solutions on the Pareto front that are diverse and well-distributed. Little attention has been paid to the *robustness* of solutions evolved in dynamic environments, and they are not always validated in out-of-sample environments. When MO algorithms are used for dynamic optimisation, two issues arise: the performance of the solutions found in training when used in new environments, and the ability of the algorithm to continuously adapt to changing environments.

Whilst retraining is an important tool in responding to the instability of the markets it is insufficient on its own; it is also necessary to ensure that the evolved models will continue to perform *reasonably* well when the market environment in which they are used is different to that in which they were trained. We do not expect them to continue to behave well but we can require that they degrade gracefully within a range of market change and do not suddenly produce catastrophically wrong results (it would be unreasonable to expect good behaviour for a sudden extreme change). We call this "solution robustness" and it is important because it provides a period of time within which either new data can be gathered for retraining or human intervention can take over prior to retraining.

The aim of this research is to investigate the robustness of MOGP solutions to the multi objective real world problem of portfolio management. We achieve this objective through introducing new definitions and metrics for robustness in the multiobjective context, and developing techniques for improving robustness of solutions on the Pareto front. We also provide preliminary results on quantifying change in the stock market environments for which solutions are no longer valid and re-optimization is required.

1.4 Contribution

This thesis provides an empirical study of using an MOGP to evolve robust non-linear factor models for stock selection in a portfolio optimization problem with multiple objectives, and an assessment of the performance/robustness of the MOGP solutions when applied to out-of-sample data. It also demonstrates the value of an MOGP approach to a finance practitioner.

The thesis makes the following contributions:

1. The development of new definitions and metrics for the robustness of MOGP solutions and robustness of the Pareto fronts in dynamic environments.
2. The use of the new definitions and metrics to assess the effect on robustness in unseen environments of:
   (a) Selection bias.
   (b) Diversity preservation.
   (c) Cluster-based mating restriction.
3. A preliminary analysis of:
   (a) The Dynamics of change.
   (b) How to quantify the severity of change in the financial environments.
   (c) The use of MOGP as an analysis tool in the financial market.

1.5 Thesis Structure

The rest of the thesis is organised as follows:

- In Chapter 2, background information is provided on genetic computation, and multi-objective optimization concepts, followed by some of the best known MOEAs, then a survey of applications of MOEA in finance and a survey on the related literature on robustness of MO algorithms.
- Chapter 3 provides a critical analysis of the current state of the art in robustness of MO algorithms, followed by suggestions for a new model for robustness.
- In Chapter 4, the system architecture and design of the experiments are presented.

- Chapter 5 provides results of experiments on the robustness of multiobjective genetic programming for portfolio management.
- Chapter 6 presents a discussion and conclusions on the implications of results on robustness of MOGP as well as on the practical use of MOGP in real world portfolio optimization.

1.6 Publications

- G.Hassan and C.Clack. Multiobjective Robustness for Portfolio Optimization in Volatile Environments. GECCO'08: Proceedings of the 10th Annual Conference on Genetic and Evolutionary Computation, pages 1507-1514, Atlanta, GA, USA, 2008. ACM. (Nominated for a best paper award).
- G.Hassan. Non-Linear Factor Model for Asset Selection using Multi Objective Genetic Programming. GECCO'08 Workshop: Advanced Research Challenges in Financial Evolutionary Computing (ARC-FEC), pages 1859-1862, Atlanta, GA, USA, 2008. ACM.
- G.Hassan and C.Clack. "Robustness of Multiobjective GP Stock-picking in Unstable Financial Markets". GECCO'09: Proceedings of the 11th Annual Conference on Genetic and Evolutionary Computation, pages 1513-1520, Montreal, 2009. ACM.
- G.Hassan and C.Clack. "Dynamic Multiobjective Optimization for Optimum Tracking in Portfolio Optimisation". Submitted for GECCO'10: Proceedings of the 12th Annual Conference on Genetic and Evolutionary Computation.

Chapter 2

Background and Literature Review

This chapter presents background information and reviews existing literature that is relevant to the work in this thesis. The chapter is divided into four main sections:

- Section 2.1 gives the formal definition of multiobjective optimisation problems, and outlines the various approaches for solving them.
- Section 2.2 concentrates on one of those approaches: the multiobjective evolutionary algorithms (MOEA) . It explains the state of the art algorithms belonging to this approach and briefly presents some of the research challenges they currently face.
- Section 2.3 presents a sample of the research on using MOEA in a range of problems from economics and finance.
- Section 2.4 reviews the existing literature on robustness of MOEA, specifically the research on their application in dynamic environments.

2.1 Introduction to Multiobjective Optimisation (MOO)

Multiobjective optimisation is the problem of finding the best solution to an optimisation problem with more one than objective. In this section we first give the formal definition to the problem in Section 2.1.1, and discuss in section 2.1.2 the different solution approaches in the literature. We review the weighted aggregation, multiple populations, Pareto dominance, and alternatives to the Pareto dominance approaches.

2.1.1 Problem Definition

Optimisation refers to the problem of finding the best solution to a problem given a set of inputs and constraints [Coe05b]. In its simpler variety, the single objective optimisation, there is a single objective to be minimised or maximised. The solution to the problem is a global optimum in the search space (as represented by the function that we seek to optimise). However, in many problems, it is usually the case that we seek to optimise a number of – often conflicting– objectives. When the problem seeks to optimise two or more objectives, it is known as multiobjective, and may require different algorithms and tools than those used in the single objective

optimisation problem [Coe05b, FA02]. In multiobjective optimisation, and when the objectives are conflicting, a single solution that can optimise all objectives simultaneously does not exist. Hence, the goal of the search is to produce a set of"trade-offs" between different objectives. The most common notion of optimality to express this trade-off is that proposed by Francis Edgeworth and generalized by Vilfredo Pareto, hence known as Edgeworth-Pareto optimality or often just Pareto optimality. The traditional methods for dealing with the single-objective and multiobjective optimisation problems are sound in the research in the mathematical field of operations research and mathematical programming. Mathematically, an optimisation problem has the following form [CS03]:

minimise $f(x)$ (function to be optimised)
with $g_i(x) \geq 0$ (m inequality constraints)
and $h_i(x) = 0$ (p equality constraints)

Definition 1. ***Objective Function***
The name given to the function f. It is the function for which the algorithm will try to find its optimal value. In multiobjective problems there are multiple objective functions to be optimised.

Definition 2. ***Decision Variables***
Consider a search space Ω with n parameters, these parameters are called the decision variables, and they are the values taken by the vector $x = [x_1, x_2, x_3, ..., x_n]^T$. It is by changing this vector that we are searching for the optimal through traversing the search space Ω.

Definition 3. ***Global Minimum***
In single objective minimisation problems, the global minimum is the optimal result of the search algorithm. The vector x^* is global minimum of the function f if $f(x^*) \leq f(x)$ for any $x \in \Omega$.

Definition 4. ***The Multiobjective Problem***
In mutliobjective problems, for each point in the search space there are m different criteria by which to judge that point. The multiobjective optimisation problem is to find the value of decision variables for $x^* = [x_1^*, x_2^*, \ldots, x_n^*]^T$, from total set of vectors in the search space Ω that will be a solution to:

$F(x) = [f_1(x), f_2(x), \ldots, f_k(x)]$, where k is the number of objective functions
subject to the m inequality constraints $g_i(x) \geq 0$, for $i = 1, 2, \ldots, m$,
and the p equality constraints $h_i(x) = 0$, for $i = 1, 2, \ldots, p$

Definition 5. ***Pareto Dominance***
A solution vector $x = [x_1, x_2, x_3, ..., x_n]^T$ is said to dominate another solution vector $v = [v_1, v_2, v_3, ..., v_n]^T$, denoted by $x \preceq v$, if and only if the following condition holds (assuming

minimisation is required):

$$\begin{aligned} & x \preceq v \quad iff: \\ & f_i(x) \leq f_i(v) \qquad \text{for all i} \in [1, 2, ..., k], \text{and} \\ & f_j(x) < f_j(v) \quad \text{for at least one j} \in [1, 2, ..., k] \end{aligned} \tag{2.1}$$

Definition 6. *Pareto Optimality*

A solution x^* is said to be Pareto optimal in the search space Ω if there is no solution v for which $v \preceq x^*$

This definition says that a vector is Pareto optimal if there exists no other vector of decision variables belonging to the feasible set of solutions that would improve some decision criterion without deterioration in at least one other criterion. This definition always gives a set of solution vectors called the Pareto optimal set (see below). The solution vectors in the set are called non-dominated in the sense that no other solution is better (dominates them). The plot in the value space of the objective functions whose non-dominated vectors constitute the Pareto optimal set is called the Pareto front (see below), as shown in Figure 2.1, where the points a,b,c,d,e lie in the feasible region of the search space, and the point d,e lie on the Pareto front as they are non-dominated by any other points in the feasible region of the space.

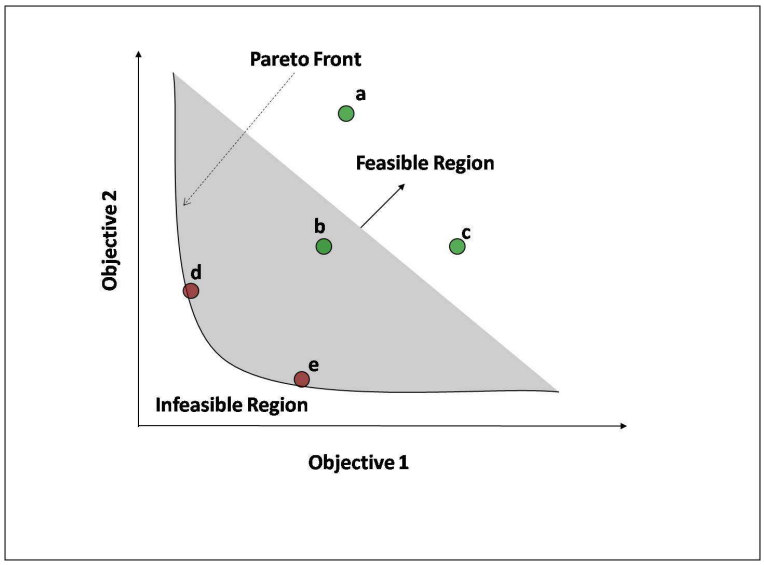


Figure 2.1: Optimisation Trade-off and Pareto Optimality

Definition 7. *Pareto Optimal Set*

For a given multiobjective problem, the Pareto optimal set is defined as:

$$P^* = x \in \Omega \mid \nexists\, v \in \Omega : F(v) \preceq F(x) \tag{2.2}$$

Definition 8. *Pareto Front*

For a given multiobjective problem and a Pareto optimal set, the Pareto front is defined as:

$$PF^* = F(x) = f_1(x), f_2(x), ..., f_n(x) \mid x \in P^* \tag{2.3}$$

2.1.2 Approaches for Multiobjective optimisation

In this section, we provide an overview of four different approaches to multiobjective optimisation problems: weighted aggregation; population approach; Pareto optimality; and alternatives to Pareto optimality.

2.1.2.1 Weighted Aggregation Approach

The intuitive way to solve the multiobjective problem is by weighted aggregation. In this approach, different objectives are weighted and combined in one single objective [LT99]. The weights are non-negative and are usually fixed during optimisation.

Effectively, using weighted aggregation ignores the multiobjective nature of the problem and attempts to solve it as a single objective one. The transformation is achieved through defining another objective f^*, equivalent to the aggregated function of the original objectives $(f_2, f_2, f_3, ..., f_k)$. When integrated within an evolutionary algorithm, the fitness function of the evolutionary algorithm is defined to be this weighted aggregation of objectives. So, for example, the fitness function of a two objective (f_1, f_2) problem will be defined as:

$$f^* = w_1 f_1 + w_2 f_2, \qquad \text{where } w_1 + w_2 = 1$$

Examples of using the weighted aggregation approach as the fitness function within an evolutionary algorithm are found in [JGSB92, SP91].

The weighted aggregation method is easy to understand and implement. However, the method will yield a single solution for every combination of weights used in any one single run. It is also quite impossible to know the correct weights to generate points evenly distributed along the Pareto front if we do not know its exact shape, as discussed and analysed in [DD97]. In addition, the weighted aggregation method will only be able to find the solutions if the Pareto front is convex. Otherwise, if the Pareto front is non-convex, the solutions cannot be found using weighted aggregation.

A Note on Weighted–Sum Optimisation on Convex and Concave Fronts

This note is based on [YOS01], in which the authors offer an explanation of why the non-convex Pareto front poses problems for weighted-aggregation optimisation. In [DD97] the author also gives a similar explanation using geometry and the characteristics of tangents to convex and concave functions to explain why solutions on the concave regions of the Pareto front cannot be obtained by weighted aggregation. In [YOS01], Jin et. al explain why their Evolutionary Dynamic Weighted Aggregation (EDWA) overcomes the difficulties encountered with a Fixed Conventional Weighted Aggregation (CWA).

Definition 9. ***Convex Pareto Set***

A set of points S in the Euclidean space is convex if, given any two distinct points in the set,

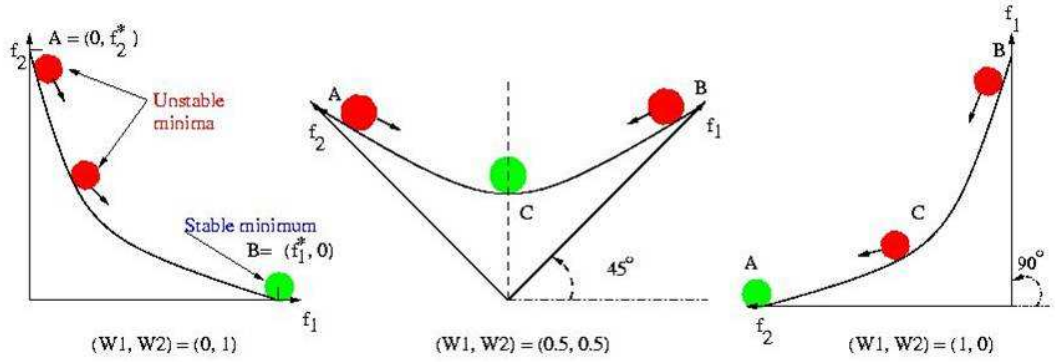


Figure 2.2: Convex Pareto Front. Each Point on the front is a stable minimum corresponding to a given weight combination (A rotation angle)

every point on the segment that links these two points lie *in S* [CS03].

Definition 10. ***Concave Pareto Set***
A set of points S in the Euclidean space is concave if, given any two distinct points in the set, every point on the segment linking these two points lie *outside S* [CS03].

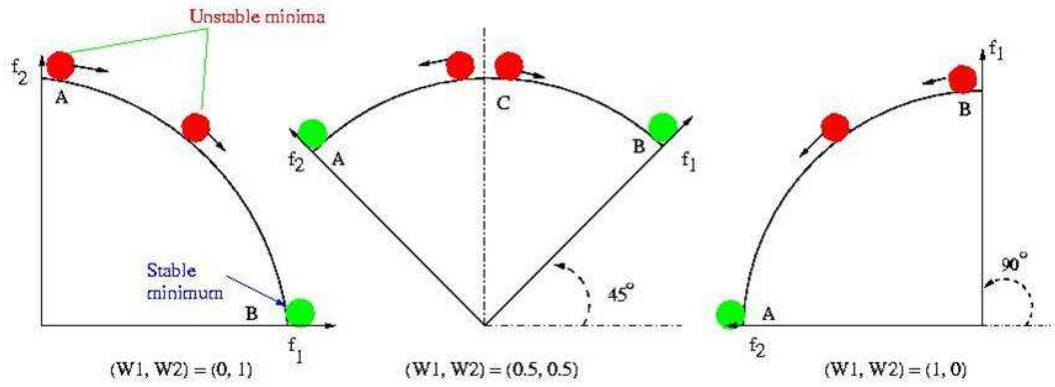


Figure 2.3: A concave Pareto front. Only the two points at the two ends of the front are stable minima

As the authors explain, the CWA is able to converge to a Pareto optimal solution if the Pareto solution corresponding to a given weight combination is a stable minimum. For the two-objective problem presented in Figure 2.2 (in the first graph to the left), point B is the stable minimum and will be the point that the algorithm converges to, given the weight combination $(w_1, w_2) = (0, 1)$. If the weight combination is changed, it is equivalent to rotating the coordinate system together with the Pareto front. Hence, when w_1 decreases and w_2 increases, it can be represented as the rotation of the coordinate system counter clockwise. For the weight combination $(w_1, w_2) = (0.5, 0.5)$, the coordinate system rotates 45 degrees, as shown in Figure 2.2 (middle graph), and the point C becomes the stable minimum. using the same principal, it becomes clear that in the case of a convex Pareto front, each weight combination corresponds to

a stable minimum on the Pareto front. Changing the weights will cause the optimiser to move from one stable minimum to another. As the weights are non-negative, the maximal rotation angle is 90 degrees at which the stable minimum is point A.

For a concave Pareto front, all solutions with exception of the two points on the two ends are unstable minima, see Figure 2.3. At a 0 degrees rotation angle, the stable minimum is point B. For all weight combinations corresponding to a rotation between 0 and 45 degrees, the solution obtained will be point B. Whereas for all weight combinations corresponding to rotations between 45 and 90 degrees, the solution will be point A. In the case of a weight combination of $(w_1, w_2) = (0.5, 0.5)$ corresponding to the rotation 45 degrees, the weight combination will be a dividing point, and the result of the optimisation will be either A or B. As a conclusion, the authors state that a weighted aggregation optimisation algorithm will only converge to either of the two points whatever weight combination is used.

To overcome some of the mentioned problems, an approach called "Dynamic Weighted Aggregation" is proposed in [YOS01] where weights are attached to objectives and are allowed to change dynamically during the evolution process. The authors use an evolutionary strategy algorithm, which proceeds as usual with fitness assignment and reproduction. However, the algorithm seeks to find the points of the Pareto front as it is going along, and the weights for the different objectives are set dynamically, then are gradually and periodically changed during optimisation. Whenever a new non-dominated solution is found, it is archived, and in this way, the whole Pareto front is archived. For a two-objective problem, the weights are defined as:

$$w_1(t) = |sin(\tfrac{2\pi t}{F})| \qquad \text{, and}$$
$$w_2(t) = 1 - w_1(t)$$

where t is the generation number and F is a constant that adjusts the frequency of changing the weights. Since the population will not be able to keep all the found Pareto solutions, an archive is used to keep the non-dominated solutions that are found. The method is tested on a range of multiobjective optimisation problems; with both convex and concave, as well as continuous and discontinuous, Pareto fronts .

This method overcomes some of the problems of fixed weighted aggregation. There is no need to decide on the weights a priori, we can obtain the Pareto solutions in one run and it is able to find solutions on the concave regions of the Pareto front as well as the convex. The reason for this, as the author explains, is that although on a concave front only two solutions corresponding to two weights are stable, all of the points of the front are reachable. He uses this fact, and does not force the algorithm to converge to one global minimum, but rather search for non-dominated solutions, which when found, are extracted from the population and added to the solution set.

Consequently, the actual technique for finding the Pareto optimal solutions is the archive maintenance and not the result of the evolutionary optimisation algorithm. The archive maintenance algorithm looks at each offspring generated and tests whether it is non-dominated by

members of the population as well as members of the archive. If this is the case, it adds it on to the archive. On the evolutionary algorithm side, and by continuously changing the weights and hence the fitness function, the EA is not being pressed to converge to any particular area but is merely used as a mechanism for traversing the space.

2.1.2.2 Population Approach

The Vector Evaluated Genetic Algorithm (VEGA) was proposed by Shaffer [Sch85]. The algorithm starts by randomly initialising a population as in standard GA, then divides it into a number of sub-populations equivalent to the number of objectives. Then, each sub-population's individuals are rated and selected according to one of the objectives, and hence, this particular subpopulation is pushed to evolve towards this one objective. Afterwards, the selected individuals are shuffled and allowed to reproduce as usual. The problem inherent in the algorithm was realised by Schaffer himself. He called it "middling". Since the algorithm looks at each sub-population with regard to only one objective and hence chooses individuals that excel in this specific criterion, we are deliberately ignoring individuals that may not be the best in any single criterion but offer a good compromise solution between all criteria.

In another population approach used in [KCV02], the Cooperative Co-evolutionary Algorithm [MAP95] was used to evolve subpopulations to solve the multiobjective optimisation problem. The approach proposed an integration between MOGA [FF93b] (see Section 2.2.2.5) and a Cooperative Co-evolutionary Genetic Algorithm (CCGA). Similar to the CCGA, each species in the MOCCGA represents a decision variable or a part of the problem that needs to be optimised. However, instead of directly assigning a fitness value to the individual of interest which participates in the construction of the complete solution, a rank value will be determined first. Similar to the MOGA, the rank of each individual will be obtained after comparing it with the remaining individuals from the same species on all the objectives. Then a fitness value can be interpolated onto the individual where a standard genetic algorithm can be applied within each sub-population. The algorithm was compared to MOGA, and the authors found that the number of non-dominated solutions found is higher and the coverage of the front is better in the case of MOCCGA.

Authors of [IL04] used a Cooperative Co-evolutionary Genetic Algorithm in which n–subpopulations are created randomly corresponding to the n decision variables to be optimised, and each is responsible for a decision variable x_i. In evaluation of individuals from the first generation, random individuals from the subpopulations collaborate and are evaluated on the objectives. The result of the evaluation is assigned as the fitness of the individual undergoing evaluation. In subsequent generations, an individual is evaluated by collaboration with a randomly selected component from the best non-domination levels in the previous generation subpopulation. Then both the parents and the children are sorted according to their domination information and the worst individuals are deleted, maintaining a constant population size. Tournament selection is used to select individuals for mating such that a solution i wins the

tournament against solution j if: it has a better domination level, or, in the case of both i and j having the same domination level, if i has a better crowding distance than j. The algorithm was found to produce comparable results to that of NSGAII (see section 2.2.2.4) on some problems.

Population approaches rely on the creation of subpopulations where each is responsible for optimising on of the objective functions, and they collaborate to achieve the required objectives. However, this class of algorithms can only deal with problems where distinct and separate components of a solution can be identified. Real world problems, where the variables are highly correlated, create a real difficulty for these algorithms and may cause its performance to deteriorate.

2.1.2.3 Pareto Optimality (PO) Approach

These algorithms are characterised by their use of the dominance concept to differentiate between solutions in the search space, and to eventually arrive at a set of solutions as close as possible to the Pareto optimal set [1].

The major advantage of the Pareto optimality approach is that it gives a large number of alternative solutions near the Pareto front in one single run. The assumption is that the decision maker will be given these solutions, out of which he will choose the most suitable. In Section 2.2, we review some of the existing multiobjective evolutionary algorithms that utilise Pareto optimality concepts.

2.1.2.4 Alternatives to Pareto Dominance

[DDB01, SDGD01] define the relationship "Preferred", in which a solution A is said to be preferred to B, if it is better than B in a larger number of objectives than B is better than A. This is in contrast to the relationship "Dominates", where A is said to dominate B if it is better than or equal to B in all objectives but is strictly better in at least one objective. The algorithm builds strongly-connected components in the relationship graph that result from the pair-wise comparison of all individuals of the population. All individuals in the same component obtain the same fitness. Then all components are hierarchically ordered followed by assignment of ranking values.

Kukkonen and Lampinen in [KL07] suggest ranking solutions according to each separate objective and the use of an aggregation function that results in a scalar fitness value for each solution to enable the sorting of solutions in the case of a large number of objectives. The aggregation functions suggested are Sum, and Minimum. The complexity of ranking is $O(MNlogN)$. Results of comparing this method to the Pareto dominance were reported on the test suite DTLZ 1-6 [DTLZ02]. The authors found that on problems (DTLZ 1-4) the new method advances the search. In other cases (DTLZ 5-6), it leads to deterioration of individual objectives if other objective values are improved, and the search does not proceed in the direction of the Pareto front. In DTLZ6, solutions are 'drifted' away from the front, by allowing most of the objective values to be slightly improved while one objective value worsens considerably.

[1] Refer to definitions in Section 2.1.1.

2.2 Multiobjective Evolutionary Algorithms (MOEA)

2.2.1 Evolutionary Computation Algorithms

Evolutionary Computation (EC) or Evolutionary Algorithms (EA) is a class of optimisation algorithms based on an analogy with concepts of natural evolution. They belong to the more general class of machine learning algorithms, which are defined as [Mit96]: "computer algorithms that improve automatically through experience". Algorithms belonging to this class are mostly used as optimisation algorithms that seek to arrive at an optimal solution through the artificial simulation of evolution. Using a method to evaluate the quality of randomly generated solutions, the progress towards the optimum is achieved by mimicking survival of the fittest, and rewarding those deemed fitter by favouring them for reproduction.

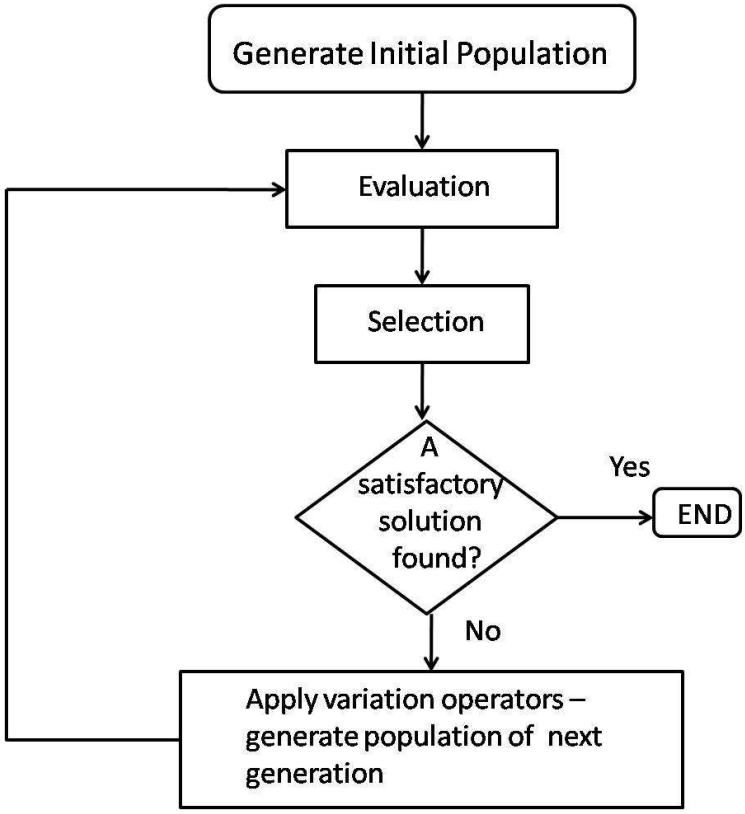


Figure 2.4: Basic Evolutionary Algorithms Life Cycle

To design an EA, the following components must be in place:

- Representation: EA solutions are represented as collections of "genes"– called "chromosomes". The most basic form of representation is as a linear chromosome. The chromosome is an encoding of the solution to the optimisation problem, and is also known as the *genotype*.

- A population of solutions – usually generated at random in the first instance.

- Quality indicator – Fitness function: designed to estimate how good a solution is in solving the given problem.
- Selection: the mechanism by which individuals from the population are selected to survive, and reproduce.
- Variation operations: as in nature, variation operations ensure exchange of genetic material between individuals (crossover), as well as the occasional changing of a random gene (mutation).

Different flavours of EAs exist, and each has been historically associated with a certain type of representation: Genetic Algorithms (GA): binary strings, Genetic Programming (GP): syntax trees, Evolutionary Strategies (ES): real-value vectors, and Evolutionary Programming (EP): finite state machines. However, these variations are mainly historical and varieties of representation within each EA exist. In the following section, we briefly introduce some background information on both GAs and GP. The life cycle of a typical EA algorithm is shown in Figure 2.4.

2.2.1.1 Genetic Algorithm

The GA was developed by John Holland in 1975 [Hol75]. The classic GA used a fixed length binary chromosome to encode solutions to the problem. Although the encoding of the solutions is problem specific and is up to the algorithm designer, the schemata theorem [Hol75, Gol89], which explains the mechanism by which the GA works, offers some guidelines on how to enhance the chromosome representation.

In the GA, the encoded chromosomes belong in the search space. To evaluate the fitness of the genotype, it is mapped onto the equivalent individual in the phenotypic space and is evaluated on the problem. Based on how well it solves the problems (achieves the objective), it is assigned a fitness value.

The variation operations used in the classic GA were crossover, mutation and sometimes copying of individuals as in the case when the reproduction operator or an elite survival mechanism[2] is used. Crossover is typically used more heavily than the other operators, especially mutation which is only used with a small probability. An example of the crossover operator is the one-point crossover, where a crossover point is chosen at random and the two parents exchange the part of the chromosome starting at the crossover point. Considering that the chromosome is a string of zeros and ones, the mutation operator simply flips the bit selected at random for mutation. Other varieties of both the crossover and mutation operators exist, especially as more complicated representations are being developed.

2.2.1.2 Genetic Programming

The term Genetic Programming was coined by Koza [Koz92] in which he suggested the use of a tree structure for the representation and automatic evolution of computer programs. He also

[2]When an elite mechanism is used, a certain number of the fittest individuals are copied to the next generation

demonstrated in his seminal book, the wide variety of problems for which the algorithm can be used. Designing a GP algorithm requires specification of the following:

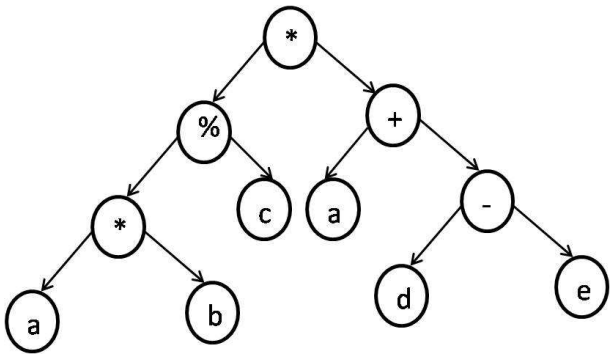


Figure 2.5: Sample Genetic Programming Tree Structure

- **Representation:** The GP uses a tree to represent its solutions (see Figure 2.5), the structure of the tree is not predefined, neither is its size, although sometimes a limit on the depth of the tree is imposed. GP trees consist of terminal and function nodes, where terminals provide input values to the system and function nodes process the input values and produce an output value. According to [BNKF98], the terminal set *consists of the inputs to the GP program, the constants supplied, and the zero argument functions*. They are often called *leaves* because they are located at the end of the tree branches. A leaf is a node that returns a numeric value, without itself having to take any input. [BNKF98] defines the function set as *the statements, operators, and functions available to the GP system*. This set encompasses the boolean, arithmetic, trigonometric, logarithmic functions, conditional statements, assignment statements, loop statements, control transfer statement, et cetera.

  The focus of the encoding issue in the GP shifts from the design of the structure to the choice of the suitable terminal and function sets. Choosing a very large function and terminal sets complicates the search, while very small sets will be restrictive and may not allow for the evolution of appropriate solutions.

- **Population:** The population is initialised by generating random tree structures to fill the population. The trees are built from the terminal and function sets (except for the root node which can only be selected from the function set) such that the tree depth does not exceed the maximum allowed depth. There are three commonly used methods for building the trees. These are *grow, full, and ramped half-and-half* [Hol75, BNKF98]:

  1. *The Grow method:* In this method, nodes are selected randomly from the function and terminal set. Once a terminal node is added to a branch, this branch terminates whether or not the maximum depth has been reached. The tree structures in this method are often of irregular shape.

2. *The Full method:* Nodes are selected only from the function set until the node is at maximum depth, at which point they are selected from the terminal set. This method results in fully filled trees with all branches at the maximum depth.

3. *The Ramped Half-and-Half method:* Devised to enhance the diversity of the initial population. This method works as follows: Suppose the maximum allowed depth is 5, then the population is divided equally among individuals to be built with trees of depth: 2, 3, 4, and 5. For each depth group, half the trees are initialised with the full method and the other half are initialised with the grow method.

- **Operators:** There are many genetic operators that have been developed over the years. The three main GP operators are:

  1. *Crossover:* The crossover operator swaps the genetic material of two parents in an attempt to propagate successful genetic material, and at the same time, vary the successful chromosomes so that the algorithm continues its exploration in the search space. One method for tree crossover would proceed by the following steps [BNKF98]:

     – Select two parents based on the selection scheme used.

     – Select a random subtree at each parent.

     – Swap the selected subtrees. The resulting individuals are the children and are placed in the population of the next generation.

  2. *Mutation:* Operates on one individual only. Many types of mutations exist. In one of them, a subtree is selected at random, removed from the individual and replaced by a randomly generated tree, following the same method for building a tree and adhering to the depth constraint.

  3. *Reproduction:* The individual is copied and placed in the population.

- **Fitness:** The fitness function is a very important part of the GP as it is in all other evolutionary algorithms. It measures the success of an individual on the problem and in effect assigns its probability of reproducing and its genetic material surviving to the next generation. As the generations evolve, the average fitness of the population is expected to improve, which is a sign that the algorithm is learning.

- **Selection:** After the fitness of individuals has been determined, we need to decide which individuals will be allowed to propagate their genetic material, which will be kept in the population and which will be replaced. Some of the commonly used selection operators are [BNKF98]:

  1. *Fitness-Proportionate Selection:* Employed in the classical generational GA, where the probability of an individual producing offspring is proportionate to the ratio of its

fitness to the average fitness of the population. This method is usually criticised for using an absolute measure for fitness.

2. *Truncation Selection:* Known in the ES community as (μ, λ), where a number of μ parents are allowed to produce λ offspring, out of which the best μ are used as parents in the next generation. This method is not dependent on the absolute fitness values, as the μ best will always be the best, regardless of the absolute fitness differences between individuals.

3. *Ranking Selection:* Individuals are sorted according to their fitness values, and given ranks. The selection probability is assigned as a function of their rank in the population.

4. *Tournament Selection:* Based on competition with a smaller subset of the candidate parents rather than the full population. A number of individuals, called the tournament size, is selected randomly, then the best among those individuals is selected. The resulting offspring (or the mutated version of the individual) then replaces the worst individuals in the population. The tournament size plays a role in adjusting the selection pressure. A small tournament causes low pressure and a higher tournament sizes increase the selection pressure. The advantage of this method is that it gets rid of the centralised fitness comparisons among all individuals that have to be carried out in the other three methods, resulting in an acceleration of the selection process.

2.2.2 State of the Art Multiobjective Evolutionary Algorithms

Multi Objective Evolutionary Algorithms (MOEA) integrate the Pareto dominance concepts into the framework of the Evolutionary Algorithms (EAs). MOEAs are distinguished from standard EAs by employing the Pareto dominance concept in the fitness evaluation to allow for the comparison between individuals based on multiple conflicting objectives. Instead of producing one best solution, they produce a Pareto front of many solutions to the problem in one run. If the MOEA algorithm is successful, the solution set should be as close to the true Pareto front as possible, has a wide coverage of the front and be diverse enough to represent useful tradeoffs of the objectives [Coe05b].

MOEAs have received considerable attention in the last decade. They have been applied to a wide variety of application problems whose optimisation was characterised by the need for the simultaneous optimisation of conflicting objectives. The cycle of an MOEA, is the same as that of the EA, except when it comes to evaluating the fitness of individuals. In the following we present a review of some of the early MOEAs (MOGA,1993; NPGA, 1994; NSGA, 1994; PAES, 1999; and SPEA, 1999), as well as some of the more recent algorithms that were most prominently used in financial applications (PESA, 2000; NSGAII, 2001; and SPEAII, 2002). For a thorough review of these algorithms and others, the reader is referred to [Coe05a, TGC07].

Some of the most recent MOEA algorithms include: OMOPSO [RC05]; MOEA/D [ZL07, LZ09]; and SMPSO [NDG+09].

2.2.2.1 SPEA

This algorithm [ZT99] uses an elitism mechanism that employs an external archive for saving (preserving) the non-dominated solutions. The algorithm starts with a random population and an empty archive with a maximum predefined size. In each generation, all non dominated individuals are copied to the archive, which is then tested for dominated individuals, which are deleted if found. If the size of the archive exceeds the predefined limit, an agglomerative clustering technique based on phenotypic distance is used to delete some of the non-dominated individuals while preserving the diversity characteristics. Each member of the archive has a strength value $S(i) \in [0, 1]$. It is defined to be the number of the j population members which are dominated by or equal to i, divided by the population size plus one. The Fitness $F(j)$ of an individual j in the population is calculated by summing the strength values $S(i)$ of all archive members i that dominate or are equal to j, and adding one. In the reproduction phase, the current population and the archived population are mixed together to form one population from which parents are selected. Since the fitness values of the archive solutions are in the range of $[0, 1]$ and minimisation of fitness is sought, the archive members have more chance of being selected. The complexity of the algorithm was found to be $O(mN^2)$, where m is the number of objectives, and N is the size of the population. Although the algorithm is very successful in comparison with other algorithms, some weaknesses have been identified by Zitzler [ZLT02] as well as others. First, the fitness assignment in the population is based solely on the number of dominating individuals in the archive. This technique is not able to reflect information about dominance regarding members of the population itself, and the selection pressure can decrease significantly. Second, the clustering technique is used in the archive to maintain diversity; however, no technique is being used to maintain diversity in the population. Finally, in spite of the effectiveness of the clustering technique, it may lose outer solutions, even though they are still non dominating solutions that may be part of the Pareto front.

2.2.2.2 SPEA2

Designed by Zitzler, Laumanns and Thiele to overcome some weaknesses in the SPEA algorithm[ZLT02]. The SPEA2 algorithm also has an archive with a predefined size. The differences from SPEA are: (1) the archive in SPEA2 has a fixed size; if the number of non dominated individuals is less than the predefined size, the archive is filled with the best dominated solutions from the population. On the other hand, if the archive is greater than the defined size, a truncation method is used instead of the clustering technique. (2) In the truncation method, individuals chosen for removal are those that have the minimum distance to another individual. If several individuals have the same distance, the second smallest distance is considered. It was found that the truncation technique is better than clustering with respect to retaining the boundary points. (3) The fitness assignment in this algorithm is defined to take into account

both dominated and dominating solutions; each individual in both the archive and the population is assigned a strength value representing the number of solutions it dominates. On the basis of the strength value, the raw fitness value is calculated as the sum of the strengths of the individual dominators in both the archive and the population. Raw fitness is to be minimised; accordingly, non-dominated individuals will have a raw fitness value of 0. (4) In addition, density information is added to the fitness function by calculating the density as the inverse of the distance to the k-th nearest neighbour[3]. This is used as a mechanism to further discriminate between individuals. The fitness of an individual is the sum of its raw fitness and density information. The complexity of the algorithm is $O(M^2)$ where $M = N + N^{'}$, and $N^{'}$ is the archive size and N is the population size. (5) Only members of the archive participate in the mating selection process.

The SPEA2 was compared to SPEA as well as NSGAII. It was found that SPEA2 had a better distribution of points especially when the number of objectives increased, probably because of its maintenance of some dominated individuals which helped to maintain diversity.

2.2.2.3 NSGA

This algorithm was proposed in 1994 by Srinivas and Deb [SD94]. The population is ranked using Pareto dominance. The non-dominated individuals found are classified into one category and assigned a dummy fitness value proportional to the population size (the highest). This category is then excluded from the population, and another search for non-dominated individuals is conducted, until all of the population is ranked. To maintain diversity, individuals in each non-domination level are shared with their dummy fitness value.

Sharing fitness method: given a set of n_k solutions in the k-th non-dominated front, each having a dummy fitness value f_k, the sharing procedure is performed in the following way for each solution $i = 1, 2, \ldots, n_k$:

- Step 1: Compute a normalised Euclidean distance d_{ij} measure with another solution j.
- Step 2: The distance d_{ij} is compared with a pre-specified parameter α_{share} and the following sharing function value is computed as:

$$sh(d_{ij}) = \begin{cases} 1 - \left(\frac{d_{ij}}{\alpha_{share}}\right)^2 & if d_{ij} < \alpha_{share} \\ 0 & otherwise \end{cases} \tag{2.4}$$

- Step 3: Repeat for all $j \leq n_k$
- Step 4: Calculate niche count for the i-th solution as: $m_i = \sum_{j=1}^{n_k} sh(d_{ij})$
- Step 5: The shared fitness value $f_k^{'}$ of the i-th solution is: $f_k^{'} = \frac{f_k}{m_i}$, where f_k is the solution's fitness value.

[3]Where k is usually taken to be the square root of the sample size (population plus archive), and the distance is calculated so it will be a value greater than zero and less than one.

Since individuals in each category obtain a fitness value proportional to their Pareto ranking (with the first front getting the highest fitness value), their selection probability is proportional to the number of individuals they dominate. The algorithm is computationally expensive as it requires the comparison of each individual to every other individual in the population. In the worst case, where each front contains exactly one solution, the complexity of the algorithm is $O(mN^3)$. The algorithm also requires the user to decide on the fitness sharing factor, for which the performance of the algorithm has been shown to be very sensitive.

2.2.2.4 NSGAII

This algorithm was introduced by Deb et al. in 2001 [Deb01] as an improvement to the NSGA algorithm . They tried to overcome the main problem for which NSGA was criticised; its computational complexity. The new algorithm consists of two main loops. In the first, for each solution i , two entities are computed: 1) The number of solutions which dominates it, denoted n_i, and 2) The number of solutions dominated by it, denoted S_i. The calculation of these two entities would require $O(mN^2)$ comparisons. Next, the solutions with $n_i = 0$ are identified and are put in a separate list F_1, which is called the current front. In the second loop, the current front is traversed, and for every solution on it, each member j on its S_i list is visited and we decrement its own n_j count by one. If the new decremented count becomes 0, it is put in a separate list H. When all the members of the current front have been checked, the list F_1 is declared as the first front, and H becomes the current front. The process is repeated with H as the current front. Each iteration will require $O(N)$ computation. Since there are at most N fronts, then the worst case complexity of the second loop in $O(N^2)$ and the overall complexity of the algorithm is $O(N^2) + O(N^2) = O(N^2)$. The fitness assignment, selection and reproduction proceed as in the NSGA. The diversity is maintained by using a crowding measure in the selection and reproduction phase. The crowding measure is a density estimation technique in which for each solution, the density of solutions surrounding it is measured by the average distance of the two points on either side of that point along each of the objectives. This measure effectively calculates the largest cuboid enclosing a solution in the objective space without including any other solution.

2.2.2.5 MOGA

Fonseca and Fleming, 1993 [FF93b] presented a scheme in which each individual is given a rank proportional to the number of individuals in the population by which it is dominated. The non-dominated individuals are hence assigned the lowest rank of 1. The population is then sorted according to the rank. The fitness is assigned by interpolating from the best rank to the worst (Pareto ranking assignment). Then, the individuals with the same rank have their fitness averaged so they will be sampled at the same rate in the population. The algorithm requires a decision on a value for the fitness sharing factor.

2.2.2.6 NPGA

Proposed by Horn in [HNG94]. NPGA uses a tournament selection scheme based on Pareto dominance. However, instead of comparing two individuals to decide whether one of them dominates the other, the two individuals are compared to a random sample of the population. If one of them is non-dominated and the other is dominated with respect to the sample selected, then the non-dominated individual is returned. If both are dominated or non-dominated by the sample, then, the result of the tournament is decided through fitness sharing in the objective domain. The approach is faster than the previous ones, because it decides the Pareto ranking based on a segment of the population. However, it requires a decision on a fitness sharing factor, as well as the size of the segment of the population that we compare the individual against.

2.2.2.7 PAES

Introduced by Knowles and Corne [KCO99]. PAES uses an evolutionary strategy with one parent producing one child using mutation. The algorithm also uses an archive[4] that keeps all the non-dominated solutions found. When a child is produced, the parent and the child are compared, and if the child dominates the parent, it is accepted. Else, it is discarded and a new (mutated) child is sought. However, if the child and the parent do not dominate each other, then the one that maintains the diversity among the obtained solutions is preferred. This is done by comparing the child to the non-dominated archive to check whether it dominates any member. If it does, it is accepted into the new population and the dominated solution is discarded from the archive. If it does not dominate any of the solutions in the archive, then, both the parent and the child are checked to find which one resides in the least crowded region of the parameter space among members of the archive and this is the one selected. The diversity maintenance approach used was different, in which the objective space is divided up like a grid, and each solution is placed on the grid based on its objective values. A crowding measure based on the number of solutions in and around each grid location was used to maintain the diversity. This had the advantage of no extra parameters to be specified, and the method has a lower computational complexity of $O(mN^2)$. That is because every member of the population is compared to its parent as well as to all members of the archive.

2.2.2.8 PESA

The Pareto Envelope-Based Selection Algorithm (PESA) incorporates ideas from both PAES and SPEA and was proposed by Corne et al. [CKO00]. It's main feature is the use of the "hyper-grid scheme" technique, for both selection and diversity maintenance. The algorithm uses a small internal population and a larger external population (archive) which stores the current approximation to the Pareto Front. The hyper-grid division of phenotype space allows it to keep track of the degree of crowding in different regions of the archive. In PESA, selection is based on this crowding measure. In addition to crossover, and mutation rate, the algorithm

[4]The archive is usually chosen of a size equivalent to the original population size.

has three extra parameters, the population size, the maximum archive size and the crowding squeeze factor, which is the number of other solutions inhabiting the same box in the hyper-grid of the phenotype space.

PESA starts by randomly generating and evaluating the initial internal population. The non-dominated solutions found are added one by one to the archive. If the added solution dominates any of those already in the archive, the dominated solutions are removed from the archive. If at any point the maximum size of the archive is exceeded, then a member from the archive is removed. The decision of which member to be removed is made by finding the maximal squeeze factor in the population and removing an arbitrary solution which has this squeeze factor. If the stop criterion is not reached, then, the contents of the internal population are deleted, and the following is repeated:

- Select two parents with probability p_c from archive, produce a single child and mutate it.
- Select one of the parents with probability $1 - p_c$ and mutate it to produce a child.

Until a certain number of new members have been generated.

The normalised phenotype space is divided into a hyper grid that divides the space into hyper boxes. In problems with two objectives the hyper boxes are squares. Each chromosome in the archive is associated with a certain hyper box, and the number of other chromosomes that inhabit the same hyper box is the chromosome's current squeeze factor. For selection of parents using tournament selection, the parents are selected at random from the population and entered into the tournament, and the parent winning the tournament is that with the lowest squeeze factor.

The authors compared their algorithm to PAES and SPEA on six experiments with varying time limits (number of generations) to analyse suitability of the algorithms to problems with varying needs for solution speed. The comparison was based on the percentage of the Pareto front that was found by the algorithm with 95%confidence. When solutions were needed quickly, PESA outperformed the other two algorithms on five out of the six problems. Given more time, PAES and SPEA start to pick up; with PESA being the best on two problems, joint best with PAES on another two and SPEA and PAES were each superior in one. Increasing the time again, PESA was best or joint best in five test problems.

2.2.3 Issues in Multiobjective Evolutionary Algorithms

2.2.3.1 Maintaining Diversity and Uniform Distribution of Solutions

Due to the fact that a solution set spread along the Pareto front is required of the multiobjective algorithm, techniques for diversity maintenance receive special attention in MOEAs. Niching or crowding are often employed, but other techniques also exist. In [KL07], the authors propose the use of an alternative to the Pareto dominance ranking. They compare the new method to the Pareto dominance in terms of diversity of solutions on the Pareto front. They first noticed that

as the number of objectives increase, the selection based on Pareto dominance without diversity maintenance techniques performs better than that selection with diversity maintenance. The explanation was that as the number of objectives increase, the diversity maintenance becomes the dominating factor differentiating between solutions due to the failure of ranking based on dominance, and hence, preventing progression of the search. However, diversity is also lost using the ranking-dominance to sort the solutions, with some problems converging to a front with very few points. The authors conclude that with an increasing number of objectives, the need for a balance between convergence and diversity maintenance is critical. They suggest increasing the diversity maintenance as the search progresses (as measured by the generation number), and in the case of the ranking dominance they add to that the suggestion of using different aggregation functions, such as a linear or a power function for the rate of increase.

2.2.3.2 Scaling Issues in Problems with a High Number of Objectives

Deb in [Deb01] reports that the number of non-dominated individuals increases with the number of objectives to the extent that in experiments with 20 objectives, a randomly generated population will have 100% of its members belonging to the non-dominated set using the Pareto dominance criterion. Using the weighted aggregation method will also be problematic, as with the increasing number of objectives, specifying the weights becomes very hard.

In [SDD07], the nurse rostering problem, where a schedule for employees in a hospital is required, is considered. The problem is highly complex with hard and soft constraints and the number of objectives to be *minimised* is 25. The researchers compared the performance of the relationship "Preferred", previously defined in [SDGD01], against the dominance relationship as represented by two algorithms; one based on "Dominates", and the NSGAII. The algorithm based on "Dominates" relationship counts for each individual the number of individuals that dominates it. If the number is zero, then this individual is in the Pareto-front, and is given the best rating. Then the elements with one dominator follow and so on. Thus, in contrast to non-dominated sorting in NSGAII, only the first Pareto-front is built and considered in the algorithm Dominates[5]. The performance metric was the average value of the one individual with the best weighted-aggregation[6] of the 25 objective values over the 10 runs. With such a high number of objectives, the algorithms based on dominance relationship achieved a $98 - 100\%$ on the performance metric. On the other hand, the relationship Preferred achieved a reduction of more than 50%[7]. Hence, the relationship preferred has more power to differentially rank individuals in high dimensional objective space. However, taking the standard deviation of each of the ten runs in relation to the average reveals that the preferred method has the highest standard deviation of 67% in comparison to Dominates 11% and NSGAII 13%, which makes it

[5]Note, that the distribution of the elements in the solution space is not taken into account. Hence, the Pareto front may not have a good coverage of the different trade-offs.

[6]The authors of [SDD07] note that a weights values used in the metric function resulted from the experience of an expert, and that a lot of time was necessary to adapt those weights.

[7]Values given as percentages, normalized for Dominates.

less stable than the Dominates relationship as it is more sensitive to the random initial seed.

2.2.3.3 Robustness of MOEAs

Robustness and behaviour of MOEAs in dynamic environments is the main topic of this work. Review of the current research carried out in this area is presented separately in Section 2.4.

2.3 MOEA in Computational Finance

Both single-objective and multi-objective evolutionary algorithms(SOEAs and MOEAs) have been used in a variety of problems in the financial domain: Portfolio Optimisation and Stock Selection (see Section 2.3.1), Pricing Derivatives (for example [FBOO07, MTES01]), Management of Financial Risk (for example see [MBDM02, SMS05]), Forecasting and Time Series Prediction (examples include: [aAD09, PTP05, ST01]), Evolving Technical Rules for Trading and Investment (Section 2.3.3), and Decision Making (for example [TLM+00]) . In addition, the various flavours of EAs (GA, GP, ES) has been investigated for such studies. We focus mostly here on research that has used multiobjective EAs.

From the spectrum of financial problems, we focus in this chapter on research carried out in the areas of: portfolio optimisation; stock ranking and selection; and trading in the stock market. A good literature review of multiobjective optimisation applied in a wider area of finance problems is found in [TGC07], and [TMJ04].

2.3.1 Portfolio Optimisation

The portfolio optimisation problem is the allocation of a limited capital to buy certain quantities of various assets. The decision of which assets to include and their quantities will depend on a number of quantitative measures, typically the maximisation of return and minimisation of risk. An optimal portfolio is one that has the maximum possible return given a certain risk or the minimum possible risk given a certain return. These optimal portfolios will give what is known as the efficient risk-return frontier. A more detailed explanation of the problem is provided in Chapter 4.

One of the early studies in Portfolio optimisation using MOEAs is that of Lin et al. [LD01] in which fixed transaction costs[8] and minimum transaction lots [9] are adopted. They used NSGAII [DPAM00] to construct a Pareto front of feasible portfolios that optimise two objectives: return and risk, where risk is expressed as the portfolio variance. The constraints imposed on the portfolio were: a maximum amount of total invested capital; short selling and borrowing were not allowed; and maximum amount of capital is imposed on each security. The authors note that with such a model, finding a feasible solution is an NP-complete problem, and hence it is significant to find some heuristic to solve the problem. Results were shown in training, with the evolved Pareto fronts of the trade-off between risk and return. The results confirmed the ability of the NSGAII to find feasible solutions in a reasonable time (average of 50 generations).

[8] Costs deducted from the portfolio return and correspond to fees associated with a transaction on the stock exchange.
[9] Where assets must be acquired in multiples of minimum lots.

A hybrid system of a multiobjective GA and linear programming was used in [SBE+05] to maximise several return measures and minimise several risk measures, where the measures might be non-linear and non-convex. The multiple objectives were: Book Yield for Portfolio return; Variance and Value at Risk for the risk. They used the multiobjective algorithm PAES that is initialised with a Randomised Linear Programming (RLP) which identifies boundaries of the search space by solving thousands of randomised linear programs. However, the aim of the research was mainly to present the architectural design of such a system and a graphical design tool to present the Pareto front, not to measure the performance of the algorithm.

Some researchers [AM10, AL05, Lau05, SKN07] were interested in comparing the performance of various flavours of MOEAs on the Portfolio optimisation problem. In [AL05], the authors compared a greedy search[10], simulated annealing and an ant colony approach, all adapted to the multiobjective context. The portfolio problem considered had a constraint on the maximum number of assets to include, and enforced a maximum and minimum holding allowed for each asset. The two objectives considered were return maximisation and risk minimisation. The results are reported in training with investments in 5 different stock indexes: Hong Kong's *Hang Seng*; the German *DAX100*; the British *FTSE100*; the U.S *S&P100*; and the Japanese *Nikkei225* from the OR-Library[11] [J.E90]. The simulated annealing and ant colony algorithm had the best performance, with no clear winner between them. They investigated varying the number of assets in the portfolio, with the results proving that diversification leads to a decrease in the total risk of the portfolio. In the case of just two assets, the algorithms select the two assets with the highest return and afterwards, they try to fit the risks in the best way, making the risk of the portfolio significantly high. [Lau05] compared the PESA, NSGAII and SPEA2 algorithms[12] on a portfolio optimisation problem with real world data of the Euronext Stock exchange. Again, the algorithms' performance was only compared in training using the S-Metric [ZT98] (the size of dominated space) and the Δ-Metric [DPAM00] (how evenly spread the points are on the Pareto front). The results showed that PESA outperformed the other two algorithms in terms of the S-Metric, and NSGAII had the best values in terms of the Δ-Metric. However, in the study of [SKN07] where the authors compared five GA[13] based multiobjective algorithms: VEGA; a fuzzy VEGA; MOGA; NSGAII; and SPEA2, a different result was obtained. The fuzzy VEGA was developed to overcome the tendency of VEGA to converge towards one objective best solution. The authors incorporate a fuzzy decision rule to combine the optimisation of the two objectives together that dictates the selection of each individual. The

[10] In a portfolio problem, the distance between two portfolios is not clearly defined. Hence, the authors define an algorithm for generating portfolios in a neighbourhood.

[11] http://people.brunel.ac.uk/mastjjb/jeb/orlib/portinfo.html

[12] All algorithms had 100 population size, 0.8 crossover rate, 0.05 mutation rate and evolved for 100 generations. The underlying EA was the GA

[13] The GA chromosome representing an individual portfolio was a pair of a binary and a real strings; with the binary string representing which stocks are included in a portfolio, and the real string representing weights of each stock in the portfolio

data set was that of Hong Kong's *Hang Seng* index from the OR-Library [J.E90] with 31 stocks. Two experiments were run with a cardinality constraint of 5 and 10 respectively. Performance was measured using the Generational Distance (GD) [ZLT02], given by: $GD = \sqrt{(\frac{1}{n}\sum_{i=1}^{n} d_i^2)}$, where d_i is the distance between the evolved Pareto front and the true Pareto front (provided by the OR-Library).

The results of the experiments showed SPEA2 to have the best results in terms of the GD metric and the distribution of points along the front (inspected visually). The results of the fuzzy VEGA were better than those of the plain VEGA and closer to the performance of MOGA. The authors conclude that, in general, the Pareto selection algorithms outperform the vector selection algorithms (represented by VEGA although the fuzzy selection improves the performance). Additionally, among the Pareto selection algorithms, SPEA2 has the best performance in the portfolio optimisation problem with realistic constraints[14]. The recent research of [AM10] experimented with three multiobjective algorithms: NSGAII, PESA, and SPEA2, to find tradeoffs between risk, return and the number of securities in the portfolio. By introducing a third objective, the efficient frontier becomes a surface in the three dimensional space. Visual comparisons of the results have shown that SPEA2 was the best algorithm with regard to the hypervolume metric. PESA was second with results that are very close to that of SPEA2. In terms of diversity of solutions, PESA and SPEA2 had the best results. PESA was the fastest technique, while SPEA2 was the slowest. Results were compared in training only and based on the comparison of the Pareto sets evolved using the three techniques.

Chiam et al. [CML07] focuses developing a GA chromosome representation suitable for handling the Portfolio optimisation constraints. The researchers used an order-based GA, and investigated the effect of the various constraints on performance. They considered the floor and ceiling constraints, and a general cardinality constraint[15]. The researchers propose an extension to the order-based GA representation to handle such constraints. A portfolio is represented by two strings; one containing the identity tags of stocks in the portfolio and the other containing the assets' weights. To find the portfolio associated with a chromosome, an empty portfolio is initialised and assets are added as per the order specified in the chromosome. The procedure will terminate once the total weight of the portfolio exceeds its maximum possible weight or when all the available assets are included. The weights are then normalised to a random value between 1 and N (the number of assets), and are also adjusted to meet the floor and ceiling constraints. The cardinality constraint is enforced using a repair algorithm that checks the individual portfolio and repairs it to comply with the cardinality restriction. The performance of the Pareto fronts evolved is measured using the S-Metric, the Δ-Metric and the GD Metric

[14] Round-lot, cardinality, and floor (lower limit on the proportion of each asset) constraints were considered. The constraints were enforced using a repair algorithm which ensures that randomly generated and crossed-over chromosomes comply with the constraints.

[15] The general cardinality constraint restricts the number of assets to be between a minimum and a maximum umber of assets, and not strictly equal to a predefined number of assets.

on the five indexes of the OR-Library. The authors use a generic MOEA that used elitism and Pareto-based dominance for selection. They used the averages results of 30 runs in addition to using the ANOVA [MBB99] test to examine the significance of the mean differences. The effect of the floor and ceiling constraint is observed in one of the problems by considering two different versions where in one the constraints were [1%, 2%] and the other [10%, 11%], it was shown that with this constraint it was not possible to approximate the entire Pareto front, and that the fewer the limits, the smaller the front found. This is because the constraint will limit the portfolio size and in effect indirectly influence the level of risk and return possible, thus only a certain region of the frontier is achievable. The effect of the cardinality constraint was already highlighted by Chang et al. [CMB+98] which is that the Pareto front achieved will be discontinuous as some portfolios will not be available to the rational investor. The results found by Chiam et al. here support this and show a discontinuous front that improves as the cardinality constraint is relaxed. With a low cardinality, the low risk-return region is under-defined since large portfolios are not allowed, and as the cardinality increases to be between [15, 20], a wider spread front is generated. However, increasing the cardinality to be between [25, 30] generates a front where no portfolios exist in the high risk-return region, and with a further increase to [50, 55], only a suboptimal Pareto front in the low risk-return region is found.

2.3.2 Stock Ranking and Selection

Stock ranking and selection is at the heart of the portfolio optimisation problem. However, stock selection can also be considered apart from the portfolio by focusing on selecting individual stocks and examining the algorithm that is able to select stocks to satisfy the objective specified.

Mullei et al. [MB98] used a GA with a linear combination of weights to select rules for a classifier system to rank stocks. Up to nine objectives were considered and the system was validated using 5 large historical US stock data sets covering 3 years of weekly data with a universe of 16 stocks. Results were compared to a polynomial network, but they were inconclusive since no technique was able to beat the other in all cases. In [BFF07], the authors used an MOGP for constructing multifactor models for stock ranking. They implemented an MOGP that simultaneously optimises : information ratio (IR)[16], information coefficient (IC)[17], and intra–fractile hit ratio of the portfolio[18]. This work was an extension to a previous work by the same authors [BFL06] where they used a single objective GP and in which they found that the GP rules were able to outperform rules generated using a linear multi-variable regression model. However, the GP rules did not generalise consistently well to out-of-sample data and the results were unbalanced in satisfying the multiple objectives: formulas trained to

[16]Information ratio is defined as the annualised average return of the portfolios constructed divided by their annualised standard deviation.

[17]Information coefficient is calculated as the Spearman rank correlation between a formula's predicted stock ranking and the actual empirical ranking of the stocks' returns.

[18]The number of the top ranked stocks that actually performed better than the average plus the number of bottom ranked stocks that performed worse than average divided by the total number of stocks in the top and bottom percentile.

maximise the information ratio had a disappointing information coefficient and vice versa. In the multiobjective study, the authors did not use any of the Pareto dominance multiobjective algorithms, instead they used three different methods to combine the objectives. They compared the performance of the three multiobjective algorithms and found that they produced more robust results in terms of over-fitting compared to the single objective GP. They found that one -*the constrained fitness function*- [19] had the best generalisation performance compared to the other two multiobjective algorithms. However, the authors realise that the parameters used in the constrained fitness function were hand tuned from a deep familiarity with the stock market examined and they list for future work examining a Pareto ranking algorithm such as the SPEA2.

2.3.3 Evolving Trading Rules

Allen and Karjaleinen (AK) study [AK99] is considered by many the pilot study in the area of using EAs to evolve trading rules. In this work, the authors used genetic programming to find *technical trading rules* for the S&P index using daily prices from 1928 to 1995. They found that although the rules were able to find periods to be in the market when returns were positive with low volatility and out of the market when the opposite was true, compared to a simple buy-and-hold strategy, the trading rules did not earn consistent excess returns after transaction costs of 0.0025. The fitness function in the AK study was the excess return over the buy-and-hold strategy, and the authors made use of a selection period to minimise over-fitting. Their study of the rules evolved indicated that many had trading patterns similar to those of the moving average rules. Neely [Nee99] extended the AK study by investigating the use of a GP to evolve risk-adjusted technical trading rules. Neely found an improvement in the results but still found no evidence that technical trading rules identified by a GP significantly outperform buy-and-hold on a risk-adjusted basis. Contrary to the previous two results, Becker and Seshadri (BS) [AS03] found that technical trading rules evolved using GP were able to outperform buy-and-hold even after transaction costs. This study had a number of significant changes from the previous two studies: it used monthly data instead of daily data; reduced the operator set and added more technical indicators to the terminal set [20]; used a complexity–penalising factor in the fitness function; and utilised a fitness function that took into account the number of periods in which the rule performed well and not just the total average excess return. The transaction cost used was even higher than that of the previous two studies at 0.005. The results of two experiments [21] found that on average the GP was able to find rules that outperform the buy-and-hold and that the difference is highly significant. In a recent publication [LC09], the authors endeavoured to closely examine the reasons behind the contradicting results of AK

[19] Which maximises the IR objective subject to the two other objectives being above certain threshold

[20] The authors rationalise this practice as being a way of adding domain knowledge, bias the search towards commonly used derived technical rules and make the evolved rules more comprehensible

[21] Where in the first, the fitness function used a complexity–penalising factor, and in the second the fitness function used the consistency-of-returns over training periods.

and Neely on one side and BS on the other. They used the same fitness function as that used by Becker with the addition of the complexity–penalizing factor and the consistency-of-returns. The transaction cost was 0.0025, crossover probability 0.7 and mutation probability 0.3. They used 31 years of S&P monthly data for training and a variety of selection and validation periods for different experiments. They investigated the use of two regimes: in the first, a period of validation directly follows training and in the second, validation is conducted using the rule which, after training, performed best on a selection period. Their results indicated sensitivity to the data periods selected for (training, selection, validation) and that the use of a selection data set was beneficial in most cases. Suspecting that the reason behind the contradiction in results found in their study and BS's in comparison to AK and Neely was the use of monthly instead of daily data, the authors proceeded with their research to investigate whether similar results could be obtained using weekly and daily data. The results in [LC10] confirm that finding effective trading rules is more difficult using the daily data and success is somehow 'in between' in the case of weekly data. The authors note the high dependency of the results on the data splits used, however they acknowledge that identifying a correlation between the characteristics of the data sets and the success of the evolution is not straightforward.

In another recent study, [CTM09] used an MOEA to evolve technical trading rules to satisfy the two objectives of risk and return. The risk was defined by the trader's exposure to loss; specifically by the proportion of trading days that an open position is maintained in the market. The trading rules are modelled as a variable length chromosome of a set of decision thresholds and technical indicators with different weights and parameters. The trading decision at every time step is the weighted average of the decision signal from the various technical indicators (TI). Trading costs are fixed at 0.5% of every complete trade. The research concentrated on comparing the trading behaviour of various individual technical indicators and hybrids of the technical indicators. The authors found the technical indicator composition along the risk return frontier revealed that each TI has varying degrees of significance in different regions of the trade-off surface. When examining the generalisation characteristics of the algorithm, the authors found what they described as low correlation between high returns in the training data and the test data. Instead they observed that higher returns in training correspond to larger volatility in the returns generated in the test data.

In [BJ08], Briza et al. used Particle Swarm Optimisation (PSO)[22] [KE95] for stock trading in a multiobjective framework (MOPSO). Their system used historical end-of-day market data and utilised the trading signal from a set of financial indicators to develop trading rules that optimise the objective functions of percent profit and Sharpe ratio [MAL03][23]. They divided their data set into 3 adjacent training and test phases (with the test phase of one training phase becoming the next training phase), and they conducted 30 independent runs. Out of the 30 runs they calculated the best (best performance among the Pareto points) and average (average

[22] A computational technique based on the social behaviour of birds flocking to look for food.

[23] The reader is referred to Chapter 4 for explanation and equation of the Sharpe ratio.

performance of all Pareto points) performance of the 30 Pareto fronts. They compared these values with a buy-and-hold strategy and the performance of individual indicators. In training phases, both the best and average performances were able to beat all technical indicators and they beat the market in two out of the three phases. In the testing phases, the best points on the Pareto front were able to beat all technical indicators but were not able to beat the market, however the results in two out of the three test phases were comparable to the market but due to a lack of statistical significance analysis, conclusions cannot be drawn. The average performance was able to beat all indicators except for the Linear regression indicator in the third testing period.

2.4 Robustness in Dynamic Environments

Several definitions exist for robustness in the literature. The majority of research in this area defines robustness of solutions as insensitivity to small perturbations in the decision variables. Other definitions of robustness include: reliability of results in environments where the input parameters or the fitness functions are uncertain, consistency of results between different runs, and ability to recover after a change in problems with dynamic environments. Some of the research under multiobjective robustness is in fact solving the problem of robustness in a single objective optimisation by formulating it as a multiobjective problem with robustness as an extra objective, an example of which is found in [LAA05]. We are not interested in this class of research, and we focus here only on robustness in genuine multiobjective problems.

The section is organised as followed. Section 2.4.1 surveys research that views robustness as the reliability of solutions evolved when the environment is noisy and hence parameters are subject to small perturbations. Section 2.4.2 reviews research that defines robustness as stability between the different runs of the evolutionary algorithm. Section 2.4.3 surveys research that focuses on recovery after a change and hence the ability to track the optimum in dynamic environments. Finally, Section 2.4.4 reviews the metrics used for performance analysis in dynamic optimisation.

2.4.1 Reliability in Uncertain Environments

In [BA06], it is suggested that the definition of, "degree of robustness", be incorporated into the evolutionary algorithm as a measure of fitness of individuals in addition to the objectives' values. The degree of robustness of a solution x is a value k, where k is the number of neighbourhoods of the decision variables in which the percentage of solutions that belong to a specified neighbourhood α around $f(x)$ in the objective space is greater than or equal to a prespecified percentage threshold p. To promote diversity, preference is granted to solutions in sparsely populated areas if two solutions have the same dominance level and the same degree of robustness. The aim of the experiments was to determine the effect of the parameter p on the robustness of the solutions evolved. The new concept was tested on two mathematical functions used in [DG05], with the Pareto front plotted such that the solutions are distinguished by their

corresponding degree of robustness with various p values. The new concept was considered an extra tool to aid the decision maker in her choice of a solution from the front. However, the test problems were only two dimensional; since the perturbations in the decision variables can occur along any dimension, when the number of dimensions increases, computation of all possible combinations of perturbations in the hyper-cube in the neighbourhood of a solution becomes very expensive. Results are shown where the parameter p was varied between 60% and 100%, and the corresponding classification of non-dominated solutions into different degrees of robustness is plotted. Results of testing solutions with a range of degrees of robustness in an environment where the decision variables are actually changing are not given. However, it is a step towards clarifying the role of robustness in achieving stability of solutions in multiobjective optimisation.

The authors of [DG05, DG06] came across the same problem of dimensionality when they extended a definition of robust solutions used in single objective optimisation to be suitable for multi objective optimisation. The definition of the robust solution was such that it was the global minimum of the *mean effective functions*, defined with respect to a predefined neighbourhood of size δ. They generated 50 or 100 solutions in the neighbourhood, which effectively makes the method 50 or 100 times slower. Another result common with the previous research was discovering that some areas of the Pareto front always seem to exhibit concentration of robust solutions, while some other areas have only a sparse number of robust solutions or none at all.

Another recent work by Gasper and Covas [GCC07] used a combination of two types of robustness measures: expectation and variance of the fitness of a particular solution x. Expectation of the fitness is calculated as the weighted average of several points in the solution neighbourhood, and the variance of the fitness assesses the deviation from the original fitness in the neighbourhood considered.

In Gupta et al. [GD05] and Deb et. al [DPGM07], robustness is defined as sensitivity to small perturbations in the decision variables. The authors in [GD05] take into account the effect of the presence of constraints on the strategies for developing a robust multiobjective procedure. They argue that the effect of the small perturbations may lead to a solution becoming infeasible due to the constraints of the problem on the decision variables. Hence, when considering the neighbourhood of a solution in computing the effective objective function, only those solutions that are feasible are considered. This leads to the position of the robust effective front being shifted from the original front. In the results of both Gupta and Deb, their sample problems were the optimisation of artificial mathematical functions. The solutions evolved did not have to face an unseen environment and hence were not tested on one. No studies were carried out to measure how these strategies for evolving a robust front actually behave when trained and tested on real world problems.

2.4.2 Stability of Performance

Robustness in the work of [SDD07] was used to describe the standard deviation of a multiobjective algorithm between different runs, each using a different random seed. Using this definition, algorithms based on the Pareto dominance relationship such as the NSGAII are quite robust, while the relationship $Preferred$ [DDB01], [SDGD01] leads to poorly robust algorithms. It was measured that the standard deviation between different runs of the preferred algorithm is 67%, where as it is only $11-13\%$ in algorithms based on the dominance relationship. To improve the robustness of the Preferred method, an enhancement termed the ϵ-Preferred is introduced. The idea of the modification is based on two observations. First, that the relationship preferred only takes into account the number of objectives that a solution is better at compared to another, but neglects the objectives in which the other solution performs better. It could be that the drop in an objective in which the preferred solution performs badly is so strong that it will cause oscillations in the overall weighted sum of objectives [24]. The second observation is derived from a natural criterion of human decision making, where we eliminate some solutions when one of the objectives is worse than a certain limit, even when the solutions are very good in the other objectives . As a result, a domain expert is asked to provide a fitness limit for each objective ϵ. A solution is rejected if it exceeds one or more ϵ-limits and is not considered for the preferred relationship evaluation. Using the ϵ-Preferred method, with an ϵ-limit of 1000 in all dimensions, improved the fitness by a further 30% (compared to Preferred), and was able to drop the standard deviation to just 10%.

2.4.3 Dynamic Optimisation Problems (DOP)

In this section, we survey the literature on Multiobjective Evolutionary optimisation in environments with time dependent fitness landscapes. This field of research is concerned with the analysis of the performance of MOEA in dynamic environments, and the ability of the MOEA algorithms to respond to changes in the environment and recover from a possible drop in performance when the change is first introduced. Research into algorithms and their performance in such dynamic environments is usually termed: adaptive optimisation, optimum tracking or robustness of optimisation algorithms in dynamic environments.

In dynamic optimisation problems, the initial training stage has static input data, static constraints and static objective function. Then, a change occurs in one or more aspects of the training environment, either during training or after training, and the old solution set is no longer optimal. Retraining is usually carried out to evolve the new Pareto front. During each retraining phase the environment is static and the solutions evolved are to be used in the same static environment until a further change occurs. Robustness is often the word used to describe either stability or adaptability of solutions in the face of changes in the environments. Most of the problems studied in EAs and in MOEAs have static environments and the research into dynamic environments has begun to gain popularity over the last decade.

[24] The performance metric used was the weighted sum of the 25 objectives.

In dynamic optimisation problems, it is often assumed that the new instance of the problem can benefit from the knowledge gained during the previous instance (or various previous instances) and that the changes are not completely random. Hence, it is worth investigating the best ways to make use of knowledge previously gained or from the discovery of the pattern of change rather than having to resort to a complete restart which, of course, could be the only solution if the change is so radical that the previous knowledge would actually hinder the search in the new fitness landscape.

Research into dynamic optimisation in Evolutionary Algorithms (EAs) is more established than it is in MOEA, where it is only just beginning. However, both EA and MOEA share common grounds and a lot can be learned from looking into research on how EAs are made more suitable for DOPs. What makes MOEA algorithms different from standard EAs is their solution to the problem. While the EAs' populations are normally expected to converge and the solution to the problem is a single point in the search space, the MOEA are not expected to converge to a single point, but rather to a set of points, and diversity in the population is always maintained.

The rest of this section is organised as follows. Section 2.4.3.1, surveys the various possible classifications of change. Section 2.4.3.2 and Section 2.4.3.3 look at the methods for detecting the occurrence of change, and measuring its severity. Section 2.4.3.4 reviews the techniques utilised for adaptive optimisation.

2.4.3.1 Characterising Change in Dynamic Environments

Branke et al. [BS02, BSU05] used the following criteria: 1) **Frequency of Change**: How often the environment changes. 2) **Severity of Change**: How different the new environment is from the old one. 3) **Predictability of Change**: Whether there is a pattern for the change or it is completely random. 4) **Cycle Length / Cycle Accuracy**: This criterion is useful in a special type of dynamics called cyclic environments, and it measures how long it takes for an environment to return to a previously encountered state and the accuracy with which it returns.

De Jong on the other hand in [DJ99] characterises the change in the fitness landscape into one of the following: 1) **Drifting Landscapes**, where the topology of the landscape gradually moves due to slow and slight changes in the environment. In this case the optimal value moves slowly over time and an algorithm capable of tracking the optimum is required. 2) **Landscapes with Significant Morphological Changes**, characterised by the appearance of new optimum values in previously "uninteresting regions" of the search space and the disappearance of prior optimum values. 3) **Abrupt and Discontinuous Change in Landscape**, where the problem is static for the majority of the time, however, it is still prone to abrupt and infrequent change. Examples for this type of change cited by De Jong are a power failure on a power distribution grid, and an accident on traffic flow. 4) **Landscapes with Cyclic Patterns**, in which the problem environment alternates between a relatively small number of landscapes.

In Trojanowski et al. [TM99], DOPs are classified according to the regularity and continuity

of change as follows: 1) **Regularity of change**: a) Random Changes: Where the change is not dependent on the previous change and is completely random. b) Non-random, non-predictable change: The change is dependent on the last change but the dependency is sufficiently complex so as to be considered unpredictable. c) Non-random and predictable changes: Where the change is determined by a function, then it may be possible to predict the next optimum. The last category is further subdivided into cyclical and non-cyclical changes, defined as before. 2) **Continuity of change**: a) Discrete changes appearing in the environment from time to time with stagnation periods between them. b) Continuous changes are such that every time the fitness is measured, it is a little different. It is assumed that the continuous change is smoother and hence more traceable, and the discrete change describes a more abrupt change which makes subsequent local search inefficient or expensive.

Weicker [Wei02] offers a classification of a landscape change based on combinations of changing one or more of: 1) Coordinate translation, an example of which is the linear transformation of a certain length in a certain direction. 2) Fitness rescaling, where the [minimum, maximum] fitness changes through the addition of a rescaling factor to the original fitness interval. 3) Alternation, where different hills alternate in being the best hill at different generations.

In summary, Branke's classification categorises the change according to three criteria: *Frequency, Severity and Predictability* of change, with a special class in the predictability criteria where the pattern of change is cyclical. De Jong's drifting landscape and landscape with significant changes are in effect describing two types of the severity of change as would be recognised according to Branke's classification. The last two classifications of De Jong are two types of frequency of change in Branke's classification (continuous, discontinuous) with the cyclical change being representative of continuous change, as opposed to random change. Trojanowski, on the other hand, focuses on the predictability of change and classifies that into its possible varieties. Summing up, a change in the environment is analysed and categorised according to the following criteria:

1. **Frequency of change**: This criterion is especially important if the change is occurring during the optimisation process, and it is measured by the number of generations (fitness evaluations) allowed before a change occurs. It is further classified into one of the following sub-categories:

   (a) Infrequent or discontinuous change

   (b) Continuous or frequent change

   This criterion is mostly assumed to be somehow controllable, either by the number of generations between changes in mathematical optimisation functions, or by the number of generations the algorithm needs to adapt after a change occurs in real world problems.

2. **Severity of change** Usually measured by the distance from the new optimum to the old one (if the two are known). It is classified into:

(a) Small changes: where techniques to cope with noise in the training environment could be suitable and sufficient to cope with this type of change, for example as in [BA06].

(b) Severe changes: The algorithm will have to be able to respond to such change to maintain a high level of fitness after each change. In the most extreme cases re-initialisation could be the answer; especially if the change is also unpredictable and infrequent.

Measuring the severity of change, (as well as the detection of change) is hardly straightforward, and is still an open research area. Efforts in this area are surveyed in more detail in Sections 2.4.3.2 and 2.4.3.3.

3. **Pattern of change**: The predictability / pattern of change can either be known beforehand and be dependent on the type of application, or can become known through observation of historical data. It is subdivided into:

   (a) Random change

   (b) Non-random but non-predictable

   (c) Non-random and predictable

   (d) Non-random, predictable and cyclical: The algorithm required would be one that remembers the different states and later detects cycles and appropriately retrieves the corresponding state, hence, memory approaches are suitable (see Section 2.4.3.4).

2.4.3.2 Detection of Change

The most intuitive method for detection of change would be through observation of population performance. A deterioration of performance was assumed to be a good indicator of change in [TM99]. However it should be noted that the mere change of performance (whether up or down) - especially after convergence - is a suitable measure of change. The change of detection area is largely merged with the area of measuring the severity of change. In many cases, the change detection is carried out by using a metric for the severity of change and periodically applying it between generations.

2.4.3.3 Severity of Change

A suitable measure of severity of change can influence the technique used to adapt to the change. If the severity is deemed low, then the new optimum is probably close to the old optimum and using the old population as a base for optimisation with the addition of some diversity will probably lead to finding the new optimum. If the two optima are known, then a measure of distance between them will be a good indicator of the severity of change. However, since the actual optimum is usually unknown, Branke [BSU05] suggests some measures as estimates for the severity of change, all based on observing the fitness of sample points from

the search space before and after a change. Three of the measures suggested in [BSU05] are: 1) **Fitness Correlation**: Measures the correlation coefficient of the finesses of all solutions before and after a change. A high value indicates increased similarity to a previous environment. Experiments have shown that this measure does yield high correlation values when used in the knapsack problem with moderate severity of change. 2) **LHC Fitness Correlation**: Correlation between fitness values reached through a hill climbing algorithm (LHC) from the sample points before and after a change. If the correlation is high it indicates that simple hill climbing after a change can be sufficient to reach good values after a change. However, this measure gives a lower correlation of values than those found using the previous measure on the same dynamic problem. The authors are puzzled by this result and do not have a logical explanation for it. 3) **Fitness Correlation of Similar Points**: This measure examines whether similar points (in terms of search space distance) experience a similar fitness change. The correlation of fitness change with the distance d is measured for n points. For each of them we pick a random point with distance d and observe the correlation of fitness with the distance. It was found in this study that the larger the distance between the pair of points selected the less the correlation of fitness change.

The measures suggested were examined on the dynamic multi-dimensional knapsack problem in which for every change, the profits, resource consumption and constraints are multiplied by a normally distributed random variable such that the random variable used has a standard deviation of 0.05, that is to say, the dynamics in the environment were tuned by hand and the severity of the change introduced was moderate.

Liu et al. [LW06] defined a *Feedback Operator*, ε, for detection of change. The operator measures the difference in fitness of solutions before and after a change divided by the maximum and minimum fitness values obtained in the two instances.

$$\varepsilon = \frac{\sum_{i=1}^{N} ||f(x_i, t) - f(x_i, t-1)||}{N||R(t) - U(t)||} \tag{2.5}$$

where R(t), U(t) are the worst and best fitness of the problem under the environment of time t. X_i are the individuals of the population with a size of N. After every generation, $\varepsilon(t_i)$ is computed, if $\varepsilon(t_i) > \eta$ (where η is a user-defined parameter), then a change is detected. The technique for adapting to change was the re-initialisation of the population.

2.4.3.4 Techniques for Adaptive optimisation in Dynamic Environments

Standard EAs are designed to converge and the knowledge obtained from the training is encompassed in the final generation. When a change in the problem occurs, we could possibly start from a new random population. However, if some of the knowledge gained in the first instance of training could be carried out to the new instance, then reusing individuals from the last generation could have a head start over starting from a random population. Nevertheless, the lack of diversity in the population could hinder the new search process. Techniques for adapting the EA algorithms to dynamic environments are grouped in the following categories:

1. **Diversity Management** In this approach, techniques for generating or maintaining diversity are studied. In the beginning, training proceeds as usual seeking to converge to an optimum. When a change occurs, the population is retrained with the hope that the increased diversity will speed the search to the new optimum while still making use of the knowledge gained in the previous stage. De Jong in [DJ99] focuses on the critical role that diversity plays in adapting to the changing environment (landscape) and suggests a number of way to accomplish diversity maintenance: through weakening the selection pressure, crowding and niching mechanisms, mating restriction techniques to maintain diversity in island models, and injecting diversity when a change is detected which is especially useful when the type of change is infrequent and abrupt.

    - **Generate diversity after a change**: Usually through increasing the mutation rate when a change is detected. Hypermutation [Cob90] is an example of such an approach, where the mutation rate is increased drastically for a few generations after a change. In an examination by [Wei02] it was found that a GA employing hypermutation outperforms a standard GA on several classes of dynamics considered. In these approaches, the problem lies with the best method of detecting the change, which is not always easy.
    - **Maintain diversity throughout**: Sharing and crowding mechanisms are usually used in this approach [CV97]. Random immigrants are also used as in [Gre92] and [YTY08]. The goal is to keep the population from converging in the hope that when a change occurs the population will still represent a widespread sampling of the search space suitable as the start of the search for a new optimum.

2. **Tuning Evolutionary Operators and Operator Adaptation**: In this approach, dynamic adaptation of algorithm parameters (usually mutation) is employed to respond to a change in the environment [Ang97, BS96].

3. **Memory Approaches**: In these approaches, the evolutionary algorithm maintains a memory system whose data can be recalled when needed. This approach in useful when it is known that the optimum is repeatedly revisited either in a cyclical manner or otherwise (Non-random change, predictable, possibly cyclical).

    - **Explicit Memory**: Specific strategies in the algorithm are employed to store and retrieve information. The algorithm remembers several fit individuals found in the past while exploring some other environments, and when a new environment is found to be similar to one of the previously visited environments, the individuals found to be fit in that environment are retrieved and are used for training in the new environment.
    - **Implicit Memory**: The EA algorithm is designed with redundancy in the representation on the assumption that some dormant parts of the representation are used

by the EA to store information which is not useful in one instance but as a change occurs, the EA could find it more useful in the new instance and the dormant parts then become active again. The use of diploidy is common is this approach. An example of this type of memory is found in [DM92, TM99], and of the use of diploidy in [GS87].

4. **Multiple Population Approaches**: In these approaches, the population is divided up into multiple sub-populations, where each is tracking one of the multiple peaks in the search space. Hence, different populations collect and maintain information about different interesting regions in the space. An example of this work is found in [BKSS00].

For a more detailed overview of the topic, the reader is referred to [JB05] which presents a survey of the field of EA optimisation in the presence of uncertainties in the domain problem in general, with the DOP as a subclass of the general class of uncertainties.

MOEA's populations do not converge in the usual sense of the word [25]. MOEA almost always employ techniques for diversity maintenance, and they do converge in the sense of driving the population towards the area of the search space where the Pareto front is located, but within this area, diversity – along the Pareto front – is promoted in all MOEA algorithms, usually through crowding and niching mechanisms. In the approaches surveyed below, diversity maintenance was always a key factor in conjunction with the technique used to increase adaptability of the algorithm. Some examples of techniques used for coping with DMOP are:

- **Diversity Management**: The adaptive mutation operator is proposed in [GAM08], where the mutation rate becomes a function of the performance of every individual in the population in every objective. Hence the value for the operator is different from one individual to another and from one generation to another. The mutation operator is tuned to the individual performance through calculating a weighted average of the normalised difference between each objective value and a fraction of its maximal value.

- **Forward Looking Approach**: In Hatzakis et al. [HW06], a forecasting model (the autoregressive model) is created using the sequence of previous optimum locations from which an estimate of the next optimum location is extrapolated. Using this forecast, a group of individuals (prediction set) around this location is created and is used to seed the population when a change in the objective landscape is detected. This technique can clearly be used with either single or multi-objective dynamic optimisation problems, but the results reported on the work cited involves a multiobjective problem. The approach assumes predictability of change such that the past sequence of locations of the best solution found during a series of time steps is seen as a time series and is used to predict the next location of the new best solution. Hence, the approach is useful in problems with non-random,

[25] In single objective EAs, the populations is said to have converged when there is very little diversity among its individuals and no further improvement in the solution quality is obtained.

predictable change that follows a function which could be extrapolated. Since in the multiobjective optimisation we are tracking a Pareto front rather than a single point, the approach used in the experiments carried out is to choose a number of points[26] on the Pareto front and to track them individually. Another approach suggested are suggested by Hatzakis et al., is to fit an analytic curve which describes the Pareto optimal set and subsequently forecast changes in the parameter values of this curve.

Two variants of the autoregressive (AR) model are used. In the first, the whole design variables vector is treated as a vector time series. In the second, each design variable is treated as a single time series and is predicted separately. The results (averages of 20 runs) of applying this technique to the FDA1 [MF04] are reported, with the AR model using separate time series achieving better results (31% reduction in Pareto front error and 50% reduction in design vector error) than the AR model using vector time series (2.5% reduction in Pareto front error and 3.4% reduction in design vector error), with errors being measured at the end of the run immediately before the change. Although the decrease in error is not huge, it was shown in further experiments how the benefits of the prediction method become evident as the frequency of change increases, giving the MOEA less time to adapt and converge.

- **Detection and Reinitialisation**: This is a straightforward method consisting of reinitialising the population in order to react to changes in the environment. This kind of approach was mainly explored for single objective optimisation in the 1990s, more techniques are now used to transfer information from the past into the new state of the problem. An example of this approach in the MOEAs is the work of [LW06].

2.4.4 Performance Analysis in DOP

In static single objective optimisation problems, ensuring the continuous improvement of the best and average fitness and ultimately a comparison between the best fitness and a known optimum or a good approximation thereof are very adequate measures for the degree of success of an optimisation algorithm. Similarly in static multiobjective optimisation problems, the continual progress of the Pareto front towards the optimum front is sufficient. In dynamic optimisation problems on the other hand, a suitable performance metric to measure the degree of success of an algorithm still represents a gap in the research. According to [Mor03], a good metric for performance in dynamic environments should possess the following characteristics: 1) have an intuitive meaning, and 2) provide straightforward methods for statistical significance testing of comparative results.

2.4.4.1 Performance Analysis in Single Objective DOP

Although there is still no universal agreement on the suitable metrics for performance in DOP, some attempts exist in the literature, and some of them do share common underlying concepts.

[26] Points selected are those achieving the minimum value for each objective function.

Examples of such metrics are:

- *Accuracy* is used [AE04], and is defined as the difference between the value of the current best individual in the population of the generation just before a change and the optimum value, averaged over the entire run. The lower the value of this measure the better, with a zero value achieved meaning that the algorithm was able to find the optimum value in T generations following every change.

- Three aspects of performance: *accuracy; stability; and recovery* are emphasised in [Wei02]. Accuracy is a value in the range [0,1], measured by subtracting the worst fitness in the search space from the current best fitness and dividing by the best possible fitness minus the worst possible fitness in each window of time that the problem is stationary. The stability metric measures the drop in accuracy when a change occurs, and an algorithm is called stable if changes in the environment do not affect the optimisation accuracy severely. The third aspect is the ability of the algorithm to react quickly to a change and is measured by the number of generations the algorithm needs to recover after a change.

- The best-of-generation averaged at each generation over several EA runs (also known as *off-line performance*) is used in [Ang97, Bäc98, BKSS00, Cob90, Gre99, KUE05, YTY08]. This measure is usually used to plot a performance graph for the algorithm, often with the average fitness rising as the generations advance, then suddenly dropping when a change occurs, gradually rising again as the algorithm adapts and so on.

- *Collective mean fitness* [Mor03, Ric04] is a measure yielding a single value that is designed to provide an aggregate picture of an EA performance. It measures the average best of generation over a sufficient number of generations required to expose the EA to a representative sample of all possible landscape dynamics, and is averaged over multiple runs.

In the above list, the first two metrics assume the precise knowledge of when the change has occurred, and that the optimum value at each stage is known. It is not always realistic to make these assumptions, which makes these metrics practical only in test problems with controlled change and known optimal solutions. The next two metrics are not subject to these assumptions, hence are more practical.

2.4.4.2 Performance Analysis in Multiobjective DOP

This is a more recent topic with few reported metrics in the literature. The existing metrics usually compare the evolved Pareto front against the true Pareto front. However, for many real world problems, the true Pareto front is not actually known. Some of the metrics proposed by other researchers are:

- Visual comparison of the Pareto fronts generated after sufficient number of generations for each change, as in [LW06].

- The performance graph of the best value achieved for each objective separately at regular intervals over the entire run [GAM08].

- A time-dependent convergence or coverage metric. Examples include: Farina et al. [MF04] and [LBK07] who propose the use of a convergence metric based on the time varying version of the Generational Distance (GD)[27] metric [ZLT02], and the time-dependent *nadir*[28] and *utopia*[29] points. This metric assumes knowledge of the true Pareto front. In [WL09], the mean of a variation on the generational distance metric(called IGD) is calculated by averaging the IGD before each change over the number of changes. The hypervolume metric calculated immediately prior to each change was the metric used in [DRK06] and [LBK07]. This method of measuring performance is beneficial when comparing two algorithms or against the real Pareto front.

- [COT09] adapts the metrics proposed in [Wei02] to suit multiobjective problems where the true Pareto front is not known. This is achieved through changing the definition for the maximum and minimum fitness in the *Accuracy* metric to be the maximum and minimum hypervolume of the Pareto front in the current time window. To overcome the need for knowing the real optimum in the original metric, he suggests an algorithm for comparing the current hypervolume with nearby approximate fronts.

2.4.4.3 Performance Analysis of MDOP in Out-of-Sample Environments

Some of the metrics in Section 2.4.4.2 could also be used for evaluating the performance in out-of-sample environments, such as a convergence metric if the true Pareto front is known. Examples of other metrics include:

- The mean fitness value (over the Pareto set solutions) of each objective separately was used in [Bin07]. Similarly, [BFF07] used the mean fitness value and the standard deviation of the top 15 rules in training and compared that performance to the rules' performance on out-of-sample data. In [BJ08], the best and average achieved in each objective were compared against a benchmark in training and validation.

- In [CTM09], discrete intervals of the first objective were plotted against the distribution of the averages of the second objective achieved within each interval. The intervals chosen for the first objective were fixed, and the varying averages achieved by the second objective were compared during training and validation.

[27] Although the metric used in [LBK07] is slightly different as it takes into account the distribution of points in comparison to the distribution on the true Pareto front

[28] In minimisation problems, this point represents the *maximum* value possible for each of the objectives.

[29] In minimisation problems, this point represents the *minimum* value possible for each of the objectives.

Chapter 3

A New Approach for Multiobjective Robustness in Dynamic Environments

3.1 Introduction

In this work, we research the effectiveness of using Mutliobjective Genetic Programming (MOGP) to generate a set of trading rules for portfolio optimisation. The evolved trading rules are used to select stocks in portfolios, such that the portfolios' characteristics correspond to the efficient frontier of tradeoffs between objectives specified in the model. MO learning systems, like all machine learning algorithms, go through a training phase, where a data set describing a sample environment is used for training; out of which a solution set (in the multi-objective optimisation case) is produced. The research field of multiobjective optimisation has focused primarily on problems with stationary environments. However, in some optimisation problems, the environment is dynamic, that is to say, it changes either during or after training. Hence, the corresponding solution set may not be fixed but is rather expected to change in response to a change in the environment.

A very good illustration of dynamic behaviour in the real world is financial optimisation problems. The environment in the financial world is constantly and possibly abruptly changing. We know for a fact, that as soon as we have trained on one data set, the environment (as described by: inputs to the learning algorithm, possible value ranges for objective functions and fitness landscape) has already changed. A trader/ fund manager will *always* be using the evolved solution set in an environment different from the one it was trained on. The changes in the financial environment will eventually become too large (when the environment has become too different from that which the solutions were derived from), and retraining will be inevitable. Several issues arise here: how long do we need to wait before a sufficient number of data points necessary for re-training have accumulated? And while waiting for new data, are the old solutions totally useless, or can their robustness be improved so that we get a graceful deterioration of performance?

Research on evolutionary algorithms' performance in dynamic environments has been directed towards adapting the algorithm so it is more suitable for retraining when a change is

detected. Hence, the research is focused on the following issues: detecting that a change has occurred, investigating how to make the algorithm able to retrain, and measuring success in tracking the optimum as it changes position in the fitness landscape. However, in the financial domain, the ability to generalise to out-of-sample data is as important as the ability to track the optimum when the training environment changes. No research has previously been carried out to assess and quantify the applicability of MOEA solutions in unseen environments in terms of stability of the solutions' objectives trade-off . We aim to take the current research on dynamic multiobjective problems a step further by focusing on the minimisation of the solutions' movement along the Pareto front when the solutions are used in practice in between episodes of re-training.

In this chapter, we:

- Examine the robustness (performance of solutions in unseen environments) of the MOGP solution set in the context of a portfolio optimisation problem, where the training environment is different from the validation environment, as is the case in real life.
- Develop suitable metrics for robustness of multiobjective solutions when tested in unseen environments.
- Suggest techniques for improving the robustness of solutions during the critical period between recognition of the need for retraining until actual retraining occurs.
- Discuss what is meant by change in environment and how to quantify change.

The rest of this chapter is organised as follows:

- Section 3.2 defines the terminology used, and explains the need for a different model for robustness assessment for MOGP applied to dynamic environments.
- Section 3.3 provides an analysis of what constitutes robust behaviour in a portfolio optimisation problem, and explains what we mean by dynamic environments.
- Section 3.4 presents the concepts and definitions developed for the analysis of MOGP in a financial dynamic environment and the suggested metrics to evaluate the solutions' robustness.
- Section 3.5 proposes techniques for improving one particular aspect of Pareto front solutions' robustness which are the basis for the experiments carried out in Chapter 5.
- Finally Section 3.6 is a brief discussion on optimum tracking in the financial domain, and presents two proxies for measurement of change which are later examined in Chapter 5.

3.2 Critique of existing Multiobjective Robustness Models

This section aims to provide a critical overview of existing research on robustness and explain why it is not sufficient to capture the deficiencies in adapting to new environments in the *multiobjective* context. However, since robustness means different things to different researchers and the literature does not show a uniform definition of robustness; we start by presenting the terminology used throughout this thesis. We follow with an illustration of the behaviour MO algorithms exhibit in dynamic environments, and hence the need for a different way of examining and quantifying robustness.

3.2.1 Terminology

- Problem: The optimisation problem considered. The problem specification describes the input variables, the output, the objectives to optimise, constraints on the solutions, and a benchmark performance against which the quality of solutions is compared.
- Objectives: Output (solutions) characteristics which the learning system is trying to optimise.
- Learning System: The machine learning technique used for learning. In this work it is Genetic Programming (GP). Since we are attempting to optimise more than one objective simultaneously, we are using a Multiobjective Evolutionary Algorithm with the GP (MOGP).
- Decision Variables: The input to the learning system. In the Genetic Programming paradigm, they will represent the leaves of the GP tree. They are the factors that the researcher believes have an impact on the objectives considered.
- Solution Set: The output of the multiobjective optimisation system. In the GP case, they are equations of decision variables and mathematical functions.
- Environment: An environment describes a period in time in which the problem is considered. Any one environment is described by the values of the decision variables corresponding to this period, the specific constraints that are observed, and the objective values (either as provided by a benchmark as an approximation of the optimal values, or as provided by the learning system).
- Stationary Problems: Stationary problems are optimisation problems for which the decision variables have the same values regardless of the time period considered. Hence, the decision variables are constants. In these problems, training is run once, and if the optimisation results are satisfactory, the solution/solutions obtained is used thereafter – these problems are also known as off-line optimisation problems.
- Dynamic Problems/ Dynamic Environments: Optimization problems for which the values of the decision variables and the achievable objectives' values are time dependent.

3.2.2 Review of Previous Research

The majority of the robustness definitions characterize robust solutions such that, when parameters are subjected to small perturbations, the resulting objective values remain in the close vicinity of their previous (recorded in training) performance (objective values). The parameters considered as subjects of change are the following:

1. Algorithm parameters.

2. Input decision variables.

3. Objectives uncertainty, within a known range.

The change in the algorithm parameter (in case of the evolutionary algorithms) may result from changing the random initial seed, or experimentation with internal algorithm parameters like crossover rate, mutation rate, population size, et cetera. The perturbation in decision variables is mostly attributed to noise, hence the assumption of the slight variations. The perturbation in objective values, within a known range and probability distribution, is attributed to uncertainty due to estimation errors or an approximation of an unknown model. All the perturbations are assumed to be small.

In real world problems where the change in the environment leads to changes in both decision variables, and the objective ranges. In such a case, the whole fitness surface is shifted in the objective space, changing the range of objectives and possibly its shape, in response to the new environment. Hence, a test for robustness that examines whether the objective values change minimally becomes inadequate.

Such a class of problems can be single-objective or multiobjective. In the case of the single-objective, we are interested in solutions that retain the optimality of the objective within the fitness landscape of the new environment. In the case of multiple-objectives, we are interested in optimality plus two additional aspects: solutions that retain their objectives' profile rank, and a front which retains its diversity and uniform distribution, so that all regions of the new trade-off surface remain well represented.

For evaluating the robustness of MOEA evolved solutions in out-of-sample environments, the previously proposed metrics only consider one criterion: the optimality of solutions / Front in the new environment. Optimality of the solutions is usually measured by the mean fitness value over the Pareto set solutions. Optimality of the Pareto front is usually measured by using a convergence / coverage metric. The stability of performance of the individual solutions, in terms of maintaining the particular trade-off between objectives achieved in training, is overlooked in previous research. In Section 3.2.3, we demonstrate the importance of this criterion in the performance analysis of multiobjective algorithms.

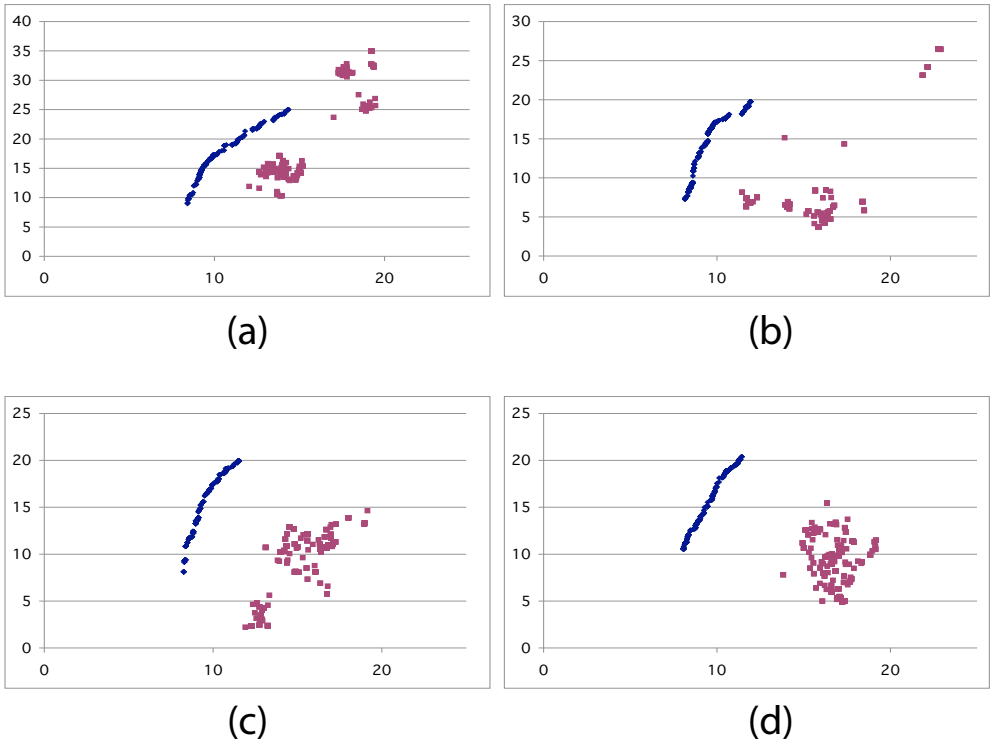


Figure 3.1: SPEA2 Pareto Fronts in Training (blue, upper left) and Validation (red, lower right) over four runs. The x-axis is risk, and y-axis is return

3.2.3 Experiment: Performance of an MOGP in a Dynamic Environment

In this section we investigate the performance of solutions on the Pareto front evolved by a multiobjective evolutionary algorithm in an out-of-sample environment. The multiobjective algorithm used is the SPEA2 algorithm [ZLT02].

To test the SPEA2 solutions' robustness on out-of-sample data, the algorithm was run 15 times on training data that spanned 60 months of financial data. After each run, the solutions on the Pareto front were tested on out-of-sample data of 20 months [1], equivalent to using an investment strategy represented by the solution tree to manage a new financial portfolio. The performance of the algorithm on the validation data varied between the runs. Figure 3.1 presents four runs with the Pareto front in training and in validation.

It is noticed in these graphs that not only is the performance different from that in training, but also that the Pareto front as a whole loses its distribution characteristics. Another more serious problem is illustrated in Figure 3.2. The figure shows a solution $P1$ that in training displayed relatively high return at relatively high risk — but in validation it had relatively the worst return with low-to-medium relative risk. Another solution $P2$ that was relatively medium-return/medium-risk in training became relatively low-return with medium-to-high-risk in validation, and also became dominated by other solutions. The solution $P3$ changed

[1] Details of the financial data used is explained in Chapter 4.

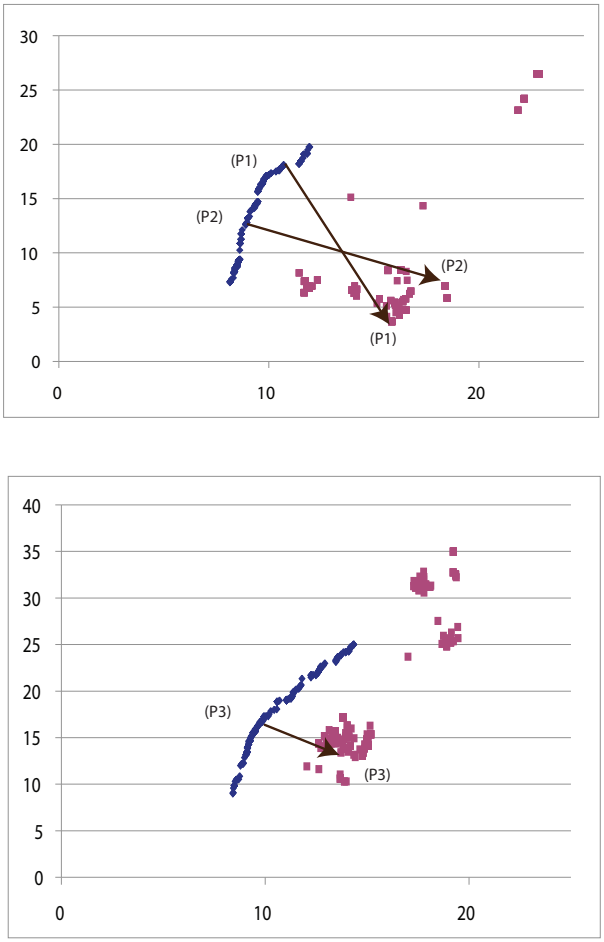


Figure 3.2: Example of Solutions Changing their Objectives Profile(Cluster). The vertical axis is return on investment, and the horizontal axis measures risk. The x-axis is risk, and y-axis is return

from relatively medium-return in training to relatively low-return in validation and clearly became dominated by several other solutions that achieved the same risk with higher returns.

The solutions changing their relative position on the front when faced with a different environment will be a problem for a variety of real world problems. Although the solution set may or may not still perform well on average in a different environment, in a real life scenario, individual solutions will be used, and these are the ones we are interested in their performance and not the average of the collective performance of solutions on the front. This is of particular importance in our application. A fund manager requiring a particular objectives profile clearly would expect that the strategy maintains its objectives profile. When used in investment, if the solution achieves objective values that are optimal but lies on the wrong portion of the efficient frontier, it would be unsuitable from the point of view of the fund manager.

3.3 Problem Analysis and Classification

In this section we provide an analysis of the necessary attributes of a multiobjective algorithm applied to optimisation in dynamic environments, and a preliminary analysis of change dynamics in the financial stock market data used in our experiments.

3.3.1 Robustness of an MOGP in a Dynamic Financial Environment

In this work robustness means the following: when an evolved solution is evaluated in an out-of-sample environment, it retains its:

1. Optimality – with respect to the objectives range of the new environment.
2. Objective characteristics – the exact or similar objectives tradeoff as that recorded in the training phase.

In addition, robustness of the front means:

1. The extent to which all solution on the front achieve the previous criteria – hence the Pareto front is still a good approximation of the ideal Pareto front in the new environment.
2. The Pareto front remains diverse and well distributed in the new environment.

The same logic applies to optimum tracking, where the environment changes during training. The ability to track the optimum is examined by the ability of the algorithm to continuously achieve the previous outlined items. However in the case of optimum tracking, the solution set will be evolving, hence there is no point in examining that one single solution retains its objectives' characteristics.

3.3.2 What is an Environment Change?

From the point of view of the financial system, an environment may be viewed as a period in time for which the market either had a consistent general direction (bull or bear), or was volatile with no consistent general direction. From the point of view of the optimisation algorithm, any

change of the input data that leads to a change in the fitness landscape is considered a change in the environment. Both points of view conforms to the idea that an environment is a period of time for which an optimisation model correctly describes the factors in play. Hence, the rules produced by the optimisation model remain optimal in comparison to the benchmark. When the results become no longer optimal, then it is a new environment which requires a new model to correctly describe it.

Ultimately, a correlation between the two views on change would be interesting – where a relationship between a change in market direction and a required re-optimisation by the algorithm is established. However, the main problem with the assumption of such a correlation is that an analysis of the market direction is mostly only possible post-hoc, and it is very difficult to establish the move into a different market environment when it is only just starting. Hence, in this research a change in the environment is identified by a change in the input data (decision variables) that may or may not result in the solutions becoming suboptimal or invalid. However we will aim to observe the links between the trend of market data and the effect on solutions performance . For this reason an understanding of the market dynamics is important in this research. In the next section we present a financial analysis of the FTSE100 market data used in this thesis.

3.3.3 Towards an Analysis of Environment Dynamics in the Stock Market

In this research we are dealing with real world stock market data, hence the change in the environment is not artificially controlled but is rather the result of a change in the values of and relationships between the input factors and resulting objectives range. A plot of the stock prices and the market return on investment would reflect the changes that occurred during a certain time span. In addition to the price information, the MOGP is fed with the values of fundamental and technical factors describing the performance of the underlying companies. The change in the price of any stock is possibly a result of a change in one of the internal company factors. On the other hand, it could also be due to an external (global) factor that had an effect on the performance of all the stocks. An example of such a factor is a change in the interest rate, oil price or even something related to the sentiment of the market which led to a change in the confidence of investors and consequently the abrupt movement of the prices either up or down.

3.3.3.1 Market Index

We are looking at the performance of 82 companies' stocks [2] that were consistently a constituent of the FTSE100 during the period from May 1999 to December 2005. We have created an investment fund that invests an equally proportionate capital in each of 82 stocks to simulate an index fund investment and plotted the cumulative return on investment (ROI) of such a fund. The performance of this index fund (Figure 3.3) is an indication of the stock market performance and can be used to identify and analyze market trends in the period considered.

[2] For explanation of the selection of these particular companies and their financial data, please refer to Chapter 4.

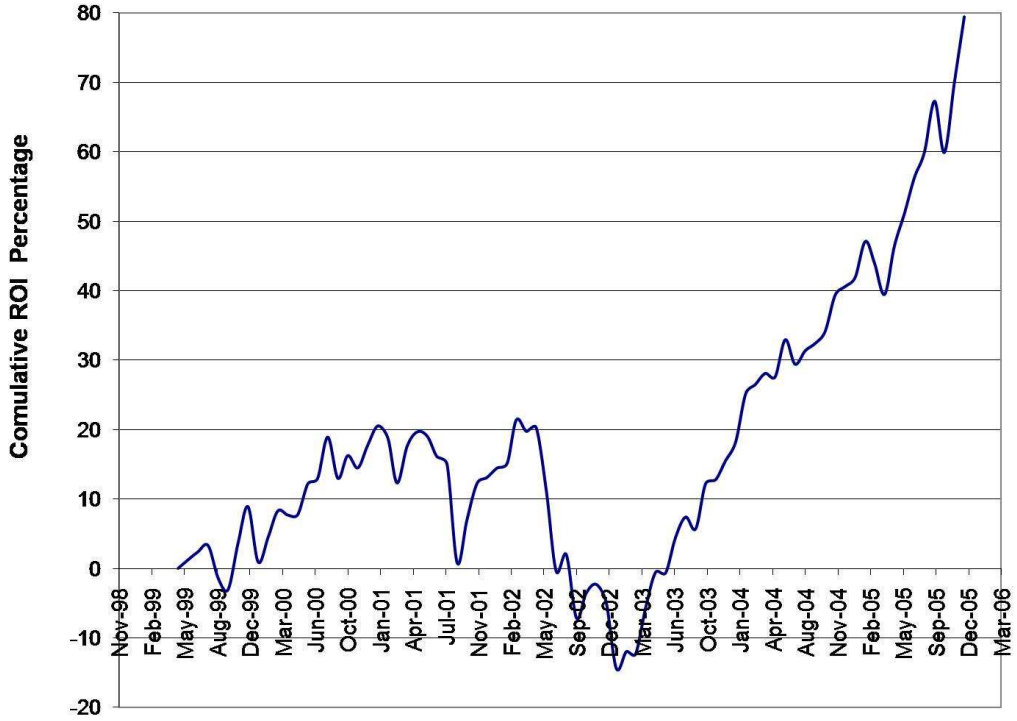


Figure 3.3: Performance of Index Fund Benchmark

3.3.3.2 Trends in the Financial Market

Primary market trends Prices fluctuate constantly in the stock market. However, when there is a continuous trend of price increase the market is described as bull. When there is a continuous trend of decrease the market is described as bear.

A bull market is associated with increasing investors' confidence leading to an anticipation of future price increases. An accepted measure is a price increase of 20% or more over a period of two months. A bear market on the other hand is accompanied by investors' lack of confidence and further losses are anticipated. It is defined by a substantial drop in prices of the majority of stocks. An accepted measure is a price decrease of 20% or more over a period of least two months [GPSW05].

Secondary Market trend Secondary trends are price fluctuations against a primary trend. It is usually a change in the price direction with a magnitude of 10%-20% and has a short duration. If the secondary trend continues then it may be the beginning of a new primary trend. A decrease in the price during a bull market is called a market correction, while an increase in price during a bear market is called a bear market rally [GPSW05].

3.3.3.3 Market Trends in Data

Using these guides to analyse the market trends of the UK FTSE100 from May 1999 to December 2005 we notice the following:

- The period from May 1999 - May 2002 shows a volatile market with a primary bull trend. Several price corrections occurred (8% -20%): a) In January 2000 where the index lost 8% of its value in one month, gained 4.5% in February and regained the whole 8% by March. b) In September 2001 (9/11 Attack on the twin towers) the index lost 14% of its value in one month but recovered 6% during October, another 6% in November and then was back to its pre-September value by February 2002.
- The period from June 2002 - June 2003 (dot com bubble) had a bear primary market trend which saw the index lose 23% of its value at one point. It started with a loss of 6% in June followed by a further 11.3% in July 2002. The trend continued until it started to reverse in July 2003.
- The period from July 2003- December 2005 was categorised by a strong bull market with few minor price corrections; in March-April 2005 the index lost 8% of its value over two months; in October 2005 the index lost 8%. Both drops recovered quickly .

3.4 New Approach for Analysis and Assessment of MOGP performance in Dynamic Environments

3.4.1 Introduction

This section presents our proposed model of analysis and assessment of the MOGP performance in dynamic financial environments by providing definitions and metrics for solution robustness and the Pareto front robustness.

We propose that in order to quantify robustness in the multiobjective context, we need to assess more aspects than just optimality of the front. To assess robustness of a solution, we need to examine its quality in the new environment as well as how much it has changed its objectives profile amongst other solutions on the Pareto front. To assess the robustness of the front, we need to examine its optimality and quality, in addition to examining the collective change of objectives profile among its solutions.

Hence, we modify the definition of robustness of multiobjective solutions to be the solutions' insensitivity to changes in the environment such that they maintain their: *optimality* and *objectives profile* when the environment changes. Specifically, to quantify robustness in dynamic environments, we need to assess the following:

- Are the solutions (presumably near optimal in the training environment) still near optimal in the new environment? From a financial perspective, a relative measure of solution performance can be obtained from a measure of their risk-adjusted return, as given by the Sharpe ratio [Sha64].

- How much have the solutions changed their objectives-cluster and rank-order (see Section 3.4.2) amongst other solutions on the Pareto front? This provides a degree of confidence that a solution expected to yield a certain relative risk-adjusted-return will have a similar behaviour in the new environment.
- How good is the quality of the whole front formed in validation? This can be measured using the same metrics used to measure the quality characteristics of the front in training.

The following section aims to provide understanding and measurement of the second aspect of robustness.

3.4.2 Definitions

Definition 1 : **Objectives' Clusters**

For a problem with m objectives to be optimised, assume that the value of each objective can be ranked as one of *high*, *medium*, *low*. We will call each unique combination of multiple objectives rankings a *cluster* of objectives ranks. Solutions on the Pareto front are then classified into a maximum of 3^m, and a minimum of 3 clusters; two on both extremes, and one with medium values for all objectives. Solutions on the Pareto front are classified into clusters such that members of a cluster have similar classifications for each of their objectives. A cluster C_i is identified by a vector of the m classification values of the cluster centroid $(c_1, c_2, ...c_m)$, where m is the number of objectives. Hence, we have:

$$Cluster(C_i) = \langle Cluster(o_1), ..., Cluster(o_m) \rangle \tag{3.1}$$

where $Cluster(o_j) \in \{L, M, H\}$ and the j^{th} value in the cluster shows the j^{th} centroid value classification.

Definition 2: **Cluster of a Solution**

Cluster membership is assigned for each solution x_k on the front, where k is the index of the solutions, $k \in [1, n]$, and n is the total number of solutions on the Pareto front. Thus, for all solutions on the front, the following function is defined:

$$Cluster(x_k) = C_i \;\; if \;\; x_k \in C_i \tag{3.2}$$

Definition 3: **Rank of a Solution**

Each solution on the front will have n *objectives rank order* relative to the other solutions on the front, and based on its objectives values. The objectives ranking order of solutions is not defined by the absolute values of the objectives, but rather by their relative value in comparison to the other solutions on the current front. Also, note that the objectives-ranking-order is not preferential; it is an indication of the relative position of a solution or a cluster of solutions in the objectives space with respect to each of the objective values.

At the last generation of training, a ranking algorithm is run after the Pareto front has been identified, after which each solution has a rank order and the following function is defined for all solutions on the front:

$$Rank(x_k) = (Rank_{o1}, Rank_{o2}, ...Rank_{om}) \tag{3.3}$$

where $Rank_{oj}$ is the rank order of the value of objective j among the values of the same objective achieved by other solutions on the front.

3.4.3 Robustness Metrics

Using the definitions above, we derive metrics to measure robustness of a single solution and of the whole Pareto front in a dynamic multiobjective environment.

3.4.3.1 Robustness of a Solution

Robustness of a solution x_k to a multiobjective problem is defined qualitatively as the degree of its insensitivity to changes in the environment, and is measured quantitatively by three measures:

1. Whether the solution is still optimal in the new environment.

2. How well it preserved its cluster in the new environment — using the cluster distance change metric Δ_k

$$\Delta_k = \sum_{j=1}^{m} Dist(Cluster(o_j)^{env1}, (Cluster(o_j)^{env2}) \tag{3.4}$$

Equation 3.4 returns a value between 0 (best), and a max of $(m*(numofclusters-1))$ (worst).

The *Dist* function measures the distance of the cluster change between environments, it calculates element-wise differences across the cluster vector and add the differences. For example, in a two-objective problem, where only three clusters exist, if a solution moves from a cluster $\langle high, high \rangle$ to $\langle medium, medium \rangle$ then we measure this as a move of length 2, whereas if it moves from $\langle high, high \rangle$ to $\langle low, low \rangle$ then this is given a measure of 4. Table 3.1 shows the values given by the *Dist* function in a two-objective problem with three clusters.

Table 3.1: Cluster Distance Change Measurement

	High(H)	Medium(M)	Low(L)
High(H)	0	1	2
Medium(M)	1	0	1
Low(L)	2	1	0

3. How well it preserved its rank-order in the new environment — measured by the rank change metric δ_k

$$\delta_k = \sum_{j=1}^{m} (Rank(o_j)^{env1} - Rank(o_j)^{env2}) \tag{3.5}$$

The smaller the value returned by Equation 3.5 the better.

3.4.3.2 Robustness of the Pareto Front

Robustness of the Pareto front between two environments is defined by four measures:

1. How close the front is to the optimal Pareto front?

2. How well its solutions maintain their objectives' clusters between the two environments — measured by calculating the mean cluster distance μ across all n solutions in the front:

$$\mu = \sum_{k=1}^{n} (\Delta(x_k)) \tag{3.6}$$

3. How well its solutions' ranks have remained closely correlated between the two environments — measured using a rank correlation test (for example, Spearman Rank Correlation [MBB99]). The Spearman test returns a number in the range $[-1, 1]$ known as the Spearman Coefficient (ρ). The closer the value is to 1, the stronger the correlation between the two rankings. A value of -1 implies negative correlation and a value of 0 implies independence between the two ranks.

$$\rho(obj_m) = 1 - \frac{6}{n(n^2-1)} \sum_{k=1}^{n} \delta_k^2 \tag{3.7}$$

where δ_k is the difference between the ranks of a solution x_k between environments.

4. How well the Pareto front maintained its diversity and uniform distribution — measured using the usual spacing (S) and hole-relative-size (HRS) metrics [CS03]. [3]

[3] A Hypervolume metric will be useful to determine optimality of the new front in comparison to an ideal front or one obtained using another optimisation algorithm.

3.5 Techniques to Enhance MOGP Robustness in Volatile Environments

In this section we present the technique used to improve the robustness of solutions in changing environments. Three techniques are outlined which constitute the basis for the experiments carried out in Chapter 5.

3.5.1 Rank-based Selection Bias

This technique is similar to the technique used to improve generalisation through the use of a selection data set, where the data sets are divided in to training, selection and out-of-sample validation data. In the general application of this approach, solutions that achieve good results in training are tested on a selection data set and only those that achieve good results on the selection data set are deemed good enough and then proceed to be evaluated on out-of-sample data. However, our approach is different in that the selection data set is used as a test for one aspect of robustness which is the ability of the solution to maintain its rank rather than a test for optimality. Also the relative success or failure of a solution on the selection data set leads to a corresponding change in its fitness values.

To implement this technique, the fitness of a solution in the MOGP population is altered as follows [4]:

- In each generation, and after the front has been identified:
  - Identify the ranks of all solutions;
  - Run every solution in a different environment and identify the rank of each solution in the new environment;
  - Assign a robustness value R to each solution based on how well the solution preserved its rank — the smaller the change, the better the robustness value.
  - The solution fitness value is incremented by the robustness value.
- Tournament selection is performed as usual. It will now prefer solutions which are: non-dominated, in less dense areas of the front, and which are more robust across the diverse training environments.

An interesting point here is how to combine the difference of ranks along all the dimensions. Since we have separate ranks for each objective dimension, we end up with m rank differences that need to be combined into one value. The rank differences have been normalised to a value in the range [0, 1], with 0 being the best robustness, and 1 the worst. In situations where a single metric is required, we choose to multiply these rank differences together, since

[4] The fitness is calculated as defined by the SPEA2 algorithm, and hence the modifications are done within this context.

this slightly biases in favour of solutions where all rank differences are very low. Hence, the robustness measure is:

$$R = \prod_{i=1}^{n} r_{oi} \tag{3.8}$$

The total fitness of the solution is then incremented by the robustness value, and is used for mating selection.

3.5.2 Diversity Enhancement

Diversity maintenance in Evolutionary Computation is essential to prevent premature convergence and improve generalisation. Also, the GP is known to suffer from the problem of pre mature convergence [BFN96] and bloat [MF01, SF98]. Larger trees are more prone to over fit the data and represent rules that are harder to understand. According to [EN02], a population of genetic programs is diverse if it contains samples of as many regions of the search space as possible. In general, diversity describes the amount of variety in the population, and variety can be seen from the point of view of the individuals structure (genotypic) or their performance (phenotypic), although Langdon [Lan96] argues that genotypic diversity is a sufficient measure of variety since a decrease in the number of unique individuals means a decrease in the number of unique fitness values as well. However, [BKK02] maintains that phenotypic diversity offers a measure of how much of the fitness landscape is actually covered.

The work of [TS99] on generalisation of GP – used for trading-rule discovery in the foreign exchange market – has found that smaller GP trees of depth two or three have led to better generalisation in the $Dollar - Yen$ and $Dollar - DM$ markets. The same result was also found in [AS03] when evolving technical trading rules for generating buy and sell decisions, where trees of depths of between 2 to 5 levels had, on average, outperformed larger trees. It was shown in [BFN96] that changing the balance of crossover and mutation in GP has a significant effect on the generalisation capability of the algorithm. Using a mutation rate of 50% yielded the best generalization results, but this decreases if the mutation rate is increased further than 50%. In addition, the probability of generating an outstanding run also increases by increasing the mutation rate. The beneficial role of mutation was attributed to the decrease in the number of introns. It was also noted that the effect of increasing the mutation rate is beneficial in the case of larger and more difficult problems. The authors believe that the improvement in generalisation was attributed to the increase in diversity and hence the GP suffered from less premature convergence. In addition they also show that higher mutation rates reduce the number of introns and hence the total size of the trees.

Proposed Diversity Enhancement

In this work, the underlying evolutionary algorithm is a Koza style GP in which no mutation is used. We investigate whether the generalisation of the MOGP algorithm used will benefit from

an increase in diversity in the same way as the standard GP does. To increase the diversity in the MOGP population, we have implemented two techniques:

- Used point mutation with 0.3 probability throughout the evolutionary cycle.
- Removed the duplicate individuals from the SPEA2 archive before selection, so that the archive contains only genotypic unique individuals. In effect, minimising the probability of crossover between two identical individuals.

3.5.3 Mating Restriction

Prior research into the effect of mating restrictions on evolutionary multiobjective algorithms has focused mostly on improving quality of the search and/or diversity of the solution set. The effect of mating restriction in the EA literature dates back to [Gol89] who suggested that the crossover between parents that are too different genotypically may hinder the search especially as the population starts to converge. [RS04] found that restricting the mating to be between a non-dominated individual and another individual that is dominated by it leads to an acceleration (albeit small) in the progress towards the Pareto front. In [DG89], it was found that recombination between individuals in different niches produces low fitness individuals, and hence a restriction was imposed to prevent mating between dissimilar parents. However, in [KR02] mating restriction was used to prevent mating between individuals that are too close together in an effort to aid diversity and help produce a better spread front. In [IS03], experiments were carried out to examine the effect of mating similar or dissimilar parents on small and large multiobjective test set problems. Again, the results varied: on small test problems, choosing dissimilar parents had improved the search ability; however, on large test problems, the search ability was improved through the choice of similar parents instead. The seemingly contradicting results may be due to (i) the large, complex problems having a large and diverse search space, where the evolutionary MO algorithm benefited from the pressure towards convergence through mating of similar parents, whereas (ii) convergence to one niche of the Pareto front happens too soon and the need for improved diversity increases in small problems.

In summary, the current research on mating restriction in evolutionary multiobjective can be divided into two main classes: mating of similar parents or mating of dissimilar parents. The former will speed up convergence and in some problems the quality of the solutions. On the other hand, mating of dissimilar parents will improve diversity, which is vitally important in the evolutionary multiobjective search. However, in this research we will investigate the effect of encouraging mating of similar parents on the performance of an evolutionary MO algorithm on out-of-sample data, which to our knowledge has not been carried out before.

3.5.3.1 Proposed Mating Restriction: Cluster-based Mating Restriction

Inspired by finding in [Has08] where it was shown that solutions evolved for each objectives-cluster have common characteristics that distinguish them from solutions in other clusters. We

believe that in this financial domain, the MOGP is discovering rules belonging to various niches (corresponding to the clusters). If this were actually the case, then by limiting the mating to parents belonging to the same cluster and hence sharing the same objectives characteristics we would further help this speciation. We will investigate the effect this special kind of similarity mating will have on one particular aspect of robustness which is the movement from one cluster to the others between training and validation environments.

In the SPEA2 [ZLT02] algorithm, individuals are compared based on Pareto dominance and the non-dominated solutions of each generation are placed in a separate archive. Furthermore, selection of parents for mating is limited to this archive. We have simulated a mating restriction technique whereby mating is restricted to parents belonging to the same cluster. Parents are selected using binary tournament selection of size 7 with replacement, exactly as in standard SPEA2. The difference is that the second parent is accepted only if it belongs to the same cluster as the first parent. If this is not the case, we attempt to reselect the second parent up to a maximum of four more times. If we fail to select a parent belonging to the same cluster after five trials, the first parent crosses over with a copy of itself. In this way, a parent never mates with another parent from outside its cluster.

3.6 Optimum Tracking in Dynamic Financial Environments

3.6.1 Introduction

The MOGP will evolve a set of equations that describe the relationship between the factors of input data and the corresponding attractiveness of a stock. When the environment changes, the change is due to one or more of the following:

- Change in the values of any of the factors considered which leads to the stock whose values changed to either move from the top quartile to the bottom quartile or vice versa. Equation evolved is still valid; system should continue to perform favourably.

- Change in the relationship between the factors (equation). Hence the system needs to discover the new equation. The system should be able to recover after sometime, maybe with the aid of some alterations to the basic MOO algorithm. The number of data points needed and the number of generations required for the system to adapt will depend on the severity of the change: how much the equations have changed and whether the change was abrupt or happened slowly over a period of time.

- Change in some external factor (whether it was something related to company performance or some global factor). For the purpose of our study both types of factors will have the same effect on the system and hence will be treated equally. The system will find it harder to recover or may not be able to recover at all.

3.6.2 Proposed Measure for Severity of Change in Dynamic Environments

After a change is detected, a measure of the severity of change is required. The accuracy of this estimation will influence the technique used to adapt to the change. If the severity is deemed low, the new optimum is possibly close to the old optimum and using the old population as a base for optimisation with the addition of some diversity could be enough to locate the new optimum. If the two optimums are known, then a simple measure of distances between them will be a good indicator of the severity of change. However, since the actual optimum is in practice not known in advance, proxies need to be used. We propose two measures for the severity of change as follows:

1. **Shape**: uses clustering techniques to divide the Pareto front solutions into three clusters; one representing the solutions that are low on all objectives (LL); the second representing solutions that are high on all objectives (HH); and the third being for solutions with medium values on all objectives (MM). The algorithm maintains and updates the centroids of the clusters. The distance between the corresponding centroids (in the old and the new environments) is measured and if it exceeds a certain threshold, then intervention in needed to help the algorithm adapt to the change. These three numbers (the movements of the three centroids) together provide a proxy for the position of the Pareto front in the search space. In addition, because we are measuring the movements of the centroids of three separate clusters, this measure is also an indication of the changing ***s*hape** of the front, and it shows which portion of the front moved the most or the least, or whether the whole front moved uniformly in space.

2. **Shuffle**: uses the Spearman correlation coefficient between the ranks of solution on the front of the old environment and their ranks when the environment first changes (before any training on the new front occurs). This measure assumes that a higher correlation value indicates stability of performance (notice that since we are using the ranks, this measure is independent of the actual objective values of the solutions, so the objective values may themselves change, but if the solutions ranks relative to each other remain relatively high, then the solutions are still valid). This measure gives an indication of the degree of ***s*huffle** that occurred on the front when the change first happened. We measure the correlation for each objective separately.

3.7 Summary

In this chapter we examined the behaviour of the MOGP solutions in an unseen environment and showed that individual solutions are prone to switching their perceived risk-return trade-off when applied in a new environment. We explained how this behaviour is particular to multiobjective problems, and why it is a serious issue that needs to be taken into account and hence measured quantitatively. We developed suitable metrics to measure the robustness of multiobjective solutions and the Pareto front when evaluated in unseen environments and

provided the required definitions of what constitutes a robust solution and a robust Pareto front in a dynamic environment.

The chapter then gives details on techniques proposed to improve the robustness of MOGP solutions. Three techniques were proposed: selection bias, increasing diversity, and cluster-based mating restriction. The metrics developed and the techniques to improve the robustness are used in experiments of Sections 5.3.3 and 5.3.4.

Finally the chapter briefly discusses the problem of optimum tracking in a dynamic financial environment and outlines two suggested techniques to indicate the severity of change in a dynamic environment. These two techniques are used in the experiment of Section 5.4.

In the next chapter (Chapter 4) we explain the portfolio optimisation problem in detail, present the system architecture and experiments design. Next, in Chapter 5 we present the experiments and results.

Chapter 4

System Architecture and Design of Experiments

4.1 Introduction

This chapter describes the specification of the real world portfolio optimisation problem, presents the system architecture and explains the experimental setup. The chapter is organised as follows:

- Section 4.2 introduces the portfolio management and stock selection problem, multifactor models, and performance measures for a fund portfolio.
- Section 4.3 presents the historical financial data used in the experiments for training and validation.
- Section 4.4 explains the design of our system architecture with its two main parts: the Investment Simulator, and the Multiobjective GP.
- Section 4.5 introduces the performance criteria employed, and the methods used for statistical analysis.
- Section 4.6 discusses some implementation details: parameters and operators of the algorithms used; the observed effect of data normalisation; extensions to the standard algorithms; as well as some general notes on the experiments design.

4.2 Real World Problem of Financial Portfolio optimisation with Multiple Objectives

4.2.1 Introduction

A portfolio is a collection of investments or assets held by an institution or a private individual. In this research we focus on the stock market and hence, all the assets are assumed to be stocks (securities) [1]. The basic problem of portfolio selection is the choice of an optimum set of n assets

[1] In the general case, a portfolio is possibly a collection of stocks, bonds and cash

to include in the portfolio and the distribution of investor's wealth among them such that the objectives sought by holding the portfolio are maximized. The solution to the problem is:

- The specification of the securities to constitute the portfolio.
- The specification of the proportions of wealth invested in each stock.

Once the initial selection of stocks is decided upon, one of two strategies is implemented. In the first, the portfolio is held for a specific period of time then sold when a profit can be made. In the second, the portfolio is frequently re-balanced, where at intervals, some stocks are sold and others are bought.

Usually the owner of the portfolio is interested in maximising his portfolio return, as well as reducing his exposure to risk. The Modern Portfolio Theory (MPT) by Harry Markowitz [Mar52] had established the well-known practice of spreading investments across several assets (diversification), such that certain types of risk are reduced. Prior to Markowitz's work, investors focused on assessing the risks and rewards of individual securities and constructed their portfolios accordingly. Then, Markowitz proposed that investors should focus on selecting *portfolios* based on their overall risk–reward characteristics instead of merely compiling portfolios from securities that each had individually attractive risk-reward characteristics. This is done by means of two techniques; increasing the number of assets in the portfolio to achieve diversification, and selecting the assets such that they have low correlation to each other which will in turn decrease the portfolio's standard deviation. Markowitz argued that for any given level of expected return, a rational investor would choose the portfolio with minimum risk amongst the set of all portfolios possible (i.e. investors are risk–averse) [FFK06]. The set of possible portfolios to be constructed is called the *feasible set*, and the set of minimum risk portfolios corresponding to desired levels of expected returns is called the *mean–variance efficient frontier*. [FFK06, p.20]. Strategies for asset selection and portfolio management are the subject of a vast area of research in finance and economics known as portfolio optimisation.

4.2.2 Problem Definition

Definition: Portfolio Optimisation Problem

Consider a fixed sum of money to be invested in n securities selected from a universe of securities. Let there be a beginning and an end of the holding period. Also, let w_i be the proportion of the fixed sum to be invested in the $i-th$ security. Being proportions, $\sum_{i=1}^{n} w_i = 1$. Markowitz [Mar52] assumed that the objectives of the investor are maximising the return on investment and minimising the associated risk. He suggested that risk should be measured by the variance of returns – the average squared deviation around the expected return, where the expected return of a security is the expected price change plus any additional income over the investment period. Hence, solving the problem requires the choice of the n assets, the specification of the vector $w = (w_1, w_2, ..., w_n)$ and the simultaneous satisfaction of:

1. Maximising the return:

$$\mu_p = \sum_{i=1}^{n} w_i \mu_i \tag{4.1}$$

2. Minimising the standard deviation:

$$\sigma_p = \sum_{i=1}^{n} \sum_{j=1}^{n} w_i w_j \sigma_{ij} \tag{4.2}$$

where n is the number of securities in portfolio, w_i is the relative amount invested in security i ,and $\sum_{i=1}^{n} w_i = 1$. μ_p is the expected portfolio return, σ_p is the portfolio variance (i.e risk), which is the average squared deviation of the return from its expected mean value, and σ_{ij} is the covariance between assets i and j. It is assumed that the covariance Matrix $\sigma_{ij}$is given by Equation 4.3.

$$\sigma_{ij} = \begin{bmatrix} \sigma_{11} & \dots & \sigma_{1n} \\ . & & . \\ . & & . \\ . & & . \\ \sigma_{n1} & \dots & \sigma_{nn} \end{bmatrix} \tag{4.3}$$

Equations 4.1 and 4.2 are solved by a set of points that constitute what is known as the *efficient frontier* of the problem. The set of points define a curve similar to that of Figure 4.1 plotted in the risk-return solution space of all possible portfolios. The points that constitute the curve represent portfolios for which there is the highest expected return given a certain amount of risk, or the minimum amount of risk given a certain expected return [Mar52].

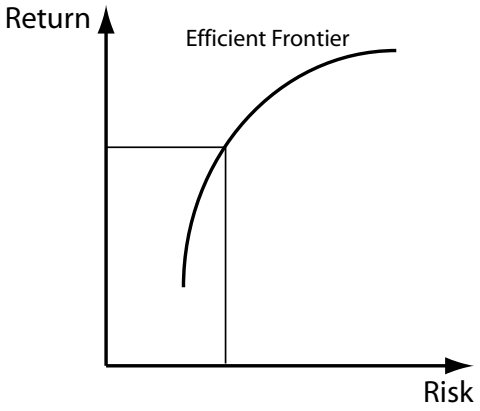


Figure 4.1: Efficient frontier in standard-deviation, expected-return space

4.2.3 Multiple Objectives of Portfolio optimisation

In real life investment in portfolios, things become more complicated than the definition above. First, the universe of assets to choose from: even if we only consider one market to invest in,

the number of stocks to choose from is quite large. Second, in practice, there is not just one beginning and end for the holding period: we are usually interested in multiple periods of buying and selling of assets. Third, the initial cash to invest in a portfolio is limited, and proper diversification according to the MPT may not be realisable. Fourth, the objectives of investment may go beyond those of maximising return and minimising risk to include more objectives.

In its simplest from the portfolio optimisation problem has the maximisation of return as the single objective of the investor. However, it has long been realised that return is the reward for taking on more risk and that actually the optimisation problem has two conflicting objectives to satisfy, thus classifying it as multiobjective problem. Furthermore, investors may still have other objectives to fulfil by holding a financial portfolio. Many researchers [CAS04, LPM03, SQ05] are now realising that investors are interested in monitoring more objectives than the risk and return. Steuer [SQ05] distinguishes between two types of investors: the standard investor, whose utility function takes only the single argument of portfolio return, and the non-standard investor, whose utility functions are allowed to take on additional arguments, such as dividends, amount invested in Research and Development, liquidity, a certain portfolio return over that of a bench mark, amount of short selling, or more than one measure of risk examination. In the following we explain in more detail what is meant by two such additional objectives: risk and portfolio liquidity.

Examples of Investor Objectives:

1. **Risk:** Defining what is meant by risk constitutes a large area of research in economics and finance. Traditionally there are two major types of risk measures: dispersion risk and downside risk [FFK06, pp.116-120]. Dispersion measures risk as the amount of uncertainty of returns. Hence it qualifies both positive and negative deviations from the mean as equally risky. The most well known dispersion measure is that used in the classical portfolio theory and it measures the standard deviation of returns as in Equation 4.2 in which risk is equated to volatility. In the downside risk measures only the risk that the return is less than the mean (or a certain acceptable level) is penalised. In this model, the investor selects a target return, and the risk is defined as return dropping below this target value .

   Another popular risk measure is *Value at Risk (VaR)* which evaluates the probability (usually 1% or 5%) that a portfolio makes a profit/loss above/under a specified threshold value in a given time period . The VaR gives a measure of the predicted maximum loss at a specified predictability level.

2. **Liquidity of Portfolio** Defining liquidity is not an easy task, mainly because the term refers to different things for different people and because the concept is inherently multidimensional. Measures of market liquidity include: frequency of trading, and ratio of trading volume to total number of shares issued; that is the number of tradable shares.

Aspects of Liquidity [LS03]: Liquid markets exhibit the following characteristics:

(a) Trading Time (immediacy): The ability to execute and settle a transaction immediately at the prevailing price. The waiting time between subsequent trades, or the number of trades per time unit, are measures for trading time.

(b) Tightness: The ability to buy and to sell an asset at about the same price at the same time. Measures for tightness are the different versions of the spread between the buy and sell price or low transaction cost.

(c) Depth: Existence of abundant orders at or around the current security price. Market depth can be measured by the order ratio, the trading volume or the flow ratio.

(d) Resiliency: The ability to buy or sell a certain amount of an asset with little influence on the quoted price. In resilient markets, new orders flow quickly to correct order imbalances that tend to move prices away from what is expected by the fundamentals.

4.2.4 Multifactor Models

Currently there are two widely accepted theories that provide a theoretical foundation for computing the trade off between risk and return and deriving the fair price of an asset. These are: (i) The Capital Asset Pricing Model (CAPM), and (ii) the Arbitrage Pricing Theory (APT) The Capital Asset Pricing Model CAPM [Sha64, Sha94] is a model which derives the required return for an asset in the market given the risk free rate and the risk of the market as a whole:

$$E(R_i) = R_f + \beta_i(E(R_m) - R_f) \tag{4.4}$$

,and

$$\beta_i = \frac{COV(R_i, R_m)}{VAR(R_m)} \tag{4.5}$$

where $E(R_m)$ is the expected return of the market, R_f is the risk free rate, and β_i is the measure of the asset sensitivity to movement in the overall market and hence how risky it is.

Once the expected return is calculated, the correct price for the asset can be established by discounting future cash flows of the asset to their present value using this rate. Hence, the CAPM is effectively establishing that the fair asset price is a function of a single risk factor, β, the sensitivity of the asset returns to the market return, also known as the systematic risk, or un-diversifiable risk.

The APT [Ros76] on the other hand holds that the expected return of a financial asset i can be modelled as a linear function of k macro-economic risk factors and the specific risk ϵ_i, where sensitivity to each factor is represented by its own beta coefficient. The CAPM is considered a special case of the APT with a single risk factor, and hence the general name for APT as the multifactor model. Both the CAPM and the APT agree that proper diversification

of the portfolio should eliminate the unsystematic risk. However, all stocks are still influenced by other sources of risk, and the pricing of a stock should reflect its level of exposure to the systematic, un-diversifiable risks.

$$E(R_i) = R_f + \beta_{i1}F_1 + \beta_{i2}F_2 + ...\beta_{ik}F_k + \epsilon_i \tag{4.6}$$

where, F_i is the macroeconomic factor, β_{ik} is the sensitivity of the asset to factor k, and ϵ_i is the risky asset's unsystematic (diversifiable risk).

The APT theory does not specify the risk factors affecting the pricing model, but rather it provides a general asset pricing model where there exists more than one source of risk. When the APT is used to build portfolios, the specific factors that an investor perceives as more influential on his choice of assets can be used to model the APT and help identify above or below the correct price of assets and hence his buy and sell decisions.

The common practice of fund managers is to use the price time series to measure the technical and fundamental factors' correlation with the security return. The factor models are then used to predict future behaviour, and in conjunction with other tools, to construct portfolios.

Since the APT does not name the risk factors, there are potentially numerous models that are built on the basis of the APT. Asset pricing models with multiple risk factors are called *multifactor models* and a lot of research is conducted to identify the risk factors involved. A *multifactor model* attempts to isolate an asset's sensitivities to the risk factors usually to predict returns, identify risk sensitivities and price assets to spot opportunities for abnormal returns. The general form of the multifactor models is that of the Equation 4.6 and hence it is often assumed that the relation is linear as depicted in the APT.

According to [FFK06, pp.242-246] there are three different types of multifactor models: macro-economic factor models, fundamental factor models, and statistical factor model:

- **Macro-economic Factor Models**: Uses economic time series like interest rates, investor confidence, and inflation as the factors in the multifactor asset return equation.

- **Fundamental Factor Models**: These models hypothesize that the risk factors are fundamental and technical indicators. The fundamental factors are economic factors that describe the company performance. They are firm or asset specific attributes such as firm size, dividend yield, and industry classification. Technical factors are factors that are derived from the stocks observed returns and price fluctuations, such as the moving average and price volatility indicators. An example of this type of factor models is the Fama-French model of 3 factors (see below).

- **Statistical Factor Models**: In these models, historical and cross-sectional data on stock returns are put into a statistical model, whose goal is to explain the observed returns with factors that are statistical measures of linear return combinations and are uncorrelated

with each other. The main task afterwards is to understand the economic meaning of the statistically derived factors produced by the model.

Examples of Risk Factors in Linear Multifactor Models

Fama and French [FF93a, FF96] studied the risk factors that affect the average returns of American stocks. They found that the corporate size (small capitalisation) and high book-equity-to-market (value stocks) can effectively account for the variations empirically found in the average returns of individual stocks. This outcome basically rejects the single risk factor of the CAPM, and implies that more risk factors are in play as the APT suggests. They concluded that these two undiversfiable sources of risk sensitivities are not captured by the CAPM, and proposed a three factor model for expected returns. The three factor model is backed up by empirical findings rather than economic reasoning.

The model says that the expected return on a portfolio in excess of the risk free rate is explained by the sensitivity of its return to three factors:

1. The excess return on a broad market portfolio;
2. The difference between the return on a portfolio of small stocks and the return on a portfolio of large stocks (SMB);
3. The difference between the return on a portfolio of high-book-to-market stocks and the return on a portfolio of low-book-to-market stocks (HML). The model is represented in Equation 4.7.

$$R_p = R_f + \beta_i(E(R_m) - R_f) + S_p(SMB) + H_p(HML) + \epsilon_i, \tag{4.7}$$

Where S_p is the coefficient loading for the excess average return of equities with small size class over large size firm equities. H_p is the coefficient loading for the excess average returns of equities with high book-to-market equity over those with low book-to-market.

The three factor model has gained much popularity and many other researchers have found evidence to support the model in different markets. Examples include:

- [Ban81] found that future returns of stocks with small firm size are higher than would be expected if the market portfolio was mean-variance efficient.
- [RRL85, Sta80] have found evidence that stocks with a high book-to-market ratio also have returns higher than what can explained through CAPM (value stocks outperforming growth stocks).
- Basu in [Bas97] found a significant effect of price-earning-ratio, where firms with low PE ratios have higher sample returns and vice-versa. He observes low P/E securities outperforming high P/E securities by more than seven percent per year. Basu regards his

results as an indication of a market inefficiency: "Securities trading at different multiples of earnings, on average, seem to have been inappropriately priced vis-a-vis one another, and opportunities for earning "abnormal" returns were afforded to investors."

- [MP02] tested the three-factor model on the stock markets of Australia, Canada, Germany, France, Japan, the UK and the US; the size effect and the value premium survive for all the countries examined and they conclude that the size and BE/ME effects are international in character.

Non-Linear Multifactor Models

Some researches have cast some doubt on the linearity assumption of the multifactor pricing models. Bansal and Viswanathan [BV93] found empirical results using size-based portfolio returns and bond yields that reject the linearity of the APT. They argue that a nonlinear APT with two factors: the market return and the one-period yield in the next period was more capable of explaining variations (especially small firms returns). Dittmar [Dit02] investigated a a pricing kernel that approximates a Taylor series expansion and the result is a polynomial in aggregate wealth. He found that both a quadratic and a cubic pricing kernel were admissible for the cross section of industry profiles, whereas the CAPM and the Fama-French linear models were not. This research is one of the few that proved the superiority of a nonlinear model to the Fama-French models. Kanas in [KY01] compared the performance of a linear and a non linear neural network for the forecast of stock returns and found that the nonlinear neural networks produced more accurate results. In [Kan03], the authors examined and compared the out-of-sample forecast performance of two parametric (linear) and two non-parametric (nonlinear) forecast models of stock returns. The parametric models included the standard regime switching and the Markov regime switching, whereas the non-parametric were the nearest-neighbour and the artificial neural network models. They found that, in terms of accuracy, the Markov and the artificial neural network models produce at least as accurate forecasts as the other models, while the Markov model outperforms all the others on encompassing. Dhar et. al [DC01] attributed the nonlinearities observed in empirical studies to noise inherent in the financial markets. They cite as an example the effect of announcing an above-expected-earning on the price of a stock, and how the price can remain unaffected if the earning is exceeded marginally, but reacts very strongly if earning exceeds a high threshold, and then the relationship increases rapidly thereafter.

4.2.5 Measuring Fund Performance

4.2.5.1 Sharpe and Sortino Ratios

The Sharpe ratio is one of the most widely used statistics in financial analysis. It is a simple measure for risk-adjusted performance; it measures the average excess returns (above risk free return rate) of a stock or a portfolio relative to its volatility as measured by its standard deviation. The same assumptions of the CAPM hold: (i) normally distributed returns, and (ii)

mean-variance preferences of investors. It was developed by William Sharpe in 1964 [Sha64], and it's most common form is:

$$Sharpe = \frac{R_p - R_f}{\sigma_p}, \tag{4.8}$$

where R_p is the expected average return of the portfolio, R_f is the risk-free return (usually government bond return rate), and σ_p is the portfolio standard deviation.

In essence it is a measure of the reward-to-variability ratio, or a measure of returns per unit of volatility, and if we consider volatility to be a measure of the riskiness of the asset/portfolio then it is a measure of returns per unit of risk. Since it takes both return and risk into consideration, it enables a one to one comparison between stocks or portfolios, giving an indication of whether the returns are due to smart investment or excess risk. The greater the Sharpe ratio, the better the performance of the portfolio analysed. It is often assumed that the Sharpe ratio is a positive value since a negative value will only result if the return of a portfolio is less than the risk-less asset return.

Using standard deviation as a measure of risk carries a major practical drawback. It implies that better-than-expected returns are just as risky as worse-than-expected returns. In reality investors would welcome better than anticipated returns and would only beware of investments that go below the predicted average. The Sortino ratio overcomes this weakness of the Sharpe ratio by considering downside risk only, that is to say only downward price volatility.

$$Sortino = \frac{R_p - R_f}{\varrho_p} \tag{4.9}$$

The Sortino ratio formula is very similar to that of the Sharpe ratio as Equation 4.9 shows, with the exception that the denominator ϱ is the downside standard deviation.

4.2.5.2 Other Measures of Fund Performance

The following are some measures used in [Cov07] to measure a fund performance:

- Net profit, annualised profit
- Number of trades
- Maximum drawdown
- Percentage of winning months,and percentage of losing months

4.3 Financial Data and Economic Factors

4.3.1 Financial Data Used in Experiments

This research was conducted on historical data from the London Stock Exchange market, the FTSE100, of 80 months from May 1999 until December 2005. The stock universe consisted

of 82 stocks [2]. For these 82 stocks, the factors describing the represented companies' data performances were downloaded from Reuters. The total period was divided into training and validation. For example, in some of the experiments the data was divided such that the training data covered 60 months from May 1999 to April 2004, and for validation, out of sample, the data was that of the following 20 months from May 2004 to December 2005.

4.3.2 Fundamental and Technical Factors

The potential number of fundamental and technical factors that could be used to describe performance or to price the stock is vast. In this research we focused on a set of 22 fundamental and technical factors suggested by financial experts and included the factors used in some well known pricing models or that were in common use by technical analysts. The factors used and their definitions are presented in Table 4.1. A sample extraction of one company's data (normalised) is shown in Table 4.2.

The choice of this particular set was a subjective decision based on: first, consultations with experts in the field on which factors they would look at when evaluating a stock; although different experts have different and some times contradicting opinions about which factors are actually influential, second, we have tried to include the factors that have previously been found to have an effect on return beyond that which could be explained by the CAPM; in Section 4.2.4 we briefly introduced examples of factors used in factor models in the finance literature. We have included in our set *Capitalisation* -indicator of company size-, and *Price-To-Book-Ratio (Book-to-market)* which were used in the highly regarded Three-factor model (see Section 4.2.4), the traded *Volume* and *Moving Average indicators*, often used in technical analysis of stocks, as well as information on stock returns as indicated by *Dividends*.

4.3.3 Data Preprocessing

Some preprocessing of the data was performed before using it in the experiments.

1. The factors in Table 4.1 are cited monthly [3]. However, in reality some of these factors are measured monthly, some daily and some are only measured yearly for any company. For example the factors: *Divided Yield, Earning on Equity, DVPS, Market Capitalisation, Change ROE, Revenue Growth, Cash-Share Yield, Adjusted EPS, One Year Earning, Equity-Asset, CPS-DPS* are all yearly factors. Hence for any one year, its twelve month value corresponding to these factors will be constant, while the daily factors *30-day moving Average, Close, Change Moving Average, Volatility* and the monthly factors *Price momentum, Price-Cash, Book-To-Price, PE-Ratio* will have their monthly values possibly changing from month to month.

2. Factors used in the data set are all numerical values that take real values: some have

[2] The 82 stocks are those that have been in the FTSE100 for the whole time period investigated; i.e. companies that merged, split or fall out of the FTSE100 during those 80 months were excluded.

[3] The price as well as the other daily factors are quoted for the first day of the month.

Table 4.1: Definition of Financial and Economic Factors Used

Close Price	Previous day last reported trade price
Price Momentum	Price per USD price change
Price-Cash Ratio	Compares stock price with cash flow from operations per outstanding shares
Volume	Total sum of shares that have traded in the security for the current or most recent days on its primary trading market place
Price to Book Ratio	Price of stock is divided by reported book value the of the issuing firm
Price-Earnings Ratio	Financial Ratio that compares stock price with earnings per share
30-Day moving average	Mean of the previous 30 days' closing prices
Dividend yield	The Company's annual dividend payments divided by its market capitalization, or the dividend per share divided by the price per share
Volatility	The degree of price fluctuations of the stock - expressed by variance or standard deviation
Earning on Equity	Net income divided by share holders equity. Measure of the net income a firm earns as a percent of stockholders' investment
Market capitalisation	Price per share multiplied by the total number of shares outstanding
Changes ROE	return on equity (current year) - return on equity (previous year)
BVPS	A measure to determine the level of safety associated with each individual share after all debts are paid. It represents the amount of money that the holder of a share would receive if the stock was liquidated
Revenue Growth	The rate at which revenue has increased annually. Can be negative. $= \frac{current\ year's\ revenues}{previous\ year's\ revenues} - 1 * 100$
Cash Share Yield	The ratio of the annual return from an investment, through dividend and capital gains, to the amount invested
Adjusted Dividend Yield	A stocks return calculated using the capital gains and dividends
Earnings Per Share (EPS)	Net income for a period is divided by the total number of shares outstanding
Adjusted EPS	Calculates earning per share using only normal trading profits and returns made from exceptional items and on offs. These are excluded as they don't help investors estimate future cash flows
1Y Earn Growth Momentum	$\frac{last\ year\ EPS - previous\ year\ EPS}{absolute\ previous\ year\ EPS} * 100$
Equity-Asset	Total assets divided by shareholder equity
Altman Z-Factor	The technique uses a statistical technique to predict the probability of a company's failure
CPS-DPS	Ratio of cash to debt per share

Table 4.2: A sample of Company Data (BT)

Date	Org Close	Close	Momen-tum	Volume	Price toCash	Price toBook
03-05	205.5	0.054	0.472	0.405	0.527	0.071
04-05	199.75	0.047	0.444	0.265	0.506	0.071
05-05	213.25	0.062	0.639	0.357	0.653	0.070
06-05	230	0.081	0.661	0.578	0.670	0.0692
07-05	227.5	0.078	0.479	0.387	0.533	0.069
08-05	215.5	0.065	0.394	0.458	0.468	0.070
09-05	222.25	0.072	0.565	0.351	0.598	0.0698
10-05	213	0.062	0.416	0.415	0.485	0.070
11-05	213.5	0.063	0.506	0.557	0.553	0.070
12-05	222.75	0.073	0.589	0.281	0.616	0.069

positive values and some negative values, and the numerical ranges vary. As the numerical range varies considerably, all the data have been (individually) normalised[4] to the range [0,1] in case of the data being positive and to the range [-1,1] in case of negative values[5]. In addition, an extra column has been added containing the value for the original non-normalised price of the stock. However, this column was not used as input to the optimisation problem, it was only used for calculations and statistics performed through the experiment and the analysis.

4.4 System Architecture

Our system consists of a simulation of an investment strategy, as well as an embedded MOGP for trading decision making.

4.4.1 The Investment Simulator

4.4.1.1 Investment Strategy

The investment strategy employed is inspired by real world fund management practises. The portfolio held consists of one cash line (GBP) and has a fixed cardinality of $n = 25$ stocks. The amount to be invested is $C_o = £1,000,000$. The initial portfolio value is C_o in cash with no stock holdings. After that, the portfolio will consist of n securities, and the current cash holding will be denoted by C, where we try to keep C less than or equal to a maximum bound $C_{max} = 3\%$ of the total fund value. S is the universe of equities, S_n is the set of securities held in the portfolio, with each stock donated by S_{ni}. For all buying and selling decisions in any day, it is assumed that we can trade at the opening price of that day. During the holding period, interest received on cash holdings is ignored. Also, income from dividends is not included in the return calculations.

[4]The reader is reffered to Section 4.6.1 for a discussion on the normalisation technique.

[5]A positive effect on GP code bloat was observed when the normalised data was used (versus the raw data), where the solutions' sizes were significantly smaller. However, this effect was not subjected to any further rigorous study.

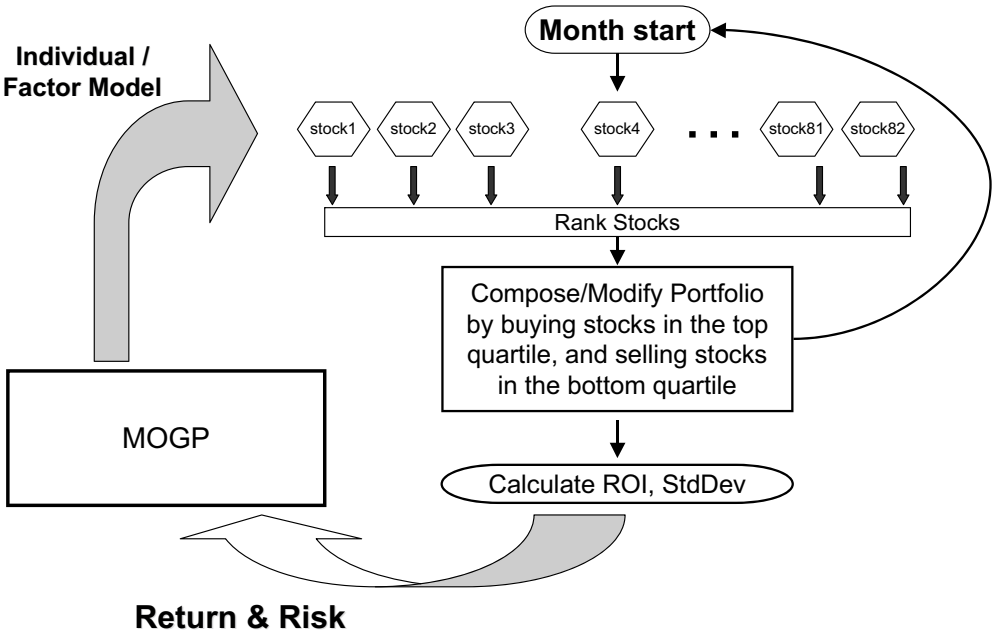


Figure 4.2: System Architecture

For the duration of the holding period, we execute the following steps: at the beginning of every month, we calculate the attractiveness of each stock in S according to the non-linear factor model examined, and sort the stocks according to their attractiveness. A sell decision is taken if any of the stocks we currently hold falls in the bottom quartile of the rank. We then start buying; if the number of stocks currently in the portfolio is less than n and $C > C_{max}$, then we need to bring it up to n, by buying stocks from the top quartile, starting with the most attractive. The proportion to be invested in each stock is C_i, and is calculated as:

$$C_i = \frac{C - C_{max}}{n - |S_n|}, \tag{4.10}$$

and

$$C_i \leq 4\% \ \ of\ total\ fund\ value \tag{4.11}$$

If we still have cash amounting to more than C_{max} of the total fund value, and there are some stock holdings with less than 4% of the total fund value, then we use all remaining cash to bring each of these stock holdings up to 4% or at least up to the maximum that the extra cash allows for.

4.4.1.2 Constraints and Additional Costs

In our problem formulation, we have included several realistic constraints and additional costs. The constraints are: portfolio cardinality, lower and upper bounds on investment per stock, and minimum cash holding. In particular:

- *Long only Constraint*
  Short selling is not allowed in our model. Hence, the weight invested in any asset is always positive. This is a frequently used constraint, as many institutional fund are prohibited from short selling [FFK06], pp.100.

- *Cardinality Constraint*
  This is the restriction on the number of assets in the portfolio to a number k significantly less than the number of assets in the investment universe. In our model $k = 25$ and the minimum holding at any point is also 25. This is in accordance with the observation in [FFK06, pp.107] that fund managers often aim to minimise the number of assets in their portfolio and at the same time make sure that their holding is larger than a certain threshold. Note that it is established that although diversification through increasing the number of assets helps to lower the standard deviation, this effect is however limited to a certain threshold, and risk cannot be totally eliminated [FFK06, pp.28]

- *Transaction Costs*
  Transaction costs are costs incurred when buying or selling securities in the form of, for example, brokerage commissions which are fees paid to brokers to execute trades, bid-ask spreads (distance between the quoted sell and buy order, and it is the immediate transaction cost charged by the market, taxes (capital gain taxes and tax on dividends), and market impact cost [FFK06, pp 51-60]). The transaction costs in effect would be a factor in deciding the frequency of trading. If a fund manager is not careful, the transaction costs can severely affect his returns. Therefore it is expected that the inclusion of transaction costs will lead to a reduction in the number of trades. However, the standard Markowitz model for asset allocation ignores the transaction costs. To take the transaction cost into account, there are several models. One model proposed in [HRM00], where a fixed sum is charged if the sum of money invested exceeds a certain boundary. Another example is found in [LLL05] where a fixed transaction cost is imposed according to the total amount of investment capital. In this case, the transaction cost function can be plotted as a step function with two or more constant values. In our model, we adopted a fixed high value of 2%, regardless of the transaction value, deducted following each trade.

With the addition of the cardinality constraint, minimum holding and transaction costs, no exact method exists for solving the portfolio optimisation problem, otherwise it can be solved in an exact manner by quadratic programming [TGC07]. Also, formulating the problem with

these constraints may lead to a Pareto front that is discontinuous or non-convex [FFK06, pp.110], [CMB+98], making the problem harder to solve by a single objective evolutionary algorithm with a fitness function defined as the linear aggregation of objectives [TGC07].

Algorithm 1 SPEA2 Algorithm [ZLT02]

Generation number = 0

Generate a random (GP) population of size N and an empty archive of size M

repeat

For every individual i in the population and the archive, find:

1) The strength of the individual, equal to the number of individuals that dominate it in the archive population.

$S(i) = COUNT(j)$, where $j \in$ set of individuals dominated by i.

2) The raw fitness of the individual, equal to the sum of the strengths of the individuals dominated by i.

$RawF(i) = \sum S(j)$, where $j \in$ set of individuals dominates i

Find the non dominated individuals($RawF = 0$), add to the archive.

If the $archivesize < M$, then fill the archive with best dominated individuals (lower RawF).

If the $archivesize > M$, *truncate* archive by eliminating individuals that have the minimum distance to some other individual.

For all individuals in the archive, calculate the density:

$D(i) = \frac{1}{distance\ to\ k-th\ neighbor + 2}$

where $k = \sqrt{N + M}$

Fitness of the archive individuals is the sum of the raw fitness and the density information

$F(i) = RawF(i) + D(i)$

Copy archive to next generation

Select parents using tournament selection – limited to the archive individuals (fitness minimization is assumed)

Apply crossover, reproduction and mutation to selected parents to form next generation population.

Increment generation number

until Stopping criteria is reached (Max number of generations)

4.4.2 The Multiobjective GP

The MOGP is the machine learning algorithm used to generate buying and selling decisions.It is a multiobjective algorithm that uses Genetic Programming as its Evolutionary Computation technique. The GP part influences the choice of individual representation, and hence the generation method, and the reproduction techniques, while the MO nature of the problem dictates the performance calculation, fitness assignment and the selection technique.

The choice of GP over other EA techniques stems from the fact that individuals of the GP

population are trees, which is a general representation that allows flexibility in the structures evolved. Also, the GP method does not dictate before hand the exact size of the genome, which is useful, since the equation size is actually an unknown [6]. Another important advantage of the GP approach is that it generates the rules relating the data describing stock performance to the likelihood of the stock generating a good risk-adjusted-return. Thus the solutions evolved can give a financial analyst an insight into what variables and functions are important in pricing stocks. The choice of the function set for the GP representation is by no means a trivial task: there is no automatic way to deterministically decide a priori on the relevant functions required. Presenting the GP with a large function set and hoping that the algorithm will eliminate the irrelevant ones is a good starting point, however, this could lead to a blow up in the size of the search space, to **bloat** in the size of the solutions (an increase in the size of solutions with generations, with much of the code having no effect on fitness (introns)), and in general a significant slow down in the evolution.

The MOGP starts by generating a random population of trees (equations) representing factor models. The investment simulator will use these factor models to rate the attractiveness of stocks and generate buying and selling decisions. Based on how well each factor model performed (in terms of the objectives sought), the objectives values achieved are calculated and a Pareto-fitness value is assigned (according to the MO algorithm) to the factor model currently considered. Using this fitness value, reproduction is performed on the GP population and evolution continues until the stopping criteria are reached. The method of tree generation was the ramped half and half [Koz92]. The terminal set for the tree consisted of the financial factors of Table 4.1, in addition to random constants. The functions set in the linear experiments are addition and subtraction. In the non linear experiments, the following set of operations is used: addition, subtraction, multiplication, division, power 2, and power 3.

The multiobjective algorithm used in all experiments is SPEA2 [ZLT02], which is a good and popular general-purpose MO algorithms. Furthermore, in a recent paper, [SKN07] demonstrated that SPEA2 had the best performance on a portfolio optimisation problem with real life constraints. They compared the SPEA2 performance with that of NSGAII [DPAM00], MOGA [FF93b], and VEGA [Sch85], and found that the SPEA2 had the best performance in terms of quality of solutions and their distribution even with a small number of generations.

The implementation language was in Java, and is based on the ECJ package [L+15]. The MOGP had two conflicting objectives to satisfy; return maximisation and risk minimization. Return is defined as the expected average annualised return, and the risk is the standard deviation of the average annualised return. We ran simulations with population sizes of 800, 400, 200, and 100. The operators used were a crossover, mutation and reproduction:

- **Crossover:** performs a strongly-typed version of Koza "Subtree Crossover". Two indi-

[6] Although it could impose limits on the minimum and maximum size of the trees, and in this case the trees are allowed to grow only between the indicated sizes.

viduals are selected, then a single tree is chosen in each. Then a random node is chosen in each sub-tree such that the two nodes have the same return type. If by swapping subtrees at these nodes the two trees will not violate maximum depth constraints, then the trees perform the swap, otherwise, they repeat the search for random nodes. If after a number of tries, it fails to find a suitable match of nodes, no swapping happens, and the node is just reproduced.

- **Mutation:** Used in some of the experiments. It implements a strongly-typed version of the "Point Mutation" operator as described in Koza. One exception is that this implementation maintains the depth restriction – if the tree gets deeper than the maximum tree depth, then the new subtree is rejected and another one is tried. Similar to how the Crossover operator is implemented.

- **Reproduction:** Simply makes a copy of the individuals it receives from its source.

The ramped half-and-half method is used for tree generation, and the a tournament of size 7 is used for selection (as described in Chapter 2).

Figure 4.2 is a systematic diagram of the interaction between the two main system components: the MOGP and the Investment Simulator. There is no automatic way to decide a priori the *relevant* functions and to build a *sufficient* function set.

4.5 Performance Analysis

We are considering the question of how to asses, measure and improve the robustness of a multiobjective GP algorithm applied to a financial real world problem. The dilemma in this real world problem is the fact that the evolved Pareto front solutions evolved will always be used in an environment that is different from the one they were trained in. Measuring the performance of a multiobjective algorithm in training is an already established research area. We are concentrating in our research on measuring the performance in validation. In training the performance of an MO algorithm depends on two things; first the quality of the solutions found, measured by comparing them to solutions on the real front if it is known or to a benchmark if it is unknown. Second is the quality of the Pareto front, which is measured by examining the distribution of the solutions and the coverage area. In validation, we propose the performance of the algorithm to be measured by the two criteria examined in training, in addition to the robustness of the front solutions in the new environment as explained in Chapter 3.

In the following, we present the metrics used in the experiments of Chapter 5 to evaluate and compare the performance of the algorithms.

4.5.1 MOGP Performance in Training

The performance criteria of the MOGP is the ability of the algorithm in training to produce high quality solutions, that is how optimal the objectives are. Since in our experiments, we do not

have the optimal solutions to measure against, we will compare the objective values achieved to the benchmark performance . The objective values considered in the experiments are:

4.5.1.1 Return on Investment

The average annual return (Annualised monthly return) on investment is the first objective. Equation 4.14 is the formula used for calculating the return.

The system traded monthly and every month the return of the portfolio is:

$$R_m = \frac{V_m - V_{m-1}}{V_m}, \tag{4.12}$$

where V_m is the fund value at month m.
If the investment period is n months, then the average monthly return is:

$$Avg\ Monthly\ Return = \frac{\sum_{m=1}^{m=n} R_m}{n} \tag{4.13}$$

and the annualized return is:

$$Return\ on\ Investment = Avg\ Monthly\ Return * 12 \tag{4.14}$$

4.5.1.2 Riskiness of investment

Two definitions of risk were used in the experiments. First, risk as defined by the variability of returns and second, risk as defined by the downside deviation of returns from the average observed. In both cases, the annualised monthly risk was used as in 4.15, 4.16.

$$Risk_1 = \sqrt{\frac{\sum_{m=1}^{m=n} (R_m - Avg\ Monthly\ Return)^2}{n}} * \sqrt{12} \tag{4.15}$$

$$Risk_2 = \sqrt{\frac{\sum_{m=1}^{m=n} (d_m * (MAR - R_m)^2)}{n}} * \sqrt{12} \tag{4.16}$$

where:

MAR is the minimal acceptable return (a certain threshold return), and d_m is an indicator function such that:

$$d_m = \begin{cases} 0 & \text{if } R_m >= MAR \\ 1 & \text{if } R_m < MAR \end{cases}$$

4.5.1.3 Pareto Front Quality

To measure quality, diversity and distribution of the solutions on the Pareto front resulting, we use standard metrics from the literature.

1. The Hypervolume metric:
   Introduced by Zitzler and Thiele [ZT99, ZTL+03] (sometimes known as the S metric), and it uses the hypervolume of the dominated portion of the objective space as a measure for the quality of the Pareto set. It provides a measure of the size of the space dominated by the given front set. The covered hypervolume corresponds to the size of the region of the objective space (bounded by a reference point) that contains solutions weakly dominated by at least one of the members of the set [ZT98].

   If the ideal Pareto front (or a good approximation) is known then a ratio between the evolved Pareto front and the true Pareto front hyper volumes can be calculated and is known as the *Hypervolume Ratio* and is another commonly used form of this metric (where the closer the ratio to 1, the better) [7].

2. The Spacing metric:
   Measures the uniformity of the spread of points on the solution set, and is given by:

$$Spacing = [\frac{1}{n-1}.\sum_{i=1}^{n}(\bar{d} - d_i)^2], \tag{4.17}$$

   where d_i is the distance from solution i to a solution j that is the minimum of the set of distances from solution i to every other solution on the front except itself. $\bar{d}$ is the mean value of all d_i.

3. The Hole-Relative-Size (HRS) metric:
   Measures the size of the biggest hole in the spacing of the points on the trade-off surface, and is given by:

$$HRS = \frac{max_i d_i}{\bar{d}}, \tag{4.18}$$

   where d_i and $\bar{d}$ have the same meaning as in the spacing metric.

4.5.2 Portfolio Performance in Training and Validation

To measure and compare the performance of the financial portfolio constructed using MOGP as the automated decision tool, we have used the Sharpe and Sortino ratios, explained in Section 4.2.5.

[7] Our implementation of this metric is a Java version of the original metric implementation by Eckart Zitzler as in [ZT99] which can be downloaded from http://www.tik.ee.ethz.ch/ sop/pisa/?page=selvar.php (Performance Assessment link)

4.5.3 Robustness of Solutions and the Pareto Front in Validation

To measure the robustness of the solutions, we used the metrics proposed in Chapter 3. For completeness, we include the formulas for the metrics in the following:

1. Robustness of a solution

   Robustness of a solution x_k to a multiobjective problem is defined qualitatively as the degree of its insensitivity to changes in the environment, and quantitatively by the following measures:

   - Whether the solution is still optimal in the context of the new environment.
   - How well it preserved its cluster in the new environment — measured by the cluster distance change metric Δ_k:

   $$\Delta_k = \sum_{j=1}^{m} Dist(Cluster(o_j)^{env1} - (Cluster(o_j)^{env2}), \tag{4.19}$$

   which returns a value between 0 (best), and a max of $(m * (numofclusters - 1))$ (worst).

   - How well it preserved its rank-order in the new environment — measured by the rank change metric δ_k:

   $$\delta_k = \sum_{j=1}^{m} (Rank(o_j)^{env1} - Rank(o_j)^{env2}), \tag{4.20}$$

   The lower the value returned by this metric the better.

2. Robustness of the Pareto front

   Robustness of the Pareto front between two environments is defined by:

   - How well its solutions maintain their objectives' clusters between the two environments — measured by calculating the mean cluster distance μ across all n solutions in the front.

   $$\mu = \sum_{k=1}^{n} (\Delta(x_k)) \tag{4.21}$$

   - How well its solutions' ranks have remained closely correlated between the two environments — measured using a rank correlation test (for example, Spearman Rank Correlation [MBB99]). The Spearman test returns a number in the range $[-1, 1]$ known as the Spearman Coefficient (ρ). The closer the value is to 1, the stronger the correlation between the two rankings. A value of -1 implies negative correlation and a value of 0 implies independence between the two ranks.

   $$\rho(obj_m) = 1 - \frac{6 \sum_{k=1}^{n} \delta_k^2}{n(n^2 - 1)} \tag{4.22}$$

- Quality, diversity and distribution of points on the Pareto front resulting in validation – We use standard metrics from the literature to judge the quality of the Pareto front. The metrics are the Hypervolume, Spacing, Hole-Relative-Size, as presented in Section 4.5.1.

4.5.4 Statistical Analysis

In the experiments conducted, $10-15$ runs of the algorithms are run and results are collected. In addition to the average and the standard deviation of the solutions' performance (as measured by the Sharpe ratio), statistical analysis of the results was conducted:

- When comparing two algorithms, we used the non-parametric ranked t–test [8] [MBB99]. The test was applied to the two distributions of the populations' means, to test the null hypothesis that the two means were drawn from identical population. If this is the case, then the performance of the two algorithms cannot be distinguished. The two-tailed test was chosen, with the alternative hypothesis being that the distribution of the two populations is different. The non-parametric ranked version of the t–test was used due to the fact that the populations cannot be assumed to be normally distributed. The p-values were calculated and compared to the required significance level α.

- In experiments where more than two systems are compared, we used Kruskal–Wallis statistical analysis and the Tukey–Kramer test [MBB99]. When calculating the Kruskal–Wallis test, the greater the difference in locations among the populations distributions, the larger is the value of the H statistic obtained [9]. The Kruskal–Wallis test is the suitable statistical test for comparing k populations based on independent random samples (with $k \geq 2$, and each sample size greater than or equal to 5). The Kruskal–Wallis H-test is calculated as:

$$H = \frac{12}{n(n+1)} \sum_{i=1}^{k} \frac{T_i^2}{n_i} - 3(n+1) \tag{4.23}$$

  where T_i is the rank sum of population i, n_i is the size of population i, $1 \leq i \geq k$, and $n_1 + n_2 + \ldots + n_k = n$

  The Tukey-Kramer test is then used to compare the means of every system to the means of every other system; that is, it applies simultaneously to the set of all pair wise comparisons and identifies where the difference between two means is greater than what the standard error would allow. The difference in means is given by ω, where if the difference between the means of any two systems is greater than ω, then these two samples have different distributions.

[8] The ranked t-test is equivalent to Mann-Whitney test, and they can both be used for statistical analysis where Gaussian distribution cannot be assumed.

[9] With a p-value that follows a chi-square distribution.

4.6 Notes on Design and Implementation of Experiments

4.6.1 Data Normalisation

The complete data set (80 months) was normalised by observing the *max* and *min* values for each factor, and using these values for normalisation as per the following formula for each factor v: $v_{norm} = \frac{v-min}{max-min}$. This technique will usually introduce a look-ahead bias in the data used since the values of *min* or *max* may actually come from the data set saved for future validation. An alternate method would be to normalise the training data set, and then use the derived parameters to normalise the associated validation set, clipping into the required range if needed. The technique used in the experiments was chosen since the values generated by the GP equations are only used for ranking the stocks. Since the look ahead-bias introduced will shift all values in the same direction, the ranking will still be accurate and will not in practice suffer from the suspected look-ahead bias.

4.6.2 Guarding Against Bias

In the following we present some of the sources of bias that could exist when using data for training, and the measures we took to guard against some of them:

- Look-ahead bias: Could occur when the experiment uses data not available at forecast time. To guard against this type of bias, we only used the data that will be available for investors at the time, for example we did not use quarterly earning, as they will not be available each month, but only after each quarter. The normalisation technique we used has the potential to add some look-ahead-bias as well, since it uses the maximum and minimum values over the whole data set. However, as discussed in Section 4.6.1, the ranking technique will eliminate the bias introduced by the normalisation technique.

- Regression-over-fitting bias: Davis Leinweber in [Lei07] provides a discussion of this type of bias. To guard against it in our model, we tested against a model using random strategy and showed that the MOGP performance is in general distinguishable from the performance of the random strategy. In addition, during the experiments, we used an out-of-sample data set that was always completely held back during training.

- Survivorship bias: This type of bias could exist when a system excludes failed companies from the experiments. Since in our experiments, we only included companies that remained in the FTSE100 for the duration of the 80 months, our results possibly suffer from this type of bias. However, our data needed to be consistent, and for companies that merged or split during our 80 months, their data will be incomplete and handling this type of inconsistency would have been more difficult. Hence, our system could be viewed as slightly simplified view of reality, and this choice had to be made based on practical considerations.

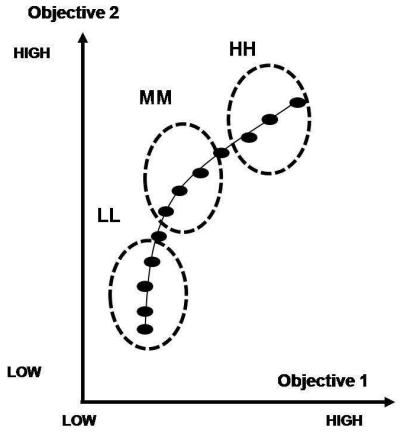


Figure 4.3: Classification of solutions into clusters — a robust system is one where solutions do not change clusters as the environment changes

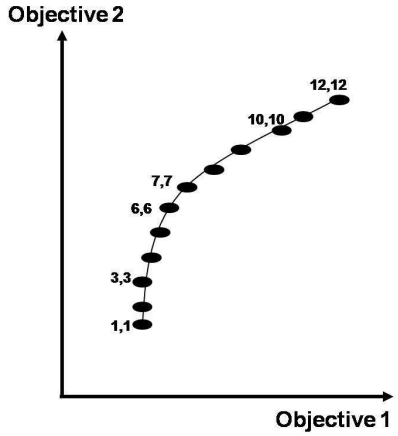


Figure 4.4: Ranking of solutions — a robust system is one where solutions *minimally* change their relative rank with respect to other solutions

4.6.3 Clustering of Solutions

To develop a metric testing for the degree of preservation of objectives clusters, we used the K-means [Mac67] clustering algorithm to classify the objective values into three clusters in the training then in the validation environments, and compared them. In training, the K-means usually ends up with two clusters on both extremes of all objectives, and one with middle values of all objectives For example, for a problem with 2 objectives, every solution is classified as belonging to one of the following clusters: $\langle \mathbf{H}igh, \mathbf{H}igh \rangle$,$\langle \mathbf{M}edium, \mathbf{M}edium \rangle$,$\langle \mathbf{L}ow, \mathbf{L}ow \rangle$. (see Figure 4.3). Where solutions on the first cluster select for portfolios with high-risk/high-return, in the middle cluster are portfolios with medium return-medium-risk and the last cluster has solutions for the more reserved investor looking for low-risk/low-return portfolios.

The clustering is only performed in the last generation of training to save the clustering state of training after the front is found. It is then performed after validation for comparison against training.

4.6.4 Ranking Solutions

We used a ranking algorithm that gave ranks to solutions based on sorting the objective-values of each objective separately. In training environments the ranking algorithms usually resulted in an equivalent ranking for the objective values achieved by each solution: see Figure 4.4. For example, if a solution had a rank of $(6, 6)$ then this means that each of its objective-values was sixth in the rank ordered list of both objective-values. The equal values of objectives ranks was almost always observed in training, but not often observed in validation. We examined the ranking order of the solutions in validation for how closely correlated they were to their rank order achieved in training. The better the rank order correlation, the more robust the solution.

The ranking of solutions' objectives values was done independently for each objective.

Each solution rank is a vector of n values, each corresponding to its rank along a specific dimension. It is implemented through sorting them in ascending order, with ties getting the same rank value equivalent to the average rank over all solutions with the same rank. In the case of the SPEA2, ranking was performed in the last generation only for comparison purposes. In the case of the modified algorithm, it was performed in each generation after the evaluation of the solutions.

4.6.5 GP Diversity

Diversity is important in evolutionary algorithms in general because it helps to prevent immature convergence and minimises over fitting. Both are issues that could have a negative effect on generalisation and robustness. There is a whole area of research aimed at improving the diversity of the GP. Fitness sharing and crowding are the usual techniques used to maintain the diversity in EAs, effectively decreasing the fitness of an individual in a crowded area of the search, and favouring individuals that are in areas of the search space with sparse occupancy. [EN02] used fitness sharing to maintain the diversity of GP programs at certain levels by keeping the population diverse at the beginning to prevent premature convergence and then adaptively changing the neighbourhood size (decreasing it) to control the level of diversity and encourage convergence in later generations. Measuring the similarity between two GP individuals is more difficult than in the case of the simple binary strings of GAs.

Multiobjective researchers were also interested in improving the diversity of solutions on the Pareto front of MO algorithms, albeit for a different purpose, which is ensuring good coverage and distribution of the solutions such that all areas of the optimal front are well represented.

The Role of Mutation:

In this work, the underlying evolutionary algorithm is a Koza style GP in which no mutation is used. Koza considered crossover and reproduction to be the most effective genetic operators to be employed in the GP operation. The crossover works by selecting a random node on each of the tree parents, and the two subtrees, of which the selected nodes are the roots, are swapped. Since the shapes and sizes of the individual trees are irregular in the GP, the result of the crossover even between two identical individuals could be two different offspring. Hence, the reliance on the crossover operator on its own to provide the diversity required for evolution and mutation, traditionally, is not used in the GP. The reproduction operator simply copies the selected individual to the next generation.

However, we build up on previous research [BFN96] outlined in Section 2, where it was shown that the generalisation of GP benefits from an increase in mutation rate. [BFN96] does not actually analyse the mechanism by which increased mutation is able to provide better generalisation. He explains his results by the increase in diversity and hence the GP suffers from less premature convergence and he also shows that higher mutation rates reduce the number of introns. We were interested to investigate whether the generalisation of GP used

in the multiobjective frame work would benefit from an increase in mutation rate in the same way. For this reason we compared the generalisation performance of a standard MOGP using no mutation and an MOGP using point mutation with 0.3 probability.

Duplicate Individuals in Archive:
To further increase the diversity (and test its effect on robustness), we have removed the duplicate individuals from the archive of non-dominated solutions before selection, so that the archive contains only genotypically unique individuals, in effect, minimising the probability of crossover between two identical individuals.

The criteria of adding individuals to the archive is based on their non dominance; moreover, if the number of solutions in the archive is less than its maximum size (usually 1/4 of the population size), dominated individuals are added until the archive is full. Afterwards, recombination is limited to individuals in the archive. Due to the archive size being smaller than the population size, and the –empirically examined– existence of multiple copies of the same individuals, diversity of the resulting child population could be severely effected especially after the algorithm starts to converge.

Hence, in each generation, and after the archive is formed, we eliminated the duplicates from the archive to enhance the diversity of the resulting child population

Chapter 5

Experiments and Results

The unifying theme for the experiments in this chapter is the thesis main research question on the use of multiobjective GP to evolve robust multifactor stock-ranking models in a dynamic and continuously changing environment. This chapter presents three sets of experiments and their results.

The first set of experiments aims to assess the suitability of the multiobjective algorithm for the problem of portfolio optimisation. This is achieved through training the MOGP on historical data and examining: the resulting Pareto front characteristics; stability between independent runs; and quality of the evolved solutions versus two benchmarks.

The second set of experiments examines the robustness of the MOGP on out-of-sample data. These experiments first demonstrate issues unique to the multiobjective nature of the algorithms, and hence that there is a need for definitions and metrics specific to multiobjective algorithms to asses the robustness of the evolved solutions (as described in Chapter 3). Then proceed to make use of the defined metrics to examine the effect of: selection bias; mutation rate; and a cluster-based mating restriction technique, on robustness.

Finally, the third set of experiments deals with optimum tracking in continuous adaptation, and provides preliminary results on techniques for detection of severity of change in the financial domain. We also present preliminary results on the use of the MOGP as an analysis tool for understanding market behaviour.

This Chapter is organized as follows. Section 5.1 presents the benchmarks against which performance will be gauged. Next, Sections 5.2, 5.3, and 5.4 present the experiments, results, and conclusions for the three sets of experiments described above.

5.1 Performance Benchmarks

These are benchmarks against which we compared the performance of the MOGP for portfolio optimisation. The first benchmark was the buy-and-hold strategy. The buy-and- hold strategy (which we hereafter call Index–Fund) is often the benchmark in many such studies, for example [AK99, AS03, CK03, LC09]. The second benchmark is a random search strategy. The random strategy was used for investment in the initial experiments to establish that the MOGP can

actually learn from historical data, by comparing the MOGP performance against performance of randomly generated rules, in both the training and validation phases.

5.1.1 Buy–and–Hold Strategy: Index–Fund

The "Index–Fund" is a portfolio that invests an initial amount of one million pounds with equal proportions in the 82 stocks of the FTSE100, over the time period selected for comparison. This is equivalent to comparing performance to a *buy-and-hold* strategy where we invest the initial cash by buying stocks in the index, hold them for the total of the investment period, and sell at the end. The argument for the buy-and-hold strategy is actually the efficient market hypothesis [MAL03], since if every security is fairly valued at all times, then there is really no incentive for trading [CTM09].

In our simulation of the buy-and-hold strategy, the revenue coming from dividends on the stocks is ignored in calculating the Index–Fund return (as well as later in all experiments for the return of the MOGP-strategy). The ROI performance of the benchmark Index portfolio during the entirety of the time period (80 months) for which the data is available is depicted in Figure 5.1. Note that the plotted ROI is cumulative. In the experiments, the comparison will be done against the buy–and–hold strategy for the specified investment period only, which will be a subset of the 80 months.

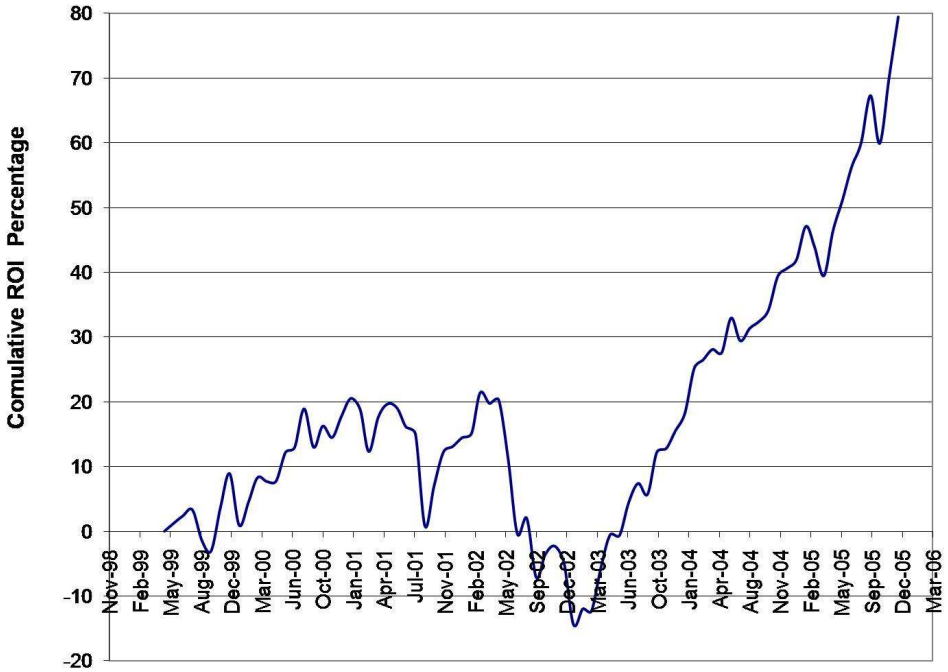


Figure 5.1: Performance of Index Fund Benchmark

5.1.2 Random Strategies: Lottery–Trading

The "Lottery–Trading" benchmark is one of the two benchmarks suggested by Chen et al. [CN06], along with the "Zero–Intelligence" strategy. The authors call the benchmarks *pretests*, and the basic idea is to use the two pretests (both based on random search variants) to give a clear answer on whether a machine–learning–based strategy is actually learning from the training data. The two pretests are: an equivalent–intensity random search (Zero–Intelligence) strategy; and a strategy based on a random decision making process (Lottery–Trading). Comparing the MOGP performance against the performance of any of these two benchmarks on training and test data, should guard against making conclusions of a profitable strategy where the profit was really due to luck (either because of favourable market conditions, or pure luck in rule generation). The comparison should also indicate if there is actually something to be learnt by the algorithm. We present a summary of the two benchmarks suggested by Chen, and our reasoning for selecting one of them as our benchmark.

5.1.2.1 Zero–Intelligence Strategy

Zero–Intelligence Strategy is a random search algorithm with equivalent search intensity to the machine learning algorithm. Equivalent search intensity means that the random search strategy generates and evaluates the same number of different solutions (trading strategies) as those evolved by the machine learning algorithm during their training life time . Hence, if the MOGP has a population of 1000 and evolved for 50 generations, the equivalent-search-intensity random algorithm should create and evaluate 50,000 random solutions.

In our implementation of this benchmark, an equivalent number of solutions to those created throughout the life of the MOGP strategy are created at random. To achieve this, we can possibly create the whole $50,000$ random solutions at once, evaluate them, construct the archive, then terminate (equivalent to having one generation in the evolutionary cycle with a population size of $50,000$) – as the authors suggest. However, to make the Zero–Intelligence strategy run exactly the same way as our MOGP, we implemented this strategy as follows: At generation zero, we create a random population with a size equal to: MOGP popSize + MOGP archiveSize – in this case 1350 –, evaluate the individuals, and construct the archive (of size 350) by selecting the non dominated individuals. For the subsequent generations, the archive is copied to the next population, and another set of random individuals of size = popSize - archiveSize = 1000 solutions – is created , evaluated, and the archive is re-built (from the new 1000 solution plus the solutions already on the archive). This process is repeated for 50 generations, in which $50,000$ random solutions are created and evaluated (actually 50,350). This is equivalent to running the system for 50 generations as in the MOGP, with the exception that no breeding occurs, and for each generation, the population is created at random. [1]

In [CN06], the authors also stress the creation and evaluation of *unique* individuals. The

[1] This way of implementation makes the strategy as close as possible to SPEA2. It is also more practical, since it does not require the memory resources needed if $50,000$ individuals are created at once.

reasoning being that random search may create some identical individuals. However, we believe that this restriction is not necessary, as the evolutionary algorithm is bound to create some identical individuals as well, and the crossover operation also sometimes leads to the creation of already-existing structures.

5.1.2.2 Lottery–Trading Strategy

In the Lottery–Trading strategy, the investment decisions are made on the basis of the outcome of a random variable.

Our implementation of this strategy was through evaluating the performance of portfolios composed by ranking the stocks based on the outcome of a random variable (instead of ranking them based on the GP solution). Afterwards, the monthly buying and selling of stocks are made based on this random ranking. Based on how each solution performs, fitness is calculated and assigned, the SPEA2 archive is constructed and standard breeding is carried out. The number of individuals in the initial population, archive size and the number of generations is the same as the MOGP. The only difference is that the rule ranking (and hence trading decisions) are made at random in every generation, and not according to the ranking assigned by the MOGP individual.

5.1.2.3 Summary: Random Strategies Benchmark

Chen et al. [CN06] suggests using two variants of random search strategies as a pretest of whether a machine–learning–based strategy is learning from the training data, and that its performance is different from that of random search. We have implemented both random strategies, used them as the search algorithm –instead of the MOGP– in the training phase simulation, and applied their results to out-of-sample data. To make the two strategies fully comparable with our MOGP strategy, we uphold compliance with the constraints imposed by our specific investment model, as detailed in Chapter 4.

The main difference between the two random strategies: "Zero-Intelligence" and "Lottery-Trading", is how the buying and selling decisions (in training) are made:

1. Random rules are generated in the Zero–Intelligence strategy, and the stocks are given ranks according to the random rules;

2. In the Lottery–Trading strategy, ranks are given to stocks based on the outcome of a random variable.

We found that results from "Zero–Intelligence" and "Lottery–Trading" have no statistical difference between their performance in any of four different training environments used [2]. In addition, in the experiments reported in [CN06], both strategies have conformed to one hypothesis or the other on the 9 markets in which they were tested. Hence, we have decided to

[2]Note that by allowing for the standard crossover and mutation in the Lottery–Trading strategy we are effectively producing new 'random' solutions during evolution.

compare against just one of them, and chose "Lottery–Trading"" for that purpose. The reason we chose Lottery–Trading is that it is less computationally expensive, as it does not involve evaluation of the GP trees in training, just their random generation and crossover – However in validation, GP trees from both random strategies are evaluated.

In Section 5.2, we report results from the "Lottery-Trading" in comparison to the MOGP results. The comparisons of the MOGP against the Lottery–Trading are based on the average performance over several runs of the MOGP–Strategy, and the average performance over the same number of runs of the Lottery–Trading.

5.1.3 Pareto Front Metrics

To evaluate the quality of the Pareto fronts evolved in the experiments whether in training or validation, standard metrics for Pareto front quality were used. Specifically: the Spread, HRS, and Hypervolume metrics.[3]. In addition, in some experiments, a plot of the Pareto front evolved provides a visual indication of the quality of the Pareto front as well as an insight into some characteristics of the financial market observed (for example: the available variability on return and risk that the current market allows).

5.2 Suitability of MOGP for Portfolio Optimization

In this section we look at the quality of the Pareto front evolved by the MOGP as well the quality of individual solutions on the front. This is largely achieved through training in a variety of environments, and validating the solutions performance on out-of-sample data in comparison to the benchmarks. By inspecting the quality of the Pareto front, we establish the MOGP value as a technique for efficient frontier generation in portfolio optimization. Examining the quality of the solutions evolved when used for investment (out-of-sample data), establishes the utility of the MOGP evolved solutions as investment strategies.

For this experiment we divided the 80 months of historical data into four periods of 20 months each (unless otherwise mentioned). We trained the system on each period separately. All training had an initial population of 1000 individuals, a SPEA2 archive of 350 , and ran for 50 generations.

5.2.1 Performance in Training: Stability

Using the four periods of 20 months each, the system was trained on each period separately for 10 independent runs each with a different random seed. Five of the resulting Pareto fronts for each of the four periods are plotted in Figures 5.2, 5.3, 5.4, 5.5, (The graphs are a plot of risk and return achieved by portfolios composed using individuals on the Pareto front at the end of the evolution cycle. The x-axis represents annualized Risk, and the y-axis the annualized return)

The Pareto fronts drawn give a good indication of the range of possible return and risk that could be achieved in each period. The difference in the risk-return tradeoff range is also a good measure of the how different the four environments are. For example the first environment

[3]Refer to Chapter 4 for more details, and formulas 4.17,4.18, **??** for how the metrics values are calculated

(Env1) shows the largest variety between obtainable risk and return, while on Env2, all the portfolios exhibit high risk with little variation between them in spite of the slightly wider range of corresponding return.

The graphs show (visually) a high stability of the algorithm in terms of the Pareto fronts evolved in the different runs, with little discrepancy between them. To numerically examine the MOGP algorithm stability, we compare the fronts using the hypervolume metric (size of space dominated by the front) and the spread metric (measuring the uniformity of spread of points on the front – the closer to zero the better).

To obtain the hypervolume ratio (the ratio between one front hypervolume and that of the true Pareto front), we need an approximation of the true Pareto front. To obtain an approximation of the true Pareto front, the fronts resulting from training in all of the 10 runs were merged and the non-dominated solutions among them extracted. The resulting fronts are plotted in Figure 5.6. Values given by hypervolume metric for each run is calculated. Table 5.1 displays values of the hypervolume ratio against the true front. The (average, standard deviation) and the hypervolume ratio of the fronts in each environment further strengthen the inference that the Pareto fronts evolved by the MOGP in independent runs are very similar to each other. This result implies: the stability of the SPEA2 (as a multiobjective algorithm) employing a GP as its evolutionary learning scheme; and also that the Pareto front evolved from one run is a good representative of the range of MOGP fronts that can be evolved (in terms of space coverage and objective values) in spite of the stochastic element in the algorithm.

The spread metric values is also calculated and is given in Table 5.2. The metric results show averages close to zero in 3 out of 4 environments and a very small standard deviation from the average in the same 3 environments. We hypothesis that the high volatility exhibited in the 20 months from May 1999 - December 2000 could possibly be causing gaps on the fronts produced and hence worse off results in the spread metric [4].

5.2.2 Performance in Training: Quality

The MOGP is optimising the two objectives of risk and return, and finance practitioners are interested in the tradeoff that exists between the two values. However, performance of different investment funds is usually judged using a risk-adjusted return measure like the Sharpe ratio. Hence, to examine performance of the portfolios formed in training, their Sharpe ratios are calculated. Table 5.3 shows the average Sharpe ratio (average of Sharpe ratio over front solutions, averaged over the 10 runs), and the average standard deviation (standard deviation of Sharpe Ratio of solutions on the front, averaged over the 10 runs) in each of the four environments. The tables compares these values against the best and average Sharpe ratio achieved by the Index–Fund and the Lottery–Trading benchmarks. Results show the MOGP outperforming the Index–Fund – even when allowing for the variability of the standard deviation – on all four

[4] It would be interesting to test this hypothesis by running an experiment with varying volatility and examining the effect on the spread of the resulting fronts.

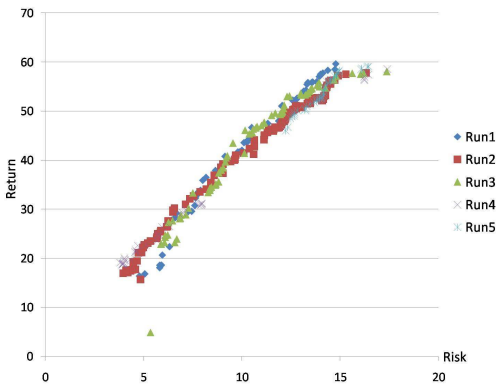


Figure 5.2: Pareto fronts for training on Months May1999-December2000

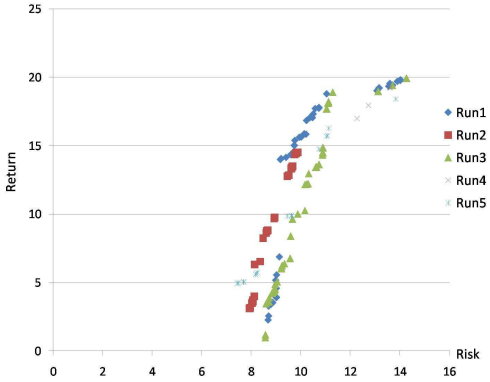


Figure 5.3: Pareto fronts for training on Months January2001-August2002

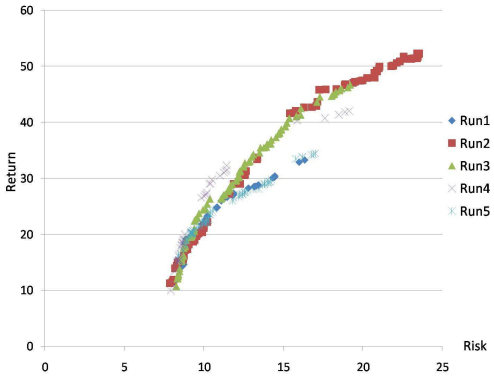


Figure 5.4: Pareto fronts for training on Months September2002-April2004

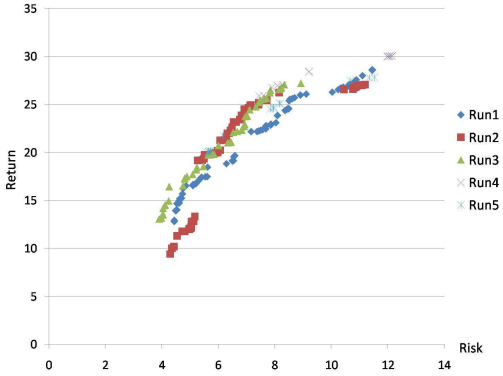


Figure 5.5: Pareto fronts for training on Months May2004-December2005

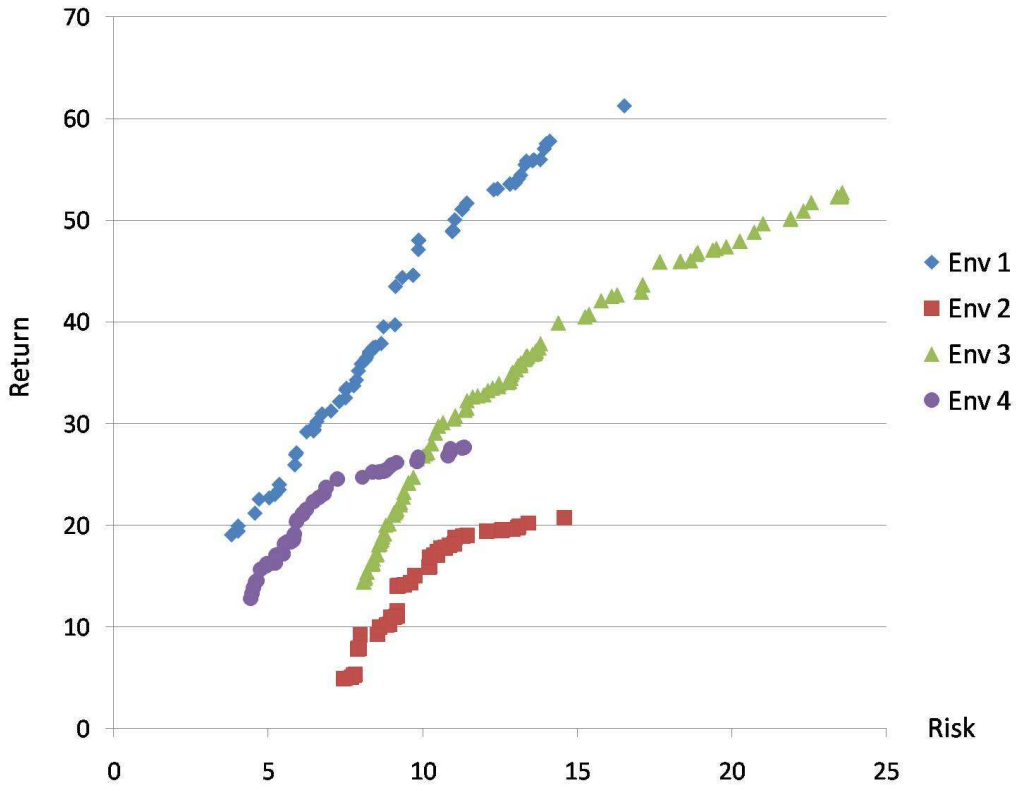


Figure 5.6: Approximation of the True Pareto Fronts in the Four Environments

environments [5]. The Lottery–Trading strategy in contrast is achieving very low average Sharpe values in comparison to both the MOGP and the Index–Fund, although the low standard deviation between its 10 runs indicate that the size of population chosen is large enough to cover a reasonable sample of the search space such that the independent runs achieve similar results.

5.3 MOGP Robustness: Performance on Unseen Data

In this set of experiments, we look at the performance of the evolved solutions on the Pareto front when used for investment in a subsequent unseen environment. We are interested in answering the following questions:

- Are the trained MOGP solutions suitable for investment in an *unseen* subsequent environment (how well do they perform in portfolio stock-picking)?
- What are the suitable performance metrics to measure success or failure of solutions in out-of-sample environments (taking into account that a drop in objective values is not a sufficient criterion since the new market conditions may not allow for a higher value)?
- What affects the evolved solutions' performance in unseen environments, and can this performance be improved?

[5]The real test for the MOGP will have to be on an out-of-sample data set which is presented in the next section

Table 5.1: Size of the space dominated by the fronts and the Hypervolume ratio of each (in comparison to the hypothetical true front)

	Env1	Env2	Env3	Env4
TRUE	**591.93**	**79.79**	**442.33**	**252.46**
Run1	0.861	0.944	0.918	0.845
Run2	0.806	0.831	0.918	0.973
Run3	0.709	0.848	850	0.834
Run4	0.885	0.970	0.721	0.834
Run5	0.744	0.965	0.729	0.86
Run6	0.841	0.839	0.98	0.866
Run7	0.811	0.996	0.99	0.813
Run8	0.686	0.994	0.91	0.898
Run9	0.987	0.996	0.877	0.815
Run10	0.895	0.997	0.811	0.825
Avg	0.830	0.938	0.87	0.856
StdDev	0.096	0.066	0.088	0.046

Table 5.2: Spread characteristics of the fronts in the four environments

	Env1	Env2	Env3	Env4
Run1	0.345	0.199	0.158	0.096
Run2	0.27	0.108	0.221	0.116
Run3	1.98	0.246	0.265	0.1
Run4	0.2	0.165	0.322	0.1
Run5	0.288	0.088	0.126	0.136
Run6	0.317	0.09	0.205	0.147
Run7	0.253	0.277	0.13	0.101
Run8	0.469	0.093	0.187	0.123
Run9	0.461	0.155	0.288	0.172
Run10	0.186	0.191	0.202	0.065
Avg	0.476	0.144	0.21	0.115
StdDev	0.536	0.079	0.065	0.030

If the stocks picking models evolved, using an MOGP, turn out to be not robust enough – when used in environments different from that on which it was trained – this can be due to one of two reasons: either the MOGP over fits its evolved solutions to the training environment and improved generalization is needed; or the risk-exposure factors themselves are time varying and a model that works in one time period is not guaranteed to work on another [6]. The first possibility is investigated in experiments of Sections 5.3.3 and 5.3.4, where we aim to embed techniques hypothesized to improve generalisation, and measure the effect they have on the performance and robustness of the MOGP in our financial domain. If the risk factors are time varying, then what is needed is: 1)A technique for proper detection of the failure of the current ranking model, and 2) Retraining the MOGP to discover the new risk factors, and accordingly re-adjust investment decisions . We explore on the second possibility in (Section 5.4).

[6] A third possibility is that the market is efficient and there is nothing to learn.

Table 5.3: MOGP Performance in Training against Index–Fund (IF) and Lottery–Trading (LT) over the four environments

	Env1	Env2	Env3	Env4
IF Sharpe	0.478	-0.566	0.8496	1.334
MOGP Avg Sharpe	3.0481	0.486	1.593	2.218
MOGP Avg Std dev	0.519	0.4568	0.222	0.1993
LT Avg Sharpe	-0.138	-1.363	0.258	0.1999
LT Avg Std Dev	0.422	0.268	0.2715	0.2445

Table 5.4: Index–Fund Performance on the 4 Environments

	Env1	Env2	Env3	Env4
Index–Fund Sharpe	0.478	-0.566	0.8496	1.334

This section is organized as follows: In Section 5.3.1, we examine the quality of the MOGP solutions evolved in the experiment described in Section 5.2.2 when used for investment in 3 out-of-sample environments. We then focus on issues particular to the algorithm being a multiobjective one in Section 5.3.2. Next, we study three techniques for improving the performance of solutions in out-of-sample- environments in Sections 5.3.3 and 5.3.4.

5.3.1 Investment Performance against Benchmarks

The aim of this experiment is to examine if the MOGP solutions evolved in training carry some predictive power that can be effectively used by practitioners for investment.

In this experiment, the dataset was divided into 4 environments of 20 months each out of the available 80 months. The MOGP system was trained on each of the four environments (Env1, Env2, Env3, Env4). The solutions evolved from each period were validated on the three other environments, and the average and best Sharpe ratio achieved by Pareto front solutions recorded.

Likewise, the Lottery–Trading strategy was also trained on the four environments separately. The solutions from each environment were then validated on the three other environments. In contrast, there was no training for the Index–Fund, and it produces one Sharpe ratio value instead of a Pareto front.

Table 5.4 shows the Sharpe ratio of the Index–Fund on the four periods. Tables 5.5, 5.6, 5.7 and 5.8 show the average and best Sharpe ratio of the validation results of the MOGP and Lottary–Trading. Both MOGP and Lottery–Trading results are the averages of 10 separate runs. Hypothesis testing is performed with the non parametric $T - test$ to examine the statistical significance of (MOGP, Lottery–Trading (LT)), and (MOGP, Index–Fund) with the null hypothesis that the MOGP performance cannot be differentiated from the Index–Fund or randomly generated rules (Lottery–Trading).

Table 5.5: Validation Performance of Training on Env1

	Env2	Env3	Env4
MOGP Avg Sharpe	0.099	0.9489	1.2431
LT Avg Sharpe	-0.74262	0.5168	1.0882
T-test Avg MOGP and Avg LT	7.5E-07	6.89E-07	2.08E-06
T-test Avg MOGP and Index–Fund	4.55E-09	0.000184689	0.010928521
MOGP Best Sharpe	0.567	1.25	1.654
LT Best Sharpe	0.357	1.29	1.789
T-test Best MOGP and Best LT	0.00157	0.5109	0.011
T-test Best MOGP and Index–Fund	4.55E-09	4.55E-09	4.55E-09

Table 5.6: Validation Performance of Training on Env2

	Env1	Env3	Env4
MOGP Avg Sharpe	1.195	0.78307	1.2223
LT Avg Sharpe	-0.042	0.4955	1.03123
T-test MOGP and LT	7.50314E-07	3.505E-05	1.20093E-05
T-test MOGP and Index–Fund	4.55345E-09	0.010928521	0.010928521
MOGP Best Sharpe	2.167	1.275	1.58
LT Best Sharpe	1.77	1.22	1.68
T-test Best MOGP and Best LT	0.029	0.771	0.046
T-test Best MOGP and Index–Fund	4.34E-09	4.55E-09	4.55E-09

Table 5.7: Validation Performance of Training on Env3

	Env1	Env2	Env4
MOGP Avg Sharpe	0.931	-0.063	1.522
LT Avg Sharpe	0.075	-.0977	0.907
T-test Avg MOGP and Avg LT	7.50314E-07	1.20093E-05	1.20093E-05
T-test Avg MOGP and Index–Fund	0.000184689	0.000184689	0.000184689
MOGP Best Sharpe	2.131	0.548	1.981
LT Best Sharpe	1.461	-0.189	1.625
T-test Best MOGP and Best LT	0.02380749	0.02380749	0.000440648
T-test Best MOGP and Index–Fund	4.55345E-09	4.55345E-09	4.55345E-09

Table 5.8: Validation Performance of Training on Env4

	Env1	Env2	Env3
MOGP Avg Sharpe	0.612	-0.498	0.767
LT Avg Sharpe	0.095	-0.654	0.577
T-test Avg MOGP and Avg LT	0.000693442	0.004540706	0.029972286
T-test Avg MOGP and Index–Fund	0.434142518	0.010928521	4.55345E-09
MOGP Best Sharpe	1.542	0.212	1.230
LT Best Sharpe	1.185	0.560	1.399
T-test Best MOGP and Best LT	0.018692511	0.003257445	0.056166935
T-test Best MOGP and Index–Fund	4.55E-09	4.55345E-09	0.107723971

Looking at Tables 5.5, 5.6, 5.7, 5.8 for the average Sharpe ratio, we notice that the MOGP performs better than the Lottery–Trading strategy in all environments and regardless of the environment it was trained on, and the difference is highly significant (except in Table 5.8 for Env4 training and Env2, Env3 validation). The less powerful results achieved by training on Env4 (a very strong consistent bull market) probably indicates Env4 being remarkably different from Env2, Env3. This result highlights the importance of the choice of training data, and that possibly different rules are at play in varying environments. In general, however, the MOGP is making use of knowledge in the training data, and the equations evolved, on average, carry some predictive power to most unseen data. Looking at the best Sharpe ratio found by the MOGP and LT, we find that the results they cannot be statistically distinguished in most cases, in spite of the fact that the average performance of the Sharpe ratio is clearly better. This result means that the search space explored by the random strategy is large enough to stumble by chance on one single rule with very good performance. The average performance however shows that the MOGP in contrast with the LT is discovering a whole set of rules that generalise favourably to out-of-sample environments.

Comparing the MOGP performance to the Index–Fund, on the other hand, shows that the equations evolved perform better in bear (Env2) and volatile markets (Env1). On strong bull markets like (Env4), the performance is comparable to the buy-and-hold strategy (actually Index–Fund outperforms MOGP albeit with a small difference (MOGP 1.2, Index–Fund 1.3) which is not statistically significant). The difference in performance is most likely due to increased transactions cost through multiple buys and sells by the MOGP strategy. This result is similar to what many researchers have found in a variety of markets. For example, Potivin et. al in [PSV04, LC09] found that GP evolved rules were good in bear and volatile markets but not in bull markets. Like the authors of [LC09], we believe that this is a natural result, and that in a bull market with high investors confidence it is intuitive to find a buy-and-hold strategy outperforming most other startegies.

Hence, from the fund performance point of view, the MOGP has proven to be able (on average) to outperform the buy-and-hold strategy in markets which investors consider as tricky (bear and volatile) on the FTSE markets considered here. However, from the algorithmic point of view we want to also investigate what happens to the Pareto front characteristics when its individuals are applied to the out-of-sample data. Also, as explained in Chapter 3, it is of vital importance from the fund manager point of view to closely inspect the performance of individual factor models on the unseen data. Since, although the Pareto front on average outperforms the buy-and-hold strategy, that in itself, does not guarantee that any particular factor model assumed to yield a risk-return in a certain cluster will actually still deliver that specific performance on out of sample data. As previously illustrated in Chapter 3, in the next section, we present the performance of the Pareto front and individual solutions on the 20 months of out-of-sample data.

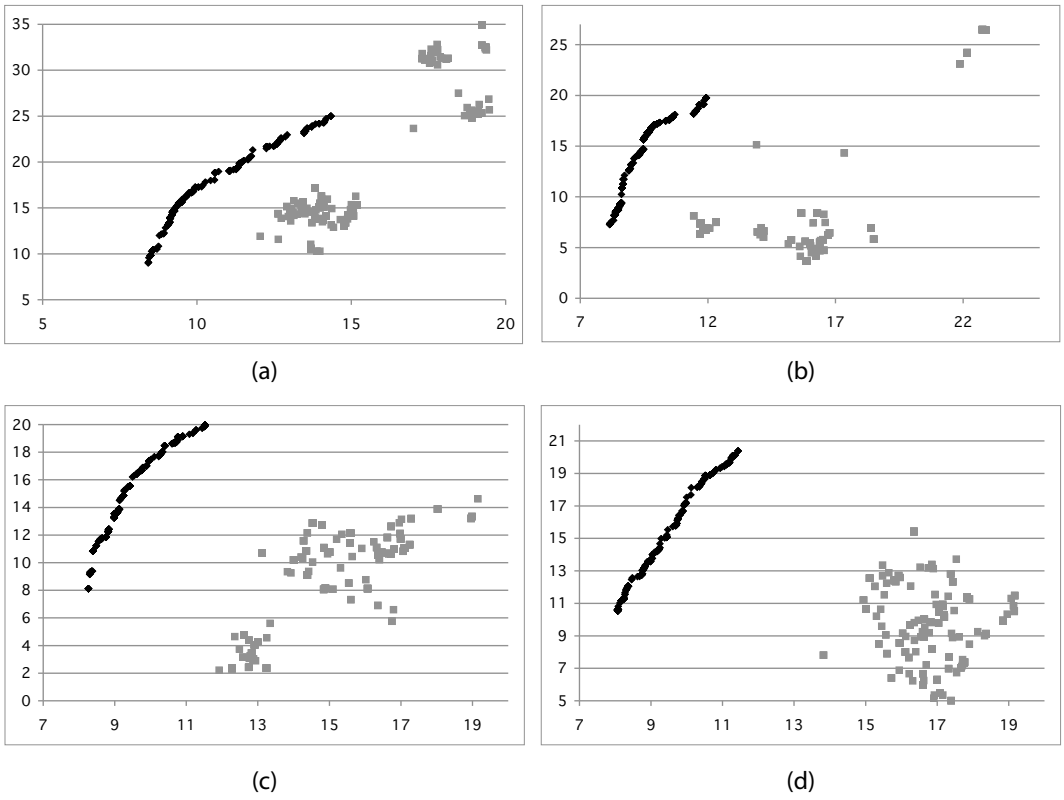


Figure 5.7: SPEA2 Pareto Fronts in Training (black) and Validation (grey) over four runs – x-axis is Risk and y-axis is Return

5.3.2 Pareto Front Performance

To test the Pareto front performance on out-of-sample data, results of 15 runs of the system performance on test data algorithms are reported in this section. The system was first trained on the first 60 months of the data set. Then for each run, the solutions on the Pareto front were applied to the test data of the following 20 months (equivalent to using the investment strategy represented by the solutions to manage a new financial portfolio). The performance of the evolved solutions on the test data varies between the runs. Figure 5.7 presents four runs with the Pareto front in training and in validation.

It is noticed in these graphs not only that the performance is worse than in training, but also that the Pareto front as a whole loses its distribution characteristics. Another more serious problem is illustrated in Figure 5.8. The figure shows a solution P1 that in training displayed relatively high returns at relatively high risk — but in validation it had relatively the worst return with low-to-medium relative risk. Another solution P2 that was relatively medium-return/medium-risk in training became relatively low-return with relatively medium-to-high-risk in validation, and also became dominated by other solutions. The solution P3 changed from relatively medium-return in training to relatively low-return in validation and clearly became dominated by several other solutions that achieved the same risk with higher return.

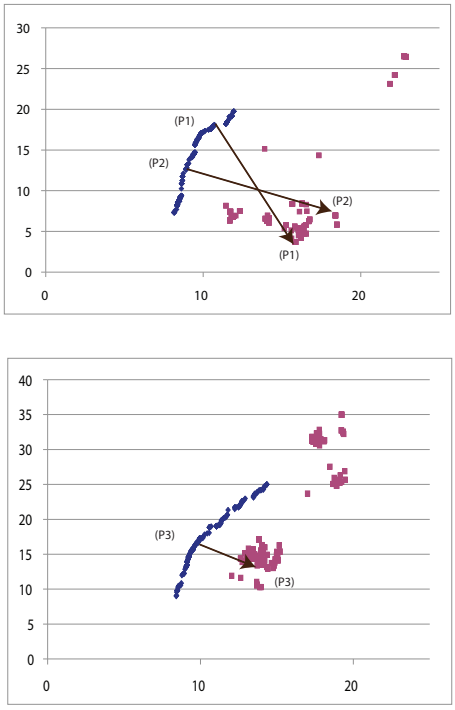


Figure 5.8: Example of Solutions Changing their Objectives Profile(Cluster). The vertical axis measures return (percentage return on investment), and the horizontal axis measures risk (standard deviation of monthly returns).

As noted before, this behaviour is of particular importance in our application. A fund manager employing an investment strategy clearly would require (at least) that the strategy maintains its objectives' characteristics relative to the other available strategies. Although the new set of solutions may still be on the efficient frontier, from the point of view of the fund manager they would still be wrong for her purpose.

In the next two sections, we present two experiments in which we further illustrate the problem and quantify robustness using metrics devised in Chapter 3. We also examine the effect of three different techniques that successfully improve this particular aspect of robustness.

5.3.3 Selection Bias Effect on Robustness of MOEA

We implement a robust version of SPEA2 — R-SPEA2 — where solutions' robustness is measured during evolution and used to bias the selection pressure. This is done through the use of a validation data set. For every generation, the non-dominated solutions' fitness and ranks are measured. Then non-dominated solutions are then validated on a separate data set. Based on how well the solutions preserve their ranks on the new data set, their fitness is incremented or decremented. The tournament selection is performed as usual. It will now prefer solutions which are: non-dominated, in less dense areas of the front, and which are more robust across the diverse training environments. The fitness assignment in the R-SPEA2 algorithm is as follows:

Algorithm 2 Fitness Assignment in R-SPEA2

```
for all gen such that 0 ≤ gen ≤ genMax do
    identify the non-dominated solutions
    get ranks of all solutions
    for all sol such that 0 ≤ sol ≤ popSize do
        apply sol in a different environment
        rank sol in the new environment
        Calculate robustness value R
        add R to sol fitness value
    end for
    Select Parents ...
end for
```

5.3.3.1 Data Sets and Experiments Specification

We simulate a long-only sector-neutral portfolio of 25 stocks. The balanced investment across several industries guards against the price shocks of any one sector. The stocks are selected from the UK stock market as represented by the FTSE100. For every stock, data of 22 financial factors[7] over 80 months is available.

All the factor values are normalized before using them within the investment simulator and the GP, in order to minimize the effect of a number of parameters with high ranges dominating the model. Also, normalization of the parameters should have a positive effect on robustness, because all perturbations in parameters in different environments are put into similar perspective with changes in other parameters and with changes from the values dealt with in the training environment.

The total period of 80 months is divided into training and testing for standard SPEA2, with 60 months for training and 20 for testing. For R-SPEA, the same data set was divided up into training, validation and testing. For training (in-sample), 48 months from May 1999 to April 2003 are used. The next 12 months (May 2003 - April 2004) were used for validation. For testing (out-of-sample), the data is that of the last 20 months from May 2004 to December 2005. This way both algorithms see the exact same data albeit in two different ways.

All experiments had a population size of 500, archive size 200, and ran for 35 generations. The method of tree generation is ramped half and half [Koz92]. The terminal set for the tree consists of technical and fundamental financial factors describing a company's performance, plus constants. The function set includes addition, subtraction, multiplication, division, power 2, and power 3.

The return on investment (ROI) of the Index Fund portfolio (invests one million pounds,

[7] For details of the factors used please refer to Chapter 4.

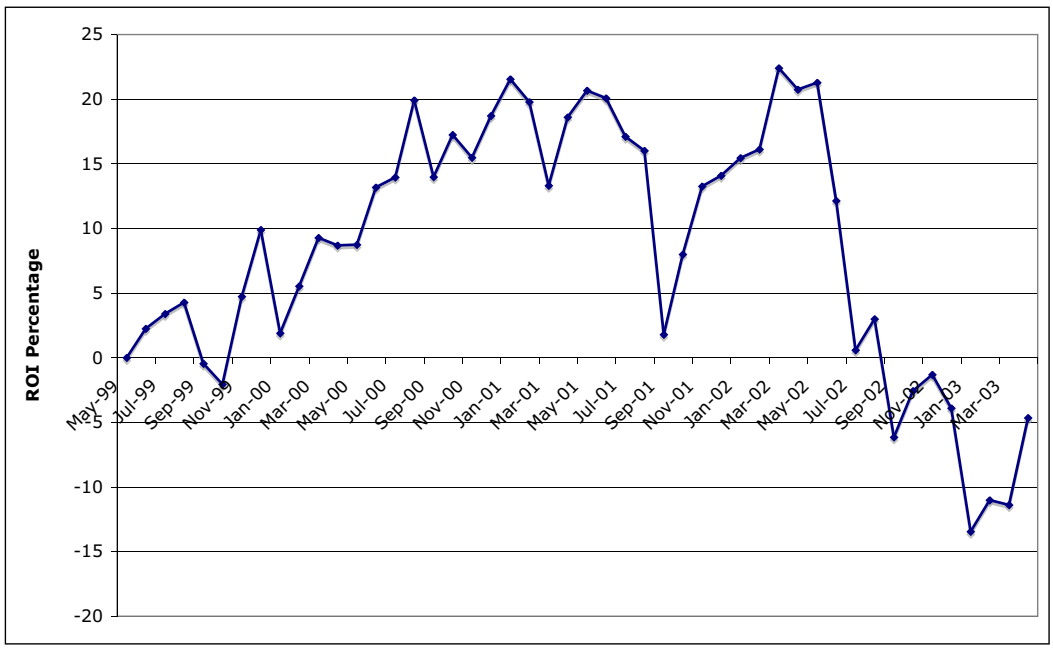


Figure 5.9: Performance of Index Fund During Training Period

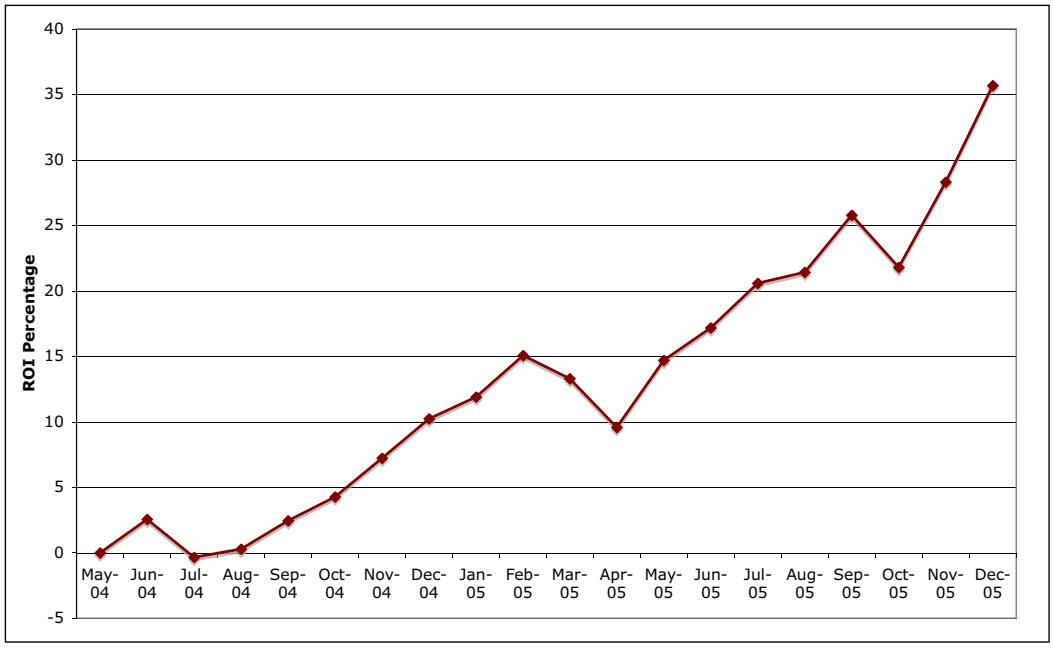


Figure 5.10: Performance of Index Fund During Validation Period

with equal proportions in the 82 stocks of the universe, over the two time periods selected for training and testing) is depicted respectively in Figures 5.9 and 5.10.

5.3.3.2 Clustering and Ranking Algorithms

In our experiments, clustering of solutions on the front is implemented using the k-means algorithm to group solutions into 3 clusters (high return/high risk), (medium return, medium risk), (low return, low risk). The clustering analysis is performed twice in both SPEA2 and R-SPEA2: at the end of the last generation of training; and after using the solutions for investment on the validation data set.

Ranks of solutions (by objectives' values) is calculated independently for each objective, resulting in each solution rank being a vector of n values, each corresponding to the solution's rank along an objective dimension. The rank algorithm is implemented by sorting the solutions, with respect to each objective value, in ascending order, with ties assigned the average rank of all values with the same value. In the case of SPEA2, ranking is performed in the last generation only for comparison purposes. In the case of the modified algorithm it is performed in each generation, after the evaluation of the solutions.

5.3.3.3 Results

In this section, we present the results of comparing the performance of SPEA2 and R-SPEA2. Visually inspecting the Pareto fronts produced by R-SPEA2. Figure 5.11 shows the Pareto front in four runs of the modified algorithm in training and out-of-sample testing phases. For the out-of-sample performance, the figure shows slightly better spread characteristics than achieved in the original algorithm (see Figure 5.7), with fewer solutions losing their non-dominance.

Table 5.9 shows (for both SPEA2 and R-SPEA2) the mean distance of cluster change between training and testing of all solutions on the front as well as the percentage of solutions that maintained their cluster. R-SPEA2 has on average more than half of the solutions on the Pareto front keeping within the objectives' profile achieved in training. A non-parametric ranked T–test applied to the two distributions of means gives a p-value of 1.2×10^{-6}, indicating that this difference between SPEA2 and R-SPEA2 is statistically highly significant (the p-value for the percentages is 0.07364).

By contrast, Table 5.10 indicates the much tougher test of Spearman rank correlation of all individuals: Results show that R-SPEA2 achieved an average improvement of only 10% of the coefficient value. A Ranked T–test comparison indicates that these differences for objective-1 and objective-2 are not statistically significant (p-values are 0.85572 for Objective-1 and 0.14373 for Objective-2 (significant only at the 15% level for Objective-2)).

5.3.4 Diversity and Cluster-Based Mating Restriction for MOEA Improved Robustness in a Financial Dynamic Environment

In this experiment we use two techniques — Mating Restriction and Diversity Preservation, and examine their effect on robustness. Following observations of phenotypic clustering in a

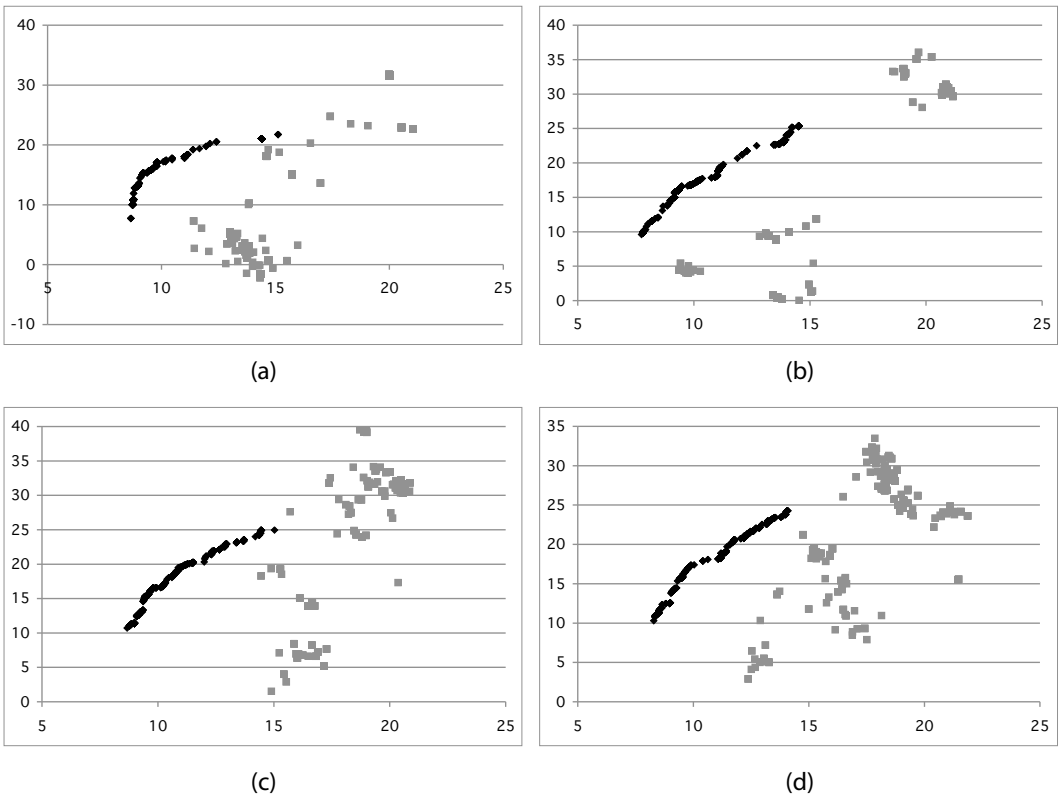


Figure 5.11: R-SPEA2 Performance in Training (black) and Validation (grey) over four runs. The vertical axis measures return (percentage return on investment), and the horizontal axis measures risk (standard deviation of monthly returns)

Table 5.9: Mean distance of cluster change and percentage number of solutions changing cluster for SPEA2 and R-SPEA2

	SPEA2		**R-SPEA2**	
Run	**Mean**	**Percentage**	**Mean**	**Percentage**
1	0.656	57.75%	0.657	100.00%
2	1.615	63.00%	0.340	20.25%
3	1.100	52.00%	0.402	68.50%
4	1.365	62.25%	0.440	47.25%
5	1.120	89.00%	0.720	47.75%
6	1.742	100.00%	0.525	54.00%
7	1.420	61.25%	0.425	28.00%
8	0.965	42.25%	0.295	46.00%
9	0.747	73.00%	0.590	61.25%
10	0.890	44.50%	0.397	70.75%
11	0.365	18.25%	0.590	56.00%
12	1.460	69.00%	0.530	42.00%
13	0.675	33.75%	0.245	35.00%
14	1.575	78.75%	0.085	21.00%
15	1.707	100.00%	0.430	32.25%
AVG	1.160	62.98%	0.444	48.66%
StdDev	0.432	23.27	0.166	21.15

Table 5.10: Correlating between Training and Validation: Spearman Coefficients of Objectives

	SPEA2		R-SPEA2	
Run	**Obj1**	**Obj2**	**Obj1**	**Obj2**
1	0.6491	0.6763	0.7664	0.0922
2	0.0228	0.1440	0.9484	0.7047
3	0.4590	0.3856	0.7482	0.7594
4	0.6331	0.4315	0.6372	0.6868
5	-0.2085	0.7051	0.5859	0.5655
6	0.0361	0.2633	0.3686	0.6912
7	0.5495	0.5905	0.2326	0.3963
8	0.6903	0.3665	0.5740	0.8032
9	0.9494	0.3184	0.5533	0.2687
10	0.7765	0.7307	0.7255	0.4083
11	0.6379	0.8857	0.4605	0.5543
12	0.6119	0.7510	0.8242	0.8078
13	0.7345	0.6060	0.8546	0.8514
14	0.7930	0.0710	0.3325	0.6808
15	0.6957	-0.3042	0.5706	0.7235
AVG	0.5354	0.4414	0.6122	0.5996
StdDev	0.3268	0.3149	0.2043	0.2189

stock-picking MOGP (see Section 5.4.2), we hypothesize that each cluster is specializing for a particular niche in the phenotypic space, and therefore restriction of mating to others within the same phenotypic cluster will possibly have a positive effect on the evolution of more robust individuals. We also know from prior work that diversity preservation in GP favours smaller trees and therefore avoids over-fitting [BFN96], which we hypothesize will also lead to more robust solutions.

Both techniques are known to provide benefits to Multiobjective Evolutionary Algorithms (see Section 4.6). However, all prior work appears to be restricted to training (e.g. to improve the distribution of solutions on the Pareto front) and we have found *no* prior work which demonstrates a beneficial effect on the robustness of solutions (nor of the Pareto front) in unseen environments.

In this section, we examined three routes to improving the robustness of the MOGP algorithm in unseen environments: enhancing population diversity; a new variant of mating restriction; and the combination of both. The effect of their use on robustness is measured.

5.3.4.1 Diversity Enhancement

Diversity is important in evolutionary algorithms in general because it helps to prevent immature convergence and minimizes over-fitting. In this work, the underlying evolutionary algorithm is a Koza style GP in which no mutation is used. However, previous research in [BFN96] showed that the generalisation of GP benefits from increasing the mutation rate. Although no analysis of the mechanism by which increased mutation is able to provide better generalisation, the results were explained by the increase in diversity and hence the GP suffers

from less premature convergence. The author also shows that higher mutation rates reduces the number of introns. We are interested to investigate if the generalisation of GP used in the multiobjective frame work will benefit from an increase in mutation rate (and hence diversity) in the same way.

5.3.4.2 Mating Restriction

The current research on mating restriction in evolutionary multiobjective algorithms can be divided into two main classes: mating of similar parents or mating of dissimilar parents. The former will speed up convergence and in some problems the quality of the solutions. On the other hand, mating of dissimilar parents will improve diversity, which is vitally important in the EMO search. However, *no* research was carried out to investigate the effect of encouraging mating of either similar or dissimilar parents on the performance of the EMO on out-of-sample data.

Previous work [Has08] has shown that solutions evolved for each objectives-cluster possible have common characteristics that distinguish them from solutions in other clusters. That has led us to believe that in this financial domain, the MOGP is discovering rules belonging to various niches (corresponding to the clusters). If this were actually the case, then by limiting the mating to parents belonging to the same cluster and hence sharing the same objectives characteristics we will further help this speciation. We are interested to investigate the effect this special kind of similarity mating will have on one particular aspect of generalization which is the movement from one cluster to the others between training and validation environments.

To test the hypothesis, we simulated a mating restriction technique, whereby mating – between solution in the non-dominated archive – is restricted to parents belonging to the same (objectives) cluster. Parents are selected using binary tournament selection of size 7 with replacement, exactly as in standard SPEA2. The difference is, the second parent is accepted only if it belongs to the same cluster as the first parent. If not, we attempt to reselect the second parent for a maximum of four more times. If we fail to select a parent belonging to the same cluster after five trials, the first parent crosses over with a copy of itself. Thus, a parent never mates with another parent from outside its cluster.

5.3.4.3 Data Sets and Experiment Specifications

We simulate a long-only sector-neutral portfolio of 25 stocks. The balanced investment across several industries guards against the price shocks of any one sector. The stocks are selected from the UK stock market as represented by the FTSE100. For every stock, data of 22 financial factors[8] over 80 months is available. All the factor values are normalized before using them within the investment simulator. The total period is divided into training and testing. For training (in-sample), 48 months from May 1999 to April 2003 are used. For testing (out-of-sample), the data is that of the last 20 months from May 2004 to December 2005. For more details on the investment strategy refer to Chapter 4.

[8]For details of the factors used please refer to Chapter 4. Data supplied by Reuters©.

Algorithm 3 Cluster-based Mating Restriction Algorithm

```
Build archive
Adjust archive so it contains only unique individuals and copy to next population
Number of solutions to breed = popSize − archiveSize
repeat
  Select parent1 and parent2 from archive
  if parent1Cluster! = parent2Cluster then
    sameCluster ← false
    trial ← 1
    while !sameCluster and trial <= 4 do
      Select parent2
      if parent1Cluster == parent2Cluster then
        sameCluster ← true
      else
        trial + +
      end if
    end while
  end if
  if parent1Cluster! = parent2Cluster then
    parent2 ← parent1
    Crossover parent1 and parent2
  end if
until New population is built
```

Four sets of simulations were conducted (see Section 5.3.4.4). Results are reported for 15 runs of each system, which are sufficient for the statistical tests that we use to compare all systems against each other – the Kruskal-Wallis H-test and the Tukey-Kumar test [MBB99]. Statistical results are based on observation of only the unique individuals in the archive to prevent multiples of either good or bad solutions biasing the results. Crossover probability is 0.7 throughout.

All experiments had a population size of 500, archive size 200, and ran for 35 generations, after which no significant improvement (in training) was observed regardless of any additional computation. The method of tree generation is ramped half and half [Koz92]. The terminal set for the tree consists of technical and fundamental financial factors describing a company's performance, plus constants. The function set includes addition, subtraction, multiplication, division, power 2, and power 3.

5.3.4.4 Experiments

Four simulations were run as follows:

1. **Standard SPEA2:** The standard SPEA2 algorithm is used in the simulations. Reproduction probability 0.3, no mutation is used.

2. **Diversity Enhancement** Standard SPEA2 with enhanced diversity is used in this set. To increase the diversity, we use the high mutation probability of 0.3, and no reproduction throughout the training. Also, in each generation, after the archive is built, duplicate (genotypically equivalent) individuals are deleted. Selection, breeding and statistics use this modified archive. This way, the probability of crossover between two identical individuals is eliminated, with the aim to increase the probability of crossover producing children that are different from their parents and hence increase the diversity in any one population and at the same time increase the chances of wider exploration of the search space.

3. **Mating Restriction** The underlying algorithm is SPEA2. However, mating restriction as described in Section 3.5.3.1 is employed. For comparison with the first set of simulations, the reproduction probability is 0.3, and no mutation is employed.

4. **Mating Restriction and Diversity Enhancement** Same as the previous set of simulations with the exception that the operators used are crossover with 0.7 probability and mutation with 0.3 probability. The duplicates in the archive are also deleted, leaving only genotypically unique individuals. Selection and breeding as well as statistics are done on the modified archive.

5.3.4.5 Results

All results reported are regarding the performance of evolved Pareto front solutions on the out-of-sample period.

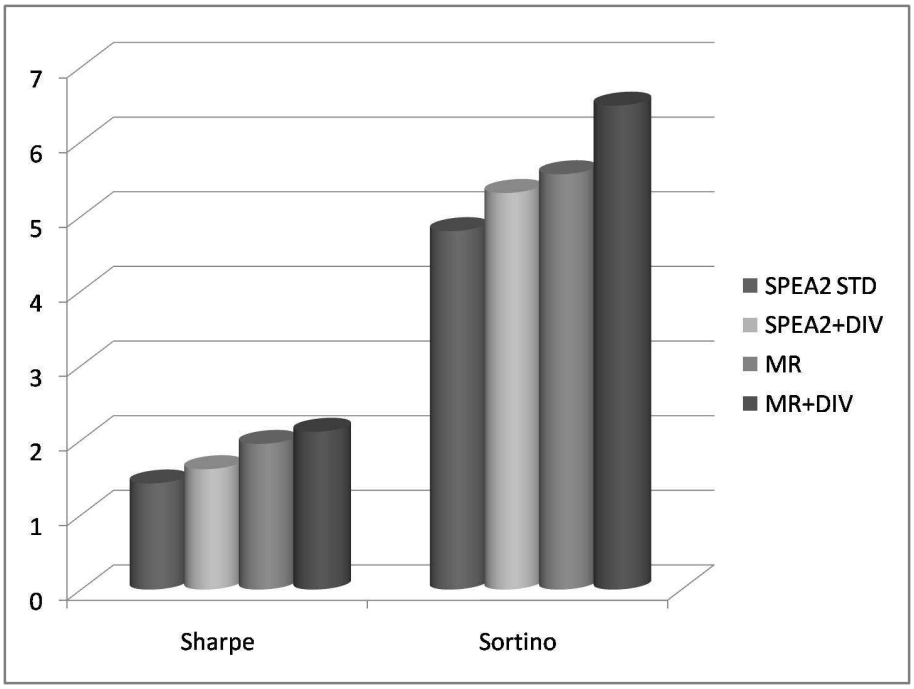


Figure 5.12: Sharpe and Sortino Ratios

1. **Quality of solution** - The average quality of solutions was compared for each system (averaged across all solutions in the Front, and across all runs). The Sharpe ratios achieved by using factor models evolved by each of the four techniques for investment during validation are reported. Figure 5.12 shows results of the four systems (both Sharpe and Sortino Ratios). Results show that using diversity-enhanced-mating-restriction gives the best result (Sharpe=2.11), mating restriction comes second (1.95), diversity preservation is third (1.6) and standard SPEA2 has the worst performance (1.42). By comparison, the index performance on the same period (i.e. the performance of an index tracker fund) had a Sharpe ratio of 1.364 — this was measured by simulating a long-only investment of £1,000,000 in equal proportion in all 82 stocks making up the index for the duration of the validation period — and the best possible Sharpe ratio achieved was 3.15 (achieved by post-hoc exhaustive training of all systems on the out-of-sample period).

2. **Preservation of solutions order** – Do solutions retain their relative order on the Front when moving from training to an unseen environment?

   This is the criteria that if achieved, will indicate that the solutions performance in the new environment is keeping with the performance in training in terms of the particular objectives niche they have occupied.

   We use three metrics: the number of solutions that changed cluster, the distance cluster change, and the Spearman correlations on each of the objectives. Figure 5.13 shows the number of solutions that changed their cluster as a percentage of the Front size (the smaller the better). Only 31% of the diversity-enhanced-mating-restriction technique

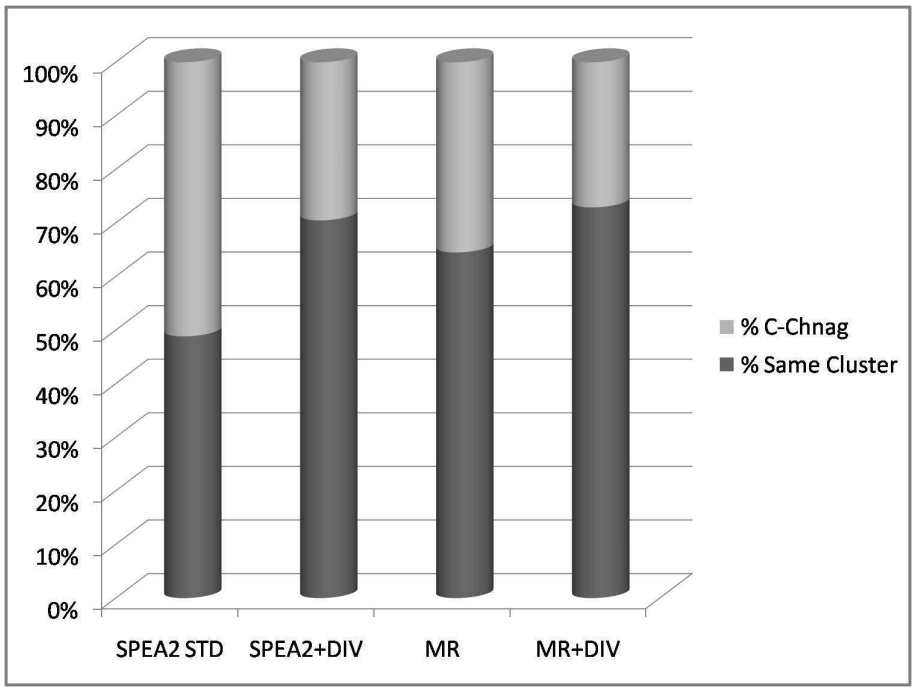


Figure 5.13: Points Changing Cluster

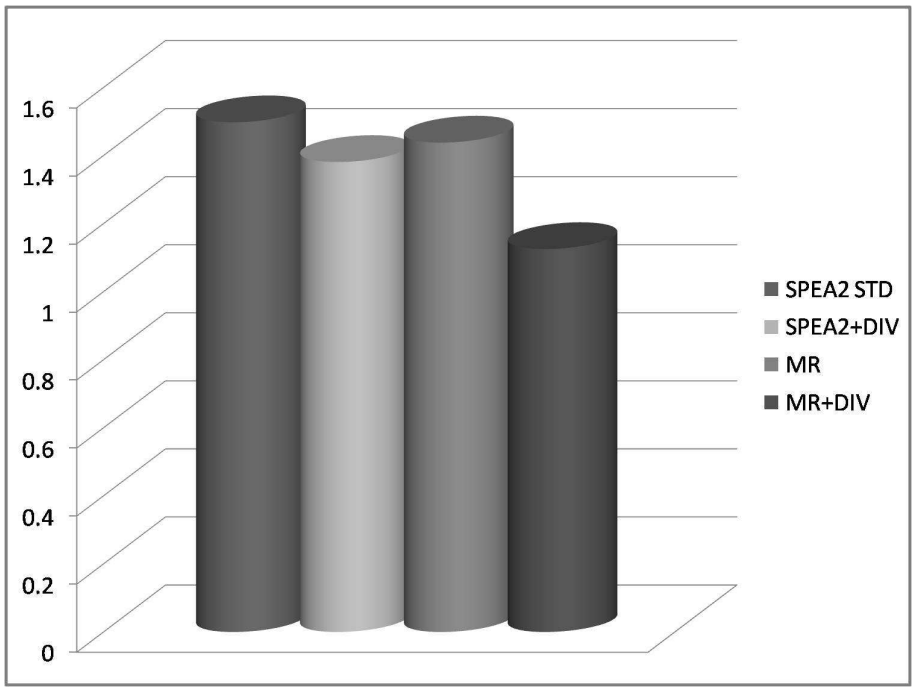


Figure 5.14: Average Distance Cluster Change

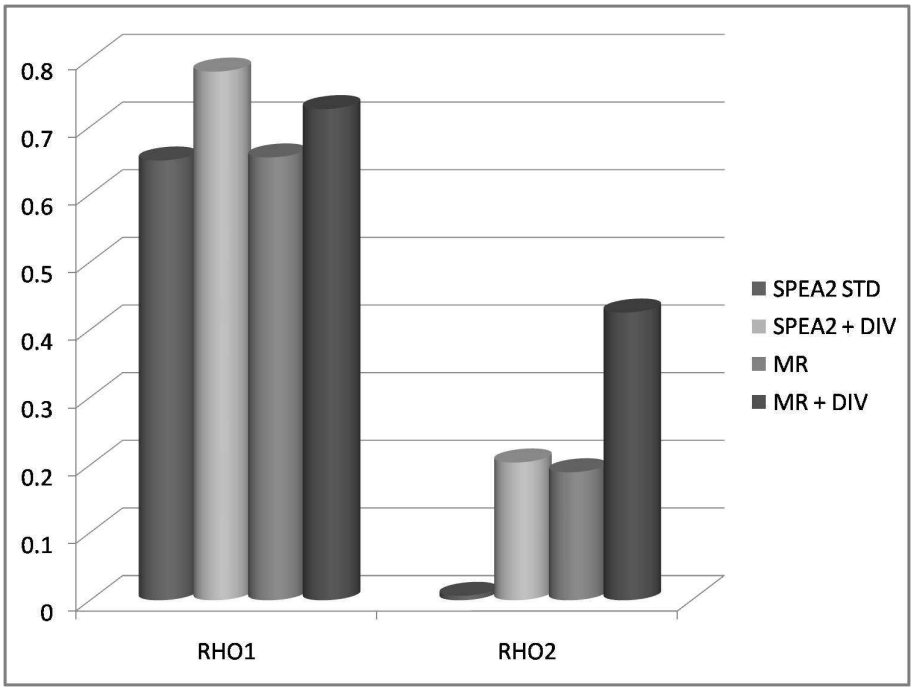


Figure 5.15: Spearman Correlation Coefficient

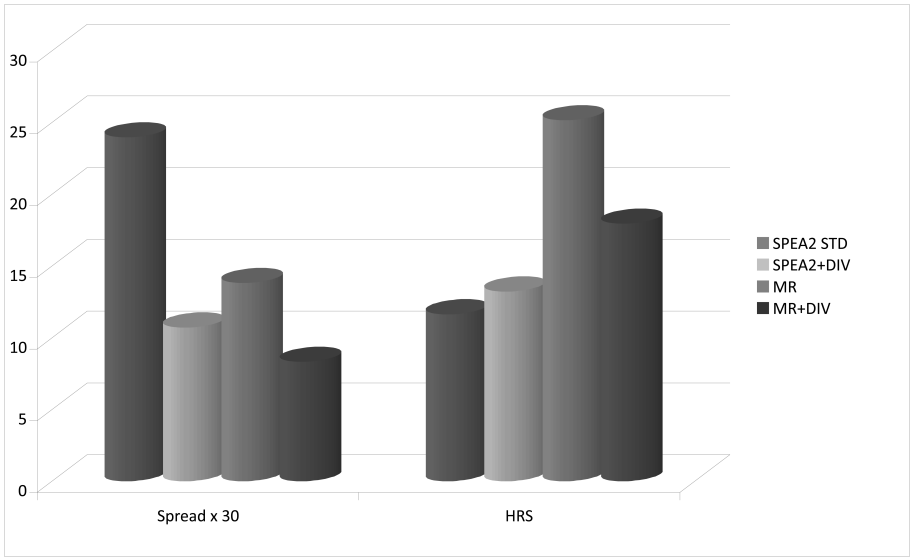


Figure 5.16: The HRS and Spread Metrics

Table 5.11: Statistical Test Results (Validation)

	Avg Dist Change	% Change	RHO1	RHO2	Sharpe
H	5.817	3.958	1.723	6.964	7.512
P	0.121	0.266	0.632	0.073	0.057
ω	11.44	0.29	0.16	0.225	0.44

have changed their cluster as opposed to 55% in the standard SPEA2.

Figure 5.14 shows the average distance cluster change (the smaller the better), and Fig. 5.15 shows the Spearman coefficient (the closer to 1 the better) for objective-1 (*Rho*1) and objective-2 (*Rho*2).

3. **Distribution of solutions on the front** – Measured using the spread and HRS metrics, where on both metrics smaller values are better. Figure 5.16 shows the average values achieved for the two metrics respectively. On the spread metric, standard SPEA2 achieved the worst, and MR+DIV achieved the best average value. However, on the HRS metric, the SPEA2 had the best value, and mating restriction the worst.

The results of the Kruskal-Wallis statistical analysis are given by H and P in Table 5.11 — the final row indicates the value of ω from a Tukey-Kramer test. For example: the Sharpe Ratio's ω value of 0.44 indicates that any two systems with Sharpe Ratio means differing by at least 0.44 are drawn from different populations with a significance given by the P-value (in this case 94%). These results indicate that SPEA2 and MR+DIV differ significantly in both the Sharpe Ratio (94%) and RHO2 (93%) (note that the results for Spearman correlation is good for all systems with no statistically significant difference).

5.3.5 Summary and Discussion

The robustness of a Multiobjective Genetic Programming (MOGP) algorithm such as SPEA2 is vitally important in the context of the real-world problem of portfolio optimisation.

We have analysed the robustness of individual solutions and of the Pareto front in terms of insensitivity to changes in the environment. We have demonstrated the problem by comparing a training environment with a very different validation environment, showing how SPEA2 solutions on the Pareto front can swap their relative positions in terms of their objectives cluster.

Three techniques to improve robustness were examined. In the first, one quantitative measures of robustness was utilized to create "R-SPEA2", a more robust variant of SPEA2. The results of experiments show that R-SPEA2 offers a statistically *highly significant* improvement in the mean number of cluster changes experienced by individual solutions when moving from a training environment to a validation environment. In the second, diversity was increased through increasing the mutation rate throughout the MOGP run. In the third, a cluster-based mating restriction technique was employed in SPEA2. Results of the second and the third

technique indicate that diversity in MOGP generalization plays a positive role similar to that played in GP. We have found that the introduction of cluster-based mating restriction in addition to the increase in diversity provided the best generalization results while also greatly enhancing the quality of solutions as measured by the Sharpe ratio.

5.4 Optimum Tracking, Change Detection, and Analysis of Market Behaviour

In this section, we provide preliminary experiments on analyzing the behaviour of the MOGP in a continuously changing environment in Section 5.4.1. In addition, we present preliminary results on the use of the MOGP as an analysis tool for understanding market behaviour in Section 5.4.2.

5.4.1 Severity of Change in Dynamic Environments

In this section, we focus on analyzing the behaviour of a the MOGP in a continuously changing environment. In particular, the MOGP ability to track the optimum in a dynamic portfolio optimization problem. Specifically, we investigate the following:

1. The ability of the MOGP to track the optimum in a dynamic environment.
2. Whether the MOGP can make use of the knowledge gained from previous training stages? i.e When there is a change, is it better to start from a new randomly generated population or from a previously trained population?
3. How to measure the severity of change in the environment.

5.4.1.1 Historical Data

We partition the available data into 4 periods (environments) of 20 months each, as shown in Figure 5.17 by vertical dotted lines. This corresponds to an MOGP system whose environment changes every 20 months. The four environments are:

1. Env1: May 1999 – December 2000, represents a volatile bull market
2. Env2: January 2001 – August 2002, a bear market
3. Env3: September 2002 – April 204, starts with a bear market followed by a bull market.
4. Env4: May 2004 – December 2005, a very strong bull market

From the financial market point of view (as represented by the index–fund), the risk and return characteristics of the markets indicate how similar or different they are from each other. Hence two markets that exhibit high returns on investment with relatively low risk will be considered more similar that two markets where one is bullish and while the other exhibits

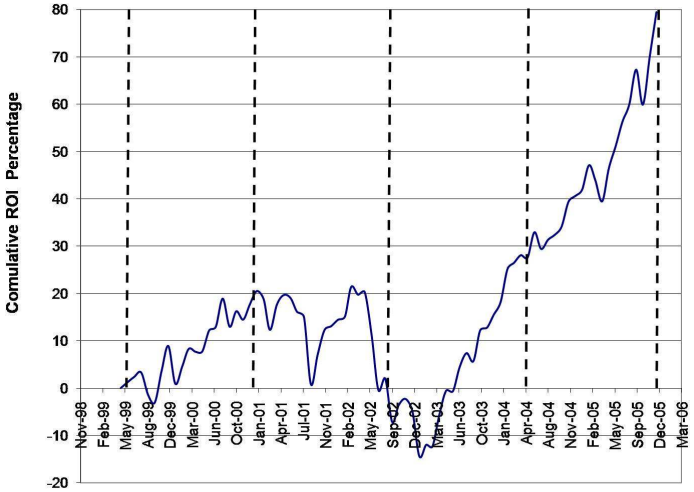


Figure 5.17: Market Index Return on Investment (ROI)

bearish behaviour with negative returns. According to this analysis, the index-fund depicted in Figure 5.17 would indicate that Env1 is most similar to Env4, very different from Env2, and somehow similar to Env3 (Env3 starts with a bear market followed by a bull market).

From the algorithm's search-space point of view, Pareto fronts that are closer to each other in the search space represent more similar environments than those whose Pareto fronts lay further apart. To inspect the location of the Efficient Frontiers of the four environments, we separately trained on each environment a population of 1000 individuals and allowed it to run for 50 generations. For each environment the experiment was repeated for 10 runs resulting in 10 Pareto fronts, which were then combined and the global non-dominated set was extracted. The resulting Pareto front was assumed to be the global efficient frontier for each of the four environments. Figure 5.18 depicts the four fronts and we observe that the Pareto front for Env2 is the furthest in space from the Env1 front, while the Env3 front has a similar wide spread of risks and returns as the Env1 front and the Env4 front is the closest to the lower section of the Env1 front. The behaviour of these four Pareto fronts therefore appears to roughly align with our knowledge of the four environments and our expectation, for example that Env1 is more similar to Env3 and Env4 than it is to Env2.

5.4.1.2 Proposed Measure for Severity of Change

After a change is detected, a measure of the severity of change is required. The accuracy of this estimation may influence the technique used to adapt to the change. If the severity is deemed low, the new optimum is possibly close to the old optimum and using the old population as a base for optimization with the addition of some diversity could be enough to locate the new optimum. If the two optimums are known, then a simple measure of distances between them will be a good indicator of the change severity. However, the actual optima are in practice

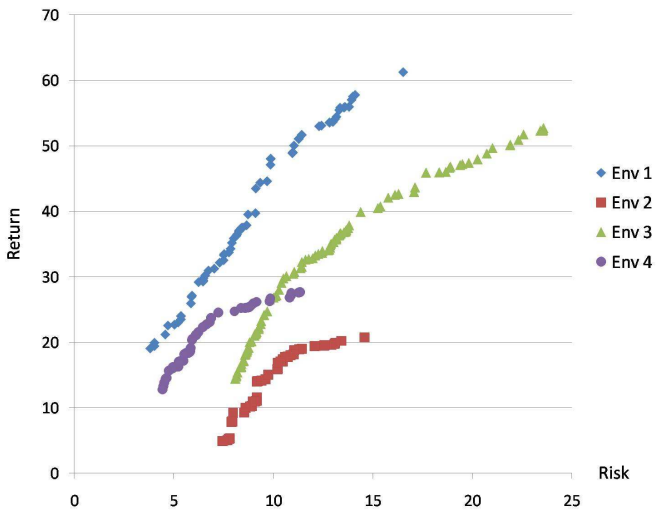


Figure 5.18: The Efficient Frontier in each of the Four Environments

not known in advance, and instead we track how far the system has moved from its previous optimum. For an MOGP system, this requires us to track the movement of the Pareto front. The following two measures for the severity of change are suggested and examined (as previously mentioned in Section 3.6):

1. **Shape**: Uses clustering techniques to divide the Pareto front solutions into three clusters; one representing the solutions which are low on all objectives (LL); the second representing solutions which are high on all objectives (HH); and the third is for solutions with medium values on all objectives (MM). The algorithm maintains and updates the centroids of the clusters. The distance between the corresponding centroids (in the old and the new environments) is measured and if it exceeds a certain threshold, then intervention in needed to help the algorithm adapt to the change. These three numbers (the movements of the three centroids) together provide a proxy for the position and of the Pareto front in the search space. In addition, because we are measuring the movements of the centroids of three separate clusters, this measure is also an indication of the changing ***s*hape** of the front, and it shows which portion of the front moved the most or the least or if all the whole front moved uniformly in space.

2. **Shuffle**: uses the Spearman correlation coefficient [MBB99] between the ranks of solution on the front of the old environment and the their ranks when the environment first changes (before any training on the new front happens). This measure assumes that a higher correlation value indicates stability of performance (notice that since we are using the ranks, this measure is independent of the actual objective values of the solutions, so the objective values may themselves change, but if the solutions ranks relative to each other remain relatively high, then the solutions are still valid). This measure gives an indication of the degree of ***s*huffle** that occurred on the front when the change first happened. We

Table 5.12: Raw (Normalised) Distance Between Cluster Centroids as a Proxy for Change in Location and Shape of Front. Lower values are better.

	Env1→Env2	Env1→Env3	Env1→Env4
LL Dist	24.768 (0.38)	12.292 (0.19)	8.971 (0.14)
MM Dist	30.77 (0.47)	17.945 (0.27)	20.694 (0.32)
HH Dist	39.593 (0.61)	24.889 (0.38)	32.546 (0.5)
Total Distance	95.137 (1.46)	55.126 (0.84)	62.212 (0.96)

measure the correlation for each objective separately.

Tables 5.12 and 5.13 give the **shape** and **shuffle** metrics (averages of 10 runs) that track the performance of an MOGP system that has been trained on Env1 and is then exposed to (validated in) Env2, Env3 and Env4.

Table 5.13: Shuffle: The Correlation between Solutions Ranks on Both Objectives a Proxy for Severity of Change. Higher values are better.

	Env1→Env2	Env1→Env3	Env1→Env4
Corr Risk	-0.21	0.15	0.46
Corr Return	0.55	0.41	0.44
Sum	0.34	0.57	0.89

The shape information in Table 5.12 is presented in both raw and normalised forms (normalised values given in brackets), and demonstrates that Env2 causes a much bigger movement in the solutions than the other two environments. The shape information also implies that in validation, Env3 solutions do not deviate as far from the original Pareto front as we might expect — the MM and HH cluster centroids both move less in Env3 than they do in Env4. The fact that this is not what we would expect from inspection of the index portfolio could be due to either (i) that similar equations will perform well in both Env1 and Env3 but not in Env4, or (ii) that the index portfolio is not a good proxy for changes in the environment, or (iii) that the new metrics are providing more information than the index portfolio, since they are not just looking at change in the environment but how well the system responds to that change. We find the latter explanation more plausible.

The shuffle information in Table 5.13 shows that Env4 solutions are reasonably well correlated (to their ranks reported in Env1 training) for both objectives, as well as when both

correlations are added. By contrast, in both Env2 and Env3, the risk objective displays significant shuffling. We will return to this point later.

5.4.1.3 Experiments and Results

We run three separate experiments to investigate the effects of training the MOGP in a dynamic environment:

1. The first experiment investigates the proximity to the optimal Pareto front of the solutions trained on Env1 when they are applied to the three validation environments.

2. In the second experiment, following initial training on the first 20 months of data (Env1), we retrain every 5 months, utilising a 20-month sliding window of training data;

3. In the third experiment, following initial training on the first 20 months of data (Env1), we retrain every 20 months, on each occasion using a fresh 20-month sample of training data (Env2, Env3 and Env4).

For the second and third experiments, we compare retraining starting from either (i) a random population or (ii) the previous population. This explores the research question "when retraining, is it better to start with the existing population or with a random population?"

Experiment 1

Figures 5.19, 5.20 and 5.21 each show the Env1 optimal front (top left), the new environment optimal front (middle) and the values given by the Env1 Pareto front solutions when they are evaluated in the new environment. We notice that in all cases the old solutions actually lie in the vicinity of the new Pareto front. However, they mostly seem to be clustered together and lack the spread required especially along the $x - axis$ which represents the risk. It could lead us to think that using these old solutions is better than starting with a totally random population.

Experiment 2 – A small change

Figure 5.22 presents the results of retraining every five months with a 5-month sliding window of training data. In this experiment, retraining from the previous population appears to have an advantage over retraining from a random population — the advantage is most pronounced in the early retraining periods. For simplicity of presentation the Sharpe ratio is used (a combination of both objectives) rather than presenting each objective separately.

Experiment 3 – A large change

Figure 5.23 presents the results of retraining every twenty months each time with a fresh 20 months of training data. In this experiment, retraining from the previous population appears to have little advantage over retraining from a random population, except for Env4 where retraining from the previous population appears to converge to a slightly higher Sharpe ratio than retraining from a random population . Again, for simplicity of presentation the Sharpe ratio is used (a combination of both objectives) rather than presenting each objective separately.

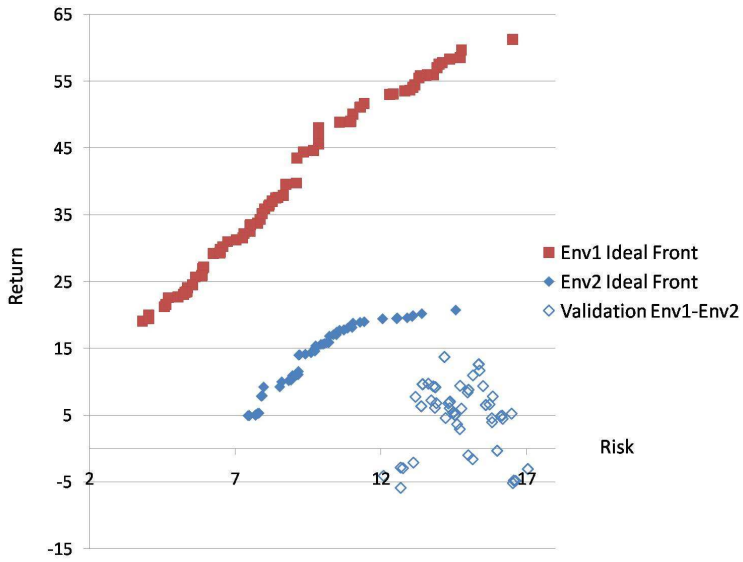


Figure 5.19: Performance of archive solutions evolved in Env1 when Env2 is introduced

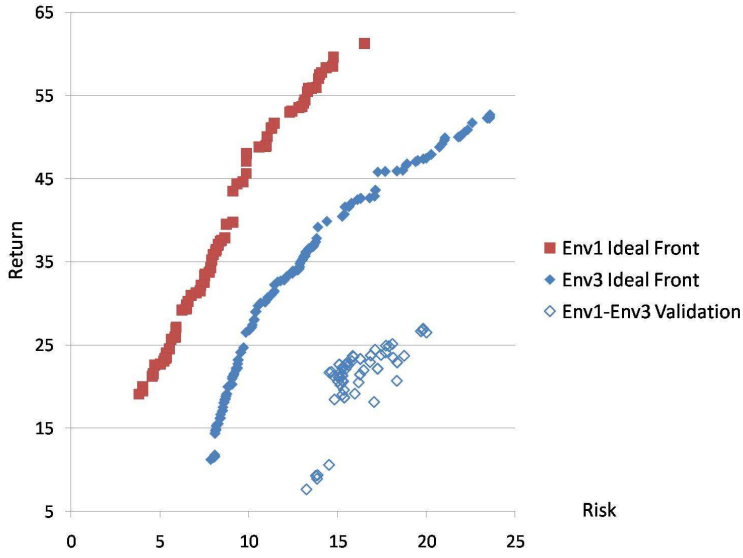


Figure 5.20: Performance of archive solutions evolved in Env1 when Env3 is introduced

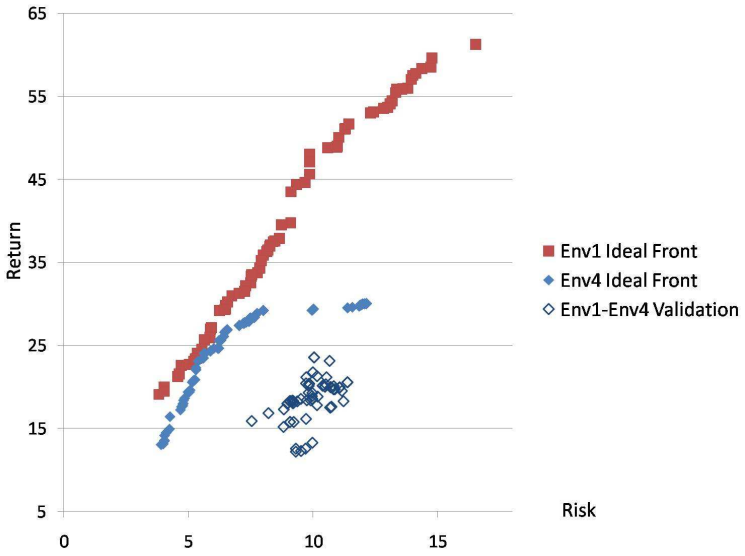


Figure 5.21: Performance of archive solutions evolved in Env1 when Env4 is introduced

Analysis of results in Experiment 3

It is interesting to recall that in Table 5.13 Env4 was the only environment to have reasonably high correlation values for both objectives — we therefore conjecture that a relationship might exist between the shuffle metric and the ability of the MOGP system to adapt to changing environments.

Although the Env3 Pareto front is not far from the Env1 Pareto front, Figure 5.23 shows a slight advantage for retraining from a random population for Env3. We conjecture that this might be due to the amount of shuffling present in the risk objective (possibly indicating a change in the market risk dynamics between Env1 and Env3).

Although the Env2 Pareto front distance metrics are high (high shuffling of the risk objective, and high distance moved), there appears to be no advantage in retraining from a random population. This is unexpected. We conjecture that the MOGP system finds it equally difficult to train on Env2, whether from a random population or from the previous population, because the market regime in Env2 was highly unusual — it includes the 9/11 attacks in the USA and consequent turmoil in the market data that is not explained by any fundamental corporate activity.

5.4.1.4 Summary and Discussion

Financial portfolio optimisation is a highly dynamic problem where the market data, and hence the fitness landscape, is continuously changing. Multiobjective algorithms are often used to track risk/reward trade-offs in portfolio optimisation, but it is not clear how well they are able to track the optimum Pareto front as the environment changes.

We provide two novel metrics for the severity of change — the first is based on the change in **shape** and position of the Pareto front when exposed to a new environment (using the

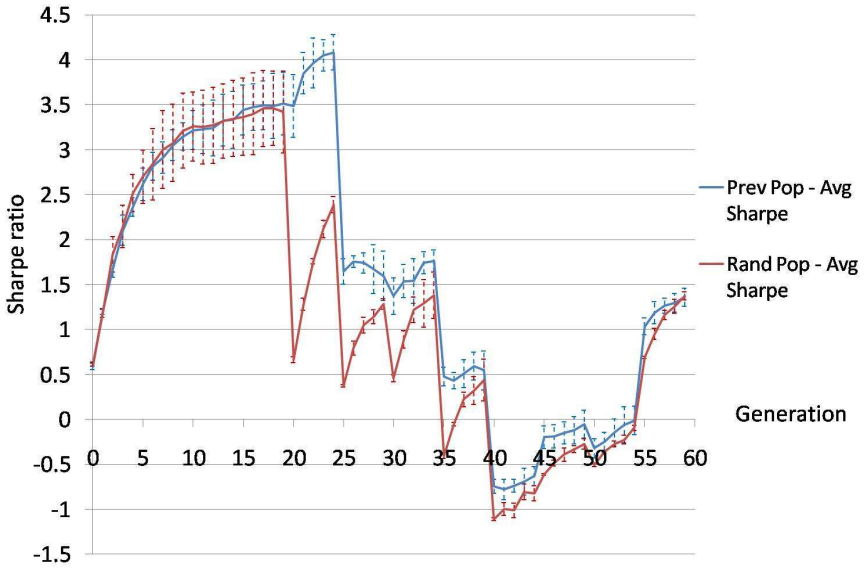


Figure 5.22: Retraining every 5 months (moving window) — previous vs. random population. With standard deviations.

movements of centroids of three clusters as proxies) and the second is based on the amount of **shuffle** amongst the solutions (using a rank Spearman correlation on the objectives before and after change). The first measure is a type of phenotypic distance measure. Intuitively we would assume that the further the centroids have moved, the larger the severity of change. The second measure investigates the stability of relative fitness values before and after a change.

To investigate optimum tracking, we use a real-world dynamic portfolio optimization problem and examine the performance of MOGP in two instances; in the first, the training data is changed slowly (through the use of a moving window), and in the second the MOGP is subjected to an abrupt changes in training data. Results of the experiments show that for the slow change, the MOGP population with knowledge from previous training was initially able to converge to higher Sharpe ratios than a population initialized with random individuals. This is possibly due to the fact that multiobjective algorithms by nature use techniques (crowding in the case of the SPEA2 algorithm [ZLT02]) to maintain diversity in the population. Although originally incorporated to ensure proper coverage of the Pareto front, such techniques also help in maintaining some degree of diversity that helps the population adapt to small changes.

We include two comparisons in this study — (i) to determine whether the new metrics provide information that correlates with our understanding of the changes that occurred in the financial markets during the period covered by the historical data, and (ii) to determine whether there is any relationship between the values provided by the new metrics and the behaviour of the MOGP system in three new environments. These comparisons resulted in the following unexpected results:

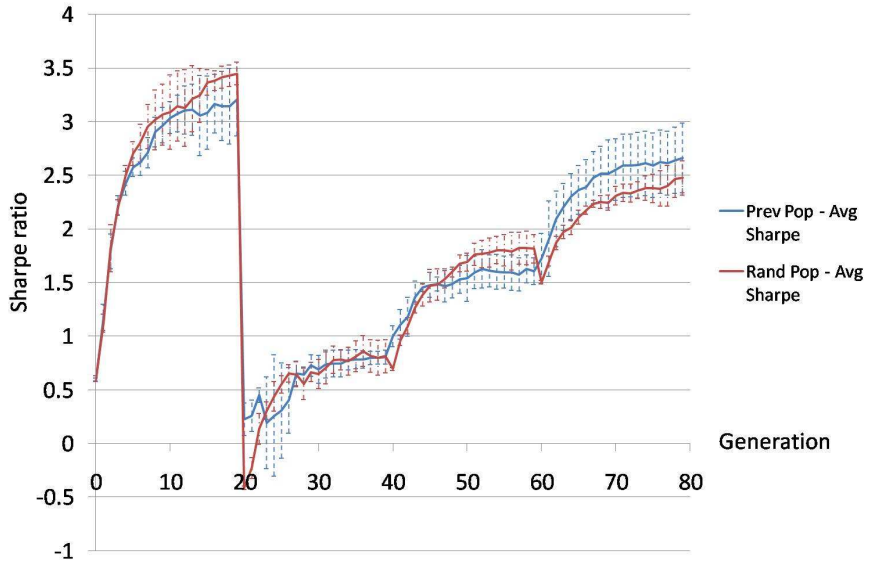


Figure 5.23: Retraining every 20 months (fresh data) — previous vs. random population. With standard deviations.

1. The portfolio index shows Env3 starting with a short bear trend, which implies that this environment differs from Env1 (a volatile bull trend), however the shape metric indicates that Env3 is closer than Env4 to Env1 (even though Env4 is a strong bull trend) — we hypothesize that the new metrics are providing more information than the index portfolio, since they are not just looking at change in the environment but how well the system responds to that change.

2. The shuffle information appears to be more useful than the shape information, in that (i) the one environment to have reasonably high shuffle correlation (Env4) was the only environment for which using a previous population for retraining produced better results, and (ii) the environment with a high degree of shuffle (i.e. a low correlation) demonstrated a slight advantage for starting with a random population.

3. Despite both metrics indicating high change for Env2, there appeared to be no clear advantage to retraining from either the previous population or from a random population, though this may be due to the effects of the 9/11 attacks which occurred during Env2.

These are early results that provide an indication of the importance of **shuffle** as a metric of change. We continue to explore the sources of change, how to measure that change, and how to measure the effects of change. Future work will examine measures to guide the diversity injected into the population if the severity of change is beyond a certain threshold. In addition we are currently investigating the effect of change on the ability of MOGP to perform well on out of sample data (actual portfolio investment).

5.4.2 Preliminary Analysis of Factors Selected in Models Evolved by MOGP

Evidence for the effect of some technical and fundamental indicators exist in the literature. For example, French and French [FF96], and Fama [Fam96] reported that small stocks (small capitalization) outperform large stocks, and value stocks (high book to market ratio or low market to book ratio) outperform growth stocks in the majority of markets and time periods studied [9]. Their research is considered a landmark in multifactor models that explain asset returns.

Factor models evolved using MOGP have the advantage over some other machine learning algorithms (like neural networks, support vector machines, and GAs to an extent) that the models they evolve are relatively easy to interpret by experts in the field. To illustrate this capability, we investigate which factors were chosen by the MOGP to form the factor models for each of the three risk-return trade-off classes (High Return–High Risk, Medium Return–Medium Risk, Low Return–Low Risk).

5.4.2.1 Factors Selected

For this experiment, the MOGP system is trained on 60 months of the data for 10 separate runs. We inspected the factor models that constituted the Pareto front of each run at the end of training. Furthermore we classified them by their cluster (models yielding high risk–high return portfolios,... et cetera). For each cluster, we analysed its equations for the usage of the factors. Using the gathered data, we plotted a histogram for the frequency of factors that was used in 100% of individuals in each of the risk-return trade-off classes in the 10 runs. Results are presented in Figure 5.24.

The figure shows evidence of the effect of price momentum, change of return on equity, and share yield on asset ranking for risk-adjusted returns. The firm size effect is more evident on the medium and low risk/return strategies, and the moving-average-changes is more evident in high risk/return strategies. Some evolved factor models are shown in Appendix 6.5.

The histogram graph explains which factors are seen as important by the MOGP system and which are not. For example, the MOGP almost never uses the BVPS, volume [10], or dividend yield (it appears to have favoured the adjusted dividend yield in this case) and hence they are deemed unimportant in designing an equation to rank the attractiveness of stocks. Factors like return-on-equity, cash-share-yield and capitalization, price-momentum and moving-averages, by contrast, are judged as important for stock ranking.

[9] Growth stocks are those stocks that are currently growing with potential for continued growth. While value stocks are those that the market has under priced and have the potential for an increase when the market corrects the price.

[10] In some preliminary experiments we ran with the liquidity as an additional objective, volume was used in 100% of runs.

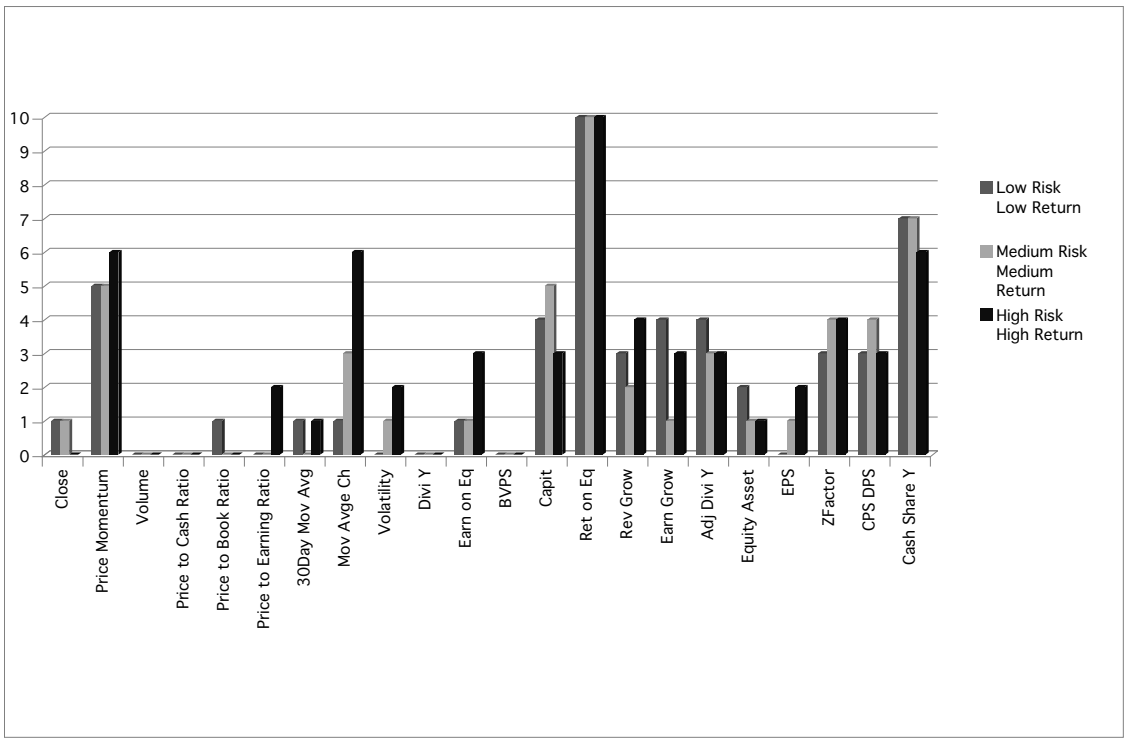


Figure 5.24: Histogram of Factors used in Investment Strategies Evolved - The y-axis indicates number of runs out of 10

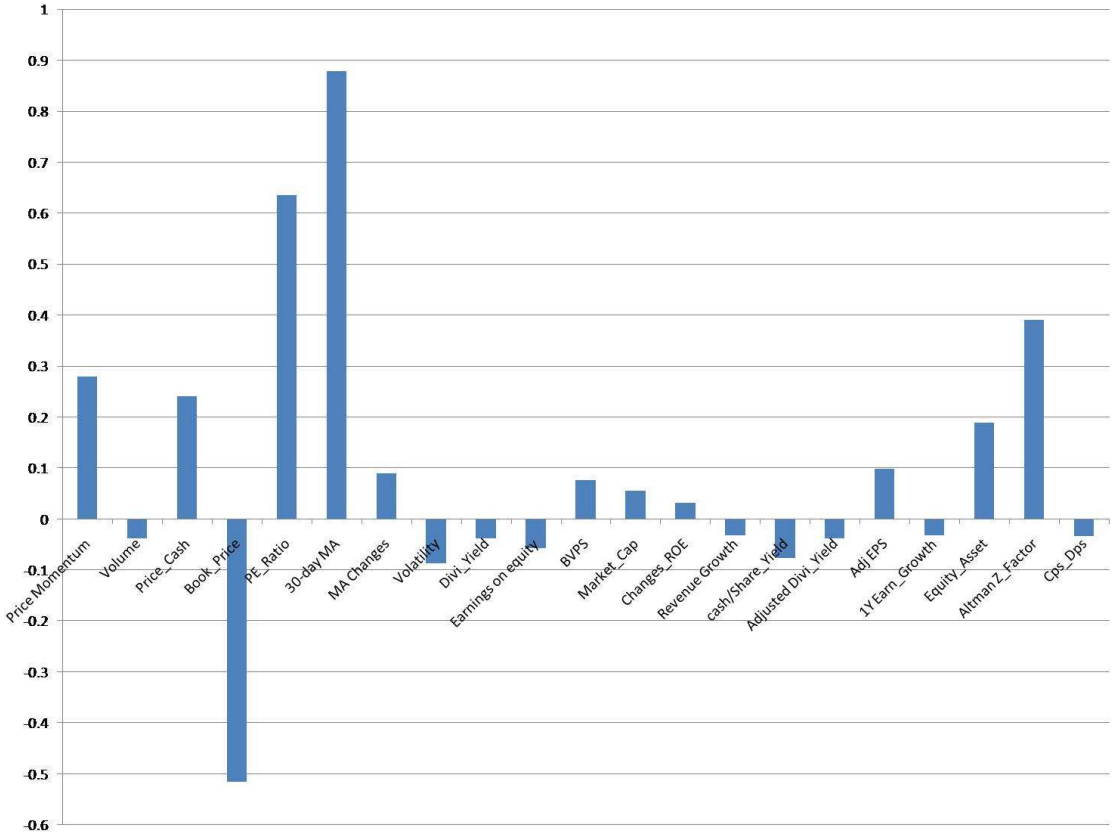


Figure 5.25: Correlation of Factors to Stock Price

5.4.2.2 Factors Correlation to Price and Pairwise Correlation

We also looked at the correlation between each of the financial factors available for the MOGP and the close price of its stock. We did the correlation analysis on data of a random sample of 15% of the stocks in our universe on the first 60 months of training data. Results of the correlation analysis are shown in Figure 5.25. This analysis is helpful in investigating if the evolved models are just taking advantage of the positive or negative correlation to discover the factors that contribute to a pricing models.

Comparing the Factor correlation to price, we find that the most positively correlated factor is the 30-Day moving average, followed by the PE-ratio factor. On the other hand, the book-to-market (book-to-price) ratio is negatively correlated to price (all explained by how the factors' values are calculated).

We have also calculated a matrix of pairwise correlation between factors to identify any factors that show high correlation [11]. Most factors have a correlation to other factors of around zero which indicates that they are independent from each other, with the exception of:

1. Price-Cash and Price-Momentum (0.9) – MOGP uses the second exclusively and not the first.

2. Change-ROE (Return on equity in the graph) and 1Y-Earn-Growth (Earn Grow in the graph) (0.8)

3. Dividend-Yield and Adjusted-Dividend-Yield (0.99) – Only the second is used.

4. Dividend Yield and Market Capitalization (-0.7) (Same correlation value observed for Adjusted Dividend Yield and Market Capitalization).

Using the histogram of frequency of factors usage as an indication of which factors the MOGP rated as important, and comparing that to the simple factor-price correlation diagram plotted above, we are further assured that the MOGP is not exclusively deriving simple relationships from the negative or positive correlations that exists between some factors and the price of the stock.

What the histogram does not say though is how these factors are used (for example the direction and strength of their correlation to the trading signal generated). In order to carry out this kind of analysis, individual equations need to be analysed in relationship to which stocks they buy or sell throughout investment, which is out of the scope of this thesis.

[11] In some cases, highly correlated factors are actually derived from each other as it is clearly the case with dividend-yield and adjusted-dividend-yield.

Chapter 6

Discussion and Conclusion

In dynamic and continuously changing environments, a solution evolved by training in one environment will in practice always be used in an environment different from that of training. Moreover, when assessing the performance of the solution set of a multiobjective algorithm in an out-of-sample environment, we are interested in both the performance of individual solutions from the solution sets as well as the solutions' collective average performance. The theme of this thesis was investigating the performance of an MOGP in a dynamic environment, developing proper metrics to quantify robustness of the MOGP solutions, and exploring techniques to improve the solutions' robustness in out-of-sample environments.

We have used the financial problem of portfolio optimisation as the case study throughout this thesis. This choice stems from: 1) Our interest to pursue research in the potential of machine learning techniques in general and MOEA in particular in financial quantitative analysis, and 2) Portfolio optimisation being a good representative of problems with both a highly dynamic environment and multiple objectives to be satisfied.

In this chapter, we start with a discussion on the research findings, the limitations and open questions in Sections 6.1 and 6.2. This is followed by a summary and the conclusion of the thesis in Section 6.3. Further work is outlined in Section 6.4, and the thesis contributions are restated in Section 6.5.

6.1 Robustness in Multiobjective Optimization

Assessment of the robustness of multiobjective solutions in this thesis was done in two stages:

- In the first stage, the collective average behaviour of the solution-set was compared against the performance of a random strategy and a buy-and-hold strategy. The results of this experiment have shown the following:
    1. The average performance of the solution set on the out-of-sample environment is statistically distinguishable from the random strategy. Hence, there is strong ground to believe that the algorithm is learning meaningful relationships from training data.
    2. The average performance of the solution set on the out-of-sample environment is better than the buy-and-hold strategy in the bear and volatile market, but is just as

good as the buy-and-hold strategy in the bull market.

The first result is simple but new, as we are not aware of any other research in the area of using MOEA in computational finance that attempted to refute the suspicion that the MOEA is discovering solutions due to luck[1]. The technique we used to devise the random strategy is based on the suggestion of [CN06], but is adapted to suit our application and to be as close as possible to the MOGP learning algorithm used (see Chapter 5).

The second result is consistent with what some other researchers have found. This result is possibly due to the negative effect of transaction costs associated with trading on the returns achieved by the evolved factor models. However, the average risk-adjusted returns achieved was not worse than the buy and hold strategy, although the latter is more diversified since the Index–Fund is composed of all 82 stocks and hence is possibly less volatile by design. In addition, the comparison with the average conceals the risk-adjusted range achieved by the multiobjective solutions. On inspection of the best Sharpe ratio achieved by the MOGP, it had higher values than the Index–fund and the difference is statistically highly significant in the three validation environments examined.

- In the second stage, we investigated the behaviour of individual solutions on the evolved front when applied to out-of-sample environments. We have demonstrated through experiments that solutions are prone to switching their relative positions on the Pareto front when evaluated in unseen environments, and explained how this behaviour is of substantial consequence to the practical use of the multiobjective algorithm. Other researchers have described the behaviour of the Pareto set as "chaotic" upon inspection of the out-of-sample results of technical trading strategies evolved using MOEA algorithm, for example Chiam et. al in [CTM09]. However, no previous work has analysed this behaviour on the level of the solutions themselves – we have done this analysis and provided metrics to measure the solutions "chaotic" behaviour.

We proceeded to investigate the effect of using four different techniques on the robustness of solutions. We have found that the combination of a mating restriction scheme based on phenotypic clustering, in addition to increased diversity, provided the best results for robustness (compared to the original SPEA2 and the three other techniques used), while also greatly enhancing the quality of solutions as measured by the Sharpe ratio. Results indicate that population diversity in MOGP plays a role similar to that played in GP regarding generalisation, and further investigation of other diversity enhancement techniques to improve MOGP generalisation should be worthwhile. The results also support our hypothesis that speciation occurs in MOGP and that preserving the niche characteristics can benefit robustness. Although the results are obtained from experiments in a financial domain, the embedded techniques are general in nature and we hope that they will extend to other domains. However, more

[1]Chen et. al [CN06] provides such a comparison for a single objective GP for the discovery of trading rules

experimental work needs to be done before generalisations can be made.

In Chapter 5, we stated that: if the stock-picking models evolved using an MOGP turn out to be not robust enough when used in environments different from that on which they were trained – causing individual solutions to switch their relative positions on the Pareto front – this behaviour can be due to one of two reasons:

1. The MOGP over-fits its evolved solutions to the training environment;
2. The risk-return exposure models themselves are time varying and a model that works in one time period is not guaranteed to work on another.

The results obtained from experiments in this research point in the direction of both reasons playing a role. Techniques linked in previous research to generalisation improvement in MOEA certainly improved on the switching characteristics of the solutions without jeopardising their quality. Nevertheless, robustness improvement will have an upper limit. We have seen from the last experiment that, even with a modest shift of the training data, as we move away from the original data set the performance starts to deteriorate. This is logical, as we cannot expect the rules evolved to perform well endlessly. Hence, we need techniques that tackle both issues. On one hand, the robustness needs to be improved so that using individual solutions from the MO Pareto set becomes practical and profitable, and gives sufficient time for more data points to be gathered that would be adequate for training once it is apparent that re-training is required. On the other hand, further improvements in this field of research will be in the direction of proper and well timed detection of change in the financial market, and consequently improving the ability to make the correct decision in response to the change. The decision would then be either modifications in the algorithm to make it better at retraining, or the more extreme re-start of optimisation through re-initialisation of the population from random.

Of critical importance at this point is analysing what constitutes a change in the financial market. For this purpose, financial indicators of change, indicators that measure solutions' performance deterioration, or monitoring changes that occur in the characteristics of the Pareto front could possibly be employed. In this work we have looked at two possible ways of characterising change and comparing the results of two techniques to the analysis of the environments in terms of bull and bear markets which are states characterised by persistent and statistically significant differences in mean returns [GPSW05]. The problem with using the bull and bear markets as indicators of different environments is that such analysis can only be done once the environment has been fully established. In addition, some researchers believe that bull and bear markets are merely the result of ex-post categorisation of the data [GPSW05]. The first metric of change was a measure of the distance that the cluster centroids move when the environment changes, and the second was a measure of the degree of shuffling in the objectives clusters when the environment changes. Counter to the intuition of the distance measure being a good proxy for the change in environment, results show that although the distance measure is higher for Env4 than in Env3, the previous population actually performs better on Env4 than

it does in Env3. We suspect that this result indicates that Env3 had a different market dynamics (as represented by the equations evolved by the algorithm), where in Env4, and although the range of possible risk and return allowed by the market is different from Env1, the underlying risk-return relationships were similar to those of Env1. The shuffle measure on the other hand seems to capture this type of change better. These are early results that provide an indication of the importance of "shuffle" as a metric of change.

6.2 Portfolio Management Using MOGP

With the vast number of stocks available to choose from, the extensive information publicly available about traded firms, ease of access to values of economic indicators, and the increasing effect international markets have on each other, the stock market investors' job is becoming more and more difficult. In order to correctly select assets for investment, we need to have a model to evaluate if a particular stock is worth investing in. Intuitively, when we are considering investing in a stock, we are mainly interested in the expected return and the associated risk. Two theories provide the foundation for analyzing the trade-off between risk and return. The Capital Asset Pricing Model (CAPM) [shar64] is a linear model that predicts the stock return to be associated with the stock's systematic risk, which is the risk that cannot be diversified away by holding a portfolio of inverse correlated assets. The CAPM assumes asset returns are (jointly) normally distributed, that variance is an adequate measurement of risk and no taxes or transaction costs are considered. The second theorem is the Arbitrage Pricing theorem (APT), [Ross76]. The APT is a generalized form of the CAPM. It draws a linear model of asset returns that depends on k multi factors, instead of a single factor of exposure to market risk as in the CAPM. It is essentially saying that the systematic risk of the CAPM should be modelled through sensitivity of the asset to several macroeconomic and/or fundamental factors, because there can hardly be one sole measure of risk. The APT, however, does not explicitly state what these factors are, which is reasonable, since we can envision the likelihood of change of factors that the risk depends on based on market, time period, period length, etc. Recently, some researchers [Qi99], [Ditt02],[Homm02], [Kana03], [McMi03], [Cece05], [Jone06] questioned the linearity framework of the model. It was shown that the market exhibits evidence for nonlinear behaviour which effects asset pricing and expected returns.

The MOEA in general provides a suitable machine learning tool for research in this area. The MOGP in particular has the advantage of being based on decision-tree-like structures. This work addressed the evolution of a nonlinear factor model using a multiobjective GP. The evolved model is used to generate buying and selling decisions in the UK stock market for constructing a portfolio and maintaining it for the investment period while closely monitoring market movement, and updating the portfolio accordingly. We have modelled the MOGP individuals as factor models that decide the attractiveness of stocks for buying or selling. The white box nature of the MOGP allows for the inspection of the evolved factor models, the deliberate insertion of factors or operations that finance practitioners believe are worth

investigating further, and in addition, we can analyse the monthly buying and selling decisions (for example whether a particular factor model is buying stocks when their prices are low and selling when they are high). The algorithm could also be used to investigate whether the evolved nonlinear models do provide improved understanding of market dynamics than what linear models are able to explain.

The previous reasonings in favour of using MOGP in financial optimisation problems are actually attributed to the GP segment of the algorithm. The multiobjective segment of the algorithm, on the other hand, gives the advantage of the tradeoff analysis between multiple objectives, as well as the production of solutions that span the tradeoff frontier, and out of which, individual solutions can be selected by fund mangers to suit different clients' attitude to risk. On the data set of the FTSE100 market that was used in this research, the MOGP solution set had (on average) a performance that either outperformed or was as good as the buy-and-hold strategy, which is a result that further strengthens the potential of MOEA use as an optimisation tool in financial applications. In addition, this research is a step towards improving the practical use of models evolved using MOEAs.

6.3 Summary and Conclusion

The main objectives of this thesis were to investigate the use of the multiobjective GP to evolve multifactor stock-ranking models in a dynamic and continuously changing environment, and to quantify the degree of robustness of the MOGP when validated on out-of-sample environments.

In pursuing this objective, we used a case study of the UK FTSE100 market data, and applied an MOGP algorithm to the evolution of factor models for stock selection in a financial portfolio management problem. The evolved solutions represent investment factor models of an underlying relationship between the financial factors considered. Due to the dynamic nature of the financial market, the optimal values of its efficient frontier are continuously changing. If these algorithms are to be judged useful in such a real world environment, the factor models evolved in the training phase must be *robust* in subsequent environments — they must remain reasonably profitable (at reasonable risk) for long enough to permit new data to be gathered for retraining.

The thesis provides detailed empirical results on the robustness of MOGP solutions in an unseen environments of real-world financial data. We have analysed the robustness of individual solutions and of the Pareto front in terms of insensitivity to changes in the environment and demonstrated the problem by comparing a training environment with different validation environments, showing how SPEA2 solutions on the Pareto front can swap their relative positions. The thesis then provides theoretical analysis of what constitutes robust behaviour of solutions in the multiobjective context and metrics to measure the robust behaviour of MOGP solutions and the Pareto front. The metrics are used in a two-objective optimization problem, but they can be generalised to problems with more than two objectives.

In Chapter 3 we demonstrated a robustness issue that is unique to multiobjective algo-

rithms, and we have provided definitions and metrics to quantify robustness in out-of-sample environments in the multiobjective context. In Chapter 4 we presented the system architecture used in the experiments, specified the financial factors used in the system and indicated the implementation details of our system. In Chapter 5 we explored four techniques to improve robustness of the MOGP solutions. In the first, one quantitative measure of robustness was utilized to create "R-SPEA2", a more robust variant of SPEA2. The results of experiments show that R-SPEA2 offers a statistically highly significant improvement in the mean number of cluster changes experienced by individual solutions when moving from a training environment to a validation environment. In the second technique, diversity was increased through increasing the mutation rate throughout the MOGP run and removing duplicates from the SPEA2 archive. In the third, a cluster-based mating restriction technique was embedded in SPEA2. The fourth technique was a combination of both diversity enhancement and cluster-based mating restriction. We have found that the last technique of cluster-based mating restriction, in addition to increased diversity, provided the best robustness results while also greatly enhancing the quality of solutions.

The results in the experiments are entirely based on empirical evaluation in the field of evolving stock selection rules for monthly investment and statistical analysis of the results. More theoretical analysis is needed to improve the understanding of factors that affect the robustness of multiobjective evolutionary algorithms and in particular the underlying causes that affect the switching behaviour of solutions when applied to out-of-sample environments. This research is a step towards this direction, and more understanding would improve their usability as optimisation tools in financial and other complex real world problems.

6.4 Future Research

For future research, it would be beneficial to expand the research to include training and validation of the MOGP in a variety of financial environments to confirm the results of the techniques used to improve robustness. We are also interested to pursue investigating a suitable measure for the severity of change in the financial context, and consequently account for the relationship between the shuffling happening on the front and the severity of change. Future work will also examine measures to guide the diversity injected into the population if the severity of change is beyond a certain threshold. In addition, we are currently investigating the effect of change on the ability of MOGP to perform well on out of sample data (actual portfolio investment).

6.5 Contributions

This thesis provides an empirical study of using an MOGP to evolve robust non-linear factor models for stock selection in a portfolio optimization problem with multiple objectives, and an assessment of the performance/robustness of the MOGP solutions when applied to out-of-sample data. It also demonstrates the value of an MOGP approach to a finance practitioner.

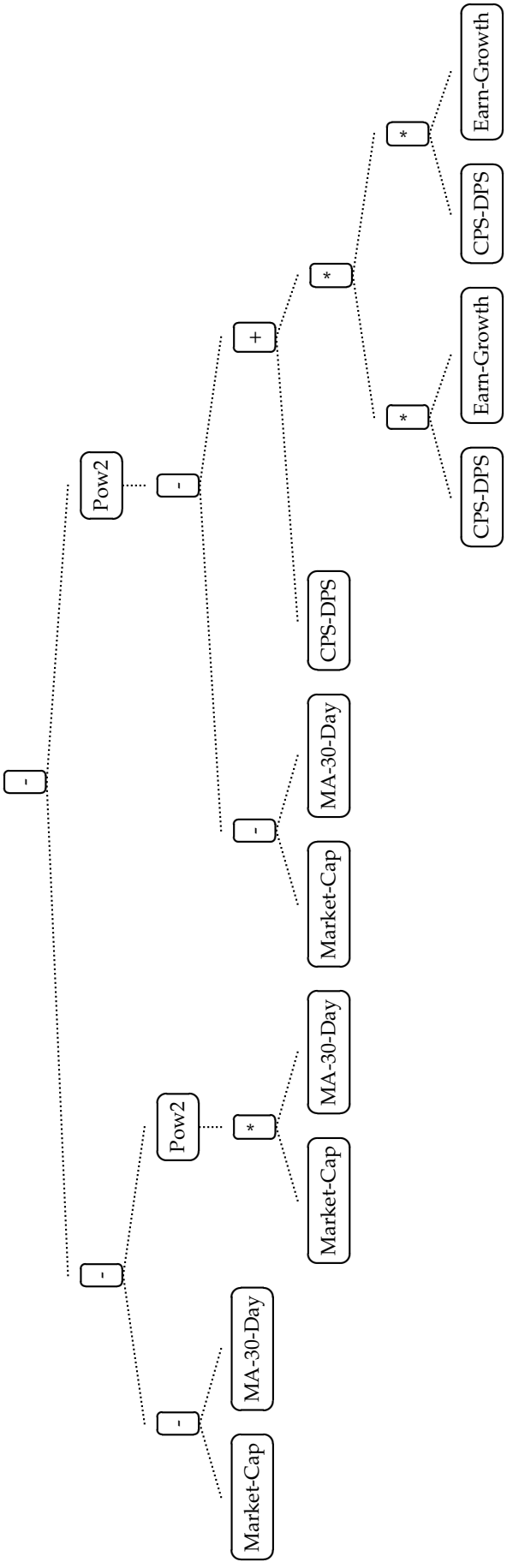
The thesis makes the following contributions:

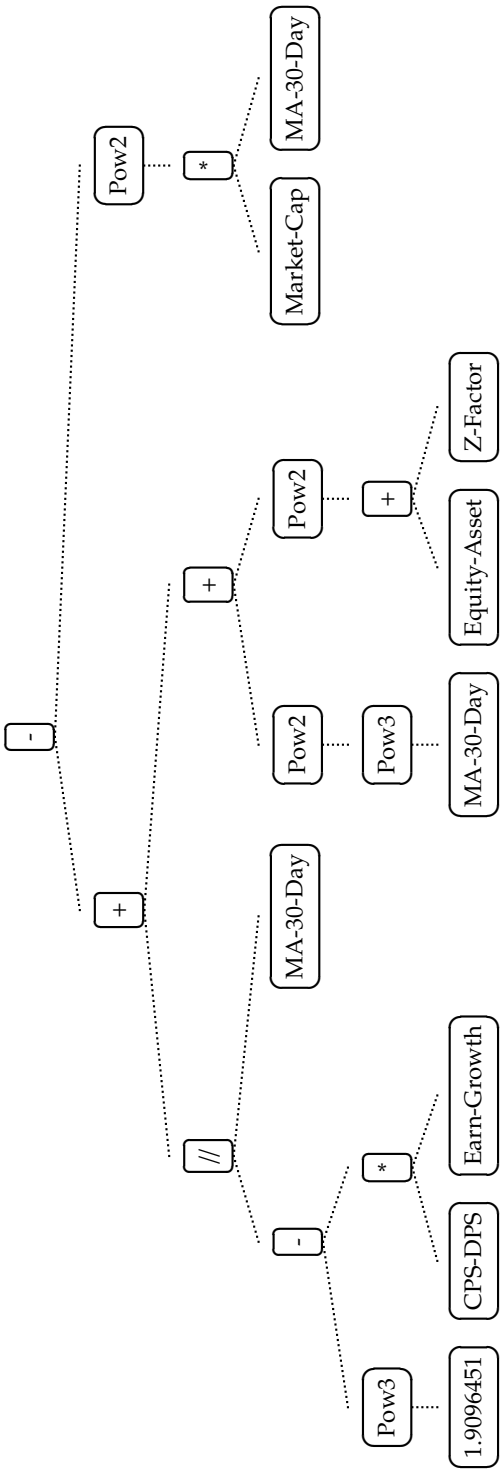
1. The development of new definitions and metrics for the robustness of MOGP solutions and robustness of the Pareto fronts in dynamic environments.
2. The use of the new definitions and metrics to assess the effect on robustness in unseen environments of:
   (a) Selection bias.
   (b) Diversity preservation.
   (c) Cluster-based mating restriction.
3. A preliminary analysis of:
   (a) The Dynamics of change.
   (b) How to quantify the severity of change in the financial environments.
   (c) The use of MOGP as an analysis tool in the financial market.

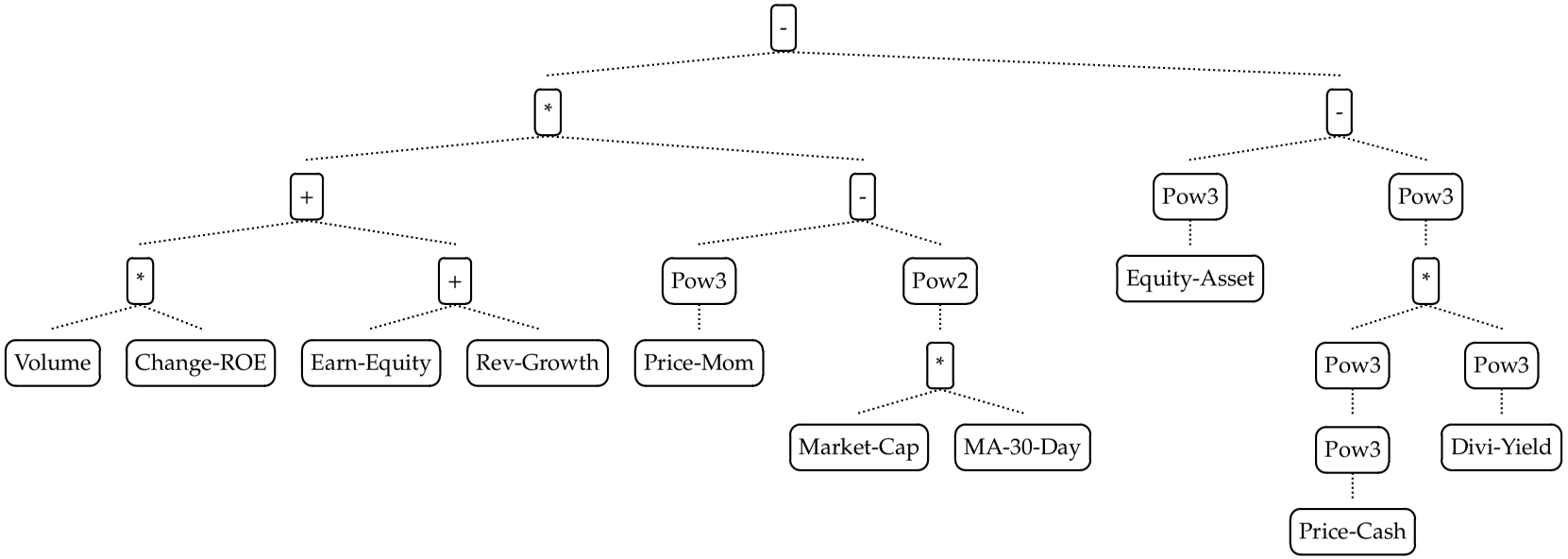
Appendix A

Sample MOGP Factor Models

Three sample trees from HH, MM, and LL clusters respectively:


-
-
-
Market-Cap
MA-30-Day
Pow2
*
Market-Cap
MA-30-Day
Pow2
-
-
Market-Cap
MA-30-Day
+
*
CPS-DPS
*
Earn-Growth
CPS-DPS
*
CPS-DPS
Earn-Growth


-
+
Pow2
*
MA-30-Day
Market-Cap
//
+
Pow2
+
Equity-Asset
Z-Factor
Pow2
Pow3
MA-30-Day
-
MA-30-Day
*
Earn-Growth
CPS-DPS
Pow3
1.9096451


-
*
-
+
-
Pow3
Pow3
*
+
Pow3
Pow2
Equity-Asset
*
Volume
Change-ROE
Earn-Equity
Rev-Growth
Price-Mom
*
Pow3
Pow3
Market-Cap
MA-30-Day
Pow3
Divi-Yield
Price-Cash

Bibliography

[aAD09] Matthew Butler ad Ali Daniyal. Multi-objective optimisation with an evolutionary artificial neural network for financial forecasting. In *GECCO'09: Proceedings of the annual conference on Genetic and evolutionary computation*, pages 1451–1457. ACM, 2009.

[AE04] V. S. Aragon and S. C. Esquivel. An evolutionary algorithm to track changes of optimum value locations in dynamic environments. *Journal of Computer Science and Technology*, 4(3):127–134, 2004.

[AK99] Franklin Allen and Risto Karjalainen. Using genetic algorithms to find technical trading rules. *Journal of Financial Economics*, 51:245–271, 1999.

[AL05] Ruben Armananzas and Jose Lozano. A multiobjective approach to the portfolio optimization problem. *IEEE Congress on Evolutionary Computation*, 2:1388–1395, 2005.

[AM10] K. P. Anagnostopoulos and G. Mamanis. A portfolio optimization model with three objectives and discrete variables. *Computes and Operations Research*, 37(7):1285–1297, 2010.

[Ang97] Peter Angeline. Tracking extrema in dynamic environments. In *6th Annual Conference on Evolutionary Programming VI*, pages 335–345. Springer–Verlag, 1997.

[AS03] Lee Becker Lee A. and Mukund Seshadri. GP–evolved technical rules can outperform buy and hold. *Proceedings of the 6th International Conference on Computational Intelligence and Natural Computing*, pages 26–30, 2003.

[BA06] Carlos Barrico and Carlos Henggeler Antunes. Robustness analysis in multi–objective optimization using a degree of robustness concept. *IEEE Congress on Evolutionary Computation*, July 16–21 2006.

[Bäc98] T. Bäck. On the behavior of evolutionary algorithms in dynamic environments. In *IEEE International Conference on Evolutionary Computation*, pages 446–451. IEEE, 1998.

[Ban81] Rolf W. Banz. The relationship between return and market value of common stocks. *Journal of Financial Economics*, 9(1):3–18, 1981.

[Bas97] S Basu. Investment performance of common stocks in relation to their price-earnings ratios: A test of the efficient market hypothesis. *Journal of Finance*, 32(3):663–82, 1997.

[BFF07] Ying L. Becker, Harold Fox, and Peng Fei. An empirical study of multi-objective algorithms for stock ranking. In Rick L. Riolo, Terence Soule, and Bill Worzel, editors, *GPTP V*, Genetic and Evolutionary Computation, chapter 14, pages 241–262. Springer–Verlag, 2007.

[BFL06] Ying Becker, Peng Fei, and Anna M. Lester. Stock selection : An innovative application of genetic programming methodology. In *GPTP IV*, volume 5 of *Genetic and Evolutionary Computation*, chapter 12, pages 315–334. Springer–Verlag, 2006.

[BFN96] Wolfgang Banzhaf, Frank D. Francone, and Peter Nordin. The effect of extensive use of the mutation operator on generalization in genetic programming using sparse data sets. In *In Parallel Problem Solving from Nature IV – Proceedings of the International Conference on Evolutionary Computation*, pages 300–309. Springer–Verlag, 1996.

[Bin07] Zafer Bingul. Adaptive genetic algorithms applied to dynamic multiobjective problems. *Applied Soft Computing*, 7(3):791–799, 2007.

[BJ08] Antonio C. Briza and Prospero C. Naval Jr. Design of stock trading system for historical market data using multiobjective particle swarm optimization of technical indicators. In *GECCO*, pages 1871–1878, 2008.

[BKK02] Edmund Burke, Graham Kendall, and Natalio Krasnogor. Advanced population diversity measures in genetic programming, 2002.

[BKSS00] Jrgen Branke, Thomas Kauler, Christian Schmidt, and Hartmut Schmeck. A multi-population approach to dynamic optimization problems. In *In Adaptive Computing in Design and Manufacturing*, pages 299–308. Springer–Verlag, 2000.

[BNKF98] Wolfgang Banzhaf, Peter Nordin, Robert E. Keller, and Frank D. Francone. *Genetic Programming – An Introduction; On the Automatic Evolution of Computer Programs and its Applications*. Morgan Kaufmann, 1998.

[BS96] Thomas Bäck and Martin Schutz. Intelligent mutation rate control in canonical genetic algorithms. In *Foundation of Intelligent Systems 9th International Symposium*, pages 158–167. Springer–Verlag, 1996.

[BS02] J. Branke and H. Schmeck. Designing evolutionary algorithms for dynamic optimization problems. In S. Tsutsui and A. Ghosh, editors, *Theory and Application of Evolutionary Computation: Recent Trends*, pages 239–262. Springer–Verlag, 2002.

[BSU05] J. Branke, E. Salihoglu, and S. Uyar. Towards an analysis of dynamic environments. In *Genetic and Evolutionary Computation Conference*, pages 1433–1439. ACM, 2005.

[BV93] Ravi Bansal and S. Viswanathan. No arbitrage and arbitrage pricing: A new approach. *The Journal of Finance*, 48(4):1231–1262, 1993.

[CAS04] Bana E Costa, Carlos A., and J. O. Soares. A multi-criteria model for portfolio management. *European Journal of Finance*, pages 198–211, 2004.

[CK03] Shu-Heng Chen and Tzu-Wen Kuo. Over fitting or poor learning: A critique of current financial applications. *EuroGP 2003*, pages 34–46, 2003.

[CKO00] David W. Corne, Joshua D. Knowles, and Martin J. Oates. The Pareto–envelope based selection algorithm for multiobjective optimization. *Proceedings of the Sixth International Conference on Parallel Problem Solving from Nature*, pages 839–848, 2000.

[CMB+98] T. Chang, N. Meade, J.E. Beasley, Y. M. Sharaiha, Morgan Stanley, and Dean Witter. Heuristics for cardinality constrained portfolio optimisation. *Computers and Operations Research*, 27:1271–1302, 1998.

[CML07] Swee Chiang Chiam, Abdullah Al Mamun, and Y. L. Low. A realistic approach to evolutionary multiobjective portfolio optimization. In *IEEE Congress on Evolutionary Computation*, pages 204–211, 2007.

[CN06] Shu-Heng Chen and Nicolas Navet. Pretests for genetic programming evolved trading programs: "Zero–Intelligence" strategies and "Lottery Trading". *LNCS 4243*, pages 450–460, 2006.

[Cob90] H. G. Cobb. An investigation into the use of hypermutation as an adaptive operator in genetic algorithms having continuous, time-dependent non-stationary environments. Technical Report AIC-90-001, Naval Research Laboratory, 1990.

[Coe05a] Carlos A. Coello. Evolutionary multi–objective optimization: Current state and future challenges. *5th International Conference on Hybrid Intelligent Systems*, 2005.

[Coe05b] Carlos A. Coello. Recent trends in evolutionary multiobjective optimization. In *Evolutionary Multiobjective Optimization: Theoretical Advances and Applications*, pages 7–32. Springer–Verlag, 2005.

[Coe06] Carlos A. Coello. *Evolutionary Multi–Objective Optimization and its Use in Finance*, chapter Handbook of Research on Nature Inspired Computing for Economy and Management. Idea Group Publishing, 2006.

[COT09] Mario Cámara, Julio Ortega, and Francisco Toro. Performance measures for dynamic multi-objective optimization. In *IWANN '09: Proceedings of the 10th Interna-*

tional Work-Conference on Artificial Neural Networks, pages 760–767. Springer–Verlag, 2009.

[Cov07] Michael W. Covel. *Trend Following*. Financial Times Press, 2007.

[CS03] Yann Collette and Patrick Siarry. *Multiobjective Optimization: Principles and Case Studies*. Springer–Verlag, 2003.

[CTM09] Swee Chiang Chiam, Kay Chen Tan, and Abdullah Al Mamun. Investigating technical trading strategy via an multi-objective evolutionary platform. *Expert Systems with Applications*, 36(7):10408–10423, 2009.

[CV97] W. Cedeno and V. Vemuri. On the use of niching for dynamic landscapes. *Proceedings of International Conference on Evolutionary Computation*, pages 361–366, 1997.

[DC01] Vasant Dhar and Dashin Chou. A comparison of nonlinear methods for predicting earnings surprises and returns. *IEEE Transactions on Neural Networks*, 12(4):907–921, 2001.

[DD97] I. Das and J.E. Dennis. A closer look at drawbacks of minimizing weighted sums of objectives for Pareto set generation in multi-criteria optimization problems. *Structural Optimization*, 14(1):63–69, 1997.

[DDB01] N. Drechsler, R. Drechsler, and B.Becker. Multiobjective optimization based on relationship Favour. *International Conference on Evolutionary Multi Criterion Optimization*, pages 154–166, 2001.

[Deb01] Kalyanmoy Deb. *Multi–Objective Optimization using Evolutionary Algorithms*. John Wiley and Sons, 2001.

[DG89] Kalyanmoy Deb and David E. Goldberg. An investigation of niche and species formation in genetic function optimization. In *Proceedings of the third international conference on Genetic algorithms*, pages 42–50. Morgan Kaufmann, 1989.

[DG05] Kalyanmoy Deb and Himanshu Gupta. Searching for robust Pareto–optimal solutions in multi–objective optimization. *Evolutionary Multiobjective Optimization*, pages 150–164, 2005.

[DG06] Kalyanmoy Deb and Himanshu Gupta. Introducing robustness in multi–objective optimization. *Evolutionary Computation*, 14:463–494, 2006.

[Dit02] Robert F. Dittmar. Nonlinear pricing kernels, kurtosis preference, and evidence from the cross section of equity returns. *The Journal of Finance*, 57:369–403, 2002.

[DJ99] K. A. De Jong. Evolving in a changing world. *International Symposium in Foundation of Intelligent Systems*, pages 512–519, 1999.

[DM92] Dipankar Dasgupta and Douglas R. Mcgregor. Nonstationary function optimization using the structured genetic algorithm. In *Parallel Problem Solving From Nature*, pages 145–154. Elsevier, 1992.

[DPAM00] Kalyanmoy Deb, Amrit Pratap, Sameer Agarwal, and T. Meyarivan. A fast elitist multi–objective genetic algorithm: NSGA2. *Proceedings of the Parallel Problem Solving from Nature VI Conference*, 2000.

[DPGM07] Kalyanmoy Deb, Dhanesh Padmanabhan, Sulabh Gupta, and Abhishek Kumar Mall. Reliability–based multi–objective optimization using evolutionary algorithm. In *LNCS – Evolutionary Multi–Criterion Optimization*, pages 1611–3349. Springer–Verlag, 2007.

[DRK06] Kalyanmoy Deb, Udaya Bhaskara Rao, and S. Karthik. Dynamic multi-objective optimization and decision-making using modified NSGA-II: A case study on hydrothermal power scheduling. In *EMO*, pages 803–817, 2006.

[DTLZ02] Kalyanmoy Deb, Lothar Thiele, Marco Laumanns, and Eckart Zitzler. Scalable multi-objective optimization test problems. In *in Congress on Evolutionary Computation CEC*, pages 825–830, 2002.

[EN02] Aniko Ekart and Sandor Z. Nemeth. Maintaining the diversity of genetic programs. *LNCS*, 2002.

[FA02] M. Farina and P. Amato. On the optimal solution definition for many–criteria optimization problems. *Proceedings of Fuzzy Information Processing Society*, 3:233–238, 2002.

[Fam96] Eugene Fama. Multifactor portfolio efficiency and multifactor asset pricing. *The Journal of Finance*, 31(4):441–465, 1996.

[FBOO07] Kai Fan, Anthony Brabazon, Conall O'Sullivan, and Michael O'Neill. Option pricing model calibration using a real-valued quantum-inspired evolutionary algorithm. In *GECCO '07: Proceedings of the 9th annual conference on Genetic and evolutionary computation*, pages 1983–1990. ACM, 2007.

[FF93a] Eugene F. Fama and Kenneth R. French. Common risk factors in the returns on stocks and bonds. *Journal of Financial Economics*, 33:3–56, 1993.

[FF93b] Carlos M. Fonseca and Peter J. Fleming. Genetic algorithms for multiobjective optimization: Formulation, discussion and generalization. *Proceedings of the Fifth International Conference on Genetic Algorithms*, pages 416–423, 1993.

[FF96] Eugene Fama and Kenneth R French. Multifactor explanation of asset pricing anomalies. *The Journal of Finance*, 51(1):55–84, 1996.

[FFK06] Frank J. Fabozzi, Sergio M. Focardi, and Petter N. Kolm. *Financial Modeling of the Equity Market*. Wiley Finance, 2006.

[GAM08] Alvaro Gomes, C. Henggeler Antunes, and A. Gomes Martins. Improving the responsiveness of nsga–ii in dynamic environments using an adaptive mutation operator — a case study. In *KES '08: Proceedings of the 12th international conference on Knowledge-Based Intelligent Information and Engineering Systems, Part I*, pages 90–97. Springer-Verlag, 2008.

[GCC07] A. Gaspar-Cunha and J.A. Covas. Robustness in multi–objective optimization using evolutionary algorithms. *Computational Optimization and Applications*, 2007.

[GD05] Himanshu Gupta and Kalyanmoy Deb. Handling constraints in robust multi–objective optimization. *IEEE Congress on Evolutionary Computation*, 1:25–32, 2005.

[Gol89] D. E. Goldberg. *Genetic Algorithms in Search, Optimization, and Machine Learning*. Addison–Wesley, 1989.

[GPSW05] Liliana Gonzalez, John G. Powell, Jing Shi, and Antony Wilson. Two centuries of bull and bear market cycles. *International Review of Economics and Finance*, 14(4):469–486, 2005.

[Gre92] John Grefenstette. Genetic algorithms for changing environments. In *Parallel Problem Solving from Nature 2, PPSN–II*, pages 137–144. Elsevier, 1992.

[Gre99] John Grefenstette. Evolvability in dynamic fitness landscapes: a genetic algorithm approach. In *CEC 99*, volume 3, 1999.

[GS87] David E. Goldberg and Robert E. Smith. Nonstationary function optimization using genetic algorithm with dominance and diploidy. In *Proceedings of the Second International Conference on Genetic Algorithms on Genetic algorithms and their application*, pages 59–68. L. Erlbaum Associates Inc., 1987.

[Has08] Ghada Hassan. Non–linear factor model for asset selection using multi objective genetic programming. In *GECCO '08 Workshop: Advanced Research Challenges in Financial Evolutionary Computing (ARC–FEC)*, pages 1859–1862. ACM, 2008.

[HNG94] Jeffrey Horn, Nicholas Nafpliotis, and David E. Goldberg. A niched Pareto genetic algorithm for multiobjective optimization. *Proceedings of the First IEEE Conference on Evolutionary Computation, IEEE World Congress on Computational Intelligence*, 1:82–87, 1994.

[Hol75] John H. Holland. *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control and Artificial Intelligence*. MIT Press, 1975.

[HRM00] Kellerer H., Mansini R., and Speranza M.G. Selecting portfolios with fixed costs and minimum transaction lots. *Annals of Operations Research*, 99:287–304, 2000.

[HW06] Iason Hatzakis and David Wallace. Dynamic multi-objective optimization with evolutionary algorithms: a forward-looking approach. In *GECCO '06*, pages 1201–1208. ACM, 2006.

[IL04] Antony W. Iorio and Xiaodong Li. A cooperative co-evolutionary multiobjective algorithm using non–dominated sorting. *GECCO 2004*, pages 537–548, 2004.

[IS03] Hisao Ishibuchi and Youhei Shibata. An empirical study on the effect of mating restriction on the search ability of EMO algorithms. In *LNCS*, pages 433–447. Springer–Verlag, 2003.

[JB05] Yaochu Jin and Jurgen Branke. Evolutionary optimization in uncertain environments – a survey. *IEEE Transactions on Evolutionary Computation*, 9(3):303–317, 2005.

[J.E90] J.E.Beasley. OR-Library: distributing test problems by electronic mail. *Journal of the Operational Research Society*, 41(11):1069–1072, 1990.

[JGSB92] Wilfried Jakob, Martina Gorges-Schleuter, and Christian Blume. Application of genetic algorithms to task planning and learning. In Reinhard Männer and Bernard Manderick, editors, *Parallel Problem Solving from Nature 2, PPSN–II*, pages 293–302. Elsevier, 1992.

[Kab00] M. A. Kaboudan. Genetic programming prediction of stock prices. *Computational Economics*, 6(3):207–236, 2000.

[Kan03] Angelos Kanas. Non–linear forecasts of stock returns. *Journal of Forecasting*, 22:299–315, 2003.

[KCO99] Joshua D. Knowles, David W. Corne, and Martin J. Oates. The Pareto archived evolution strategy: A new baseline algorithm for multiobjective optimization. *IEEE Congress on Evolutionary Computation*, pages 98–105, 1999.

[KCV02] Nattavut Keerativuttitumrong, Nachol Chaiyaratana, and Vara Varavithya. Multi–objective co–operative co–evolutionary genetic algorithm. In *LNCS*, volume 2439. Springer–Verlag, 2002.

[KE95] J. Kennedy and R. C. Eberhart. Particle swarm optimization. *Proceedings of IEEE International Conference on Neural Networks*, pages 1942–1948, 1995.

[KL07] Saku Kukkonen and Jouni Lampinen. Ranking-dominance and many-objective optimization. *IEEE Congress on Evolutionary Computation CEC*, pages 3983–3990, 2007.

[Koz92] John R. Koza. *Genetic Programming: On the Programming of Computers by Means of Natural Selection*. MIT Press, 1992.

[KR02] Rajeev Kumar and Peter Rockett. Improved sampling of the Pareto-front in multiobjective genetic optimizations by steady-state evolution: A Pareto converging genetic algorithm. *Evolutionary Computation*, 10(3):283–314, 2002.

[KUE05] A. Karaman, S. Uyar, and G. Eryigit. The memory indexing evolutionary algorithm for dynamic environments. In *Applications of Evolutionary Computing*, volume 3449 of *LNCS*, pages 563–573. Springer–Verlag, 2005.

[KY01] Angelos Kanas and Andreas Yannopoulos. Comparing linear and nonlinear forecasts for stock returns. *International Review of Economics and Finance*, 10:383–398, 2001.

[LAA05] Mian Li, Shapour Azram, and Vikrant Aute. A multi–objective genetic algorithm for robust design optimization. *GECCO*, 2005.

[Lan96] W. B. Langdon. Evolution of genetic programming populations. Research Note – University College London, 1996.

[Lau05] Diosan Laura. A multi–objective evolutionary approach to the portfolio optimization problem. *International Conference on Computational Intelligence for Modelling, Control, and Automation*, 2005.

[LBK07] Xiaodong Li, Jürgen Branke, and Michael Kirley. On performance metrics and particle swarm methods for dynamic multiobjective optimization problems. In *IEEE Congress on Evolutionary Computation*, pages 576–583, 2007.

[LC09] D. Lohpetch and D. Corne. Discovering effective technical trading rules with genetic programming: Towards robustly outperforming buy-and-hold. *World Congress on Nature and Biologically Inspired Computing NABIC*, pages 431–467, 2009.

[LC10] Dome Lohpetch and David Corne. Outperforming buy-and-hold with evolved technical trading rules: Daily, weekly and monthly trading. *EvoApplications*, 2010.

[LD01] Yan H Lin D, Wang S. A multiobjective genetic algorithm for portfolio selection problem. In *Proceedings of ICOTA*, 2001.

[Lei07] David J . Leinweber. Stupid data miner tricks: Over-fitting the S&P 500. *The Journal of Investing*, 16(1), 2007.

[LLL05] Dan Lin, Xiaoming Li, and Minqiang Li. A genetic algorithm for solving portfolio optimization problems with transaction costs and minimum transaction lots. In *ICNC (3)*, pages 808–811, 2005.

[L+15] Sean Luke et al. A java–based evolutionary computation research system, version 15. http://www.cs.gmu.edu/ eclab/projects/ecj/.

[LPM03] Andrew W. Lo, Constantin Petrov, and MartinWierzbicki. It's 11 pm.Do you know where your liquidity is? The mean-variance liquidity frontier. *Journal OF Investment Management*, 1(1):55–93, 2003.

[LS03] Tonny Lybek and Abdourahmane Sarr. Measuring liquidity in financial markets. IMF Working Papers 02/232, International Monetary Fund, 2003.

[LT99] Jin Li and Edward P.K. Tsang. Investment decision making using FGP: A case study. *Proceedings of the Congress on Evolutionary Computation*, 1999.

[LT06] Jin Li and Sope Taiwo. Enhancing financial decision making using multi–objective financial genetic programming. *Proceedings of IEEE Congress on Evolutionary Computation*, pages 7935–7942, 2006.

[LW06] Chun-an Liu and Yuping Wang. New evolutionary algorithm for dynamic multi-objective optimization problems. *LNCS*, pages 889–892, 2006.

[LZ09] Hui Li and Qingfu Zhang. Multiobjective Optimization Problems With Complicated Pareto Sets, MOEA/D and NSGA-II. *IEEE Transactions on Evolutionary Computation*, 13(2):229–242, 2009.

[Mac67] J. MacQueen. Some methods for classification and analysis of multivariate observations. In *Proceedings of the 5th Berkeley Symposium on Mathematical Statistics and Probability*, pages 281–297. University of California Press, 1967.

[MAL03] Burton G. MALKIEL. The efficient market hypothesis and its critics. *Journal of Economic Perspectives*, 17(1):59–82, 2003.

[MAP95] Kenneth A. De Jong Mitchell A. Potter. A cooperative co–evolutionary approach to function optimization. *Parallel Problem Solving from Nature*, 1995.

[Mar52] H.M Markowitz. Portfolio selection. *Journal of Finance*, 7(1):77–91, 1952.

[MB98] Silla Mullei and Peter Beling. Hybrid evolutionary algorithms for a multiobjective financial problem. *Proceedings of the 1998 IEEE International Conference on Systems, Man, and Cybernetics*, 4:3925–3930, 1998.

[MBB99] William Mendenhall, Robert Beaver, and Barbera Beaver. *Introduction to Probability and Statistics*. Duxbury Press, 1999.

[MBDM02] Amitabha Mukerjee, Rita Biswa, Kalyanmoy Deb, and Amrit P. Mathur. Multi–objective evolutionary algorithms for the risk–return trade–off in bank loan management. *International Transactions in Operations Research*, 9:583–597, 2002.

[MERTS06] Moral-Escudero, R. Ruiz-Torrubiano, and R. Suarez. Selection of optimal investment portfolios with cardinality constraints. *IEEE Congress on Evolutionary Computation, CEC*, pages 2382–2388, 2006.

[MF01] Patrick Monsieurs and Eddy Flerackers. Reducing bloat in genetic programming. In *Proceedings of the International Conference, 7th Fuzzy Days on Computational Intelligence, Theory and Applications*, pages 471–478. Springer–Verlag, 2001.

[MF04] P. Amato M. Farina, K. Deb. Dynamic multiobjective optimization problems: test cases, approximations, and applications. *IEEE Transactions on In Evolutionary Computation*, 8(5):425–442, 2004.

[Mit96] Tom M. Mitchell. *Machine Learning*. McGraw-Hill, 1996.

[Mor03] Ronald W. Morrison. Performance measurement in dynamic environments, 2003.

[MP02] N Maroney and A. Protopapadakis. The book-to-market and size effects in a general asset pricing model: Evidence from seven national markets. *European Finance Review*, 6:189–221, 2002.

[MTES01] Sheri Markose, Edward Tsang, Hakan Er, and Abdel Salhi. Evolutionary arbitrage for FTSE-100 index options and futures. In *Proceedings of the 2001 Congress on Evolutionary Computation CEC2001*, pages 275–282. IEEE Press, 2001.

[NDG+09] A.J. Nebro, J.J. Durillo, J. García-Nieto, C.A. Coello Coello, F. Luna, and E. Alba. Smpso: A new pso-based metaheuristic for multi-objective optimization. In *2009 IEEE Symposium on Computational Intelligence in Multicriteria Decision-Making (MCDM 2009)*, pages 66–73. IEEE Press, 2009.

[Nee99] Christopher Neely. Risk–adjusted, ex–ante, optimal technical trading rules in equity markets. *International Review of Economics and Finance*, 12(1):69–87, 1999.

[PSV04] Jean-Yves Potvin, Patrick Soriano, and Maxime Vallee. Generating trading rules on the stock markets with genetic programming. *Computers and Operations Research*, 31(7):1033–1047, 2004.

[PTP05] N.G. Pavlidis, D.K. Tasoulis, and V.P. Plagianakos. Computational intelligence methods for financial forecasting. *Brill Academic Publishers, Lecture Series on Computer and Computational Sciences*, 1:1–4, 2005.

[RC05] M. Reyes and C.A. Coello Coello. Improving pso-based multi-objective optimization using crowding, mutation and ϵ-dominance. In C.A. Coello, A. Hernández, and E. Zitler, editors, *Third International Conference on Evolutionary MultiCriterion Optimization, EMO 2005*, volume 3410 of *LNCS*, pages 509–519. Springer, 2005.

[Ric04] Hendrik Richter. Behavior of evolutionary algorithms in chaotically changing fitness landscapes. In *Proceedings of Parallel Problem Solving from Nature VIII*, volume 3242, pages 111–121. LNCS, Springer–Verlag, 2004.

[Ros76] S.A. Ross. The arbitrage theory of capital asset pricing. *Journal of Economic Theory*, 13(1):341–60, 1976.

[RRL85] Barr Rosenberg, Kenneth Reid, and Ronald Lanstein. Persuasive evidence of market inefficiency. *Journal of Portfolio Management*, 11:9–17, 1985.

[RS04] Olga Roudenko and Marc Schoenauer. Dominance based crossover operator for evolutionary multi-objective algorithms. *LNCS*, 3242:812–821, 2004.

[SBE+05] Raj Subbo, Piro Bonissone, Neil Eklund, S. Bollapragada, and K Chalermkraivuth. Multiobjective financial portfolio design: A hybrid evolutionary approach. *IEEE Congress on Evolutionary Computing*, 2:1722–1729, 2005.

[Sch85] J. David Schaffer. Multiple objective optimization with vector evaluated genetic algorithms. *Genetic Algorithms and their Applications: Proceedings of the First International Conference on Genetic Algorithms*, pages 93–100, 1985.

[SD94] N. Srinivas and Kalyanmoy Deb. Multiobjective optimization using nondominated sorting in genetic algorithms. *Evolutionary Computation*, 2(3):221–248, 1994.

[SDD07] Andre Sulfow, Nicole Drechsler, and Rolf Drechsler. Robust multi–objective optimization in high dimensional spaces. *Evolutionary Multiobjective Optimization EMO*, pages 715–726, 2007.

[SDGD01] F. Schmiedle, N. Drechsler, D. GroBe, and R. Drechsler. Priorities in multiobjective optimization for genetic programming. *Proceedings of the GECCO'01: 6th Annual Conference on Genetic and Evolutionary Computation*, pages 129–136, 2001.

[SF98] Terence Soule and James A. Foster. Effects of code growth and parsimony pressure on populations in genetic programming. *Evolutionary Computation*, 6:293–309, 1998.

[Sha64] William F. Sharpe. Capital asset prices: A theory of market equilibrium under conditions of risk. *Journal of Finance*, 19(3):425–442, 1964.

[Sha94] William F. Sharpe. The sharpe ratio. *Journal of Portfolio Management*, 21:49–58, 1994.

[SKN07] Prisadarng Skolpadungket, Dahal Keshav, and Harnpornchai Napat. Portfolio optimization using multi–objective genetic algorithms. *IEEE Congress on Evolutionary Computation, CEC*, 2007.

[SMS05] Frank Schlottmann, Andeas Mitschele, and Detlef Seese. A multiobjective model framework for the integrated management of financial risks. *Quantitative Methods in Finance Conference*, 2005.

[SP91] Gilbert Syswerda and Jeff Palmucci. The application of genetic algorithms to resource scheduling. *Proceedings of the Fourth International Conference on Genetic Algorithms*, pages 502–508, 1991.

[SQ05] Ralph E. Steuer and Yup Qi. Multiple objectives in portfolio selection. *Journal of Financial Decision Making*, 1(1):5–20, 2005.

[ST01] Massimo Santini and Andrea Tettamanzi. Genetic programming for financial time series prediction. In *Proceeding of EuroGP*, pages 361–370. LNCS 2038, Springer–Verlag, 2001.

[Sta80] Dennis Statman. Book values and stock returns. *The Chicago MBA: A Journal of Selected Papers*, 4:25–45, 1980.

[TGC07] Castillo Tapia, M Guadalupe, and Carlos Coello Coello. Applications of multi–objective evolutionary algorithms in economics and finance: A survey. *IEEE Congress on Evolutionary Computation*, pages 532–539, 2007.

[TLM+00] Edward P. K. Tsang, Jin Li, Sheri Markose, Hakan Er, Abdel Salhi, and Giulia Iori. EDDIE in financial decision making. *Journal of Management and Economics*, 2000.

[TM99] Krzysztof Trojanowski and Zbigniew Michalewicz. Evolutionary algorithms for non-stationary environments. In *In Proceedings of 8th Workshop: Intelligent Information systems*, pages 229–240. ICS PAS Press, 1999.

[TMJ04] Edward P.K. Tsang and Serafin Martinez-Jaramillo. Feature article: Computational finance. *IEEE Computational Intelligence Society*, pages 8–13, 2004.

[TS99] James D Thomas and Katia Sycara. The importance of simplicity and validation in genetic programming for data mining in financial data. In *Data Mining with Evolutionary Algorithms: Research Directions*, pages 7–11. AAAI Press, 1999.

[Wei02] K. Weicker. Performance measures for dynamic environments. In *Parallel Problem Solving from Nature*, volume 2439 of *LNCS*, pages 64–73. Springer–Verlag, 2002.

[WL09] Yu Wang and Bin Li. Investigation of memory-based multi-objective optimization evolutionary algorithm in dynamic environment. *CEC*, pages 1817–1824, 2009.

[YC06] W. Yan and C. D. Clack. Behavioral gp diversity for dynamic environments: an application in hedge fund investment. *Proceedings of the 8th annual conference on Genetic and Evolutionary Computation*, 1:1817–1824, 2006.

[YOS01] Jin Yaochu, Markus Olhofer, and Bernard Sendhoff. Dynamic weighted aggregation for evolutionary multiobjective optimization: Why does it work and how? *Genetic and Evolutionary Computation Conference, GECCO*, 2001.

[YTY08] Xin Yu, Ke Tang, and Xin Yao. An immigrants scheme based on environmental information for genetic algorithms in changing environments. In *IEEE Congress on Evolutionary Computation*, pages 1141–1147, 2008.

[ZL07] Qingfu Zhang and Hui Li. Moea/d: A multiobjective evolutionary algorithm based on decomposition. *IEEE Transactions on Evolutionary Computation*, 11(6):712–731, 2007.

[ZLT02] Eckart Zitzler, Marco Laumanns, and Lothar Thiele. SPEAII: Improving the strength Pareto evolutionary algorithm for multiobjective optimization. *Evolutionary Methods for Design, Optimization, and Control, CIMNE*, pages 95–100, 2002.

[ZT98] Eckart Zitzler and Lothar Thiele. Multiobjective optimization using evolutionary algorithms – A comparative case study. *Parallel Problem Solving from Nature*, pages 292–301, 1998.

[ZT99] Eckart Zitzler and Lothar Thiele. Multiobjective optimization using evolutionary algorithms – A comparative case study and the strength Pareto approach. *IEEE Transactions on Evolutionary Computation*, 3(4):257–271, 1999.

[ZTL+03] Eckart Zitzler, Lothar Thiele, Marco Laumanns, Carlos M. Fonseca, and Viviane Grunert de Fonseca. Performance assessment of multiobjective optimizers: An analysis and review. *IEEE Transactions on Evolutionary Computation*, 7(2):117–132, 2003.