Computational Intelligence in Financial Forecasting and Agent-Based Modeling: Applications of Genetic Programming and Self-Organizing Maps

By
Michael Kampouridis

A thesis submitted for the degree of
Doctor of Philosophy

School of Computer Science and Electronic Engineering
University of Essex

November 2011

To my mother, who left us early...

Acknowledgements

First of all, I would like to thank my supervisor Professor Edward Tsang for his help, patience and support all these years. In addition, I would like to thank Professor Shu-Heng Chen, who also guided me and supported me during the 6 months I spent at the National Cheng Chi University of Taiwan, and afterwards. I would also like to express my gratitude to Dr Alexandros Agapitos, for his support and advices, and also for providing me with his GP tool. Also, a great thank you to my family for their support, emotional and financial. Lastly, a special thank you to my wife Nicole-Shuning for her love and support all these years. It has been a long journey and I wouldn't have managed to be here without all of these people! A big thank you once again!

Abstract

This thesis focuses on applications of Computational Intelligence techniques to Finance and Economics. First of all, we build upon a Genetic Programming (GP)-based financial forecasting tool called Evolutionary Dynamic Data Investment Evaluator (EDDIE), which was developed, and reported on in the past, by researchers at the University of Essex. The novelty of the new version we present, which we call EDDIE 8, is its extended grammar, which allows the GP to search in the space of the technical indicators in order to form its trees. In this way, EDDIE 8 is not constrained into using pre-specified indicators, but it is left up to the GP to choose the optimal ones. Results show that, thanks to the new grammar, new and improved solutions can be found by EDDIE 8. Furthermore, we present work on the Market Fraction Hypothesis (MFH). This hypothesis is based on observations in the literature about the fraction dynamics of the trading strategy types that exist in financial markets. However, these observations have never been formalized before, nor have they been tested under real data. We therefore first formalize the hypothesis, and then propose a model, which uses a two-step approach, for testing the hypothesis. This approach consists of a rule-inference step and a rule-clustering step. We employ GP as the rule inference engine, and apply Self-Organizing Maps (SOMs) to cluster the inferred rules. After running experiments on real datasets, we are able to obtain valuable information about the fraction dynamics of trading strategy types, and their long and short term behavior. Finally, we present work on the Dinosaur Hypothesis (DH), which states that the behavior of financial markets constantly changes and that the population of trading strategies continually co-evolves with their respective market. To the best of our knowledge, this observation has only been made and tested under artificial datasets, but not with real data. We formalize this hypothesis by presenting its main constituents. We also test it with empirical datasets, where we again use a GP system to infer rules and SOM for clustering purposes. Results show that for the majority of the datasets tested, the DH is supported. Thus this indicates that markets have non-stationary behavior and that strategies cannot remain effective unless they continually adapt to the changes happening in the market.

List of Publications

REFEREED JOURNALS

- Kampouridis, M., Chen, S.-H., Tsang, E.: "Market Fraction Hypothesis: A Proposed Test", International Review of Financial Analysis, special issue on Complexity and Non-Linearities in Financial Markets: Perspectives from Econophysics, forthcoming. (Chapter 7) (Kampouridis et al, 2011b)
- Kampouridis, M., Chen, S.-H., Tsang, E.: "Microstructure Dynamics and Agent-Based Financial Markets: Can Dinosaurs Return?", Advances in Complex Systems. (Chapter 8)

REFEREED BOOK CHAPTERS

- Kampouridis, M., Chen, S.-H., Tsang, E.,"The Market Fraction Hypothesis under different GP algorithms", Information Systems for Global Financial Markets: Emerging Developments and Effects, IGI Global, 2011, (Invited book chapter) forthcoming. (Chapter 7) (Kampouridis et al, 2011c)
- Kampouridis, M., Chen, S.-H., Tsang, E.: "Market Microstructure: A Self-Organizing Map Approach for Investigating Behavior Dynamics under an Evolutionary Environment", in Brabazon, A., O'Neil, A. (Eds.), Natural Computing in Computational Finance, Volume 4, Studies in Computational Intelligence Series, Springer, 2011 (Invited book chapter) forthcoming. (Chapter 8) (Kampouridis et al, 2011d)

REFEREED PAPERS IN CONFERENCE PROCEEDINGS

- Kampouridis, M., Tsang, E.: "EDDIE for Investment Opportunities Forecasting: Extending the Search Space of the GP", In Proceedings of the IEEE Congress on Evolutionary Computation, p. 2019–2026, Barcelona, Spain, 2010. (Chapter 6) (Kampouridis and Tsang, 2010)
- Kampouridis, M., Tsang, E.: "Using Hyperheuristics under a GP framework for Financial Forecasting", in Carlos A. Coello Coello (Ed.) - Learning and Intelligent Optimization – LION 5, Lecture Notes in Computer Science, Rome, Italy, 2011. (Extension of Chapter 6) (Kampouridis and Tsang, 2011)

- Chen, S.-H., Kampouridis, M., Tsang, E.,"Microstructure Dynamics and Agent-Based Financial Markets", In T. Bosse, A. Geller, and C.M. Jonker (Eds.): Multi-Agent-Based Simulation XI, 11th International Workshop, Revised Papers, LNAI 6532, Springer, Heidelberg, pp. 121-135. (Chapter 7) (Chen et al, 2011)
- Chen, S.-H., Kampouridis, M., Tsang, E.: "Microstructure Dynamics and Agent-Based Financial Markets", In Tibor Bosse, Armando Geller, Catholijn M. Jonker (eds.), Proceedings of the 11th International Workshop on Multi-Agent-Based Simulation (MABS), p. 117–128, Toronto, 11 May 2010. (Chapter 7) (Chen et al, 2010)
- Kampouridis, M., Tsang, E. "Testing the dinosaur hypothesis under empirical datasets". In R. Schaefer et al., editor, Parallel Problem Solving in Nature (PPSN) XI, Part II, LNCS 6239, p. 199–208, Heidelberg, 2010. Springer. (Chapter 8) (Kampouridis et al, 2010c)
- Kampouridis, M., Chen, S.-H., Tsang, E.,"Market Microstructure: Can Dinosaurs Return? A Self-Organizing Map Approach under an Evolutionary Framework", in C. Di Chio et al. (Eds.): EvoApplications 2011, Part II, LNCS 6625, pp. 91–100. Springer, Heidelberg (2011). (Chapter 8) (Kampouridis et al, 2011e)
- Kampouridis, M., Chen, S.-H., Tsang, E.,"Investigating the Effect of Different GP Algorithms on the Non-Stationary Behavior of Financial Markets", IEEE Symposium on Computational Intelligence for Financial Engineering & Economics, 11-15 April 2011, Paris, France. (Chapter 8) (Kampouridis et al, 2011a)
- Kampouridis, M., Chen, S.-H., Tsang, E.: "Testing the Dinosaur Hypothesis Under Different GP Algorithms", In Proceedings of the UK Computational Intelligence (UKCI) Workshop, Essex, IEEE Xplore, 2010. (Chapter 8) (Kampouridis et al, 2010b)

EXTENDED ABSTRACTS

- Kampouridis, M., Tsang, E.: "Hyper-Heuristics for Investment Opportunities Forecasting", In Computational Management Science (CMS), London, 2008 (Chapter 6) (Kampouridis et al, 2008)
- Kampouridis, M., Chen, S.-H., Tsang, E.: The Market Fraction Hypothesis: A proposed test, In Proceedings of the Econophysics Colloquium 2010, Taipei, 2010. (Chapter 7) (Kampouridis et al, 2010a) (earlier version also presented in Complex'09 and CMS 2009-see below)
- Kampouridis, M., Chen S.-H., Tsang, E.: "Market fraction hypothesis: A proposed test", In Proceedings of the 9th Asia-Pacific Complex Systems Conference, Tokyo, 2009 (Chapter 7) (Kampouridis et al, 2009a)

- Kampouridis, M., Chen S.-H., Tsang, E.: "Market fraction hypothesis: A proposed test", In Computational Management Science (CMS), Geneva, 2009 (Chapter 7) (Kampouridis et al, 2009b)

TECHNICAL REPORTS

- Kampouridis, M., Tsang, E.: "EDDIE on Artificial Dataset", Technical Report CES492, University of Essex. (Chapter 6) (Kampouridis and Tsang, 2009)
- Kampouridis, M., Chen S.-H., Tsang, E.: "A summary for the Brock and Hommes 'Heterogeneous beliefs and routes to chaos in a simple asset pricing model' 1998 JEDC paper", Technical Report CES497, University of Essex. (Chapter 7) (Kampouridis et al, 2009c)

List of Works Under Review

- Kampouridis, M., Tsang, E.: "Investment Opportunities Forecasting: Extending the Grammar of a GP-based Tool", International Journal of Computational Intelligence Systems. (Chapter 6)
- Kampouridis, M., Alsheddy, A.: "On the investigation of hyper-heuristics on a financial forecasting problem", Annals of Mathematics and Artificial Intelligence, Springer. (Invited paper) (Extension of Chapter 6)
- Kampouridis, M., Glover, T., Rais Shaghaghi, A., Tsang, E.: "Deciding the optimal roll-out plan for the deployment of fiber optic networks", Engineering Optimization.

Contents

List of Figures

List of Tables

List of Algorithms

Acronyms

GP Genetic Programming
GE Grammatical Evolution
GA Genetic Algorithm
EDDIE Evolutionary Dynamic Data Investment Evaluator
BNF Backus Naur Form
MA Moving Average
TBR Trade Break Out
FLR Filter
Vol Volatility
Mom Momentum
MomMA Momentum Moving Average
RC Rate of Correctness
RMC Rate of Missing Chances
RF Rate of Failure
AARR Annualised Average Rate of Return
RPR Rate of Positive Returns
TP True Positive
TN True Negative
FP False Positive
FN False Negative
GDT Genetic Decision Tree
EMH Efficient Market Hypothesis
AMH Adaptive Market Hypothesis
MFH Market Fraction Hypothesis
SOM Self-Organizing Map
DH Dinosaur Hypothesis
SFI-ASM Santa Fe Institute Artificial Stock Market

Part I

Introduction

Chapter 1

Introduction

Computational intelligence (CI) is a part of the Computer Science field, which attempts to address complex problems of the real world applications by using nature-inspired computational methodologies. A number of different algorithms exist, such as artificial neural-networks, decision trees, support-vector machines, naive bayes, genetic programming, and reinforcement learning (Mitchell, 1997). CI techniques have been extensively applied to a variety of real-world problems and applications. This thesis focuses on three such cases, which come from the fields of finance and economics. The next section presents our motivation for investigating these specific problems.

1.1 Motivation

As we mentioned above, this thesis applies CI techniques to economics and finance. The first application comes from finance and is the well-known application of investment opportunities forecasting (a.k.a. financial forecasting). Traditionally, forecasting algorithms use predefined indicators from technical analysis (Edwards and Magee, 1992) to forecast the future movements of the price, with a pre-specified period length. The problem that arises in these cases is with the choice of the period length. No-one can guarantee that the length chosen is the optimal one. For instance, if we are using indicators from technical analysis, how could we decide that a 12 days Moving Average is preferable to a 15 days Moving Average? Experts could argue that the former has been extensively used in the market and has been proven to be effective. However, nobody could assure us that this length is actually the best. This thus motivates us to look for new methods, which would dynamically search in the space of technical indicators to find better solutions.

The second application of this thesis is motivated by the field of economics. This work is based on observations made under agent-based artificial financial models, regarding the fraction dynamics of trading strategies that exist in markets. These observations state that the fractions of trading strategy types keep swinging and as a result, in the long run, a "winner" type of trading strategy does not exist. This has significant implications, because it implies that it is not meaningful to look for forecasting rules in the market. However, the majority of these observations has not

been tested under real data. This thus motivates us to formulate these observations into a concrete hypothesis, which we call the Market Fraction Hypothesis (MFH), and test them under different international markets. Running these tests will provide us with valuable information about the fraction dynamics of financial markets, which can offer a better understanding of how the markets are structured.

The final application this thesis discusses is again from the field of economics, and is also based on observations made under agent-based financial models. These observations have again not been formalized before, which we attempt to do. The observations are related to the behavior of financial markets, which is said to be non-stationary. However, as these observations were made under an artificial market framework, this motivates us to test them under 'real' datasets and thus investigate the nature of financial markets. After formulating these observations into another hypothesis, called the Dinosaur Hypothesis (DH), we examine the markets' behavior dynamics in light of this hypothesis. This will then give us insight on how financial markets behave in the long run.

1.2 Thesis overview

This thesis is divided into 6 parts. The first part contains one chapter, which is the introduction of the thesis. Part II aims to provide a brief introduction on the two computational intelligence techniques used in this thesis: Genetic Programming (GP) and Self-Organizing Map (SOM). It contains two chapters, Chapter 2 and 3, and each chapter presents one technique. More specifically, Chapter 2 starts with a general description of the GP algorithm. It then focuses on specific topics such as GP representations, initialization methods, operators, breeding methods, and selection strategies. Finally, there is a short discussion on the different data types in Genetic Programming. Next, Chapter 3 focuses on SOM, which is the second computational intelligence technique used in this thesis. The chapter starts by presenting some general information about SOM, along with the main algorithm. Then it focuses on specific topics such as initialization methods, learning parameters, update rules, and topologies.

Part III presents a review on topics related to financial markets. It is divided into two chapters, Chapter 4 and 5. Chapter 4 provides information about financial forecasting and is related to the work which will be presented in Chapter 6. Chapter 5 provides information about agent-based financial models and is related to the work which will be presented in Chapters 7 and 8. More specifically, Chapter 4 presents the Efficient Market Hypothesis (EMH), and then continues by presenting a review on previous financial forecasting attempts. In addition, it presents the two methods that are widely used for financial forecasting, fundamental and technical analysis. Furthermore, this chapter discusses some computational intelligence techniques that have commonly been used for the purposes of financial forecasting. Finally, the chapter presents EDDIE, which is a Genetic Programming tool for financial forecasting. EDDIE is the GP tool used for the experiments in this thesis. Chapter 5 presents the different types of agent-based financial models. Observations from these models have led to the formulation of the MFH and the DH, the two hy-

potheses tested in Chapters 7 and 8. Finally, Chapter 5 provides some background information on these two hypotheses.

Moreover, it should be said that none of the chapters in Parts II and III pretend to be either an extensive, or a complete review of their respective topics. Their objective is only to present the topics that are closely related to this thesis.

Part IV presents the main contributions of the thesis, and is divided into three chapters: Chapter 6, Chapter 7, and Chapter 8. The first chapter, Chapter 6, focuses on the first problem we described earlier, the one of pre-specified period lengths in the indicators of forecasting tools. It presents a new version of the EDDIE algorithm, which we call EDDIE 8, which allows the GP to search in the search space of the indicators. EDDIE 8 is then compared with a previous version, EDDIE 7, under both empirical and artificial datasets. We finally present the results from these comparisons. Chapter 7 formalizes the MFH, by presenting its main constituents. It also presents in detail a new agent-based financial model, which we use in this thesis for testing the hypothesis. In addition, Chapter 7 suggests a testing methodology for the MFH, runs tests under 10 empirical financial datasets, and presents these results. Finally, Chapter 8 formalizes the DH, and also suggests a testing methodology for it. Tests again take place under the same 10 empirical datasets used in the previous chapter.

Part V contains one chapter, Chapter 9, which is the conclusion of the thesis. It summarizes the work done in this thesis and also discusses future work.

Part VI is the final part of the thesis, and is composed of the appendix.

Part II

Methods

Chapter 2

Genetic Programming

This section gives some background information for the first of the two main techniques used for the experiments of this thesis, which is Genetic Programming. We should highlight that we are not aiming at offering a long and extended review of this method. The scope of this chapter is to introduce the technique to the reader, and also to present the main parts of it, which we used to facilitate our experiments. When other methods than the ones we have used in this thesis exist, we briefly mention them and refer the reader to the relevant literature. The rest of this chapter is organized as follows: Section 2.1 presents some general information about Genetic Programming, and Section 2.2 discusses different Genetic Programming representations. Section 2.3 presents different initialization methods, Section 2.4 presents Genetic Programming operators, and Section 2.5 discusses different breeding methods. Section 2.6 presents the different selection strategies available, Section 2.7 presents the different data types in Genetic Programming, and Section 2.8 briefly discusses the different choices of grammar that a GP system can use. Finally, 2.9 concludes this chapter.

2.1 General Information

GP (Koza, 1992, 1994; Koza et al, 1999, 2003; Banzhaf et al, 1998; Poli et al, 2008) is an evolutionary technique inspired by natural evolution, where computer programs act as the individuals of a population. The GP process has several steps. To begin with, a random population is created by using terminals and functions appropriate to the problem domain, where the former are variables and constants of the programs, and the latter are responsible for processing the values of the system (either terminals or other functions' output). Some examples of functions are:

- Arithmetic operations (+, -, *, etc.)
- Mathematical functions (sin, cos, exp, log)
- Boolean operations (AND, OR, NOT)
- Conditional operator (If-Then-Else)
- Functions causing iteration (Do-Until)
- Comparison operators ($>$, $<$, $\geq$, $\leq$, $\neq$, =)
- Other domain-specific functions may also be defined

In the next step, each individual is measured in terms of a pre-specified fitness function. The purpose of assigning a fitness to each individual is to measure how well it solves the problem. In the following step, individuals are chosen to produce new offspring programs. A typical way of doing this is by using the fitness-proportionate selection, where an individual's probability to be selected is equal to its normalized fitness value (Koza, 1992). The individuals chosen from the population are manipulated by genetic operators such as crossover and mutation, in order to produce offspring. The new offspring constitute the new population. Finally, each individual in the new population is again assigned a fitness and the whole process is repeated again, until a termination criterion is met (usually a number of generations). At the end of this procedure (last generation), the program with the highest fitness is considered as the result of that run.

Koza (1992) summarized the above procedure in three main steps:

- Population initialization.
- Evaluation of each individual and fitness assignment.
- Selection of individuals in order to produce new offspring by the means of different operators. These offspring form the new population.

As we mentioned earlier, the last two steps are iterated for a number of generations.

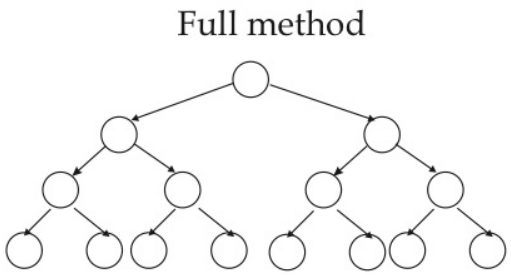


Figure 2.1: An example of a tree created with the Full method. Source: Garcia Almanza (2008)

2.2 GP Representations

Representation is the way to symbolize the genotype of the individuals. Originally, individuals were represented in the form of LISP S-expressions. Later, Koza used a tree-based GP representation, where a set of functions and constants form the individuals. This representation is widely used nowadays, especially after Koza demonstrated its efficiency in Koza (1992). In this work we use the tree-based GP representation, due to its popularity amongst researchers. Finally, other representations that can be found in the literature are linear GP (Cramer, 1985; Banzhaf, 1993), graph-based GP (Teller and Veloso, 1995), and Cartesian GP (Miller, 1999; Miller and Thomson, 2000) (Cartesian GP is actually a development of the graph-based GP).

2.3 Initialization of the population

Three widely used ways for initializing a GP population exist: Full, Grow and Ramped-Half-And-Half, and they were established by Koza (1992). In the first method, a tree root is randomly selected from the set of functions. Then this selected function takes other functions as children; this process is repeated until we reach one level before the maximum depth for the trees. Finally terminals are randomly selected, to form the leaves of that tree. As a result, trees created by this method have a regular shape. In other words, the length along any path from the tree root to any leaf is always the same. An example of a tree created by the Full method is shown in Figure 2.1. In the second method, Grow, a tree root is again randomly selected; however, this time selection can happen from either the functions or the terminals set. If the root is a function, then the arguments are filled with random functions or terminals. The same happens for any consecutive functions, until all branches are ending with terminals, as long as the maximum depth has not been reached. Obviously, the shape of trees created with the Grow method is now different, as they can have a variety of shapes. An example of such tree is presented in Figure 2.2.

Finally, the Ramped half-and-half method is a method that combines the previous two. More

specifically, for every depth (i.e. from two up to the maximum initial depth), half of the trees are created by using the full method and the other half by using the grow method. In this way, the population has both irregular and full tree shapes.

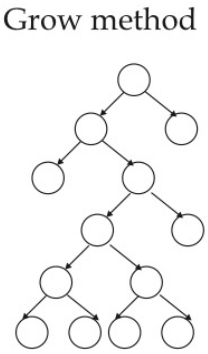


Figure 2.2: An example of a tree created with the Grow method. Source: Garcia Almanza (2008)

We should also mention that apart from Koza's initialization methods, trees can also be initialized by other methods, such as the uniform initialization (not used in this thesis). If the reader is interested in this method, we refer him to Iba (1996); Bohm and Geyer-Schulz (1996).

2.4 Genetic Operators

Banzhaf et al (1998) mentions that the three main genetic operators are: crossover, mutation, and reproduction. The latter is quite simple, since all it does is take an identical copy of an individual and put it into the new population. Crossover creates offspring by combining parts from two individuals, also called parents. After choosing the two parents, a node from each one of them is randomly selected. This point is called the cross-point. An example of the cross-point can be viewed in Figure 2.3. As a result, each tree can be divided into two parts: the subtree from the root to the cross-point, and the subtree from the cross-point and after. Therefore, two offspring can be produced as a consequence of the above crossover. The first offspring will be formed by the rooted subtree of the first parent and the subtree after the cross-point from the second parent. The remaining subtrees from the two parents form the second offspring. Figure 2.4 shows an example of a crossover.

The mutation differs from the crossover, in the sense that only one parent is needed instead of two. After selecting a node of a tree that is going to be mutated, the subtree after this point is removed and is replaced by a randomly generated subtree. Figure 2.5 presents a mutation example.

Other examples of operators are also Poli and Langdon's uniform crossover and one-point mutation (Poli and Langdon, 1998; Page et al, 1999), Chelapilla's multi-mutation (Chellapilla, 1997), and Lee and Yao's Lévy mutation (Lee and Yao, 2004). These operators are not used in this

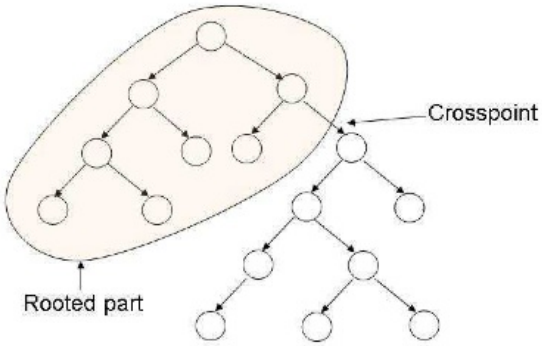


Figure 2.3: An example of a cross-point and the division in that tree. Source: Garcia Almanza (2008)

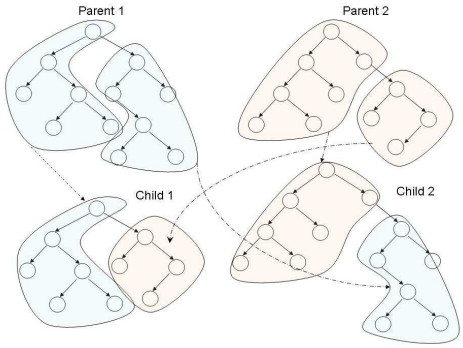


Figure 2.4: An example of the crossover operator. Source: Garcia Almanza (2008)

thesis and we thus do not discuss them here. If the reader is interested in them, we refer him to the relevant literature.

2.5 Breeding methods

Breeding methods are strategies that define which individuals are allowed to mate and also organize how the operators are going to be applied. There are two main criteria that are used for distinguishing among the different breeding methods:

- Mating restrictions between population members
- Type of operators used to create offspring, and the probability of their application

Let us start with the first criterion, which says that there can be mating restrictions between the population members. There are several categorizations, which can be organized into two different

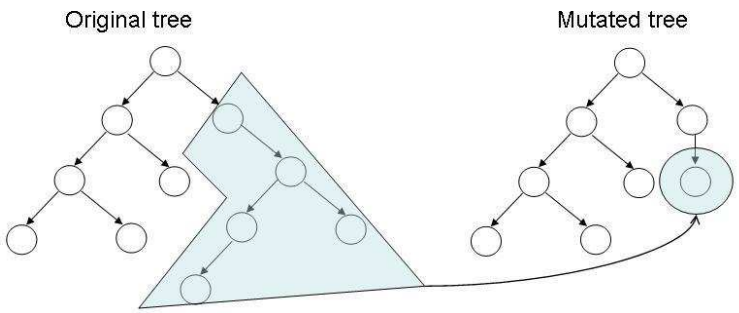


Figure 2.5: An example of the mutation operator. Source: Garcia Almanza (2008)

layers. In the outermost layer, breeding can be either generational or steady state (Reynolds, 1992; Syswerda, 1990). In generational breeding, the generations do not overlap with each other. First, all offspring are generated and form an intermediate population; this population then replaces the old one. In the case of steady state breeding, the new offspring are continuously added to the population; as a result, they can immediately be selected as parents in order to form new individuals. However, the fact that each new offspring is immediately added to the existing population does not mean that this population expands. Usually an existing member is removed and taken over by the new offspring. In this way, the population always remains the same size. In the next layer, breeding can be either panmictic (Godlberg, 1989) or have some form of restricted mating (Langdon, 1998). Panmictic means that every individual of the population is allowed to mate with any other individual from this population. Examples of forms of restricted mating can be demic sub-populations (Langdon, 2009), pygmy algorithms (Ryan, 1994), and genetic lineage strategies (McPhee and Hopper, 1999). In demic sub-populations the population is divided into demes. Then the majority of recombination happens between members of the same demes. A pygmy algorithm divides the population into two sub-populations. Crossover is then allowed only between individuals that do not belong in the same sub-population. Finally, genetic lineage strategies select parents based on their genetic lineage. For our work, we are using the generational, panmictic breeding method for its simplicity and its popularity in the GP literature (Agapitos, 2010).

Let us now move to the second criterion, where we can use different types of operators which are used depending on some probability settings (Poli et al, 2008). We saw earlier in Section 2.4 a few examples of genetic operators. The main distinction here is whether the operator uses a single parent (like mutation) or a set of parents (like crossover). The following section firstly presents the widely used breeding method of Koza's crossover extended with mutation, which is the breeding method used in this thesis; it also briefly presents other popular operator setups.

2.5.1 Koza's Crossover Extended with Mutation

A very typical breeding strategy is Koza's crossover operator, combined with mutation. Although Koza does not use mutation in his experiments (Koza, 1992), it has been shown to be beneficial using mutation with a low percentage, because it helps in 'refreshing' the population's genotypic content (Agapitos, 2010). Thus it is quite common to use a combination of crossover with mutation. Usually the probability for crossover is quite high (e.g. 90%) and the remaining percentage (e.g. 10%) is the probability for mutation. Individuals are selected via a tournament selection. There is no reproduction in this method, but there is *elitism*. This means that the best individual from every generation is directly copied to the next generation.

2.5.2 Other breeding methods

The previous breeding method is by no means the only one used in Genetic Programming. Other methods can be used as well, such as those based on different operators, such as Chellapilla's Multi-mutation, and Poli and Langdon's uniform crossover and point-mutation. Another breeding method that can be found in the literature is broad-selection (Altenberg, 1994). A good discussion of these breeding methods can be found in Agapitos (2010).

2.6 Selection Strategy

Selection strategies are used to select the individuals to whom we are going to apply genetic operators. It is based on the fitness of each program, in relation to the other members of the population. A common selection strategy is the roulette wheel selection, where the probability of an individual to be chosen is in proportion to its fitness value. Another commonly used selection strategy is the tournament selection. Here a subset of the population is chosen, according to a pre-defined tournament size. Then the members of this subset compete with each other and the best one (i.e. the one with the highest fitness) is selected. In this work, we use the tournament as our selection strategy.

2.7 Data Types in Genetic Programming

Koza states in Koza (1992) that in order for a GP to be applied to a specific problem, the concept of *closure* must be satisfied. This means that the arguments for functions, as well as the values returned from these functions, must be of the same data type. In this way, all functions can deal with any values that they receive as input. As a result, early works in this area required that only one type of function is used for generating trees. For instance, all functions should only return and accept boolean values.

In order to address the issue of closure, Koza suggested using *constrained syntactic structures*. These syntactic rules describe the terminals and non-terminals that can be used as the children

nodes of each non-terminal. Koza then enforced these syntactic constraints by applying them during tree initialization and while genetic operations were taking place. Similarly, *structure-preserving crossover* makes sure that any crossover between trees is legal. For instance, once the point of crossover from the first parent is chosen, the crossover point from the second parent is randomly chosen among points of the same type. In this way, subtrees that are going to be exchanged are going to be of the same type and thus any offspring derived from this crossover will be valid.

Soon after, more work took place in order to introduce further constraints in the data types in order to improve the performance of the GP. Montana (1995) extended Koza's idea by proposing the *Strong Typed Genetic Programming (STGP)*. As Montana says, Koza's GP and STGP are similar approaches, with the difference that the latter does not need to directly specify which children each non-terminal can have. Instead, this is done indirectly in STGP, by specifying the data types of the arguments of each non-terminal and the data types returned by terminals and non-terminals.

Finally, other works on data typing include branch typing and strong typing. More information about both of them can be found in Koza (1994).

2.8 Grammar-based GP

We mentioned in the previous section that in order to satisfy the issue of closure, Koza used constrained syntactic structures. Another way to provide a formal specification of syntactic constraints is to use 'grammar'. Grammar is a set of rules used to specify the syntax of a language, and thus specify syntactic constraints when building trees with GP. Some popular grammars that have been used for building GP trees are the Disjunctive Normal Form (DNF) (Gruau, 1996), the Context Free Grammar GP (CFG-GP) (Whigham, 1995), and the Backus Naur Form (BNF) (Backus, 1959) grammar. The last one, the BNF, is the grammar we have used for generating trees in this thesis. More details about the BNF grammar we have used can be found in Chapters 4 and 6.

However, what is important to say at this point is that apart from the obvious advantage of dealing with closure, the choice of grammar can limit the solutions of a system. This is because, depending on the details of the grammar, different types of trees (and thus solutions) are generated. Hence, there might be better solutions that a GP would never visit and thus detect, because of the limitations of its grammar. EDDIE (Tsang et al, 1998, 2000; Li and Tsang, 1999a; Tsang et al, 2005), which is a GP system developed at the University of Essex and used BNF grammar, faced these kinds of limitations due to the grammar it was using. Chapter 6 presents this problem in detail and also explains in what way we extended EDDIE's grammar and thus improved the performance of its trees.

2.9 Chapter Summary

In this chapter we presented Genetic Programming, which is a technique used for all of the research chapters in this thesis. We started by presenting some general information about the algorithm, and then continued by giving some more information on specific GP topics, such as GP representations, operators, breeding methods, selection strategies, data types, and grammar. The next chapter presents the second method that we have used in our experiments, Self Organizing Maps.

Chapter 3

Self Organizing Maps

This chapter presents the second technique used in this thesis, Self-Organizing Maps (SOM). We start with a short introduction, and then continue with the main parts of the algorithm. Finally, we present some well-known variants of SOM's learning parameters. We should again remind the reader, as we did in the previous chapter, that the goal of this chapter is not to offer an extensive tutorial and review of Self Organizing Maps. The purpose of this chapter is to introduce the algorithm, along with its main methods that have been used in this thesis in order to facilitate our experiments. The material presented in this chapter has mainly been based on Kohonen's work about SOM (Kohonen, 2001, 1982). Other useful works that present SOM are Fausett (1994); Ham and Kostanic (2001); Haykin (1999); Garson (1998); Deboeck and Kohonen (1998). The rest of the chapter is organized as follows: Section 3.1 presents some general information about SOM, while Section 3.2 presents the main SOM algorithm. Furthermore, Section 3.3 presents and discusses the different initialization methods of SOM, and Section 3.4 presents the different learning parameters and explains their variants. In addition, Section 3.5 discusses the different update rules used in the SOM algorithm, then Section 3.6 presents the different topologies used for visualization of the maps, and Section 3.7 presents the MathWorks Neural Network Toolbox, which is used for parts of the experimental work in this thesis. Finally, Section 3.8 summarizes this chapter.

3.1 General Information

SOM is a type of artificial neural networks (Bishop, 1995; Hassoun, 1995; Gurney, 1997) which takes an input data with high dimensionality, and returns a low-dimensional representation of these data, along with their topological representation. This representation is called a map. A self-organizing map consists of components called neurons. Associated with each neuron is a weight vector, which has the same dimensions as the input data.[1] During the SOM procedure, the weight

[1]Input data can be represented as vectors, matrices, continuous functions or even other SOMs. In this thesis, the input data consists of vectors. Thus, this chapter presents SOM under the assumption that the input data is vectors.

vector of each neuron is dynamically adjusted via a competitive learning process. Eventually, each weight vector becomes the center (a.k.a. centroid) of a cluster of input vectors. Thus, at the end of the SOM procedure, all input vectors have been assigned to different clusters of a map. One of the advantages of SOM is the preservation of topology, where inputs that are similar are grouped in the same clusters. In addition, points that are near each other in the input space are mapped to nearby clusters of the SOM. As a result, SOM can act both as a clustering tool and as a tool which allows us to visualize high-dimensional data.

3.2 SOM incremental-learning Algorithm

This section presents, in detail, the SOM process, which was just described above. This process is called the "SOM incremental-learning algorithm".

The first thing that needs to be done is to initialize the weights of the neurons. This initialization can happen in different ways, which are presented in Section 3.3. The rest of the algorithm constitutes two parts: learning and mapping. During learning, the input vectors are fed into the network. Then, the distance[2] of each input vector to all weight vectors is computed. The neuron with a weight vector closest to the input vector is called the best matching unit (BMU), and is the one that best represents the sample of weight vectors. The BMU neuron is, from this point on, also going to be referred to as neuron c. The weight vector of neuron c is then rewarded by being adjusted towards the input vector; the same adjustment also happens for the weight vectors of the neighboring neurons of c. The magnitude of this adjustment decreases with time and with distance from neuron c. This adjustment is given by Equation (3.1):

$$w_i(t+1) = w_i(t) + h_{ci}(t)[x(t) - w_i(t)] \tag{3.1}$$

where w_i is the weight vector of neuron i, x is the input vector, h_{ci} is the so-called neighborhood function of neuron c, which decreases monotonically,[3] and t is the discrete-time coordinate.

This process is repeated for λ number of times, for all input vectors. At the end of this learning process, the resulting neurons are associated with groups of input vectors.

Finally, the mapping process takes place. Here each input vector is allocated to the neuron whose weight vector has the smallest distance. Algorithm 1 summarizes this process.

3.3 Initialization

Initialization refers to the initial values of the weight vectors, before learning begins. As we have said, a weight vector has the dimensionality of the input vector. Traditionally, there are three different types of initialization: random, using initial samples, and linear. In addition to these

[2]There can be different ways of calculating this distance. More information about this can be found in Section 3.4.3.

[3]More details about the neighborhood function will follow in Section 3.4.1.

Algorithm 1 The "incremental-learning" SOM algorithm

Initialize the SOM's weight vectors, initialize neighborhood function
`/* Learning process */`
while $t < \lambda$ **do**
  for *each input vector* x **do**
    for *each neuron* i *in the SOM* **do**
      Calculate the distance between the input vector x and the weight vector w_i of the current neuron i
    end
    Return the weight vector of the neuron with the smallest distance from x (a.k.a. BMU cluster)
    Update the weight vector of the BMU and of the neurons in the neighborhood of BMU by using the formula:
    $w_i(t+1) = w_i(t) + h_{ci}(t)[x(t) - w_i(t)]$
    Update the neighborhood function h_{ci}
  end
  Increment t by 1
end
`/* Mapping process */`
for *each input vector* x **do**
  for *each cluster* i *in the SOM* **do**
    Calculate the distance between the input vector x and the weight vector w_i of the current cluster i
  end
  Map (place) the input vector x to the weight vector with the smallest distance
end

initialization methods, we also present the midpoint initialization, which is a built-in method in MATLAB's toolbox, which we will be using for our experimental work. Further details about this toolbox follow in Section 3.7.

3.3.1 Random Initialization

This is the simplest form of initialization. Each weight vector starts with random initial values. Initially, the reason for using this type of initialization was to demonstrate the strength of SOM: even with arbitrary values for the weight vectors, SOM will eventually manage to cluster the input vectors. However, this method is not necessarily the best or the fastest (Kohonen, 2001).

3.3.2 Initialization by using Initial Samples

In this type of initialization, we pick samples from the available input vectors x and use them as the initial weight vectors. The samples we pick can be arbitrary.

3.3.3 Linear Initialization

This type is based on linear algebraic principles, such as eigenvalues and eigenvectors. As presenting a full account of this method falls outside the scope of this thesis, this method is not further discussed. The reader is referred to the relevant literature, such as Kohonen (1982, 2001).

3.3.4 Midpoint Initialization

This weight initialization function is used by MathWorks Neural Network Toolbox (MathWorks, 2011), which is the SOM toolbox used in this thesis. More details about the toolbox follow in Section 3.7.

Midpoint initialization sets the weight vectors to the center of the input ranges. It takes two arguments: S, which denotes the number of neurons, and PR, which denotes an $R \times 2$ matrix of input value ranges equal to $[P_{min}, P_{max}]$. R is the number of elements of the input vector. Midpoint initialization returns an $S \times R$ matrix with weights set to $(P_{min} + P_{max})/2$. For instance, let us assume we wanted to calculate the initial weight values for a 5-neuron layer of input vectors with 2×1 dimension, where the first element of the vectors ranges over [0 1], and the second one ranges over [-2 2]. We could then set the 2 weights[4] of each neuron to take the midpoint of this range: $\frac{0+1}{2} = \frac{1}{2}$ and $\frac{-2+2}{2} = 0$, respectively. Therefore, each one of the 5 neurons would have its weight vector equal to $[\frac{1}{2}, 0]$.

3.4 Learning Parameters and their Variations

3.4.1 Neighborhood Function

As we saw earlier, during learning the weight vectors of the neurons that are topographically close to the BMU (i.e. in the neighborhood of BMU) are updated according to Equation (3.1). The purpose of this is that all these neurons will learn something from input x. This results in a smoothing effect on the weight vectors of these neurons. During this smoothing the important role of the smoothing kernel is given to the neighborhood function $h_{ci}(t)$. In the literature, there are two types of neighborhood functions that frequently occur: the first one refers to a *neighborhood set* of neurons around the BMU (neuron c), and the second one refers to a Gaussian function.

[4] A reminder that the dimensionality of the weight vector is the same as the dimensionality of the input vectors. Thus, in this example the weight vector is a 2×1 array.

Let us start with the first type. Each neuron that lies within a radius σ around neuron c belongs to a neighborhood that can be denoted by N_c. Figure 3.1 presents the neighborhood of Neuron 13 for radius $\sigma = 1$ (left) and for radius $\sigma = 2$ (right). Thus, the function takes values as follows: if $i \in N_c$, then $h_{ci}(t) = \alpha(t)$, where $\alpha(t)$ is the learning coefficient ($0 < \alpha(t) < 1$),[5] and decreases monotonically; if on the other hand $i \notin N_c$, then $h_{ci}(t) = 0$. Therefore, Equation (3.1) takes the form of $w(t+1) = w(t)$ for neurons that are not in the neighborhood of the BMU. This basically means that these neurons that are not neighbors of BMU will continue to have the same weight vector in time $t+1$. The radius σ also decreases monotonically in time. Hence, as time passes the number of neighbors that get updated (if any), decreases.

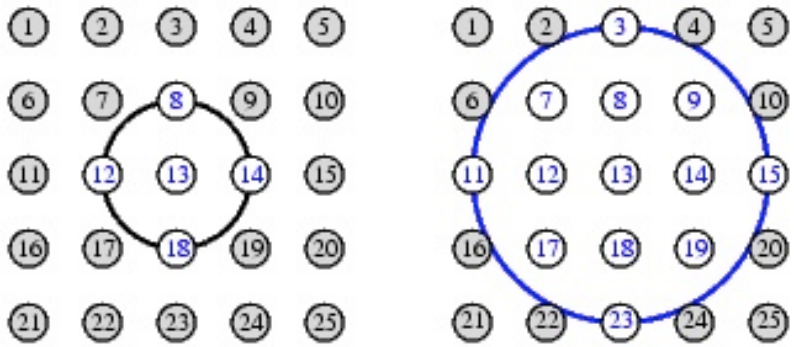


Figure 3.1: Neighborhood examples. In the first case the radius $\sigma = 1$. In the second case, the radius $\sigma = 2$. (Source: MathWorks (2011))

The second type of neighborhood function can be written in terms of the Gaussian function. Equation (3.2) presents the neighborhood function for a neuron i:

$$h_{ci}(t) = \alpha(t) exp(-\frac{||w_c - wi||^2}{2\sigma^2(t)}) \tag{3.2}$$

where w_c and w_i are the weight vectors of neurons c and i, respectively, $||\cdot||$ is the Euclidean distance between the two weight vectors, and σ is the radius of the neighborhood N_c. As earlier, the learning coefficient $\alpha(t)$ and the radius σ decrease monotonically in time. Thus, the smaller these values get, the more h_{ci} decreases, and less and less neighbors of the BMU get updated, as time passes.

Kohonen states that for small SOM networks (i.e. not more than a few hundred neurons) selection of process parameters is not crucial, and thus we can use the simple neighborhood-set definition of $h_{ci}(t)$ (Kohonen, 2001, p. 88).

[5]The learning coefficient is presented in detail in Section 3.4.2.

3.4.2 Learning Coefficient

The learning coefficient $a(t)$ is a monotonically decreasing function of time. During learning, $a(t)$ should start with a high value, which then decreases monotonically in the subsequent time periods. An accurate function is not important for small SOMs (Kohonen, 2001). Thus $a(t)$ could take different forms, such as: linear function, exponential decay function, and a function that is inversely proportional to time t. A linear function can be of the form $\alpha(t) = \alpha_0(1 - \frac{t}{B})$ (Kohonen, 2001), where is α_0 is the initial value of α at time $t = 1$, B is a constant and t is time. The second type of function uses an exponential decay for α, where $\alpha(t) = \alpha_0 e^{-\frac{t}{T_2}}$ (Haykin, 1999), where α_0 is the initial learning rate, T_2 is a time constant, and t is time. Finally, the corresponding formula for the inverse of the time function was suggested by Mulier and Cherkassky (1994) and is equal to $\alpha(t) = \frac{A}{t+B}$, where A and B are again constants, and t is time.

For very large maps, it is crucial to select an optimal $a(t)$ function. Kohonen (2001) also discusses other examples of learning coefficients, which can be used for optimizing the learning coefficient. Nevertheless, as in our experiments we will be using small SOMs, it is outside the scope of this thesis to present and explain, in detail, these coefficients. We thus refer the reader to the relevant books of Kohonen (2001, 1982).

3.4.3 Distance

As we have seen, during the SOM procedure we need to calculate the distance of the input vectors x from the weight vectors w. There are different ways of doing this. The most common distance is the well-known Euclidean distance. For instance, if there is an input vector $x = (x_1, x_2, ..., x_n)$ and a weight vector $w = (w_1, w_2, ..., w_n)$, and both of them have the same dimension $1 \times n$, then the Euclidean distance of x from w is equal to

$$d(x, w) = ||x - w|| = \sqrt{(x_1 - w_1)^2 + (x_2 - w_2)^2 + ... + (x_n - w_n)^2} = \sqrt{\sum_{i=1}^{n}(x_i - w_i)} \quad (3.3)$$

The neuron that is thus the BMU is the one with the smallest Euclidean distance from the input vector. This can also by expressed by Equation (3.4):

$$||x(t) - w_c(t)|| = \min_i\{||x(t) - w_i(t)||\} \quad (3.4)$$

Another distance metric that can be used is the dot-product. Given the same vectors x and w, their dot-product is given by Equation (3.5):

$$x \cdot w = \sum_{i=1}^{n} x_i w_i = x_1 w_1 + x_2 w_2 + ... + x_n w_n \quad (3.5)$$

Here it is the opposite case from the Euclidean distance, and thus the BMU is the neuron with the maximum vector dot-product:

$$x(t) \cdot w_c(t) = \max_i \{x(t) \cdot w_i(t)\} \tag{3.6}$$

However, as the distance and the updating rule of the weight vectors should be mutually compatible with respect to the same metric (Kohonen, 2001, p. 91), Equation 3.1 should also change. We will present this other equation in the next section, where we discuss alternatives to the weight vectors' updating rule.

3.5 Update Rules

So far we have presented the incremental-learning algorithm, which uses Equation 3.1 as the updating rule for the weights of the neurons in SOM. However, there are other rules, too, which can be used for updating the weighting vectors. Two well known rules are the "Dot-Product" and the "Batch Map".

3.5.1 Dot-product

As we mentioned in Section 3.4.3, one of the possible distance functions can be the dot-product. However, if we change the distance rule, we should also change the weight vectors update rule, so that they are compatible with respect to the same metric. As a result, Equation (3.1) is changed to:

$$w_i(t+1) = \begin{cases} \frac{w_i(t)+h_{ci}(t)x(t)}{||w_i(t)+h_{ci}(t)x(t)||} & \text{if } i \in N_c(t) \\ w_i(t) & \text{if } i \notin N_c(t) \end{cases} \tag{3.7}$$

The advantage of this approach is that the weight vectors are normalized at each step.[6] In terms of algorithmic speed, the normalization slows down the algorithm; nevertheless, this can be compensated by the use of the dot-product distance metric, which is quite fast (Kohonen, 2001, p. 91-92). Algorithm 2 presents the pseudo code for the dot-product SOM.

3.5.2 Batch Map

As we have seen, the incremental-learning algorithm assumes that the weight vectors are updated for every input/output training pair. The batch map process differs at this stage. First of all, several input vectors are presented, together, to the network (or possibly all input vectors, if there are not too many). The algorithm then finds the BMU for each input vector. Then, each weight vector is updated by moving to the average position of all of the input vectors for which it is the BMU, or for which it is in the neighborhood of a BMU. Thus every weight vector is updated by the following rule, as shown in Equation (3.8):

[6] Although normalization is not necessary in SOM, it allows the weight vectors to be in the same range, and can thus improve numerical accuracy

Algorithm 2 The "Dot-Product" SOM algorithm

Initialize the SOM's weight vectors, initialize neighborhood function
`/* Learning process */`
while $t < \lambda$ **do**
  for *each input vector* x **do**
    for *each neuron* i *in the SOM* **do**
      Calculate the dot-product between the input vector x and the weight vector w_i of the current neuron i
    end
    Return the weight vector of the neuron with the biggest dot-product
    Update the weight vector of the BMU and of the neurons in the neighborhood of BMU by using the formula:
    $$w_i(t+1) = \begin{cases} \frac{w_i(t)+h_{ci}(t)x(t)}{||w_i(t)+h_{ci}(t)x(t)||} & \text{if } i \in N_c(t) \\ w_i(t) & \text{if } i \notin N_c(t) \end{cases}$$
    Update the neighborhood function h_{ci}
  end
  Increment t by 1
end
`/* Mapping process */`
for *each input vector* x **do**
  for *each cluster* i *in the SOM* **do**
    Calculate the distance between the input vector x and the weight vector w_i of the current cluster i
  end
  Map (place) the input vector x to the cluster with the smallest distance
end

$$w_i(t+1) = \frac{\sum_{j=1}^{n} h_{c(j)i}(t)x_j(t)}{n(h_{c(j)i}(t)x_j(t))} \tag{3.8}$$

where $c(j)$ is the BMU of the sample of input vectors x_j, and n is the number of input vectors mapped in the BMU or are in its neighborhood. The algorithm is essentially the same as Algorithm 1, with the only difference being in the update rule.

The Batch Map produces essentially similar results with the iterative SOM algorithm, but an order of magnitude faster. This is specifically true if we use tools like MATLAB, where matrix operations can be utilized efficiently and thus produce much faster results (Vesanto et al, 1999; Deboeck and Kohonen, 1998). For this reason, it is considered to be an improvement to the other methods.

3.6 Topologies

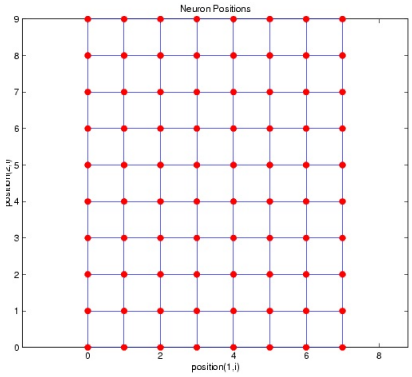


Figure 3.2: An example of a rectangular SOM topology (Source: MathWorks (2011))

The usual arrangement of the neurons in a SOM is in a rectangular or in a hexagonal grid. Results are not affected by the topology chosen. However, the latter topology is usually preferred for visual inspection (Kohonen, 2001). In a hexagonal grid, the distance of all neighboring neurons from the BMU is always the same, whereas this is not the case with the rectangular one. Let us have a look in Figures 3.2 and 3.3, which present a rectangular and a hexagonal topology, respectively. Any selected neuron from the rectangular grid has 8 other neurons around it (apart from the ones at the four corners of the grid), but only 6 of them are at a distance equal to 1. This can also be clearly seen by the left graph of Figure 3.1. On the other hand, a neuron in a hexagonal grid has a distance equal to 1 with all of its 6 neighboring neurons.

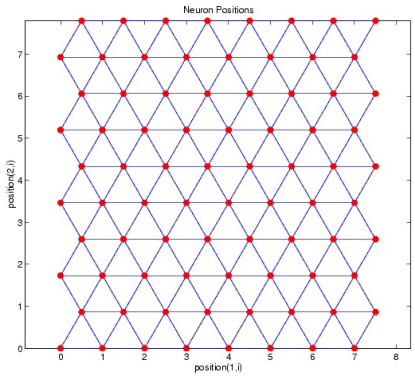


Figure 3.3: An example of a hexagonal SOM topology (Source: MathWorks (2011))

3.7 The MathWorks Neural Network Toolbox

MATLAB is a powerful programming language, which includes a number of toolboxes for different applications. Two well-known toolboxes for SOM are the MathWorks' Neural Network (MathWorks, 2011), and the toolbox from the Laboratory of Computer and Information Science at the Helsinki University of Technology (Vesanto et al, 1999). The former comes pre-installed with MATLAB, and is the toolbox that we use in our experimental work. The purpose of this section is to introduce the reader to this specific toolbox. We specifically focus on presenting the default parameters of the toolbox. If readers are interested in the other functions of this toolbox, we refer them to the toolbox's documentation.

Let us first start with the training algorithm. The default training algorithm is the batch algorithm, which uses the batch map process as its update rule. In MATLAB, this algorithm is called $trainbuwb$, and the weight updating function is denoted by $learnsomb$. The default initialization algorithm in the toolbox is $midpoint$, which was presented in Section 3.3. The default neighborhood function is the same as the one we presented earlier, which uses the set of neurons around the BMU. Its initial value is a radius $\sigma = 3$. The distance function which will be used in our experiments is the Euclidean distance, and is denoted by $dist$. Furthermore, the toolbox has two default values for the learning coefficient, one for during the learning phase and one for during the mapping phase. The former is set to 0.9 and the latter to 0.02. According to MathWorks documentation (MathWorks, 2011), during the learning phase the learning coefficient is "adjusted down" from 0.9 to 0.02.[7] This phase lasts for as many steps as the learning lasts (specified by the user). In addition, during the mapping phase, the coefficient further "decreases slowly" from the value of 0.02. Finally, the default topology used is the hexagonal one, and it is denoted by $hextop$.

3.8 Chapter Summary

In this chapter, we presented the Self-Organizing Map (SOM) method, which is going to be used as part of our experiments in Chapters 7 and 8. We started by presenting a general version of SOM, and then continued by presenting some common variants of the learning parameters. Of course, this was by no means a full description of the method. Our aim was to present the reader with some basic and important information about SOM. If readers are interested in more details about SOM, or more information about other SOM variants, we refer them to the relevant literature mentioned in the early sections of this chapter. Kohonen (2001) also presents a very detailed overview on SOM literature. Another example of good literature of the different SOM variants can be found in Salah Salhi et al (2009).

This concludes Part II, which presented the two main techniques used in the experimental

[7] The MathWorks documentation does not give details on how exactly this adjustment takes place. This fact should not alarm us, however, because in our experiments in Chapters 7 and 8 we will be dealing with very small SOMs (maximum 9 clusters), and as Kohonen says in Kohonen (2001), an accurate learning coefficient function is not important for small SOMs.

work of this thesis. The next part, Part III, presents a literature review on different topics that are related to the works of the thesis. Such topics are financial markets, and more specifically, financial forecasting, market models and market behavior.

Part III

Literature Review

Chapter 4

Financial Forecasting

4.1 General Information

The predictability of financial markets has been attracting the interest of many market professionals, academics and even 'amateur' investors for many years. In this chapter, we attempt to provide a brief review on the forecasting of the financial markets, also known as *financial forecasting*. We can find, in the literature, early attempts to look for temporal patterns in data that could allow people to make predictions, which date back to 1662. At that time John Graunt, a London cloth merchant, published several social and epidemiological comparisons using bills of mortality (Mills, 2002a). If the readers are interested in those early attempts at forecasting, we refer them to Klein (1997), who provides detailed discussion on this topic.

However, not everybody agrees that forecasting can be extended to financial markets. More specifically, the Efficient Market Hypothesis (EMH) tells us that the market cannot be predicted. In the next section (Section 4.2) we present this well-known hypothesis, and also offer a short discussion on it. We then continue by presenting a review on financial forecasting attempts, in Section 4.3, and then present two methods widely used for financial forecasting: fundamental analysis (Section 4.4), and technical analysis (Section 4.5). Furthermore, we discuss some computational intelligence techniques that have commonly been used for financial forecasting, in Section 4.6. Section 4.6 also comments on the limitations of many of these techniques and thus briefly discusses our motivation for the work in Chapter 6. Moreover, Section 4.7 presents EDDIE, which is a genetic programming tool for financial forecasting. EDDIE is the main tool used throughout this thesis for our experimental work. Section 4.7 briefly discusses the limitations of the current EDDIE versions. Finally, Section 4.8 concludes this chapter.

4.2 Efficient Market Hypothesis

Fama (1965) defined the concept of efficient capital markets, which says that if a market is "efficient", prices will fully reflect the available information that relates to the financial asset being

traded. Therefore, it is not possible to make predictions by using information that is available for that market. This is known as the Efficient Market Hypothesis.

The EMH has three forms: weak-form, semi-strong-form, and strong-form. The first one suggests that one cannot make a profit based on past stock prices and returns. The semi-strong-form suggests that we cannot make a profit based on information that is publicly available, because this (new) information will quickly be reflected in the price. Finally, the last form of the EMH suggests that we cannot make a profit even if we have available all information, publicly or not (i.e. insider information).

Thus the question that arises is whether it is futile to attempt to predict the market. There have been a number of works that examine the EMH, both theoretical and empirical. In addition, empirical results have been found both in favor of and against the hypothesis. Tsang (2009) states that early research provided strong evidence in support of the EMH, whereas more recent works tend to show anomalies. Tsang also mentions the example of Scleifer (2000), which poses challenges to the weak-form of the hypothesis. In the following section, we present several examples where forecasting has been applied in financial markets.

4.3 Financial Forecasting Attempts

We referred at the beginning of this chapter to John Graunt, who made an early attempt in forecasting. Since then, forecasting has been extended to different types of markets and a variety of models have been used. Mills (2002a,b) provides an excellent collection of both early and more recent works (up to 2002) on financial forecasting. Although these works could be considered slightly old now, we still believe that Mills' collection is worth mentioning, because the list of works he includes is extensive and informative. Some examples of forecasting that Mills mentions are (Fama, 1972; Pesaran and Timmermann, 1995, 2000). Furthermore, De Gooijer and Hyndman (2006) provide an extensive review on time series forecasting until 2006. In addition, financial forecasting can also be extended to the foreign exchange market (Cheong et al, 2011; McMillan and Speight, 2011), the bond market[1] (Viceira, 2011) and the forward market[2] (Liu and Chou, 2003). Moreover, there have been several works related to volatility forecasting (Poon and Granger, 2005, 2003), and forecasting of extreme events, such as a stock market crash (Garcia Almanza and Tsang, 2007). Finally, over the last few years, we have witnessed new forecasting methods and applications, such as technical analysis (Brock et al, 1992; Taylor and Allen, 1992; Lo et al, 2000) and high frequency forecasting (Matas and Reboredo, 2011).

In this thesis, we are interested in the forecasting of trading rules in the stock market. Traditionally, there have been two forecasting methods that are widely used: the fundamental analysis and the technical analysis. Since, in this work, we are using indicators from the latter, we first

[1]The environment in which trading of debt securities takes place.

[2]Forward contracts take place between individual parties and thus are not frequently traded on exchanges. Therefore, a forward market is a general term used to refer to the informal market in which these contracts take place.

provide a short description of the fundamental analysis, and then continue with a more detailed description of the technical analysis.

4.4 Fundamental Analysis

The aim of fundamental analysis is to examine a company's financial statements and balance sheets in order to predict future trends of their shares. Fundamental analysis thus depends on statistics; it studies past records of assets, earnings, dividends, interest rates, sales, products, management and markets. Fundamental analysts believe that a stock price will return to its real, fundamental value. Thus, if they believe that the current price of a stock is below what they consider to be its fundamental value, they recommend to buy, because they expect the asset's price to rise. On the other hand, when an asset is overvalued, they recommend to sell.

4.5 Technical Analysis

Financial analysts have been using historical data in order to try predict future events for many years. It is believed that traders usually tend to react in the same way to the same types of events (Kahn, 2006). Technical analysts thus believe that there can be patterns in the stock prices and that these patterns repeat themselves (Murphy, 1999).

Technical analysis is a technique which allows someone to evaluate stocks by analysing the statistics generated by the market activity, such as past prices and trading volume (Edwards and Magee, 1992). Technical analysts believe that historical information about stocks and markets can give them an indication of the future performance of that stock or market. Therefore, the goal of technical analysis is to find any kind of patterns in the data (e.g. patterns in price changes or in volume changes), so that the analysts can use this information in order to make a forecast regarding the future movements of this stock or a market. It should also be mentioned that technical analysis supposes that stock prices have trends; thus, the aim is to detect the direction and the strength of the trend. The earlier the trend is detected the more profit a trader can make.

Lastly, it should be mentioned that there is a debate among financial theorists regarding the value of technical analysis. Especially in previous years, technical analysis was not enjoying that much popularity nor acceptance among practitioners or academics. Over the last few years, however, things seem to have changed; technical analysis is being used more and more. Some notable examples are: Brock et al (1992); Taylor and Allen (1992); LeBaron (1998). Some additional and more recent bibliography can also be found in the following works: Lo and Hasanhodzic (2010); Keller (2007); Nordern (2006); Appel (2005); Kirkpatrick and Dahlquist (2006); Irwin and Park (2007). The reader should bear in mind that this list is by no means exhaustive. Besides, it is not the purpose of this thesis to provide neither theoretical nor empirical justification for the use of technical analysis. Our purpose is to use indicators derived from technical analysis, which will then be fed as input for our GP algorithm.

4.5.1 Technical Indicators

There are many indicators derived from technical analysis. This section presents the technical indicators that have been used for the experiments of this thesis. These indicators have also been found to be quite popular among traders (Edwards and Magee, 1992; Murphy, 1999; Pring, 1991; Brock et al, 1992; Martinez-Jaramillo and Tsang, 2009; Garcia Almanza, 2008). Nevertheless, the techniques developed in this work are not limited to such indicators.

We have performed a sort of standardization in the indicators; the reason for this is because when we later on use these indicators with GP, we do not want their results to be in a big range, because this would mean that the GP's search space would be increased. Given a price time series [price $P(t)$ at time $t \geq 0$], and a period of length L, here are the indicators we used, along with their formulas (Equations (4.1) to (4.6)).

Moving Average (MA) Indicator
Nowadays there are many different types of MA. Some examples are the Simple MA, the Cumulative MA, the Weighted MA, the Exponential MA, and the Modified MA. However, the basic one is the Simple MA and this is the one we use in this work.

Thanks to the MA, traders are able to observe any changes in the trend of the prices of a stock. Typically, when a short-term MA (e.g. 12 days) goes above a long-term MA (e.g. 60 days), this indicates upward momentum. On the other hand, when a short-term MA goes below a long-term one, this indicates downward momentum. The reason traders are usually using MAs with different periodicity (usually a short-term and a long term, as above) is because this allows them to detect changes in smaller and bigger trends, respectively.

The formula for MA is given by Equation (4.1):

$$\text{MA(L,t)} = \frac{P(t) - \frac{1}{L}\sum_{i=1}^{L} P(t-i)}{\frac{1}{L}\sum_{i=1}^{L} P(t-i)} \tag{4.1}$$

Filter (FLR) Indicator
This indicator is used to indicate buy or sell actions, depending on whether the price movement goes in the opposite direction by a predefined percentage. For instance, if the price reverses from a downward trend and rises by a specific percentage from the low price that it was previously, then the trader would perform a 'buy' action.

Equation (4.2) presents our interpretation of the FLR indicator:

$$\text{FLR(L,t)} = \frac{P(t) - min\{P(t-1), \ldots, P(t-L)\}}{min\{P(t-1), \ldots, P(t-L)\}} \tag{4.2}$$

Trade Break Out (TBR) Indicator

In order to understand this indicator better, we first need to explain two terms: *support* and *resistance*. Support is the point where the price stops going down any further, whereas resistance is the point where the price does not go up any further. Technical analysts suggest that downward price trends tend to reverse at support points, whereas upward trends tend to reverse at resistance points. However, when these points are breached (breakout), perhaps because of some new information regarding the market, it is likely that the price will continue in the same direction. Hence, traders tend to observe these breakouts and when a stock goes above its point of resistance, they buy; when on the other hand the stock price goes below its point of support, traders sell.

Equation (4.3) shows the formula for this indicator:

$$\text{TBR(L,t)} = \frac{P(t) - max\{P(t-1), \ldots, P(t-L)\}}{max\{P(t-1), \ldots, P(t-L)\}} \tag{4.3}$$

Volatility (Vol) Indicator

A period of an increasing volatility could indicate a reversal in the trend or strong downward trends. This would thus give an indication to a trader that he should be cautious. On the contrary, when there is a period of decreasing volatility, this indicates upward trends and traders should buy.

Vol is equal to:

$$\text{Vol(L,t)} = \frac{\sigma(P(t), \ldots, P(t-L+1))}{\frac{1}{L}\sum_{i=1}^{L} P(t-i)} \tag{4.4}$$

Momentum (Mom) Indicator

The Mom Indicator measures the acceleration or speed at which a stock's price is changing. In order to better understand this indicator, (Appel, 2005, p. 52) gives the following example:

"Consider a golf drive, for example. A well-hit ball leaves the tee quickly, rising and gaining altitude quickly. Momentum is very high. Although it might be difficult to estimate the carry of the drive in its initial rise from the tee, it is often possible to determine, from the initial rate of rise of the ball, that this is a well-hit drive, likely to carry for some considerable distance. Sooner or later, the rate of climb of the ball clearly diminishes and the ball loses momentum. At this time, an estimate of the final distance of the drive can be be more readily made. The important concept involved is that rates of rise diminish before declines actually get under way. The falling rate of change of the drive provides advance warning that the ball is soon going to fall to the ground. In its price movements, the stock market often demonstrates momentum characteristics that are very similar to the momentum characteristics of the golf drive".

Thus, when Mom is positive, this indicates an upward trend. If Mom starts decreasing, this could be an indication that there is going to be a reverse in the previously upwards trend, and hence the traders should be cautious. Finally, when Mom is negative, this indicates a downwards trend.

Equation (4.5) provides the Mom formula:

$$\text{Mom(L,t)} = \frac{P(t) - P(t-L)}{P(t-L)} \tag{4.5}$$

Finally, from Mom we can also calculate its MA, as shown in Equation (4.6).

Momentum Moving Average (MomMA) Indicator

$$\text{MomMA(L,t)} = \frac{1}{L}\sum_{i=1}^{L} Mom(L, t-i) \tag{4.6}$$

4.6 Computational Intelligence techniques for Financial Forecasting

Computational Intelligence focuses on the design and development of algorithms that are capable of producing new knowledge to improve their performance (Kodratoff et al, 1990). There are several CI techniques that have been used for financial forecasting. Among the most popular and successful ones we can find Artificial Neural Networks, Genetic Algorithms, Genetic Programming and Grammatical Evolution. In this section we provide a brief presentation for each one of the above techniques, along with references for some important works on them.

4.6.1 Artificial Neural Networks

Artificial Neural Networks (ANNs) is a model that emulates the biological neural networks. An ANN is characterized by three things, as Fausett (1994) explains: its pattern of connections between the neurons (i.e. its architecture), its method of determining the weights of the connections (i.e. training) and its activation functions. ANNs are a very well-exploited CI technique in financial forecasting. The literature in this area is expansive; some early works on ANN are: Trippi and Turban (1992); Refenes (1994); Azoff (1994); Zhang et al (1998). Some other more recent examples are Chang et al (2009); Hou and Duan (2009); Lam (2004); West et al (2005); Roh (2007). Furthermore, Wong and Selvi (1998) and Shachmurove (2005) offer useful literature surveys on the applications of Neural Networks in different fields of finance. Finally, ANNs have also been used in combination with other techniques, like Evolutionary Algorithms (Kwon and Moon, 2007; Hayward, 2006; Kim, 2006; Versace et al, 2004; Yao and Liu, 1997; Yao and Islam, 2008; Liu and Yao, 2001).

4.6.2 Genetic Algorithms

Genetic Algorithms (GAs) were invented by John H. Holland (Holland, 1975). An informative introductory book to this area is Michalewicz (2002). GAs are search algorithms. They have not been used only for financial forecasting applications, but also for optimization and machine learning problems. Moreover, GAs have been used for the modeling of economic learning in the context of agent-based computational economics and in the context of multi-agent systems in general.

In the past, GAs had been used in several works to perform financial forecasting; some examples are Badawy et al (2005); Bauer (1994); Mani (1996); Allen and Karjalainen (1999); Leigh et al (2002). However, they have been less popular recently due to certain limitations, such as the fixed size structure[3] of the individuals and their representations (Martinez-Jaramillo, 2007). Nonetheless, GAs can be used in conjunction with other techniques like Neural Networks, as was shown earlier.

4.6.3 Genetic Programming

Genetic Programming (GP) is also inspired by natural evolution and its individuals are computer programs, as was shown earlier in Chapter 2. GP has been broadly used over the last years for financial forecasting purposes. One main advantage of GP is its 'transparency', since it is not a black box technique, and thus allows the user to see and understand the trees, i.e. the mechanism that leads to a specific decision. In this way, a user/trader can have a better understanding of the market mechanism. Some recent examples in this area are Bernal-Urbina and Flores-Méndez (2008); Xie et al (2007); Bhattacharyya et al (2002); Dempster et al (2001); Agapitos et al (2010); Garcia Almanza and Tsang (2007); Tsang et al (2004); Abdelmalek et al (2009); Lohpetch and Corne (2009); Wang et al (2010); Wilson and Banzhaf (2010).

For our experiments in this thesis, we have used GP. More specifically, we have used EDDIE, a well-tested financial forecasting tool, which we present in detail in Section 4.7.

4.6.4 Grammatical Evolution

Grammatical Evolution (GE) is a relatively new technique, which can evolve computer programs in any language (Ryan et al, 1998) and is based on the BNF grammar. O'Neill, one of the founders of GE, explains in (O'Neill et al, 2001, p. 346) how GE works.

"Rather than representing the programs as parse trees, as in traditional GP, a linear genome representation is adopted. A genotype-phenotype mapping process is used to generate the output program for each individual in the population. Each individual, a variable length binary string, contains in its codons (groups of 8 bits) the information to select production rules from a BNF

[3] Variable length representations can be used, but it is more complex to implement the crossover operator.

grammar. The BNF is a plug-in component to the genotype-phenotype mapping process, that represents the output language in the form of production rules. It is comprised of a set of non-terminals that can be mapped to elements of the set of terminals, according to the production rules".

Some relevant GE examples for financial forecasting are O'Neill et al (2001); Dempsey et al (2004); Bradeley et al (2010).

4.6.5 Other Computational Intelligence (CI) techniques

In this section we present some further CI techniques that are used for financial forecasting. It would of course be ambitious to claim that we will offer a full account of the research in this area; our goal is just to provide some illustrative examples.

Reinforcement learning is another CI technique used for financial forecasting. Examples of this can be found in Dempster and Romahi (2002); Dempster and Leemans (2006). Support Vector Machines (SVMs) are powerful mechanisms that have been used to perform financial forecasting, like in Yuan and Zou (2009); Sapankevych and Sankar (2009); Cao and Tay (2006); Gestel et al (2001); Huang et al (2004a). In particular, such mechanisms have been extensively used to predict bankruptcy like in Fan and Palaniswami (2000); Gestel et al (2003) or to perform credit rating like in Huang et al (2004b). A detailed literature survey on the topic can be found in Sapankevych and Sankar (2009). Moreover, Learning Classifier Systems (LCSs) is another CI technique that has been used for financial forecasting. Examples can be found in Schulenburg and Ross (2001, 2002). Bayesian Kernel Models have also been used for volatility forecasting (Tino et al, 2005). Finally, Genetic Network Programming and Differential Evolution have also been used for financial forecasting purposes (Chen et al, 2007; Worasucheep, 2008).

4.6.6 Limitations of the current financial forecasting approaches

So far, we have presented different CI techniques which have been applied for financial forecasting purposes. Many of these works, use technical analysis indicators to form their predictors. Such examples are: Bhattacharyya et al (2002); Dempster et al (2001); Becker and Seshadri (2003); Dempster and Jones (2000); Dempster and Leemans (2006); Dempster et al (2001); Dempster and Romahi (2002); Agapitos et al (2010); Cao and Tay (2001). However, a limitation of all of the above works is that the indicators they use have a pre-specified period. For instance, they use '12 days Moving Average'. Nevertheless, nobody can guarantee that the period of '12 days' is the optimal period for an indicator. We thus consider important to work towards this direction and investigate the effects of allowing the evolution to decide on the optimal indicators. In the next section, we present a GP system that was developed at the University of Essex, which also faced the limitation of pre-specified periods. Later, in Chapter 6, we explain how we have extended this tool, by allowing the GP to look for the optimal indicators.

One last thing that should be mentioned, is that the only work known to us that has not used fixed periods for the indicators is Brabazon and O'Neill's (Brabazon and O'Neill, 2004). However,

this work does not offer any discussion on the effects of this choice. This is because this work focused on the advantages of the application of Grammatical Evolution to Foreign Exchange (FX) markets. No discussion was attempted on the benefits of allowing the evolution to look for the optimal periods.

4.7 EDDIE for Financial Forecasting

In this section, we present a GP tool for financial forecasting, called EDDIE (which stands for Evolutionary Dynamic Data Investment Evaluator), which uses indicators from technical analysis in order to make its forecasts. EDDIE is the main tool used for the experiments in Chapter 6 and is also used for parts of the experiments in Chapters 7 and 8.

EDDIE learns and extracts knowledge from a set of data. It was first implemented in horse races (Tsang et al, 1998). It was then implemented in the stock market, in order to forecast price movements and help investors in their decision process (Li and Tsang, 1999b). Later EDDIE was developed into an interactive tool for discovering investment opportunities (Tsang et al, 2004). Li and Tsang (Li and Tsang, 2000; Tsang and Li, 2002) further developed the algorithm and introduced constraints to its fitness function. This version was called FGP-2 (stands for Financial Genetic Programming, a.k.a EDDIE-4), and will be discussed in the next section. The introduction of the constraints did not affect the overall correctness of the program, but it improved the rate of failure (i.e. the algorithm would make less mistaken predictions-see Section 4.7.3 for more details). EDDIE was also used in arbitrage opportunities discovery (Tsang et al, 2005), where it managed to successfully forecast arbitrage opportunities. However, it only managed to pick up a very small proportion of these opportunities; it thus had a quite high rate of opportunities that it was failing to discover. This fact thus limited any commercial potential for that version. Furthermore, EDDIE was also applied to detect scarce opportunities. Garcia Almanza (Garcia Almanza and Tsang, 2007; Garcia Almanza, 2008) invented the Repository Method, which was successfully used for detecting rare opportunities in the market (e.g. crashes). Finally, EDDIE has recently been used in Wang et al (2011, 2010), where new fitness function and new operators were applied; in addition, Wang et al (2010) also used the ensemble method. Results were promising, showing improvement in the performance of the algorithm, but were not always consistent across all datasets tested (Wang et al, 2010).

In this work we are interested in the financial forecasting abilities of EDDIE, and therefore we will focus on those versions of EDDIE. More specifically, we will focus on Jin Li's FGP-2 (Li, 2001), which has been extensively presented in the literature (Tsang et al, 2000; Li and Tsang, 1999a; Tsang et al, 2004). For the rest of this chapter, we will hence provide more information on this algorithm. Later, in Chapter 6 we present how we have extended FGP-2 and the reasons for this.

4.7.1 FGP-2: An Overview

In his framework, Li was interested in predicting price movements. The kind of question he tried to answer was 'will the price increase within the n following days by r%'? Let us now show how FGP-2 attempts to address this question.

First of all, the user feeds FGP-2 with a set of past data; it then uses this data and, through a GP process, it produces Genetic Decision Trees (GDTs), which make recommendations of buy (1) or not-to-buy (0). It then evaluates the performance of the GDTs on a training set, for each generation. The GDT with the highest fitness at the last generation is finally applied to a testing set.

The set of data FGP-2 uses is composed of the daily closing price of a stock, a number of attributes, and signals. Stocks' daily closing prices can be obtained online on websites such as `http://finance.yahoo.com` and also from financial statistics databases like *Datastream*. The attributes are indicators commonly used in technical analysis (Edwards and Magee, 1992); which indicators to use depends on the user and his belief in their relevance to the prediction. Table 4.1 presents the technical indicators that FGP-2 used.

Table 4.1: Technical Indicators used by FGP-2.

Technical Indicators (Abbreviation)	Period
Moving Average (MA)	12 & 50 days
Trade Break Out (TBR)	5 & 50 days
Filter (FLR)	12 & 63 days

The signals are calculated by looking ahead of the closing price for a time horizon of n days, trying to detect if there is an increase of the price by r% (Tsang et al, 2000). Thus, when there is an increase in the price by r% within the next n days, then we denote this by 1 (true), else by 0 (false). Let us for instance assume that n is set to 20 and r to 4%. The GP is thus trying to use some of the indicators of Table 4.1 in order to forecast whether the daily closing price is going to increase by 4% within the following 20 days.

After we feed the data to FGP-2, it creates and evolves a population of GDTs. Figure 4.1 presents the BNF (grammar) of these GDTs for FGP-2. The root of each tree is an If-Then-Else statement. Then the first branch is a Boolean (testing whether a technical indicator is greater than/less than/equal to a value[4]), or a logic operator (and, or, not), which can hold multiple Boolean conditions. The 'Then' and 'Else' branches can be a new GDT, or a decision, to buy or not-to-buy (denoted by 1 and 0).

We would also like to draw the reader's attention to the *Variable* symbol of Figure 4.1; here are the 6 indicators which we mentioned earlier in Table 4.1 that FPG-2 uses. They are pre-specified and should thus be considered as constants of the system. This was considered a limitation of all

[4]This value is a real number, which is optimized through a hill climbing process.

<Tree> ::= If-then-else <Condition> <Tree> <Tree> | Decision
<Condition> ::= <Condition> “And” <Condition> |
<Condition> “Or” <Condition> |
”Not” <Condition> |
Variable <RelationOperation> Threshold
<Variable> ::= MA_12 | MA_50 | TBR_5 | TBR_50 | FLR_12 |
FLR_63
<RelationOperation> ::= “>” | “<” | “=”
Decision is an integer, Positive or Negative implemented
Threshold is a real number

Figure 4.1: The Backus Normal Form of FGP-2

EDDIE versions and thus motivated us to extend the algorithm. More information about how we have extended the algorithm will follow in Chapter 6.

4.7.2 Performance evaluation

Each GDT's performance is evaluated by a fitness function, presented below.

If the prediction of the GDT is positive (1), and also the signal in the training data for this specific entry is also positive (1), then this is classified as True Positive (TP). If the prediction is positive (1), but the signal is negative (0), then this is False Positive (FP). On the other hand, if the prediction is negative (0), and the signal is positive (1), then this is False Negative (FN), and if the prediction of the GDT is negative (0) and the signal is also negative (0), then this is classified as True Negative (TN). These four together give the familiar confusion matrix (Provost and Kohavi, 1998), which is also presented in Table 4.2.

Table 4.2: Confusion Matrix

	Actual Positive	Actual Negative
Positive Prediction	True Positive (TP)	False Positive (FP)
Negative Prediction	False Negative (FN)	True Negative (TN)

As a result, we can use the following 3 metrics:

Rate of Correctness (RC)

$$RC = \frac{TP + TN}{TP + TN + FP + FN} \tag{4.7}$$

Rate of Missing Chances (RMC)

$$RMC = \frac{FN}{FN + TP} \tag{4.8}$$

Rate of Failure (RF)

$$RF = \frac{FP}{FP + TP} \tag{4.9}$$

Li (2001) combined the above metrics and defined the following fitness function:

$$ff = w_1 * RC - w_2 * RMC - w_3 * RF \tag{4.10}$$

where w_1, w_2 and w_3 are the weights for RC, RMC and RF respectively.[5] Li states that these weights are given in order to reflect the preferences of investors. For instance, a conservative investor would want to avoid failure; thus a higher weight for RF should be used. However, Li also states that tuning these parameters does not seem to affect the performance of the GP.

4.7.3 Constraints

As we said at the beginning of the chapter, Li and Tsang introduced constraints to EDDIE, and thus created FGP-2. These constraints were implemented in the fitness function, which allowed FGP-2 to achieve lower RF at the price of a higher RMC. The effectiveness of this constrained fitness function has been discussed in detail in Li (2001); Tsang et al (2005). The constraint is denoted by R, which consists of two elements represented by percentage, given by

$$R = [Cmin, Cmax],$$

where C_{min} is the percentage of minimum positive predictions required, and C_{max} is the percentage of maximum positive predictions required.

Therefore, a constraint of $R = [50, 65]$ would mean that the percentage of positive signals that a GDT predicts[6] should fall into the range of 50-65%. When this happens, then the coefficient of RC (w_1) remains as it is. Otherwise, w_1 it takes the value of zero.

During the evolutionary procedure, we allow three operators: crossover, mutation and reproduction. After reaching the last generation, the best-so-far GDT (in terms of fitness) is applied to the testing data.

[5] Here we should note that $RMC = 1 - Recall$, and $RF = 1 - Precision$, where Recall and Precision are the two well-known metrics for evaluating the correctness of pattern recognition algorithms. There is no doubt that Recall and Precision are more popular in the literature than RMC and RF. However, we chose to continue using Li's RMC and RF for consistency purposes, since we are after all extending his work.

[6] As we have mentioned, each GDT makes recommendations of buy (1) or not-to-buy (0). The former denotes a positive signal and the latter a negative. Thus, within the range of the training period, which is t days, a GDT will have returned a number of positive signals

Algorithm 3 presents the pseudo code that FGP-2 uses for its experiments.

Algorithm 3 Pseudo code for the procedure that FGP-2 and EDDIE in general follows. (Based on (Li, 2001, p. 76))

Procedure EDDIE ( )
Begin
Partition whole data into training data and testing data; */* While training data is employed to train EDDIE to find the best-so-far-rule; the test data is used to determine the performance of predictability of the best-so-far-rule */*
Pop <- InitializePopulation (Pop); */* randomly create a population of GDTs.*/*
Evaluation (Pop); */* calculate fitness of each GDT in Pop */*
Repeat
Pop <- Reproduction (Pop) + Crossover (Pop); */*new population is created after genetic operators of reproduction which reproduces M*Pr individuals and crossover which creates M*(1-Pr) individuals. Pr denotes the reproduction probability and M is the population size */*
Pop <- Mutation (Pop); */*Apply mutation to population */*
Evaluation (Pop); */* Calculate the fitness of each GDT in Pop */*
Until (TerminationCondition( )) */* determine if we have reached the last generation */*
Apply the best-so-far rule to the test data;
End

4.8 Chapter Summary

This concludes this chapter, which offered a review on financial forecasting. We started by presenting a brief history for financial forecasting and then continued by presenting its two main constituents, fundamental and technical analysis. We then presented a number of technical indicators, which are going to be used as part of our GP tool in the later chapters of this thesis. Then, we presented some literature examples of different computational intelligence techniques that have been used for financial forecasting and also explained why the current approach of pre-specified periods of the technical indicators is considered to have limitations. Finally, we presented EDDIE, a genetic programming tool which can be used for forecasting purposes. We also discussed previous applications of the tool in the literature. We finally explained in detail how the algorithm works and how it can be evaluated. The next chapter focuses on financial agent-based models and dynamics of financial markets. These models are closely related to the Market Fraction Hypothesis and the Dinosaur Hypothesis, which constitute the second and third research chapter of this thesis (Chapters 7 and 8).

Chapter 5

Agent-Based Financial Models

This chapter focuses on agent-based financial models. These models are of interest to us for two reasons: first of all, observations from these models about fraction[1] and behavior dynamics have led to the formulation of the Market Fraction Hypothesis (MFH) and the Dinosaur Hypothesis (DH), which are the two hypotheses we test in this thesis, in Chapters 7 and 8. Secondly, for the purposes of testing these hypotheses we use a new agent-based financial model, which acts as an extension to the models that already exist in the literature. We thus consider it important to start by offering a brief review on the existing types of agent-based financial models.

Thus, the rest of this chapter is organized as follows. Section 5.1 presents the different designs of agent-based financial models and gives some information for each design. Then, Section 5.2 discusses some of the limitations of the above designs and thus explains our motivation for suggesting an extension to these models. Finally, Sections 5.3 and 5.4 discuss the observations made by the agent-based financial models, which as we have already mentioned, led to the formulation of the MFH and the DH.

5.1 Agent-based Financial Models

Agent-based financial models are models of financial markets, where artificial agents can trade with each other. These models simulate the simultaneous operations and interactions of the different agents that exist in the market, with the goal of re-creating or predicting the appearance of complex phenomena (e.g., stylized facts of financial time series, such as volatility clustering, fat tails, non-Guassianity (Cont, 2001)). Building such models can give valuable information about different aspects of market dynamics, such as behavior dynamics (Chan et al, 1999). In this section we present the different designs of financial agent-based models, along with some examples.

According to Chen et al (2012), there are two basic designs (models) of financial agents: the N-type design, and the autonomous-agent design. A typical example of the former is the

[1] The term 'fraction dynamics' refers to the fractions of the different trading strategy types that can exist in a market. More details about this are provided later in Section 5.3.

fundamentalist-chartist model and of the latter the Santa Fe Institute Artificial Stock Market (SFI-ASM). The rest of this section presents these two designs.

5.1.1 N-type Designs

5.1.1.1 Two- and Three-Type Designs

In this type of design, agents have beliefs regarding the price of a stock in the next period. In the two-type design, there are two types of agent beliefs. Consequently, there are two types of 'fixed' trading strategies. Each agent can choose between these two types. These two types are usually the fundamentalists and the technical traders.[2]

The three-type design is an extension of the two-type one, where there are three types of agents. One way to implement this design is to have two types of chartists, the momentum traders and the contrarian traders (Sansone and Garofalo, 2007). The former is the kind of agents we described above as 'chartists'. The latter, the contrarian traders, extrapolate past movements of the price into the future, the opposite way that the trend goes. This happens because contrarians believe that the price trend will finish soon and will start to reverse.

Finally, we should mention that several extensions of the above designs exist, by enriching their behavioral rules. For instance, a typical way to do this is by adding a memory factor to these rules. More information about this can be found in Chen et al (2012).

Adaptive behavior

In the original fundamentalist-chartist design, there is no learning. This means that once an agent chooses to be a fundamentalist (or chartist), he will never change this believe and always continue being a fundamentalist (or chartist). A way to make these models more realistic is to add an adaptive behavior, where the agents can learn from their previous experiences and then update their beliefs. Thus, an agent that started as a fundamentalist is now able to update its beliefs at every period. As a result, the proportion of the different strategy types that exist in the market (e.g., fundamentalists and chartists), which is called the *Market Fraction*, can change at every period. Therefore, agents can switch between the strategy types, depending on which one they consider to be more promising.

A good example of adaptive behavior can be found in Brock and Hommes (1998), where Brock and Hommes use 2-, 3-, and 4-type models. Other adaptive behavior examples include Kirman's ANT Model (Kirman, 1991, 1993) and Lux's Interactive Agent Hypothesis Model (Lux, 1995, 1997, 1998).

[2]Other equivalent names for technical traders are chartists, trend-followers and noisy traders. For a reminder on what the fundamental and the chartist strategies are, see Chapter 4.

5.1.1.2 Many-Type Designs

So far we have seen designs with several 'fixed' types. In this section we present some examples of we have N-type designs, where $N > 3$.

Adaptive Belief Systems
A useful example of a many-type design is the Adaptive Belief System (ABS) of Brock and Hommes (Brock and Hommes, 1998, 1997). This system can be considered as an extension of the two- and three-type designs we have seen. The number of strategies used is between 1 and N, and these are known and fixed as before. This means that agents can choose from a finite and fixed number of beliefs.

Large Type Limit and Continuous Belief Systems
Other many-type designs include the Large Type Limit (LTL) (Brock et al, 2005) and the Continuous Belief System (CBS) (Dicks and Van der Weide, 2005). In these systems, the number N of strategies is not finite, but infinite, i.e., $N \to \infty$. Both of these systems are based on an idea called *distribution of beliefs*, where there is a belief space from which the observed beliefs are sampled.

5.1.2 Autonomous Agent Designs

So far, we have talked about the N-type designs, where the strategies are pre-specified and fixed by the model designer. Thus, the agents are restricted to using these specific strategies and cannot come up with any new ones. Although the N-type design had been characterized as a major class of agent-based financial models, it has also been agreed that it severely restricts the degree of autonomy available for financial agents (Chen et al, 2012).

The above issue of autonomy was addressed by the autonomous agent designs, where we can have artificial agents who are autonomous and thus have the ability to discover new strategies, which have never been used before. An example of this is the well-known SFI-ASM (Palmer et al, 1994; Arthur et al, 1997), where a Genetic Algorithm (GA) was used. Thus, there is not a fixed number of strategies; on the contrary, each artificial agent can have a different forecasting behavior which is "customized" by a GA. SFI-ASM is of course not the only application of GAs in artificial stock markets. Another example is AGEDASI TOF[3] (Izumi and Okatsu, 1996; Izumi and Ueda, 1999). If the reader is interested in these topics, an informative literature review can be found in Chen et al (2009).[4]

[3] It stands for A GEnetic-algorithmic Double Auction SImulation in TOkyo Foreign exchange market

[4] It should also be said that apart from GA, other population-based learning models have been used, such as GP. We refer the readers to Chen et al (2012) for more details.

5.2 Limitations of the Agent-Based Financial Models

In the previous section, we described two main agent-based financial designs: the N-type and the SFI-like (autonomous agents design). The former design consists of N pre-specified strategy types, and the agents have to choose among these N types. Consider the fundamentalist-chartist model. Agents are presented with these two strategy types at different time periods, and have to choose between these two types. An advantage of this design is that it allows us to observe the changes in the market fraction[5] dynamics of the above strategy types. However, as was shown, a disadvantage of this type of model is that the agents always need to choose from the given N strategy types. In addition, another limitation of this type of model is heterogeneity. Agents that belong in the same trading strategy type follow *exactly the same behavioral rule*. Nevertheless, in the real world, the behavior of each trader is expected to be heterogeneous (Chen et al, 2012); even if some traders are clustered into a strategy type, it does not mean that they behave in exactly the same way.

The issue of heterogeneity is addressed by the SFI-like models (autonomous agent designs). This type of model allows for the creation of autonomous and heterogeneous agents, as was shown above. Nonetheless, even under the autonomous agent models, agents are presented with a pre-specified number of trading strategy types (Chen et al, 2012). To the best of our knowledge, there is no model that uses autonomous agents that are not restricted to predefined, fixed strategy types.

In this thesis, and more specifically in Chapter 7, we present such an agent-based financial model. It first allows the creation of novel, autonomous and heterogeneous agents by the use of GP. The reason for using GP is that the market is considered to be undergoing an evolutionary process; this is inspired by Andrew Lo's Adaptive Market Hypothesis (AMH) (Lo, 2004, 2005), where Lo argued that the principles of evolution (i.e., competition, adaptation, and natural selection) can be applied to financial agents' expectations, i.e., agents' forecasting rules. Thus, agents can be considered to be organisms that learn and try to survive.

After creating and evolving novel agents, we then cluster them into strategy types via SOM. The trading strategy types are thus not pre-specified, but depend upon the strategies of the agents. In this way, we are able to reconstruct the microscopic level of markets (SFI-like designs), where financial agents are created, and connect it to the mesoscopic level (N-type designs), where agents are clustered into N strategy types. Figure 5.1 illustrates this process. More details about this model follow in Chapter 7. Lastly, as we have already mentioned, this model will facilitate the experiments of Chapters 7 and 8, which test the MFH, and the DH, respectively. A brief review on these two hypotheses is presented below.

[5] A reminder that 'market fraction' refers to the proportion of different types of trading strategies in a financial market.

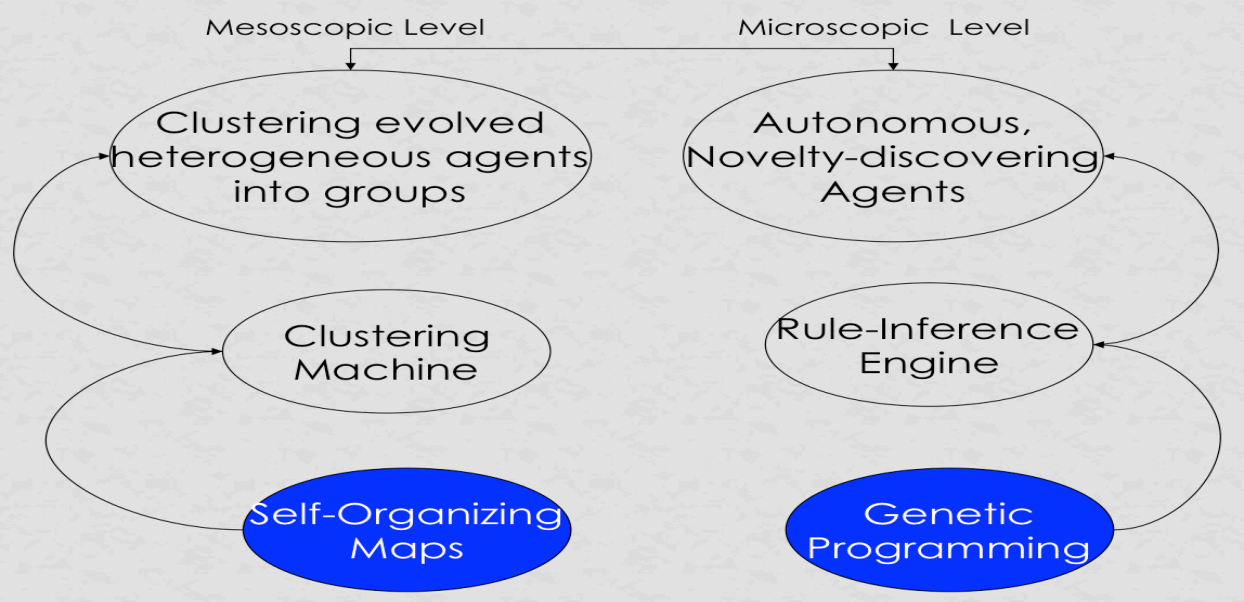


Figure 5.1: Connecting the microscopic to the mesoscopic level of financial markets.

5.3 The Market Fraction Hypothesis

As we mentioned earlier, in the N-type designs, the proportion of different types of trading strategies in a market is called the Market Fraction. A common observation in many of the above models is that the market fraction of the strategies that exist in a market keeps swinging. In other words, if for instance we have two types of agents in the market (e.g., fundamentalists and chartists), it has been found that the fraction of these two strategies keeps swinging over time. If, for example at a time t 90% of the market participants adopt the fundamental strategy and 10% of them adopt the chartist strategy, these fractions change continuously over time; therefore, in a future time period, we could observe that the fundamentalists occupy only 10% of the agents, and the chartists the other 90%. This observation is very interesting, because it gives us valuable information about market fraction dynamics, by suggesting that popularity can be interchanged among all available trading strategy types. More importantly, the above implies that there cannot be a 'winner' type of trading strategy and thus questions the necessity of attempting to forecast trading rules.

This swinging feature has been observed in many N-type financial models (Brock and Hommes, 1998; Amilon, 2008; Lux, 1995, 1997, 1998; Boswijk et al, 2007; Kirman, 1991, 1993). In addition, it has also been observed in artificial agent-based models (Winker and Gilli, 2001; Gilli and Winker, 2003). For a more detailed description of agent-based financial models that have this swinging property, the reader is referred to Chen et al (2012).

Based on these observations about the swinging of market fraction, Chen (Chen, 2008; Chen et al, 2012) suggested a new hypothesis, called the Market Fraction Hypothesis (MFH). The MFH states that there is a constant swinging among the fractions of the types of trading strategies that

exist in a market. However, although the term 'MFH' was introduced and used by Chen, it was never formalized as a hypothesis. This thus motivates us to formalize the MFH, by presenting its main constituents, and also suggesting a method to test it. This takes place in Chapter 7. In addition, motivated by the fact that the majority of the observations about the swinging of the market fraction have so far only taken place under artificial market frameworks, we test the MFH under *empirical* datasets. More details about the MFH tests will follow in the relevant chapter, Chapter 7.

5.4 The Dinosaur Hypothesis

Let us now move our focus to the autonomous agent-based financial models. An observation that has been made in some of these models (Arthur, 1992; Chen and Yeh, 2001) is regarding the markets' behavior, which is said to be non-stationary. In other words, the market behavior constantly changes over time, and thus the same trading strategy that used to perform well at a certain point in time, will not continue performing well in the future.

Brian Arthur, a pioneer of the Santa Fe Institute Artificial Stock Market (SFI-ASM), built an artificial stock market, where he investigated the agents' level of adaption to their market environment (Arthur, 1992). He found that agents that did not co-evolve (i.e., adapt), with the changes in the market, became obsolete. This led Arthur to conclude that the market behavior changed constantly. In addition, Arthur used 'dinosaurs' as a metaphor to describe this constantly-changing property. More specifically, Arthur stated:

> *"We find no evidence that the market behavior ever settles down; the population of predictors*[6] *continually co-evolves. One way to test this is to take agents out of the system and inject them in again later on. If market behavior is stationary they should be able to do as well in the future as they are doing today. But we find that when we "freeze" a successful agent's predictors early on and inject the agent into the system much later, the formerly successful agent is now a dinosaur. His predictions are unadapted and perform poorly. The system has changed"* (Ibid, p.24).

This observation is very interesting for two reasons. Firstly, it informs us that the behavior of a market does not cycle or repeat. On the contrary, it keeps changing. In addition, Arthur informs us that the trading strategies that exist in this market need to continue adapting to the market conditions in order to survive. If they do not co-evolve with the market, then their performance is poor and they can thus be considered as dinosaurs. This also indicates that *any successful trading strategy can only live for a finite amount of time*.

Chen and Yeh (2001) also tested for the existence of this phenomenon in their artificial stock

[6] According to Arthur (1992), an agent's basic problem is to profit in the next period, and the only way to do that is by predicting the direction of the market. Thus, predictors are models that map the patterns which are formed by the current price set into a forecast for the next period's price. In other words, agents in the market use different predictors to forecast future price movements. In this thesis, we refer to predictors as 'trading strategies'.

market. Their results verified Arthur's observations. In addition, Chen and Yeh found that *a dinosaur's performance decreases monotonically*.

To the best of our knowledge, the above observations have only been made by Arthur, Chen and Yeh in 1992 and 2001, respectively. Later, in 2008, Chen (2008) suggested a new hypothesis, called the Dinosaur Hypothesis (DH), based on these observations. The DH states that the market behavior never settles down and that the population of trading strategies continually co-evolves with the market. However, this hypothesis has also never been formalized. Chapter 8 thus formalizes the hypothesis, by presenting its main constituents. Furthermore, motivated by the DH observations, *we are interested in examining the behavior of financial markets in detail*, by using our suggested agent-based financial model that we briefly mentioned earlier in Section 5.2. We thus also suggest a testing methodology and again run tests under *empirical* datasets, as in the MFH framework, since both Arthur's and Chen and Yeh's observations were made under artificial stock market frameworks.

5.5 Chapter Summary

This chapter focused on agent-based financial models. We started in Section 5.1 by presenting different agent-based financial market models, where we identified some of their limitations. These limitations were discussed in Section 5.2, where we also explained that these limitations have motivated us to create a new agent-based financial model, which we will present in detail later, in Chapter 7. Finally, observations from several of the above financial models led to the formulation of the MFH and the DH, which were presented in Sections 5.3 and 5.4.

This concludes the literature review section of the thesis. The next part, Part IV, presents the contributions of this thesis. The first chapter, Chapter 6, presents the work that has been done on EDDIE, and investment opportunities forecasting.

Part IV

Thesis Contributions

Chapter 6

Extending the grammar of a GP-based tool

6.1 Introduction

Earlier in Chapter 4 we presented EDDIE and more specifically EDDIE 4, also known as FGP-2. We also explained that EDDIE, as well as other similar GP financial forecasting tools, use indicators from technical analysis in order to form their Genetic Decision Trees (GDTs). These indicators use pre-specified lengths, e.g. 12 days Moving Average (MA). It is thus left up to the user of the forecasting tool and his expertise, to choose the appropriate period for these indicators. This basically means that two different users could use very different periods and as a result get very different outcomes. *This is the source of our motivation*. We are interested in eliminating this drawback. Why should somebody choose 12 days MA and not 14 or 15? We believe that *it should be left to the evolutionary procedure to make these decisions*. In order to do that we have implemented a new version, EDDIE 8, which allows the GP to search in the search space of the periods of the technical indicators. The novelty of this algorithm is in its rich, extended grammar. Instead of using a fixed number of pre-specified indicators from technical analysis, like the previous versions do, EDDIE 8 allows the GP to search in the space of the technical indicators and use the ones that it considers to be optimal. Thanks to its extended grammar, EDDIE 8 is considered to be an improvement, because it has the potential, through the learning process, to discover better solutions that its predecessors cannot.

In order to present the value of EDDIE 8 we compare it with EDDIE 7, which is a re-implementation of Jin Li's FGP-2 (Li and Tsang, 1999a; Li, 2001), with the addition of some indicators that Martinez-Jaramillo (2007) found helpful and used in his own version of EDDIE.[1]

Therefore, the main contributions of this chapter can be summarized as follows: (i) Extending EDDIE's grammar by allowing the GP to search in the space of technical indicators, and thus creating EDDIE 8, and (ii) Comparing EDDIE 8 with its predecessor, EDDIE 7, and reporting the results from this comparison under both empirical and artificial markets.

[1]Martinez-Jaramillo was also a researcher at the University of Essex and also used EDDIE as part of his work. This is why we have taken into account his conclusions about the usefulness of some specific technical indicators.

The rest of this chapter is organized as follows: Section 6.2 presents the two versions discussed in this chapter, EDDIE 7 and EDDIE 8. Section 6.3 presents the experimental parameters for our tests, and Section 6.4 shows the results of the comparison of the two versions, on 10 different empirical datasets. However, because these results are not conclusive, we also test the performance of the two versions under artificial datasets, where we can create our own patterns in the data and avoid any noise that an empirical dataset might sometimes have. This is done in Section 6.5. We then extend the conclusions drawn from the artificial datasets experiments to the empirical ones, in Section 6.6. Finally, Section 6.7 concludes this chapter and discusses future work.

6.2 EDDIE 7 vs EDDIE 8

6.2.1 EDDIE 7

As already mentioned, EDDIE 7 is our re-implementation of Jin Li's FGP-2 (Li and Tsang, 1999a; Li, 2001), including some additional indicators that Martinez-Jaramillo (Martinez-Jaramillo, 2007) used in his version of EDDIE. The indicators that EDDIE 7 uses are presented in Table 6.1[2]. For simplicity, each indicator uses the same two periods, 12 and 50 days. The reason for using these specific period lengths is that we want to take into account a short-term and a long term period.

Table 6.1: Technical Indicators used by EDDIE 7. Each indicator uses 2 different periods, 12 and 50, in order to take into account a short-term and a long-term period.

Technical Indicators (Abbreviation)	Period
Moving Average (MA)	12 & 50 days
Trade Break Out (TBR)	12 & 50 days
Filter (FLR)	12 & 50 days
Volatility (Vol)	12 & 50 days
Momentum (Mom)	12 & 50 days
Momentum Moving Average (MomMA)	12 & 50 days

Figure 6.1 presents the Backus Naur Form (BNF), which is basically very similar to the one of FGP-2, presented in Figure 4.1. The only difference is, of course, in the use of the additional technical indicators. Once again, we would like to draw the reader's attention to the *'Variable'* symbol, which takes as input the pre-specified indicators, which act as constants of the system. In addition, Figure 6.2 presents a sample GDT produced by EDDIE 7 based on this BNF grammar.

[2]These indicators have been used because they have been proved to be quite useful in developing investment opportunity decision rules for forecasting rises and drops of the price in previous works like Allen and Karjalainen (1999); Austin et al (2004); Martinez-Jaramillo (2007). Of course, there is no reason why not use other information like fundamentals or limit order book information. However, the aim of this work is not to find the ultimate indicators for financial forecasting.

It should be noted here that this tree is a simplified version, which has smaller depth. As a typical GDT could have a depth up to 8, it could be quite hard to visualize it.

<Tree> ::= If-then-else <Condition> <Tree> <Tree> | Decision
<Condition> ::= <Condition> “And” <Condition> |
<Condition> “Or” <Condition> |
“Not” <Condition> |
Variable <RelationOperation> Threshold
<Variable> ::= MA_12 | MA_50 | TBR_12 | TBR_50 | FLR_12 |
FLR_50 | Vol_12 | Vol_50 | Mom_12 | Mom_50 |
MomMA_12 | MomMA_50
<RelationOperation> ::= “>” | “<” | “=”
Decision is an integer, Positive or Negative implemented
Threshold is a real number

Figure 6.1: The Backus Naur Form of the EDDIE 7

As we can see, the If-Then-Else statement starts with the Boolean ‘less than’ method: if the 12 days Moving Average Indicator is smaller than the threshold 6.4, then the user is advised to-buy, which is indicated by ‘1’ in the second branch of the GDT. Otherwise, if the condition under the first branch does not hold, then the GDT returns another If-Then-Else statement, at the third branch of the GDT. This statement, depending on whether its condition is true or false, advises to not-to-buy or buy, respectively.

Figure 6.2: Sample GDT produced by EDDIE 7

This concludes the short presentation of EDDIE 7. We should say at this point that EDDIE 7 will be acting as the benchmark of our experiments. Next, Section 6.2.2 presents how we have extended EDDIE 7, by allowing the GP to search in the space of the technical indicators.

6.2.2 EDDIE 8

Let us consider a function $[y = f(x)]$, where the input x is the indicators and the output y is the prediction made by EDDIE 7. The function f is unknown to the user and is the GDTs that EDDIE 7 generates, in order to make its prediction. As we said earlier, EDDIE 7 uses a number of indicators, with different pre-specified periods. This therefore means that the input x consists of constants. On the other hand, EDDIE 8 uses another function $y = f(g(z))$, where $x = g(z)$; in other words, g is a function that generates indicators and periods for EDDIE to use. EDDIE 8 is not only searching in the space of GDTs, but also in the space of indicators. It can thus return GDTs that are using any period within a range that is defined by the user.

<Tree> ::= If-then-else <Condition> <Tree> <Tree> | Decision
<Condition> ::= <Condition> “And” <Condition> |
<Condition> “Or” <Condition> |
“Not” <Condition> |
VarConstructor <RelationOperation> Threshold
<VarConstructor> ::= MA period | TBR period | FLR period | Vol period |
Mom period | MomMA period
<RelationOperation> ::= “>” | “<” | “=”
Terminals:
MA, TBR, FLR, Vol, Mom, MomMA are function symbols
Period is an integer within a parameterised range, [MinP, MaxP]
Decision is an integer, Positive or Negative implemented
Threshold is a real number

Figure 6.3: The Backus Naur Form of EDDIE 8

As we can see from the new syntax at Figure 6.3, the *Variable* symbol has been replaced by the *VarConstructor* function, which takes two children. The first one is the indicator, and the second one is the Period. Period is an integer within the parameterized range [MinP, MaxP] that the user specifies. The function $VarConstructor$ could be considered as an Automatically Defined Function (ADF) (Koza, 1994), which always takes two arguments (indicator and period). Since these two arguments can take many different values, the number of the ADFs in the system can be quite big; for instance, under a choice of 6 indicators and 64 periods (from 2 to 65 days), the total number of ADFs is $6 \times 64 = 384$.

As a result, EDDIE 8 can return decision trees with indicators like 15 days Moving Average, 17 days Volatility, and so on. The period is not an issue anymore, and it is up to EDDIE 8, and as a

consequence up to the GP and the evolutionary process, to decide which lengths are more valuable to the prediction.

The immediate consequence of this is that now EDDIE 8 is not restricted only to the 12 indicators that EDDIE 7 uses (*which are still part of EDDIE 8's search space*); on the contrary, it now has many more options available, thanks to this new grammar. As an example we again present the same simplified GDT that we presented earlier in Figure 6.2, after having reconstructed it under EDDIE 8's grammar; this "reconstructed" GDT is presented in Figure 6.4. If we look closely at the figure, we will notice that the indicators are no longer pre-specified. Instead, there is the "VarConstructor" function, which as we have mentioned takes two children, the Indicator and the Period length. The 12 and 50 days Period lengths are now in a separate branch, and are thus subject to GP operators, such as crossover and mutation.

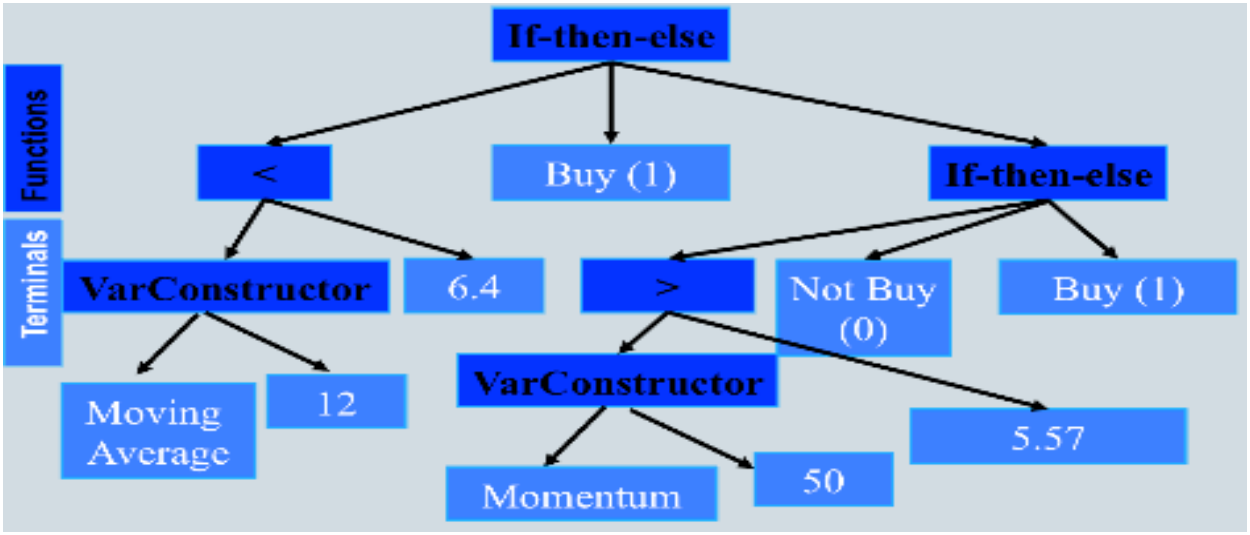


Figure 6.4: Sample GDT produced by EDDIE 8

Now that we have presented and explained the differences of the two versions, we will proceed by presenting our experiments and the results from the comparison between EDDIE 7 and EDDIE 8.

6.3 Experimental Parameters

As we have already said, the data we feed to EDDIE consist of daily closing prices. These closing prices are from 10 arbitrary stocks from the FTSE 100. These stocks are: British American Tobacco (BAT), British Petroleum (BP), Cadbury, Carnival, Hammerson, Imperial Tobacco, Next, Schroders, Tesco, and Unilever. The training period is 1000 days and the testing period 300. Table 6.2 presents the training and testing dates for the above datasets. For simplicity, we chose periods

where there was a bull market. In addition, the GP parameters are presented at Table 6.3. For statistical purposes, we run the GP 50 times for each of EDDIE 7 and EDDIE 8.

Table 6.2: Training and Testing dates for the 10 stocks.

Stock	Training Period	Testing Period
BAT	20/05/2003 - 19/02/2007	20/03/2007 - 13/05/2008
BP	26/06/1992 - 25/04/1996	26/04/1996 - 19/06/1997
Cadbury	03/06/1992 - 02/04/1996	03/04/1996 - 27/05/1997
Carnival	02/05/2001 - 01/03/2005	02/03/2005 - 25/04/2006
Hammerson	30/11/2000 - 29/09/2009	30/09/2004 - 23/11/2005
Imperial Tobacco	19/06/1997 - 18/04/2001	19/04/2001 - 12/06/2002
Next	05/10/1990 - 04/08/1994	05/08/1994 - 28/09/1995
Schroders	13/06/2002 - 12/04/2006	13/04/2006 - 06/06/2007
Tesco	01/08/1996 - 31/05/2000	01/06/2000 - 25/07/2001
Unilever	19/10/1989 - 18/08/1993	19/08/1993 - 12/10/1994

Table 6.3: GP Parameters.

GP Parameters	
Max Initial Depth	6
Max Depth	8
Generations	50
Population size	500
Tournament size	6
Reproduction probability	0.1
Crossover probability	0.9
Mutation probability	0.01

Thus, the process is as follows: we create a population of 500 GDTs, which are evolved for 50 generations, over a training period of 1000 days. At the last generation, the best performing GDT in terms of fitness is saved and applied to the testing period. As we have already said, this procedure is done for 50 individual runs.

In addition, we should emphasize that we want the datasets to have a satisfactory number of actual positive signals. By this we mean that we are neither interested in datasets with a very low number of actual positive signals, nor in datasets with an extremely high one. Such cases would be categorized as 'chance discovery' (Garcia Almanza and Tsang, 2007), where people are interested in predicting rare events, such as a stock market crash. Clearly this is not the case in our current work where we use EDDIE for investment opportunities forecasting. We are thus interested in datasets that have opportunities of around 50-70% (i.e. 50-70% of actual positive

signals). Therefore, we need to change the values of r and n accordingly,[3] so that we can obtain the above percentage from our data. For our experiments, the value of n is set to 20 days. The value of r varies, depending on the dataset. This is because one dataset might reach a percentage of 50-70% with $r = 4\%$, whereas another one might need a higher or lower r value. Accordingly, we need to calibrate the value of the R constraint, so that EDDIE produces GDTs that forecast positive signals in a range which includes the percentage of the *actual* positive signals of the dataset we are experimenting with. We thus set R to take values in the range of $[-5\%, +5\%]$ of the number of positive signals that the dataset has. For instance, if under $r = 4\%$ and $n = 20$ days, a dataset has 60% of actual positive signals, then R would be set to [55,65].

Table 6.4: EDDIE Parameters

EDDIE Parameters	Value
n	20
w_1	0.6
w_2	0.1
w_3	0.3
Period (EDDIE 8)	[2, 65]

Furthermore, Table 6.4 presents the parameters used by EDDIE. As we have already mentioned, n is set to 20 days. The three weights w_1, w_2, w_3 of the fitness function are set to 0.6, 0.1 and 0.3, respectively, and are given in this way in order to reflect the importance of each performance measure (i.e. RC, RMC, RF) to our predictions. As we can see, we chose to include strategies that mainly focus on correctness and reduced failure. The last entry of Table 6.4 refers to the period length, which as we know is a parameter of EDDIE 8. It is set to the range of 2 to 65 days, which means that the technical indicators of EDDIE 8 can have any period length within this range.

We should also mention that we have chosen to use the same fitness function that Li introduced in his work (Li, 2001). As we have already explained in Chapter 5, the fitness function contains three metrics: RC, RMC, RF. There is no doubt that other metrics could be used of course, such as 'actual financial returns after transaction costs'. Nevertheless, since we are extending Li's work, we chose for consistency purposes to use the same fitness function. In addition, later tests in the chapter provide as a reference additional performance metrics, such as Annualized Average Rate of Return, and Rate of Positive Returns, which are more relevant to financial applications.

[3] A reminder that EDDIE attempts to answer the following question: "Will the price of a stock rise by $r\%$ within the next n days"?

6.4 Test Results

This section presents the experimental results after having tested the 10 datasets under EDDIE 7 and EDDIE 8. We start by observing how EDDIE 8 affects the fitness of the population during the training period. We are interested in seeing whether the extended grammar is giving EDDIE 8 an advantage, and if this is the case, how fast this happens during the evolutionary procedure. We then continue by presenting a summary statistics comparison between the two versions, under the data of the testing period. At this point we should mention that all fitness results have been normalized to a scale of [0,1]. The other measures (RC, RMC, RF) were already in this scale and thus no further normalization took place.

6.4.1 Training performance comparison

In this section, we compare the training fitness of the two algorithms. As we have said, we are interested in examining the behaviour of the GP, now that it searches in a much larger search space. Does it find well-performing solutions from the beginning of the evolutionary procedure, because it now has more options to look into? Or does it start with low performance due to these many options and later manages to focus on the promising ones? These are just two examples of behavioral questions that we could be asking.

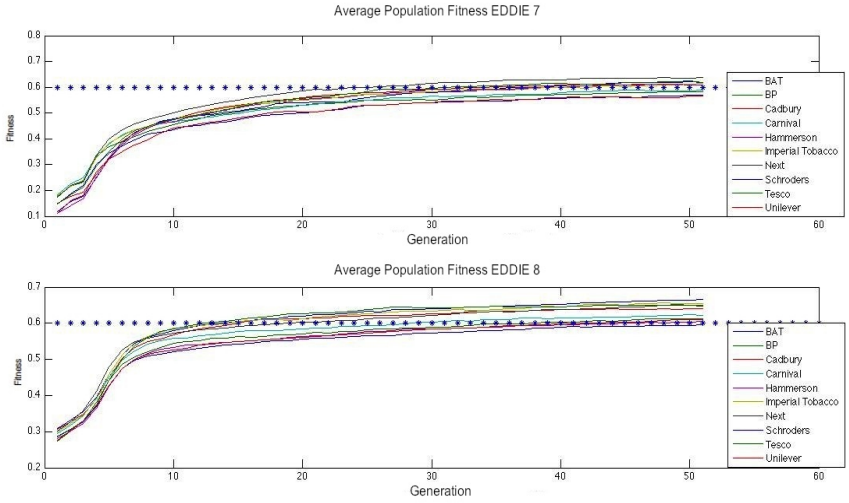


Figure 6.5: Average of the average fitness of the population of the GDTs for EDDIE 7 and EDDIE 8. This means that we first obtain the average fitness of the whole population, per generation. Then we find the average of this number over the 50 runs.

We conduct our analysis in two different parts. Firstly, we compare the training fitness in terms of the whole population. To do that, we calculate the average fitness for the whole population of GDTs; this process is done for each generation. Let us call this average Avg_{Fit}. Thus, we can

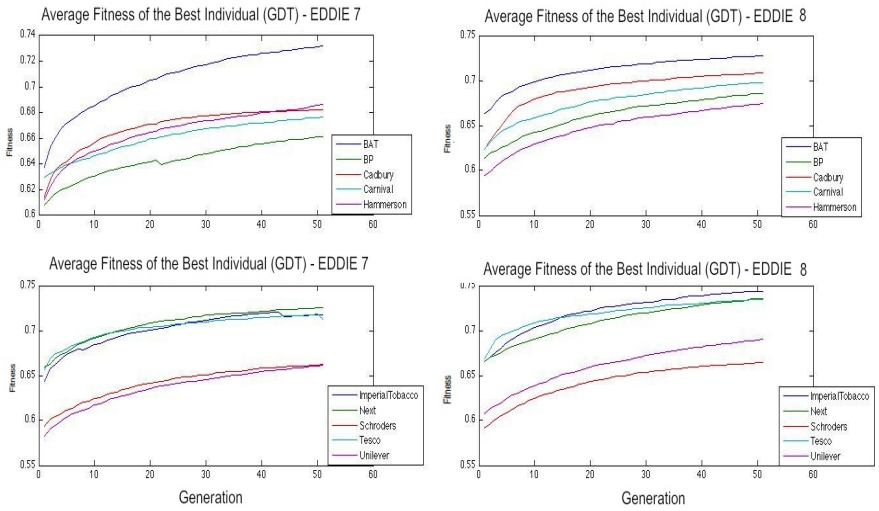


Figure 6.6: Average $Best_{Fit}$. We first obtain the best GDT's fitness per generation, for each one of the 50 runs. This happens for both algorithms. We then calculate the average of these fitness values (over the 50 runs) and present them in this figure. For the convenience of the reader, we have split the stocks into 5 per graph (by alphabetical order). The graphs in the first column are for EDDIE 7 and the others for EDDIE 8.

observe how the GDTs' Avg_{Fit} changes over the 50 generations of a single run. We then repeat this procedure for each one of the 50 runs. Finally, we calculate the average, over these 50 runs, of Avg_{Fit}. Figure 6.5 presents these results. Each line in the graph denotes the average Avg_{Fit} for a different dataset. As we can see, the population of EDDIE 7 starts at generation 1 with an average fitness between 0.1-0.2, for all stocks. This quickly rises to 0.4-0.5 and stabilizes around 0.6, with half of the stocks slightly exceeding this level. On the other hand, EDDIE 8's population average fitness for all stocks starts from a much higher point, around 0.3. Fitness here also rises quickly to 0.5-0.6 and stabilizes between 0.6 and 0.7. As we can observe, the average training fitness population of EDDIE 8 is somewhat higher than EDDIE 7's. It is obvious that EDDIE 8's grammar has allowed it to come up with better individuals in the first generation, and thus start with a population that has higher fitness.

For the second part of our analysis, we compare the fitness of the best individual (i.e. the GDT with the highest fitness) per generation; this fitness is called $Best_{Fit}$. So now instead of calculating the average fitness of the whole population for each generation, we just obtain the highest fitness. We can thus present how the highest fitness changes over the 50 generations of a single run. We then repeat this procedure for each one of the 50 runs. Finally we find the average, over these 50 runs, of $Best_{Fit}$. Figure 6.6 presents these results. In order to get a clearer idea of these results, we have divided them into two graphs per algorithm. The first column presents the graphs for EDDIE 7, and the second one for EDDIE 8. The graphs at the top are for the first 5 stocks (in alphabetical order) and the bottom graphs are for the remaining 5 stocks. We can see that results vary per stock

for both algorithms, although they seem to follow the same pattern. The $Best_{Fit}$ values for EDDIE 7 start from a range of [0.58,0.66], for generation 1, and reach up to a range of [0.64,0.74], at the last generation. The datasets for EDDIE 8 seem to follow a very similar behavior: the $Best_{Fit}$ values start in the range of [0.58,0.67] and end up in the range of [0.65,0.75].

Table 6.5 also presents the average $BestFit$ values for the first and the last generation. Each stock has 4 values, 2 for EDDIE 7 and 2 for EDDIE 8. The top value represents the average $Best_{Fit}$ for generation 1, and the bottom value represents the average $Best_{Fit}$ for generation 50. EDDIE 8's $Best_{Fit}$ starts with higher fitness for 7 stocks. This means that there are 3 stocks for which EDDIE 7 has better initial values: Carnival (0.6298), Hammerson (0.6121), and Schroders (0.5935). In addition, at the end of the evolutionary procedure (generation 50), there are 2 stocks for which EDDIE 7's $Best_{Fit}$ is higher than EDDIE 8's: BAT (0.7320), and Hammerson (0.6894). However, these differences from EDDIE 7 are relatively small (below 1%).

Table 6.5: Average $Best_{Fit}$ for generation 1 and 50, for EDDIE 7 and EDDIE 8, over the 10 stocks. Each stock has 4 values, 2 for EDDIE 7 and 2 for EDDIE 8. The top value represents the average $Best_{Fit}$ for generation 1, and the bottom value represents the average $Best_{Fit}$ for generation 50.

Stock		EDDIE 7	EDDIE 8
BAT	Generation 1	0.6373	0.6635
	Generation 50	0.7320	0.7273
BP	Generation 1	0.6079	0.6138
	Generation 50	0.6612	0.6860
Cadbury	Generation 1	0.6144	0.6236
	Generation 50	0.6822	0.7084
Carnival	Generation 1	0.6298	0.6235
	Generation 50	0.6763	0.6977
Hammerson	Generation 1	0.6121	0.5944
	Generation 50	0.6864	0.6743
Imp. Tobacco	Generation 1	0.6438	0.6651
	Generation 50	0.7178	0.7439
Next	Generation 1	0.6591	0.6655
	Generation 50	0.7257	0.7360
Schroders	Generation 1	0.5935	0.5915
	Generation 50	0.6626	0.6643
Tesco	Generation 1	0.6569	0.6684
	Generation 50	0.7123	0.7346
Unilever	Generation 1	0.5825	0.6073
	Generation 50	0.6613	0.6906

As we can see, there are times where EDDIE 7 outperforms EDDIE 8, although this is only to a small degree. Nonetheless, this is quite interesting, because it indicates that there can be cases

where EDDIE 8 might not be able to outperform its predecessor. Of course, at this moment this is only an indication that comes from results during the training period and this is why more analysis needs to be conducted.

6.4.2 Summary results for testing period

In this section we present summary results for the two algorithms, after the GDTs were applied to the testing period. The first part presents the averages of the metrics we used and the second part presents the improvements and diminutions caused by the best GDT evolved by EDDIE 8.

6.4.2.1 Average Results

We start with the average results for Fitness. In this way, we can have a general view of how the two algorithms have performed. We then move to the performance measures (RC, RMC and RF).

Figure 6.7 presents the average fitness results over the 50 runs for EDDIE 7 and EDDIE 8. As mentioned at the beginning of this section, the results have been normalized and are in the scale of [0,1]. As we can see, EDDIE 8 performs better than EDDIE 7 at 5 stocks (BAT, BP, Carnival, Hammerson, Tesco) and worse at the other 5 (Cadbury, Imperial Tobacco, Next, Schroders, Unilever). In order to test for the statistical significance of these results, we use the Kolmogorov-Smirnov test (K-S). We find that EDDIE 8 is better in only 3 stocks (BP, Carnival, Hammerson) and worse in 4 (Cadbury, Next, Schroders, Unilever), at a 5% significance level. A detailed table of the p-values for the K-S tests is provided in Appendix A.

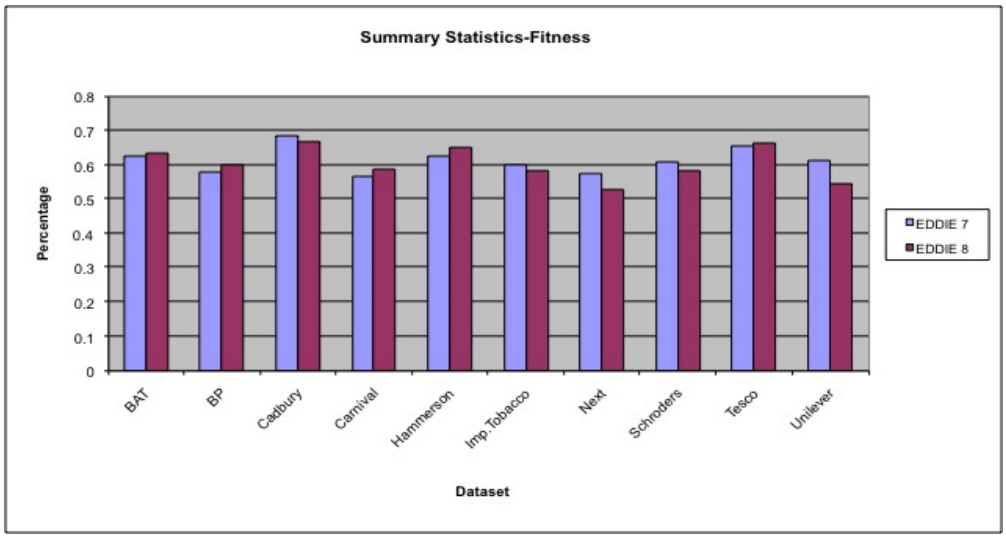


Figure 6.7: Summary results over 50 runs for fitness for EDDIE 7 and EDDIE 8. Results are normalized to [0,1] scale.

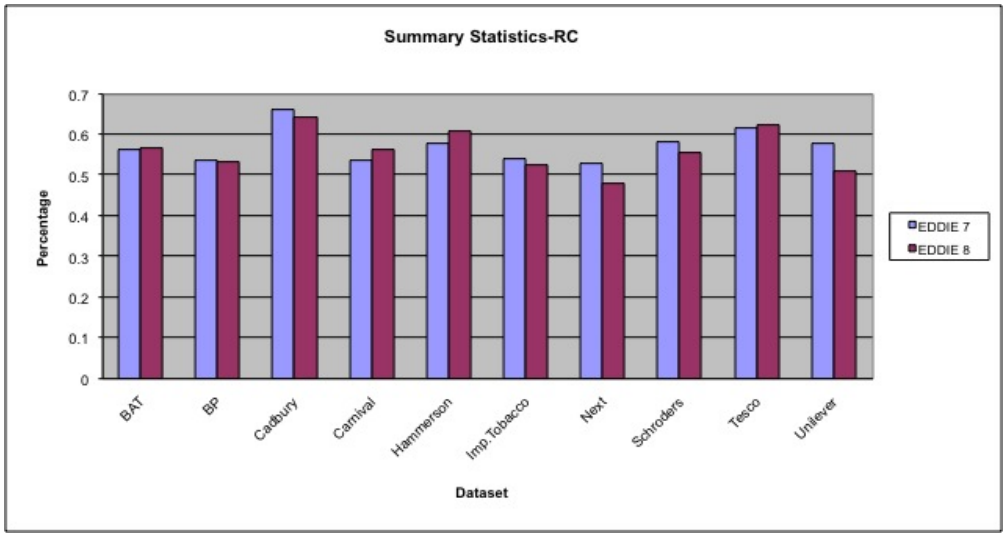


Figure 6.8: Summary results over 50 runs for RC for EDDIE 7 and EDDIE 8. Results are on the scale of [0,1].

We get a similar picture for the rest of the summary statistics results, namely RC, RMC and RF. Regarding the average RC (Figure 6.8), EDDIE 8 is significantly better at 2 stocks only (Carnival, Hammerson), whereas it performs worse in 5 (Cadbury, Imperial Tobacco, Next, Schroders, Unilever). Figure 6.9 shows that EDDIE 8 is better at only 1 stock (Hammerson), in terms of average RMC, whereas EDDIE 7 performs better at 5 (BAT, BP, Next, Schroders, Unilever). Finally, Figure 6.10 informs us that EDDIE 8 is better at 4 stocks (BAT, BP, Carnival, Tesco), in terms of RF, and worse at 5 (Cadbury, Imperial Tobacco, Next, Schroders, Unilever). The reader should bear in mind when reading the figures that we are interested in maximizing the values of Fitness and RC, and minimizing the values of RMC and RF. So when we say that EDDIE 8 performs better, in terms of fitness and RC, it means that these values have increased; on the other hand, when we say that EDDIE 8 performs better, in terms of RMC and RF, this means that these values have decreased. In addition, we should again mention that all of the results reported here have been tested by the K-S test and were found significant at a 5% significance level, and that the p-values of these tests are provided in Appendix A.

Finally, we should mention that a single run of either version does not last for more than a few minutes. EDDIE 8 is slightly slower than EDDIE 7, due to its large search space, but this fact does not seem to significantly affect its runtime.

6.4.2.2 Best GDTs

In this section, we investigate the improvements and diminutions caused by the best GDT that was evolved by EDDIE 8. From now on, we will be referring to this GDT as *Best-8*. *Best-8* is essentially the GDT with the highest fitness at the end of the training period, among all 50 runs. It

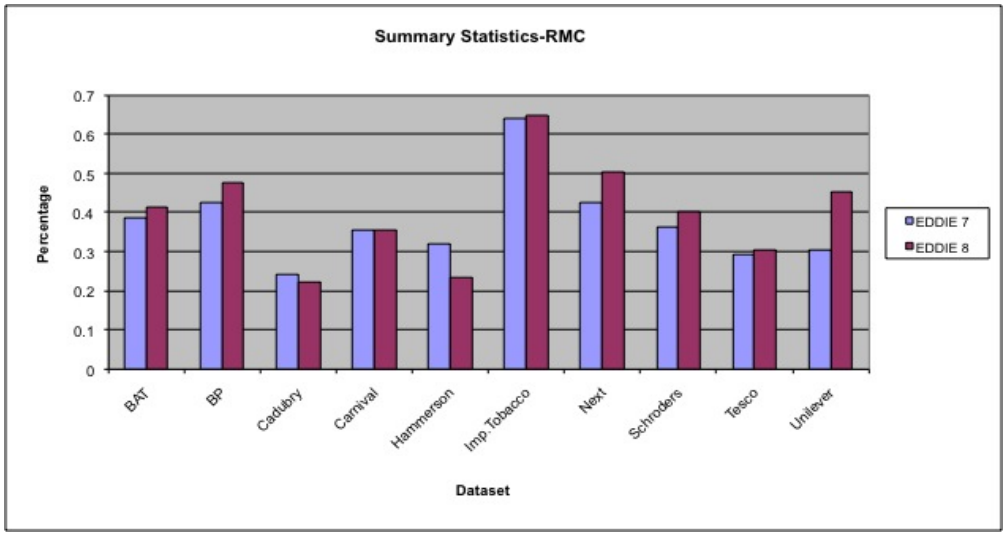


Figure 6.9: Summary results over 50 runs for RMC for EDDIE 7 and EDDIE 8. Results are on the scale of [0,1].

is thus the best solution that EDDIE 8 could come up with, after these 50 individual runs. After obtaining *Best-8*, we apply it to the testing period. Likewise, we obtain the best GDT evolved by EDDIE 7, named *Best-7*, and also apply it to the testing period.

The reason for choosing to compare the best GDTs is quite obvious. If an investor was using EDDIE to assist him with his investments, he would run the algorithm many times, and then pick the best GDT that was produced during training. Thus, by comparing *Best-7* and *Best-8*, we can get insight into which EDDIE version would be more effective to an investor's predictions.

Table 6.6 presents the improvements and diminutions caused by *Best-8*, after having calculated the differences between *Best-7* and *Best-8*, for each metric. Thus, an entry with a positive sign indicates that *Best-8* has improved the results in that metric by the respective percentage. Likewise, an entry with a negative sign indicates that *Best-8*'s results for that metric have declined by the respective percentage.

In addition, the last two rows of Table 6.6 present the mean of the above improvements and diminutions. Therefore, when we want to calculate the mean of improvements for Fitness, we sum up the values where Fitness is positive; we hence sum up the Fitness values for BAT (7.31), BP (1.05), Carnival (10.15), Tesco (3.27), Unilever (9.72) and then divide them by 5 (since that is the number of stocks with positive sign). Thus, when we want to calculate the mean of improvements for a metric, we calculate the mean for those values that have *positive sign*. On the other hand, when we want to calculate the mean of diminutions, we calculate the mean for those values that have *negative sign*. The same process stands for all metrics on the table.

Finally, apart from Fitness and the three metrics presented earlier in Chapter 4, Table 6.6 uses two additional metrics: Annualised Average Rate of Return (AARR), and Rate of Positive Returns (RPR). Since the EDDIE application lies in finance, we consider that it would be beneficial to an

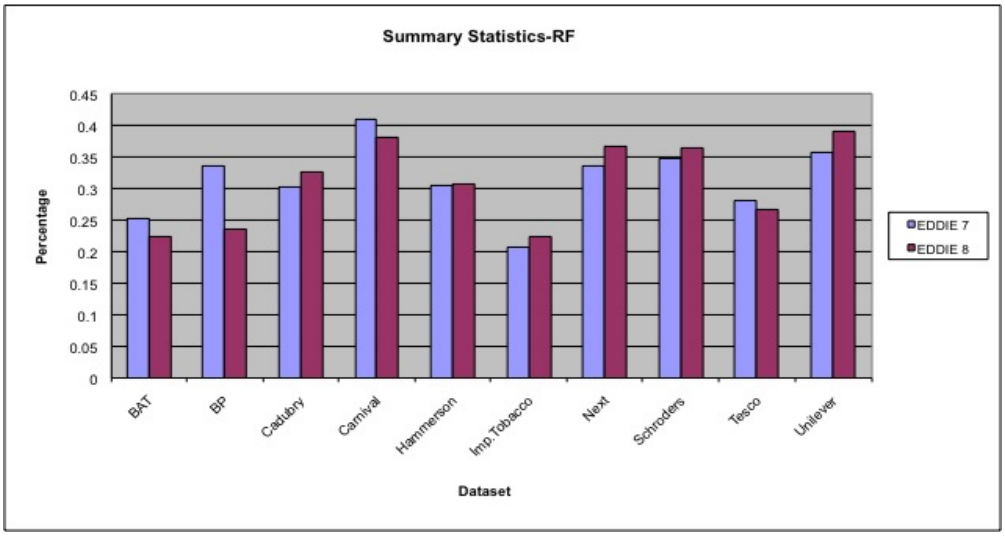


Figure 6.10: Summary results over 50 runs for RF for EDDIE 7 and EDDIE 8. Results are on the scale of [0,1].

investor to use as a reference performance criteria that are related to investment return. Obviously, the higher these metrics are, the higher the return for the investor. The formulas for these two additional metrics are presented in Appendix B. We should emphasize that these two metrics are given here only as reference, and are not part of the fitness function that EDDIE 7 and EDDIE 8 use.

What we can observe from Table 6.6 is that *Best-8* does better than Best-7 for 5 stocks in terms of Fitness (BAT, BP, Carnival, Tesco, Unilever), for 4 stocks in terms of RC (BAT, Carnival, Tesco, Unilever), for 4 stocks in terms of RMC (BAT, Carvival, Imp. Tobacco, Schroders), for 6 stocks in terms of RF (BAT, BP, Carnival, Hammerson, Tesco, Unilever), for 5 stocks in terms of AARR (BAT, BP, Imp. Tobacco, Tesco, Unilever), and for 5 stocks in terms of RPR (BAT, BP, Carnival, Tesco, Unilever). The differences in the values of the metrics are often quite big; for instance, EDDIE 8 has improved the Fitness of BAT and Carnival by 7.31 and 10.15%, respectively. What is even more remarkable is the differences in AARR: 31.03% for Tesco, and 48.81% for Unilever. Similar extremes can be observed for the diminutions. However, it seems that the improvements of *Best-8* have a greater impact than its diminutions.

To make this clearer, let us move our focus to the last two rows of the table, where the mean of *Best-8*'s improvements and diminutions in all metrics is presented. As we can see, with the exception of RF, improvements have, on average, had a greater effect than diminutions(6.30% vs -4.22% [Fitness], 8.00% vs -3.83% [RC], 11.78% vs -6.35% [RMC], 6.92% vs -6.96% [RF], 25.55% vs -16.61% [AARR], 8.62% vs -5.46% [RPR]). *This is a very important result, because it indicates that an investor using EDDIE 8's best GDT would on average gain more than if he was using EDDIE 7's best tree*. We can thus conclude from the above that, given a high number of individual runs (e.g., 50 runs), EDDIE 8 would find a very good solution, which on average,

Table 6.6: Improvements and diminutions of *Best-8* in Fitness and the other metrics, for all 10 stocks. The data presented in the first ten rows is basically the difference between metric values of EDDIE 8 and EDDIE 7. Finally, the last two rows of the table present the mean of the improvements and diminutions of *Best-8* to the metrics.

Stock	Fitness	RC	RMC	RF	AARR	RPR
BAT	7.31%	8.00%	8.07%	5.66%	10.95%	4.08%
BP	1.05%	-1.67%	-1.86%	7.45%	17.07%	5.26%
Cadbury	-10.48%	-11.33%	-17.32%	-6.48%	-7.03%	-7.11%
Carnival	10.15%	10.67%	13.87%	7.86%	-6.19%	10.90%
Hammerson	-0.22%	-0.33%	-1.94%	0.57%	-3.29%	-0.03%
Imp.Tobacco	-1.85%	-1.33%	12.97%	-7.83%	19.90%	-5.31%
Next	-7.59%	-7.33%	-7.07%	-8.28%	-17.92%	-5.31%
Schroders	-0.96%	-1.00%	12.22%	-5.27%	-48.61%	-9.55%
Tesco	3.27%	3.00%	-3.43%	6.05%	31.03%	5.46%
Unilever	9.72%	10.33%	-6.52%	13.91%	48.81%	17.38%
Mean Improvement	6.30%	8.00%	11.78%	6.92%	25.55%	8.62%
Mean Diminution	-4.22%	-3.83%	-6.35%	-6.96%	-16.61%	-5.46%

outperforms EDDIE 7's best solution *in terms of all performance measures*. This thus guarantees that an investor who uses EDDIE 8 will have more profit than using EDDIE 7.

6.4.2.3 Discussion on the summary statistics results

So far we have presented summary statistics for EDDIE 7 and EDDIE 8. From what we saw in the previous sections, EDDIE 7 outperforms EDDIE 8 in more stocks, in terms of all average statistics (Fitness, RC, RMC and RF). On the other hand, EDDIE 8 outperforms EDDIE 7 in terms of the average results of the best GDT.

An interesting observation from the above is that although EDDIE 8's best GDT can on average perform better than the one of EDDIE 7, this superiority is not reflected in the mean values of Fitness, RC, RMC, and RF. EDDIE 8 is able to come up with very good GDTs, sometimes even better than EDDIE 7's. However, the problem is that it does not come up with such trees often enough. Figure 6.11 illustrates this problem. It presents the relationship between performance (i.e. fitness) (x-axis) and precision (y-axis).[4] It is divided into two parts. The top graph (Figure 6.11a) presents the performance-precision values for stocks where EDDIE 8's average fitness is lower than EDDIE 7's. Let us denote these two fitness values by $ED8_{Fit}$ and $ED7_{Fit}$, respectively. The bottom graph (Figure 6.11b) presents the performance-precision relationship for stocks where $ED8_{Fit} > ED7_{Fit}$.

[4] "Precision shows the degree to which repeated measurements under unchanged conditions show the same results". (Robert Taylor, 1999) It is defined as the fraction of TP predictions, over the sum of TP and FP.

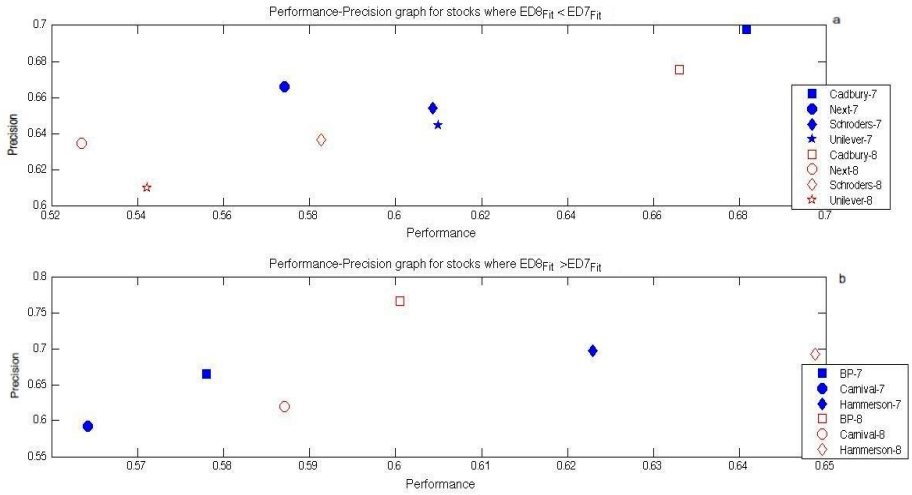


Figure 6.11: Performance-precision relationship. The x-axis presents the performance (average fitness) and the y-axis the average precision over 50 runs. The top graph (a) presents the stocks where $ED8_{Fit} < ED7_{Fit}$. The bottom graph (b) presents the stocks where $ED8_{Fit} > ED7_{Fit}$.

What we can observe from Figure 6.11 is that EDDIE 8 always has lower precision than EDDIE 7, for stocks where $ED8_{Fit} < ED7_{Fit}$. This indicates that EDDIE 8's GDTs are spread in a bigger fitness range, than the ones of EDDIE 7. It seems that *there is something preventing EDDIE 8 from having on a regular basis results with high fitness*. The picture is exactly opposite in Figure 6.11b, where $ED8_{Fit} > ED7_{Fit}$. We can see that here EDDIE 8 is not having difficulties finding good solutions, with precision at least as good as EDDIE 7's.

To summarize, the conclusions we can draw are the following:

- EDDIE 8 can perform better than EDDIE 7
- However, there are stocks where $ED8_{Fit} < ED7_{Fit}$
- EDDIE 8's best GDT does, on average, better than EDDIE 7's best GDT
- EDDIE 8's precision is lower than EDDIE 7's, for stocks where $ED8_{Fit} < ED7_{Fit}$. This does not happen for stocks where $ED8_{Fit} > ED7_{Fit}$
- Therefore, there is something which prevents EDDIE 8 from returning high fitness GDTs more often. This *unknown factor* reduces EDDIE 8's precision and only happens when $ED8_{Fit} < ED7_{Fit}$.

Hence, our next goal is to identify the reason why EDDIE 8 cannot return high fitness GDTs more often, for the stocks where $ED8_{Fit} < ED7_{Fit}$. One explanation could be that there is something special in the nature of the patterns of these stocks. We therefore need to deepen our analysis and try to provide an explanation of *when* and *why* EDDIE 8 outperforms EDDIE 7.

6.5 Artificial Datasets

So far, the experiments were tested with 10 empirical datasets. As we saw, results were not conclusive. It is not yet clear why EDDIE 8 cannot always outperform EDDIE 7. This section attempts to provide an answer to this question, by re-running the experiments under artificial datasets.

The reason for using artificial datasets is twofold. Let us start with the first reason. A potential drawback of experimental work with real data is that we cannot be sure that there are always patterns in the data. As a result, the failure of an algorithm in finding patterns could also be attributed to this fact. Of course, somebody could argue that both EDDIE 7 and EDDIE 8 have managed to find patterns and that EDDIE 7 just happens to be better in more cases. Nonetheless, creating our own artificial dataset can reassure us of the existence of such patterns. At the same time, artificial datasets can guarantee the absence of any noise. The second reason for using artificial datasets is that we have control over the nature of the patterns. This is very important, because it enables us to study the weaknesses and strengths of the algorithms, i.e. in what kind of data would EDDIE 7 or EDDIE 8 perform better. This can thus help us understand the reason why we cannot always have $ED8_{Fit} > ED7_{Fit}$.

6.5.1 Artificial Datasets Methodology

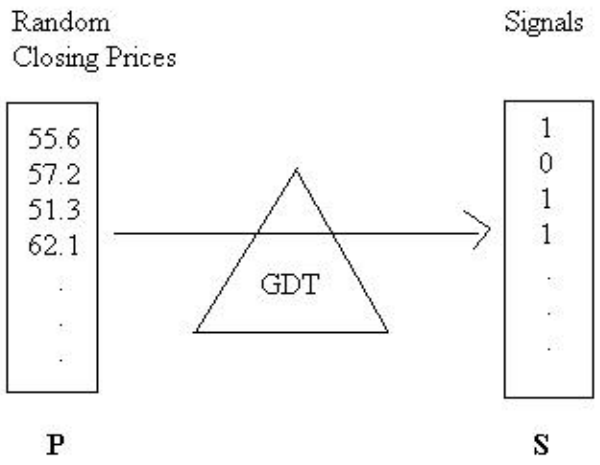


Figure 6.12: Methodology for creating an artificial dataset. The random closing prices (P) use a GDT previously derived from EDDIE, in order to create the set of signals.

It was explained earlier that, in traditional experiments for EDDIE, a dataset would consist of three parts: the daily closing prices, the technical indicators, and the buy/not-to-buy signals. In order to create the artificial data set, we need to replicate these three parts. First of all, we generate a set of random prices, which represents the daily closing prices. We then calculate the technical

indicators for this set. Finally, in order to create the signals for the random prices, we apply to these prices a GDT that was previously evolved with EDDIE. After the application of the GDT, a new set of signals is created. Basically the difference here from the traditional approach is that we do not use the question "will the price of the stock increase by $r\%$ in the next n days". The signals are created in a new way, based on a given GDT, which should be considered as a hidden function; EDDIE 7 and EDDIE 8 are thus asked to rediscover this hidden function. Therefore, after these three steps, we create a dataset like the ones EDDIE uses for its traditional experiments. Figure 6.12 shows the procedure we have just explained. The first column is the random prices, which are fed into a GDT for generating a set of signals.

It should also be mentioned that the evolved GDT which acts as the hidden function could be obtained either from EDDIE 7 or from EDDIE 8. In this way, the patterns could come from EDDIE 7's search space only, or from a larger search space (EDDIE 8). As mentioned above, this is the strength of this approach. Not only are we sure that patterns exist in our dataset, we are also able to determine which search space these patterns come from. Hence, having an artificial dataset allows us to control the nature of the patterns. And of course, being able to control the nature of the patterns allows us to observe the differences in the behaviour of the two versions.

Finally, let us introduce some important terminology. As mentioned, the evolved GDT which acts as the hidden function can be obtained either from EDDIE 7 or from EDDIE 8. Thus, when it is obtained by EDDIE 7, this GDT is called GDT-7, whereas when it is obtained by EDDIE 8, this GDT is called GDT-8. In addition, when we present results from EDDIE 7, we are going to denote these results as EDDIE 7_GDT-7, if the patterns come from EDDIE 7's vocabulary, or EDDIE 7_GDT-8, if the patterns come from EDDIE 8's. Equivalently, EDDIE 8's results will be denoted either as EDDIE 8_GDT-7 or EDDIE 8_GDT-8, depending on which vocabulary the patterns come from.

6.5.2 Experimental Parameters

As we said in the previous section, the prices of the data were randomly generated. This can be clearly observed in Figure 6.13. The training period was 1000 days and the testing period 300, as in the empirical datasets' experiments.

Moreover, Table 6.7 presents the parameters of the EDDIE algorithm. The R constraint is set in the range of [50,65], with n and r being 20 days and 4%, respectively. The weights and the Period length remain the same, as earlier in Table 6.4.

The GP parameters are the same, as presented earlier at Table 6.3. For statistical purposes, we again run the GP for 50 times.

6.5.3 Artificial Dataset Results

This section is divided into two parts. The first part presents the results for signals generated by GDT-7, and the second one the results for signals generated by GDT-8.

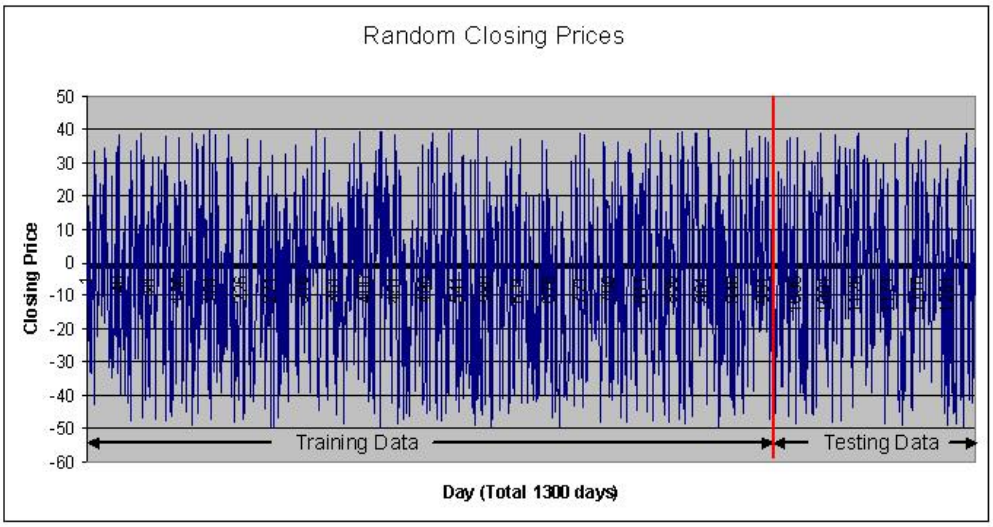


Figure 6.13: Random closing prices for a period of 1300 days. The first 1000 consist the training period, and the rest 300 the testing period.

Table 6.7: EDDIE Parameters

EDDIE Parameters	Value
R	[50,65]
n	20
r	4
w_1	0.6
w_2	0.1
w_3	0.3
Period (EDDIE 8)	[2, 65]

6.5.3.1 GDT-7

Table 6.8 presents the summary results for the testing period over 50 GP runs. As we can observe, EDDIE 7_GDT-7 (EDDIE 7 with patterns that have been created by GDT-7) is doing significantly better in all performance measures and is very close to finding a perfect solution[5] (RC=97.36, RMC=2.4, RF=1.3). It is also interesting to observe that the standard deviation of EDDIE 7_GDT-7's results is small, which basically indicates that the values for RC, RMC and RF are very similar among the 50 runs. This however does not happen with EDDIE 8_GDT-7 (EDDIE 8 with patterns that have been created by GDT-7), where the standard deviation is bigger for all Fitness, RC, RMC and RF. As we can also see from Table 6.8, the mean values of all Fitness, RC, RMC and RF have

[5] A perfect solution can be defined as any GDT that fits the testing dataset perfectly. This essentially means that RC would be 100%, and RMC=RF=0

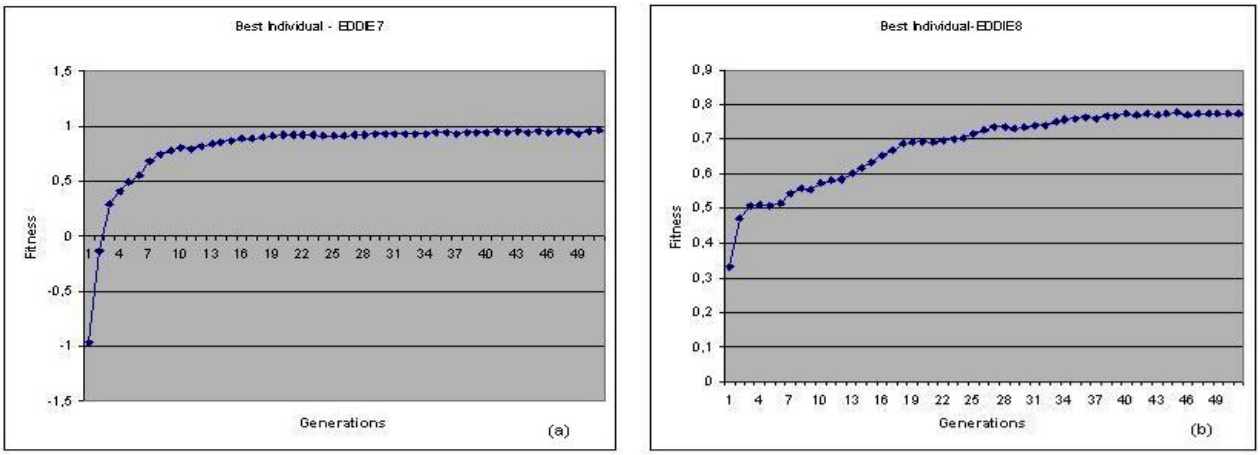
Table 6.8: Summary Results for the testing period, over 50 runs, for EDDIE 7 and EDDIE 8. The patterns were created by GDT-7. The results are shown in % percentages.

Summary Results						
EDDIE 7_GDT-7						
	Fitness	RC	RMC	RF	AARR	RPR
Mean	97.78	97.36	2.4	1.3	47691.23	90.42
St.Dev.	1.36	1.58	1.97	1.51	1146.75	0.2
Max	99.75	99.66	4.8	6.36	49246.4	90.73
Min	94.79	94.66	0.4	0	100	90.1
EDDIE 8_GDT-7						
	Fitness	RC	RMC	RF	AARR	RPR
Mean	80.28	77.66	17.11	15.36	46730	90.48
St.Dev.	8.95	10.28	10.75	6.8	4885.08	1.03
Max	92.04	91	35.57	25.32	51648.92	92.85
Min	67.95	63.66	2.4	6.03	34138.56	88.29

worsen to 80.28, 77.66, 17.11 and 15.36, respectively. Furthermore,we can also observe that the Min and Max values of the above metrics are in a much bigger range for EDDIE 8. Also, EDDIE 7 has higher mean AARR, whereas the mean RPR is quite similar, for both EDDIE 7 and EDDIE 8.

Furthermore, Figure 6.14 presents the training fitness of the best individuals per generation, as we did earlier in Section 6.4.1. As a reminder, what we do is to calculate the highest fitness of the whole population for each generation. After doing this for each one for the 50 generations, we repeat this whole procedure for each one of the 50 runs. Finally, we calculate the average highest fitness for each generation, over the 50 runs. As we can see from Figure 6.14, EDDIE 7_GDT-7 comes to a solution very quickly, which is actually very close to the optimal one (i.e. $Fitness = 1$). On the other hand, EDDIE 8_GDT-7 does not seem to reach fitness levels as high as EDDIE 7_GDT-7 does. It only manages to reach around 80%, which is quite high, but not as high as EDDIE 7_GDT-7's.

The poor results could be explained by the large increase in the search space of EDDIE 8_GDT-7. For this reason, we tested EDDIE 8_GDT-7's performance with a bigger population (1500 individuals) and more generations (100). The reasoning for this was that because of the big search space, EDDIE might have needed more candidate solutions or more time in order to perform better. However, as we can see from Table 6.9, EDDIE 8_GDT-7's summary results did not seem to have any significant improvement (mean of Fitness was improved from 80.28 to 81.19, mean of RC was improved from 77.66 to 78.72, mean of RMC improved from 17.11 to 15.86 and mean of RF 15.36 to 14.91).



(a) EDDIE 7 (b) EDDIE 8

Figure 6.14: Highest training fitness per generation for EDDIE 7_GDT-7 [Figure 6.14(a)] and EDDIE 8_GDT-7 [Figure 6.14(b)], over 50 runs.

6.5.3.2 GDT-8

The results in this section are quite different. As we can see from Table 6.10, none of EDDIE 7_GDT-8 (EDDIE 7 with patterns that have been created by GDT-8) or EDDIE 8_GDT-8 (EDDIE 8 with patterns that have been created by GDT-8) seem to be able to find solutions very close to the optimal one. In addition, this time EDDIE 8_GDT-8 is performing better than EDDIE 7_GDT-8, in terms of measures (Fitness, RC, RMC and RF). Furthermore, EDDIE 8_GDT-8's maximum values for Fitness (94.00) and RC (92.67) and minimum values for RMC (8.25) and RF (0) are significantly better than the ones of EDDIE 7_GDT-8 (80.83, 74.33, 29.13 and 19.42 respectively). Finally, EDDIE 8's average AARR is significantly better; average RPR is also slightly better for EDDIE 8.

In order to see whether the difference in the performance measures is indeed significant, we run a two-sample Kolmogorov-Smirnov non-parametric test. The null hypothesis is that the two samples come from the same continuous distribution; it is rejected if the value obtained by the test is greater than the critical value. Table 6.11 shows us that H_0 is clearly rejected for all performance measures at a 5% significance level, with p-values well below 5%.

6.5.4 Discussion on the artificial datasets' results

From the above experiments, we have shown that both EDDIE 7 and EDDIE 8 have been able to discover solutions very close to the hidden functions (see Figure 6.12). This is very important and proves the effectiveness of these two methods. Also, it should not be considered as something trivial, since it cannot be assumed that other, arbitrary methods would be able to do this.

Table 6.9: Summary Results for the testing period, over 50 runs, for EDDIE 8_GDT-7. The results are shown in % percentages. The numbers of generations and population have changed to 100 and 1500, respectively.

Summary Results						
EDDIE 8_GDT-7						
	Fitness	RC	RMC	RF	AARR	RPR
Mean	81.19	78.72	15.86	14.91	47460.18	90.74
St.Dev.	8.25	7.67	10.48	4.35	4857.71	0.9
Max	92.16	91.33	35.57	21.21	57879.55	93.29
Min	67.76	64.66	0.48	6.17	38344.04	89.44

However, our analysis also showed that EDDIE 8 cannot always perform better than EDDIE 7. It seems that there is a trade-off between 'searching in a bigger space' and 'search effectiveness'. It is obvious that the results are affected by the patterns in the dataset. If these patterns come from EDDIE 8's vocabulary (and thus are in its search space), EDDIE 8 can find better solutions. This is something we anticipated, since EDDIE 7 cannot search for these solutions. From Figure 6.15, a look into the components of the trees that EDDIE 8 used during the evolutionary procedure of a single run would show us that EDDIE 8 indeed took advantage of its large search space and came up with solutions that it is impossible for EDDIE 7 to find. The x-axis of this figure presents the range of the periods (2-65 days) that the 6 technical indicators are using. The y-axis shows the occurrence of these indicators, in the logarithmic scale, after 50 generations of a single run. As we can see, all indicators are used and they use many different periods within the range of 2-65 days.

However, a question arises, whether just using a bigger number of indicators is enough to get better prediction results. This point becomes even clearer in cases where the patterns in the dataset come from a very small search space, like the one of EDDIE 7's. It then seems very hard for EDDIE 8 to find as good a solution as EDDIE 7 does. The solutions are indeed in its search space, but because they come from a very small area of it, it seems that EDDIE 8 cannot search effectively enough to find them. The search space has increased significantly. To make this clearer, let us give an example: if a GDT can have a maximum of k Variables (EDDIE 7) or k VarConstructors (EDDIE 8), then the permutations of the available 12 indicators[6] under EDDIE 7 are 12^k; on the other hand, the permutations of the available 384 indicators[7] under EDDIE 8 are 384^k. It is thus obvious that EDDIE 8's search space is significantly larger, which can therefore explain the difficulties of EDDIE 8 of consistently finding good solutions. There is an obvious trade-off between the more expressive language that EDDIE 8 provides and the search efficiency of EDDIE 7.

[6]We are using 6 different indicators, with 2 periods each, thus $6 * 2 = 12$.
[7]We are using 6 different indicators with 65-1=64 periods each, thus $64 * 6 = 384$.

Table 6.10: Summary Results for the testing period, over 50 runs, for EDDIE 7 and EDDIE 8. The patterns were created by GDT-8. The results are shown in % percentages.

Summary Results						
EDDIE 7_GDT-8						
	Fitness	RC	RMC	RF	AARR	RPR
Mean	79.07	72.09	23.49	18.30	22569.97	90.02
St.Dev.	0.75	72.0996	23.495	18.30	22569.97	190.02
Max	80.83	74.33	29.13	20.30	34438.01	92.27
Min	76.15	68.33	19.42	16.67	10000	88.71
EDDIE 8_GDT-8						
	Fitness	RC	RMC	RF	AARR	RPR
Mean	85.28	81.91	21.14	5.83	34741.70	91.14
St.Dev.	5.36	5.48	3.73	6.52	4547.87	0.70
Max	94.00	92.67	32.04	20.85	48613.98	92.07
Min	72.65	68.33	8.25	0	10000	88.73

Table 6.11: Kolmogorov-Smirnov test for testing whether the differences between EDDIE 7 and EDDIE 8 are significant at a 5% significance level.

Kolmogorov-Smirnov test						
	Fitness	RC	RMC	RF	AARR	RPR
p-value	4.0089e-15	1.2488e-16	4.0202e-07	3.2843e-18	4.9988e-19	1.2561e-07

6.6 Extending the Artificial Datasets' Results

So far, we have made two valuable observations in Sections 6.4 and 6.5:

- EDDIE 8 has lower precision than EDDIE 7, for stocks where $ED8_{Fit} < ED7_{Fit}$
- EDDIE 8 performs better than EDDIE 7 (on artificial datasets), when patterns come from EDDIE 8's vocabulary. If, on the other hand, patterns come from EDDIE 7's vocabulary, then EDDIE 8 is having difficulties discovering them, and thus ends up with lower performance

We also mentioned at the end of Section 6.4 that a plausible explanation for EDDIE 8's lower precision is that the nature of the patterns in the data prevents EDDIE 8 from performing well more often. Now, after having the insight from the artificial datasets' results, we want to see if we can apply our conclusions to the empirical datasets. We shall therefore move our focus to the indicators that EDDIE 8's GDTs use and examine their relation with EDDIE 7's vocabulary. We saw earlier that if patterns in the hidden function come from EDDIE 7's vocabulary, then EDDIE

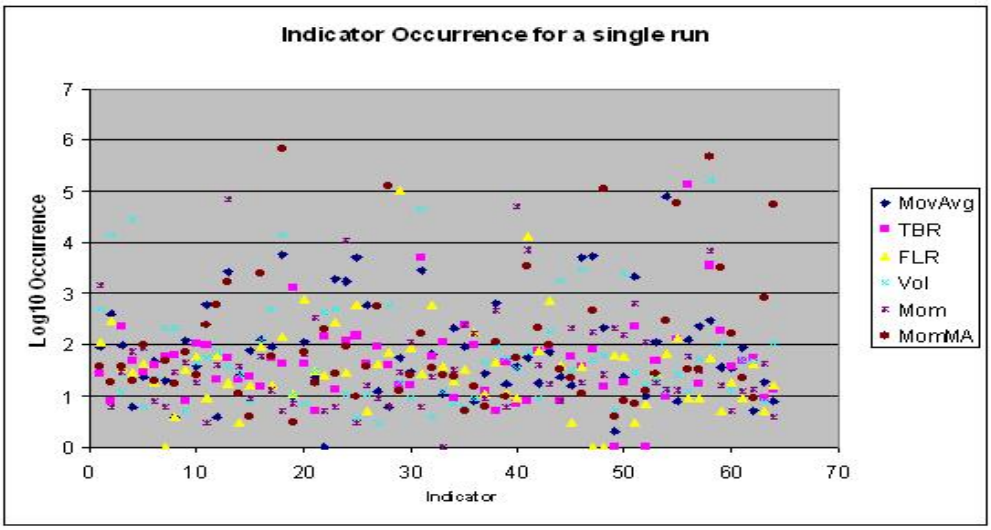


Figure 6.15: Indicators occurrence after 50 generations for a single run. This occurrence is presented in the y-axis and is in a logarithmic scale with a base of 10. The range of the period for the 6 indicators is from 2 to 65 days, and is presented in the x-axis. There are 6 different colours in the graph, each one denoting a different technical indicator.

8 is having difficulties discovering them. This is what we are going to investigate now with the empirical datasets. Our aim is to show that when $ED8_{Fit} < ED7_{Fit}$, it is because the GDTs of EDDIE 8 contain a high percentage of indicators that come from the vocabulary of EDDIE 7, or indicators very close to it. If this happens, it means that EDDIE 8 needs to look for patterns in a very small search space, and thus faces difficulties in doing so.

One more thing to say is that here there are no hidden functions that EDDIE 8 is trying to discover. When dealing with empirical datasets, we have "solutions". A solution should be considered as the GDT that had the highest fitness at the end of the training period, and was then applied to the testing period. This GDT was the best solution EDDIE 8 could come up with for that specific run, for that specific dataset.

Let us now have a look into the components of the best solution of EDDIE 8, which as we said in Section 6.4.2.2 is called *Best-8*, which as we already know is the best tree that EDDIE 8 could find among a total of 50 runs[8]. We want to examine the components of *Best-8*, and calculate the percentage of indicators that come from the vocabulary of EDDIE 7. Figures 6.16 and 6.17 present us these results, for stocks where $ED8_{Fit} < ED7_{Fit}$ (6.16) and stocks where $ED8_{Fit} > ED7_{Fit}$ (6.17). The x-axis shows the number of days that an indicator of EDDIE 8 is away from the pre-specified indicators of EDDIE 7. For instance, "+/-1" means that EDDIE 8's indicator has

[8]The ideal case would be of course to consider as a solution a GDT that would be able to fit the data with 100% accuracy. However, this is not possible in forecasting with real data. We should thus consider as a solution other GDTs, which would be able to fit the data sufficiently well. In our case, we choose the best performing GDT out of 50 runs.

a distance from EDDIE 7's indicators by +1 or -1 day. Thus, since EDDIE 7's indicators have lengths 12 and 50 days, EDDIE 8's indicators in this example could be 11, 12, 13, 49, 50 and 51. The y-axis presents the percentage of EDDIE 8's indicators that come from EDDIE 7's vocabulary.

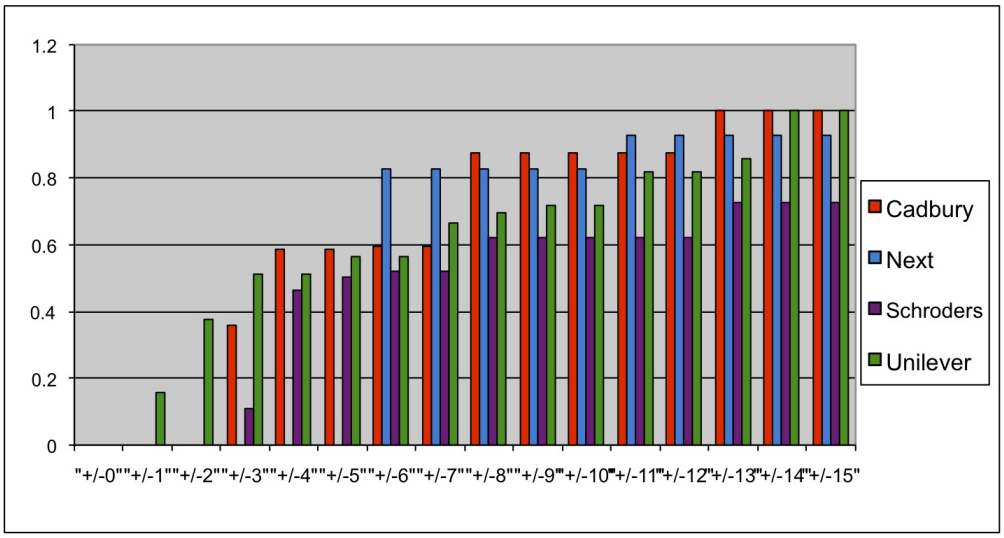


Figure 6.16: Percentage of EDDIE 8's indicators that are close to EDDIE 7's vocabulary, for the stocks that $ED8_{Fit} < ED7_{Fit}$. This percentage can be viewed in the y-axis. The x-axis presents the number of days that an EDDIE 8's indicator is away from the pre-specified indicators of EDDIE 7.

As we can see from Figure 6.16, even though none of the 4 stocks' *Best-8* trees are using any indicators from the vocabulary of EDDIE 7 (all stocks have 0% at +/- 0 days), they are using indicators in a very close range. To be more specific, 50-60% of the *Best-8* indicators for these stocks are close to indicators from EDDIE 7's search space, in a range of [-4,+4] days; this percentage increases to 50-80% for range [-6,+6] days.

On the other hand, for stocks where EDDIE 7 is outperformed by EDDIE 8 (Figure 6.17), the previous percentage is much lower. For the range of [-4,+4] days, *Best-8* for all 3 stocks has a percentage of 18-30%. For the range of [-6,+6] days, this percentage increases only a little, and is in the range of 18-44%, which is clearly much lower than the percentages we observed in Figure 6.16.

Our intuition hence seems to be verified. EDDIE 8's performance is indeed affected by the nature of the patterns in the GDTs. When these patterns come from EDDIE 8's broader vocabulary, then EDDIE 8 has no problem finding these GDTs. On the other hand, when solutions come from a very small space (in our case a search space around the one of EDDIE 7), then EDDIE 8 is having difficulties focusing there. This, as a consequence, affects EDDIE 8's performance results, which become poorer than those of EDDIE 7.

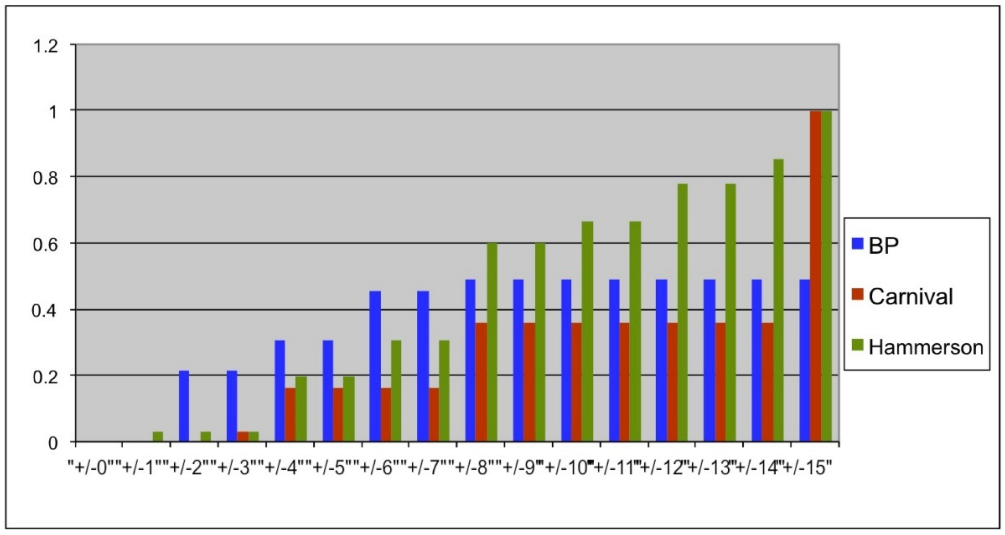


Figure 6.17: Percentage of EDDIE 8's indicators that are close to EDDIE 7's vocabulary, for the stocks that $ED8_{Fit} > ED7_{Fit}$. This percentage can be viewed in the y-axis. The x-axis presents the number of days that an EDDIE 8's indicator is away from the pre-specified indicators of EDDIE 7.

6.7 Conclusion

In this chapter we presented two investment opportunities forecasting algorithms, EDDIE 7 and EDDIE 8. EDDIE 8 is an extension of EDDIE 7, because it extends its grammar. Traditionally, EDDIE 7 and other similar GP algorithms use predefined indicators from technical analysis to forecast the future movements of the price by using a pre-specified period length. In this approach, we suggested that it should be left to the GP to decide the optimal period length. EDDIE 8 is thus an improvement to the previous algorithm because it has richer grammar and also because it can come up with solutions that EDDIE 7 can never discover. In addition, the improvements introduced by the best GDT evolved by EDDIE 8, called *Best-8*, have on average a greater impact than its diminutions. This is a very important finding, because it indicates that an investor using EDDIE 8's best GDT would on average gain more than if he was using EDDIE 7's best tree. EDDIE 8's best tree was, on average, able to outperform EDDIE 7's best tree in terms of all performance measures. In addition, *Best-8* had significantly higher average annual return (AARR) than *Best-7*, which means that on an annual basis, an investor would make more profit if he was using EDDIE 8. The above thus allow us to characterize EDDIE 8 as a successful extension to its predecessor, and a valuable forecasting tool.

However, there seems to be a trade-off between 'discovering new solutions' and 'effective search'. Results from 10 empirical datasets from FTSE 100 showed that EDDIE 8 cannot always outperform EDDIE 7 in terms of average fitness, RC, RMC and RF. In order to further understand this behaviour, we created an artificial dataset. The reason we did this was because empirical

datasets are exposed to noise. Thus, by using artificial datasets, which were noise-free, we could avoid the impact of noise. In addition, with artificial datasets we could control the nature of patterns. This thus helped us to identify the strength and weakness of EDDIE 7 and EDDIE 8. Results suggested that EDDIE 8 can outperform EDDIE 7, as long as the solutions come from its own vocabulary. If they come from EDDIE 7's, then EDDIE 8 is having difficulties finding these solutions, due to the fact that it has to look in EDDIE 7's narrow search space. These results were also verified by our empirical datasets.

We can thus conclude that the current version of EDDIE 8 has its limitations. Nevertheless, EDDIE 8 is still a very valuable tool, due to the fact that it can guarantee significantly higher profits than its predecessor, as already explained.

Future research will focus on improving EDDIE 8's search effectiveness. Of course there are different ways to do this. A promising way that we have already investigated is hyper-heuristics (Özcan et al, 2008; Hart et al, 1998; Burke et al, 2006, 2003), where we created a framework for financial forecasting (Kampouridis and Tsang, 2011). The results were promising, because hyper-heuristics led to a significant decrease in the RMC; another promising result was that the search became more effective, since more areas of the search space were visited. We believe that more sophisticated hyper-heuristic frameworks, which will include more heuristics than the ones used in Kampouridis and Tsang (2011), can lead to even better results. We therefore intend to focus in that direction. Moreover, another direction of our research could be to produce some new search operators or to create a new constrained fitness function.

This concludes Chapter 6. In the next chapter, we move our focus to the market fraction dynamics of financial markets.

Chapter 7

The Market Fraction Hypothesis

7.1 Introduction

As we saw earlier in Chapter 5, several works have focused on the mesoscopic structure of the financial markets, where market fraction[1] dynamics were investigated. One of the observations made about the market fraction of trading strategies that exist in a market is that it constantly changes (swings). As mentioned in Chapter 5, this is an interesting observation, because it gives us valuable insight about the fraction dynamics of financial markets, i.e., how the popularity among different types of trading strategies is interchanged. In addition, an immediate implication of this observation is that there cannot be a 'winner' type of trading strategy in the long run.[2] However, this challenges the necessity of attempting to forecast the market. Motivated by the above, we are interested in investigating in detail the dynamics of market fraction. We thus formulate this swinging feature, observed at the mesoscopic level of agent-based financial models, into a concrete hypothesis, called the Market Fraction Hypothesis (MFH). Formalizing the MFH is a very important task, because it is something that has not happened before and also because it allows us to suggest and formulate tests that will examine its plausibility.

After formalizing the hypothesis, we are interested in testing it. In order to do that, we create a new agent-based financial model, which combines characteristics from the N-type and the SFI-like models. The reason of using this new model is because of the limitations that exist in the N-type and SFI-like models, which we presented earlier in Chapter 5. For instance, we saw that a limitation of the N-type models is that they assume that the trading strategy types are static and pre-specified. By this we mean that the N-type models endow their agents with a specific number of trading strategy types from which they have to choose. To the best of our knowledge, the MFH has not been empirically examined under a more dynamic environment, where strategy types are not fixed and are not exogenously given. Thus, in this chapter we do not assume any prefixed

[1] A reminder that the term 'market fraction' refers to the fraction of the different strategy types that exist in a financial market.

[2] 'Winner' in this context means a trading strategy type that will attract a big fraction of financial traders over many time periods.

behavioral rule for any type of trading strategy. In other words, we are not going to define fixed types like fundamentalists and chartists, as it happens in the N-type models.

Moreover, we consider that agents that belong to the same trading strategy type are heterogeneous, while they can be similar. We consider that this departure will lead us to a more general and *realistic implication* of the MFH. Consider the two-type model (fundamentalists-chartists) we mentioned in Chapter 5. In this model, traders that belong to the same type always behave in *exactly the same way* at any given point in time. This could be considered as very unrealistic. In the real world, the behavioral rules of each trader are expected to be heterogeneous, and even if they can be clustered into types, this does not mean that these rules are exactly the same. Therefore, another characteristic of our agent-based financial model is the heterogeneity of the agents that belong in the same type of trading strategy.

After presenting our financial model, we suggest a testing methodology. Because of the fact that the observations of the MFH have so far been made under artificial market frameworks, we are interested in investigating the underlying market fraction dynamics under 'real' financial markets. We thus run tests under 10 international markets and hence provide a general examination of the plausibility of the MFH. *Another goal of our empirical study is to use the MFH as a benchmark and examine how well it describes the empirical results which we observe from various markets*. In particular, we are interested in knowing how this benchmark performs when we tune a key parameter of our model, i.e., the number of trading strategy types in the market. More details regarding this can be found in Section 7.6.

Therefore, the contributions that are going to be presented in this chapter can be summarized in the following way: (i) Formalizing the MFH, (ii) Suggesting a testing methodology, (iii) Testing the hypothesis under real datasets, (iv) Proposing a new agent-based financial model which (a) does not assume pre-fixed types of trading strategies and (b) allows heterogeneity among the strategies that belong in the same type.

The remainder of this chapter is organized as follows. Section 7.2 elaborates on the MFH. Section 7.3 presents our suggested model and also explains our motivation for the choice of tools we used for our tests, namely Genetic Programming (GP) (Koza, 1992; Poli et al, 2008) and Self-Organizing Map (SOM) (Kohonen, 1982). Section 7.4 presents the experimental designs. Section 7.5 addresses the methodology employed to test the MFH and explains the technical approaches needed to be taken to facilitate the testing of the MFH. These proposed approaches play an important role in our experiments, since they allow for a comparison of the types of trading strategies displayed in SOMs, throughout the different time periods. Section 7.6 presents the test results. It first starts by presenting the results over a single run for a single dataset. Then it continues by presenting the summary results over 10 runs for this dataset and it finally presents summary results for all datasets. Section 7.7 then tests the hypothesis under EDDIE 7 and EDDIE 8, in order to investigate whether the previously derived results are independent on the choice of the GP algorithm. Finally, Section 7.8 concludes this chapter and briefly discusses possible directions for further research.

7.2 The Market Fraction Hypothesis

As we have already pointed out, within a market there exist different types of trading strategies. The MFH tells us that the fraction among these types of strategies keeps changing (swinging) over time. The following two statements are the basic constituents of the MFH, and are based on a summary of the empirical development of the agent-based financial models, presented in Chen et al (2012).

1. In the short run, the fraction of different clusters of strategies keeps swinging over time, which implies a short dominance duration for any cluster.

2. In the long run, however, different clusters are equally attractive and thus their market fractions are equal.

The first statement means that it is not possible for a single strategy type to dominate the market by attracting an overwhelming fraction of market participants for many consecutive periods. In other words, according to the MFH a 'winner' type of trading strategy does not exist. Let us give an example by again using the fundamentalist-chartist model. If at time t fundamentalists dominate the market, the first MFH statement says that this should not happen for too long. Eventually there should be a "switch", and chartists would take over as the dominant strategy in the market. The term 'dominance duration' refers to the amount of time that a type of strategy attracts a high number of traders. This term will become technical for testing the MFH, and we shall make it precise later in the chapter, in Section 7.6.

Let us now move to the second statement. If the above continuous happening, then in the long run both the fundamentalist and chartist strategy types should have occupied about the same market share, i.e., about one half.[3].

As we can see, the implications of the MFH are very important. First of all, if the MFH holds, this means that all types of trading strategies that exist in a financial market will, at some point, become popular[4]. In other words, any type of trading strategies has equal chances of attracting a significant amount of traders. Nevertheless, this seems unrealistic, because this means that even a bad strategy can become popular. It is thus interesting to investigate if this can happen under real data. In addition, another implication of the MFH is that no strategy can remain popular over a long period of time, as it will soon be succeeded by another popular strategy. It is therefore also interesting to examine if this is true, because it would give us an insight of how financial markets are structured.

Hence, what we shall do in this chapter is test the above two MFH properties against our empirical data. It should again be said that we are interested in qualitative results, meaning that we want to see how close the real market behaves to what is described by the MFH under different

[3]This idea is first made rigorous by Kirman (1993), who attempted to solve a puzzling entomological problem, i.e., ants swinging among themselves within two identical sources of food.

[4]Popularity is equivalent to dominance, which means that a strategy type occupies many market participants (strategies).

numbers of clusters. The next section presents our model, along with the basic tools we used for testing the MFH.

7.3 Model

In this section, we present our agent-based financial model. This model first allows the creation of novel, autonomous and heterogeneous agents by the use of GP. The reason for using GP is because the market is considered to undergo an evolutionary process; this is inspired by Andrew Lo's Adaptive Market Hypothesis (AMH) (Lo, 2004, 2005), where Lo argued that the principles of evolution (i.e., competition, adaptation, and natural selection) can be applied to financial interactions. Thus, agents can be considered to be organisms that learn and try to survive.

After creating and evolving novel agents, we cluster them into types of trading strategies via SOM. These types are thus not pre-specified, but depend on the strategies of the agents.

The advantages of this approach are thus twofold: first of all, agents can create autonomous and heterogeneous trading strategies. Thus, even if two trading strategies belong to the same type of trading strategy (e.g. fundamental), it does not mean that these two strategies have to follow exactly the same trading rule, as it happens in the traditional agent-based model literature. In addition, when these trading strategies are categorized into types of trading strategies, they are not clustered into pre-specified, fixed types; on the contrary, the types depend on the existing trading strategies. *This thus makes our model more realistic*.

Next, we present the two techniques of our model, GP and SOM.

7.3.1 Genetic Programming as a Rule-Inference Engine

In this work, we assume that traders' trading strategies, are either not observable or not available. Instead, their strategies have to be *estimated* by the observable market price. Using macro data to estimate micro behavior is not new,[5] as many N-type empirical agent-based models have already performed such estimations[6] (Winker and Gilli, 2001; Gilli and Winker, 2003; Boswijk et al, 2007). However, such estimations are based on very strict assumptions, as we saw earlier (e.g., having pre-specified trading strategy types is considered to be a strict and unrealistic assumption). Since we no longer keep these assumptions, an alternative must be developed, and in this work we recommend Genetic Programming (GP).

As we have already mentioned, the use of GP is motivated by regarding the market as an evolutionary and selective process, as Lo (2004, 2005) suggests in his AMH. In this process, traders with

[5] 'Macro data' is generally a term used to mainly describe two categories of data: aggregated data, and system-level data (Diez-Roux, 2002). The former refers to data that combine information, such as unemployment statistics and demographics. The latter refers to information that cannot be disaggregated to lower level unities; such examples are the prices of a stock. On the other hand, 'micro behavior' refers to the study of the behavior of components of a national economy, such as individual firms, households and traders.

[6] For more information on such papers, we refer the reader to Chen et al (2012), which provides an excellent literature review on the topic.

different behavioral rules participate in the markets. Those behavioral rules which help traders gain lucrative profits will attract more traders to imitate them, and rules which result in losses will attract fewer traders. An advantage of GP is that it does not rest upon any pre-specified class of behavioral rules, like many other models in the agent-based finance literature (Chen et al, 2012). Instead, in GP, a population of behavioral rules is randomly initiated, and the survival-of-the-fittest principle drives the entire population to become fitter and fitter in relation to the environment. In other words, given the non-trivial financial incentive from trading, traders are aggressively searching for the most profitable trading rules. Therefore, the rules that are outperformed will be replaced, and only those very competitive rules will be sustained in this highly competitive search process.[7]

Hence, even though we are not informed of the behavioral rules followed by traders at any specific time horizon, GP can help us infer what these rules are approximately, by simulating how the market evolves. For instance, if a market under a certain time period is dominated by chartists, it is very likely that the generated GP trees are a type of chartist. Traders can then be clustered based on realistic, and possibly complex, behavioral rules.[8]

The GP algorithm used to infer the rules is a *simple GP*, which is based on EDDIE 7, which we presented in the previous chapter. The difference between this simple GP and EDDIE 7 is in the fitness function; the simple GP does not use constraints at all. In other words, the constraints that EDDIE uses are completely relaxed and thus set to $[C_{min}, C_{max}] = [0, 100]$ (see Section 4.7.3. The reason for doing this is because we want to test the hypothesis with a more general algorithm, rather than with a specialized one. At the end of this chapter, we also test the hypothesis with EDDIE 7 and EDDIE 8, for completion, and to investigate whether the results are independent from the choice of GP algorithm.

7.3.2 Self Organizing Maps for Clustering

Once a population of rules (GDTs) is inferred from GP, we are interested in clustering these strategies to N-types, so as to provide a concise representation of the market. The clustering takes place based on a similarity criterion, which in this work is the *observed trading behavior*.[9] Based on this criterion, two rules are similar if they are *observationally equivalent* or *similar*, or, alternatively

[7] It does not necessarily mean that all types of traders surviving must be smart and sophisticated. They can be dumb, naive, randomly behaved or zero-intelligent. Obviously, the notion of rationality or bounded rationality applying here is *ecological* (Simon, 1956; Gigerenzer and Todd, 1999) ('ecological' refers to the adaptive behavior of agents and it is used in contrast to the classical definitions of rationality, where agents conform to norms of logic, statistics and probability theory). Nevertheless, it could be argued that the above methodology does not infer the existence of dumb and naive strategies, but only evolves dumb strategies that do not necessarily represent the ones that actually existed in the market. This could thus be considered as a limitation of our suggested agent-based model.

[8] Duffy and Engle-Warnick (2002) provide the first illustration of the use of genetic programming to infer the behavioral rules of human agents in the context of ultimatum game experiments. Similarly, Izumi and Ueda (1999) use genetic algorithms to infer behavioral rules of agents from market data.

[9] Other similarity criteria could take place, too, such as risk averseness. However, it is not the purpose of this work to investigate the effect of different clustering criteria.

put, they are similar if they generate the same or similar market timing[10] behavior.

Given the criterion above, the behavior of each trading rule can be represented by its series of market timing decisions over the entire trading horizon, for example, 6 months. Therefore, if we denote the decision "buy" by "1" and "not-to-buy" by "0", then the behavior of each rule is a binary vector. The dimensionality of these vectors is then determined by the length of the trading horizon. For example, if the trading horizon is 125 days long, then the dimension of the market timing vector is 125×1. Thus, each GDT can be represented by a vector which contains a series of 1s and 0s, denoting the tree's recommendations to buy or not-buy on each day. Let us call this vector the 'behavior vector'. Once each trading rule is concretized into its behavior vector, we can then easily cluster these rules by applying Kohonen's self-organizing maps. At the end of the SOM process, trading strategies that behave in the same or in a similar way will have been clustered into the same cluster.

Figure 7.1 presents the results after running 3×3 SOM for a population of 500 individuals[11] for the daily TAIEX[12] index for the first and second half of 2007, respectively. Here, 500 artificial traders are grouped into nine clusters (types of trading strategies). In a sense, this could be perceived as a snapshot of a nine-type agent-based financial market dynamics. Traders of the same type indicate that their market timing behavior is very similar. The market fraction or the size of each cluster can be seen from the number of traders belonging to that cluster. As we can see, there are usually a few strategies that are occupying the majority of the population, whereas the rest of the strategies have significantly fewer members. For instance, we can see that in the left map, 193 trading strategies have been clustered to the bottom-right cluster, 152 strategies have been clustered into the top-left cluster, 92 into the bottom-left cluster, and so on. Similar observations can be made for the second map, on the right of Figure 7.1. Having different maps for different periods in time allows us to observe how the market fraction dynamics change from period to period, e.g., whereas a cluster which occupies a high number of trading strategies, will continue doing this in the future periods, and for how long.

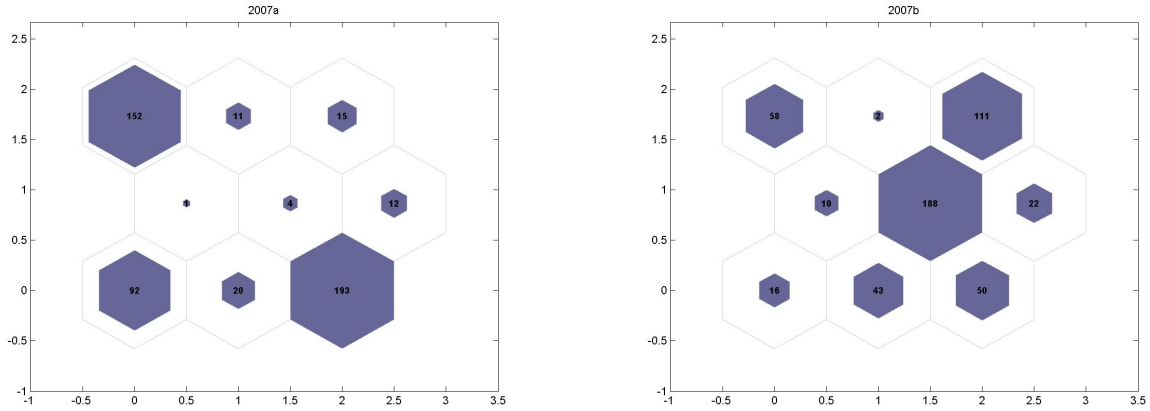
The main advantage of SOM over other clustering techniques such as K-means (MacQueen, 1967) is that the former can present the results in a visualizable manner so that we can not only identify these types of traders, but can also locate their 2-dimensional position on a map, i.e., a distribution of traders over a map. Furthermore, if we suppose that maps over time are directly comparable, we end up with a rather convenient grasp of the dynamics of the markets' structure, as if we were watching the population density on a map over time.[13]

[10] 'Market timing' refers to the strategy of making buy or sell decisions of stocks, by attempting to predict future price movements.

[11] In this thesis, we run experiments with 500 individuals. We leave it as a future work to investigate whether a different number of individuals could affect our results.

[12] Taiwan Stock Exchange Capitalization Weighted Stock Index. Available from http://finance.yahoo.com

[13] However, the assumption of directly comparable maps over time does not necessarily hold. In order to address this issue, we have introduced a technical step, which is referred to as time-invariant SOM. More details about this follow in Section 7.5.



(a) First semester of the year 2007

(b) Second semester of the year 2007

Figure 7.1: Two self-organized maps constructed from the rules inferred using the daily data of the TAIEX, the first half (the left panel) and the second half (the right panel) of 2007, respectively.

7.4 Experimental Designs

This section summarizes the experimental designs. The experiments were conducted for a period of 17 years (1991-2007) and the data were taken from the daily closing prices of 10 international market indices. These 10 markets are the CAC 40 (France), DJIA (USA), FTSE 100 (UK), HSI (Hong Kong), NASDAQ (USA), NIKEI 225 (Japan), NYSE (USA), S&P 500 (USA), STI (Singapore) and the TAIEX (Taiwan). For each of these markets, we run each experiment 10 times. To make it easier for the reader, we first present the testing methodology and results for a single run of the TAIEX dataset. Figure 7.2 presents the daily closing price of the TAIEX. We then proceed by presenting summary results over the 10 runs for all datasets.

Each year was split into 2 halves (January-June, July-December), so in total, out of the 17 years, we have 34 periods.[14] Since our data is divided into semesters, the first semester of a year is denoted with an 'a' at the end (e.g. 1991a), and the second semester of a year is denoted with a 'b' at the end (e.g. 1991b).

The GP was therefore implemented 34 times, once per each period. As we said earlier, in Section 7.3.1, this was done in order to estimate the traders' trading strategies. Thus, each period from 1991a to 2007b has a number of GDTs (500 in our experiments), which represent the traders' strategies. Table 7.1 presents the GP parameters for our experiments, which are essentially the same as the ones in the previous chapter.

After obtaining the 500 GDTs per period, we are interested in clustering them into types of

[14] At this point the length of the period was chosen arbitrarily as 6 months. We are aware that dividing the data in fixed semesters might "hide" some bias in the results, such as seasonality effect. We leave the investigation of this to future research. For instance, one potential approach of this investigation could be the use of sliding-windows.

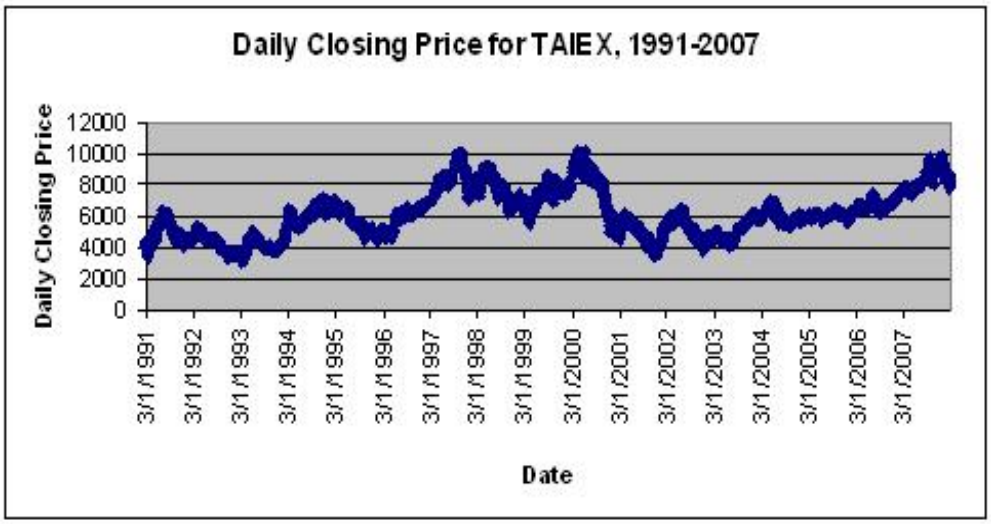


Figure 7.2: Daily Closing Price for the TAIEX:1991-2007

trading strategies, as already explained in Section 7.3.2. More specifically, after obtaining the 500 trees from 1991a, we cluster them into the 9 clusters of a SOM,[15] where each cluster represents a trading strategy type. The same procedure is followed for 1991b, 1992a, and so on. Hence, at the end of the SOM process, all trading strategies from all periods have been clustered into the 9 trading strategy types of each period. We have therefore ended up with 34 different SOMs, one per semester, which represent the market in different time periods over the 17-year horizon. We can thus observe how the proportion (market fraction) of these types of trading strategies changes over time.

Table 7.2 presents the SOM parameters for our experiments. As already mentioned in Chapter 3.7, we have used the MATLAB MathWorks Neural Network Toolbox (MathWorks, 2011), and the parameters used are the toolbox's default ones.

7.5 Testing Methodology

After having presented the necessary tools and the experimental designs, we can now proceed to present the testing methodology. Our methodology consists of three parts: GP, SOM and time-invariant SOM.

Let us start with GP. As we have already seen, we have used a *simple GP* in order to generate and evolve trading strategies. However, there is a problem with comparing trading strategies for different periods. This happens because we cannot compare the fitness function of a trading strategy (GDT) from one period with the fitness function of a strategy (GDT) for another period, since

[15] Later in the chapter we experiment with different SOM dimensions. For the moment, we investigate what happens when the number of trading strategy types is 9 (3×3). This number has been chosen arbitrarily.

Table 7.1: GP Parameters. The GP parameters for our experiments are essentially the same to the ones from Chapter 6.

GP Parameters	
Max Initial Depth	6
Max Depth	8
Generations	50
Population size	500
Tournament size	2
Reproduction probability	0.1
Crossover probability	0.9
Mutation probability	0.01
$\{w_1, w_2, w_3\}$	$\{0.6, 0.1, 0.3\}$

Table 7.2: Default SOM parameters of the MathWorks SOM Toolbox (MathWorks, 2011)

SOM Parameters	
Algorithm	Batch ($trainbuwb$)
Initialization	$midpoint$
Update rule	Batch map ($learnsomb$)
Distance	Euclidean ($dist$)
Neighborhood	Simple neighborhood set of radius $\sigma = 3$
Topology	Hexagonal ($hextop$)
Steps	100

they were presented with different datasets (environments).

This also applies to the clusters' comparison among different SOMs. *Maps over time are not directly comparable*. In order to better understand this, consider Figure 7.1. The way SOM works is that it creates the clusters after it is given a specific population of, in our cases, GDTs. When we have different periods with different populations, the nine clusters from different periods will generally be different, because they represent different populations of investment behavior, which were generated by different data environments. For example if we name the bottom-left cluster of each SOM (Figure 7.1) as 'Cluster 1', we are then saying that 'Cluster 1' of the SOM derived using the data for 2007a will in general not be the same as 'Cluster 1' of the SOM derived from the data using 2007b. It is quite likely that they will have different centroids (weighting vectors), representing different investment behaviors. This, therefore, makes the strategy types incomparable crossing different periods.

In order to tackle this problem, we introduce a *time-invariant SOM* based on the idea of *emigrating* and *reclustering*, which is the third and last part of our testing methodology. The following

section thus presents these "translations" needed in order to make SOMs from different periods comparable.

7.5.1 Translations

7.5.1.1 Emigrating

As we have just mentioned, after obtaining the trading strategies from GP, we cannot directly compare them with strategies from other periods, because each period has its own dataset. What therefore needs to be done is that all periods use the same dataset as a common base. In other words, all GDTs that are derived from each period emigrate to a common base period. For convenience, we call these *emigrant GDTs*. Therefore, after applying a GDT to the new dataset, new signals are created. In this way, the behavioral vectors of all GDTs derived from different periods are re-built based on the same grounds and hence become comparable. Here, we choose the second half of 2007 (2007b) as the base period.[16] Consider the GDTs of 1991a as an example. What we do is that we apply (emigrate) each GDT to the data of 2007b. Each tree then generates a series of new signals, under the new data (2007b). Thus, each GDT has a new behavior vector, which is comparable to the behavior vectors of the GDTs from 2007b. The same process is followed for the GDTs of periods 1991b-2007a.

7.5.1.2 Reclustering

Reclustering or time-invariant SOM is the second part of translations, which allows SOM clusters to be compared throughout different periods. We again use 2007b as the common base period. This time, we keep the centroids of the clusters originally derived from the common base period (2007b) fixed and assign the behavior vectors from other periods (emigrated GDTs) to one of the clusters of 2007b. This reclustering is conducted in the following way: the behavior vector of each emigrated GDT is compared with each centroid of the nine clusters of the 2007b map, and is then assigned to the one with the minimum Euclidean distance. We do this period by period from 1991a to 2007a. 33 SOMs are constructed in this way[17], and now these SOMs can be directly compared with each other, given that they all share the same centroids.[18] We call these SOMs 'time-invariant' SOMs.

Figure 7.3 presents 4 of these 34 SOMs, where we can examine how the fraction of the clusters changes over time. These SOMs are now directly comparable over time. For instance, we can observe that although the bottom-right cluster of the top-left map (2006a) occupied a high number of trading strategies (390), this did not continue happening in the future periods. Only 1, 6 and 50

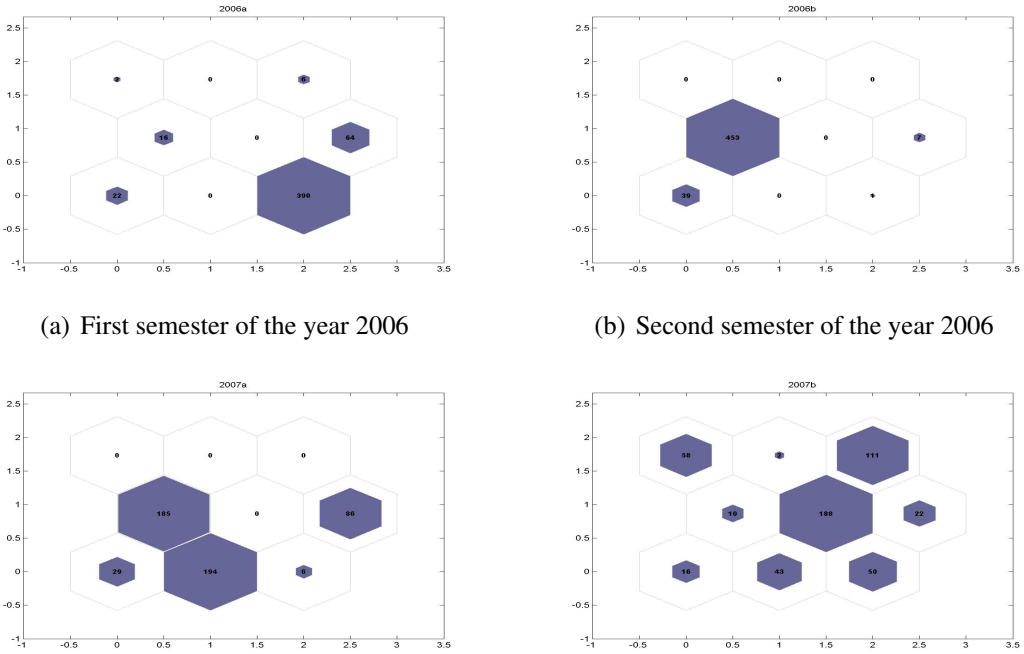
[16] The base period was chosen arbitrarily. However, which base period is chosen does not affect the results.

[17] 2007b does not need reclustering, since we use it as the base period.

[18] While this process offers the major advantage of comparing maps from different time periods, it inevitably has to assume that since after reclustering we have the same clusters over time, the strategy types behind these clusters also remain the same. This might be considered a strict assumption, but it is necessary for allowing some kind of topological equivalence. In the next chapter, we investigate the market dynamics without this assumption.

trading strategies were clustered in this cluster in periods 2006b, 2007a, and 2007b, respectively. This figure thus gives a clear picture of what we mean by market fraction dynamics. As we can observe, the distribution over the clusters is uneven over time. In each period of time, some clusters obviously dominate others, but that dominance changes over time. This can be seen from the constant renewing of the major blocks.

We will thus use all 34 SOMs that have been generated for the years 1991-2007 to test Statements 1 and 2 (see Section 7.2), and thus examine how the market fraction dynamics change in the short and in the long run.



(a) First semester of the year 2006

(b) Second semester of the year 2006

(c) First semester of the year 2007

(d) Second semester of the year 2007

Figure 7.3: SOM of Trading Strategies After "Translation": Samples from TAIEX Cells (1,1), (1,2), (2,1) and (2,2) correspond to period 2006a, 2006b, 2007a, and 2007b, respectively.

7.6 Results

7.6.1 Results of a single run of a single dataset

7.6.1.1 Test 1: The Short-Run Test

The first test regards the short-run behavior of market fractions. In the short run, the fraction of different clusters of strategies is expected to keep swinging over time, which implies a short

dominance duration for any cluster. To be operational, a type of strategy is said to be *dominant* if its fraction is greater that the threshold,

$$TH = \frac{1+p}{N+p}, \tag{7.1}$$

where TH denotes a threshold, N is the number of clusters and p is a free parameter to manipulate the degree of dominance.[19] By varying the parameter p, one can therefore have an operational meaning that is consistent with our intuition regarding dominance. For example, as $N = 9$, the threshold of being a dominant type changes with p as follows. It is 11.11% when $p = 0$, 20% when $p = 1$, and 27.27% when $p = 2$. Clearly, the higher the value of p, the higher the threshold. If all clusters were to have the same number of members, then each cluster would be occupying 11% (1/9) of the population. Hence, the case where $p = 0$ corresponds to a threshold that just breaks the tie. However, to be dominant, we may expect a value of p to be higher than just breaking the tie. Hence, in this work, p is set to be 2.

Furthermore, we need to be precise as to what we mean by short duration for a dominant type. Here, any specific number may be arbitrary; after all, short is only a matter of degree. We, therefore, first present the statistics of duration observed for each type. Figure 7.4 summarizes the dominance results over the 34 periods. It presents the minimum, average and maximum of the duration times of each type. What we can observe from Figure 7.4 is that the longest duration observed is nine periods (four and a half years) for type 6. For other types, the longest duration is barely over two periods. Hence, if we look at the average duration, with the exception of type 6, no type remains dominant for more than 2 consecutive periods, i.e., a year. Nevertheless, because of the long dominance duration of Cluster 6, we can argue that evidence for the support of Test 1 is quite weak.

7.6.1.2 Test 2: The Long-Run Test

The second hypothesis concerns the long-run behavior of market fractions. It says that, in the long run, different clusters are equally attractive and thus their market fractions are equal. As we said earlier, we expect to see that the fraction of strategies keeps changing. In the left SOM of Figure 7.1, for instance, we can see that three strategies are occupying a quite large fraction of the population (around 39%-193 members out of a total of 500, 30%-152 out of 500, and 18%-92 out of 500). The rest of the strategies have lower percentages. According to the MFH, these percentages should keep changing from period to period so that, in the long run, these percentages should be close to each other. In other words, if we have N types of traders, their long-term frequency of appearance should be close to $\frac{1}{N}$. Let $Card_{it}$ be the number (cardinality) of traders in Cluster i in time period t.

[19] We should state at this point that the terms 'dominance' and 'dominance duration' used in this thesis should not be confused with the concept of 'dominant strategy' that can be met in game theory.

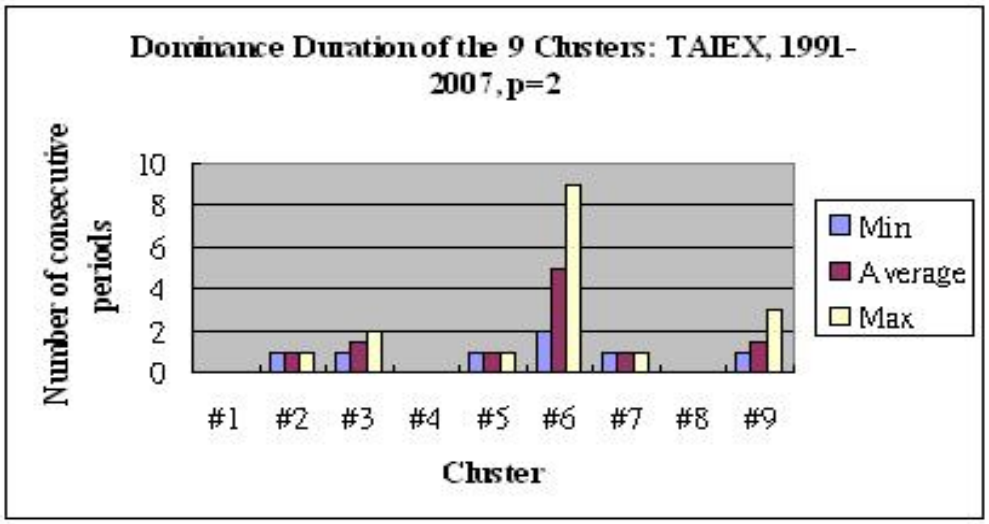


Figure 7.4: Min, average and max number of consecutive times that a strategy remains dominant over the 34 periods for p=2 Daily Closing Price for theTAIEX:1991-2007

$$\sum_{i=1}^{N} Card_{it} = M, \forall t \tag{7.2}$$

In our current setting, M, the total number of traders, is 500. The long-term histogram can be derived by simply summing up the number of traders per cluster over all periods, and dividing it by a total of $M \times T$ (# of periods),

$$w_i = \frac{\sum_{t=1}^{T} Card_{it}}{M \times T} \tag{7.3}$$

Figure 7.5 gives the long-term histogram of these clusters, $\{w_i\}$. Obviously, they are not equal and thus we present them in descending order from the left to the right. Cluster 6 has the largest market fraction of up to almost 60% , whereas Cluster 4 has the smallest market fraction, which is not even up to 1%.

Of course, it is obvious that this distribution is very different from the uniform one. In order to provide a measure of how far away it is from the uniform distribution, we use the familiar *entropy* as a metric. Let us denote the empirical distribution presented in Figure 7.5 as f_X, and the uniform distribution f_Y. By definition, $f_Y = \frac{1}{N}$, where N is the number of clusters, which in this case is 9. In order to measure how close f_X is to the uniform distribution f_Y, we calculate the entropy of both distributions. For the discrete random variable, entropy is defined as

$$Entropy = -\sum_{i=1}^{N} P_i \ln P_i \tag{7.4}$$

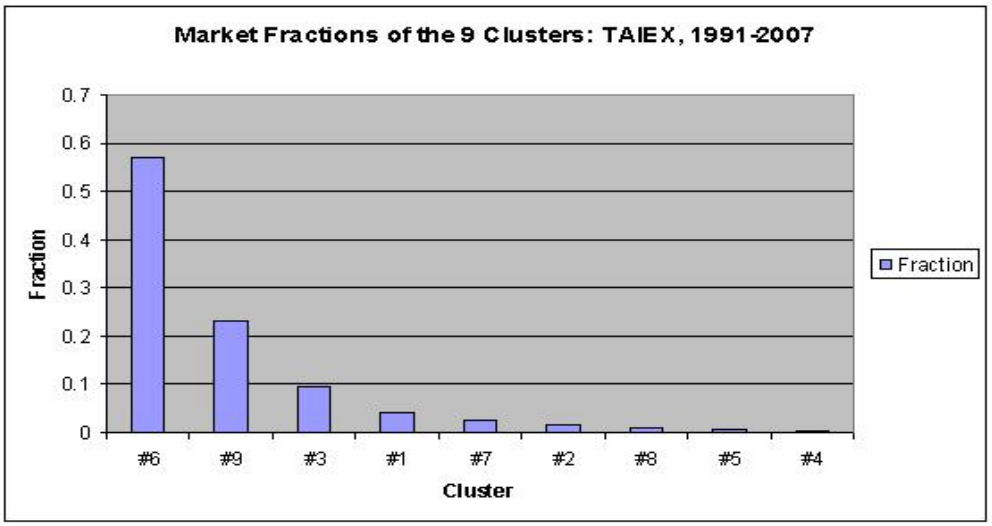


Figure 7.5: Market Fractions of the Nine Clusters: TAIEX, 1991-2007

where P_i is the fraction of each cluster. It is well known that for the uniform distribution $Entropy(Y) = \ln N$. When $N = 9$, it is $\ln 9 \approx 2.2$. The closer $Entropy(X)$ is to 2.2, the closer X is to the uniform distribution. After calculating X's entropy, we find it equal to 1.3, which is only 59% of the entropy of the uniform distribution. This thus allows us to argue that f_X is far away from the uniform distribution, and hence Test 2 is not supported by TAIEX.

Now that we have seen the test results of a single run for one dataset, it is interesting to see if these results can be generalized for more runs and more datasets. The next part of this section presents and discusses these summary results.

7.6.2 Summary results for all datasets under 9 clusters (3×3 SOM)

As we saw in the previous section, the experimental results of the two tests seem to deviate from what the MFH predicts to some extent. Test 1 has one cluster that dominates the market for 9 consecutive periods, which appears to be too long. In addition, Test 2 shows an even larger deviation, since the long term market fraction is very different from the uniform distribution. Altogether, the evidence for the MFH is weak. However, so far we have only presented a single run, for a single dataset. Table 7.3 thus presents the results over 10 runs, for all the datasets tested. The first two numeric columns are related with Test 1. They present the averages over the 10 runs for the average and maximum dominance duration of the 9 clusters. Furthermore, the last column is related to Test 2 and shows the ratio of the average realized entropy (over the 10 runs) over the base entropy (equal to 2.2).

The first observation we can draw from Table 7.3 is that homogeneity exists across the majority of the results. Let us first start with Test 1. We can see that on average there is no cluster that remains dominant for 2 consecutive periods. This is in line with Test 1. However, the second

Table 7.3: Summary results over 10 runs, for all datasets, for 3×3 SOM. The first two numeric columns are related with Test 1 and present the averages over the 10 runs for the average and maximum dominance durations of the 9 clusters, respectively. The last column presents the ratio of the average realized entropy (over the 10 runs) over the base entropy (Test 2). This ratio is maximized when $\frac{RealizedEntropy}{BaseEntropy} = 1$.

Summary Statistics			
	Test 1		Test 2
	Average	Max	Entropy Ratio
CAC 40	1.81	5.5	0.68
DJIA	1.93	5.78	0.66
FTSE 100	1.77	6	0.64
HSI	1.71	4.6	0.7
NASDAQ	1.59	4.11	0.69
NIKEI 225	1.51	3.4	0.79
NYSE	1.93	6.56	0.6
S&P 500	2.16	6.89	0.64
STI	1.67	3.7	0.75
TAIEX	2.02	8.25	0.55

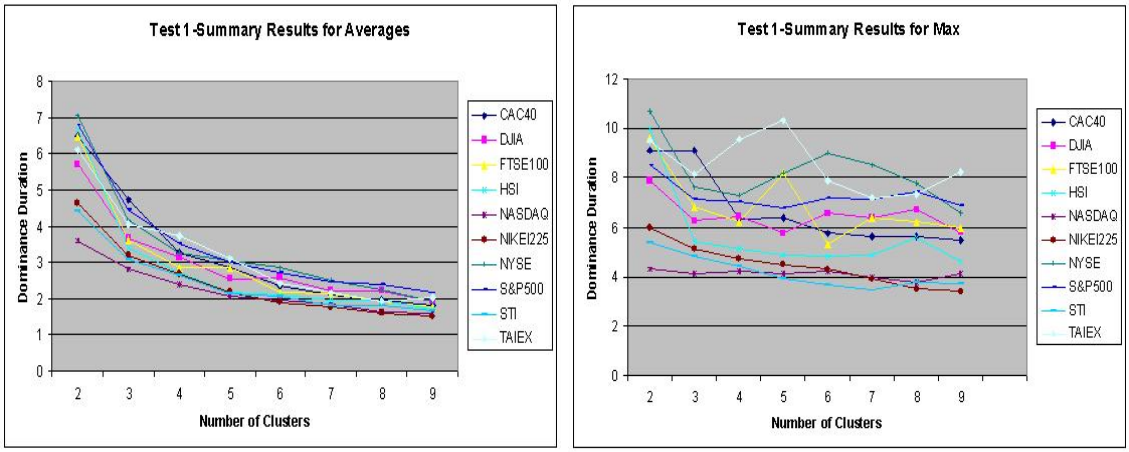
column tells us that even though on average no cluster dominates for more than 2 periods, there is always an outlier that can remain dominant for longer, e.g. 8 consecutive periods for the TAIEX. Thus Test 1 is not supported by any of the 10 datasets tested, under the 3×3 SOM. Regarding Test 2, the entropy ratios for all datasets are somewhat distant from their maximum values. All entropy ratios are in the range 0.55-0.79, which basically is a 21-45% difference from the entropy of the uniform distribution. This essentially means that the distributions are on average different from the uniform and therefore the clusters, in the long run, are not equally attractive, as Test 2 requires. *Overall, the MFH seems to be relatively weak for all 10 indices tested under the* 3×3 *SOM.*

7.6.3 Changing the number of clusters

So far, all of our tests have been performed for 3×3 SOMs. It is therefore interesting to investigate how sensitive are the results if we tune the number of clusters. Therefore, we repeat the whole procedure mentioned above for different SOM dimensions: 2×1, 3×1, 2×2, 5×1, 3×2, 7×1 and 4×2.

7.6.3.1 Test 1 under all SOM dimensions

Figure 7.6 presents the averages, over 10 runs, for the average and maximum dominance duration for number of clusters 2-9. The x-axis presents the number of types of trading strategies (clusters),



(a) Test 1-Averages (b) Test 1-Maximums

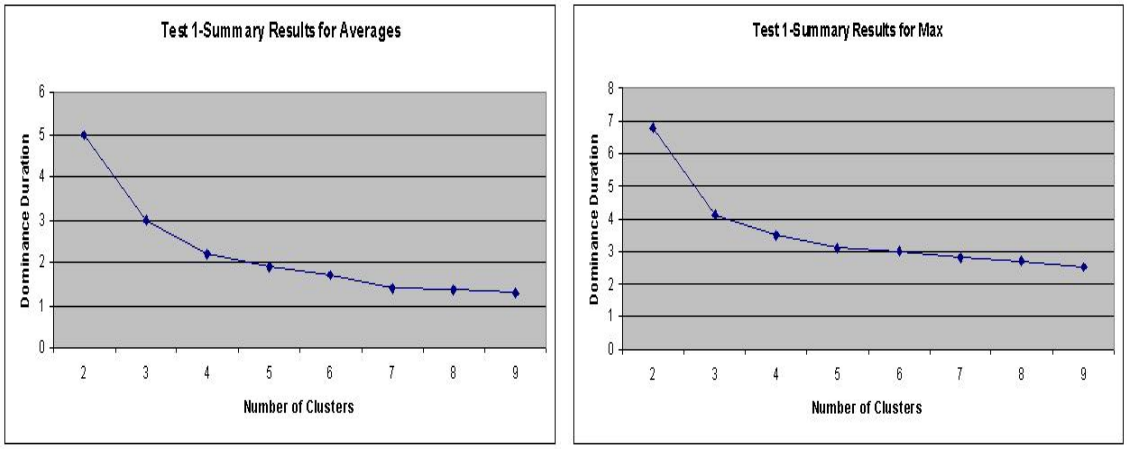
Figure 7.6: Summary Results for Test 1-Averages and Test 1-Maximums. The x-axis presents the number of types of trading strategies (clusters), and the y-axis the dominance duration.

and the y-axis the dominance duration. What we observe in these graphs, especially in Figure 7.6(a), is that the dominance duration decreases as the number of clusters increases. Nevertheless, we can again see from Figure 7.6(b) that there are always clusters with strong dominance, even under the 3×3 SOM. In addition, standard deviations results for Test 1-Max (Table E.1) are relatively low, and thus Test 1-Max results are reliable. *Test 1 thus is not supported under any number of clusters*.

To see how significant the above patterns are, we run a Monte Carlo simulation as follows. Starting with two clusters, we randomly assign a winner (dominant cluster) to either cluster 1 or cluster 2. We then conduct this binomial experiment 34 times. Considering this to be one run, we do it for 10 runs. Hence, we have 10 artificial series of dominant clusters, with each series lasting for 34 runs. We then conduct the same analysis as above by figuring out the average duration and maximum duration of each series, and the average of the whole. We then apply this Monte Carlo experiment for 2 to 9 clusters. A comparable result is then drawn in Figure 7.7.

By comparing Figure 7.6 with Figure 7.7, we can see that the behavior of the real markets is very different from that of the multinomial experiment. For the latter, the average of the maximum duration (Figure 7.7(b)) decays, from above 7 to slightly above 3, but for the former (Figure 7.6(b)) this decaying tendency is shown in none of the ten markets. Instead, they all fluctuate slightly around a horizontal line, and, depending on the market, the line is situated at a interval from four to eight. For the average duration (Figures 7.6(a) and 7.7(a)), while both figures feature a decaying tendency, the one with financial data decays much more slowly than the one based on the artificial data. Therefore, our result cannot be treated as an incident from a random draw of the multinomial experiments, and in this sense this pattern is not spurious.

One last thing that we would like to comment is the low average dominance duration that can



(a) Test 1-Averages (b) Test 1-Maximums

Figure 7.7: Summary Results for Test 1-Averages and Test 1-Maximums under Monte Carlo simulation

be observed in Figure 7.6(a), for the high number of clusters. One might wonder what the reason for this 'phenomenon' is, especially since the average dominance duration started at quite high levels (for the low number of clusters). We believe that this can be better explained if we also take in to account Figure 7.6(b). What seems to happen for all datasets is that there are always a few clusters that have strong (long) dominance over the 34 periods, whereas the rest have very low dominance. The low average dominance duration we see in Figure 7.6(a) for the high number of clusters can therefore be explained by the extremely low dominance duration of the majority of clusters.[20]

7.6.3.2 Test 2 under all SOM dimensions

As we said earlier, we are interested in obtaining the distance of the entropy of the empirical distribution f_X (fractions of clusters) from the uniform distribution (benchmark). We have also said that the closer the entropy of distribution f_X is to the entropy of the uniform distribution, the closer distribution f_X is to the uniform one. After obtaining the entropies over 10 runs for each dataset, we first calculated the average of these runs, we then divided each one of these averages with the benchmark entropy and thus obtained 10 different ratios (one per dataset). Of course, this ratio is maximized when the two entropies are equal, and therefore their ratio is equal to 1. Hence, the higher the ratio, the closer to the uniform distribution the empirical distribution will be. Figure 7.8 presents these ratios for all datasets.

HSI and NASDAQ seem to be the only datasets that are close to the uniform distribution, with

[20]For instance, if 8 out of the 9 clusters have a dominance duration of 2 periods, and only 1 cluster has a dominance duration of 9 periods, then the average dominance duration is driven down to $\frac{(8*2)+9}{9} \approx 2.7$ periods.

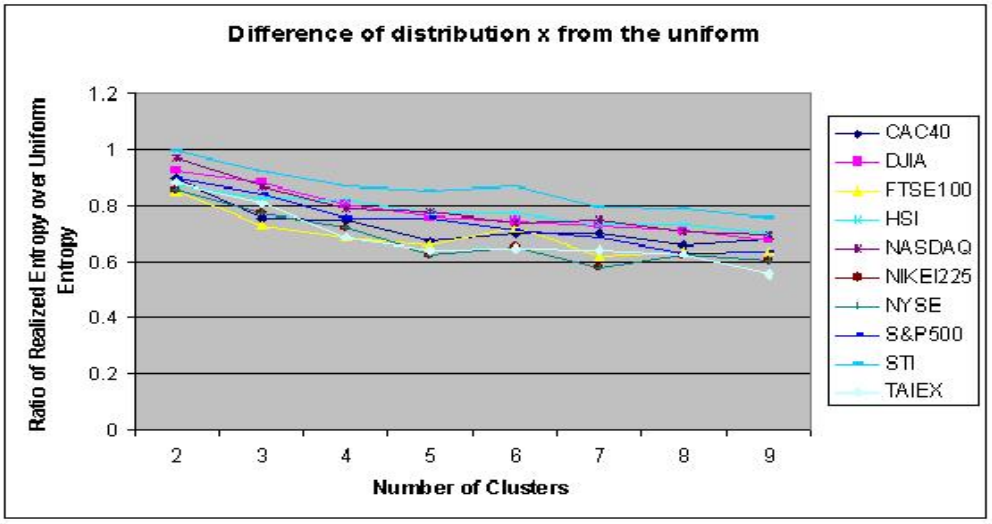


Figure 7.8: Test 2: Difference of the empirical distribution x (fractions of clusters) from the uniform distribution.

a ration of 0.99 and 0.97, respectively; this happens only under the 2×1 SOM. In addition, we can observe that the ratios tend to decrease as the number of clusters increases, and hence the support for Test 2 gets weaker, for all datasets. Such a divergence of the two distributions indicates again that the strong dominance of a few clusters continues to exist, even in the long run. Hence, *evidence for the support of Test 2 is quite weak*.[21] Therefore, after combining Tests 1 and 2, we can have a quite clear picture. *Clusters tend to dominate the markets for long periods of time.*

To make this argument even clearer, we also present Figure 7.9, which shows the cumulative fractions for the TAIEX for different number of clusters. A graph named 'Number of Clusters: 2' means that the strategies are allocated to 2 clusters. When 'Number of Clusters: 3', the strategies are allocated 3 clusters, and so on. The clusters have been sorted by their size (fraction) in descending order. Therefore, Cluster 1 on the x-axis denotes the cluster with the highest fraction, Cluster 2 the cluster with the second highest fraction, etc. The y-axis presents the cumulative fraction of the clusters.

An observation we can make from Figure 7.9 is that the contribution of each ranked cluster decreases when the number of clusters increases, which causes the entire cumulative curve to shift down.[22] For instance, when the number of clusters is 2, the largest cluster has a size of about 70%. However, while we move to a higher number of clusters, we can see that this size gradually decreases and finally falls below 60%, when the number of clusters is 9. The same happens for the contribution of the rest of the clusters. Hence, each graph moves a bit below, when the number of clusters increases. Nevertheless, what is important is that even when the number of clusters is 9,

[21] This is also supported by the standard deviation results presented in Table E.2, where we can see that these values are quite low.

[22] The graphs for the other markets can be found in Appendix C.

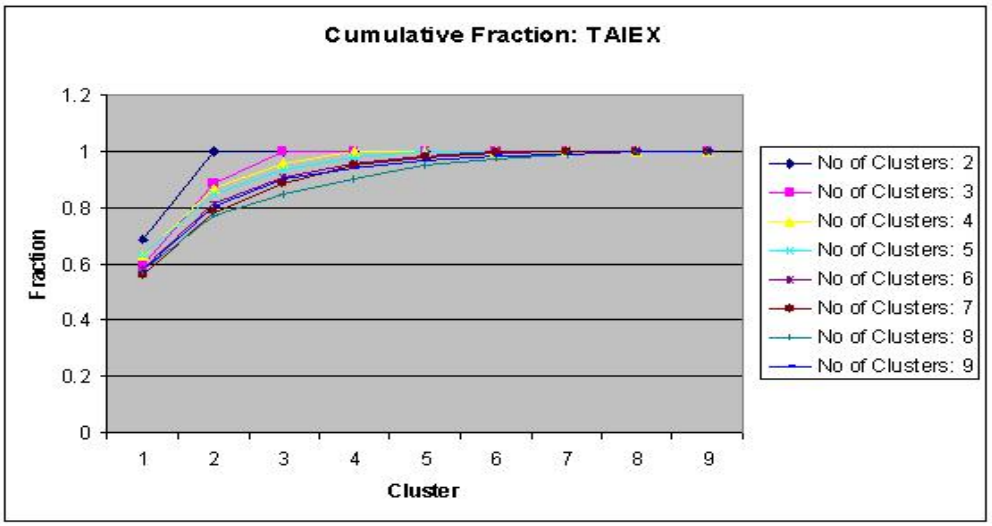


Figure 7.9: Cumulative Fraction for TAIEX

the clusters with the highest two or three ranks occupy a very large fraction (approximately 90%) of market participants.

The above observation leads us to consider that maybe there is a minimum number of clusters that covers a certain fraction of the market. Table 7.4 gives the result of the minimum number of clusters required to cover a targeted fraction of market participants. The three targeted values given in the table are 90%, 95% and 99%. Since the purpose was to see whether only a small number of clusters is required, we started with a larger number clusters, namely, nine, and saw how much reduction we could make. If the target is to cover 90% of the market participants, then most markets need only four to five types, and if the target rises even higher to 95%, then most markets need five to six types.[23] We can thus conclude from this that a few types of strategies are enough to characterize the market behavior.

7.7 Testing the MFH under different GP algorithms

In this section, we test the MFH again, but this time with EDDIE 7 and EDDIE 8 as the rule inference engines, instead of the simple GP. The reason we do this is because we are interested in seeing if the previous results can hold under different GP algorithms, which would indicate that they are algorithm-independent. This would thus allow us to generalize the results.

[23]These results are similar to another work by Aoki (2002).This is probably the only paper known to us that deals with a number of types of agents in the multi-agents system. In his work, Aoki determined the minimum number of types of behavior required to capture multi-agent economic systems. He showed that it would be enough to characterize the 95% of the market behavior only by a few types, say, two to three. The remaining types were rather marginal.

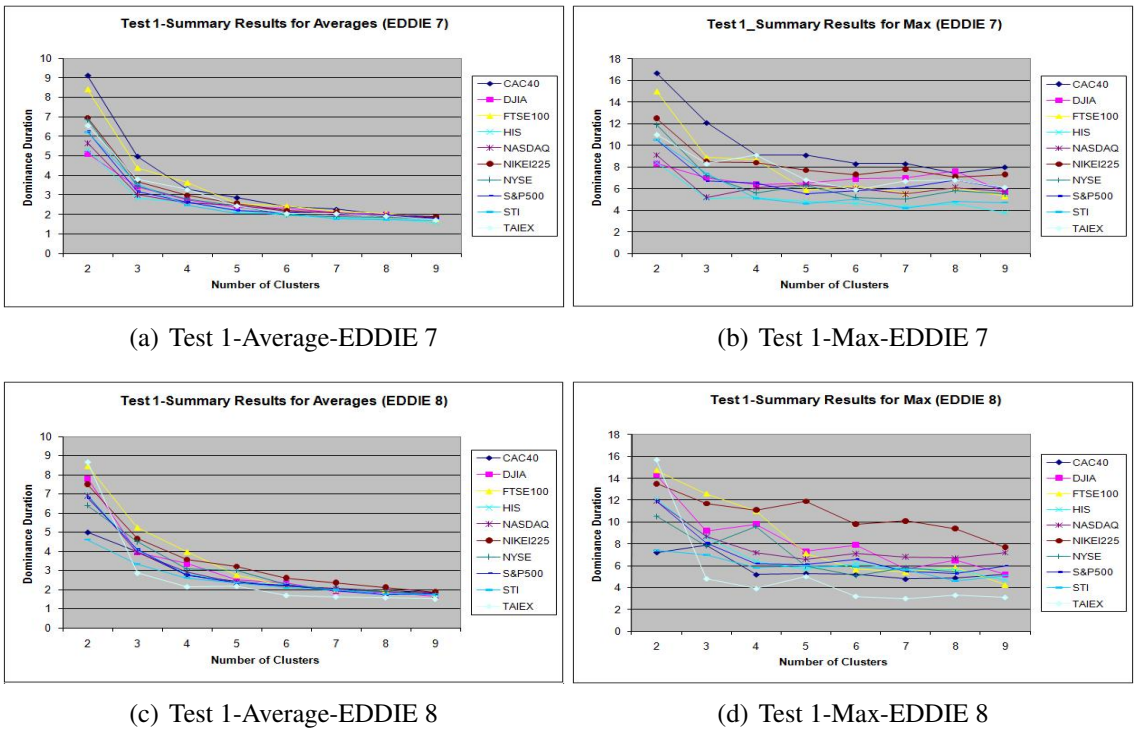
Table 7.4: Minimum number of clusters whose cumulative fraction is above the required threshold of 90%, 95% and 99%, respectively

Minimum Number of Clusters			
	Threshold		
	90%	95%	99%
CAC 40	4	5	7
DJIA	5	6	8
FTSE 100	4	5	8
HSI	5	6	8
NASDAQ	4	6	8
NIKEI 225	5	6	8
NYSE	7	8	9
S&P 500	7	8	9
STI	5	6	8
TAIEX	4	6	7

7.7.1 Test 1 under EDDIE 7 and EDDIE 8

We set again the parameter that manipulates the degree of dominance duration to $p = 2$, as earlier. Figure 7.10 presents the Test 1 results for Averages (left) and Maximums (right), for both EDDIE 7 (up) and EDDIE 8 (down). What we observe in these graphs is that again the dominance duration decreases as the number of clusters increases, for both EDDIE 7 (top) and EDDIE 8 (bottom). This suggests that when we move from a lower to a higher number of clusters, the empirical results reveal stronger evidence of the MFH. Of course, the fact remains that both maximum duration graphs suggest that there is always a cluster that remains dominant for a number of periods that is greater than the average. Therefore, results under EDDIE 7 and EDDIE 8 suggest that there are always a few clusters that have strong dominance over the 34 periods, whereas for the rest the dominance of duration is very low. Test 1 is hence not justified.

These observations are in line with the earlier ones. However, it should be stated that the GP algorithms in this section seem to have slightly worsened the results. Especially in the case of the averages of maximum duration, both the EDDIE 7 and EDDIE 8 results reveal a significant increase on the duration period. Earlier, the results started in the range of 4-11 periods for the 2×1 SOM, for all datasets, and finished in the range of 3-8 periods, under the 3×3 SOM. However, the results in this section start in the range of 8-17 (EDDIE 7) and 7-15 (EDDIE 8) periods, and finish in the range of 4-9 (EDDIE 7) and 3-8 (EDDIE 8). What we can thus conclude from this test is that both versions of EDDIE have increased the dominance duration of the few clusters with strong dominance.



(a) Test 1-Average-EDDIE 7

(b) Test 1-Max-EDDIE 7

(c) Test 1-Average-EDDIE 8

(d) Test 1-Max-EDDIE 8

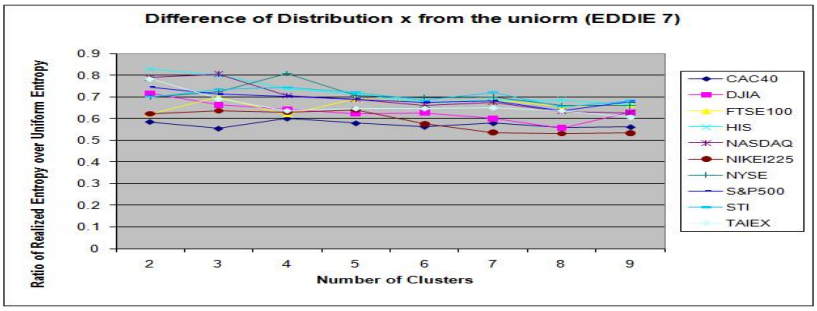
Figure 7.10: Test 1: Summary Results for Averages (left) and Maximums (right). The x-axis presents the summary results under different numbers of clusters. The graphs at the top present the EDDIE 7 results, whereas the ones at the bottom present the EDDIE 8 results.

7.7.2 Test 2 under EDDIE 7 and EDDIE 8

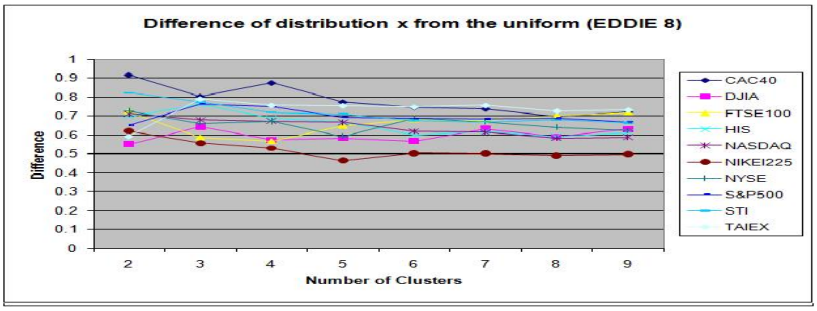
Figure 7.11 presents the entropy ratios for all datasets. What we observe from this figure is that the ratios tend to decrease as the number of clusters increases (thus, the difference between the two distributions increases). This observation holds for both EDDIE 7 and 8. Furthermore, when only a few number of strategy types exist, we get closer to the uniform distribution, which as we know implies that in the long run the clusters are equally competitive.

Such a high diversity of the two distributions (especially with a high number of clusters) indicates again that the strong dominance of a few clusters continues to exist, even in the long run. Therefore, Test 2 is again not justified under neither EDDIE 7, nor EDDIE 8.

These observations are also very similar to the original ones. However, once again the results under EDDIE 7 and EDDIE 8 seem to have slightly deteriorated. Under the simple GP, the above ratios for low SOM dimensions (e.g. 2×1) were in the range of 0.82-0.99, for all datasets. Hence, some of the empirical distributions were very close to the uniform distribution, which of course indicates support for Test 2. On the other hand, this does not happen here, where the difference between f_X and f_Y is quite big for both GPs, with their maximum ratios not exceeding 0.83



(a) Test 2-EDDIE 7



(b) Test 2-EDDIE 7

Figure 7.11: Test 2: Difference of distribution f_X from the uniform distribution for EDDIE 7 (top) and EDDIE 8 (bottom). The x-axis presents the results under different numbers of clusters.

(EDDIE 7), and 0.91 (EDDIE 8), respectively. Thus there is no support for Test 2 under EDDIE 7 and EDDIE 8.

Finally, we present Table 7.5, which shows the minimum number of clusters required to pass the 90, 95, and 99% fraction thresholds, under under EDDIE 7 and EDDIE 8. Results are again very similar as earlier, in Table 7.4. In order to cover 90% of the market participants, all markets need four to five types; if the target is to cover 95%, then markets need five to six types, for both EDDIE 7 and 8.

7.8 Conclusion

To summarize, this chapter presented the Market Fraction Hypothesis (MFH), which states that the fraction of the types of trading strategies that exist in a financial market changes over time. This is a very interesting statement, because it suggests that popularity can be interchanged among all available trading strategy types, and thus a 'winner' type of trading strategy cannot exist. As a result, it might be futile to attempt to identify successful forecasting opportunities, since no rule can be a 'winner' in the long run. In addition, this statement motivated us to investigate the underlying dynamics of the different types of trading strategies that exist in financial market. However, this

Table 7.5: Minimum number of clusters whose cumulative fraction is above the required threshold of 90%, 95% and 99%, respectively. Results are under the EDDIE 7 and EDDIE 8 algorithms.

		CA40	DJIA	FTSE100	HSI	NASDAQ	NIKEI	NYSE	S&P	STI	TAIEX
EDDIE 7	90%	4	4	4	4	4	4	5	5	5	4
	95%	5	5	5	5	5	5	6	6	6	5
	99%	7	7	8	7	7	7	8	8	8	8
EDDIE 8	90%	5	5	5	4	4	3	4	5	4	5
	95%	6	6	6	5	5	5	5	6	5	6
	99%	8	8	8	7	7	7	7	7	7	8

hypothesis had never been formalized in the past. Our first contribution was thus to formalize the hypothesis. This then allowed us to suggest a testing methodology and test the statements of the hypothesis under 10 real markets. The latter was very important, because until now the majority of the observations of the MFH had only been made under artificial market frameworks. Lastly, in order to test the hypothesis we proposed a new agent-based financial model. The novelty of this model was twofold: first, it did not assume pre-fixed types of trading strategies, like the typical N-type models do (Chen et al, 2012). In addition, our model allowed the strategies that belonged in the same type to be heterogeneous, which was also something that was not happening in the N-type models.

The results showed that the hypothesis cannot always be supported and can sometimes be affected by the number of clusters (trading strategy types). In fact, the MFH seems to be weak for the majority of the datasets we tested. More specifically, we found that, even in the long run, the market tends to favor few types of agents, hence *the property of the long-term uniform distribution (Test 2) does not hold*. The only exception to this was when there were only 2 types of trading strategies in the markets (2×1 SOM). In addition, we found that we only need four to five (five to six) types, to account for the behavior of 90% (95%) of market participants. Therefore, having more types of strategies in the market does not change much; the traders tend to focus to only a few of these strategies. Finally, while most types of agents cannot be dominant consecutively for more than 2 years, few exceptions can sustain up to 5.5 years (TAIEX maximum duration: 11 consecutive periods). Therefore, *the property of short-term dominance duration (Test 1) also does not hold*.

The above observations have led us to valuable conclusions about market fraction dynamics. From our experimental results we can argue that popular (i.e., dominant) types of trading strategies can remain popular over long time periods. Hence, winners *can* exist. In addition, there is no swinging among the fractions of the different types of trading strategies. Thus, even if many strategy types exist in a market, only a few of them become and remain popular over time. *It is therefore enough to characterize the market behavior by using a few types of trading strategies, even if many more types exist in this market.*

We also used two different GP algorithms, EDDIE 7 and EDDIE 8, to test the previous derived results of the Market Fraction Hypothesis. We were interested in observing whether our previous

results were sensitive to the choice of GP and to what extent. In general, we can say that the same patterns were observed for all 2 tests, under EDDIE 7, EDDIE 8, and *simple GP*. Thus, we can argue that our arguments and conclusions regarding the Market Fraction Hypothesis are rigorous. Nevertheless, the introduction of both EDDIE versions has shown a minor deterioration in the individual test results. Thus, evidence for the MFH has got even weaker. This becomes especially true for low number of clusters. For instance, under the *simple GP* tests, there were a few datasets that would show support for Test 2 under a very low SOM dimension, e.g. 2×1. However, this does not happen any more, with any of the EDDIE versions. There is even less support for the MFH under EDDIE 7 and 8. Our explanation for this is that the constrained fitness function that both versions share is responsible for this, although we cannot yet explain how. We leave this as future research. More specifically, if we completely loosen the constraints in the fitness function of EDDIE 7, we will end up with the *simple GP* algorithm. We can thus start from there and then fine-tune the constraint parameters, so that we can experiment with looser and tighter constraints. This should allow us to observe how the MFH results are affected by the constrained fitness function. We believe that this will give us a better idea of why the MFH results under EDDIE are different than the original ones. Finally, as there are no significant differences between EDDIE 7 and EDDIE 8, we have concluded that EDDIE 8's new grammar is not responsible for any more changes in the MFH results.

Future research will include some changes in our model. At the moment, we have used GP and SOM as our two main tools. A question that arises is whether our results can be affected by different choices of GP and SOM algorithms. Also, it would be interesting to investigate whether our results would stand under a different framework, where different rule-inference machines and clustering techniques would be used. For example, Genetic Algorithms could be used instead of GP. In addition, standard hierarchical clustering (Xu and Wunsch, 2008) or growing hierarchical self-organizing maps (Dittenbach et al, 2001) could be used instead of SOM. Also, individual parameters of the algorithms we have used in this work, like the choice of fitness function or the crossover rate, could affect the tests' results. Lastly, one aspect that we consider as an interesting future work is to examine and understand the underlying trading strategy types behind the SOM clusters. This could give valuable insight for the markets and how they operate. For instance, if we knew which is the underlying strategy type behind a cluster that had remained dominant for 2 consecutive years, we might have been able to explain what were the reasons that allowed this strategy to remain popular. This could also lead to understand what makes strategies popular in the first place.

Furthermore, we should mention that despite the advantages of our new agent-based model that were presented in this chapter, there are some limitations, as well. One of these limitations is that our model has departed from the 'traditional' approach of estimating agent-based models. Such models follow the approach of generating trading strategies, which they then use to generate artificial datasets and then calibrate the vector of parameters of the above strategies, so that the datasets' statistical properties resemble the properties of real financial markets. However, because of limitations that exist in these models (see Chapters 5 and 7), we chose to depart from this approach and use Genetic Programming to infer the strategies that existed in the real financial

markets. Nevertheless, it could be argued that this approach does not guarantee that the inferred strategies are a close approximation of the reality. These strategies were inferred based on the aggregate behavior of the real markets, which was the price movements, and not on individual strategies. It would thus be interesting to investigate, as a future work, how close the inferred strategies are to the real strategies.

Finally, while the introduction of time-invariant SOM has offered an important advantage to our work (i.e., allowing the comparison of SOMs among different time periods), it has also introduced a drawback. In order to make these comparisons, we had to assume that trading strategy types remain the same over time. This could be considered as a strict assumption, because this does not necessarily happen in real markets. In the next chapter, we do not need to make any more comparisons of clusters from different periods, and thus relax the above assumption. We consider that this will add more realism to our financial model.

Chapter 8

The Dinosaur Hypothesis

8.1 Introduction

In the previous chapter, we presented an agent-based financial model, where we developed an approach to empirically examine the dynamics of market fractions of different types of trading strategies. That study used GP as a rule-inference engine to find out the behavioral rules of these strategies, and then used SOM to cluster the strategies into types of trading strategies. Once after the clusters and the associated map were obtained from period to period, the dynamics of market fractions were studied based on a time series of these maps. More specifically, we studied two main properties motivated by the Market Fraction Hypothesis: the short-term horizon and the long-term frequency.

However, that study rested upon an important assumption, i.e., the maps derived from each period were comparable with each other. This comparability assumption itself required that the clusters, as well as their operational specification, would not change over time. If this were not the case, then the subsequent study would be questioned. This was mainly due to the technical step in our analysis called translation. The purpose of translation was to place the behavior of agents observed in one period into a different period and to recluster it for the further cross-period comparison. We could not meaningfully have done this without something like topological equivalence, which could not be sustained without the constancy of the types. However, this assumption can be considered as a strict one. Strategy types do not necessarily remain the same over time. For instance, if a chartist strategy type exists in time t, it is not certain it will also exist in $t + 1$. If market conditions change dramatically, the agents might consider other strategy types as more effective and choose them. The chartist strategy would then stop existing.

In this chapter, we are interested in extending our previous work by investigating the market structure when both the types and the fractions of strategies are subject to change. Thus, we again employ GP as a rule inference engine and SOM as a clustering machine, with the only difference being that we relax the above assumption of static clusters over time. This makes our model more *realistic* and also allows us to test for the existence of the constantly-changing property in the

behavior of financial markets that Arthur (1992), and Chen and Yeh (2001) observed, or in other words what Chen first called the *Dinosaur Hypothesis (DH)* (Chen, 2008).

More specifically, Arthur (1992), and Chen and Yeh (2001) argued that the behavior of financial markets constantly changes and as a result, the trading strategies of financial agents need to continuously co-evolve with the market in order to remain successful. In addition, both Arthur, Chen and Yeh showed that strategies that did not adapt to the changes happening in the market became obsolete, or what Arthur called 'dinosaurs'. Finally, Chen and Yeh also showed that dinosaurs' performance deteriorates monotonically with time.

These are very interesting observations, because they give us valuable information about the behavior of financial markets. In addition, these observations tell us that there is a finite lifetime for trading strategies that do not adapt to the changes happening in the market. However, as Arthur's, Chen and Yeh's observations took place under artificial stock market frameworks, we are interested in *investigating whether these observations can also hold in the 'real' world, under empirical datasets*.

Furthermore, we are interested in formalizing the DH, by presenting its main constituents. As we have already mentioned, although the term 'DH' has been used before in the literature (Chen, 2008), it has never been formalized as a concrete hypothesis. Doing this allows us to suggest a testing methodology and test its plausibility in the real markets. We then run tests under 10 international markets and hence provide a general examination of the plausibility of the DH. *More importantly, these tests will allow us to observe the dynamics of market behavior and investigate how they change in the long run*.

The main contributions that will be presented in this chapter can thus be summarized as follows: (i) Formalizing the observations of Arthur, Chen and Yeh into a concrete hypothesis, the DH, (ii) Suggesting a testing methodology, (iii) Extending the agent-based financial model presented in Chapter 7 by allowing trading strategy types to change over time, and (iv) Testing the hypothesis under 10 international markets.

The rest of the chapter is organized as follows: Section 8.2 elaborates on the DH. Section 8.3 presents the experimental designs. Section 8.4 addresses the methodology employed to test the DH and explains the technical approaches needed to be taken for facilitating the test of the DH. Section 8.5 presents the test results. First it starts by presenting the results over a single run for a single dataset. Then it continues by presenting the summary results over 10 runs for this dataset and it finally presents summary results for all datasets. Section 8.6 then tests the hypothesis under EDDIE 7 and EDDIE 8, in order to investigate whether the previously derived results are dependent on the choice of the GP algorithm. Finally, Section 8.7 concludes this chapter and briefly discusses possible directions for further research.

8.2 The Dinosaur Hypothesis

The following statements form the basic constituents of the DH and are based on Arthur, Chen and Yeh's work:

1. The market behavior never settles down. The population of trading strategies that exists in the market continuously co-evolves with it.

2. If these trading strategies do not co-evolve with the market, their performance deteriorates over time.

These two statements indicate the non-stationary nature of financial markets and imply that strategies need to evolve and adapt to the changes in these markets, in order to survive. If they do not co-evolve[1] with the market, their performance[2] deteriorates and they become ineffective.[3]

As we mentioned earlier, Arthur's, Chen and Yeh's observations were made in an artificial stock market framework. What we shall thus do in this work is to test the above statements against empirical data. Section 8.4 presents the two tests derived from the Statements 1 and 2.

Finally, in order to make the reading of this chapter more comprehensive, we present two definitions, inspired by Arthur's work: *Dinosaur*, is a trading strategy which has performed well in some periods, but then ceased performing well in the periods that followed. This means that this strategy may or may not become effective again. If it does, then it is called a *returning dinosaur*.

8.3 Experimental Designs

The tools used for the experiments in this chapter are again the simple GP we used in Chapter 7 and SOM. For simplicity, we have kept the same parameters and setup for both tools.

The experiments were again conducted for a period of 17 years (1991-2007) and the data was taken from the daily closing prices of the same 10 international market indices. These 10 markets are: CAC 40 (France), DJIA (USA), FTSE 100 (UK), HSI (Hong Kong), NASDAQ (USA), NIKEI 225 (Japan), NYSE (USA), S&P 500 (USA), STI (Singapore) and TAIEX (Taiwan). For each of these markets, we run each experiment for 10 times. To make it easier to the reader, we first present the testing methodology and results for a single run of the STI dataset. Figure 8.1 presents the daily closing price of STI. We then proceed with presenting summary results over the 10 runs for all datasets.

[1] The meaning of co-evolution here refers to the idea that trading strategies need to follow the changes that occur in a financial market. For instance, if a market that was earlier dominated by fundamentalists is now dominated by chartists, a trading strategy should adopt to this change in order to remain successful.

[2] Performance can be measured in different ways. In this work, we do this by measuring how well old, unadapted strategies fit in future market environments. We call this measure 'dissatisfaction'. More details about this are provided later in Section 8.4

[3] At this point, we have neglected the fact that Chen and Yeh specifically state that this performance deterioration is monotonic. Later, in Section 8.4, we explain why we have chosen to relax this statement.

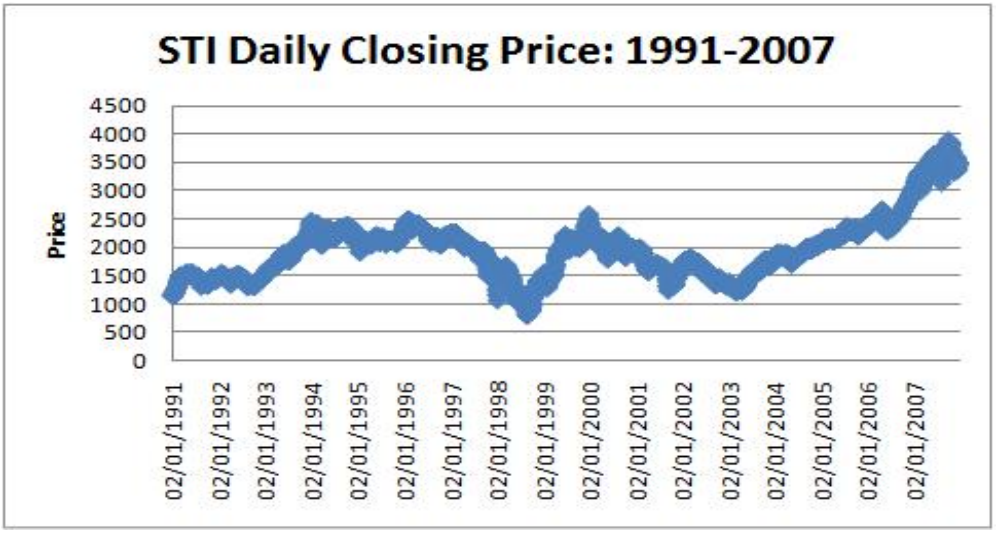


Figure 8.1: Daily Closing Price for STI:1991-2007

8.4 Testing Methodology

In this section we discuss the testing methodology. But before we do this, let us first present some frequently used terms:

- *Base period*, is the period during which we create and evolve GDTs that are going to be used for testing the DH
- *Future period(s)*, is a period or periods which follow the base period (in chronological order)

Furthermore, our data is again divided into semesters. Thus the first semester of a year will again be denoted with an 'a' at the end (e.g. 1991a), and the second semester of a year will be denoted with a 'b' at the end (e.g. 1991b). Let us now present the tests and our suggested testing methodology.

8.4.1 Dissatisfaction Test

In order to investigate whether the behavior of markets is non-stationary, we do the following. After generating and evolving GDTs for each one of the 34 periods, and clustering them into strategy types via SOM, we recluster the GDTs of each base period to the SOM of their future periods.[4] If the Dinosaur Hypothesis holds, we should observe that GDTs (strategies) from past periods have more and more difficulties fitting into the environments of the future periods. This is because these strategies have remained unadapted to the new environments and have thus turned into dinosaurs.

[4]The process of reclustering is explained in the next section.

Let us give an example, and assume that there is a strategy type (cluster) in time t, which represents 'fundamentalists'; then, all GDTs which follow a fundamental strategy are placed in that cluster. When we then take the GDTs from that period (base period) and recluster them to strategy types of future periods, it is not guaranteed that there will again be a cluster that represents fundamentalists. If market behavior constantly changes, there is a possibility that this type of strategies does not exist any more. Thus, the GDTs find themselves *unadapted* to the new environment (clusters) and have to choose another cluster, which represents them as closely as possible. This cluster will be the one that has the centroid with the smallest Euclidean distance from the market-timing vectors[5] of these GDTs. Of course, since now the SOM of the future period is formed by different clusters, the GDTs might not fit as well as they did in the base period. In order to measure this 'unfitting', we use a 'dissatisfaction rate', i.e., how dissatisfied these GDTs will be when placed into a future period's cluster that does not represent their strategy. *If the market is non-stationary, the GDTs' dissatisfaction rate will be high*, as a result of the changes that took place in the market. The dissatisfaction rate is defined as the Euclidean distance of a GDT's market-timing vector to the centroid of the cluster in which it is placed, after the reclustering procedure.

In addition, motivated by Chen and Yeh's stricter test results (Chen and Yeh, 2001), who found that a dinosaur's performance decreases monotonically, we want to investigate if this also applies to our empirical datasets. Hence, under the Dinosaur Hypothesis the following two statements should hold:

The average dissatisfaction rate of the population of trading strategies (GDTs) from future periods should

1. *Not be less than or equal to the dissatisfaction of the base period* (**Test 1**)
2. *Increase continuously, as the testing period moves further away from the base period* (**Test 2**)

As we can see, we have a population of GDTs and we then monitor their future performance in terms of their dissatisfaction rate, in accordance with Arthur's and Chen and Yeh's experiments. Test 1 is basically the procedure we have just described above and is inspired by Arthur's work (Arthur, 1992). Test 2 is inspired by the observation that Chen and Yeh made (Chen and Yeh, 2001), regarding the monotonic decrease of a GDT's performance. However, in our framework we do not require that changes in the dissatisfaction rate are monotonic. The reason for this is because when Chen and Yeh tested for the Dinosaur Hypothesis (they did not explicitly use this term), they only tested it over a period-window of 20 days, which is relatively short, and hence makes it easier to observe an example of monotonically decreasing behavior. Thus, requiring that the GDTs' dissatisfaction rate increases monotonically in the long run would be very strict, and indeed hard to achieve. For that reason, Test 2 requires that the dissatisfaction changes are continuous, but not monotonic. In addition, since the dissatisfaction rate is a concept opposite to 'performance',

[5] A reminder that a market-timing vector is a vector of buy and not-buy signals (1 and 0, respectively).

Test 2 does not require that the dissatisfaction rate decreases with time, as it happened with the performance of dinosaurs, but that it increases over time.

Let us now explain the process of reclustering.

8.4.2 Re-clustering Process

Let us give an example of how reclustering works when 1991a is the base period. Each evolved GDT is first moved to the next period, 1991b, and reclustered into one of the clusters of that period. In order to 'decide' which cluster to choose, the GDT compares the Euclidean distance of its market-timing vector with each cluster; it is then placed into the cluster with the smallest Euclidean distance. The same procedure follows for all GDTs of the population. At the end, the population of evolved GDTs from the base period of 1991a has been reclustered into the clusters of period 1991b. We also follow the same procedure with all future periods. This means that the GDTs from 1991a are also reclustered in 1992a, 1992b, ..., 2007b.

The same process is followed for all other base periods (i.e., 1991b, 1992a, ..., 2007a). This means that when 1991b is the base period, the GDTs that were created and evolved during 1991b will be reclustered into the SOMs of all future periods. After 1991b, 1992a takes over as the base period and the same procedure takes place again. We do this until 2007a. We obviously cannot do this for 2007b, since there are no periods available after this year. The reader should also bear in mind that we only apply the evolved GDTs to the SOMs of future periods; for instance, when the base period is 2000a, we do not apply the evolved GDTs backwards in time, only forwards. We are not interested in looking at what would happen in the past; we are only interested to observe how the dissatisfaction of the GDTs is affected in the future. Figure 8.2 presents the re-clustering process we have just described.

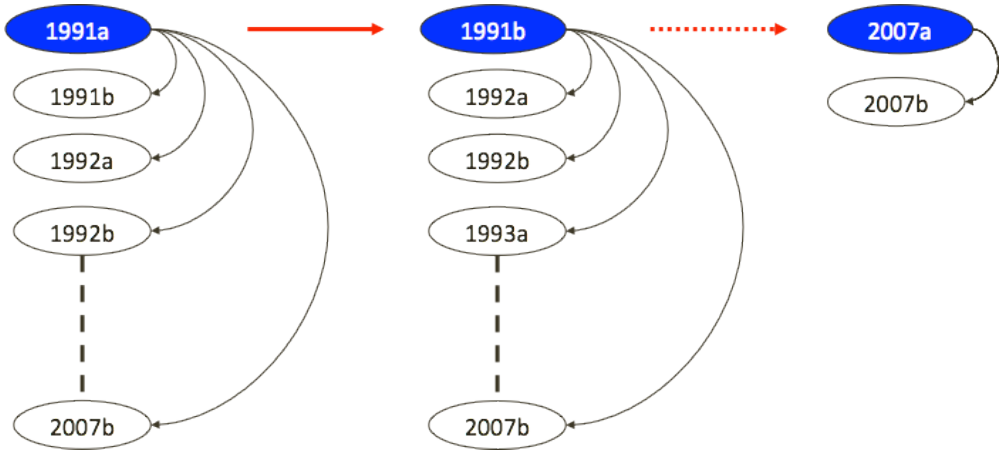


Figure 8.2: Re-clustering process.

Once the process of reclustering is complete, we calculate the dissatisfaction rate of each GDT in the population. Next, we calculate the population's average dissatisfaction rate. We do the same

for all 34 periods. Given a base period, the population average dissatisfaction of all periods is normalized by dividing those population average dissatisfaction rates by the population average dissatisfaction rate in the base period. For instance, if the population dissatisfaction rates for periods 1991a, 1991b, 1992a, ..., 2007b are 0.8, 0.85, 0.9, ..., 0.85, respectively, the the normalized population dissatisfaction for the base period 1991a would be $\frac{0.8}{0.8}$, $\frac{0.85}{0.8}$, $\frac{0.9}{0.8}$,..., $\frac{0.85}{0.8}$. Hence, each base period has its normalized average dissatisfaction rate equal to 1. In order to prove that the market is non-stationary, we need to show that the normalized average dissatisfaction rate of the GDTs increases in the future periods, and never returns to its initial value of 1, which was during the base period. If, on the other hand, this rate reaches 1 or below, *this is an indication of a cyclic market behavior*, since the GDTs have found the same conditions with the base period, and as a result feel as 'satisfied' as before. In this case, we refer to these GDTs as *returning dinosaurs*.

Therefore, Tests 1 and 2 can be re-written as:
The average dissatisfaction rate of the population of trading strategies (GDTs) from future periods should

1. *Not be less than or equal to 1, which is the normalized average dissatisfaction rate of the base period* (**Test 1**)

2. *Increase continuously, as the testing period moves further away from the base period* (**Test 2**)

As we can observe, Test 1 has been modified so that it takes into account the normalized average dissatisfaction rate, which is equal to 1, and acts as the benchmark for returning dinosaurs. Test 2 has remained unchanged. These are thus the two tests that we will be running in Section 8.5 to test if the DH holds under real data.

8.5 Results

This section presents the experimental results. In order to make it easier for the reader, we first start by presenting the results of a single dataset (STI), in Section 8.5.1. We then continue with presenting the summary results for all 10 datasets, in Section 8.5.2.

8.5.1 Results of a single run of a single dataset

Because of the fact that this test is connected with SOM, we are also experimenting with different number of clusters (i.e. different SOM dimensions). This happens because we want to see if our results are sensitive to the number of clusters that can exist in a market. Thus, we first begin by presenting the results for the 3×3 SOM (9 clusters), and then present the results under different SOM dimensions.

8.5.1.1 Results under 3×3 SOM (9 clusters)

As Test 1 states, we want to make sure that the population dissatisfaction from future periods will not return to the range of dissatisfaction of the base period. In other words, we want to make sure that there are no *returning dinosaurs*. Then, Test 2 requires that we examine whether this population dissatisfaction from future periods continues to increase, as we move further away from the base period.

8.5.1.1.1 Test 1 We test this statement for one period at a time. The subject period forms our base period. As already mentioned, a returning dinosaur is observed if any future normalized population dissatisfaction rate is less than or equal to 1. Let us now have a look at Figure 8.3. The x-axis presents the 34 periods and the y-axis presents the dissatisfaction rate for each period. Period 1991a acts here as the base period.

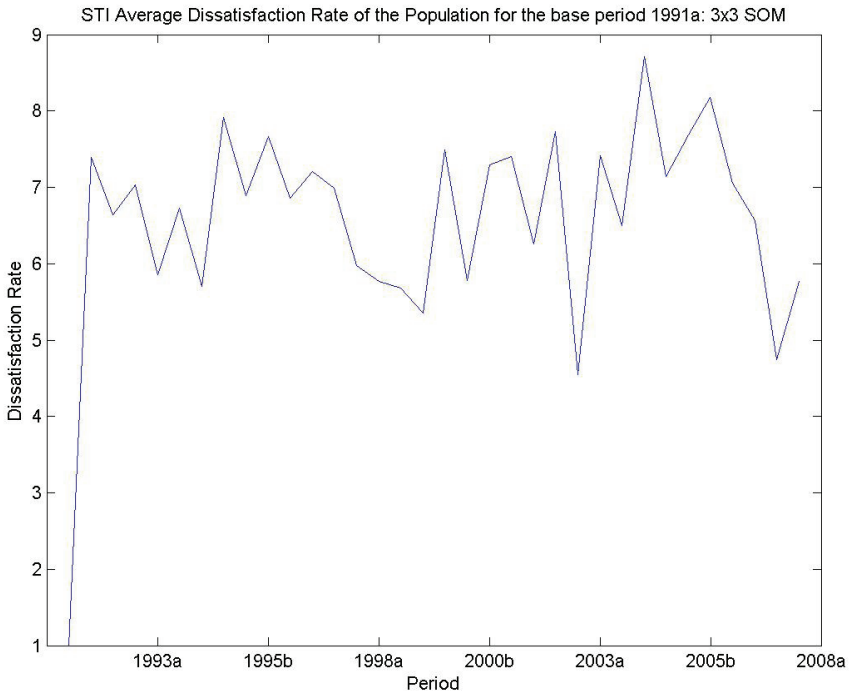


Figure 8.3: Average dissatisfaction rate of the population of GDTs that were evolved in 1991a, over the future 34 periods. Period 1991a acts here as the base period, because this is the period that the evolved trees were initially clustered. After calculating the dissatisfaction rate of each GDT in 1991a, we calculated the average dissatisfaction rate of the population. We then re-clustered these GDTs to all future periods. In this way, we are able to observe how the average dissatisfaction rate of the population is affected while it is moving away from the base period. Results are for 3×3 SOM dimension.

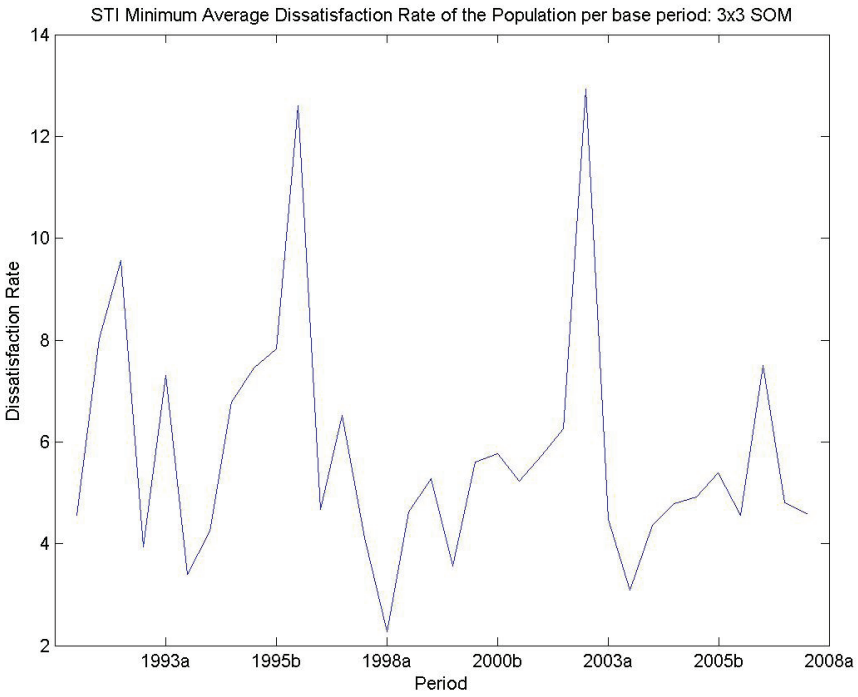


Figure 8.4: Minimum average dissatisfaction rate of the population of GDTs per base period, for 3×3 SOM. After obtaining the series of the average dissatisfaction rate of each base period, over its future periods, we calculate the minimum average dissatisfaction rate per base period. In this way, we have an indication of whether there are any returning dinosaurs in each base period.

As we can see, the dissatisfaction rate immediately increases, after we move away from the base period 1991a. It starts from around 7.5, and never drops again to a range close to 1. On average, the dissatisfaction rate is 6.73, which is far away from the baseline of 1. Thus no returning dinosaurs are observed.

Nevertheless, this result is when 1991a is the base period. Let us now present what happens after we have followed the same procedure for all periods, i.e. after all periods have acted as the base period. In order to do that, we iterate through each base period and calculate the minimum dissatisfaction rate among its future periods. For instance, in Figure 8.3, the minimum dissatisfaction rate is around is around 4.5 in period 2002b. This means that for the base period of 1991a, the lowest dissatisfaction rate that any future period managed to get was 4.5. So we extract this value. We do the same for all 33 base periods and thus end up with a 1×33 vector, $MinAvgDis$, which shows the potential returning dinosaur per base period. The plot of the $MinAvgDis$ vector is shown in Figure 8.4.

As we can observe, there is no base period with a minimum dissatisfaction rate of 1 or below. There are a few periods with relatively low dissatisfaction rates (1992b, 1993a, 1999b, 2003b). However, the lowest rate any period gets is 2, in period 1998a. In addition, the average minimum

dissatisfaction rate for STI is for 3×3 SOM is 5.79. We can therefore conclude that strategies are on average not 'satisfied' in future years, and thus that Test 1 is supported by this single run.

8.5.1.1.2 Test 2 Let us now move to Test 2. To show a continuous increase in the population dissatisfaction rate, we calculate the sum of the dissatisfaction rates of all those future periods that have distance from the base period equal to 1, then the sum of those future periods with distance equal to 2, and so on, up to distance equal to 33. In order to do this, we first need to create a table of distances, like the one in Table 8.1. Each row of this table presents the distance of the future periods from their base period. For instance, if 91a is the base (first row), then future period 91b has distance equal to 1, future period 92a has distance equal to 2, and so on. Table 8.2 presents an example of series of normalized population dissatisfaction rates for the future periods of each base period. For example, when the base period is 91a (first row), the normalized dissatisfaction rate starts from 1 in 91a, then rises to 7.66 (91b), then goes to 6.82 (92a), and so on, until it reaches dissatisfaction equal to 5.94 in future period 07b. Let us now denote the sum of dissatisfaction rates we mentioned at the beginning of this section by $\sum_{|i-j|=m} Dis(i,j)$, where i, j are the base and future period respectively, $|i-j|$ is their absolute distance, as presented in Table 8.1, and m is the distance from the base period and takes values from 1 to 33. We divide this sum by the number of occurrences where $|i-j| = m$. This process hence returns the average of the normalized population dissatisfaction, and allows us to observe how it changes, as the distance m from the base period increases. We call this metric D_m and it is presented in Equation 8.1.

Table 8.1: Distance of future periods from their base period, over the 17 years 1991-2007. The further away we move from a period, a single unit of distance is added.

		j					
		91a	91b	92a	92b	...	07b
i	91a	0	1	2	3	...	33
	91b	1	0	1	2	...	32
	92a	2	1	0	1	...	31
	92b	3	2	1	0	...	30
	...	...	...	...	...	...	...
	07b	33	32	31	30	...	0

$$D_m = \frac{\sum_{|i-j|=m} Dis(i,j)}{\{\#(i,j), |i-j| = m\}} \tag{8.1}$$

Let us give an example: if we want to calculate D_{32}, we need to sum up the population dissatisfaction rates that have distance $m = 32$. This happens with $Dis(91a, 07a)$ (dissatisfaction rate of

Table 8.2: Example of series of future population dissatisfaction rates per base period. Each base period's series is presented as a horizontal line of this table. Dissatisfaction rates have been normalized, so that the rate in the base period is always equal to 1.

		j					
		91a	91b	92a	92b	...	07b
	91a	1	7.66	6.82	7.08	...	5.94
	91b		1	3.76	5.72	...	4.70
i	92a			1	9.74	...	7.77
	92b				1	...	8.80
	...	...	...	...	...	...	...
	07b					...	1

GDTs from base period 91a, when applied to future period 07a) and $Dis(91b, 07b)$ (dissatisfaction rate of GDTs from base period 91b, when applied to future period 07b). Therefore D_{32} would be equal to the sum of these two rates over 2, as there are only 2 periods we are interested in.[6] By calculating D_m for all m values, we can have a clear idea of how the average of the population dissatisfaction changes when we move from periods that are close to the base (low m), to periods that are further away (high m), and thus observe whether there is a continuous increase.

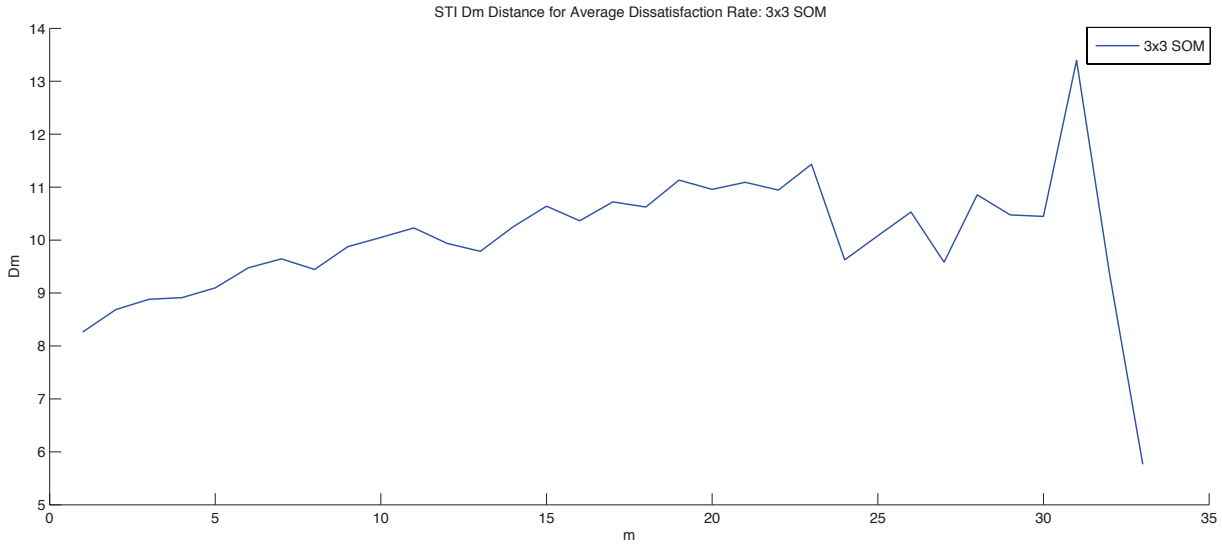


Figure 8.5: STI D_m distance of the dissatisfaction rate for 3×3 SOM.

[6]The distance $m = 32$ can also be found in 07a91a and 07b91b. However, we do not take them into account because, as we said earlier in Section 8.4, we are not interested in applying the evolved GDTs of a base period (here 07a and 07b) backwards in time (91a and 91b, respectively).

Figure 8.5 presents the results. It is interesting that for $m = 1-22$, D_m experiences an increase. However this changes for higher values of m. Overall, Test 2 seems not to be justified for STI, under 3×3 SOM. Later in the chapter, after we will have presented the results under all indices, we run linear trend regression for the D_m observations, calculate the slope of the trend line and investigate whereas the above observation indeed holds.

To summarize, the experimental results of Test 1 for the STI dataset under the 3×3 SOM showed that there are no returning dinosaurs. Regarding Test 2, a continuous increase in the average dissatisfaction rate of the population of GDTs could only be observed for some periods.

8.5.1.2 Experimenting with the number of clusters

Let us now examine if the above results hold under different numbers of clusters. We experiment with the following SOM dimensions: 2×1, 3×1, 2×2, 5×1, 3×2, 7×1, 4×2, 3×3, for a range of trading strategy types 2-9. All graphs in this section include information for all of the different dimensions.

First let us start with the average dissatisfaction rate for the base period 1991a, which is presented in Figure 8.6. As we can observe, the dissatisfaction rate follows the exact same pattern we observed earlier: it immediately increases, after we move away from the base period. It is in a range from 3 to around 9, depending on the SOM dimension, and never drops again to a range close to 1. On average, the average dissatisfaction rate for 1991a varies in the range of 4.29 under 2 types of trading strategies (clusters), to 6.73 (9 trading strategy types). Thus there are again no returning dinosaurs for any SOM dimension. In addition, we again do not observe a continuous increase in the dissatisfaction rate. Hence, Test 1 holds for all SOM dimensions, whereas Test 2 does not. One final observation we can make is that the dissatisfaction rate increases as the number of clusters increases. However, it seems that the pattern of repetitive behavior we observed earlier remains, under all trading strategy types. The number of clusters does not affect Test 2's results.

Let us now present the same results for all base periods. Figure 8.7 presents these results. The horizontal line denotes when the dissatisfaction rate is equal to 1, and is given as a reference.

As we can observe, nothing really changes when compared to the results presented earlier, for the 3×3 SOM. There is again no base period, for any SOM dimension, reaching average dissatisfaction rate of 1. The lowest rate we observe is again for period 1998a, which is around 2. On average, the minimum dissatisfaction rate varies in the range of 3.56 (2 clusters) to 5.79 (9 clusters). Therefore, we can again see that dinosaurs do not return and thus that Test 1 is supported under all SOM dimensions for the STI dataset.

Regarding Test 2, we again use the D_m metric. Figure 8.8 presents the results. Once again, we observe the same pattern for all SOM dimensions, as with the ones we saw earlier for the 3×3 SOM: continuous increase up to a point, and then upward and downward movements. In addition, we again see that as the number of clusters increases, there is also an increase in the dissatisfaction rate.

To summarize, the experimental results of Test 1 for the STI dataset show that there are no returning dinosaurs, under all different SOM dimensions tested. The market seems to be constantly

Figure 8.6: Average dissatisfaction rate of the population of GDTs that were evolved in 1991a, over the future 34 periods. Period 1991a acts here as the base period, because this is the period that the evolved trees were initially clustered. After calculating the dissatisfaction rate of each GDT in 1991a, we calculated the average dissatisfaction rate of the population. We then re-clustered these GDTs to all future periods. In this way, we are able to observe how the average dissatisfaction rate of the population is affected while it is moving away from the base period. Each line in the figure represents a different SOM dimension.

changing and hence strategies are not 'satisfied' in their new environments. Furthermore, changing the SOM dimension (and thus the number of strategy types in the market) does not seem to significantly affect the results. The only difference we can observe is that the dissatisfaction rate increases when more types of strategies exist in the market (more SOM dimensions). Finally, there seems that there is no continuous increase in the average dissatisfaction rate of the GDTs, after we move away from the base period, and thus Test 2 is not supported by the STI dataset.[7]

Let us now examine if the above results generalize under different international markets.

8.5.2 Summary results for all datasets

So far we have presented the results for a single dataset, STI. In this section we will present the results for all 10 datasets. As a reminder, these datasets are: CAC40, DJIA, FTSE100, HSI, NASDAQ, NIKEI, NYSE, S&P500, STI, TAIEX.

[7] As already mentioned, later in Section 8.5.2.2 we run linear trend regression for the D_m observations, calculate the slope of the trend line and show that indeed there is no continuous increase.

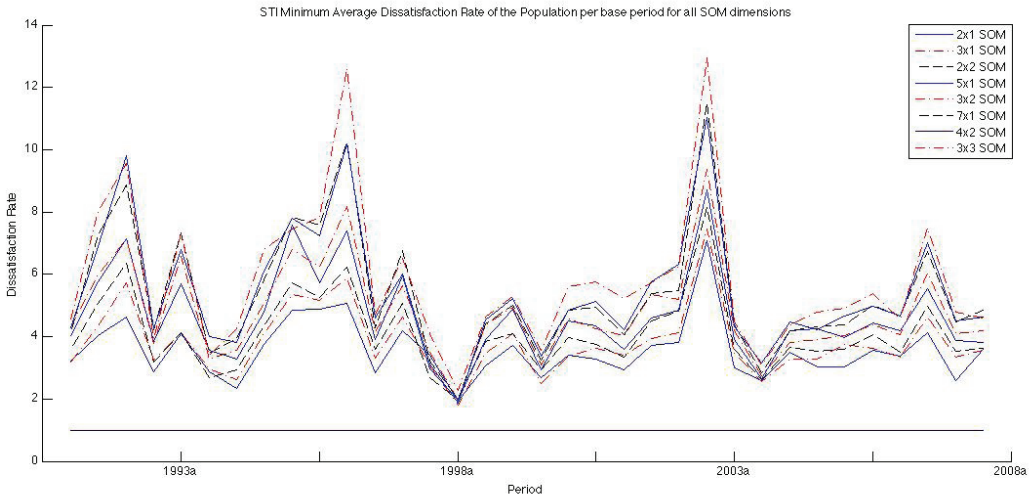


Figure 8.7: STI Minimum average dissatisfaction rate of the population of GDTs per base period, for all SOM dimensions. After obtaining the series of the average dissatisfaction rate of each base period, over its future periods, we calculate the minimum average dissatisfaction rate per base period. In this way, we have an indication of whether there are any returning dinosaurs in each base period. The horizontal line denotes dissatisfaction rate equal to 1, and is given as a reference.

8.5.2.1 Test 1

Figures 8.9 and 8.10 present the results for Test 1. There are ten subfigures, each one presenting the average dissatisfaction rate for all SOM dimensions, for each one of the datasets. A reminder that the horizontal line at each subfigure denotes when the average dissatisfaction rate is equal to 1; as we already know, whenever a period's rate goes close to 1, then this period has a returning dinosaur. From the graphs it is clear that no dataset reaches a minimum average dissatisfaction rate of 1. Tables 8.3 and 8.4 also present the average, over 10 runs, of average and minimum dissatisfaction rate per cluster, per dataset. As we can observe, on average all datasets have an average dissatisfaction rate of 4.78 for 2×1 SOM, which climbs to 7.95 for 3×3 SOM. There seems to be a lower boundary, which does not allow the average dissatisfaction rate to go below it. This lower boundary is around 4.5-5 for the 2×1 SOM and rises with the number of clusters, reaching around 7-8 for the 3×3 SOM. Standard deviation results provided in Table E.3 are relatively low, and thus results are liable. But even if we want to take into account the outliers, Table 8.4 informs us that on average the minimum dissatisfaction rate does not go below 3.40 for 2×1 SOM and 5.50 for 3×3 SOM. It is thus obvious that GDTs do not find a familiar environment in future periods. Hence, *dinosaurs do not return or return only as lizards*. The latter refers to the exceptional cases when dissatisfaction goes relatively low, for instance around 2. Market conditions have allowed old strategies to feel 'satisfied' again, although this satisfaction is not as high as in the base period. Thus market conditions have some similarities to the ones of

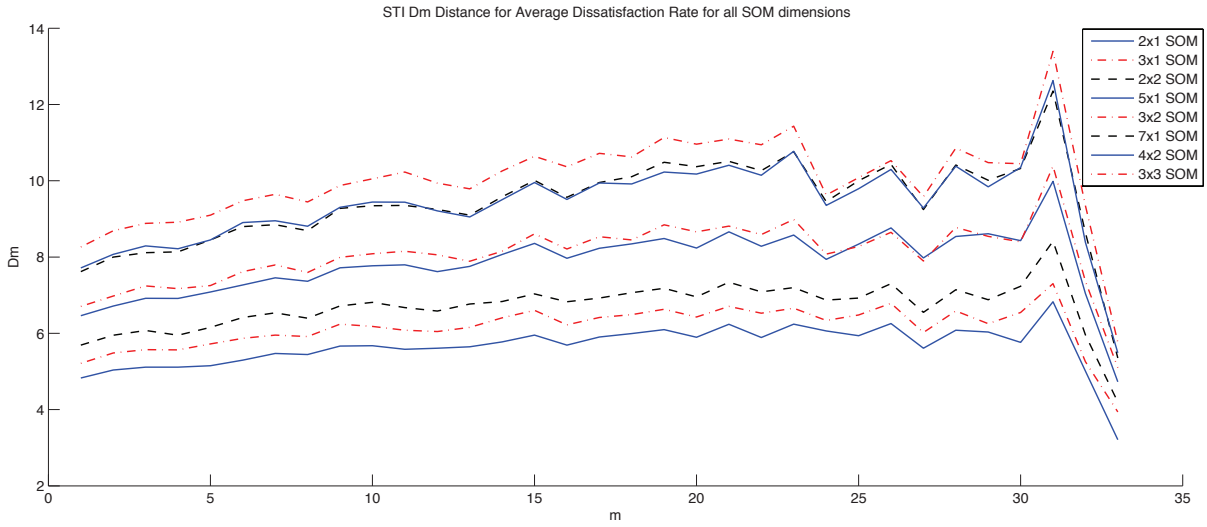


Figure 8.8: STI D_m distance of the dissatisfaction rate for all SOM dimensions.

the base period, but by no means are the same or offer the same 'satisfaction' to the strategies. Hence, Test 1 is supported by all 10 datasets, under all SOM dimensions.

8.5.2.2 Test 2

Let us now move to Test 2, which is presented in Figure 8.11. Each subfigure presents the results under a different market. DJIA, FTSE 100 and NIKEI seem to experience a continuous increase in the D_m metric. On the other hand, evidence for a continuous increase in the other markets seems weak.

In order to confirm this eye-browsing observation, we run a linear trend regression for each dataset. Our goal is to examine if the slope of the trend is positive; if this happens, then it denotes a continuous increase in the metric. One more thing we should say is that we only run the linear regression for a single SOM dimension, the 3×3 one. The reason for this is because for each dataset, the D_m metric experiences the same patterns, under all SOM dimensions. Thus it would not make a big difference if we took another dimension and applied linear regression. It should also be mentioned that because the variance of the D_m observations seem relatively high, we apply variance stabilizing transformation, and use the logarithmic values instead of the raw values of the metric. Figure 8.12 presents the results of the linear regression. From the graphs we can clearly see that HSI and NYSE experience a continuous *decrease* in the metric. The majority of the remaining datasets' metric seems to remain stable over time, with the exception of DJIA, FTSE 100 and NIKEI.

Table 8.5 presents the coefficients, which denote the slope of the trend. In order to see whether the coefficients are statistically significant, we run a t-test and present the respective p-values. As we can observe, the p-value is below 5% (significance level) for the following datasets: DJIA,

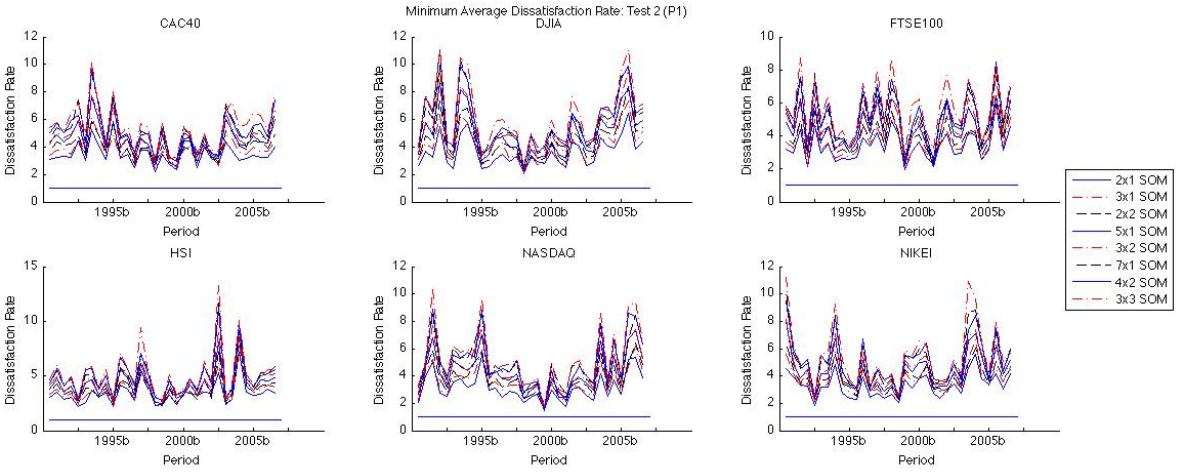


Figure 8.9: Test 1: Minimum average dissatisfaction rate of the population of GDTs per base period, for all SOM dimensions, for all datasets. Each subfigure represents a single dataset. From left to right, top to bottom: CAC40, DJIA, FTSE100, HSI, NASDAQ, NIKEI.

FTSE 100, HSI, NIKEI and NYSE. This means that DJIA, FTSE 100 and NIKEI 's positive slope is statistically significant. Thus *these markets experience a continuous increase* in their D_m metric. The remaining markets have either a statistically significant negative coefficient, thus a continuous decrease (HSI, NYSE), or no statistically significant positive/negative coefficient, thus no evidence for continuous increase (CAC 40, NASDAQ, S&P 500, STI, TAIEX). Therefore, Test 2 is supported only by 3 out of the 10 datasets tested.

8.6 Testing the DH under different GP algorithms

In this section, we are interested in observing whether the previously derived results hold under different GP algorithms. If they do, this allows us to argue that our results are rigorous and independent of the GP algorithm and we can thus generalize our previous conclusions. We again use EDDIE 7 and EDDIE 8 for our tests, as we also did earlier in Chapter 7, in the MFH experiments. The same methodology is used, and of course the same 10 datasets.

8.6.1 Test 1

Figures 8.13-8.16 present the results for Test 1, under EDDIE 7 and EDDIE 8. As earlier, there are ten subfigures, each one presenting the average dissatisfaction rate for all SOM dimensions, for each one of the datasets. It is again clear that no dataset reaches the minimum average dissatisfaction rate of 1. On average, the minimum average dissatisfaction does not go below 3.58 (EDDIE 7) and 3.37 (EDDIE 8) for the 2×1 SOM, and 5.71 (EDDIE 7) and 5.30 (EDDIE 8) for the 3×3

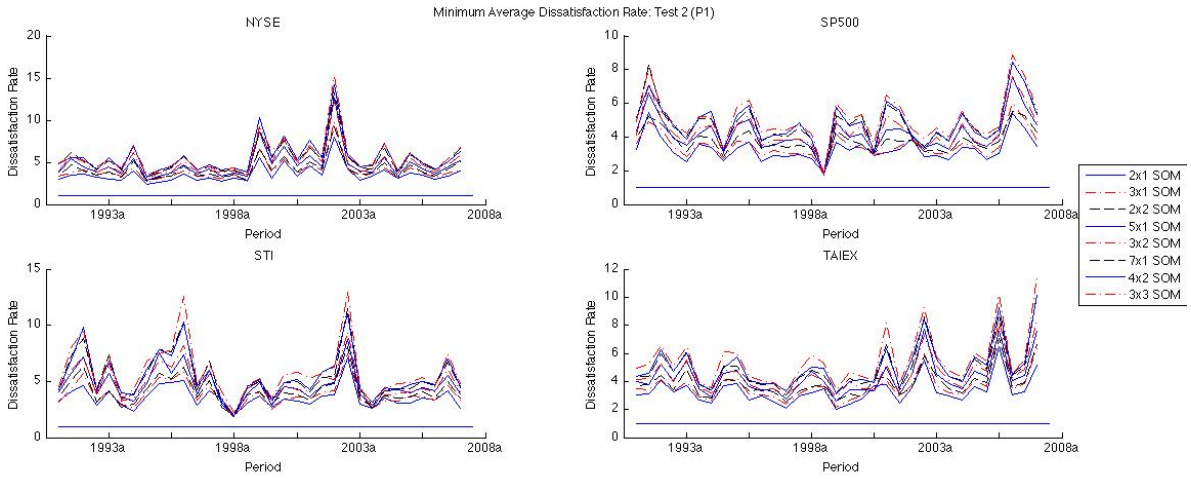


Figure 8.10: Test 1: Minimum average dissatisfaction rate of the population of GDTs per base period, for all SOM dimensions, for all datasets. Each subfigure represents a single dataset. From left to right, top to bottom: NYSE, S&P500, STI, TAIEX

SOM. As we can see, the differences between the EDDIE results and the earlier ones from simple GP are very small. In addition, the average dissatisfaction rate per cluster is on average 4.39 (EDDIE 7) and 4.17 (EDDIE 8) for the 2×1 SOM, and 7.62 (EDDIE 7) and 7.14 (EDDIE 8) for the 3×3 SOM. We can thus see that in both terms of average and minimum dissatisfaction rate, results under both EDDIE 7 and EDDIE 8 verify the simple GP ones, i.e. that *there are no returning dinosaurs*. We can therefore conclude that Test 1 is also supported by EDDIE 7 and EDDIE 8. The tables of the dissatisfaction rates, under the EDDIE algorithms, are presented in Appendix D.

8.6.2 Test 2

Figures 8.17 and 8.18 present the results for Test 2, under EDDIE 7 and EDDIE 8, respectively. A first observation that can be made is that similar patterns exist, as the ones with the *simple GP*. In order to understand these patterns better, we again run linear trend regression and calculate the statistical significance of the slope at 5% significance level. Figures 8.19 and 8.20 present the regression graphs for the D_m metric, for EDDIE 7 and EDDIE 8, respectively. In addition, Table 8.6 presents the slope of each trend, as well as the p-value of the t-test.

Results for EDDIE 7 show that only 3 datasets experience a positive slope: CAC 40, FTSE 100, and HSI. However, only FTSE 100 and HSI's coefficients are statistically significant, thus only these two markets experience a statistically significant continuous increase in the dissatisfaction rate. The remaining 8 markets either don't have a statistically significant continuous increase, or even have a continuous decrease.

Results for EDDIE 8 show a similar picture. Only one market (NASDAQ) experiences a statistically significant continuous increase in the dissatisfaction rate. There are 3 more datasets that

Table 8.3: Average of Average Dissatisfaction Rate per Cluster per Dataset.

	2×1	3×1	2×2	5×1	3×2	7×1	4×2	3×3
CAC40	4.69	5.04	5.32	6.03	6.27	7.09	7.13	7.46
DJIA	4.93	5.46	5.88	6.69	7.05	7.76	7.98	8.51
FTSE100	4.77	5.12	5.55	6.37	6.62	7.43	7.52	8.05
HIS	5.06	5.49	5.79	6.68	6.90	7.75	7.87	8.32
NASDAQ	4.42	4.84	5.21	5.94	6.16	6.91	6.93	7.50
NIKEI	4.88	5.26	5.50	6.54	6.69	7.68	7.64	8.28
NYSE	4.44	4.90	5.22	5.92	6.17	6.88	6.96	7.27
S&P500	4.36	4.61	4.93	5.53	5.79	6.45	6.56	6.86
STI	5.17	5.65	6.11	6.98	7.19	8.30	8.33	8.88
TAIEX	5.04	5.48	5.74	6.54	6.96	7.76	7.78	8.40
Mean	4.78	5.19	5.53	6.32	6.58	7.40	7.47	7.95

experience an insignificant increase (DJIA, HSI, NYSE), whereas the remaining 6 datasets experience a continuous decrease.

Overall, all three GP algorithms show that for the majority of the datasets tested, there is no strong evidence for a continuous increase in the dissatisfaction rate. We can hence conclude that Test 2 is not supported for the majority of the datasets, under EDDIE 7 and EDDIE 8. It is, of course, interesting that there are always a few exceptions. At the moment it is unclear what determines the fact of a continuous increase in the dissatisfaction rate of the respective datasets. We thus leave this as a future work.

8.7 Conclusion

To summarize, this chapter presented the Dinosaur Hypothesis (DH), which states that the behavior of a market never settles down and that the strategies in this market continuously co-evolve with it. This observation was first made by Arthur (1992) and later by Chen and Yeh (2001). However, these models made this observation based on simulations under artificial stock markets. In our current work, we were interested in examining whether these observations could hold as well in the real world, because this could give us valuable information regarding the nature of financial markets (stationary vs non-stationary). We thus first formalized the hypothesis by presenting its main constituents. This allowed us to form a testing methodology and then run tests under 10 international markets. In order to run these tests, we extended the framework from Chapter 7, where we had empirically examined the dynamics of market fractions of different types of agents. In that study, we used Genetic Programming (GP) as a rule-inference engine to find out the behavioral rules of agents, and Self-Organizing Map (SOM) to cluster these agents. However, because of an

Table 8.4: Average of Minimum Dissatisfaction Rate per Cluster per Dataset.

	2×1	3×1	2×2	5×1	3×2	7×1	4×2	3×3
CAC40	3.51	3.77	4.01	4.53	4.68	5.10	5.13	5.51
DJIA	3.60	4.00	4.37	4.85	5.14	5.58	5.77	6.14
FTSE100	3.35	3.56	3.88	4.34	4.50	5.01	5.11	5.54
HSI	3.38	3.58	3.77	4.15	4.30	4.80	4.83	5.15
NASDAQ	3.16	3.51	3.74	4.20	4.35	4.89	4.84	5.25
NIKEI225	3.36	3.74	3.88	4.40	4.53	5.13	5.18	5.57
NYSE	3.52	3.94	4.14	4.69	4.98	5.33	5.53	5.56
SP&500	3.28	3.48	3.77	4.18	4.31	4.78	4.80	5.07
STI	3.56	3.83	4.09	4.61	4.79	5.34	5.41	5.79
TAIEX	3.29	3.64	3.83	4.25	4.44	4.88	5.00	5.43
Mean	3.40	3.71	3.95	4.42	4.60	5.09	5.16	5.50

important assumption in that work, we had to require that SOM clusters, as well as their operational specification, would remain the same over time. In this chapter, we relaxed that assumption. This offered more realism to our model and allowed us to investigate market behavior dynamics and test for the DH.

We used two tests, which were inspired by the works of Arthur (1992), and Chen and Yeh (2001) (see Chapter 5 for details). The first test investigated whether strategies from the past can successfully be re-applied to the future. The results showed that this is not possible and thus verified the hypothesis. Strategies that have not co-evolved with the market become dinosaurs and cannot fit new environments. The second test investigated whether the dissatisfaction rate of such strategies, which are being applied to new environments, can follow a continuous increase. This was verified only by 3 out of the 10 datasets tested. Strategies are dissatisfied, or in other words do not fit their environments if they do not adapt to the changes that have taken place; however, this dissatisfaction does not necessarily increase continuously with time. Nevertheless, it is quite interesting that results regarding the continuous increase of the dissatisfaction rate are not homogeneous. This certainly deserves further investigation and we thus leave it as a future work.

What we can safely conclude from our experimental results is that trading strategies always need to follow changes that take place in the markets, and co-evolve with them. The implications of our results are very important. If strategies do not co-evolve with the market, they eventually become ineffective, even if they were very successful in the past. Markets constantly change and we need to follow and adapt to these changes if we want to remain successful. We can thus also conclude that *the behavior of financial markets is non-stationary*.

In addition to the standard testing of the hypothesis, where we used a *simple GP*, we also presented tests on the DH under two different GP algorithms. The reason for doing this was because we wanted to demonstrate that the results under the *simple GP* are rigorous and not algorithm-

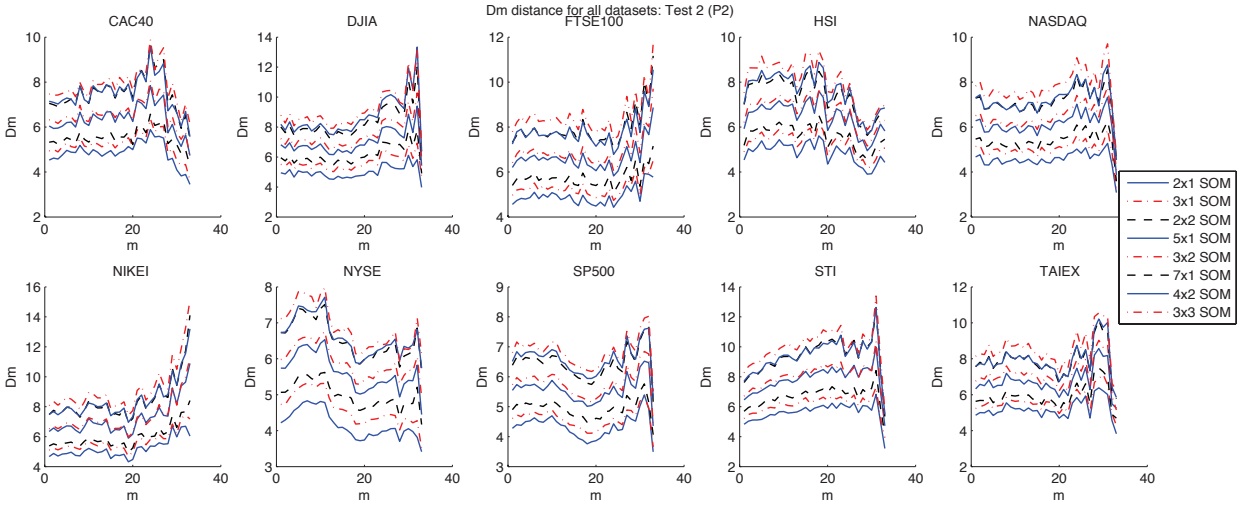
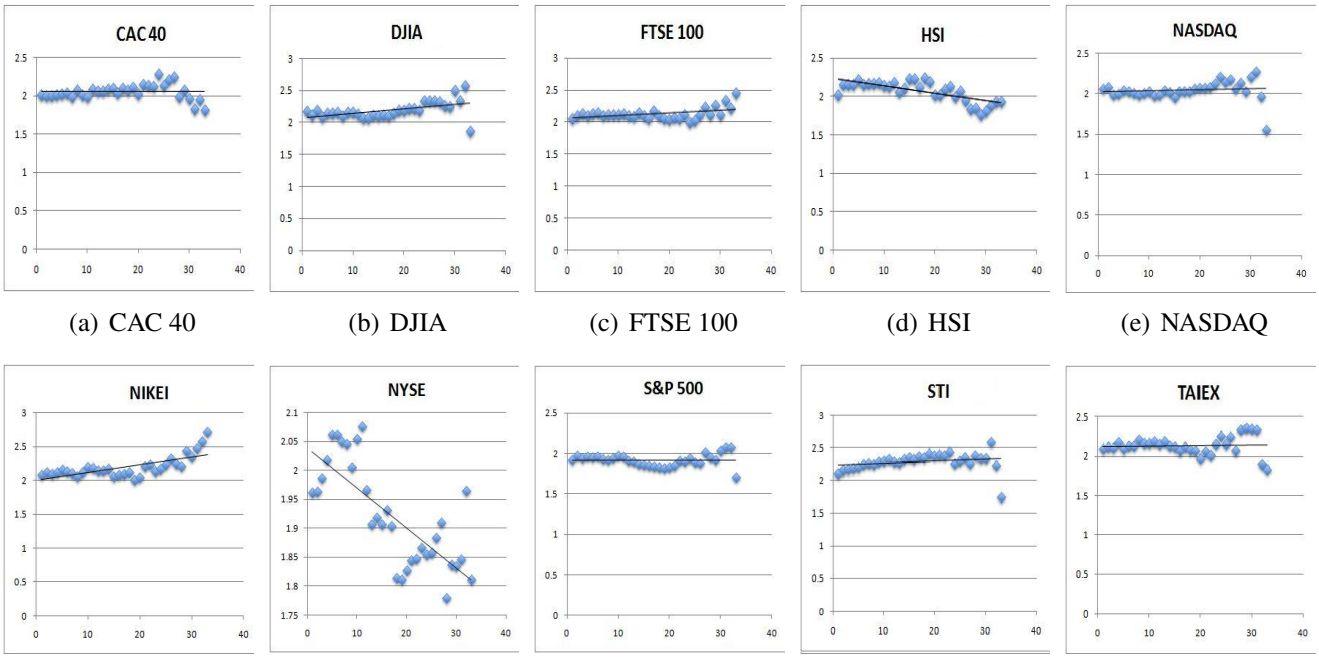


Figure 8.11: Test 2: D_m distance of the dissatisfaction rate for all SOM dimensions for all datasets. Each subfigure represents a single dataset.

dependent. We thus ran the same tests under EDDIE 7 and EDDIE 8. Results showed that both versions agree with the original results. More specifically, both versions demonstrated the non-existence of returning dinosaurs for all 10 datasets. Thus, Test 1 is supported by our experimental results. On the other hand, Test 2 is not supported, as we did not observe a continuous increase in the average dissatisfaction rate. These findings are of course very important, because they strengthen our argument of dinosaurs not being able to return in financial markets.

Finally, as we mentioned in the previous chapter, future research will include experimentation with the control parameters of our framework, as discussed at the end of Chapter 7. We also plan to use different clustering tools in order to investigate the sensitivity of our results to the SOM algorithm. Moreover, as with the MFH chapter, future work also includes investigating the underlying strategy types behind the SOM clusters. Doing so here would allow us to understand what makes a trading strategy successful in the first place. Lastly, it would be interesting to investigate the reason of having returning "lizards" under certain periods. Thus, the question we need to address in the future is what are the underlying conditions in a market that allow the dissatisfaction rate of a dinosaur to be similar to the rate of the base period.



(a) CAC 40 (b) DJIA (c) FTSE 100 (d) HSI (e) NASDAQ

(f) NIKEI (g) NYSE (h) S&P 500 (i) STI (j) TAIEX

Figure 8.12: Linear Regression for the 3×3 SOM observations for the D_m metric.

Table 8.5: Slope of the trend and p-values per dataset.

	Coefficient	p-value
CAC 40	0.000256	0.8905560
DJIA	0.006977	0.0027034
FTSE 100	0.003843	0.0154329
HSI	-0.009849	0.0000024
NASDAQ	0.001508	0.4836810
NIKEI	0.011413	0.0000053
NYSE	-0.006919	0.0000004
S&P 500	-0.000095	0.9468793
STI	0.003035	0.2231986
TAIEX	0.000558	0.7977585

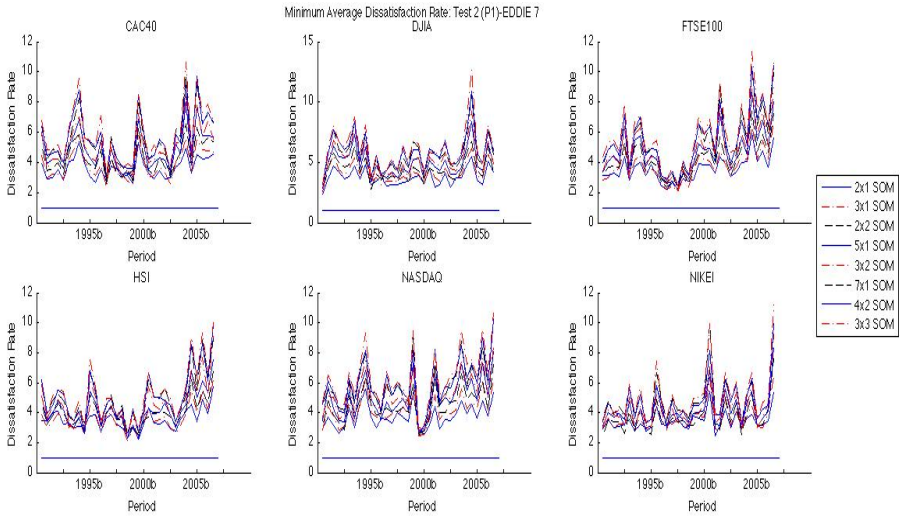


Figure 8.13: Test 1: Minimum average dissatisfaction rate of the population of GDTs per base period, for all SOM dimensions, for all datasets, for EDDIE 7. Each subfigure represents a single dataset. From left to right, top to bottom: CAC40, DJIA, FTSE100, HSI, NASDAQ, NIKEI.

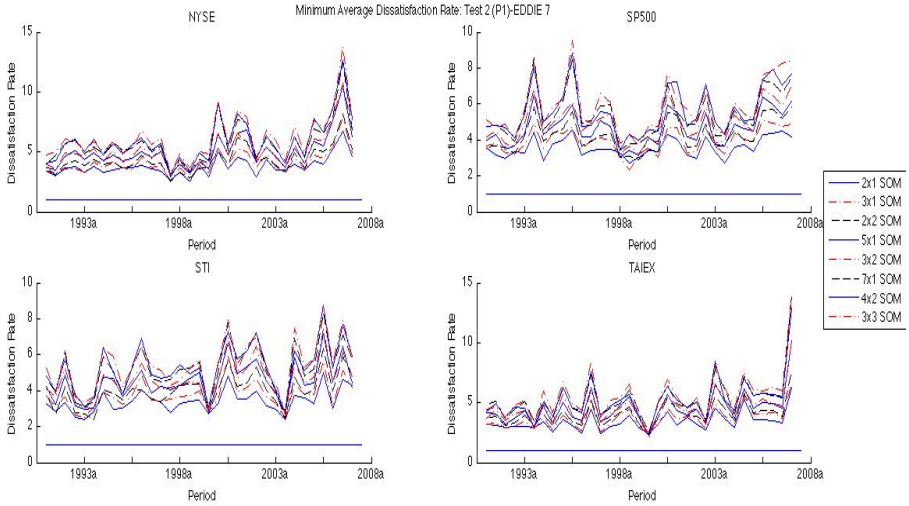


Figure 8.14: Test 1: Minimum average dissatisfaction rate of the population of GDTs per base period, for all SOM dimensions, for all datasets, for EDDIE 7. Each subfigure represents a single dataset. From left to right, top to bottom: NYSE, S&P500, STI, TAIEX

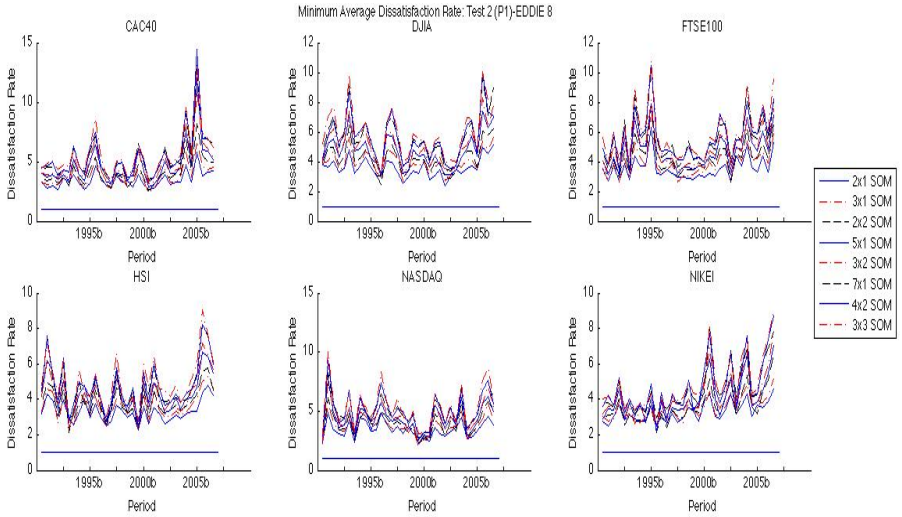


Figure 8.15: Test 1: Minimum average dissatisfaction rate of the population of GDTs per base period, for all SOM dimensions, for all datasets, for EDDIE 8. Each subfigure represents a single dataset. From left to right, top to bottom: CAC40, DJIA, FTSE100, HSI, NASDAQ, NIKEI.

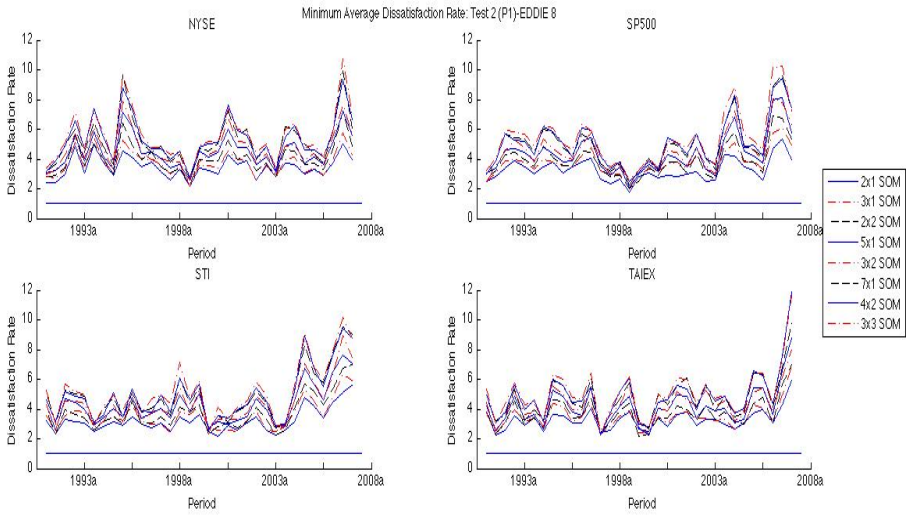


Figure 8.16: Test 1: Minimum average dissatisfaction rate of the population of GDTs per base period, for all SOM dimensions, for all datasets, for EDDIE 8. Each subfigure represents a single dataset. From left to right, top to bottom: NYSE, S&P500, STI, TAIEX

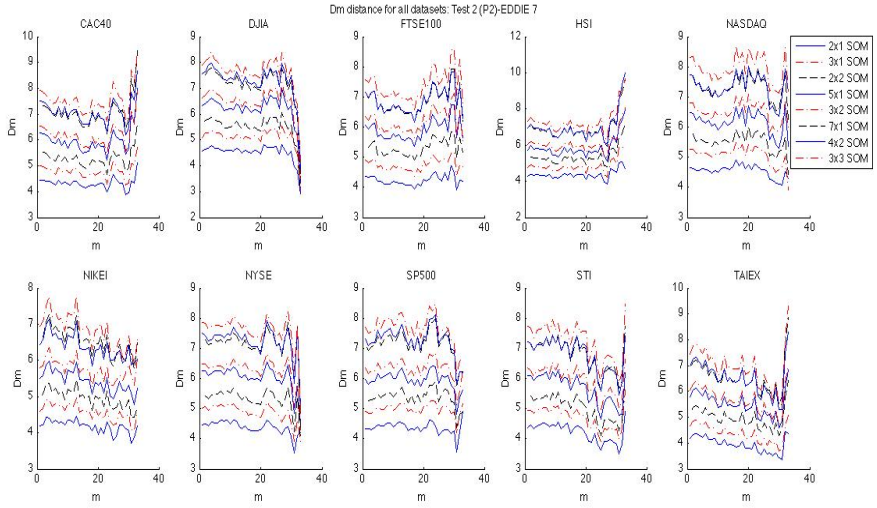


Figure 8.17: Test 2: D_m distance of the dissatisfaction rate for all SOM dimensions for all datasets for EDDIE 7. Each subfigure represents a single dataset.

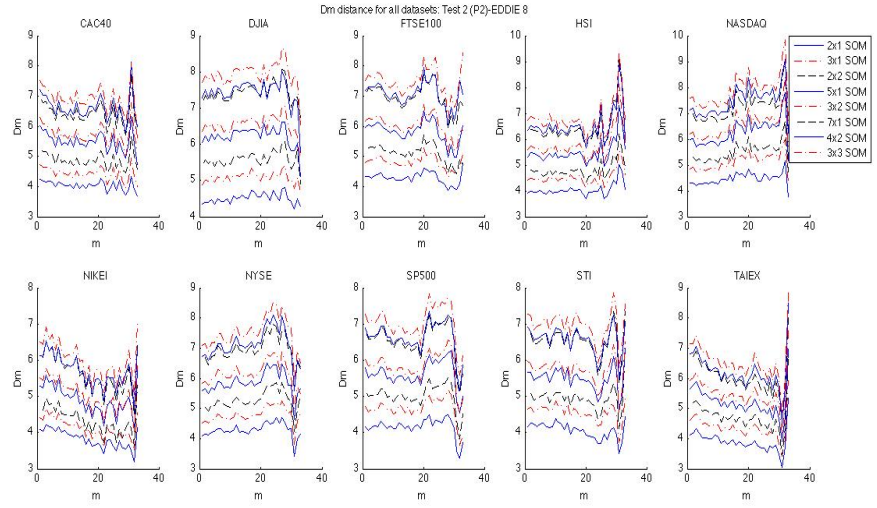
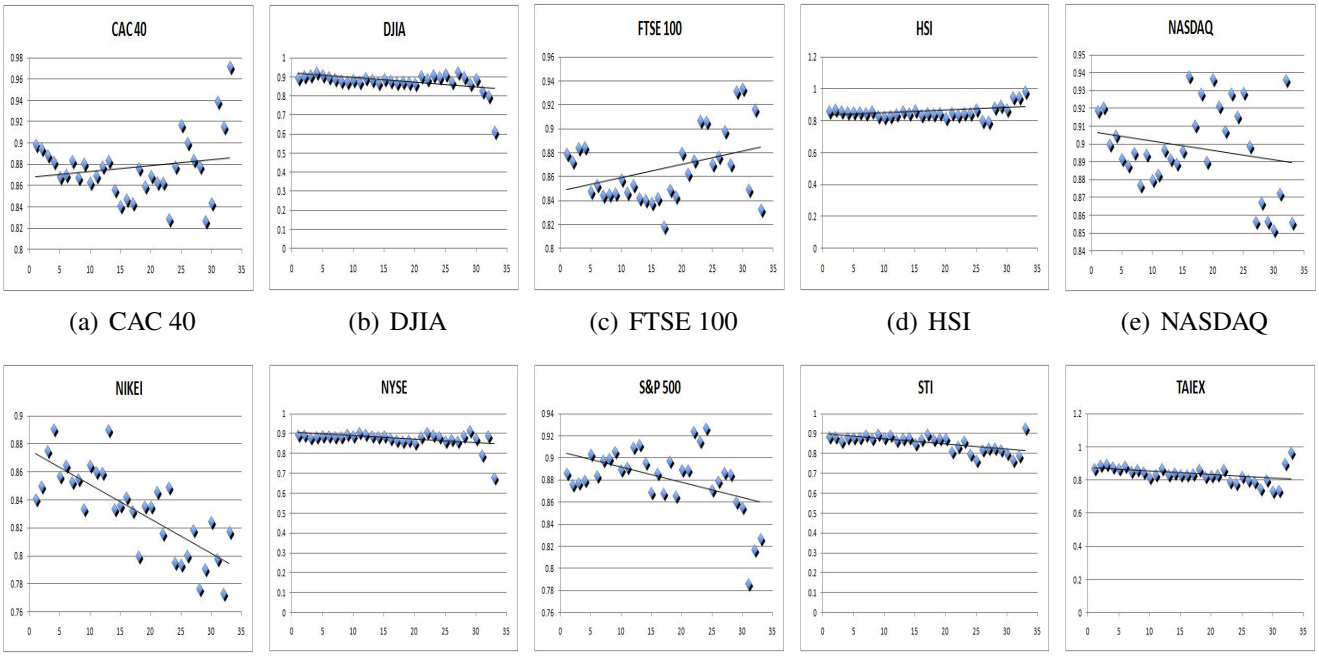


Figure 8.18: Test 2: D_m distance of the dissatisfaction rate for all SOM dimensions for all datasets for EDDIE 8. Each subfigure represents a single dataset.



(a) CAC 40 (b) DJIA (c) FTSE 100 (d) HSI (e) NASDAQ

(f) NIKEI (g) NYSE (h) S&P 500 (i) STI (j) TAIEX

Figure 8.19: Linear Regression for the 3×3 SOM observations for the D_m metric for EDDIE 7.

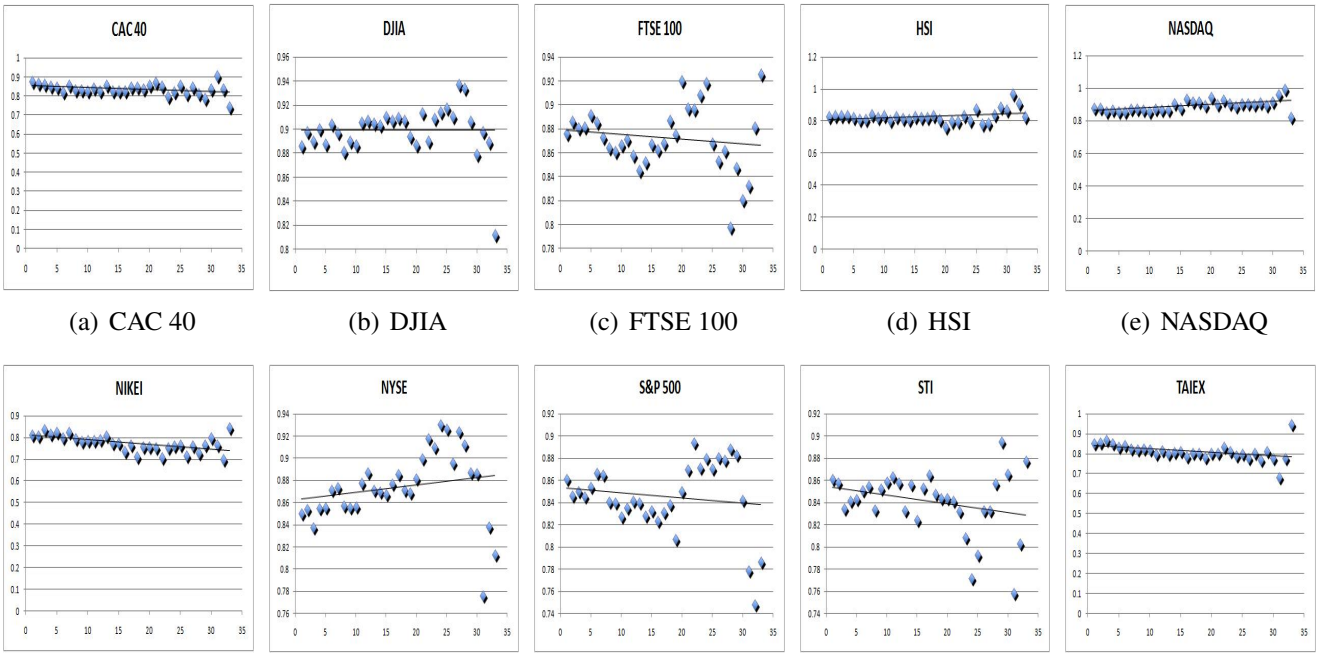
Table 8.6: Slope of the trend and p-values per dataset for EDDIE 7 (a) and EDDIE 8 (b).

(a) EDDIE 7

	Coefficient	p-value
CAC 40	0.0005547	0.316325054
DJIA	-0.0024297	0.010205554
FTSE 100	0.0010992	0.033923973
HSI	0.0015698	0.024336519
NASDAQ	-0.0005332	0.246262357
NIKEI	-0.0024659	8.69461E-08
NYSE	-0.0017806	0.013860675
S&P 500	-0.0013807	0.007529298
STI	-0.0025821	0.000107345
TAIEX	-0.0022289	0.010355922

(b) EDDIE 8

	Coefficient	p-value
CAC 40	-0.0010161	0.051839157
DJIA	0.0000143	0.970421371
FTSE 100	-0.0003952	0.435197783
HSI	0.0013025	0.06250284
NASDAQ	0.0019621	0.000775323
NIKEI	-0.0021504	0.000781767
NYSE	0.0006542	0.276663372
S&P 500	-0.0004565	0.44375302
STI	-0.0007854	0.128372175
TAIEX	-0.0017310	0.01904134



(a) CAC 40 (b) DJIA (c) FTSE 100 (d) HSI (e) NASDAQ

(f) NIKEI (g) NYSE (h) S&P 500 (i) STI (j) TAIEX

Figure 8.20: Linear Regression for the 3×3 SOM observations for the D_m metric for EDDIE 8.

Part V

Concluding Remarks

Chapter 9

Conclusion

This thesis applied Computational Intelligence techniques to applications from the fields of finance and economics. More specifically, it used Genetic Programming to address the problem of pre-specified technical indicators that are used for investment opportunities forecasting. In addition, we also used Genetic Programming and Self-Organizing Map to test two hypotheses that derive from the economics field: the Market Fraction Hypothesis and the Dinosaur Hypothesis. Conclusions for the experiments on these problems are presented in the following sections. We start by discussing the motivation behind each experimental work, we then continue by discussing the novelty of the presented research, and then present the conclusions of each work. Finally, we discuss future work at the end of this chapter.

9.1 Summary of EDDIE and financial forecasting

9.1.1 Motivation of the presented research

In the first research chapter, we were motivated by the fact that many financial forecasting tools use pre-specified indicators from technical analysis. EDDIE 7, is a GP financial forecasting tool that falls in this category. The user of EDDIE 7 suggests which indicators EDDIE should use. However, in this way EDDIE 7 was restricted in using pre-specified indicators such as '12 days MA'. Nevertheless, whether '12' was the most appropriate period length for MA was questionable. This thus motivated us to look for ways to address this problem.

9.1.2 Novelty of the presented research

In order to overcome EDDIE 7's drawback, we proposed a new version, called EDDIE 8. This version used a new BNF grammar, which allowed the GP to look in the space of technical indicators. More specifically, instead of using the technical indicators as constants of the system (like EDDIE 7 and other similar forecasting tools do), EDDIE 8 used a function which takes two children, namely the indicator and the period length, the latter being a number within a parameterized

length defined by the user. As a result, EDDIE 8 came up with new solutions, which EDDIE 7 could have never come up with.

9.1.3 Conclusions

In order to show the value of the new version (EDDIE 8), we compared it with EDDIE 7. Results showed that the new version was able to take advantage of the enlarged search space and thus come up with improved trees. In addition, the best tree produced from EDDIE 8 could on average outperform the best tree from EDDIE 7. This was characterized as a significant finding, because it demonstrated the value of the new version.

However, experimental results also showed that on average, EDDIE 8 was outperformed by its predecessor in terms of fitness and the other metrics. Analysis in both empirical and artificial datasets showed that there could be cases where EDDIE 8 had difficulties searching effectively. This behavior was observed when solutions came from EDDIE 7's search space. When this happened, EDDIE 8 was having difficulties focusing its search on such a narrow search space, and could not always return results that would outperform EDDIE 7. It was obvious that there was a trade-off between 'discovering new solutions' and 'effective 'search'.

Nevertheless, we should again emphasize that the best tree of EDDIE 8 is better than the best tree from EDDIE 7. We consider this as a very important contribution, because practitioners would be interested in the best results of the algorithm, and not necessarily in the average results. Therefore, since EDDIE 8's best tree can on average perform better than EDDIE 7's in terms of *all performance metrics*, this means that EDDIE 8 can offer higher profit to its user. We thus believe that this allows us to characterize EDDIE 8 as an important contribution to the literature of financial forecasting tools.

9.2 Summary of the Market Fraction Hypothesis

9.2.1 Motivation of the presented research

In the following chapter, we focused on observations regarding the fraction dynamics of financial markets, which have led to the Market Fraction Hypothesis (MFH). This hypothesis states that the fraction between the trading strategy types that exist in a financial market changes continuously over time. This was an important observation, because it suggests that a 'winner' type of trading strategy cannot exist in the long run. However, this observation had neither been formalized before, nor tested under 'real' data.[1] This thus motivated us to formalize the hypothesis, and test it under real data. Investigating the hypothesis would also allow us to make valuable conclusions regarding the market fraction dynamics.

[1] Actually, there is a limited number of works that has observed the MFH under real data. However, these observations were made under either N-type or autonomous agent-based models, which as we have already explained make strict assumptions.

9.2.2 Novelty of the presented research

We first formalized the hypothesis, by presenting its main constituents. Formalizing the MFH was an important task, because this made it possible to test the hypothesis. We then proceeded by suggesting a testing methodology. In addition, we suggested a new agent-based financial model, which used Genetic Programming as a rule inference engine, and Self-Organizing Map as a clustering machine. The novelty of this model was twofold: first, it did not assume the existence of pre-fixed types of strategies, like the traditional N-type models do (e.g., fundamentalist-chartist model). Secondly, our agent-based model allowed trading strategies that belong to the same type to be heterogeneous, while they could behave in a similar way. The last contribution on this chapter was that we tested the plausibility of the MFH under 10 international markets.

Despite the novelty of our work, we should also acknowledge its limitations. We mentioned in Chapter 7 that our agent-based model creates and evolves dumb and naive trading strategies, which might not necessarily represent the actual naive strategies that existed in the real market. Throughout our work we have assumed that this is not the case and that the naive strategies evolved behave like the ones in the real markets. This could of course be considered as a limitation of our agent-based model.

9.2.3 Conclusions

The experimental results gave us valuable insight about the dynamics of market fraction. Our findings suggest that the properties of the hypothesis do not hold for the majority of the cases tested. We found that even in the long run, the market tends to favor few types of agents. More specifically, we observed that we only needed four to five types (five to six types), to account for the behavior of 90% (95%) of market participants. This is a very important finding, because it indicates that even if a high number of trading strategy types exists, the traders focus only on a few of these available types. We also found that in general, most types of agents are usually dominant for a period of up to 2 years, with a few exceptions managing to remain dominant for a period of 5.5 years. The latter was characterized as 'long' dominance, and thus led us to conclude that evidence for the MFH was quite weak. Lastly, this long dominance allowed us to argue that there can be winning types of trading strategies. We can thus conclude that the MFH observations made under 'real' markets differ significantly from observations made in the past, under artificial market frameworks (see Chen et al, 2012). Nevertheless, the investigation of this hypothesis has offered us valuable information about the dynamics of financial markets.

After these tests, we were interested in examining whether these results could hold under different GP algorithms. If they indeed held, that would allow us to argue that the results from our tests are independent of the GP algorithm. We hence used EDDIE 7 and EDDIE 8, to test the previous derived results of the Market Fraction Hypothesis. What we observed was that the same patterns existed across both test tests that we used, under all three GP algorithms. Therefore, our results seem to be insensitive to the choice of the algorithm.

9.3 Summary of the Dinosaur Hypothesis

9.3.1 Motivation of the presented research

Finally, the last research chapter presented, tested and discussed the Dinosaur Hypothesis (DH), which states that the behavior of a market never settles down and that the strategies in this market continuously co-evolve with it. This observation was first made by Arthur (1992) and later by Chen and Yeh (2001). However, these models made this observation under artificial stock market frameworks. In our current work, we were interested in examining whether these observations could also hold in the real world. This thus motivated us to examine the behavior of different financial markets and see whether the findings of Arthur, Chen and Yeh could still hold. In addition, we were interested in formalizing the DH, since this had not happened before. Doing the above was very important, because it would allows us to have a better understanding of how the financial markets behave in the long run.

9.3.2 Novelty of the presented research

In order to investigate the market behavior, we firstly formalized the hypothesis by presenting its main constituents. As with the MFH, formalizing the DH was an important task because this allowed us to test for the plausibility of the hypothesis. We again employed GP as the rule inference engine, and SOM as the clustering machine. In addition, we extended our agent-based financial model from the previous chapter by allowing the SOM clusters to change throughout time, rather than being static as was the case in the MFH framework. This offered more realism to our model and allowed us to investigate complex market behavior dynamics. Finally, we tested the plausibility of the DH under 10 international markets.

9.3.3 Conclusions

Results showed that parts of the Dinosaur Hypothesis were supported by our data. More specifically, we found that agents that do not adapt to changes that happen in the markets cannot fit their environment any more. In other words, these agents, and as a consequence their trading strategies, become obsolete or dinosaurs. However, only a few markets experienced a continuous decrease in dinosaurs' performance. Our tests thus did not confirm the observation made by Chen and Yeh regarding the continuous decrease in the performance of dinosaurs.

In addition to the standard testing of the hypothesis, where we used the *simple GP*, we also presented tests on the DH under two different GP algorithms. The reason for doing this was that we again wanted to demonstrate that the results under the *simple GP* are rigorous and not algorithm-dependent. We thus ran the same tests under EDDIE 7 and EDDIE 8. Results showed that both versions agree with the original results. This therefore strengthened our argument of non-returning dinosaurs in financial markets, and allowed us to argue that our results are not algorithm-dependent.

The implications of these findings are very important, because they indicate that the behavior of financial markets is non-stationary. In addition, trading strategies seem to have a finite lifetime, after which they become ineffective. Trading strategies need to continuously co-evolve with the market in order to remain effective.

9.4 Future work

The work on any of the above projects has by no means finished. Many extensions could take place. In the first research project, some work has already been done by using hyper-heuristics (Kampouridis and Tsang, 2011) for improving EDDIE 8's search effectiveness. We aim to focus towards that direction. More specifically, we are planning to test different heuristics (e.g. hill-climbing, tree element swaps, period mutation) to investigate which type of heuristics can have the greatest improvement effect on EDDIE 8. This would then allow us to combine certain heuristics into a hyper-heuristics framework, which would then utilize all previously successful heuristics. We believe that this approach would considerably improve the performance of EDDIE 8 and that it would also give us valuable insight of how to search in large search spaces, while at the same time focusing on the small, but also important areas of the space. Lastly, successful results in the above project would allow us to further extend the grammar of EDDIE, by not allowing any pre-specified indicators at all. Thus, the user would not have to give an indicator, such as Moving Average, as input. On the contrary, the GP would use mathematical functions that can re-create the Moving Average. This would then lead to the creation of completely new indicators that experts in financial markets have never been aware of.

Regarding the work on the MFH, there are still several questions to be answered. First of all, are our results sensitive to the other control parameters of our experiments, such as population size, crossover rate, mutation rate, fitness function, and so on? In addition, we discussed the sensitivity of the results to different GP algorithms. However, we did not do the same with SOMs. Could different results be produced if we followed a different SOM clustering method? This also deserves further investigation.

In addition, we mentioned at the end of Chapter 7 that a limitation of our agent-based financial model is the absence of the traditional approach of estimation. This is because we cannot know how close the inferred strategies are to the real strategies that have existed in the market. We also mentioned that it would be interesting as a future work to investigate this issue. One way of conducting the above investigation would be to first obtain the trading strategies of 'real' traders. This is not of course straight-forward, because this kind of information is usually not available. However, there are works in the literature that have used human subjects to act as traders. An example such work is Hommes et al (2008). In this study, each participant was making a prediction regarding the price movement. These predictions led to the formulation of the market price. We could thus apply our GP system to the market prices generated in Hommes et al (2008), and infer trading strategies. Then, we could examine how close the forecasts of the inferred strategies are to the forecasts of the real market participants. This could be achieved by using an objective function

which compares the forecasts made by the GP system to the forecasts made by the real traders.

Furthermore, another future plan that we consider interesting is to understand the underlying trading strategy types behind the SOM clusters. This would then allow us to understand how each cluster behaves. We could then compare this behavior to the behavior of other well-known strategies, such as the chartist and the contrarian. This would thus give us an understanding of where each cluster "stands" in terms of behavior and would therefore give us insight as to what makes a strategy type successful and popular.

Moreover, we should say that the above suggestions for future work in the MFH field also apply to the DH work, since both hypotheses were tested under the same financial model and the same framework.

Lastly, we mentioned in Chapter 8 that although dinosaurs do not return, there can be times we observe returning lizards, which means that the market conditions between two different periods resemble, and thus allowed trading strategies from the past to feel 'satisfied' again. We consider as an interesting future work to examine the underlying market conditions between these periods and look for similarities that might exist. For instance, one way to do this could be to examine the statistical properties and stylized facts of these markets, such as calendar effect, skewness, kurtosis, and so on. Finding similarities between these stylized facts could give as an indication of why strategies behave similarly in certain time periods, and thus anticipate this behavior in future periods.

References

Abdelmalek W, Hamida S, Abid F (2009) Selecting the best forecasting-implied volatility model using genetic programming. Journal of Applied Mathematics and Decision Sciences 33

Agapitos A (2010) The evolution of recursive algorithms and object-oriented programs. PhD thesis, School of Computer Science and Electronic Engineering, University of Essex 11, 12

Agapitos A, O'Neill M, Brabazon A (2010) Evolutionary learning of technical trading rules without data-mining bias. In: Schaefer R, Cotta C, Kołodziej J, Rudolph G (eds) Parallel Problem Solving from Nature – PPSN XI, Springer, Lecture Notes in Computer Science, vol 6238, pp 294–303 33, 34

Allen F, Karjalainen R (1999) Using genetic algorithms to find technical trading rules. Journal of Financial Economics 51:245–271 33, 49

Altenberg L (1994) The evolution of evolvability in genetic programming, MIT Press, pp 47–74 12

Amilon H (2008) Estimation of an adaptive stock market model with heterogeneous agents. Journal of Empirical Finance 15 (2):342–362 44

Aoki M (2002) Open models of share markets with two dominant types of participants. Journal of Economic Behavior and Organization 49(2):199–216 93

Appel G (2005) Technical analysis. Power tools for active investors. Pearson Education 29, 31

Arthur B (1992) On learning and adaptation in the economy, working paper 92-07-038, Santa Fe Institute 45, 101, 104, 117, 118, 130

Arthur W, Holland J, LeBaron B, Palmer R, Tayler P (1997) Asset pricing under endogenous expectations in an artificial stock market. In: Arthur B, Durlauf S, Lane D (eds) The Economy as an Evolving Complex System II, Addison-Wesley, Reading, MA, pp 15–44 42

Austin M, Bates G, Dempster M, Leemans V, Williams S (2004) Adaptive systems for foreign exchange trading. Quantitative Finance 4(4):37–45 49

Azoff E (1994) Neural Network Time Series Forecasting of Financial Markets. John Wiley & Sons, Inc., New York, NY, USA 32

Backus J (1959) The syntax and semantics of the proposed international algebraic language of Zurich. In: International Conference on Information Processing, UNESCO, pp 125–132 13

Badawy F, Abdelazim H, Darwish M (2005) Genetic algorithms for predicting the egyptian stock market. In: Information and Communications Technology, 2005. Enabling Technologies for the New Knowledge Society: ITI 3rd International Conference on, pp 109 –122, DOI 10.1109/ITICT.2005.1609619 33

Banzhaf W (1993) Genetic programming for pedestrians. In: Forrest S (ed) Proceedings of the 5th International Conference on Genetic Algorithms, Morgan Kaufmann, University of Illinois at Urbana-Champaign, p 628 8

Banzhaf W, Nordina P, Keller R, Francone F (1998) Genetic Programming: An Introduction: On the Automatic Evolution of Computer Programs and its Applications. Heidelberg and Morgan Kaufmann 7, 9

Bauer R (1994) Genetic Algorithms and Investment Strategies. John Wiley & Sons 33

Becker L, Seshadri M (2003) GP-evolved technical trading rules can outperform buy and hold. In: Sixth International Conference on Computational Intelligence and Natural Computing, Embassy Suites Hotel and Conference Center, Cary, North Carolina USA 34

Bernal-Urbina M, Flores-Méndez A (2008) Time series forecasting through polynomial artificial neural networks and genetic programming. In: Proceedings of the IEEE Congress on Evolutionary Computation, Hong Kong, pp 3324–3329 33

Bhattacharyya S, Pictet O, Zumbach G (2002) Knowledge-intensive genetic discovery in foreign exchange markets. Evolutionary Computation, IEEE Transactions on 6(2):169 –181, DOI 10.1109/4235.996016 33, 34

Bishop C (1995) Neural Networks for Pattern Recognition. Oxford University Press, Oxford 15

Bohm W, Geyer-Schulz A (1996) Exact uniform initialization for genetic programming. In: Belew R, Vose M (eds) Foundations of Genetic Algorithms IV, University of San Diego, CA, USA: Morgan Kaufmann, pp 379–407 9

Boswijk P, Hommes C, Manzan S (2007) Behavioral heterogeneity in stock prices. Journal of Economic Dynamics and Control 31(6):1938–1970 44, 78

Brabazon A, O'Neill M (2004) Evolving technical trading rules for spot foreign-exchange markets using grammatical evolution. Computational Management Science 1 (3-4):311–328 34

Bradeley R, Brabazon A, O'Neill M (2010) Evolving trading rule-based policies. In: EvoFIN 2010 the 4th European Event on Evolutionary and Natural Computation in Finance and Economics, Springer 34

Brock W, Hommes C (1997) A rational route to randomness. Econometrica 65:1059–1095 42

Brock W, Hommes C (1998) Heterogeneous beliefs and routes to chaos in a simple asset pricing model. Journal of Economic Dynamics and Control 22:1235–1274 41, 42, 44

Brock W, Lakonishok J, LeBaron B (1992) Simple technical trading rules and the stochastic properties of stock returns. Journal of Finance 47(5):1731–1764 28, 29, 30

Brock W, Hommes C, Wagener F (2005) Evolutionary dynamics in markets with many trader types. Journal of Mathematical Economics 41:7–42 42

Burke E, Hart E, Kendall G, Newall J, Ross P, Schulenburg S (2003) Hyper-heuristics: An emerging direction in modern search technology, Kluwer, pp 457–474 74

Burke E, MacCloumn B, Meisels A, Petrovic S, Qu R (2006) A graph-based hyper heuristic for timetabling problems. European Journal of Operational Research 176:177–192 74

Cao L, Tay F (2001) Financial forecasting using support vector machines. Neural Computing & Applications 10:184–192 34

Cao L, Tay F (2006) Support vector machine with adaptive parameters in financial time series forecasting. IEEE Transactions on Neural Networks 14:1506–1518 34

Chan N, LeBaron B, Lo A, Poggio T (1999) Agent-based models of financial markets: A comparison with experimental markets. MIT Artificial Markets Project Paper No. 124 40

Chang PC, Fan CY, Liu CH (2009) Integrating a piecewise linear representation method and a neural network model for stock trading points prediction. Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on 39(1):80 –92, DOI 10.1109/TSMCC.2008.2007255 32

Chellapilla K (1997) Evolving computer programs without subtree crossover. IEEE Transactions on Evolutionary Computation 1(3):209–216 9

Chen SH (2008) Financial Applications: Stock Markets, Wiley Encyclopedia of Computer Science and Engineering, John Wiley & Sons, Inc, pp 481–498 44, 46, 101

Chen SH, Yeh CH (2001) Evolving traders and the business school with genetic programming: A new architecture of the agent-based artificial stock market. Journal of Economic Dynamics & Control 25:363–393 45, 101, 104, 117, 118, 130

Chen SH, Huang YC, Wang JF (2009) Bounded rationality and the elasticity puzzle: An analysis of agent-based computational consumption capital asset pricing models. In: Zambelli, S. (ed.), Routledge 42

Chen SH, Kampouridis M, Tsang E (2010) Microstructure dynamics and agent-based financial markets. In: Bosse T, Geller A, Jonker CM (eds) Proceedings of the 11th International Workshop on Multi-Agent-Based Simulation (MABS 2010), Toronto, Canada, pp 117–128 v

Chen SH, Kampouridis M, Tsang E (2011) Microstructure dynamics and agent-based financial markets. In: Bosse T, Geller A, Jonker CM (eds) Multi-Agent-Based Simulation XI, 11th International Workshop, Revised Papers, LNAI, Springer-Verlag, Berlin Heidelberg, vol 6532, pp 121–135 v

Chen SH, Chang CL, Du YR (2012) Agent-based economic models and econometrics. Journal of Knowledge Engineering Review (forthcoming) 40, 41, 42, 43, 44, 77, 78, 79, 97, 129

Chen Y, Mabu S, Hirasawa K, Hu J (2007) Genetic network programming with sarsa learning and its application to creating stock trading rules. In: Proceedings of the IEEE Conference on Evolutionary Computation, Singapore, pp 220–237 34

Cheong C, Kim YJ, Yoon SM (2011) Can we predict exchange rate movements at short horizons? Journal of Forecasting pp n/a–n/a, DOI 10.1002/for.1236, URL http://dx.doi.org/10.1002/for.1236 28

Cont R (2001) Empirical properties of asset returns: Stylized facts and statistical issues. Quantitative Finance 1 (2):223–236 40

Cramer N (1985) A representation for the adaptive generation of simple sequential programs. In: Grefenstette J (ed) Proceedings of an International Conference on Genetic Algorithms and the Applications, Carnegie-Mellon University, Pittsburgh, PA, USA, pp 183–187 8

De Gooijer J, Hyndman R (2006) 25 years of time series forecasting. International Journal of Forecasting 2 (3):443–473 28

Deboeck G, Kohonen T (eds) (1998) Visual Explorations in Finance with Self-Organizing Maps. Springer 15, 22

Dempsey I, O'Neill M, Brabazon A (2004) Live trading with grammatical evolution. In: Proceedings of the Grammatical Evolution Workshop 34

Dempster M, Jones C (2000) A real-time adaptive trading system using genetic programming. Quantitative Finance 1:397–413 34

Dempster M, Leemans V (2006) An automated FX trading system using adaptive reinforcement learning. Expert Systems with Applications, Special Issue on Financial Engineering 30 (3):543–552 34

Dempster M, Romahi YS (2002) Intraday FX trading: An evolutionary reinforcement learning approach. In: H. Yin, N. M. Allinson, R. T. Freeman, J. A. Keane, and S. J. Hubbard, editors, Proceedings of Third International Conference on Intelligent Data Engineering and Automated Learning - IDEAL 2002, vol 2412 of Lecture Notes in Computer Science, pp 347–358 34

Dempster M, Payne T, Romahi Y, Thompson G (2001) Computational learning techniques for intraday FX trading using popular technical indicators. Neural Networks, IEEE Transactions on 12(4):744 –754, DOI 10.1109/72.935088 33, 34

Dicks C, Van der Weide R (2005) Herding asynchronous updating and heterogeneity in memory in a CBS. Journal of Economic Dynamics and Control 29 (4):741–763 42

Diez-Roux A (2002) A glossary for multilevel analysis. Journal of Epidemiology and Community Health 56:588–594 78

Dittenbach M, Rauber A, Merkl D (2001) Recent advances with the growing hierarchical self-organizing map. In: Allinson N, Yin H, Allinson L, Slack J (eds) Proceedings of the 3rd Workshop on Self-Organizing Maps, Springer, Lincoln, England, Advances in Self-Organizing Maps, pp 140–145 98

Duffy J, Engle-Warnick J (2002) Using symbolic regression to infer strategies from experimental data, Springer, pp 61–82. Evolutionary Computation in Economics and Finance 79

Edwards R, Magee J (1992) Technical analysis of stock trends. New York Institute of Finance 2, 29, 30, 36

Fama EF (1965) Random walks in stock market prices. Financial Analysts Journal 21(5):55–59 27

Fama EF (1972) Components of investment performance. Journal of Finance XXVII(5):551–567 28

Fan A, Palaniswami M (2000) Selecting bankruptcy predictors using a support vector machine approach. In: IEEE-INNS-ENNS International Joint Conference on Neural Networks, vol 6, pp 354–359 34

Fausett L (1994) Fundamentals of Neural Networks. Architectures, algorithms, and applications. Prentice Hall International, Inc 15, 32

Garcia Almanza A (2008) New classification methods for gathering patterns in the context of genetic programming. PhD thesis, Department of Computer Science, University of Essex 8, 9, 10, 11, 30, 35

Garcia Almanza A, Tsang E (2007) Detection of stock price movements using chance discovery and genetic programming. International Journal of Knowledge-based and Intelligent Engineering Systems 11(5):329–344 28, 33, 35, 53

Garson D (1998) Neural Networks. An Introductory Guide for Social Scientists. SAGE Publications 15

Gestel T, Suykens J, Baestaens DE, Lambrechts A, Lanckriet G, Vandaele B, Moor B, Vandewalle J (2001) Financial time series prediction using least squares support vector machines within the evidence framework. IEEE Transactions on Neural Networks 12:809–821 34

Gestel T, Baesens B, Suykens J, Espinoza M, Baestaens DE, Vanthienen J, Moor B (2003) Bankruptcy prediction with least squares support vector machine classifiers. In: IEEE International Conference on Computational Intelligence for Financial Engineering, pp 1–8 34

Gigerenzer G, Todd P (1999) Fast and Frugal Heuristics: The Adaptive Toolbox, Oxford University Press, pp 3–34. Gigerenzer, G. Todd, P. and the ABC Research Group 79

Gilli M, Winker P (2003) A global optimization heuristic for estimating agent-based models. Journal of Computational Statistics and Data Analysis 42(3):299–312 44, 78

Godlberg D (1989) Genetic Algorithms in Search Optimisation and Machine Learning. Addison-Wesley 11

Gruau F (1996) On using syntactic constraints with genetic programming, MIT Press, Cambridge, MA, pp 377–394. Advances in Genetic Programming II 13

Gurney K (1997) An Introduction to Neural Networks. Routledge, London 15

Ham F, Kostanic I (2001) Principles of Neurocomputing for Science & Engineering. McGraw-Hill, Inc 15

Hart E, Ross P, Nelson J (1998) Solving a real-world problem using an evolving heuristically driven schedule builder. Evol Comput 6(1):61–80 74

Hassoun M (1995) Fundamentals of artificial neural networks. MIT Press, Cambrdige, Mass 15

Haykin S (1999) Neural Networks. A comprehensive foundation, 2nd edn. Prentice Hall 15, 20

Hayward S (2006) Genetically optimised artificial neural network for financial time series data mining. Computing in Economics and Finance 2006 417, Society for Computational Economics 32

Holland J (1975) Adaptation in Natural and Artificial Systems. University of Michigan Press, Ann Arbor, MI 33

Hommes C, Sonnemans J, Tuinstra J, van de Velden H (2008) Expectations and bubbles in asset pricing experiments. Journal of Economic Behavior & Organization 67(1):116–133, URL http://ideas.repec.org/a/eee/jeborg/v67y2008i1p116-133.html 131

Hou ZE, Duan FJ (2009) The neural network method of financial forecasting. In: Intelligent Systems and Applications, 2009. ISA 2009. International Workshop on, pp 1 –3, DOI 10.1109/IWISA.2009.5073193 32

Huang W, Nakamori Y, Wang SY (2004a) Forecasting stock market movement direction with support vector machine. Computers & Operations Research 34

Huang Z, Chen H, Hsu CJ, Chen WH, Wu S (2004b) Credit rating analysis with support vector machines and neural networks: a market comparative study. Decision Support Systems 37:543–558 34

Iba H (1996) Random tree generation for genetic programming. In: Voigt HM, Ebeling W, Rechenberg I, Schwefel HP (eds) Proceedings of the International Conference on Evolutionary Computation, Springer Verlag, Berlin, Germany, LNCS, pp 144–153 9

Irwin S, Park CH (2007) What do we know about the profitability of technical analysis? Journal of Economic Surveys 21(4):768–826 29

Izumi K, Okatsu T (1996) An artificial market analysis of exchange rate dynamics. In: Fogel L J BT Angeline P J (ed) Evolutionary Programming V, MIT Press, pp 27–36 42

Izumi K, Ueda K (1999) Analysis of dealers' processing financial news based on an artificial market approach. Journal of Computational Intelligence in Finance 7:23–33 42, 79

Kahn M (2006) Technical analysis plain and simple, 2nd edn. Pearson Education Limited, Great Britain 29

Kampouridis M, Tsang E (2009) EDDIE on artificial dataset, Technical Report CES-492, School of Computer Science and Electronic Engineering, University of Essex, UK. Available from http://www.kampouridis.net/publications vi

Kampouridis M, Tsang E (2010) EDDIE for investment opportunities forecasting: Extending the search space of the GP. In: Proceedings of the IEEE Conference on Evolutionary Computation, Barcelona, Spain, pp 2019–2026 iv

Kampouridis M, Tsang E (2011) Using hyperheuristics under a GP framework for financial forecasting. In: Coello Coello CA (ed) Proc. Fifth International Conference on Learning and Intelligent Optimization (LION5), Springer, Lecture Notes in Computer Science, forthcoming iv, 74, 131

Kampouridis M, Chen SH, Tsang E (2008) Hyper-heuristics for investment opportunities forecasting. In: Computational Management Science, London, Also available from http://kampouridis.net/publications/ v

Kampouridis M, Chen SH, Tsang E (2009a) Market Fraction Hypothesis: A proposed test. In: The 9th Asia-Pacific Complex Systems Conference, Tokyo, Also available from http://kampouridis.net/publications/ v

Kampouridis M, Chen SH, Tsang E (2009b) Market Fraction Hypothesis: A proposed test. In: Computational Management Science, Geneva, Also available from http://kampouridis.net/publications/ vi

Kampouridis M, Chen SH, Tsang E (2009c) A summary for the Brock and Hommes 'Heterogeneous beliefs and routes to chaos in a simple asset pricing model' 1998 JEDC paper, Technical Report CES-497, School of Computer Science and Electronic Engineering, University of Essex, UK. Available from http://www.kampouridis.net/publications vi

Kampouridis M, Chen SH, Tsang E (2010a) Market Fraction Hypothesis: A proposed test. In: Econophysics Colloquium, Taipei v

Kampouridis M, Chen SH, Tsang E (2010b) Testing the dinosaur hypothesis under different GP algorithms. In: Proceedings of the UK Computational Intelligence Workshop (UKCI), IEEE Xplore, Essex, pp 1–7 v

Kampouridis M, Chen SH, Tsang E (2010c) Testing the dinosaur hypothesis under empirical datasets. In: Schaefer R, Cotta C, Kołodziej J, Rudolph G (eds) Parallel Problem Solving from Nature – PPSN XI, Springer, Lecture Notes in Computer Science, vol 6239, pp 199–208 v

Kampouridis M, Chen SH, Tsang E (2011a) Investigating the effect of different GP algorithms on the non-stationary behavior of financial markets. In: Computational Intelligence for Financial Engineering and Economics, IEEE Press, IEEE Symposium Series on Computational Intelligence, p forthcoming v

Kampouridis M, Chen SH, Tsang E (2011b) Market fraction hypothesis: A proposed test. International Review of Financial Analysis, special issue on Complexity and Non-Linearities in Financial Markets: Perspectives from Econophysics, forthcoming iv

Kampouridis M, Chen SH, Tsang E (2011c) The Market Fraction Hypothesis under different GP algorithms, IGI Global. Information Systems for Global Financial Markets: Emerging Developments and Effects, forthcoming iv

Kampouridis M, Chen SH, Tsang E (2011d) Market Microstructure: A Self-Organizing Map Approach for Investigating Behavior Dynamics under an Evolutionary Environment, Springer. Natural Computing in Computational Finance, Volume 4, Studies in Computational Intelligence Series, forthcoming iv

Kampouridis M, Chen SH, Tsang E (2011e) Market Microstructure: Can dinosaurs return? A self-organizing map approach under an evolutionary framework. In: Di Chio et al C (ed) EvoApplications 2011, Part II, Springer, Heidelberg, Lecture Notes in Computer Science, pp 91–100 v

Keller D (2007) Breakthroughs in Technical analysis. New thinking from the world's top minds. Bloomberg Press, New York 29

Kim K (2006) Artificial neural networks with evolutionary instance selection for financial forecasting. Expert Systems with Applications 30:519–526 32

Kirkpatrick D, Dahlquist J (2006) Technical analysis: The complete resource for financial market technicians. Financial Times Press 29

Kirman A (1991) Epidemics of Opinion and Speculative Bubbles in Financial Markets, M. Taylor (Hrsg.), "Money and Financial Markets", London, Macmillan, pp 354–368 41, 44

Kirman A (1993) Ants, rationality and recruitment. Quarterly Journal of Economics 108(1):137–156 41, 44, 77

Klein J (1997) Statistical visions in time. Cambridge University Press 27

Kodratoff Y, Michalski R, Garbonell J, Mitchell T (1990) Machine learning: an artificial intelligence approach. Morgan Kaufmann Publishers, Inc 32

Kohonen T (1982) Self-organized formation of topologically correct feature maps. Journal of Biological Cybernetics 43:59–69 15, 18, 20, 76

Kohonen T (2001) Self-Organizing Maps, 3rd edn. Springer Series in Information Sciences, Springer 15, 17, 18, 19, 20, 21, 23, 24

Koza J (1992) Genetic Programming: On the programming of computers by means of natural selection. Cambridge, MA: MIT Press 7, 8, 12, 76

Koza J (1994) Genetic Programming II: Automatic Discovery of Reusable Programs. Cambridge, MA: MIT Press 7, 13, 51

Koza J, Andre D, Bennett III F, Keane M (1999) GeneticProgramming 3: Darwinian Invention and Problem Solving. Morgan Kaufman 7

Koza J, Keane M, Streeter M, Mydlowec W, Yu J, Lanza G (2003) Genetic Programming IV: Routine Human-Competitive Machine Intelligence. Kluwer Academic Publishers 7

Kwon YK, Moon BR (2007) A hybrid neurogenetic approach for stock forecasting. Neural Networks, IEEE Transactions on 18(3):851 –864, DOI 10.1109/TNN.2007.891629 32

Lam M (2004) Neural network techniques for financial performance prediction: integrating fundamental and technical analysis. Decision Support Systems 37(4):567–581 32

Langdon W (1998) Genetic programming and data structures: Genetic programming + data structures = automatic programming! Kluwer, Boston, Genetic Programming, vol 1 11

Langdon W (2009) Evolving data structures with genetic programming. ICGA pp 295–302 11

LeBaron B (1998) Technical trading rules and regime shifts in foreign exchange. Butter-worth Heinemann 29

Lee C, Yao X (2004) Evolutionary programming using the mutations based on the Lévy probability distribution. IEEE Transactions on Evolutionary Computation 8(1):1–13 9

Leigh W, Purvis R, Ragusa J (2002) Forecasting the NYSE composite index with technical analysis, pattern recognizer, neural networks, and genetic algorithm: a case study in romantic decision support. Decision Support Systems 32:361–377 33

Li J (2001) FGP: A genetic programming-ased financial forecasting tool. PhD thesis, Department of Computer Science, University of Essex xvii, 35, 38, 39, 48, 49, 54, 150

Li J, Tsang E (1999a) Improving technical analysis predictions: An application of genetic programming. In: The 12th International FLAIRS Conference, USA, pp 108–112 13, 35, 48, 49

Li J, Tsang E (1999b) Investment decision making using FGP: a case study. In: Proceedings of Congress on Evolutionary Computation 35

Li J, Tsang E (2000) Reducing failures in investment recommendations using genetic programming. In: The 6th International Conference on Computing in Economics and Finance, Society for Computational Economics, Barcelona 35

Liu SM, Chou CH (2003) Parities and spread trading in gold and silver markets: A fractional cointegration analysis. Applied Financial Economics 13:12:899–911 28

Liu Y, Yao X (2001) Evolving neural networks for Hang Seng Stock Index Forecast. In: Proceedings of the IEEE Congress on Evolutionary Computation, USA, pp 256–260 32

Lo A (2004) The adaptive market hypothesis: market efficiency from an evolutionary perspective. Journal of Portfolio Management 30:15–29 43, 78

Lo A (2005) Reconciling efficient markets with behavioral finance: The adaptive markets hypothesis. Journal of Investment Consulting 2:21–44 43, 78

Lo A, Hasanhodzic J (2010) The evolution of technical analysis: Financial prediction from babylonian tablets toBloomberg terminals. Bloomberg Press 29

Lo A, Mamaysky H, Wang J (2000) Foundations of technical analysis: computational algorithms, statistical inference, and empirical implementation. Journal of Finance LV (4):1705–1765 28

Lohpetch D, Corne D (2009) Discovering effective technical trading rules with genetic programming: towards robustly outperforming buy-and-hold. In: Nature Biologically Inspired Computing, 2009. NaBIC 2009. World Congress on, pp 439 –444, DOI 10.1109/NABIC.2009.5393324 33

Lux T (1995) Herd behavior, bubbles and crashes. Economic Journal 105:880–896 41, 44

Lux T (1997) Time variation of second moments from a noise trader/infection model. Journal of Economic Dynamics and Control 22:1–38 41, 44

Lux T (1998) The socio-economic dynamics of speculative markets: Interacting agents, chaos and the fat tails of return distributions. Journal of Economic Behavior and Organization 33:143–165 41, 44

MacQueen JB (1967) Some methods for classification and analysis of multivariate observations. In: Proceedings of the 5th Berkeley Symposium on Mathematical Statistics and Probability, University of California Press, pp 281–297 80

Mani S (1996) Financial forecasting using genetic algorithms. Applied Artificial Intelligence 10:543–566 33

Martinez-Jaramillo S (2007) Artificial financial markets: An agent-based approach to reproduce stylized facts and to study the red queen effect. PhD thesis, CFFEA, University of Essex 33, 48, 49

Martinez-Jaramillo S, Tsang E (2009) A heterogeneous, endogenous and co-evolutionary GP-based financial market. IEEE Transactions on Evolutionary Computation 13(1):33–55 30

MathWorks T (2011) Self organizing feature maps. http://www.mathworks.com/access/helpdesk/help/toolbox/nnet/self_or4.html xv, 18, 19, 23, 24, 82, 83

Matas JM, Reboredo JC (2011) Forecasting performance of nonlinear models for intraday stock returns. Journal of Forecasting p forthcoming 28

McMillan DG, Speight AEH (2011) Daily FX volatility forecasts: Can the garch(1,1) model be beaten using high-frequency data? Journal of Forecasting p forthcoming 28

McPhee N, Hopper N (1999) Analysis of genetic diversity through population history. In: Banszhaf W, Daida J, Eiben A, Garzon M, Honavar V, Jakiela M, Smith R (eds) Proceedings of the Genetic and Evolutionary Computation Conference, Morgan Kaufmann, Orlando, Florida, USA, Genetic Programming, vol 2, pp 1112–1120 11

Michalewicz Z (2002) Genetic algorithms + data structures = evolution programs. Springer 33

Miller J (1999) An empirical study of the efficiency of learning boolean functions using a cartesian genetic programming approach. In: Banzhaf W, Daida J, Eiben A, Garzon M, Honavar V, Jakiela M, Smith R (eds) Proceedings of the Genetic and Evolutionary Computation Conference, Morgan Kaufmann, Orlando, Florida, USA, vol 2, pp 1135–1142 8

Miller J, Thomson P (2000) Cartesian genetic programming. In: Poli R, Banzhaf W, Langdon W, Miller J, Nordin P, TC F (eds) Proceedings of EuroGP2000, Springer-Verlag, Edinburgh, LNCS, vol 1802, pp 121–132 8

Mills T (2002a) Forecasting Financial Markets, The International Library of Critical Writings in Economics, vol I. Edward Elgar Publishing 27, 28

Mills T (2002b) Forecasting Financial Markets, The International Library of Critical Writings in Economics, vol II. Edward Elgar Publishing 28

Mitchell T (1997) Machine learning. McGraw-Hill, Boston 2

Montana D (1995) Strongly typed genetic programming. Evolutionary Computation 3(2):199–230 13

Mulier F, Cherkassky V (1994) Learning rate schedules for self-organizing maps. In: Pattern Recognition, 1994. Vol. 2 - Conference B: Computer Vision Image Processing., Proceedings of the 12th IAPR International. Conference on, vol 2, pp 224 –228 vol.2, DOI 10.1109/ICPR.1994.576908 20

Murphy J (1999) Technical analysis of the financial markets. New York Institute of Finance 29, 30

Nordern G (2006) Technical analysis and the active trader. McGraw-Hill 29

O'Neill M, Brabazon T, Ryan C, Collins J (2001) Evolving market index trading rules using grammatical evolution. In: Proceedings of EvoIASP 33, 34

Özcan E, Bilgin B, Korkmaz EE (2008) A comprehensive analysis of hyper-heuristics. Intelligent Data Analysis 12(1):3–23 74

Page J, Poli R, Langdon W (1999) Smooth uniform crossover with smooth point mutation in genetic programming: A preliminary study. In: Poli R, Nordin P, Langdon W, Fogarty T (eds) Genetic Programming, Proceedings of EuroGP99, Springer-Verlag, Goteborg, Sweden, LNCS, vol 1598, pp 33–49 9

Palmer R, Arthur W, Holland J, LeBaron B, Tayler P (1994) Artificial economic life: a simple model of a stock market. Physica D 75:264–274 42

Pesaran M, Timmermann A (1995) Predictability of stock returns: robustness and economic significance. Journal of Finance L (4):1201–1228 28

Pesaran M, Timmermann A (2000) A recursive modelling approach to predicting UK stock returns. Economic Journal 110 (460):159–191 28

Poli R, Langdon W (1998) On the search properties of different crossover operators in genetic programming. In: Koza JR, Banzhaf W, Chellapilla K, Deb K, Dorigo M, Fogel DB, Garzon MH, Goldberg DE, Iba H, Riolo R (eds) Genetic Programming 1998: Proceedings of the Third Annual Conference, Morgan Kaufmann, University of Wisconsin, Madison, Wisconsin, USA, pp 293–301 9

Poli R, Langdon W, McPhee N (2008) A Field Guide to Genetic Programming. Lulu.com 7, 11, 76

Poon SH, Granger C (2003) Forecasting volatility in financial markets: A review. Journal of Economic Literature XLI:478–539 28

Poon SH, Granger C (2005) Practical issus in forecasting volatility. Financial Analysts Journal 61(1):45–56 28

Pring M (1991) Technical analysis explained: the successful investors guide to spotting investment trends and turning points. New York: McGraw-Hill Inc 30

Provost F, Kohavi R (1998) Glossary of terms. Journal of Machine Learning 30(2-3):271–274 37

Refenes AP (1994) Neural Networks in the Capital Markets. John Wiley & Sons, Inc., New York, NY, USA 32

Reynolds C (1992) An evolved, vision-based behavioral model of obstacle avoidance behaviour. In: Langton C (ed) Artificial Life III, Addison-Wesley, Santa Fe Institute, New Mexico, USA, SFI Studies in the Sciences of Complexity, vol XVII, pp 327–346 11

Robert Taylor J (1999) An Introduction to Error Analysis: The Study of Uncertainties in Physical Measurements. University Science Books 62

Roh T (2007) Forecasting the volatility of stock price index. Expert Systems with Applications 33(4):916–922 32

Ryan C (1994) Pygmies and civil servants, MIT Press, chap 11, pp 243–263 11

Ryan C, Collins J, O'Neill M (1998) Grammatical evolution: Evolving programs for an arbitrary language. In: EuroGP 33

Salah Salhi M, Arous N, Ellouze N (2009) Principal temporal extensions of SOM: Overview. International Journal of Signal Processing, Image Processing and Pattern Recognition 2(4):61–84 24

Sansone A, Garofalo G (2007) Asset price dynamics in a financial market with heterogeneous trading strategies and time delays. Physica A 382:247–257 41

Sapankevych N, Sankar R (2009) Time series prediction using support vector machines: A survey. Computational Intelligence Magazine, IEEE 4(2):24 –38, DOI 10.1109/MCI.2009.932254 34

Schulenburg S, Ross P (2001) Strength and money: An LCS approach to increasing returns, Springer Berlin / Heidelberg, pp 291–298. Lecture Notes in Computer Science 34

Schulenburg S, Ross P (2002) Explorations in LCS models of stock trading, P.L. Lanzi, W. Stolzmann, and S.W. Wilson, editors, Advances in Learning Classifier Systems: 4th International Workshop, IWLCS 2001, vol 2321, Springer Berlin / Heidelberg, pp 150–179 34

Scleifer A (2000) Inefficient markets, an introduction to behavioral finance. Oxford University Press 28

Shachmurove Y (2005) Business applications of emulative neural networks. International Journal of Business 10 32

Simon H (1956) Rational choice and the structure of environments. Psychological Review 63:129–138 79

Syswerda G (1990) A study of reproduction in generational and steady stage genetic algorithms. In: FOGA, pp 94–101 11

Taylor M, Allen H (1992) The use of technical analysis in the foreign exchange market. Journal of International Money and Finance 11:304–314 28, 29

Teller A, Veloso M (1995) Program evolution for data mining. The International Journal of Expert Systems 8(3):216–236 8

Tino P, Nikolaev N, Yao X (2005) Volatility forecasting with sparse bayesian kernel models. In: Proceedings of the 4th International Conference on Computational Intelligence in Economics and Finance (CIEF'05), Salt Lake City, Utah, USA, pp 1150–1153 34

Trippi R, Turban E (1992) Neural Networks in Finance and Investing: Using Artificial Intelligence to Improve Real World Performance. McGraw-Hill, Inc., New York, NY, USA 32

Tsang E (2009) Forecasting-where computational intelligence meets the stock market. Frontiers of Computer Science in China 3(1):53–63 28

Tsang E, Li J (2002) EDDIE for financial forecasting, Springer-Verlag New York, LLC, chap 7, pp 161–174. Genetic Algorithms and Genetic Programming in Computational Finance 35

Tsang E, Li J, Butler J (1998) EDDIE beats the bookies. International Journal of Software, Practice & Experience, Wiley 28(10):1033–1043 13, 35

Tsang E, Li J, Markose S, Er H, Salhi A, Iori G (2000) EDDIE in financial decision making. Journal of Management and Economics 4(4) 13, 35, 36

Tsang E, Yung P, Li J (2004) EDDIE-Automation, a decision support tool for financial forecasting. Journal of Decision Support Systems, Special Issue on Data Mining for Financial Decision Making 37(4):559–565 33, 35

Tsang E, Markose S, Er H (2005) Chance discovery in stock index option and future arbitrage. New Mathematics and Natural Computation, World Scientific 1(3):435–447 13, 35, 38

Versace M, Bhatt R, Hinds O, Shiffer M (2004) Predicting the exchange traded fund DIA with a combination of genetic algorithms and neural networks. Expert Systems with Applications 27:417–425 32

Vesanto J, Himberg J, Alhoniemi E, Parhankngas J (1999) Self-organizing map in Matlab: the SOM Toolbox. In: Proceedings of the Matlab DSP Conference, Espoo, Finland, pp 35–40 22, 24

Viceira LM (2011) Bond risk, bond return volatility, and the term structure of interest rates. International Journal of Forecasting In Press, Corrected Proof:– 28

Wang P, Tsang E, Weise T, Tang K, Yao X (2010) Using GP to evolve decision rules for classification in financial data sets. In: Cognitive Informatics (ICCI), 2010 9th IEEE International Conference on, pp 720–727, DOI 10.1109/COGINF.2010.5599820 33, 35

Wang P, Weise T, Chiong R (2011) Novel evolutionary algorithms for supervised classification problems: An experimental study. Evolutionary Intelligence 4:3–16 35

West D, Dellana S, Qian J (2005) Neural network ensemble strategies for financial decision applications. Computers & Operations Research 32(10):2543–2559 32

Whigham P (1995) Grammatically-based genetic programming. In: Rosca J (ed) Proceedings of the Workshop Genetic Programming: From Theory to Real-Wrold Applications, New York: Morgan Kaufmann, pp 33–41 13

Wilson G, Banzhaf W (2010) Fast and effective predictability filters for stock price series using linear genetic programming. In: Evolutionary Computation (CEC), 2010 IEEE Congress on, pp 1 –8, DOI 10.1109/CEC.2010.5586297 33

Winker P, Gilli M (2001) Indirect estimation of the parameters of agent-based models of financial markets. FAME Research paper, University of Geneva 38 44, 78

Wong B, Selvi Y (1998) Neural network applications in finace: A review and analysis of literature (1990-1996). Information & Management 34:129–139 32

Worasucheep C (2008) Trading index mutual funds with evolutionary forecasting. In: Proceedings of the IEEE Conference on Evolutionary Computation, Hong Kong, pp 430–435 34

Xie H, Zhang M, Andreae P (2007) Genetic programming for New Zealand CPI inflation prediction. In: Proceedings of the IEEE Congress on Evolutionary Computation, Singapore, pp 2538–2545 33

Xu R, Wunsch D (2008) Clustering. Wiley-IEEE Press 98

Yao X, Islam MM (2008) Evolving artificial neural network ensembles. IEEE Computational Intelligence Magazine 3(1):31–42 32

Yao X, Liu Y (1997) A new evolutionary system for evolving artificial neural networks. Neural Networks, IEEE Transactions on 8(3):694 –713, DOI 10.1109/72.572107 32

Yuan X, Zou Y (2009) Technology program financial forecast model based on caco-svm. In: Intelligent Systems and Applications, 2009. ISA 2009. International Workshop on, pp 1 –4, DOI 10.1109/IWISA.2009.5073158 34

Zhang G, Patuwo B, Hu M (1998) Forecasting with artificial neural networks: The state of the art. International Journal of Forecasting 14:35–62 32

Part VI

Appendix

Appendix A

Kolmogorov-Smirnov tests

Here we present the p-values for the Kolmogorov-Smirnov tests that took place in Chapter 6 (Section 6.4.2.1), and more specifically for the comparative results between the average values of Fitness, RC, RMC, and RF, for the 10 FTSE 100 stocks, between EDDIE 7 and EDDIE 8. Table A.1 presents these values in the next page. Whenever the p-value is below 5%, this indicates that the difference between the average values of the performance measures is significant at 5% level. A significant improvement by EDDIE 8 is denoted by formatting the number in bold fonts, whereas a significant improvement by EDDIE 7 is denoted by underlying the respective number.

Overall, EDDIE 7 has better results than EDDIE 8, under more stocks. In terms of Fitness, EDDIE 7 is doing better in 4 stocks, whereas EDDIE 8 is doing better in 3. In terms of RC, RMC, and RF, EDDIE 7 is doing better in 5, 5, and 5 stocks respectively, whereas EDDIE 8 is doing better in 2, 1 and 4 stocks, respectively.

Table A.1: Kolmogorov-Smirnov test for testing whether the differences between the average values for the Fitness, RC, RMC, and RF between EDDIE 7 and EDDIE 8 are significant at 5% significance level.

p-value				
	Fitness	RC	RMC	RF
BAT	0.507658	0.677937	0.044629	**0.000966**
BP	**0.017144**	0.507658	0.002112	**2.07e-17**
Cadbury	0.008899	0.008899	0.35842	0.000966
Carnival	**0.009509**	**0.039676**	0.55034	**0.026563**
Hammerson	**2.76e-05**	**2.76e-05**	**1.02e-05**	0.095096
Imp.Tob.	0.095096	0.017144	0.095096	0.031661
Next	1.02e-05	1.02e-05	0.000178	0.002112
Schroders	0.004428	0.008899	0.002112	0.017144
Tesco	0.596643	0.442784	0.677937	**7.16e-05**
Unilever	4.02e-07	1.23e-06	3.76e-08	0.000178

Appendix B

Additional Performance Measures

Here we present the formulas for the two additional metrics AARR and RPR, as presented in Li (2001). We would once again like to remind the reader that these metrics should be used for reference only, since they are not part of the fitness function.

***Hypothetical Trading Behaviour**: We assume that when a positive position is predicted by a GDT, one unit of money is invested in a stock reflecting the current closing price. If the closing price does rise by $r\%$ or more at day t within the next n trading days, we then sell the portfolio at the closing price of day t. If not, we sell the portfolio on the n_{th} day, regardless of the price.*

Given a positive position predicted, for example, the i_{th} positive position, for simplicity, we ignore transaction cost, and annualize its return by the following formula, presented in Equation (B.1):

$$ARR_i = \frac{255}{t} * \frac{P_t - P_0}{P_0} \tag{B.1}$$

Where P_0 is the buy price, P_t is the sell price, t is the number of days in markets, 255 is the number of total trading days in one calendar year. Given a GDT that generates N_+ number of positive positions over the period examined, its average ARR is shown in Equation (B.2):

$$AARR = \frac{1}{N} \sum_{i=1}^{N_+} ARR_i \tag{B.2}$$

RPR (Equation (B.3)) refers to the ratio of the number of signals, which turn out to achieve positive returns, to the total number of positive positions predicted, where a specific GDT is invoked for a finite period

$$RPR = \frac{1}{N_+} \sum_{i=1}^{N_+} I_i \tag{B.3}$$

where

$$I_i = \begin{cases} 1 & if ARR_i \geq 0 \\ 0 & otherwise \end{cases}$$

and

$$0 < i \leq N_+$$

where N_+ is the number of positive positions generated by the GDT, and ARR_i is an annualised rate of return for the i_{th} signal.

Appendix C

Figures of Cumulative Fractions

In this section we present the figures of cumulative fractions for all ten datasets, as discussed in Section 7.6.3.2. We can again see here that most of the markets tend to have a few gigantic clusters. The only exceptions are NYSE and S&P500.

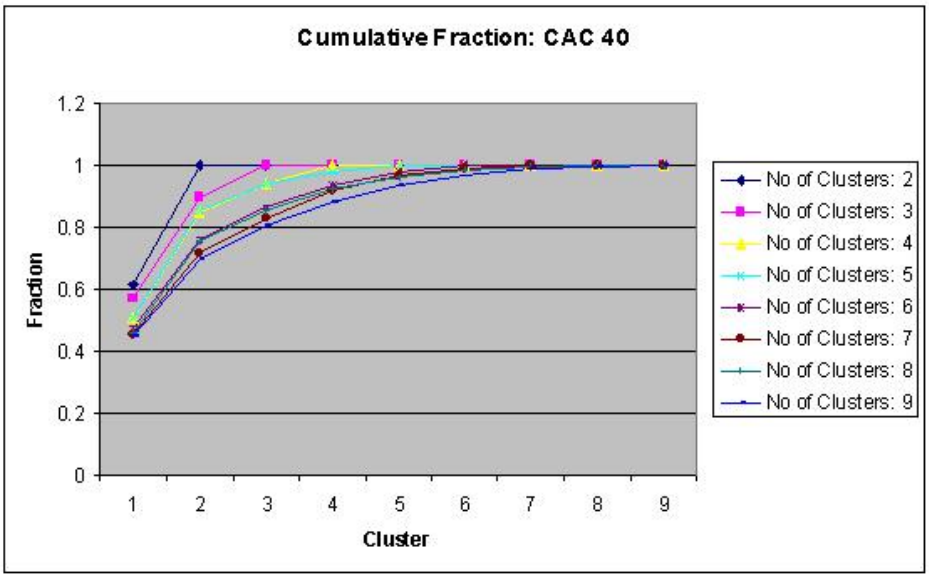


Figure C.1: Cumulative Fraction for CAC 40

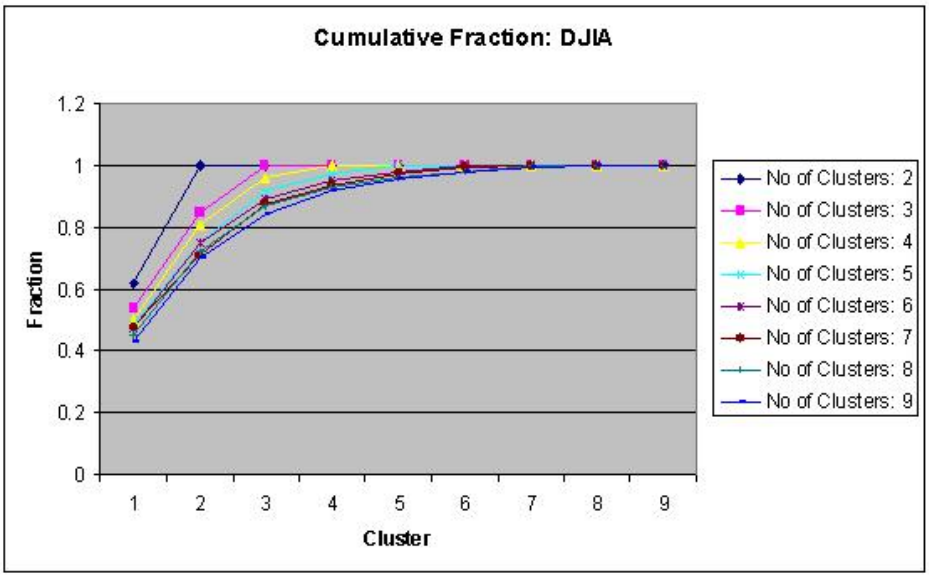


Figure C.2: Cumulative Fraction for DJIA

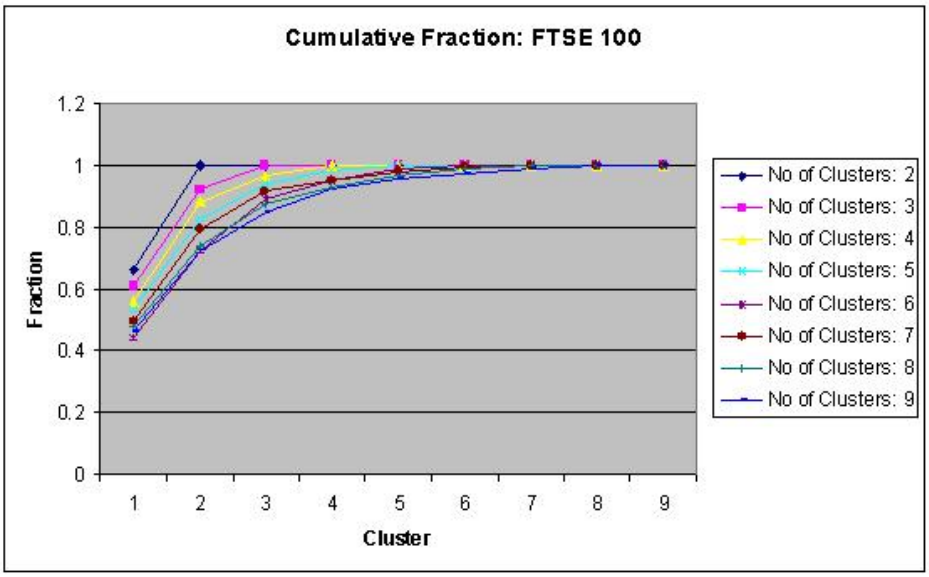


Figure C.3: Cumulative Fraction for FTSE 100

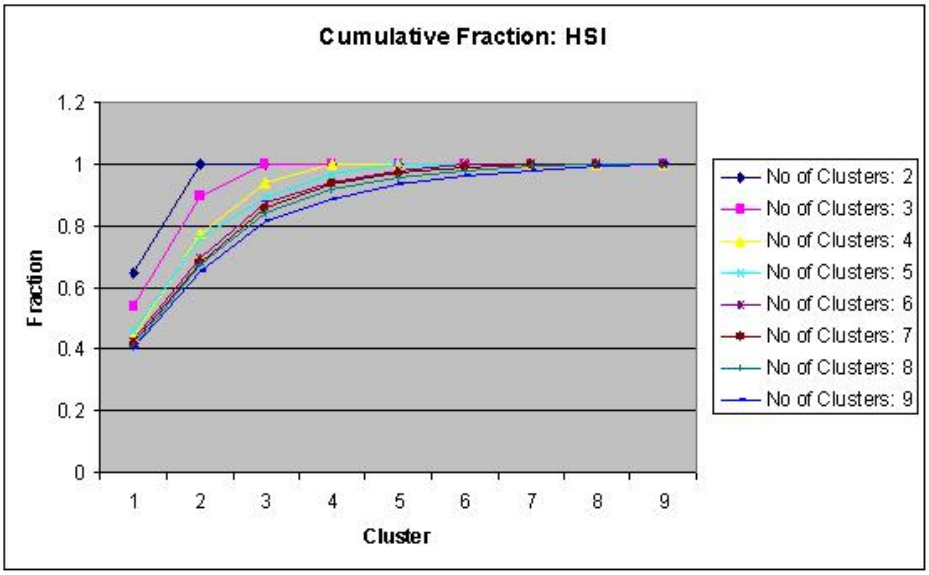


Figure C.4: Cumulative Fraction for HSI

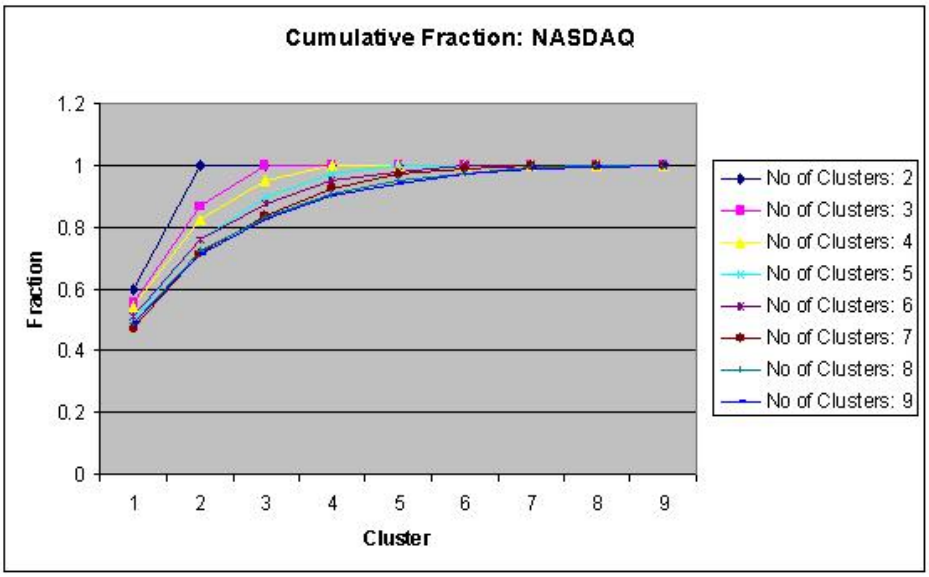


Figure C.5: Cumulative Fraction for NASDAQ

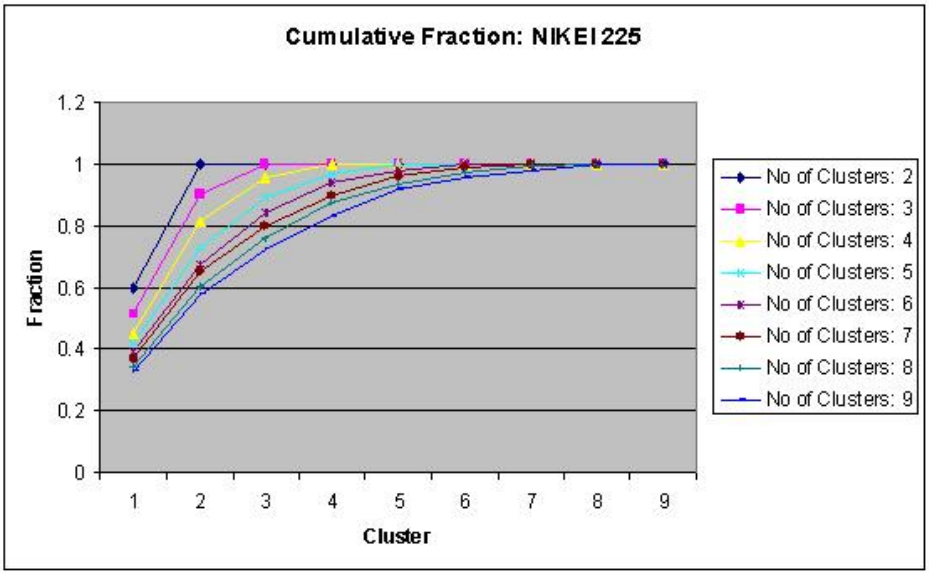


Figure C.6: Cumulative Fraction for NIKEI 225

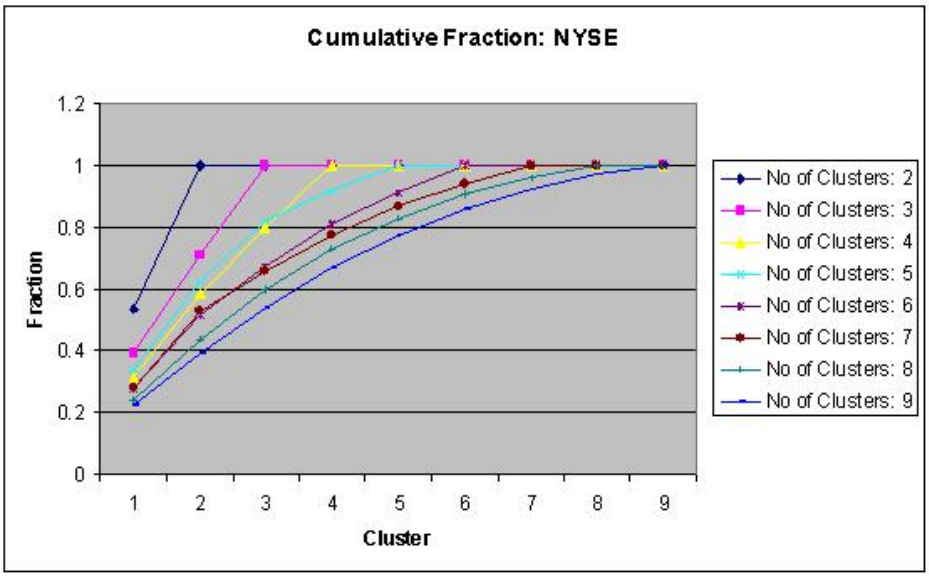


Figure C.7: Cumulative Fraction for NYSE

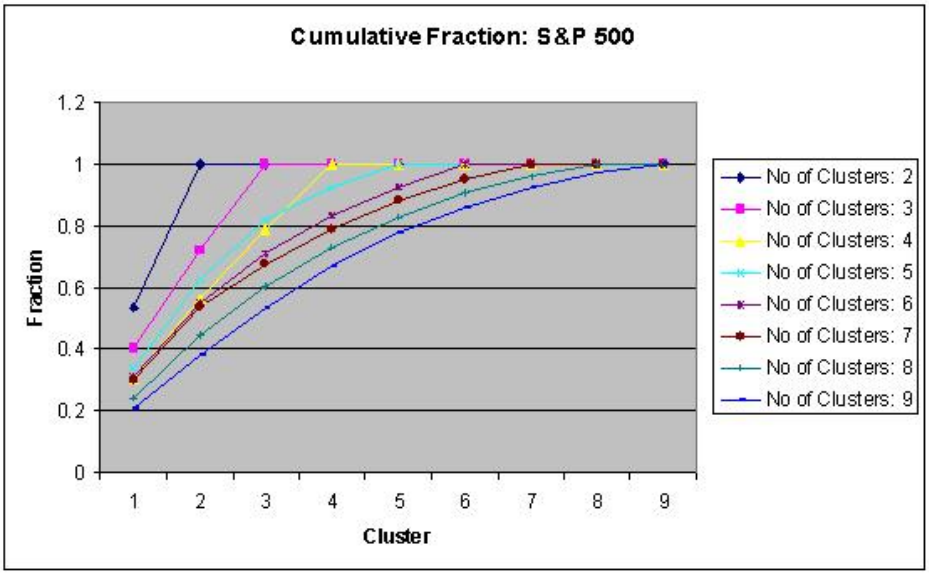


Figure C.8: Cumulative Fraction for S&P 500

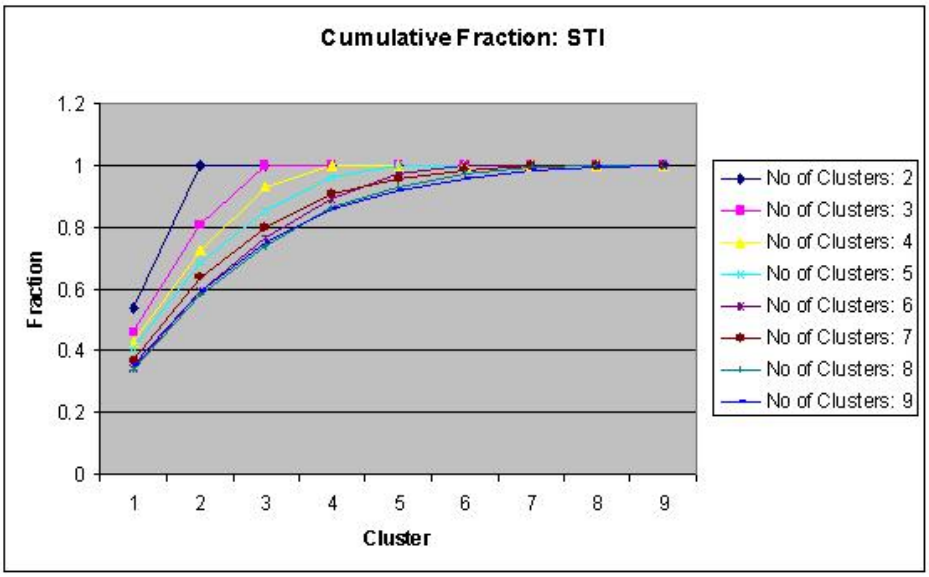


Figure C.9: Cumulative Fraction for STI

Appendix D

Dissatisfaction Rate tables under EDDIE 7 and EDDIE 8

Here we present the tables which show the mean of average and minimum dissatisfaction rates for the DH test, under EDDIE 7 and EDDIE 8. As we can see, dissatisfaction tends to increase, as the number of clusters increases. In terms of average dissatisfaction, it is in the range of 4.39 (2×1 SOM) to 7.62 (3×3 SOM), for EDDIE 7, and 4.17 (2×1 SOM) to 7.14 (3×3 SOM), for EDDIE 8. In terms of minimum dissatisfaction, results are in the range of 3.58 (2×1 SOM) to 5.71 (3×3 SOM) for EDDIE 7, and 3.37 (2×1 SOM) to 5.30 (3×3 SOM), for EDDIE 8. The differences between the two GP algorithms are very small.

Table D.1: Average of Average Dissatisfaction Rate per Cluster per Dataset for EDDIE 7.

	2×1	3×1	2×2	5×1	3×2	7×1	4×2	3×3
CAC40	4.37	4.84	5.34	6.03	6.27	7.14	7.22	7.60
DJIA	4.57	5.13	5.55	6.25	6.56	7.33	7.47	7.83
FTSE100	4.31	4.83	5.36	5.95	6.27	7.02	7.06	7.52
HSI	4.34	4.83	5.30	5.82	6.17	6.91	6.98	7.37
NASDAQ	4.59	5.18	5.65	6.32	6.65	7.45	7.52	8.10
NIKEI	4.26	4.69	5.11	5.60	5.89	6.73	6.57	7.12
NYSE	4.56	5.11	5.55	6.37	6.60	7.40	7.60	7.98
S&P500	4.35	4.92	5.38	5.96	6.32	7.12	7.30	7.68
STI	4.39	4.89	5.32	5.97	6.17	7.02	7.07	7.51
TAIEX	4.19	4.67	5.20	5.86	6.06	6.95	6.98	7.50
Mean	4.39	4.91	5.38	6.01	6.30	7.11	7.18	7.62

Table D.2: Average of Average Dissatisfaction Rate per Cluster per Dataset for EDDIE 8.

	2×1	3×1	2×2	5×1	3×2	7×1	4×2	3×3
CAC40	4.13	4.54	5.03	5.75	5.95	6.67	6.83	7.11
DJIA	4.43	4.99	5.47	6.15	6.44	7.26	7.32	7.78
FTSE100	4.31	4.86	5.24	5.96	6.18	7.08	7.17	7.56
HSI	3.96	4.45	4.71	5.35	5.57	6.21	6.32	6.66
NASDAQ	4.25	4.83	5.18	5.79	6.08	6.74	6.90	7.29
NIKEI	4.06	4.41	4.74	5.34	5.53	6.19	6.25	6.59
NYSE	4.17	4.61	5.12	5.70	5.99	6.77	6.91	7.25
S&P500	4.19	4.73	5.10	5.71	5.95	6.71	6.79	7.18
STI	4.18	4.63	5.03	5.64	5.86	6.68	6.77	7.12
TAIEX	4.03	4.49	4.91	5.50	5.72	6.38	6.59	6.88
Mean	4.17	4.65	5.05	5.69	5.93	6.67	6.79	7.14

Table D.3: Average of Minimum Dissatisfaction Rate per Cluster per Dataset for EDDIE 7.

	2×1	3×1	2×2	5×1	3×2	7×1	4×2	3×3
CAC40	3.66	3.89	4.31	4.77	4.95	5.54	5.65	5.91
DJIA	3.74	4.11	4.34	4.88	5.07	5.65	5.79	6.09
FTSE100	3.55	3.78	4.20	4.62	4.82	5.38	5.47	5.80
HSI	3.50	3.58	3.93	4.12	4.49	4.92	4.95	5.13
NASDAQ	3.75	4.14	4.48	4.92	5.19	5.72	5.87	6.26
NIKEI	3.47	3.65	3.76	4.11	4.22	4.85	4.67	5.03
NYSE	3.80	4.10	4.33	4.99	5.11	5.74	5.90	6.22
S&P500	3.55	3.91	4.21	4.61	4.89	5.41	5.61	5.83
STI	3.37	3.72	3.95	4.45	4.54	5.11	5.22	5.34
TAIEX	3.43	3.71	4.07	4.53	4.58	5.31	5.23	5.54
Mean	3.58	3.86	4.16	4.60	4.79	5.36	5.43	5.71

Table D.4: Average of Minimum Dissatisfaction Rate per Cluster per Dataset for EDDIE 8.

	2×1	3×1	2×2	5×1	3×2	7×1	4×2	3×3
CAC40	3.44	3.68	3.99	4.54	4.66	5.23	5.32	5.50
DJIA	3.56	3.94	4.18	4.66	4.92	5.49	5.49	5.84
FTSE100	3.54	3.89	4.24	4.65	4.83	5.51	5.56	5.84
HSI	3.29	3.56	3.78	4.20	4.19	4.51	4.66	4.88
NASDAQ	3.43	3.72	3.92	4.26	4.44	4.91	4.95	5.24
NIKEI	3.28	3.47	3.58	3.99	4.14	4.54	4.64	4.77
NYSE	3.44	3.72	4.02	4.44	4.56	5.13	5.26	5.47
S&P500	3.25	3.72	3.86	4.30	4.49	4.93	5.04	5.34
STI	3.18	3.52	3.70	4.19	4.28	4.81	4.86	5.10
TAIEX	3.26	3.48	3.68	4.09	4.27	4.68	4.85	5.01
Mean	3.37	3.67	3.90	4.33	4.48	4.97	5.06	5.30

Appendix E

Standard Deviation Results: Chapters 7 and 8

This chapter provides standard deviations results for MFH and DH tests.

Table E.1: Standard Deviations over 10 runs for MFH Test 1-Max for 2-9 clusters.

Cl.	CAC 40	DJIA	FTSE100	HSI	NASDAQ	NIKEI	NYSE	S&P500	STI	TAIEX
2	1.236	3.414	3.642	3.393	0.948	0.816	3.000	3.504	1.776	3.041
3	1.807	1.165	1.864	2.203	0.994	1.663	1.349	1.595	1.201	1.772
4	1.732	0.975	1.873	1.922	0.971	1.337	1.251	1	1.333	3.154
5	1.832	1.581	3.494	1.407	0.994	1.433	2.936	1.813	1.054	2.658
6	1.908	1.618	2.058	1.669	0.971	1.337	2.494	2.658	0.866	3.132
7	1.831	1.302	1.837	1.846	0.942	1.286	2.990	2.643	0.527	1.643
8	1.407	1.889	2.444	2.356	1.092	1.080	3.457	2.603	1.092	2.658
9	1.126	1.481	2.138	1.726	1.166	0.843	2.833	2.602	0.866	2.866

Table E.2: Standard Deviations over 10 runs for the MFH Test 2 for 2-9 clusters.

Cl.	CAC 40	DJIA	FTSE100	HSI	NASDAQ	NIKEI	NYSE	S&P500	STI	TAIEX
2	0.114	0.091	0.122	0.087	0.013	0.020	0.004	0.006	0.002	0.068
3	0.120	0.082	0.124	0.092	0.071	0.074	0.011	0.015	0.079	0.097
4	0.115	0.109	0.111	0.102	0.081	0.062	0.033	0.013	0.081	0.114
5	0.121	0.124	0.105	0.105	0.088	0.079	0.029	0.043	0.106	0.066
6	0.101	0.101	0.125	0.113	0.105	0.087	0.021	0.040	0.094	0.112
7	0.096	0.118	0.103	0.108	0.119	0.093	0.053	0.064	0.107	0.119
8	0.108	0.127	0.090	0.119	0.113	0.119	0.047	0.064	0.120	0.113
9	0.082	0.113	0.093	0.100	0.127	0.112	0.024	0.034	0.118	0.094

Table E.3: Standard Deviations over 10 runs for Table E.3.

	2×1	3×1	2×2	5×1	3×2	7×1	4×2	3×3
CAC40	0.6130	0.6733	0.7159	0.8649	0.8696	1.1295	1.0788	1.0252
DJIA	0.755	0.7964	0.7972	0.9762	1.0256	1.1205	1.1458	1.2519
FTSE100	0.8444	0.9242	0.9744	1.1796	1.2017	1.2958	1.3312	1.4687
HSI	1.0901	1.1832	1.1611	1.4833	1.4611	1.7639	1.7546	1.8979
NASDAQ	0.6748	0.7423	0.8124	0.9436	1.0021	1.1788	1.1377	1.2453
NIKEI	1.1188	1.0078	1.0911	1.3812	1.4042	1.6736	1.6261	1.7493
NYSE	0.5051	0.5483	0.5925	0.6993	0.6855	0.8225	0.7939	0.9160
S&P500	0.6660	0.6791	0.7277	0.8264	0.9114	1.01731	1.0788	1.1257
STI	0.9986	1.1455	1.2410	1.4000	1.4638	1.7425	1.7299	1.8211
TAIEX	1.0322	1.0428	1.03770	1.2596	1.3345	1.5001	1.4510	1.6140