

Anthony Brabazon and Michael O'Neill (Eds.)

Natural Computing in Computational Finance: Volume 2

Studies in Computational Intelligence, Volume 185

Editor-in-Chief

Prof. Janusz Kacprzyk
Systems Research Institute
Polish Academy of Sciences
ul. Newelska 6
01-447 Warsaw
Poland
E-mail: kacprzyk@ibspan.waw.pl

Further volumes of this series can be found on our homepage:
springer.com

Vol. 163. Pawel Delimata, Mikhail Ju. Moshkov,
Andrzej Skowron and Zbigniew Suraj
Inhibitory Rules in Data Analysis, 2009
ISBN 978-3-540-85637-5

Vol. 164. Nadia Nedjah, Luiza de Macedo Mourelle,
Janusz Kacprzyk, Felipe M.G. França
and Alberto Ferreira de Souza (Eds.)
Intelligent Text Categorization and Clustering, 2009
ISBN 978-3-540-85643-6

Vol. 165. Djamel A. Zighed, Shusaku Tsumoto,
Zbigniew W. Ras and Hakim Hacid (Eds.)
Mining Complex Data, 2009
ISBN 978-3-540-88066-0

Vol. 166. Constantinos Koutsojannis and Spiros Sirmakessis
(Eds.)
Tools and Applications with Artificial Intelligence, 2009
ISBN 978-3-540-88068-4

Vol. 167. Ngoc Thanh Nguyen and Lakhmi C. Jain (Eds.)
Intelligent Agents in the Evolution of Web and Applications, 2009
ISBN 978-3-540-88070-7

Vol. 168. Andreas Tolk and Lakhmi C. Jain (Eds.)
*Complex Systems in Knowledge-based Environments: Theory,
Models and Applications*, 2009
ISBN 978-3-540-88074-5

Vol. 169. Nadia Nedjah, Luiza de Macedo Mourelle and
Janusz Kacprzyk (Eds.)
Innovative Applications in Data Mining, 2009
ISBN 978-3-540-88044-8

Vol. 170. Lakhmi C. Jain and Ngoc Thanh Nguyen (Eds.)
*Knowledge Processing and Decision Making in Agent-Based
Systems*, 2009
ISBN 978-3-540-88048-6

Vol. 171. Chi-Keong Goh, Yew-Soon Ong and Kay Chen Tan
(Eds.)
Multi-Objective Memetic Algorithms, 2009
ISBN 978-3-540-88050-9

Vol. 172. I-Hsien Ting and Hui-Ju Wu (Eds.)
Web Mining Applications in E-Commerce and E-Services, 2009
ISBN 978-3-540-88080-6

Vol. 173. Tobias Grosche
Computational Intelligence in Integrated Airline Scheduling,
2009
ISBN 978-3-540-89886-3

Vol. 174. Ajith Abraham, Rafael Falcón and Rafael Bello (Eds.)
Rough Set Theory: A True Landmark in Data Analysis, 2009
ISBN 978-3-540-89886-3

Vol. 175. Godfrey C. Onwubolu and Donald Davendra (Eds.)
*Differential Evolution: A Handbook for Global
Permutation-Based Combinatorial Optimization*, 2009
ISBN 978-3-540-92150-9

Vol. 176. Beniamino Murgante, Giuseppe Borruso and
Alessandra Lapucci (Eds.)
Geocomputation and Urban Planning, 2009
ISBN 978-3-540-89929-7

Vol. 177. Dikai Liu, Lingfeng Wang and Kay Chen Tan (Eds.)
Design and Control of Intelligent Robotic Systems, 2009
ISBN 978-3-540-89932-7

Vol. 178. Swagatam Das, Ajith Abraham and Amit Konar
Metaheuristic Clustering, 2009
ISBN 978-3-540-92172-1

Vol. 179. Mircea Gh. Negoita and Sorin Hintea
Bio-Inspired Technologies for the Hardware of Adaptive Systems,
2009
ISBN 978-3-540-76994-1

Vol. 180. Wojciech Mitkowski and Janusz Kacprzyk (Eds.)
Modelling Dynamics in Processes and Systems, 2009
ISBN 978-3-540-92202-5

Vol. 181. Georgios Miaoulis and Dimitri Plemenos (Eds.)
Intelligent Scene Modelling Information Systems, 2009
ISBN 978-3-540-92901-7

Vol. 182. Andrzej Bargiela and Witold Pedrycz (Eds.)
*Human-Centric Information Processing Through Granular
Modelling*, 2009
ISBN 978-3-540-92915-4

Vol. 183. Marco A.C. Pacheco and Marley M.B.R. Vellasco (Eds.)
Intelligent Systems in Oil Field Development under Uncertainty,
2009
ISBN 978-3-540-92999-4

Vol. 184. Ljupco Kocarev, Zbigniew Galias and Shiguo Lian
(Eds.)
Intelligent Computing Based on Chaos
ISBN 978-3-540-95971-7

Vol. 185. Anthony Brabazon and Michael O'Neill (Eds.)
Natural Computing in Computational Finance, 2009
ISBN 978-3-540-95973-1

Anthony Brabazon
Michael O'Neill (Eds.)

Natural Computing in Computational Finance

Volume 2



Springer

Prof. Anthony Brabazon

Head of Research, School of Business

Director, Natural Computing Research and Applications Group,

University College Dublin Belfield,

Dublin 4

Ireland

E-mail: anthony.brabazon@ucd.ie

Dr. Michael O'Neill

Director, Natural Computing Research and Applications Group,

School of Computer Science and Informatics,

University College Dublin Belfield,

Dublin 4

Ireland

E-mail: m.oneill@ucd.ie

ISBN 978-3-540-95973-1

e-ISBN 978-3-540-95974-8

DOI 10.1007/978-3-540-95974-8

Studies in Computational Intelligence

ISSN 1860949X

Library of Congress Control Number: 2008922057

© 2009 Springer-Verlag Berlin Heidelberg

This work is subject to copyright. All rights are reserved, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, reuse of illustrations, recitation, broadcasting, reproduction on microfilm or in any other way, and storage in data banks. Duplication of this publication or parts thereof is permitted only under the provisions of the German Copyright Law of September 9, 1965, in its current version, and permission for use must always be obtained from Springer. Violations are liable to prosecution under the German Copyright Law.

The use of general descriptive names, registered names, trademarks, etc. in this publication does not imply, even in the absence of a specific statement, that such names are exempt from the relevant protective laws and regulations and therefore free for general use.

Typeset & Cover Design: Scientific Publishing Services Pvt. Ltd., Chennai, India.

Printed in acid-free paper

9 8 7 6 5 4 3 2 1

springer.com

To Maria
Tony

To Gráinne and Aoife
Michael

Preface

Recent years have seen the widespread application of Natural Computing algorithms (broadly defined in this context as computer algorithms whose design draws inspiration from phenomena in the natural world) for the purposes of financial modelling and optimisation. A related stream of work has also seen the application of learning mechanisms drawn from Natural Computing algorithms for the purposes of agent-based modelling in finance and economics. In this book we have collected a series of chapters which illustrate these two faces of Natural Computing. The first part of the book illustrates how algorithms inspired by the natural world can be used as problem solvers to uncover and optimise financial models. The second part of the book examines a number agent-based simulations of financial systems.

This book consists of ten chapters each of which was selected following a rigorous, peer-reviewed, selection process. The chapters illustrate the application of a range of cutting-edge natural computing and agent-based methodologies in computational finance and economics. While describing cutting edge applications, the chapters are written so that they are accessible to a wide audience. Hence, they should be of interest to academics, students and practitioners in the fields of computational finance and economics.

The inspiration for this book was due in part to the success of EvoFIN 2008, the 2nd European Workshop on Evolutionary Computation in Finance and Economics. EvoFIN 2008 took place in conjunction with Evo* 2008 in Naples, Italy (26-28 March 2008). Evo* is an annual collection of European conferences and workshops broadly focused on Evolutionary Computation, and is the largest European event dedicated to this field of research. Some of the chapters presented in this book are extended versions of papers presented at EvoFIN 2008, which have also undergone the same rigorous, peer-reviewed, selection process as the other chapters.

This book follows on from **Natural Computing in Computational Finance** (Volume 100 in Springer's *Studies in Computational Intelligence* series) which in turn arose from the success of EvoFIN 2007, the very first European

Workshop on Evolutionary Computation in Finance & Economics held in Valencia, Spain in April 2007.

We would like to thank all the authors for their high-quality contributions and the reviewers who generously gave of their time to peer-review all submissions. We also extend our thanks to Dr. Thomas Ditzinger of Springer-Verlag and to Professor Janusz Kacprzyk, editor of this book series, for their encouragement of, and their support during, the preparation of this book.

Dublin
November 2008

Anthony Brabazon
Michael O'Neill

Contents

1 Natural Computing in Computational Finance (Volume 2): Introduction

Anthony Brabazon, Michael O'Neill 1

Part I Financial Modelling

2 Statistical Arbitrage with Genetic Programming

Philip Saks, Dietmar Maringer 9

3 Finding Relevant Variables in a Financial Distress Prediction Problem Using Genetic Programming and Self-organizing Maps

*E. Alfaro-Cid, A.M. Mora, J.J. Merelo, A.I. Esparcia-Alcázar,
K. Sharman* 31

4 Ant Colony Optimization for Option Pricing

Sameer Kumar, Ruppa K. Thulasiram, Parimala Thulasiraman 51

5 A Neuro-Evolutionary Approach for Interest Rate Modelling

Robert Bradley, Anthony Brabazon, Michael O'Neill 75

6 Who's Smart and Who's Lucky? Inferring Trading Strategy, Learning and Adaptation in Financial Markets through Data Mining

*Christopher R. Stephens, José Luis Gordillo,
Enrique Martínéz Miranda* 95

Part II Agent-Based Modelling

7 Financial Bubbles: A Learning Effect Modelling Approach

Tsung-Han Hsieh, Youwei Li, Donal G. McKillop 117

8 Evolutionary Computation and Artificial Financial Markets

Serafin Martinez-Jaramillo, Edward P.K. Tsang 137

9 Classical and Agent-Based Evolutionary Algorithms for Investment Strategies Generation

Rafał Dreżewski, Jan Sepielak, Leszek Siwik 181

10 Income Distribution and Lottery Expenditures in Taiwan: An Analysis Based on Agent-Based Simulation

Shu-Heng Chen, Bin-Tzong Chie, Hui-Fen Fan, Tina Yu 207

11 The Emergence of a Market: What Efforts Can Entrepreneurs Make?

Shuyuan Wu, Anthony Brabazon 225

Index 245

Author Index 249

Natural Computing in Computational Finance (Volume 2): Introduction

Anthony Brabazon^{1,2} and Michael O'Neill^{1,3}

¹ Natural Computing Research & Applications Group,
Complex & Adaptive Systems Laboratory,
University College Dublin, Ireland
{anthony.brabazon,m.oneill}@ucd.ie

² School of Business, University College Dublin, Ireland

³ School of Computer Science and Informatics, University College Dublin, Ireland

1.1 Introduction

Natural computing (NC) can be broadly defined as the development of computer programs and computational algorithms using metaphorical inspiration from systems and phenomena that occur in the natural world. A growing community of researchers are engaged in the application of NC methodologies in computational finance and agent-based modelling. The scale of NC applications in finance is illustrated by Chen & Kuo [4] who list nearly 400 papers that had been published by 2001 on the use of evolutionary computation alone in computational economics and finance. Since then several hundred additional papers have been published illustrating the continued growth in this application area (see also [2, 3, 9, 10] for additional examples of NC applications in finance). Examples of the many conferences, workshops and special sessions in this area include the annual track on evolutionary computation in finance & economics at the IEEE Congress on Evolutionary Computation, the IEEE symposium on Computational Intelligence for Financial Engineering (CIFEr), the annual international conference on Computational Intelligence in Economics & Finance (CIEF), and the European workshop on Evolutionary Computation in Finance (EvoFIN) held annually as part of Evo*.

This book consists of ten chapters each of which was selected following a rigorous, peer-reviewed, selection process. The chapters illustrate the application of a range of cutting-edge natural computing and agent-based methodologies in computational finance and economics. In the first half of this book, we see a series of applications of natural computing algorithms for financial modelling (including forecasting, algorithmic trading, portfolio optimisation, risk management and derivative modelling), and the second part illustrates a series of applications of agent-based models to gain insight into financial markets and other economic phenomena. In the process, we expose a number of natural computing algorithms drawn from a number of the sub-fields of Natural Computing, including, Genetic Programming, Self-Organising Maps, Ant Colony Optimisation, Genetic Algorithms, Multi-layer perceptrons, and examples of hybridisations of these methods.

The rest of this chapter is organised as follows. Sect. 1.2 briefly overviews of some key families of NC methods. The individual chapters in the book are then introduced in Sect. 1.3.

1.2 Natural Computing

Natural computing (NC) algorithms can be clustered into different groups depending on the aspects of the natural world upon which they are based. The main clusters are illustrated in Fig.1.1.

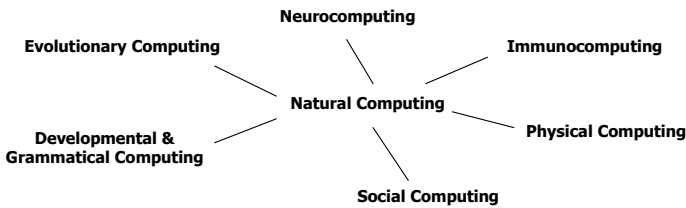


Fig. 1.1. An overview of the main families of Natural Computing Algorithms

Neurocomputing (or artificial neural networks) typically draws inspiration from the workings of the human brain / nervous system. ANNs can be characterised by a set of neurons, the network structure describing the pattern of connectivity between neurons, and the learning approach used. The predominant neurocomputing paradigms include multi-layer perceptrons, radial basis function networks, self-organising maps, and adaptive resonance theory.

Evolutionary Computation (EC) is based upon neo-Darwinian principles of evolution. A population-based search process is used, whereby better (fitter) members of the population are preferentially selected for reproduction and modification, leading to a new population of individuals increasingly adapted to their environment. The main streams of EC are genetic algorithms, evolution strategies, evolutionary programming and genetic programming.

Social Computing adopts a swarm metaphor and includes algorithms inspired by the flocking and schooling behaviour of birds and fish, and the behaviours observed in social insects such as ants. The characteristics of these social systems facilitate self-organisation, flexibility, robustness, and direct/indirect communication among members of the population. Examples of social computing include ant colony, particle swarm and bacterial foraging algorithms.

Immunocomputing encompasses a family of algorithms inspired by the complex and adaptive biological immune system of vertebrates. The natural immune system represents an intricate network of specialised chemicals, cells, tissues and organs with the ability to recognise, destroy and remember an almost unlimited number of foreign bodies, and to protect the organism from misbehaving cells.

Physical Computing draws inspiration from the physical processes of the natural world, such as simulated annealing and quantum mechanics.

Developmental and Grammatical Computing borrows from both a developmental and a grammar metaphor. Grammatical computing adopts concepts from linguistic grammars, where generative grammars are used to construct a sentence(s) in the language specified by the grammar. This process is metaphorically similar to the developmental process in biology where ‘rules’ govern the production of a multi-cellular organism from a single cell. An example is Grammatical Evolution (GE), which is a grammatical variant of genetic programming.

These families of natural computing algorithms provide a rich set of tools for the development of quality optimisation and model induction applications, and all have seen applications in finance. Readers requiring detailed information on these algorithms are referred to [1, 5, 6].

Agent-based Modelling

Agent-based modelling (ABM) has become a fruitful area of financial and economic research in recent years. ABM allows the simulation of markets which consist of heterogeneous agents, with differing risk attitudes, and with differing expectations to future outcomes. This contrasts with traditional assumptions in financial economics of investor homogeneity and rational expectations. The essence of ABM lies in the notion of autonomous agents whose behaviour evolves endogenously, producing complex, emergent, system dynamics which are not predictable from the behaviours of individual agents. In designing agent-based models of financial markets, NC methods can be used to model information processing and adaptive learning by the agents.

A key output from the ABM literature is that it illustrates that complex system behaviour can arise from the interaction of relatively simple agents. When carefully constructed and validated, agent-based models can increase our understanding of market processes and can potentially provide insights for policy makers and regulators. Readers requiring a detailed introduction to agent-based modelling are referred to [7, 8].

1.3 Chapters

A wide variety of natural computing methodologies have been applied for optimisation and model induction purposes in finance. The next five chapters illustrate the application of a selection of NC methods for the purposes of developing models for financial trading, risk management, option pricing and interest rate modelling.

Chap. 2 (*Statistical Arbitrage with Genetic Programming* by Philip Saks and Dietmar Maringer) uses genetic programming to discover statistical arbitrage strategies for a portfolio of banking stocks within the Euro Stoxx index. Unlike most prior applications of GP for trading purposes, the study examines the use of a dual tree structure where two decision trees are generated and the evaluation is contingent on the current market position. Hence, buy and sell rules are co-evolved for long and short positions.

Risk management is an important area in finance. One strand of this work is the assessment of the financial stability of a company. Although many studies in this area focus on the prediction of corporate failure, commonly, companies will display a ‘trajectory’ towards financial distress for several years. Indicators of this typically include mounting financial losses and an associated loss of liquidity. Chap. 3 (*Finding Relevant Variables in a Financial Distress Prediction Problem Using Genetic Programming*

and *Self-organizing Maps* by E. Alfaro-Cid, A.M. Mora, J.J. Merelo, A.I. Esparcia-Alcázar and K. Sharman) looks at the prediction of book losses and uses GP as a model-induction tool to uncover a suitable prediction model. Kohonen's self organizing maps (SOMs) are used to assist with data pre-processing in order to reduce the dimensionality of the search space.

Ant colony optimisation (ACO) has been used extensively for combinatorial optimisation across a range of application areas but as yet has seen relatively little application in finance. Chap. 4 (*Ant Colony Optimization for Option Pricing* by Sameer Kumar, Ruppa K. Thulasiram and Parimala Thulasiraman) illustrates a novel application of ACO for option pricing purposes. While the primary focus of this chapter is concerned with the pricing of vanilla options, the methods developed offer the prospect of developing quality pricing systems for exotic options.

Chap. 5, (*A Neuro-evolutionary Approach for Interest Rate Modelling* by Robert Bradley, Anthony Brabazon and Michael O'Neill) presents an application of a neuro-evolutionary methodology for the purposes of the intraday trading of German government bond futures.

The final chapter in this section, chap. 6 (*Who's smart and who's lucky? Inferring trading strategy, learning and adaptation in Financial Markets through Data Mining* by Christopher R. Stephens, José Luis Gordillo and Enrique Martínéz Miranda) provides a thought-provoking examination of some of the problems facing designers of trading systems, focussing in particular on the difficulty of performance measurement. In other words, how can we determine whether the result produced by a trading strategy occurred as a result of luck or because the strategy is "superior"?

The second section of this book includes five chapters that adopt an ABM approach. In chap. 7, (*Financial Bubbles: A Learning Effect Modelling Approach* by Tsung-Han Hsieh, Youwei Li, and Donal G. McKillop) ABM is used to examine the formation of financial bubbles. The study examines the impact of learning and feedback effects in a market of heterogeneous investors, and examines how these contribute to price multiplicity and market demand.

A comprehensive overview of artificial financial markets is provided in chap. 8 (*Evolutionary Computation and Artificial Financial Markets* by S. Martinez-Jaramillo and E. P. K. Tsang). The chapter also introduces a software platform called Co-evolutionary, Heterogeneous Artificial Stock Market (CHASM); which allows the authors to perform a series of experiments with the purpose of identifying the aspects that could be responsible for the statistical properties (stylized facts) of financial prices. The artificial agents (technical traders in this study) are represented as genetic programming (GP) based agents which co-evolve in the market forecasting price changes on the basis of technical indicators.

In chap. 9, (*Classical and Agent-Based Evolutionary Algorithms for Investment Strategies Generation* by Rafał Dreżewski, Jan Sepielak, Leszek Siwik) an agent-based co-evolutionary system is introduced and is employed for the purposes of generating investment strategies. The results from this system are compared with those of a classic GP-based system in terms of investment return and strategy diversity.

The final two chapters in the book are drawn from an agent-based computational economics perspective, exploring lottery markets and the emergence of new technology

markets respectively. Chap. 10 (*Classical and Agent-Based Evolutionary Algorithms for Investment Strategies Generation* by Shu-Heng Chen, Bin-Tzong Chie, Hui-Fen Fan and Tina Yu) describes an ABM simulation of a lottery market. The agents in the model are potential lottery buyers, whose characteristics are described by three features: the percentage of income spent on the lottery, the preferences among lottery numbers selected and the aversion to regret. This model is used to investigate the impact of income distribution on lottery expenditures in Taiwan.

The final chapter (*The Emergence of a Market: What Efforts can Entrepreneurs Make?* by Shuyuan Wu and Anthony Brabazon) constructs an agent-based model in order to gain insight as to how entrepreneurs create markets for new, disruptive, technologies through an effectuation process. Starting from dispersed knowledge components held by both the demand and supply sides, a market emerges from the interactive learning behaviours of entrepreneurs and potential customers. The results indicate that the process of market creation is significantly impacted by factors including exploration tendency, alertness, and participant prior knowledge.

References

- [1] Brabazon, A., O'Neill, M.: *Biologically Inspired Algorithms for Financial Modelling*. Springer, Heidelberg (2006)
- [2] Brabazon, A., O'Neill, M. (eds.): *Natural Computing in Computational Finance*. Springer, Berlin (2008)
- [3] Brabazon, A., O'Neill, M., Dempsey, I.: An Introduction to Evolutionary Computation in Finance. *IEEE Computational Intelligence Magazine* 3(4), 42–55 (2008)
- [4] Chen, S.-H. (ed.): *Evolutionary Computation in Economics and Finance*. Physica-Verlag (2002)
- [5] de Castro, L.N.: Fundamentals of natural computing: an overview. *Physics of Life Reviews* 4(1), 1–36 (2007)
- [6] Kari, L., Rozenberg, G.: The many facets of natural computing. *Communications of the ACM* 51(10), 72–83 (2008)
- [7] LeBaron, B.: A Builder's Guide to Agent-Based Financial Markets. *Quantitative Finance* 1, 254–261 (2001)
- [8] LeBaron, B.: Agent-Based Computational Finance: Suggested Readings and Early Research. *Journal of Economic Dynamics and Control* 24(5-7), 679–702 (2000)
- [9] Tsang, E., Martinez-Jaramillo, S.: *Computational Finance*. IEEE Computational Intelligence Society Newsletter, pp. 8–13 (2004)
- [10] Wong, B., Lai, V., et al.: A bibliography of neural network business applications research: 1994–1998. *Computers and Operations Research* 27, 1045–1076 (2000)

Statistical Arbitrage with Genetic Programming

Philip Saks¹ and Dietmar Maringer²

¹ Centre for Computational Finance and Economic Agents (CCFEA), University of Essex, UK
psaks@essex.ac.uk

² University of Basel, Switzerland
dietmar.maringer@unibas.ch

Summary. This chapter employs genetic programming to discover statistical arbitrage strategies for a portfolio of banking stocks within the Euro Stoxx index. Binary decision rules are evolved using two different representations. The first is the classical single tree approach, where one decision tree for buy and sell orders is developed. The second version uses a dual tree structure where two decision trees are generated and the evaluation is contingent on the current market position. Hence, buy and sell rules are coevolved for long and short positions. Both single and dual trees are capable of discovering significant statistical arbitrage strategies, even in the presence of realistic market impact. This implies the existence of market inefficiencies within the chosen universe. However, the performance of the successful strategies deteriorate over time and the inefficiencies have disappeared in the second half of the out-of-sample period. The advantage of the dual trees, however, becomes apparent when transaction costs are increased and a clear asymmetric response between the two methodologies emerges. Naturally, increased costs have a negative impact on performance, but the dual trees are much more robust and can adapt to the changed environment, whereas the single trees cannot.

2.1 Introduction

During recent decades the *Efficient Markets Hypothesis* (EMH) has come increasingly under attack. Cognitive psychology has produced a large body of evidence against the rational agent model, which is a basic premise for efficient markets. Specifically, it has been shown that people rely on heuristics to simplify decision problems. This is both useful and necessary in everyday life, but in certain situations it can lead to biases such as base rate neglect, sample size neglect, insensitivity to predictability, effectiveness of a search set and insufficient adjustment [23]. What is more important is that these biases manifest themselves as anomalies and puzzles in the markets when viewed from an EMH perspective [3].

Where agents are heterogeneous and boundedly rational the market clearing price cannot be determined formally, since agents need to form expectations about other agents' expectations. This causes an infinite regress in subjectivity, such that expectations cannot be formed by deductive means. Thus, perfect rationality is not well-defined [2]. The *Adaptive Markets Hypothesis* (AMH) views the markets as ecosystems driven by evolutionary principles. Efficiency in this context is an emergent phenomenon driven by competition between agents, but it is not there by construction [17]. In such a world

it is indeed sensible to develop expectational models beyond traditional equilibrium analysis. In this paper, such models are built using *genetic programming* (GP).

The empirical literature has produced mixed results in this field. A classical contribution by Allen and Karjalainen [1] do not significantly outperform the buy-and-hold strategy on a daily frequency in the period from 1928 to 1995. In contrast, Becker and Seshadri [4] come to a positive conclusion by using monthly data, a reduced function set and more derived indicators. Moreover, buy and sell rules are coevolved separately. In foreign exchange markets, Neely, Weller, and Dittmar [18] generate significant excess returns using daily data until 1995, but a later study suggests that performance has since deteriorated [20]. There is a general consensus that significant inefficiencies existed in the high-frequency intraday domain in the early nineties [14, 5]. However, another result suggests that this has later changed [19].

In this chapter, we consider GP for statistical arbitrage. Arbitrage in the traditional sense is concerned with identifying situations where a self-funding portfolio is generated that will provide only non-negative cash flows at any point in time. Obviously, such portfolios are possible only in out-of-equilibrium situations. Statistical arbitrage is a wider concept where, again, self-funding portfolios are sought where one can expect non-negative pay outs at any point in time. Here one accepts negative pay-outs with a small probability as long as the expected positive payouts are high enough and the probability of losses is small enough; ideally this shortfall probability converges to zero. In practice, such a situation can occur when price processes are closely linked. In the classical story of Royal Dutch and Shell, the pair of stocks are cointegrated due to their fundamental link via their merger in 1907 [8]. In most cases, however, such links are not as obvious, but that does not eliminate the possibility that such relationships might exist and can be detected by statistical analysis. In this study, stocks within the same industry sector are considered, since it can be argued that these stocks are exposed to many of the same risk factors and should therefore have similar behavior.

As mentioned previously, an arbitrage portfolio is constructed by using the proceedings from short selling some stocks to initiate long positions in other stocks. More formally, the cumulative discounted value (v_t) of a statistical arbitrage strategy has to satisfy the following conditions [10]

$$v_0 = 0 \quad (2.1)$$

$$\lim_{t \rightarrow \infty} E(v_t) > 0 \quad (2.2)$$

$$\lim_{t \rightarrow \infty} \text{prob}(v_t < 0) = 0 \quad (2.3)$$

$$\lim_{t \rightarrow \infty} \frac{\text{Var}(v_t)}{t} = 0 \quad \text{if} \quad \text{prob}(v_t < 0) > 0 \quad \forall t < \infty \quad (2.4)$$

This means that it has to have zero initial cost and be self-financing (2.1); a positive discounted value (2.2); and a probability of loss converging to zero (2.3). Condition (2.4) states that a statistical arbitrage produces riskless incremental profits in the limit. By taking relative value bets between highly correlated stocks from the same industry, much of the market uncertainty is hedged away. Hence, profits made from this strategy are virtually uncorrelated with the market index. Furthermore, by modeling the relationships between stocks, the attention is focused in a direction where more stable patterns

should exist rather than making specific predictions about future developments. This statement defies the EMH in its weakest form, that no trading system based on historical price and volume information should generate excess returns [6].

The rest of the chapter is organized as follows. Sect. 2.2 gives a brief introduction to GP. Sect. 2.3 introduces the data, model and framework. Results are presented in Sect. 2.4, where trading is associated with realistic frictions. Sect. 2.5 contains a sensitivity analysis and stress testing where the effects of increasing transaction costs are investigated. Finally, Sect. 2.6 concludes and gives pointers to possible future research.

2.2 Genetic Programming

Genetic programming (GP) was pioneered by Koza [15] and is often seen as a derivative of genetic algorithms (GA). The GA was invented by Holland [11] in his ambitious quest to understand the principles of adaptive systems in a broad sense. An obvious inspiration came from biology, where the success of natural adaptive systems rests on competition and innovation in order to survive in changing and uncertain environments.

The GA is a population based search method, where the individuals (in the canonical version of the GA) are fixed length binary strings, known as *genotypes* or *chromosomes* [24]. Generally, this representation requires an encoding which is problem specific. For example if the binary GA is used for real-valued parameter optimization, then it is necessary to discretize the search space, where the resolution depends on the number of bits chosen to represent a given variable.

GAs work as follows. To begin with an initial population of M individuals is generated randomly in generation zero. Hereafter, the fitness of each individual is calculated according to the pre-specified objective function. Then a new population is created by selecting between the operators, reproduction, crossover and mutation according to the probabilities p_r , p_c and p_m , respectively. Each of these operators selects individuals from the parent population, such that better solutions are favored. A popular mechanism for doing this is *tournament selection*, in which a fixed number of individuals are chosen uniformly from the parent population, and the fittest individual wins the tournament and is selected. By controlling the tournament size it is possible to regulate the selection pressure. The reproduction operator simply copies the selected string to the new population. For the crossover operation, two individuals are selected from the parent population. Hereafter a position or index is uniformly selected within the bit-string and genetic material from the two parents are simply swapped around this point. The two resulting offspring are then inserted into the new population. The mutation operator simply selects a random element within an individual and negates the value, i.e., zero becomes one and vice versa. An advantage of the mutation operator is that it can introduce diversity into a population, but usually mutation is only invoked with a small probability. When the population size of the new population is equal to M , the algorithm has completed one generation and the process repeats itself until a termination criteria has been satisfied, e.g., until a maximum number of generations is reached.

Genetic programming is basically a GA operating on hierarchical computer programs instead of binary strings. Any problem that is concerned with finding an optimal mapping from a set of inputs to a set of outputs, can be reformulated as a search for

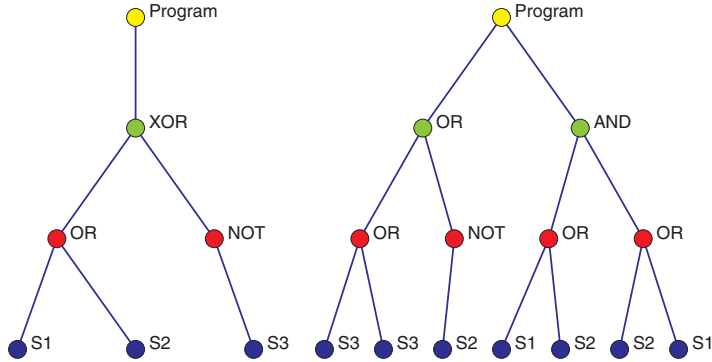


Fig. 2.1. Examples of a single (left) and dual tree (right), constructed using Boolean operators. The root node of the single tree is XOR, while the root nodes for the dual tree are OR and AND.

an optimal computer program. GP provides the means to search the space of possible programs. It is therefore a much more direct approach to problem solving than GAs, that are heavily dependent on problem encoding. In practice the individuals in GP are computer programs represented as tree structures. The programs are constructed from *functions* and *terminals*, where by definition the former take arguments and the latter do not. The sets of available functions and terminals for a given problem is known as the *function set* and *terminal set*. Traditionally, GP uses single trees to represent programs; for a general introduction to GP, see Koza [15]. Additionally, this paper considers a dual tree approach [16, 22]. Fig. 2.1 gives examples of a single and dual tree program constructed using Boolean operators.

2.3 Framework

2.3.1 Data

The data comprises of Volume-Weighted Average Prices (VWAP) and volume, sampled at an hourly frequency for stocks in the Euro Stoxx index. VWAP is widely used in industry, and it allows for a more precise estimation of transaction costs. It covers the time period from 01-Apr-2003 to 29-Jun-2007, corresponding to a total of 8648 observations. We only consider stocks from the same industry sector, namely Banking. It is frequently argued that stocks within the same industry sector are exposed to many of the same risk factors and should therefore have similar behavior. This should also increase the chances of discovering statistical arbitrage opportunities. Due to missing data, a portfolio of 30 out of 48 assets is considered. The components and summary statistics are documented in Table 2.4 and the VWAP prices are depicted in Fig. 2.2. In order to extract information from the raw data, a number of indicators are constructed via preprocessing.

When analyzing high frequency data it is important to take intraday effects into account, e.g., the intraday volume is higher after open and before close than during the middle of the day [12]. In the context of trading rule induction it is important to remove this bias, which is basically a proxy for the time of day, and prohibits sensible

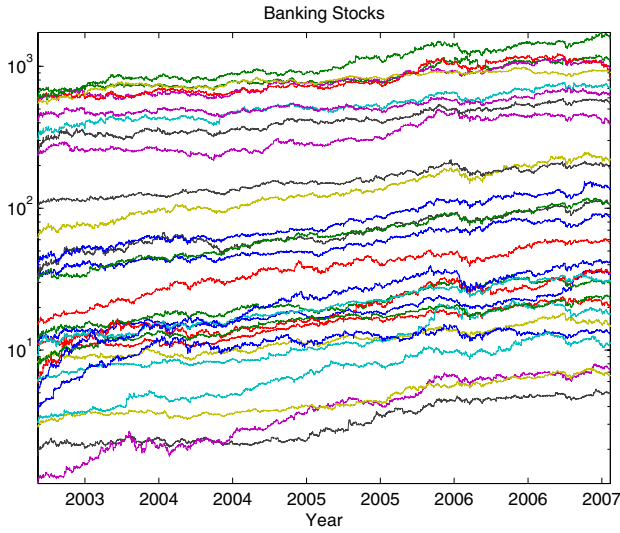


Fig. 2.2. Hourly VWAP prices for banking stocks within the Euro Stoxx index

conditioning on intraday volume. A volume indicator is constructed that removes the intraday bias. Specifically, we take the logarithm of the ratio between the realized and expected volume, where this ratio has been hard limited in the range between 0.2 and 5. The return series for each stock is standardized with respect to its volatility, estimated using simple exponential smoothing.

Since the main focus is on cross-sectional relationships between stocks, rather than their direction, we subtract the cross-sectional average from the normalized returns and volume series for each stock. Based on these series, we calculate the moving averages over the last 8, 40 and 80 periods; at an hourly frequency this corresponds to one day, a week and two weeks, respectively. All indicators are scaled in the range between zero and one.

2.3.2 Model

The strategies evolved in this study, return Boolean values corresponding to long and short positions. As mentioned previously, the focus of this study is on arbitrage portfolios, where the purchase of stocks is financed by short selling others. Naturally, a precondition for this is that not all the forecasts across the 30 stocks are the same. For example if the trading rule takes a bullish view across the board, then short-selling opportunities have not been identified and proper arbitrage portfolios cannot be constructed. In this case no stocks are held. However, when forecasts facilitate portfolio construction, long and short positions are taken, respectively. In this case, the portfolio weights are adjusted for the stocks' volatilities.

More technically, let $o_t^i \in \{-1, 1\}$ denote the forecast on stock i at time t , such that -1 and 1 corresponds to a bearish and bullish view, respectively. Then the holding is given by

$$h_t^i = o_t^i \cdot \frac{\frac{1}{\sigma_t}}{\sum_{j=1}^n \frac{1}{\sigma_t^j} \cdot I_{\{o_t^j = o_t^i\}}} \quad (2.5)$$

where σ_t is the volatility, n is the number of stocks in the universe, and $I_{\{o_t^j = o_t^i\}}$ is an indicator variable that ensures that forecasts are normalized correctly, i.e., it discriminates between long and short positions. By construction the long and short positions sum to -1 and 1, respectively. By down weighting more volatile stocks the portfolio becomes more stable.

$$\begin{aligned} \sum_{j=1}^n h_t^j I_{\{h_t^j > 0\}} &= 1 \\ \sum_{j=1}^n h_t^j I_{\{h_t^j < 0\}} &= -1 \end{aligned} \quad (2.6)$$

We employ two different methods for solving the binary decision problem. The first uses a standard single tree structure, while the second considers a dual tree structure in conjunction with cooperative coevolution [4]. In both methods, the trees return boolean values.

For the dual tree structure, program evaluation is contingent on the current market position for that particular stock, i.e., the first tree dictates the long entry, while the second enters a short position. In other words, which of these two trees is evaluated, depends on the previous position; if stock i at time t was in a short position ($o_{t-1}^i < 0$), then tree $k = 1$ is evaluated and dictates if a long position should be initiated. Alternatively, tree $k = 2$ is evaluated to decide whether to enter a short position. Let $b_t^{k,i} \in \{0, 1\}$ be the truth value for tree k on stock i at time t , whether or not to switch positions, then $b_t^{1,i} = 1$ ($b_t^{2,i} = 1$) indicates to enter a long (short) position, while $b_t^{1,i} = 0$ ($b_t^{2,i} = 0$) leaves the current position unchanged. Then the new forecast is given as

$$o_t^i = \begin{cases} 2 \cdot b_t^{1,i} - 1 & \text{if } o_{t-1}^i < 0 \\ -2 \cdot b_t^{2,i} + 1 & \text{otherwise} \end{cases} \quad (2.7)$$

All trees are constructed from the same grammar, which in addition to type constraints also introduces semantic restrictions. This improves the search efficiency significantly, since computational resources are not wasted on nonsensical solutions [5]. We consider a fairly restricted grammar, which is documented in Table 2.1. It consists of numeric comparators, boolean operators and *if-then-else* statements (ITE). Furthermore, a special function BTWN has been introduced, that takes three arguments and evaluates if the first is between the second and third. The terminals comprise of the six return and volume indicators introduced in Sect. 2.3.1. A distinction is made between return (qrtn) and volume (qvo1) information. Additionally, the terminal set includes numerical real-valued constants (qconst) ranging from 0 to 1. The parsimonious grammar reduces the risk of overfitting, and enhances interpretability of the evolved solutions.

The choice of a suitable objective function is essential in evolutionary computation. Previous studies suggest that a risk-adjusted measure improves out-of-sample performance when compared to an absolute return measure [5]. In this context, the ratio between the average profits and their volatility would be an obvious candidate. However,

Table 2.1. Statistical arbitrage grammar, where BTWN checks if the first argument is *between* the second and third. ITE represents the *if-then-else* statement.

Function	Arguments	Return Type
<, >	(qrtn, qconst)	bool
<, >	(qvol, qconst)	bool
BTWN	(qrtn, qconst, qconst)	bool
BTWN	(qvol, qconst, qconst)	bool
AND, OR, XOR	(bool, bool)	bool
NOT	(bool)	bool
ITE	(bool, bool, bool)	bool

under this measure strategies might evolve that do extremely well only on a subset of the in-sample data and mediocre on the remainder; from a practical point of view, this can lead to additional vulnerability to market timing as the overall success might depend more on the entry and exit points than on the overall time. Instead, the *t-statistic* of the linear fit between cumulated profits and time is employed, since it maximizes the slope while minimizing the deviation from the ideal straight line performance graph. This measure favors a steady increase in wealth.

2.3.3 Parameter Settings

In the following computational experiments a population of 250 individuals is initialized using the *ramped half-and-half* method. It evolves for a maximum of 51 generations, but is stopped after 15 generations if no new elitist (*best-so-far*) individual has been found. A normal tournament selection is used with a size of 5, and the crossover and mutation probabilities are 0.9 and 0.1, respectively. Moreover, the probability of selecting a function node during reproduction is 0.9, and the programs are constrained to a maximum complexity of 50 nodes. Again, this constraint is imposed to minimize the risk of overfitting, but also to facilitate interpretability. If the models lose tractability, it defies the purpose of GP as a knowledge discovery tool.

The data is split into a training and test set. The former contains 6000 samples and covers the period from 01-Apr-2003 to 10-Mar-2006, and the latter has 2647 samples in the period from 13-Mar-2006 to 29-Jun-2007.

2.4 Empirical Results

In practice, trading is associated with market impact, in particular on a high frequency level in a market with continuous auctions. The slippage depends on the order size relative to the liquidity of the stock and the time horizon over which the VWAP is executed. It is a common assumption in the industry that a good execution algorithm is capable of targeting the VWAP within one basis point for moderate order sizes.

Trading on the VWAP differs from a traditional market order, where a trade is executed at the current observed price. The VWAP is a backward looking measure, and it is therefore not possible to trade on the observed VWAP at time t . Instead the execution

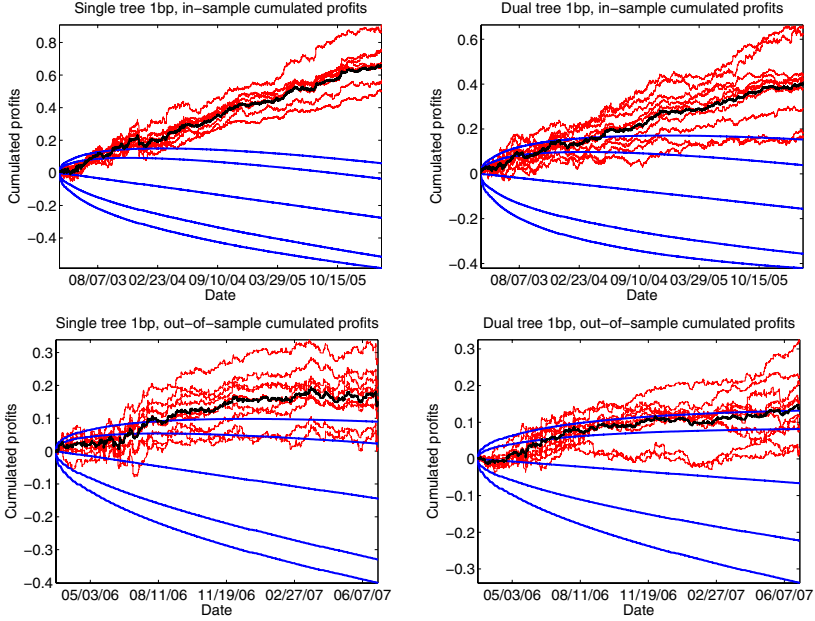


Fig. 2.3. In-sample (top) and out-of-sample cumulated profits (bottom) of the ten strategies, assuming a market impact of 1bp. The thick solid line is average performance and the 95% and 99% confidence intervals are constructed using the stationary bootstrap procedure. Single trees (left column) and dual trees (right column).

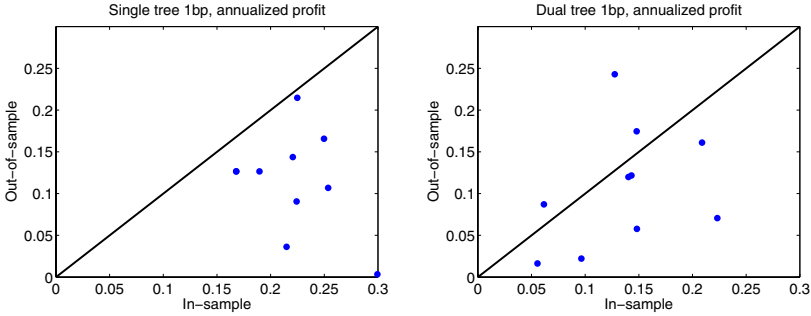


Fig. 2.4. Annualized in-sample versus out-of-sample profits with the 45°-line. Single trees (left) and dual trees (right).

occurs gradually between t and $t + 1$, resulting in the VWAP at $t + 1$. In summary, a trading decision is formed based on the VWAP at time t , the entry price is observed at time $t + 1$ and the one period profit is evaluated at $t + 2$.

We perform 10 experiments using both the single and dual tree method, according to the settings outlined in Sect. 2.3.3. For each experiment, the *best-so-far* individual is evaluated on the training and test set. The self-financing property of a statistical arbitrage

Table 2.2. Strategy performance statistics of single trees under 1bp market impact. The following abbreviations are used; TF – *t*-statistic fitness, AP – annualized profits, PRR – profit-risk ratio, MDD – maximum drawdown, TO – average daily turnover.

Strategy	In-sample					Out-of-sample				
	TF	AP	PRR	MDD	TO	TF	AP	PRR	MDD	TO
1.	1324	0.250	2.64	-0.089	3.91	256	0.166	1.54	-0.089	3.75
2.	1368	0.254	2.68	-0.072	4.51	198	0.107	1.00	-0.088	4.36
3.	1257	0.300	2.45	-0.082	5.37	55	0.003	0.03	-0.157	5.07
4.	979	0.215	2.67	-0.060	3.03	177	0.036	0.40	-0.076	2.95
5.	1065	0.225	2.36	-0.085	3.65	259	0.215	2.02	-0.077	3.71
6.	909	0.168	2.60	-0.053	2.94	227	0.127	1.95	-0.053	2.81
7.	1349	0.221	2.69	-0.078	4.08	219	0.144	1.56	-0.051	4.05
8.	963	0.190	1.97	-0.114	3.18	288	0.127	1.11	-0.101	3.03
9.	909	0.168	2.60	-0.053	2.94	227	0.127	1.95	-0.053	2.81
10.	1205	0.224	2.38	-0.098	4.18	286	0.091	0.89	-0.073	4.10
Aggregate	1651	0.221	3.12	-0.073	2.98	274	0.114	1.48	-0.058	2.88

Strategy	Out-of-sample, 1 st half					Out-of-sample, 2 nd half				
	TF	AP	PRR	MDD	TO	TF	AP	PRR	MDD	TO
1.	181	0.278	2.28	-0.059	4.04	81	0.053	0.58	-0.089	3.45
2.	170	0.256	2.17	-0.058	4.53	35	-0.042	-0.45	-0.088	4.20
3.	20	0.071	0.46	-0.157	5.27	-15	-0.064	-0.57	-0.082	4.88
4.	67	0.035	0.35	-0.043	2.90	82	0.038	0.46	-0.076	3.00
5.	316	0.385	3.56	-0.048	3.90	79	0.044	0.43	-0.077	3.52
6.	235	0.206	3.01	-0.053	2.86	38	0.047	0.77	-0.046	2.75
7.	228	0.233	2.22	-0.048	4.21	51	0.054	0.70	-0.051	3.88
8.	98	0.186	1.65	-0.073	3.24	107	0.067	0.58	-0.101	2.82
9.	235	0.206	3.01	-0.053	2.86	38	0.047	0.77	-0.046	2.75
10.	96	0.114	1.02	-0.056	4.41	103	0.067	0.74	-0.073	3.78
Aggregate	214	0.197	2.41	-0.038	3.00	82	0.031	0.44	-0.058	2.76

portfolio implies that its return in the strict sense is not well defined: The return of an investment is the ratio of terminal to initial wealth ($v_t/v_0 - 1$), but by definition $v_0 = 0$ for statistical arbitrage portfolios. Instead of the returns, the log-profits are evaluated at each time period

$$p_t = \sum_{i=1}^n r_t^i \cdot h_t^i \quad (2.8)$$

where r_t^i is the log-return of stock i at time t . Due to the constraint (2.6), the log-profit approximates the money amount made from investing one currency unit on both the long and short side of the portfolio. In the subsequent analysis profits and wealth, refers to log-profits and log-wealth unless otherwise specified.

Fig. 2.3 shows the growth in wealth for the evolved trading strategies, and Tables 2.2 and 2.3 provide more detailed performance statistics such as the *t*-statistic fitness (TF), annualized profits (AP), profit-risk ratios (PRR; ratio between profits and their standard deviation), maximum draw down (MDD) and turnover (TO). Casual inspection of the

Table 2.3. Strategy performance statistics of dual trees under 1bp market impact. The following abbreviations are used; TF – *t*-statistic fitness, AP – annualized profits, PRR – profit-risk ratio, MDD – maximum drawdown, TO – average daily turnover.

Strategy	In-sample					Out-of-sample				
	TF	AP	PRR	MDD	TO	TF	AP	PRR	MDD	TO
1.	439	0.140	2.14	-0.050	1.16	237	0.120	1.77	-0.036	1.11
2.	821	0.148	2.25	-0.059	1.30	105	0.058	0.85	-0.077	1.24
3.	824	0.209	2.05	-0.084	2.58	249	0.161	1.45	-0.058	2.41
4.	289	0.062	0.96	-0.086	2.38	141	0.087	1.39	-0.074	2.41
5.	1775	0.223	2.86	-0.049	1.31	128	0.071	0.95	-0.076	1.08
6.	379	0.056	0.72	-0.099	3.03	26	0.016	0.22	-0.074	2.96
7.	827	0.096	1.53	-0.067	0.70	43	0.022	0.34	-0.071	0.61
8.	802	0.128	1.91	-0.062	1.66	352	0.243	3.56	-0.045	1.63
9.	1219	0.143	2.31	-0.046	0.68	311	0.122	1.95	-0.040	0.65
10.	1320	0.148	2.46	-0.043	0.46	476	0.175	2.76	-0.030	0.42
Aggregate	1143	0.135	3.02	-0.038	1.19	263	0.107	2.41	-0.026	1.13

Strategy	Out-of-sample, 1 st half					Out-of-sample, 2 nd half				
	TF	AP	PRR	MDD	TO	TF	AP	PRR	MDD	TO
1.	139	0.162	2.16	-0.036	1.08	103	0.077	1.31	-0.028	1.14
2.	132	0.118	1.56	-0.042	1.21	-0	-0.002	-0.04	-0.076	1.27
3.	123	0.222	1.88	-0.046	2.45	9	0.100	0.96	-0.058	2.38
4.	169	0.142	2.12	-0.038	2.53	2	0.032	0.56	-0.074	2.29
5.	193	0.196	2.49	-0.038	1.08	-19	-0.054	-0.78	-0.076	1.07
6.	23	0.010	0.13	-0.060	2.92	4	0.022	0.31	-0.068	3.01
7.	36	0.008	0.13	-0.052	0.63	32	0.036	0.56	-0.052	0.59
8.	155	0.267	3.54	-0.045	1.57	122	0.219	3.64	-0.022	1.69
9.	124	0.158	2.30	-0.040	0.64	124	0.086	1.55	-0.031	0.66
10.	199	0.196	2.77	-0.030	0.41	174	0.153	2.79	-0.027	0.42
Aggregate	208	0.148	3.03	-0.023	1.14	95	0.067	1.67	-0.026	1.13

in-sample results reveal that the *t*-statistic measure works as intended, since all strategies have steady increasing wealth over time. The values range between 909 and 1368 for the single trees, and 379 and 1775 for the dual trees. The annualized profits range between 0.168 and 0.254 with an average of 0.221 for the single trees, and 0.056, 0.223 and 0.135 are the equivalent statistics for the dual trees.

In practice, a statistical arbitrage strategy requires a margin deposited in a risk-free account. By taking additional exposure in the self financing risky strategy relative to the margin the profits can be scaled to suit investor utility. Hence, they are merely a function of leverage. The PRR which is leverage invariant is therefore a more descriptive measure. Here the range is between 1.97 and 2.69 for the single trees, and 0.72 and 2.86 for the dual trees. Due to the stochastic nature of GP, the evolved rules are generally different and it is therefore possible to improve the performance due to diversification. The aggregated holdings are simply the average of the holdings for the ten individual strategies. Under aggregation, the PRRs increase to 3.12 and 3.02, for the single and dual trees, respectively. Unfortunately, aggregation or bagging destroys

any simple structure of the model, or in other words it, “a bagged tree is no longer a tree” [9]. Consequently, interpretability is lost, which is clearly a drawback. However, by bagging the evolved strategies it is possible to make general inferences about their properties. In this study all ten evolved strategies are aggregated, but one could employ various schemes to improve out-of-sample performance. Generally, this requires the use of additional validation sets, but since data is limited this is problematic. Moreover, aggregating all strategies is clearly the conservative approach and is therefore preferred. The worst in-sample drawdowns are 0.114 and 0.099 for the single and dual trees, respectively. Under aggregation they fall to 0.073 and 0.038.

Another important statistic is the average daily turnover, which measures the extent to which the portfolio holdings (Eq. (2.5)) are changing. Formally it is defined as

$$\tau = \frac{1}{T} \sum_{t=1}^T \sum_{i=1}^n |\Delta h_t^i| \quad \Delta h_t^i = h_t^i - h_{t-1}^i \quad h_0^i = 0 \quad (2.9)$$

where T is the number of time periods. For the single trees it ranges between 2.94 to 5.37, whereas the turnover for the dual trees is much lower, 0.46 to 3.03.

Naturally, the value of a trading strategy is not dictated by its in-sample performance but is assessed out-of-sample. A drawback of the t -statistic measure is that it is not sample-size invariant, hence out-of-sample and in-sample results are not comparable. On an aggregate level the TF is 274 and 263 for the single and dual trees, respectively. The annualized profits for the single trees range between 0.003 and 0.215 with an average of 0.114, while for the dual trees the numbers are 0.022, 0.243 and 0.107. Likewise, the PRRs vary between 0.03 and 2.02 for the single trees and, 0.22 and 3.56 for the dual trees.

To investigate the strategies market timing capabilities confidence intervals are constructed using the *stationary bootstrap* method, which is a superior alternative to well known *block bootstrap* procedure [21]. Instead of using a fixed block size, it varies probabilistically according to a geometric distribution. With a probability parameter of $p = 0.01$, it generates blocks with an expected length of 100 samples. Thus, sampling with replacement is performed from the strategy holdings, and statistics are gathered from 500 runs. For the single tree method eight out of ten (8/10) strategies exceed the 99% upper confidence limit at the end of the out-of-sample period, while only 5/10 do amongst the dual trees.

Despite good overall out-of-sample performance, Fig. 2.3 reveals that it is deteriorating as a function of time. To examine this in more detail, Tables 2.2 and 2.3 report the out-of-sample performance statistics on two sub-periods, from 13-Mar-2006 to 02-Nov-2006 and from 02-Nov-2006 until 29-Jun-2007. In the first half, both the single and dual trees generalize extremely well obtaining average annualized profits of 0.197 and 0.148, where the latter actually exceeds its in-sample performance. Moreover, the TF of the aggregated dual trees is considerably higher than the average TF for the individual strategies. In the second period, however, the single trees fare poorly, and make on average 0.031, while the performance of the dual trees halved to 0.067. A similar conclusion is reached by analyzing the profit-risk ratios.

Positive out-of-sample performance need not imply market inefficiency. Traditionally, this is investigated by comparing the trading strategy to the buy-and-hold strategy,

i.e., a passive long only portfolio. This, however, is not a suitable benchmark for statistical arbitrage strategies. This is mainly because a statistical arbitrage is self-financing and the buy-and-hold is not. Obviously one could short the risk-free asset and invest in the proceeds in an equally weighted portfolio of the underlying stocks. This benchmark has an annualized profit of 0.045, and a PRR of 0.35 in the out-of-sample period. However, this is a naive approach contingent on a specific equilibrium model. In benchmarking this gives rise to the joint hypothesis problem, that abnormal returns need not imply market inefficiency, but can be due to misspecification of a given equilibrium model [7]. Another benchmark which closer to the spirit of the statistical arbitrage application presented in this paper, is based on the idea that stocks exhibit momentum [13]. Based on the in-sample returns, a portfolio is formed by selling (buying) the bottom (top) quintile with respect to performance. In the out-of-sample period this portfolio generates an annualized profit of 0.069 and has a PRR of 0.61, but its maximum drawdown is a substantial 0.157. This is clearly inferior to both the single and dual trees.

A better alternative to these types of benchmarks is to employ a special statistical test for statistical arbitrage strategies, which circumvents the joint hypothesis problem [10]. The profits are made from investing one currency unit in both the long and short positions, but they are not compounded, instead they are invested in a risk-free account. Hence, proportionally less are invested in the risky strategy over time. As a discount rate we employ the 1-month LIBOR rate for the Eurozone. The constant mean version of the test assumes that the discounted incremental profits satisfy

$$\Delta v_i = \mu + \sigma i^\lambda z_i \quad i = 1, 2, \dots, n \quad (2.10)$$

where $z_i \sim N(0, 1)$. The joint hypothesis, $H1 : \mu > 0$ and $H2 : \lambda < 0$ determines the presence of statistical arbitrage. The p -values for the joint hypothesis are obtained via the *Bonferroni inequality*, $\text{prob}(\bigcup_{i=1}^n A_i) \leq \sum_{i=1}^n \text{prob}(A_i)$. The Bonferroni inequality provides an upper bound for the likelihood of the joint event, by simply summing the probabilities for the individual events without subtracting the probabilities of their intersections.

Tables 2.5 and 2.6 provide the statistics for the statistical arbitrage test [10]. For entire out-of-sample period, 2/10 single trees and 5/10 dual trees reject the null hypothesis at a 0.05 level of significance. Likewise, both aggregate models constitute significant statistical arbitrage strategies. For the first half of the out-of-sample period, 4/10 single trees and 7/10 dual trees are significant. During the second half none of the strategies are significant. Based on these observations it must be concluded that the markets are not efficient in this high frequency domain. However, it also holds that these inefficiencies disappear over time, or rather, that a static model can only be expected to have a limited lifespan in a dynamic market.

2.5 Sensitivity Analysis

2.5.1 Decomposition and Timing

A unique feature of statistical arbitrage strategies is the equity market neutral constraint, i.e., the long and short positions balance out each other. To gain further understanding

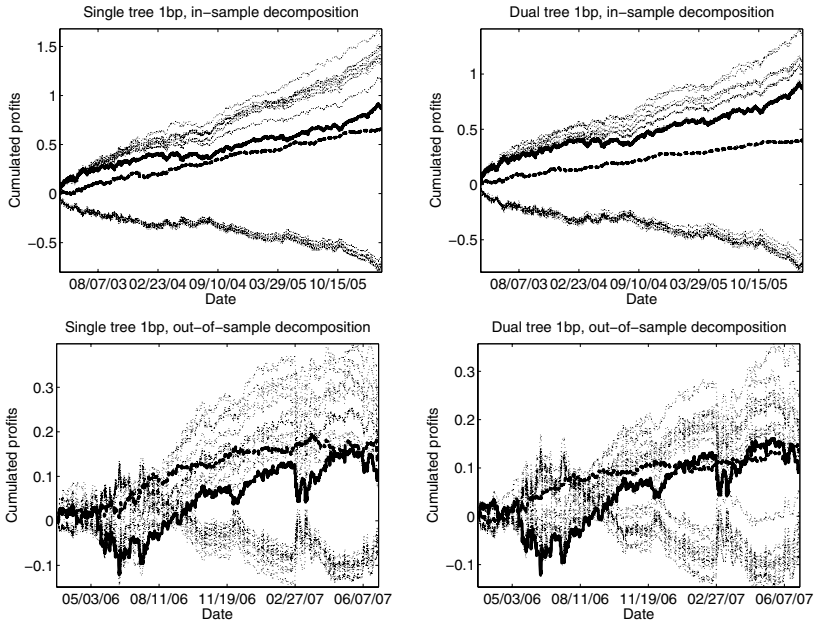


Fig. 2.5. Decomposition of long and short position profits. The thick dashed line is the aggregate strategy performance and the thick solid line is the cumulated returns of an equally weighted portfolio of banking stocks. Single trees (left column) and dual trees (right column).

of the portfolios it is instructive to decompose the profits from the long and the short side. Fig. 2.5 shows this decomposition. During the in-sample period the market has a phenomenal bull run and appreciates by approximately 90%, which implies that the short positions are loss making for both the single and dual trees. The long side, however, outperforms the market by up to 50%. Naturally, in this scenario financing via the riskless asset would be a better option than short selling the stocks. The problem is that this strategy requires knowledge about the market direction *ex ante*.

The out-of-sample period contains the crash of May 2006, during which the short side makes money and prevents a large drawdown which would otherwise have occurred with cash financing. By hedging the long side using highly correlated stocks within the same industry sector the majority of market uncertainty disappears, which results in strategies with higher profit-risk ratios.

On the individual stock level, the graphs in Fig. 2.6 show the conditional distribution of the long positions for the bagged models in the out-of-sample period. Specifically, for each stock all the positive holdings are summed over time, and are then normalized by the total sum of the positive holdings for all stocks. As expected there are variations across stocks, but all of them are held at some point. The dual tree holdings appear slightly more uniform. However, there is a strong positive correlation (0.81) between the holdings of the two methods. The correlation between the out-of-sample returns for the bagged models is 0.59, which indicates that similar dynamics are discovered. The previous bootstrap exercise suggests that many of the strategies have significant market

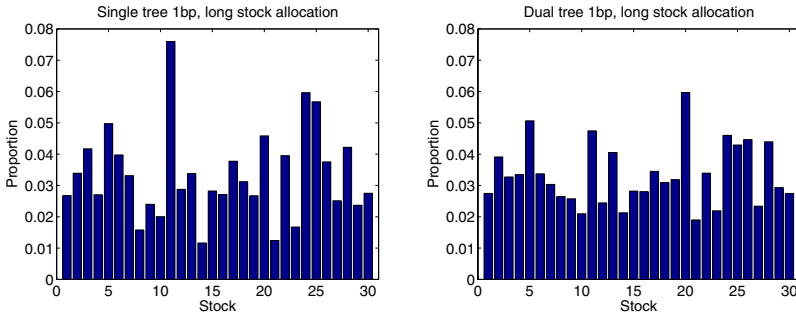


Fig. 2.6. Conditional distribution of long positions across stocks. Single trees (left) and dual trees (right).

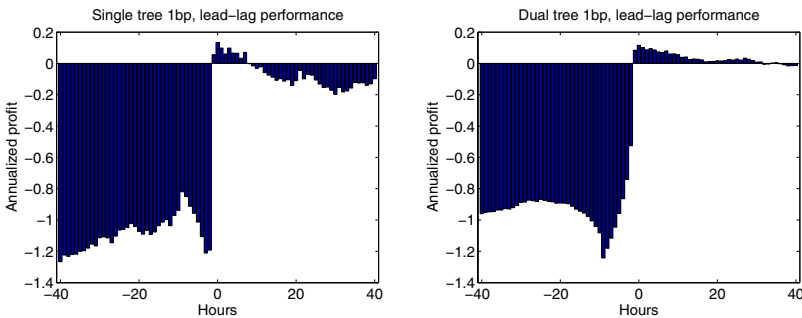


Fig. 2.7. Lead-lag performance of bagged models. Single trees (left) and dual trees (right).

timing capabilities. For practical purposes it is important to realize just how sensitive the performance is with respect to timing. This is especially true in the field of high frequency finance. To assess the temporal robustness the *lead-lag* performance is considered. In this analysis, the holdings of the bagged models are shifted in time relative to the VWAP returns, and the annualized strategy returns are evaluated. In Fig. 2.7 the lead-lag performance is calculated up to one week prior and after signal generation, and some very interesting results emerge. For both the single and dual trees, leading the signal results in significantly negative performance, contrary to intuition where decision making based on future information should improve results dramatically. This, however, is not the case for mean reverting signals. Consider a scenario where a stock has a large relative depreciation, after which speculators believe it is undervalued and therefore buy it, causing the price to appreciate in the subsequent time period. Had the buy decision been made one period earlier, it would have resulted in a great loss, due to the large initial depreciation.

When the holdings are lagged, the performance deteriorates gradually. For the single trees it has disappeared after 8 hours, i.e., a trading day, while the effect persists for the dual trees up to four days. We suspect this can be attributed to different ways, in which the two methods capture the underlying dynamics of the system. The single trees attempt to classify whether a stock is in a relative *bull* or *bear* regime, while the

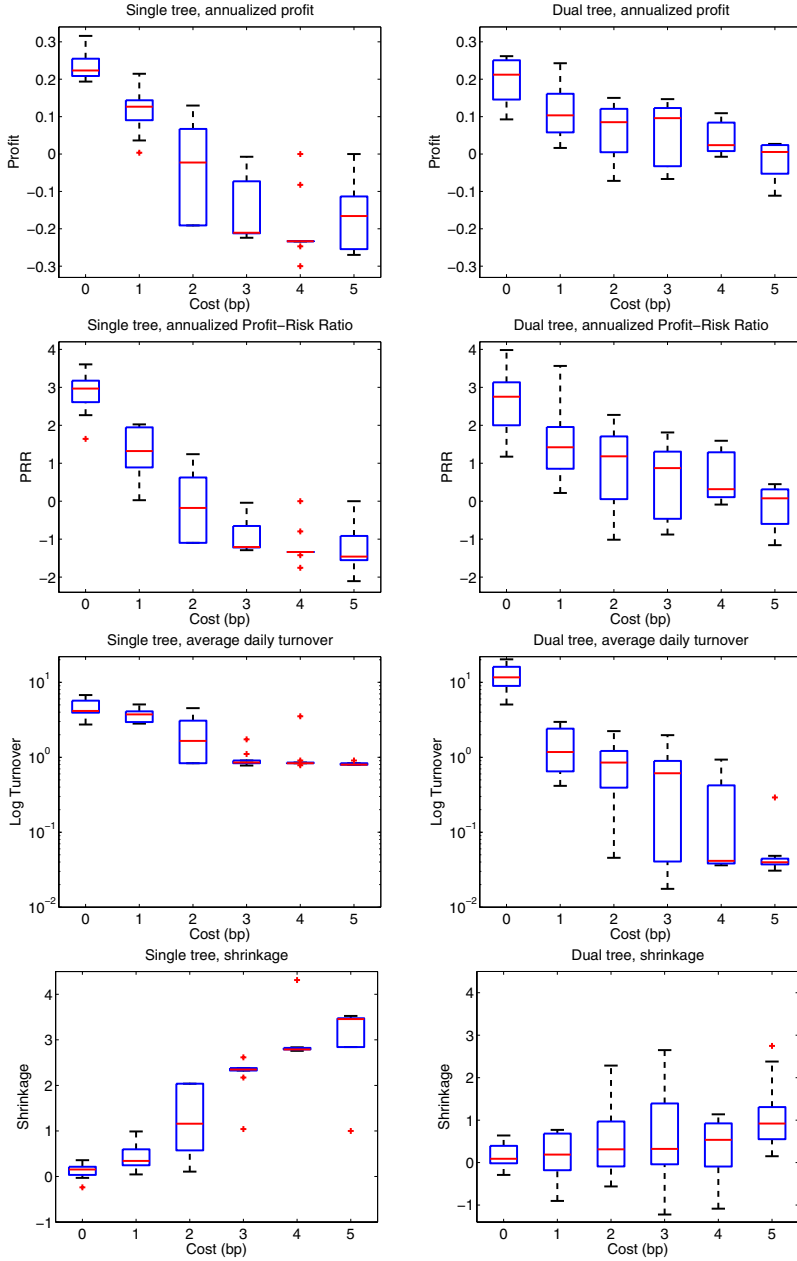


Fig. 2.8. Annualized returns (top), Sharpe ratios (center top), average daily turnover (center bottom) and shrinkage (bottom) as a function of market impact. Single trees (left column) and dual trees (right column).

dual trees have implicit knowledge of the current regime, and therefore models regime changes. The latter method has more information which probably enables better and more robust decision making. In passing, note that a single tree can be expressed as a dual tree with two equivalent trees, such that decision making independent from the current market position.

2.5.2 Stress Testing

Naturally, trading cost has an adverse effect on performance. In the presence of 1bp market impact both methods have similar out-of-sample performance, however they react quite differently when the trading cost is increased, as illustrated in Fig. 2.8. At only 2bp market impact, the single trees break down, yielding an average annualized profit of -0.040. The dual trees are much more robust and have positive out-of-sample performance up to 4bp with an annualized profit of 0.039. However they capitulate at 5bp, resulting in a negative average performance of -0.013. A similar conclusion is reached by analyzing the profit-risk ratios. Thus, it can be inferred that the volatilities of the portfolios are fairly orthogonal to changes in transaction costs. This is not surprising, since the portfolio holdings are constructed on a volatility adjusted basis.

The turnover provides an explanation to the asymmetric impact on performance for the two methods. Fig. 2.8 demonstrates how the average daily turnover decreases as a function of market impact. Under the assumption of frictionless trading, the median daily turnover is 4.15 and 11.65, for the single and dual trees, respectively. Obviously, this is not viable when market impact is introduced and for the dual trees the turnover is greatly reduced to a median value of 1.17 at 1bp cost. For the single trees the effect is less pronounced and the median changes to 3.73. As the transaction costs are increased further, the median turnover of the dual trees continue to fall, while for the single trees it stagnates at a value slightly below one. This is clearly a manifestation of different ways in which the two models capture the underlying dynamics of the system as discussed previously.

From a generalization perspective, there is also substantial asymmetric impact due to trading costs. As a proxy for generalization, it is instructive to consider the *shrinkage* which is defined, $\psi = (X_{train} - X_{rest}) / X_{train}$, where X is an arbitrary performance measure [5]. The bottom panel in Fig. 2.8 shows the shrinkage based on annualized profits as a function of cost. At 2bp market impact the single trees have a median shrinkage above one, whereas the dual trees do not exceed that value even at 5bp. The shrinkage is only evaluated from models with positive in-sample performance, since that is a minimum requirement for out-of-sample application.

The poor generalization of the single trees can also be explained via the turnover. They simply cannot evolve portfolios with sufficiently low turnover, and therefore suffer more when transaction costs increase.

2.6 Conclusion

In this chapter genetic programming is employed to evolve trading strategies for statistical arbitrage. This is motivated by the fact that stocks within the same industry sector

should be exposed to the same risk factors and should therefore have similar behavior. Traditionally there has been a gap between financial academia and the industry. This also applies to statistical arbitrage, an increasingly popular investment style in practice, but to the authors' knowledge little formal research has been undertaken within this field. This chapter addresses this imbalance and aims to narrow this gap. We consider two different representations for the trading rules. The first is a traditional single tree structure, while the second is a dual tree structure in which evaluation is contingent on the current market position. Hence, buy and sell rules are coevolved. Both methods evolve models with substantial market timing, but what is more important significant statistical arbitrage strategies are uncovered even in the presence of realistic market impact. Using a special statistical arbitrage test [10] this leads to the conclusion of the existence of market inefficiencies within the chosen universe. However, it should be mentioned that during the second half of the out-of-sample period the performance deteriorates and any statistical arbitrage there might have been seems to have disappeared. Does this imply that the chosen universe have become efficient? We do not believe this is the case, rather the deterioration in performance is the manifestation of using a static model in a dynamic environment. This is consistent with the *Adaptive Markets Hypothesis* [17]. A natural avenue for future research is therefore to investigate the effects of adaptive retraining of the strategies.

A unique feature in statistical arbitrage is that long and short positions within the portfolio balance out each other. By decomposing the profits it is found that both sides of the portfolio are essential to the performance. The in-sample period is a massive bull market, during which more profits could have been generated using the risk-free asset for borrowing. This approach, however, is not viable. Firstly it requires the knowledge of market direction *ex ante*, and second, it defies the essence of statistical arbitrage where the "market" is effectively hedged out.

In the final part of the chapter the impacts of increased transaction costs are investigated. Not surprisingly performance deteriorates for both the single and dual trees. The impacts on the two methods are highly asymmetric. As transaction costs increase, the single trees are not capable of reducing their turnover sufficiently and consequently they suffer greatly out-of-sample. The dual trees, however, have implicit knowledge of the previous market position and can effectively adapt to the changed environments.

References

- [1] Allen, F., Karjalainen, R.: Using genetic algorithms to find technical trading rules. *Journal of Financial Economics* 51, 245–271 (1999)
- [2] Arthur, B., Holland, J.H., LeBaron, B., Palmer, R., Tayler, P.: Asset pricing under endogenous expectations in an artificial stock market. Tech. rep., Santa Fe Institute (1996)
- [3] Barberis, N., Thaler, R.: *Handbook of the Economics of Finance*, Elsevier Science, pp. 1052–1090 (2003)
- [4] Becker, L.A., Seshadri, M.: Gp-evolved technical trading rules can outperform buy and hold. In: *Proceedings of the Sixth International Conference on Computational Intelligence and Natural Computing* (2003)

- [5] Bhattacharyya, S., Pictet, O.V., Zumbach, G.: Knowledge-intensive genetic discovery in foreign exchange markets. *IEEE Transactions on Evolutionary Computation* 6(2), 169–181 (2002)
- [6] Fama, E.F.: Efficient capital markets: A review of theory and empirical work. *Journal of Finance* 25(2), 383–417 (1970)
- [7] Fama, E.F.: Market efficiency, long-term returns, and behavioral finance. *Journal of Financial Economics* 49, 283–306 (1998)
- [8] Froot, K.A., Dabora, E.M.: How are stock prices affected by the location of trade? *Journal of Financial Economics* 53(2), 189–216 (1999)
- [9] Hastie, T., Tibshirani, R., Friedman, J.: *The Elements of Statistical Learning*. Springer, Heidelberg (2001)
- [10] Hogan, S., Jarrow, R., Teo, M., Warachaka, M.: Testing market efficiency using statistical arbitrage with applications to momentum and value strategies. *Journal of Financial Economics* 73, 525–565 (2004)
- [11] Holland, J.H.: *Adaption in Natural and Artificial Systems*. Univesity of Michigan Press (1975)
- [12] Jain, P.C., Joh, G.H.: The dependence between hourly prices and trading volume. *Journal of Financial Economics* 23(3), 269–283 (1988)
- [13] Jegadeesh, N., Titman, S.: Returns to buying winners and selling losers: Implications for stock market efficiency. *Journal of Finance* 48(1), 65–91 (1993)
- [14] Jonsson, H., Madjidi, P., Nordahl, M.G.: Evolution of trading rules for the fx market or how to make money out of gp. Tech. rep., Institute of Theoretical Physics, Chalmers University of Technology (1997)
- [15] Koza, J.R.: *Genetic Programming: On the Programming of Computers by Means of Natural Selection*. MIT Press, Cambridge (1992)
- [16] Langdon, W.B.: *Genetic Programming and Data Structures: Genetic Programming + Data Structures = Automatic Programming*. Kluwer Academic Publishers, Dordrecht (1998)
- [17] Lo, A.W.: The adaptive markets hypothesis. *The Journal of Portfolio Management*, 15–28 (2004) (30th Anniversary issue)
- [18] Neely, C., Weller, P., Dittmar, R.: Is technical analysis in the foreign exchange market profitable? a genetic programming approach. *Journal of Financial and Quantitative Analysis* 32, 405–426 (1997)
- [19] Neely, C.J., Weller, P.A.: Intraday technical trading in the foreign exchange market. Tech. rep., Federal Reserve Bank of St. Luis (1999)
- [20] Neely, C.J., Weller, P.A., Ulrich, J.M.: The adaptive markets hypothesis: Evidence from the foreign exchange market. Tech. rep., Federal Reserve Bank of St. Louis (2007)
- [21] Politis, D.N., Romano, J.P.: The stationary bootstrap. *Journal of the American Statistical Association* 89(428), 1303–1313 (1994)
- [22] Saks, P., Maringer, D.: Single versus dual tree genetic programming for dynamic binary decision making. Tech. rep., Centre for Computational Finance and Economic Agents, University of Essex (2008)
- [23] Tversky, A., Kahneman, D.: Judgment under uncertainty: Heuristics and biases. *Science* 185(4157), 1124–1131 (1974)
- [24] Whitley, D.: A genetic algorithm tutorial. *Statistics and Computing* 4, 65–85 (1994)

A Appendix

Table 2.4. Annualized returns and volatilities for banking stocks within the Euro Stoxx index

	Name	Return	Volatility
1.	ABN AMRO Holding NV	26.6	18.8
2.	Alliance & Leicester PLC	13.6	17.6
3.	Allied Irish Banks PLC	14.3	17.9
4.	Barclays PLC	19.0	20.2
5.	Bradford & Bingley PLC	11.9	19.7
6.	Bank of Ireland	13.8	18.8
7.	Banca Monte dei Paschi di Siena SpA	23.7	19.0
8.	BNP Paribas	24.1	19.9
9.	Credit Agricole SA	21.5	19.9
10.	Commerzbank AG	41.5	26.9
11.	Natixis	25.1	20.5
12.	Capitalia SpA	43.2	24.3
13.	UniCredito Italiano SpA	19.3	15.2
14.	Deutsche Bank AG	26.8	20.8
15.	Depfa Bank PLC	29.0	24.4
16.	Dexia SA	26.3	17.2
17.	Erste Bank	31.3	20.5
18.	Fortis	28.0	19.2
19.	HBOS PLC	13.2	17.9
20.	HSBC Holdings PLC	12.0	12.4
21.	Lloyds TSB Group PLC	18.9	18.9
22.	National Bank of Greece SA	49.0	24.3
23.	Nordea Bank AB	29.3	17.5
24.	Northern Rock PLC	9.9	18.2
25.	Banca Popolare di Milano Scarl	30.0	20.4
26.	Royal Bank of Scotland Group PLC	10.1	17.2
27.	Skandinaviska Enskilda Banken AB	29.7	19.3
28.	Svenska Handelsbanken AB	15.3	15.2
29.	Societe Generale	29.0	20.3
30.	Standard Chartered PLC	23.1	18.8

Table 2.5. Out-of-sample statistical arbitrage test results for single trees under 1bp market impact

Out-of-sample						
Strategy	$\mu (\cdot 10^{-4})$	$\sigma (\cdot 10^{-5})$	$\lambda (\cdot 10^{-1})$	H1	H2	H1+H2
1.	0.7090	3.8864	-1.4343	0.0556	0.0000	0.0556
2.	0.8319	5.5430	-2.4060	0.1738	0.0000	0.1738
3.	-0.0932	23.0714	-4.8673	0.5226	0.0000	0.5226
4.	0.8296	4.5428	-3.1347	0.2932	0.0000	0.2932
5.	5.5094	2.3301	0.4183	0.0067	1.0000	1.0067
6.	3.5796	3.2687	-4.3740	0.0151	0.0000	0.0151
7.	4.3059	19.4080	-9.8599	0.0542	0.0000	0.0542
8.	5.2725	4.6709	0.2396	0.0851	0.9661	1.0512
9.	5.3695	4.9030	-6.5609	0.0151	0.0000	0.0151
10.	4.3570	22.1748	-11.1114	0.1513	0.0000	0.1513
Aggregate	0.6456	0.6537	-0.6220	0.0233	0.0000	0.0233

Out-of-sample, 1 st half						
Strategy	$\mu (\cdot 10^{-4})$	$\sigma (\cdot 10^{-5})$	$\lambda (\cdot 10^{-1})$	H1	H2	H1+H2
1.	1.4027	1.5993	-0.6343	0.0292	0.0000	0.0292
2.	2.5672	3.7179	-1.6253	0.0361	0.0000	0.0361
3.	1.1144	11.7142	-2.9947	0.3431	0.0000	0.3431
4.	0.9082	3.0801	-1.5716	0.3523	0.0000	0.3523
5.	9.6970	2.0578	1.3027	0.0016	1.0000	1.0016
6.	6.1911	3.7696	-4.9463	0.0063	0.0000	0.0063
7.	8.3920	2.9048	1.4888	0.0306	1.0000	1.0306
8.	8.2825	15.5883	-7.4196	0.0639	0.0000	0.0639
9.	9.2866	5.6544	-7.4195	0.0063	0.0000	0.0063
10.	5.8946	16.6138	-8.1364	0.1912	0.0000	0.1912
Aggregate	1.1082	0.6270	-0.5279	0.0129	0.0000	0.0129

Out-of-sample, 2 nd half						
Strategy	$\mu (\cdot 10^{-4})$	$\sigma (\cdot 10^{-5})$	$\lambda (\cdot 10^{-1})$	H1	H2	H1+H2
1.	0.4878	0.1051	1.0554	0.1818	1.0000	1.1818
2.	0.1644	0.1533	2.6634	0.4395	1.0000	1.4395
3.	-0.5132	0.2623	4.5488	0.6053	1.0000	1.6053
4.	1.7130	0.2092	5.6897	0.1814	1.0000	1.1814
5.	3.4272	0.1224	11.6530	0.1006	1.0000	1.1006
6.	1.7482	0.3956	4.8485	0.2142	1.0000	1.2142
7.	2.8709	0.6345	6.4806	0.1874	1.0000	1.1874
8.	5.4380	0.1808	20.3875	0.1202	1.0000	1.1202
9.	2.6223	0.5934	7.2727	0.2142	1.0000	1.2142
10.	5.4774	0.8301	12.1285	0.1479	1.0000	1.1479
Aggregate	0.4813	0.0332	1.5544	0.1175	1.0000	1.1175

Table 2.6. Out-of-sample statistical arbitrage test results for dual trees under 1bp market impact

Out-of-sample						
Strategy	$\mu (\cdot 10^{-4})$	$\sigma (\cdot 10^{-5})$	$\lambda (\cdot 10^{-1})$	H1	H2	H1+H2
1.	0.5616	1.3266	-1.3337	0.0225	0.0000	0.0225
2.	0.4357	2.1637	-2.3541	0.2206	0.0000	0.2206
3.	2.4700	5.9105	-2.6596	0.0388	0.0000	0.0388
4.	1.6000	3.0816	-4.1689	0.0639	0.0000	0.0639
5.	1.5022	2.9575	-2.9591	0.1703	0.0000	0.1703
6.	0.4801	2.7877	-2.4133	0.4009	0.0000	0.4009
7.	0.8946	1.5847	-0.5873	0.3227	0.0000	0.3227
8.	9.5990	10.3110	-10.3717	0.0000	0.0000	0.0000
9.	5.5076	6.5143	-9.0015	0.0098	0.0000	0.0098
10.	8.4098	14.2737	-14.8195	0.0007	0.0000	0.0007
Aggregate	0.5510	0.4109	-1.0776	0.0017	0.0000	0.0017

Out-of-sample, 1 st half						
Strategy	$\mu (\cdot 10^{-4})$	$\sigma (\cdot 10^{-5})$	$\lambda (\cdot 10^{-1})$	H1	H2	H1+H2
1.	0.8301	0.7644	-0.8357	0.0333	0.0000	0.0333
2.	1.1584	0.7101	-0.4012	0.1038	0.0000	0.1038
3.	3.7076	9.2456	-3.7302	0.0386	0.0000	0.0386
4.	2.9040	2.7992	-3.8099	0.0356	0.0000	0.0356
5.	4.8148	2.5767	-2.1679	0.0221	0.0000	0.0221
6.	0.2698	2.1389	-0.8768	0.4624	0.0000	0.4624
7.	0.4799	1.5110	-0.0939	0.4325	0.3113	0.7438
8.	10.9036	6.6737	-7.1572	0.0014	0.0000	0.0014
9.	7.3459	3.4537	-3.7681	0.0252	0.0000	0.0252
10.	9.8926	5.8844	-7.1353	0.0106	0.0000	0.0106
Aggregate	0.7882	0.2712	-0.6919	0.0040	0.0000	0.0040

Out-of-sample, 2 nd half						
Strategy	$\mu (\cdot 10^{-4})$	$\sigma (\cdot 10^{-5})$	$\lambda (\cdot 10^{-1})$	H1	H2	H1+H2
1.	0.3994	0.0765	0.6293	0.1310	1.0000	1.1310
2.	0.3352	0.0632	2.6392	0.3139	1.0000	1.3139
3.	0.8956	0.3228	3.7711	0.3125	1.0000	1.3125
4.	0.6556	0.5499	0.5441	0.3213	1.0000	1.3213
5.	-0.8630	0.1888	7.2044	0.6653	1.0000	1.6653
6.	0.7175	0.9314	2.2604	0.3914	1.0000	1.3914
7.	1.1606	0.5307	5.3785	0.3332	1.0000	1.3332
8.	7.7866	0.4188	7.6383	0.0032	1.0000	1.0032
9.	3.8859	0.2989	10.6625	0.0934	1.0000	1.0934
10.	7.6731	1.1616	1.8053	0.0108	1.0000	1.0108
Aggregate	0.3783	0.0280	0.8117	0.0590	1.0000	1.0590

Finding Relevant Variables in a Financial Distress Prediction Problem Using Genetic Programming and Self-organizing Maps

E. Alfaro-Cid¹, A.M. Mora², J.J. Merelo², A.I. Esparcia-Alcázar¹,
and K. Sharman¹

¹ Instituto Tecnológico de Informática, Universidad Politécnica de Valencia, Spain
{evalfaro, aesparcia, ken}@iti.upv.es

² Departamento de Arquitectura y Tecnología de Computadores, Universidad de Granada, Spain
{amorag, jmerelo}@geneura.ugr.es

Summary. This chapter illustrates the usefulness of Kohonen's self organizing maps (SOMs) and genetic programming (GP) for identifying relevant variables in a financial distress prediction problem. The approach presented here uses GP as a classification/prediction tool to produce models that can predict if a company is going to have book losses in the future. In addition, the analysis of the resulting GP trees provides information about the relevance of certain variables when solving the prediction model. This analysis in combination with the conclusions yielded using a SOM allowed us to significantly reduce the number of variables used to solve the book losses prediction problem while improving the error rates obtained.

3.1 Introduction

The prediction of financial distress is one of the best instruments for making decisions in an enterprise environment; this makes it a worthwhile research pursuit that does not lack difficulties. To begin with, it is a difficult problem to solve because there are lots of variables to take into account and, besides, there are many relationships (visible or hidden) among them which contradicts the requirements of usual statistical techniques. If we look at the history of this subject, it can be seen that the path of the studies moves from accurate but difficult to implement approaches like univariate statistical analysis [3, 5]; to more generic (less restrictive) tools, like the application of artificial neural networks (ANN) [6, 8, 9, 10, 13, 14, 21] and recently, genetic programming (GP) [7, 16, 17, 23].

The problem of using statistical techniques is the requirement of a functional relation between dependent and independent variables in order to get good results. In addition, these methods generalize badly because they are very sensitive to exceptions. On the other hand, soft-computing techniques are more flexible, as was illustrated in our initial papers on the topic [4, 9].

In this chapter we use GP to create a model from a data set of more than 400 companies that predicts book losses. The database used (which is the same that was previously employed for the prediction of bankruptcy [2, 1]) includes not only financial data from the companies but also general company information that can be relevant when

predicting failure. One of the factors that was found to be key for the successful application of GP to the bankruptcy prediction problem was the reduction of the number of variables. Usually bankruptcy prediction models are based on less than ten economic ratios [3, 28]. In our earlier work [2, 1], in order to handle the amount of data in the database and in order to ensure that the GP generated models which were understandable, the prediction was done in two steps. First of all, GP was run with all the available variables and then it was used to identify which data was relevant for solving the problem. This could be done since GP creates analytical models as a final result. Then, the prediction models were evolved using only those variables that had been identified as important in the first stage. The results in [2, 1] showed that reducing the number of variables not only simplifies the structures of the GP classifier but also improves the classification rates.

In this study a self-organizing map (SOM) [15] is applied to reduce the dimensionality of the book losses prediction problem. SOMs are able to highlight non-linear relationships among variables. This method is usually employed as a clustering/classification tool, or to find unknown relationships between data. For instance, in [19] a Kohonen's SOM was used for surveying the financial status of Spanish companies. From the map, the authors inferred which were the most relevant variables so that a fast diagnosis on the status of a company could be reached. Thus, we decided to consider some of the conclusions yielded in that work to choose a set of variables which may be significant to predict the book losses for a company. We demonstrate that the combination of the ability of SOMs for highlighting relevant data clusters and the key variables in each one, and the capability of GP for building understandable models leads to the generation of efficient and simple classifier structures for the prediction of book losses.

The prediction of book losses is interesting since there is a direct relationship between continued book losses and legal bankruptcy [22]. Moreover, book losses usually happen at a stage prior to insolvency so predicting book losses gives the management of a company more time to react and find a solution to the problem (if there is one).

The remainder of this chapter is organized as follows. Sect. 3.2 describes the dataset used to make the study; the methods used to process the samples are introduced in Sect. 3.3. Sect. 3.4 shows the results yielded by these methods, and the related conclusions are reported in Sect. 3.5.

3.2 Dataset and Problem Description

The data used in this work was extracted from the Infotel database (bought from <http://infotel.es>). The dataset is composed of data from about 470 companies. 170 of these companies had continuous book losses during the years 2001-2003, and the remaining 300 companies presented good financial health. Data of these companies from years 1998, 1999 and 2000 was used to for training and testing.

Table 3.1 shows the independent variables, their description and type. As can be seen, the variables can take values from different numerical ranges: real, integer and binary. Also, some of the non-financial data take categorical values; these are the size of the company, the type of company and the auditor's opinion. Usually, company size is a

Table 3.1. Independent Variables

Financial Variables	Description	Type
Debt Structure	Long-Term Liabilities /Current Liabilities	Real
Debt Cost	Interest Cost/Total Liabilities	Real
Leverage	Liabilities/Equity	Real
Cash Ratio	Cash Equivalent /Current Liabilities	Real
Working Capital	Working Capital/ Total Assets	Real
Debt Ratio	Total Assets/Total Liabilities	Real
Operating Income Margin	Operating Income/Net Sales	Real
Debt Paying Ability	Operating Cash Flow/Total Liabilities	Real
Return on Operating Assets	Operating Income/Average Operating Assets	Real
Return on Equity	Net Income/Average Total Equity	Real
Return on Assets	Net Income/Average Total Assets	Real
Asset Turnover	Net Sales/Average Total Assets	Real
Receivable Turnover	Net Sales/Average Receivables	Real
Stock Turnover	Cost of Sales/Average Inventory	Real
Current Ratio	Current Assets/Current Liabilities	Real
Acid Test	(Cash Equivalent + Marketable Securities + Net receivables) /Current Liabilities	Real
Non-financial Variables	Description	Type
Size	Small/Medium/Large	Categorical
Type of company		Categorical
Auditor's opinion		Categorical
Audited	If the company has been audited	Binary
Delay	If the company has submitted its annual accounts on time	Binary
Linked to a group	If the company is part of a group holding	Binary
Number of partners		Integer
Number of employees		Integer
Age of the company		Integer
Number of changes of location		Integer
Number of incidences in court	Number of times the company was sued last year	Integer
Historic number of incidences in court	Since the company was created	Integer
Historic number of serious incidences	Such as strikes, accidents...	Integer
Amount of money spent on incidences in court	Last year	Real
Historic amount of money spent on incidences in court	Since the company was created	Real

real variable but in this case the companies are grouped in three separated categories according to their turnover. Each categorical variable can take 3 different values. To work with them they have been transformed into 3 binary variables each. For example, if we have the variable *size*, with 3 possible values, '1', '2' or '3', we can create three new variables *size1*, *size2* and *size3*. Each one will have a value of '1' if the old value of *size*, was '1', '2' or '3', respectively, and a value of '0' otherwise. Therefore, after this modification the available data set for each company includes 37 independent variables: 18 real, 7 integer and 12 binary variables. To perform the experiments the data was divided into training and testing sets. The training set comprised 70% of the available data. The remaining 30% was used for testing the evolved solutions.

3.3 Methodology

In this section we briefly describe the SOM algorithm and the GP framework that we have used to predict book losses.

3.3.1 Self-organizing Map

The Self-Organizing Map (SOM) was introduced by Teuvo Kohonen in 1982 (see [15] for details and [27] for a recent review). It is a non-supervised neural network that tries to imitate the self-organization done in the sensory cortex of the human brain, where neighbouring neurons are activated by similar stimulus. It is usually employed either as a clustering/classification tool or as a method to find unknown relationships among a set of variables that describe a problem. The main property of the SOM is that it makes a nonlinear projection from a high-dimensional data space (one dimension per variable) to a regular, low-dimensional (usually 2D) grid of neurons (see Fig. 3.1), and, from the self-organization process, the projection preserves the topologic relations while simultaneously creating a dimensional reduction of the representation space (the transformation is made in a topologically ordered way, though this is not guaranteed in advance and must be assessed during the training process).

The SOM processes a set of input vectors (samples or patterns), which are composed by variables (features) typifying each sample. It then creates an output topological network where each neuron is also associated to a vector of variables (model vector) which

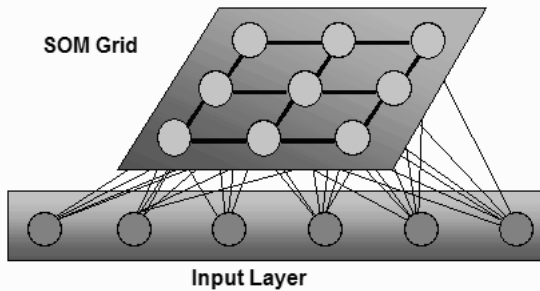


Fig. 3.1. SOM Grid structure. There is an input layer (to which the vectors in the training set are submitted) and a process layer (where the neurons of the network are) which takes a grid shape.

is representative of a group of the input vectors. Note in Fig. 3.1 that each neuron of the network is completely connected to all the nodes of the input layer. So, the network represents a feed-forward structure with only one computational layer. There are four main steps in the processing of the SOM. Apart from the first one, the others are repeated until a stop criterion is met.

1. **Initialization of model vectors:** It is usually made by assigning small random values to their variables, but there are some other possibilities such as an initialization using random input samples.
2. **Competitive process:** For each input pattern X , all the neurons (model vectors) V “compete” using a *similarity function* in order to identify the one most similar or closest to the sample vector. Usually, the similarity function is a distance measure (such as an Euclidean distance). The winner neuron is called the best matching unit (BMU).
3. **Cooperative process:** The BMU determines the centre of a topological neighbourhood where those neurons inside it (the model vectors) will be updated to be even more similar to the input pattern. A *neighbourhood function* is used to determine the neurons to consider. If the lattice where the neurons are is rectangular or hexagonal, it is possible to consider as neighbourhood functions rectangles or hexagons with the BMU as centre. However, it is more usual to use a Gaussian function to assure that the farther the neighbour neuron is, the smaller the updating to its associated vector is. In this process, all the neurons within a neighborhood cooperate to learn.
4. **Learning process:** In this step the variables of the model vectors within the neighbourhood are updated to be closer to those of the input vector. It means making the neuron more similar to the sample. The *learning rule* used to update the vector (V) for every neuron i in the neighbourhood of the BMU is

$$V_i^t = V_i^{t-1} + \alpha^t \cdot N_{BMU}^t(i) \cdot (X - V_i^{t-1}) \quad (3.1)$$

Where t is the current iteration of the algorithm, X is the input vector, N_{BMU} is the neighbourhood function for the BMU, which returns a high value (in $[0,1]$) if the neuron i is in the neighbourhood and close to the BMU (1 if $i = BMU$), and a small value otherwise (0 if i is not located inside the neighbourhood); and α is the *learning rate* (in $(0,1]$). Both neighbourhood and learning rate depend on t , since it is usual to decrease the radius of the first one and the value of the second in order to impose a higher updating rate at the beginning of the process and almost none in the final iterations.

The recurrent application of Eq. 3.1, and the update of the neighbourhood function, has the effect of ‘moving’ the model vectors, V_j from the winning neuron towards the input vector X_i . That is, the model vectors tend to follow the distribution of the input vectors. Consequently, the algorithm leads to a topological arrangement of the characteristic map of the input space, in the sense that adjacent neurons in the network tend to have similar weight vectors.

The SOM is further processed using Ultsch’s method [25], the Unified distance matrix (U-Matrix). It uses SOM’s code vectors (vectors of variables of the problem) as

a data source and generates a matrix where each component is a distance measure between two adjacent neurons. This matrix is represented as a lattice with one cell per neuron in the map, and one additional node is placed between every pair of neurons, which represents the distance between them. This distance is showed using a colour code (usually blue means near or little distance, and red far or large distance). Each cell corresponding to one neuron in the map may be labeled with the related tag of the pattern which it represents. It allows us to visualize any multi-variated dataset in a two-dimensional display, so we can detect topological relations among neurons and infer about the input data structure. High values in the U-matrix represent a frontier region between clusters, and low values represent a high degree of similarities among neurons in that region, i.e. clusters.

Therefore, looking at the output of a SOM and its corresponding U-Matrix, it is possible to recognize some clusters as well as the metric-topological relations of the data items (vectors of variables of the problem) and the outstanding variables.

Although Kohonen's SOMs are not as accurate as other tools at the task of classification, they can be applied to many different types of data, yielding a visualization of natural structures in the data and their relations, as well as the natural groupings that could be among them. In addition, SOMs facilitate the identification of the variables that have more influence on these groupings, via the so-called planes analysis. Other statistical and soft computing tools can also be used for this purpose, but Kohonen's SOMs offer a visual way of doing it which is much more intuitive.

SOMs have been successfully applied to financial problems. A standard reference in the field would be the paper by Serrano-Cinca [24]. In this work the author develops a complete decision support system for financial diagnosis based on SOMs.

3.3.2 Genetic Programming

Genetic Programming (GP) is based on the idea that in nature structure undergoes adaptation. The structure created over a period of time is the outcome of natural selection and sexual reproduction. Thus, GP is a structural optimisation technique (as opposed to a parametric optimisation technique). The individuals in GP are represented as hierarchical structures (typically tree structures) and the size and shape of the solutions are not defined a priori as in other methods from the field of evolutionary computation, but they evolve along the generations. The flow of a GP algorithm is the same as any other evolutionary technique: a population is created at random, each individual in the population is evaluated using a fitness function, the individuals that performed better in the evaluation process have a higher probability of being selected as parents for the new population than the rest and a new population is created once the individuals are subject to the genetic operators of crossover and mutation with a certain probability. The loop is run until a certain termination criterion is met.

In this section we briefly describe the GP framework that we have used for representing systems for financial distress prediction. Basically, the GP algorithm must find a structure (a function) which can, once supplied with the relevant data from the company, decide if this company is going to have book losses or not. In short, it is a binary classification problem.

Classification

The classification works as follows. Let $X = \{x_0, \dots, x_N\}$ be the vector comprising the data of the company undergoing classification. Let $f(X)$ be the function defined by a GP tree structure. We can apply X as the input to the GP tree and calculate the output $f(X)$. Once the numerical value of $f(X)$ has been calculated, it will give us the classification result according to

$$f(X) > 0, X \in L \quad (3.2)$$

$$f(X) \leq 0, X \in \bar{L} \quad (3.3)$$

where L represents the class to which companies with book losses belong and \bar{L} represents the class to which healthy companies belong. The task of GP is to find the function $f(X)$.

Fitness Evaluation

Since the database is unbalanced in the sense that only 170 companies have book losses versus 300 healthy companies, we have modified the cost associated to misclassifying the positive and the negative classes to compensate for the imbalanced ratio of the two classes as proposed in [12]. For example, if the imbalance ratio is 1:10 in favor of the negative class, the penalty for misclassifying a positive example should be 10 times greater. Basically, it rewards the correct classification of examples from the small class over the correct classification of examples from the over-sized class. It is a simple but efficient solution. Therefore, the fitness function to be maximized is

$$Fitness = \sum_{i=1}^n u_i \quad (3.4)$$

where

$$u_i = \begin{cases} 0 & : \text{incorrect classification} \\ \frac{n_h}{n_l} & : \text{company with losses classified correctly} \\ 1 & : \text{healthy company classified correctly} \end{cases}$$

n_l is the number of companies with book losses and n_h is the number of healthy companies.

GP Implementation

The GP implementation used is based on ECJ (<http://cs.gmu.edu/~eclab/projects/ecj>), a research evolutionary computation system developed at George Mason University's Evolutionary Computation Laboratory (ECLab). Table 3.2 shows the main parameters used during evolution.

The mutation, crossover and bloat control operator rates were chosen according to the results obtained in a comparison study of bloat control methods in [2]. Several combinations of the bloat control operator rate and the crossover rate were tested and in this present work the best combination was used. The population size and the number of generations were chosen by trial and error. Elitism was included because its inclusion improved the results. All the other parameter settings are the default in ECJ.

Table 3.2. GP parameters

Initialization method	Ramped half and half
Replacement operator	Generational with elitism (0.2%)
Selection operator	Tournament selection
Tournament group size	7
Cloning rate	0.1
Crossover operator	Bias tree crossover
Internal node selection rate	0.9
Crossover rate	0.4
Mutation rate	0.1
Bloat control operator rate	0.4
Tree maximum initial depth	7
Tree maximum depth	18
Population size	1000
Termination criterion	250 generations

As a method of bloat control we have used a genetic operator. This operator implements a bloat control approach described in [11] and inspired in the “prune and plant” strategy used in agriculture. It is used mainly for fruit trees and it consists of pruning some branches of trees and planting them in order to grow new trees. The idea is that one of the branches of the selected tree will be “pruned” and substituted by a terminal. The pruned branch will be then “planted” as a new tree in the population. This way both offspring trees will be of smaller size than the ancestor, effectively reducing bloat.

Strong Typing

Strongly typed GP (STGP) [18] is an enhanced version of GP that enforces data-type constraints, since standard GP is not designed to handle a mixture of data types. In STGP, each function node has a return-type, and each of its arguments also have assigned types. STGP permits crossover and mutation of trees only with the constraint that the return type of each node matches the corresponding argument type in the node’s parent.

A STGP has been implemented in order to ensure that in the resulting classifying models the functions operate on appropriate data types so that the final model has a physical meaning. That is, the objective is to avoid results that operate on data which

Table 3.3. Function set

Functions	Number of arguments	Arguments type	Return type
$+$, $-$, $*$, $/$	2	real	real
If $arg_1 \leq arg_2$ then arg_3 else arg_4	4	real	real
If arg_1 then arg_2 else arg_3	3	arg_1 is a boolean arg_2, arg_3 are real	real
If $arg_1 \leq int$ then arg_2 else arg_3 (int is randomly chosen)	3	arg_1 is an integer arg_2, arg_3 are real	real

are not compatible, for instance, models which add up the liabilities and the age of a company. The terminal set used consists of 38 terminals: the independent variables from Table 3.1 plus Koza's ephemeral random constant. Table 3.3 shows the function set used and the chosen typing.

3.4 Experiments and Results

In this section we initially present the results obtained using the SOM in a previous paper [19]. After that, the experiments performed for GP (considering the conclusions reached with SOM) and the results of these experiments are shown.

3.4.1 SOM Experiments

In [19], the SOM was applied to the same dataset as in this work, and the same preprocessing was performed in order to 'transform' categorical variables into numerical ones. After this step, a normalization process was carried out. It is a necessary pre-processing in order to prevent the differences in the values of some variables from dominating the map organization. Values of '-1' and '1' were assigned to those independent variables which took values of '0' and '1' (old categorical ones). The rest of independent variables were also normalized by using the method of variance, so all their values were located in the range $[-1, 1]$ except those which were outstanding. For these experiments, Matlab 6.5 along with SOM Toolbox [26], Version 2.0 beta were used. The parameters used for training are shown in Table 3.4.

The SOM was trained using all the sample data, corresponding to the accounting periods 1998-2000. After training and post-processing, the U-Matrix graph [25] was obtained. The samples were labeled in that graph with a 'P' when they corresponded to a firm with continued losses, otherwise the unit was labeled with a 'n'. The variables are named with capitals excepting the new non-categorical ones obtained by transforming their categorical counterparts.

In this case, the SOM was not trained with the ideal amount of data in order to differentiate companies of both classes (maybe there were not enough data or maybe the samples used were similar), therefore there was not a clear boundary between clusters of failed companies (which have had continued losses) and those of successful companies. Looking at the U-Matrix graph (see Fig. 3.2), only 3 clusters stand out: a general cluster, formed by dark gray color (blue in color images), that is associated to vectors representing companies close to each other; a *warm spot* at the lower right corner, and a *hot spot* at the upper right corner; these spots represent clusters of companies whose

Table 3.4. SOM Parameters

<i>Map Grid Size</i>	8x8
<i>Map Lattice</i>	hexagonal
<i>Size of Vectors</i>	37
<i>Normalization</i>	Var (in $[-1, 1]$)
The rest of parameters take the default values in SOM Toolbox.	

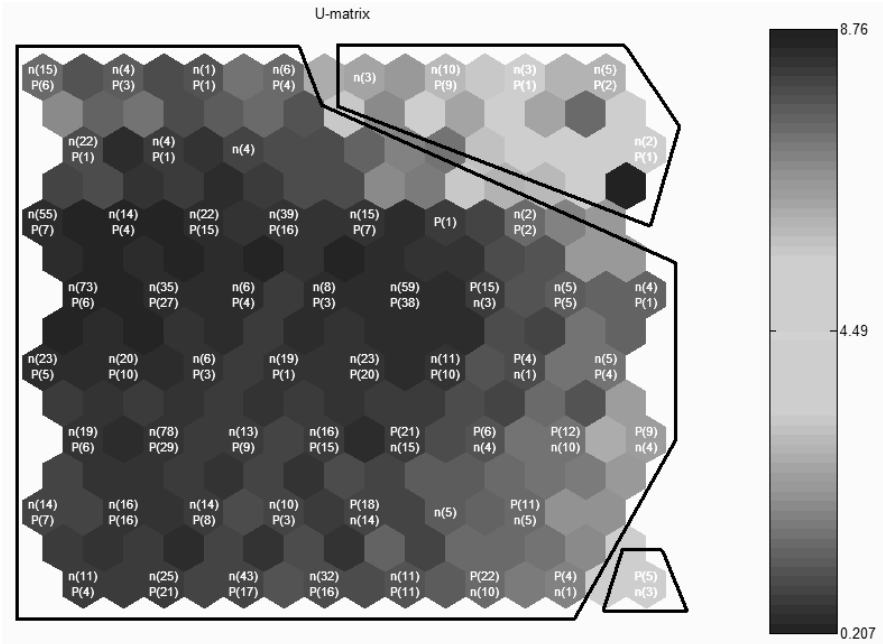


Fig. 3.2. U-Matrix graph (1998-2000 data). There are three main clusters: a warm spot at the lower right corner, a hot spot at the upper right side and a general and central one (For a color and better-quality image you can visit: http://geneura.ugr.es/~amorag/som/finan/UMatrix_98_00.bmp).

features stand out against the rest. There is also a *small cluster* in the 3rd hexagon from the right in the top row.

Every subplot in Fig. 3.3 represents the values of one particular variable in every vector (or *neuron*) that constitutes the trained self-organizing map. The color code used is as follows: bluish tones represent that the variable takes small values and reddish tones represent that the variable takes large values within the range of that variable. The most representative variables (or key variables) of a cluster can be identified by visual inspection of the figure. They are those which take a distinctive value (usually very small or very large) in the area of the map where the cluster has been discovered. So, looking at Fig. 3.3, the key variables identified in each cluster are: those which take completely different (usually very high or very low) values from the rest of the variables in the zones of the map corresponding to each one of the clusters. So, in summary we have

- *Warm spot*: large companies (*size3 (tamano3)* set to 1), which have been audited (5th component, *AUDITADA*), with *type of company1* set to 1 (*cod_formasoc1*, 6th component), with a favorable auditor's opinion (5th component starting by the last one, *opi_auditorial1*), high acid test (*TEST_ACI*), high current ratio (*SOLVEN-CIA*), delay in reporting the annual accounts (*RETRASO*), high leverage (*NIV_END*) and belonging to a group (last component, *VINC_GRUPO*). There are more failed

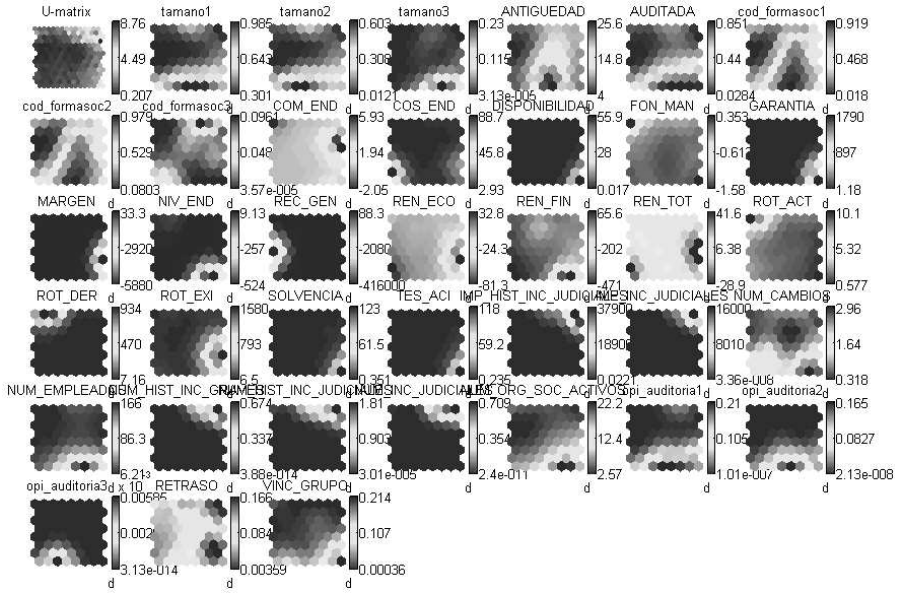


Fig. 3.3. SOM Plane Analysis (1998-2000 data) (For a color and better-quality image you can visit: http://geneura.ugr.es/~amorag/som/finan/SOM_98_00.bmp)

companies in this cluster than successful ones, so the cluster probably corresponds to old members of company conglomerates, big-sized, which are conveniently closed without incurring big losses.

- *Hot spot*: successful companies, with small size (*size1 (tamano1)* set to 1) and most economic indicators in a healthy shape.
- *Small spot*: most companies with losses with *type of company3 (cod_formasoc3)* set to 1 and no other distinctive value.

3.4.2 GP Experiments

In these experiments, the conclusions yielded by the previously commented work are considered. But firstly we present the results obtained using GP without considering SOM results. The mean and standard deviation for 30 runs was calculated. Table 3.5 shows the results obtained using all variables available for prediction. Note that the reported error figures are calculated as the percentage of misclassifications. The fitness function in Eq. 3.4 is only used during the optimisation process to deal with the unbalanced ratio of the two classes.

The results show the difficulty of predicting book losses, since the same GP strategy obtained better classification rates when using the database for predicting bankruptcy [2, 1]. Book losses happen a time prior to bankruptcy and do not lead always to it. For instance, if a company is linked in a group, it may suffer book losses for a long time without being forced to close down. Therefore, predicting book losses is a more difficult problem.

Table 3.5. Average results for the prediction of book losses using all variables

Training error	Testing error
22.58 ± 2.18	34.32 ± 2.11

Table 3.6. Percentage of final trees that use each variable in the first set of GP results

Variable	% Trees	Variable	% Trees
Debt Structure (DS)	60.00	<i>size1</i> (s1)	13.33
Debt Cost (DC)	83.33	<i>size2</i> (s2)	23.33
Leverage (L)	96.67	<i>size3</i> (s3)	16.67
Cash Ratio (CR)	60.00	<i>type of company1</i> (tc1)	30.00
Working Capital (WC)	56.67	<i>type of company2</i> (tc2)	20.00
Debt Ratio (DR)	70.00	<i>type of company3</i> (tc3)	10.00
Operating Income Margin (OIM)	73.33	<i>auditor's opinion 1</i> (ao1)	0.00
Debt Paying Ability (DPA)	60.00	<i>auditor's opinion 2</i> (ao2)	13.33
Return on Operating Assets (ROA)	73.33	<i>auditor's opinion 3</i> (ao3)	20.00
Return on Equity (RE)	86.69	Audited (A)	26.67
Return on Assets (RA)	70.00	Delay (D)	20.00
Asset Turnover (AT)	66.67	Linked in a group (LG)	6.67
Receivable Turnover (RT)	60.00	Number of partners (NoP)	23.33
Stock Turnover (ST)	80.00	Number of employees (NoE)	23.33
Current Ratio (CuR)	66.67	Age of the company (Age)	16.67
Acid Test (AcT)	53.33	No. changes of location (NoL)	13.33
Number of judicial incidences (NJI)			13.33
Historic number of judicial incidences (HNJI)			26.67
Historic number of serious incidences (HNSI)			30.00
Amount of money spent on judicial incidences (MJl)			46.67
Historic amount of money spent on judicial incidences (HMIJ)			56.67

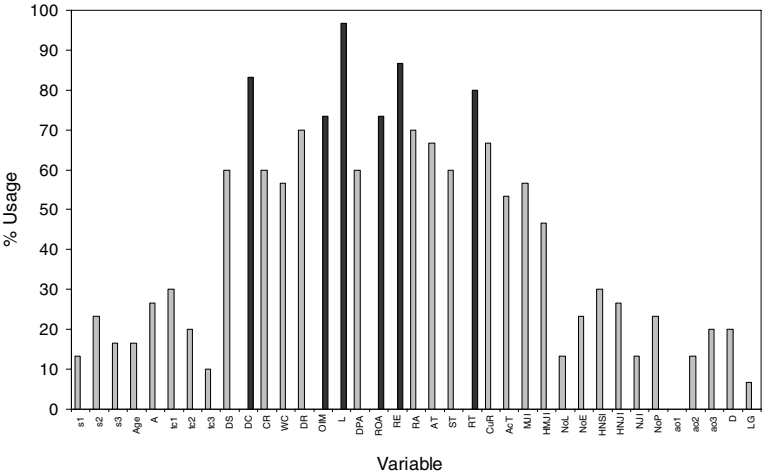


Fig. 3.4. Graphical representation of the percentage of final trees that use each variable in the first set of GP results. The six variables used more frequently are marked in black.

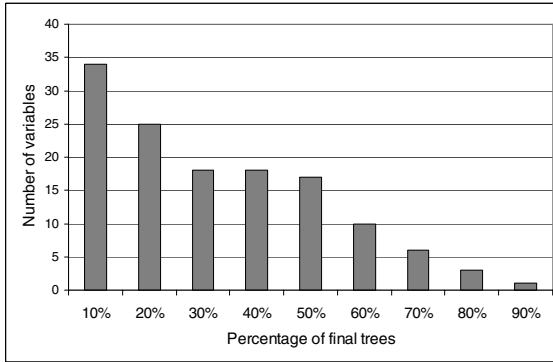


Fig. 3.5. Cumulative number of variables used in the final trees

Table 3.7. Variables used most frequently in the GP trees

Variable	Type	Variable	Type
Debt Cost	Real	Return on Operating Assets	Real
Operating Income Margin	Real	Return on Equity	Real
Leverage	Real	Stock Turnover	Real

Table 3.8. Average results for the prediction of book losses using the variables used most frequently by GP

Training error	Testing error
23.61 ± 1.53	35.85 ± 1.21

The success of our proposal is based on what follows. As a first approach we used GP to reduce the number of independent variables, since this method yielded good results in previous work [2, 1]. We analyzed the final 30 tree structures the GP algorithm converged to in the previous experimental runs (we run the optimisation 30 times and we kept the best tree in the final generation of each run). See Table 3.6 and Fig. 3.4 for the percentage of trees in this group that used each variable. In addition, Fig. 3.5 shows the cumulative number of variables that appear in more than a given percentage of final trees. For instance, there are 18 variables that are used in more than 30% of the final trees, that is, each of these variables is used in, at least, more than 9 trees out of the final 30 trees.

From these results we observed that there are 6 variables that were used more frequently than others (in more than 70% of the final trees). We then ran a second set of experiments considering only these variables. In Table 3.7 the variables used in the second set of experiments are presented.

Table 3.9. Variables identified as relevant by the SOM

Variable	Type	Variable	Type
<i>size1</i>	Binary	Audited	Binary
<i>size3</i>	Binary	Delay	Binary
<i>type of company1</i>	Binary	Leverage	Real
<i>type of company3</i>	Binary	Acid Test	Real
<i>auditor opinion1</i>	Binary	Current Ratio	Real
Linked in a group	Binary		

Table 3.10. Average results for the prediction of book losses using the variables identified as relevant by the SOM

Training error	Testing error
21.94 ± 1.39	32.79 ± 1.33

Table 3.11. Average results for the prediction of book losses obtained in each set of experiments

	Training error	Testing error
All variables	22.58 ± 2.18	34.32 ± 2.11
GP subset	23.61 ± 1.53	35.85 ± 1.21
SOM+GP subset	21.94 ± 1.39	32.79 ± 1.33

Table 3.8 shows the numerical results obtained when solving the prediction problem using the reduced set of variables. This method for reducing the number of variables yielded good results for the bankruptcy prediction problem [2, 1] but it has not worked well in this case. Both the average training and testing errors have increased.

Given that the results obtained using GP for reducing the variables are not good, we have used a SOM to fulfill the task. In the previous subsection the identified clusters and key variables have been shown. These variables are summarized in Table 3.9.

Thus, a third set of experiments was run where the book losses prediction problem was solved using the variables identified by the SOM as relevant plus two other variables that the GP algorithm used in more than 80% of the final trees: *return on equity* and *debt cost*.

Table 3.10 shows the error rates obtained in this third set of experiments, which are smaller than when using all the variables, and Table 3.11 summarizes the error rates obtained in each set of runs for comparison purposes. Kruskal-Wallis tests were used to compare the results. When comparing the latter results with those obtained using all variables, no significant differences were found between training error rates. However, differences in the testing error rate were significant to level 95%. If the comparison is made between the results obtained using the subset of variables identified by the SOM

and those obtained using the subset of variables identified by GP, the differences are significant for both training and testing test with confidence levels of 99%.

Fig. 3.6 presents an example of a GP tree obtained for the classification [20]. This particular tree was chosen because it is an ‘average’ tree with a classification error very close to the mean. In the figure, x_5 represents the binary variable *type of company1*, x_9 is the debt cost, x_{14} is the leverage, x_{17} is the return on equity, x_{22} is the current ratio, x_{23} is the acid test, x_{32} represents the binary variable *auditor opinion1* and x_{36} represents the binary variable *linked to a group*. If $y > 0$ the company is classified as in financial distress, otherwise the company is considered healthy. This tree achieves a classification error of 32.75%.

We are going to analyze what happens in the resulting tree when a company is not attached to a group (i.e. $x_{36} = 0$). This assumption simplifies considerably the analysis and more of 97% of the companies in the database satisfy this condition; however, it is interesting to note that if the value of that variable is true, the company would be in the *warm spot* we mentioned before, which would mean it is very likely a company that will be in financial distress. Under this assumption we can express y_0 (see right hand side of Fig. 3.6) as two nested conditional clauses

$$\begin{aligned} y_0 &= \text{if } x_5 = 1 \\ &\quad \text{then } x_{14}x_{22} \\ &\quad \text{else if } x_{17} \leq x_{23} \\ &\quad \quad \text{then } x_{23}^2 \\ &\quad \quad \text{else } -20.6x_{23} \end{aligned} \quad (3.5)$$

And the overall output would be

$$y = x_{14} - \text{if } y_0 \leq x_{17} \text{ then } x_{14} \text{ else } x_{17} \quad (3.6)$$

Therefore,

$$\begin{aligned} &\text{if } y_0 \leq x_{17} \\ &\quad \text{then } y = x_{14} - x_{14} = 0 \Rightarrow \text{healthy company} \\ &\quad \text{else } y = x_{14} - x_{17} = \begin{cases} > 0 \Rightarrow \text{company with losses} \\ \leq 0 \Rightarrow \text{healthy company} \end{cases} \end{aligned} \quad (3.7)$$

Basically it predicts that if $y_0 \leq x_{17}$ the company is healthy. Otherwise, that is, assuming $y_0 > x_{17}$, the company will have losses if the leverage is greater than the return on equity, while if the return on equity (ROE) can compensate the leverage the company will be healthy. Given that leverage = $\frac{\text{liabilities}}{\text{equity}}$ and ROE = $\frac{\text{net income}}{\text{equity}}$, the previous statement could be rewritten as: assuming $y_0 > x_{17}$, the company will have losses if the liabilities are greater than the net income, while if the net income can compensate the liabilities the company will be healthy. Let us analyze the inequality $y_0 \leq x_{17}$. The variable y_0 can take 3 different values

$$y_0 = \begin{cases} x_{14}x_{22} & : \text{if } x_5 = 1 \\ x_{23}^2 & : \text{if } x_5 = 0 \text{ and } x_{17} \leq x_{23} \\ -20.6x_{23} & : \text{if } x_5 = 0 \text{ and } x_{17} > x_{23} \end{cases} \quad (3.8)$$

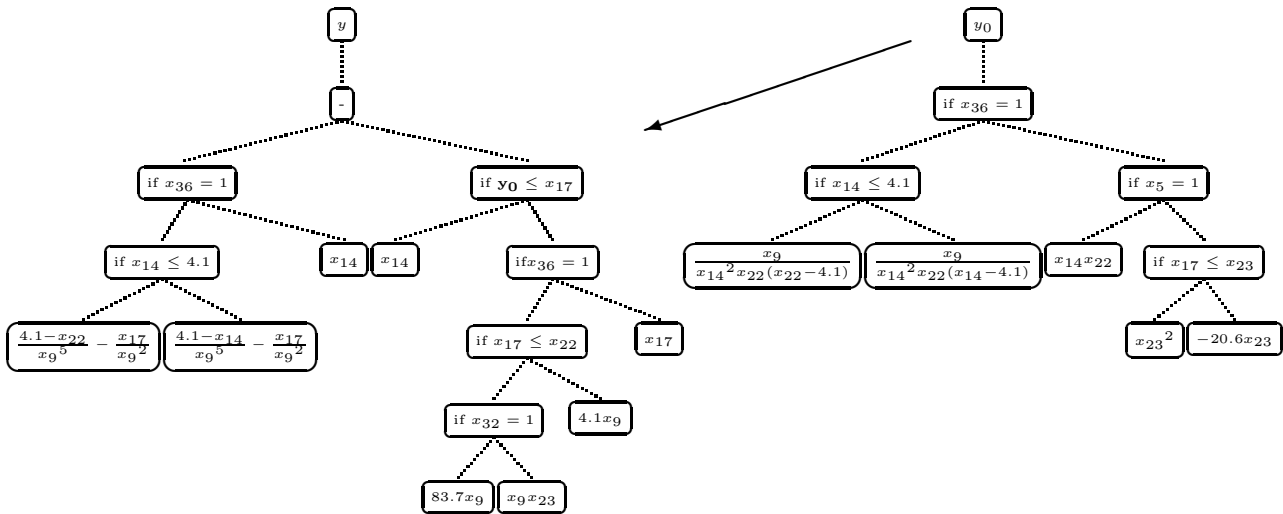


Fig. 3.6. Example of resulting GP tree. The y_0 symbol in the left hand side tree (marked in boldface) has been expanded on the right hand side.

If $x_5 = 0$ and $x_{17} \leq x_{23}$ then y_0 takes the value x_{23}^2 which yields $y_0 \leq x_{17}$ being false. That is, for a company type 2 or 3, the prediction is that if the acid test value is greater than the return on equity, the company will have losses if the liabilities are greater than the net income, otherwise the company will be healthy.

On the other hand if $x_5 = 0$ and $x_{17} > x_{23}$ then y_0 takes the value $-20.6x_{23}$ which yields $y_0 \leq x_{17}$ being true if $x_{23} > 0$ (which happens in around 98% of the companies in the database). That means that for a company of type 2 or 3 if the acid test value is positive and smaller than the return on equity the company is healthy.

The last possible case is that the company is of type 1. Then y_0 takes the value $x_{14}x_{22}$. Again, leverage = $\frac{\text{liabilities}}{\text{equity}}$ and ROE = $\frac{\text{net income}}{\text{equity}}$, so the condition could be expressed as: if the product of the liabilities and the current ratio is smaller than the net income the company will be healthy otherwise the prediction depends on whether the liabilities are greater than the net income or not. Regarding the condition that the multiplication of the liabilities and the current ratio must be smaller than the net income for a type 1 company to be healthy, it penalizes high current ratios that may indicate that the firm has too many assets tied up in current assets and is not making efficient use of them.

Thus, in general the prediction of financial distress for companies not linked to a group in this GP tree is based on the comparison of two values: liabilities and net income. If the liabilities are greater than the net income the company will suffer losses, otherwise the company will not. If the companies in the testing set are classified according to this rule the classification error is 34.75%. In addition the GP has detected two groups that do not follow the rule, improving the classification error to 32.75%. Therefore, if a company is of type 1 and the multiplication of the liabilities and the current ratio is smaller than the net income or if a company is of type 2 or 3 and its acid test value is positive and smaller than the return on equity, the company is predicted as healthy.

3.5 Conclusions

In this study we present an improved method for prediction of financial distress using GP and SOMs. It is based on work carried out in [2, 1], where the importance of variable selection when applying GP to the financial failure problem was emphasized. We have merged our results with those obtained by [19] employing Kohonen's Self-Organizing Maps on the same database. As a result, we have got a set of variables that are significant for the book losses prediction problem. Considering this set of variables and the previously tested GP approach, the classification (prediction) rates have improved.

So, reducing the dimensionality of the prediction of financial distress problem can not only simplify the understanding of the resulting classifiers, but also improve the prediction error rates. On the other hand, the analysis of the resulting GP trees, which in previous work provided enough information for reducing the number of variables, has not yielded satisfactory results in this case, given that the error rates increased. However, when the variables used more frequently in the resulting GP trees have been combined with those variables identified by the SOM as more relevant, the error rates have decreased. Thus, the application of SOM to the analysis of variables has allowed

an improvement in the prediction rates and a reduction of the number of variables from 37 to 13.

As future work we plan to use the same technique for other financial prediction problems such as bankruptcy prediction. We also plan to compare the results obtained with other methods used frequently in classification/prediction such as logistic regression or support vector machines. We want also test our method with a larger financial database, so that it includes more companies and further information from each company.

Acknowledgements

This work has been supported by project TIN2007-68083-C02 (Spanish Ministry of Education and Culture, NOHNES Project).

References

- [1] Alfaro-Cid, E., Sharman, K., Esparcia-Alcázar, A.: A genetic programming approach for bankruptcy prediction using a highly unbalanced database. In: Giacobini, M. (ed.) *EvoWorkshops 2007*. LNCS, vol. 4448, pp. 169–178. Springer, Heidelberg (2007)
- [2] Alfaro-Cid, E., Cuesta-Cañada, A., Sharman, K., Esparcia-Alcázar, A.I.: Strong Typing, Variable Reduction and Bloat Control for Solving the Bankruptcy Prediction Problem Using Genetic Programming. In: *Natural Computing in Computational Economics and Finance*. Studies in Computational Intelligence Series, vol. 100, pp. 161–186. Springer, Heidelberg (2008)
- [3] Altman, E.: The success of business failure prediction models. An international survey. *Journal of Banking, Accounting and Finance* 8, 171–198 (1984)
- [4] Vieira, A., Castillo, P.A., Merelo, J.J.: Application of HLVQ and G-Prop neural networks to the problem of bankruptcy prediction, pp. 655–662 (2003), <http://www.springerlink.com/link.asp?id=2gqqar9cv3et5nlg>
- [5] Beaver, W.: Financial ratios as predictors of failures. *Empirical research in accounting: Selected studies*. *Journal of Accounting Research Supplement* 5, 71–111 (1966)
- [6] Bell, T., Ribar, G., Verchio, J.: Neural nets versus logistic regression: A comparison of each model's ability to predict commercial bank failures. In: *Proceedings of the 1990 Deloitte and Touche/University of Kansas Symposium on Auditing Problems*, pp. 29–53 (1990)
- [7] Brabazon, A., O'Neill, M.: Biologically inspired algorithms for financial modelling. *Natural Computing Series*. Springer, Berlin (2006)
- [8] Martín-del Brío, B., Serrano-Cinca, C.: Self-organizing neural networks for the analysis and representation of data: Some financial cases. *Neural Computing & Applications* 1(3), 193–206 (1993)
- [9] Castillo, P.A., de la Torre, J.M., Merelo, J.J., Román, I.: Forecasting business failure. A comparison of neural networks and logistic regression for Spanish companies. In: *Proceedings of the 24th European Accounting Association, Athens, Greece* (2001)
- [10] Charalambous, C., Charitou, A., Kaourou, F.: Application of feature extractive algorithm to bankruptcy prediction neural networks. In: *Proceedings of the IEEE-INNS-ENNS International Joint Conference (IJCNN 2000)*, vol. 5, pp. 303–308 (2000)
- [11] Fernández de Vega, F., Rubio del Solar, M., Fernández Martínez, A.: Implementación de algoritmos evolutivos para un entorno de distribución epidémica. In: Arenas, M.G., et al. (eds.) *Actas del IV Congreso Español de Metaheurísticas, Algoritmos Evolutivos y Bioinspirados (MAEB 2005)*, Granada, Spain, pp. 57–62 (2005)

- [12] Japkowicz, N., Stephen, S.: The class imbalance problem: a systematic study. *Intelligent Data Analysis* 6(5), 429–449 (2002)
- [13] Kaski, S., Sinkkonen, J., Peltonen, J.: Bankruptcy analysis with self-organizing maps in learning metrics. *IEEE Transactions on Neural Networks* 12(4), 936 (2001)
- [14] Kiviluoto, K.: Predicting bankruptcies with the self-organizing map. *Neurocomputing* 21(1–3), 191–201 (1998)
- [15] Kohonen, T.: *The Self-Organizing Maps*. Springer, Heidelberg (2001)
- [16] Lee, W.C.: Genetic programming decision tree for bankruptcy prediction. In: *Proceedings of the 2006 Joint Conference on Information Sciences (JCIS 2006)*, Atlantis Press, Kaohsiung, Taiwan (2006)
- [17] Lensberg, T., Eilifsen, A., McKee, T.E.: Bankruptcy theory development and classification via genetic programming. *European Journal of Operational Research* 169, 677–697 (2006)
- [18] Montana, D.J.: Strongly typed genetic programming. *Evolutionary Computation* 3(2), 199–230 (1995)
- [19] Mora, A.M., Laredo, J.L.J., Castillo, P.A., Merelo, J.J.: Predicting financial distress: A case study using self-organizing maps. In: Sandoval, F., Prieto, A.G., Cabestany, J., Graña, M. (eds.) *IWANN 2007*. LNCS, vol. 4507, pp. 774–781. Springer, Heidelberg (2007)
- [20] Mora, A.M., Alfaro-Cid, E., Castillo, P.A., Merelo, J., Esparcia-Alcázar, A.I., Sharman, K.: Discovering causes of financial distress by combining evolutionary algorithms and artificial neural networks. In: *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO 2008)*, Atlanta, USA (accepted for publication, 2008)
- [21] Rahimian, E., Singh, S., Thammachote, T., Virmani, R.: Bankruptcy prediction by neural networks. In: Trippi, R., Turban, E. (eds.) *Neural Networks in Finance and Investing*, pp. 159–176. Probus Publishing Company, Cambridge (1993)
- [22] Román, I., Gómez, M.E., de la Torre, J.M., Merelo, J.J., Mora, A.M.: Predicting financial distress: Relationship between continued losses and legal bankruptcy. In: *Proceedings of the 27th Annual Congress European Accounting Association*, Dublin, Ireland (2006)
- [23] Salcedo-Sanz, S., Fernández-Villacañes, J.L., Segovia-Vargas, M.J., Bousoño-Calzón, C.: Genetic programming for the prediction of insolvency in non-life insurance companies. *Computers and Operations Research* 32, 749–765 (2005)
- [24] Serrano-Cinca, C.: Self organizing neural networks for financial diagnosis. *Decision Support Systems* 17(3), 227–238 (1996)
- [25] Ultsch, A., Siemon, H.: Kohonen’s self organizing feature maps for exploratory data analysis. In: *Proc. INNC 1990, Int. Neural Network Conf.*, pp. 305–308. Kluwer, Dordrecht (1990)
- [26] Vesanto, J., Himberg, J., Alhoniemi, E., Parhankangas, J.: Self-organizing map in Matlab: the SOM toolbox. In: *Proceedings of the Matlab DSP Conference*, Espoo, Finland, pp. 35–40 (1999), <http://www.cis.hut.fi/projects/somtoolbox/>
- [27] Yin, H.: *The self-organizing maps: Background, theories, extensions and applications*, vol. 115 (2008), www.scopus.com
- [28] Zavgren, C.V.: The prediction of corporate failure: The state of the art. *Journal of Accounting Literature* 2, 1–38 (1983)

Ant Colony Optimization for Option Pricing

Sameer Kumar, Ruppa K. Thulasiram, and Parimala Thulasiraman

EITC Department of Computer Science, University of Manitoba, Winnipeg, MB, Canada
{sameer,tulsi,thulasir}@cs.umanitoba.ca

Summary. Option pricing is one of the fundamental problems in finance. This chapter proposes a novel idea for pricing options using a nature inspired meta-heuristic algorithm called Ant Colony Optimization (ACO). ACO has been used in many NP-hard combinatorial optimization problems and most recently in self-organized environments in dynamic networks such as ad hoc and sensor networks. The dynamic changes in financial asset prices poses greater challenges to exercise the option at the right time. The dynamic nature of the option pricing problem lends itself very easily in using the ACO technique to the solution of computing option prices. ACO is as intuitive as other techniques such as binomial lattice approach. ACO searches the computational space eliminating areas that may not provide a profitable solution. The computational cost, therefore, tends to decrease during the execution of the algorithm. There has been no study reported in the literature on the use of ACO for pricing financial derivatives. We first study the suitability of ACO in finance and confirm that ACO could be applied to financial derivatives. We propose two ACO based algorithms to apply to derivative pricing problems in computational finance. The first algorithm, named Sub-optimal Path Generation is an exploitation technique. The second algorithm named the Dynamic Iterative Algorithm captures market conditions by using an exploration and exploitation technique. We analyze the advantages and disadvantages of both the algorithms. With both the algorithms we are able to compute the option values and we find that the sub-optimal path generation algorithm outperforms the binomial lattice method. The dynamic iterative algorithm can be used on any random graph and the uncertainties in the market can be captured easily but it is slower when compared to the sub-optimal path generation algorithm.

4.1 Introduction

Computational Finance is a cross-disciplinary area that relies on mathematics, statistics, finance and computational algorithms to make critical decisions. One of the core tasks in this area is to analyze and measure the risk component that a financial portfolio would create. A portfolio would generally comprise of stocks, bonds and other financial instruments such as options.

4.1.1 Basic Terminology

An *option* is a contract in which the buyer (*holder* of an option) has the right but not the obligation to buy (with *call* option) or sell (with *put* option) an underlying asset (for example, a stock) at a predetermined price (*strike price*) on or before a specified date

(*expiration date*). The seller (also known as *writer*) has the obligation to honor the terms specified in the option contract (*option*). The holder pays a premium to the writer (see, for example, [23, 27]). In option pricing, the object is to find the price $F(S, t, T, r, \sigma, K)$ at time t of an option (call or put) on stock (the underlying asset of the option), given the following information: S =the current stock price; T =time to expiration of the option contract; r =risk-free rate of interest; σ =the volatility of stock prices; and K =the strike price of the option.

The following paragraph illustrates a simple example of a call option. Suppose that the price of a particular stock today is \$20. An investor enters a call option to buy this stock at \$21 six months from today. On the expiry date, assume the stock price is \$30. The holder of the option can exercise the option and buy the stock at \$21; and by immediately selling it in the open market at \$30 the holder can therefore have a profit of \$9. If the stock price on the expiry date is \$15, the holder is not obligated to buy this (why buy something at a higher price when the same is available at a lower cost in the open market?). The option expires worthless.

There are various styles of options including European options (which can be exercised only at the maturity date), American options (which can be exercised any time prior to the maturity date), barrier options (looking for first stopping time), Bermudan options (which have multiple exercise points during the contract period), and Russian options (the date of option expiration is floating).

Option pricing is one of the fundamental problems in finance which has led to two Nobel prize awards. In 1997, Myron S. Scholes and Robert C. Merton shared the Nobel prize for the Black-Scholes-Merton model [1]. Recently, Engle received a Nobel prize in 2003 for his Auto-Regressive Conditional Heteroskedasticity (ARCH) model [17]. The generalized ARCH (GARCH) model has been a subject of intense research and use in option pricing (see for example [15, 16]). Based on the fundamental concepts in these models, there are many numerical techniques proposed in the literature for option pricing.

4.1.2 Nature Inspired Algorithms in Finance

One of the current trends in science and engineering research is the introduction of nature inspired algorithms. Nature inspired algorithms [13], including evolutionary and swarm algorithms, have been used to solve many combinatorial optimization problems, including problems in telecommunications [3, 4] and dynamic networks such as mobile ad hoc networks [20]. Swarm intelligence is an artificial intelligence technique inspired by animal societies such as bees [31, 34], termites [30], and ants [11]. These creatures live in a hostile, decentralized environment and communicate with one another through stigmergy to accomplish their tasks such as finding the food source. Ant Colony Optimization (ACO) is one such swarm intelligence technique inspired by real ants. The ants communicate with one another by depositing *pheromone* (scent) on the ground to attract their fellow ants to follow their trail, one of the stigmergic approaches in the animal world.

The feasibility of evolutionary algorithms in the field of finance has gained importance and is being explored [2]. These algorithms have prospects in many areas of finance such as to evolve trading rules, diagnosis of company's future etc. A key

Table 4.1. Similarities between ACO and the real market

ACO	Market
Meta-heuristic search technique	Investors look for best time to buy or sell
Based on the collective behavior of decentralized ants; self-organized systems	Based on the collective behavior of decentralized traders, investors etc.; self-organized systems
No centralized control structure	No central control among investors
Local interactions between agents lead to the emergence of global behavior	Interaction between investors lead to emergence of market behavior
The agents follow very simple rules which leads to very complex rules/algorithms at the global level	Investors follow simple rules that lead complex nature for the market

advantage of using evolutionary algorithms to design a trading process is that these algorithms can simultaneously evolve good rules and good parameters for such trading processes [2].

We found many similarities between the ACO and the real market (see Table 4.1). These similarities acted as motivation for the current study i.e. to apply ACO to the option pricing problem. For option pricing, the solution space consists of a large number of price nodes, each representing a time and price of the underlying asset during the option contract period. The ants are basically agents of an investor. They have a large bounded space of price nodes to search, in deciding the best time (node) to exercise the option. The nodes within the search space are dispersed in many locations and are connected in some random manner. The collective goal of the ants is to find the best node to exercise the option to help the investor in making an informed decision. This can be achieved by directing a path to the node, thereby allowing other agents to quickly arrive at the node. ACO is a probabilistic method. The underlying search space modeled as a graph is unstructured. Therefore, ACO allows flexibility by distributing the nodes randomly and thereby capturing the real marketplace.

The current study is a proof of concept using ACO for derivative pricing. We use American options as they are computationally challenging. To the best of our knowledge there is little if any prior literature on pricing options using ACO. We believe that ACO can be used for option pricing problem as (i) ACO is as intuitive as other techniques; (ii) other techniques and approaches (for example, binomial lattice) need to compute the entire solution space before an option value is calculated; and (iii) the computing cost could be smaller for ACO when compared to other techniques since the solution space explored can be restricted towards certain directions based on the local optimum solution.

We divide the rest of the chapter as follows. In the next section, we provide some background information on option pricing with a detailed description of one of the common and classical techniques, the binomial lattice approach. We start Sect. 4.3 with a description of the ACO algorithm and then highlight some related work in general finance that uses other nature inspired algorithms. We describe the sub-optimal path generation and dynamic iterative algorithms in Sect. 4.4 . In Sect. 4.5 we present the

implementation details. We provide results and discussion in Sect. 4.6, and some important conclusions and future work in Sect. 4.7.

4.2 Binomial Lattice Option Pricing Model

Derivative pricing is the backbone of many major financial problems such as value at risk and portfolio optimization. Derivative products have become so complex that even the writers themselves do not always understand how they behave in the market and therefore pricing options has become a complicated, challenging and computationally intensive problem in finance.

One of the early models for option pricing is the Nobel prize-winning Black-Scholes-Merton (BSM) model [1, 26]. Black-Scholes developed a model to alleviate risk involved in financial investments. Merton [26] augmented it with stochastic calculus resulting in a stochastic partial differential equation (PDE) for the option. This model is valid for simple European options and a major assumption was that the underlying stock would have constant volatility. A closed-form solution is available only for simplified BSM model with many assumptions, especially since numerical techniques for solving PDE were not popular in the finance community at that time. This scenario changed in 1979 when first discrete time approach was proposed by Cox-Ross-Rubinstein (CRR) [6]. The CRR binomial model is a simple and intuitive numerical technique developed for pricing options. The binomial option pricing model has proved over time to be the most flexible, and popular approach to price options. If constructed assuming the same initial conditions, binomial model agrees asymptotically with the Black-Scholes model. Moreover, the binomial model can be used for pricing American style options. The standard binomial option pricing model assumes that the binomial tree is recombining with constant volatility, constant risk-less return and constant payout return. However, these assumptions could be relaxed unlike those of the BSM Model.

The binomial model uses a binomial tree structure to price options. In this approach, we divide the time between valuation (current) date and expiration date into a certain number of time steps. Each node in the tree represents a possible price of the stock (underlying asset) at a particular time. In binomial method, knowing the asset price is the basis for computing the option value. The valuation of the binomial tree is iterative. After building the tree with asset price distribution it starts from the leaf nodes and works backwards towards the root node, which represents the valuation date. The option price at the valuation date is calculated by pricing option at all the intermediate nodes between the expiration date and the valuation date.

Computing option prices using this iterative method involves three steps: (1) Price tree generation (2) Computing option price (local pay-off) at each leaf node (3) Iterative computation of option values at earlier nodes using a discounting factor. The value at the root node is the value of the option. Some of the variables and parameters that are required to price an option are: Asset Price: S ; Strike Price: K ; Time to Maturity: T ; Interest Rate: r ; Number of Steps: N ; Interval time between 2 steps: Δt ; Volatility: σ ; Probability: p ; Upward Movement: u ; Downward Movement: d .

Binomial Tree

The prices at each node is computed by working forward through the tree from valuation date (current date) to expiration date. An example of the binomial price tree is shown in Fig. 4.1. At each step (or level) of the tree, the price of underlying asset can increase or decrease by a factor of u or d respectively, where u and d are generally 10% or 20% change. If S is the current price of the underlying asset, then the price at the next step will be either $S_u = S \times u$ or $S_d = S \times d$. Asset price computation is repeated until the expiration date is reached. The value of the increase (u) or decrease (d) factor is computed using the volatility σ of the underlying asset and the interval time between two steps [6] as follows: $u = e^{\sigma\sqrt{\Delta t}}$ $d = e^{-\sigma\sqrt{\Delta t}} = \frac{1}{u}$. It is easy to notice that the down movement is assumed to be inversely proportional to the up movement.

Option Value at the Expiration Date

The computation of the option value starts at the leaf nodes (maturity date) of the binomial tree. The value of the option at the leaf nodes is simply its local pay off, which is defined as follows. For a call option it is $= \text{Max}[(S - K), 0]$ and for a put option it is $= \text{Max}[(K - S), 0]$.

Option Value at Intermediate Nodes

Once the value of the option is computed at the leaf nodes, working backward (Fig. 4.2) towards the root node (valuation date) gives the value of the option. The option value at a node is computed using the option values of the two children nodes (*optionup* and *optiondown*) weighted by respective probabilities, *optionup* is multiplied by p , which is generally understood (interpreted) as the probability of underlying asset to move up and *optiondown* is multiplied by $(1 - p)$, which is the probability that the price of the underlying asset will decrease. The value of p is given by [6] $p = \frac{e^{(r-q)\Delta t} - d}{u - d}$ and the

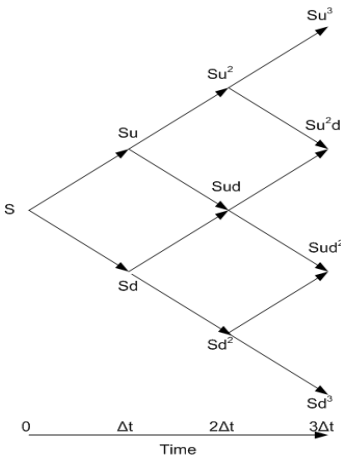


Fig. 4.1. Binomial Price Tree

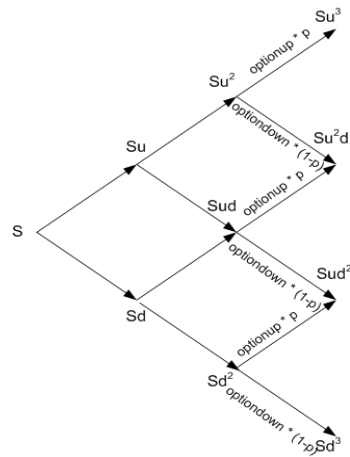


Fig. 4.2. Backward Computation of Option Prices

option value is given by $OptionValue = e^{-r\Delta T}(p \times optionup + (1 - p) \times optiondown)$ where the $e^{-r\Delta T}$ is the discount factor. In other words, the option value at a node is the discounted value of the weighted sum of option values at a future time due to increases or decreases in the value of the underlying asset.

For European options, the value at each node is simply the option value computed using the above formula. The value at the root node is the value of the option on the valuation day. For an American option, the value at each node is

Max[option value based on discounting, local pay-off]

where the local pay off is $\max(S - K, 0)$ for a call option or $\max(K - S, 0)$ for a put option. This allows us to identify the best possible time to exercise the American option.

Thulasiram et al. [33] have designed and developed a parallel, multithreaded algorithm for the binomial lattice model. Thulasiram and Bondarenko [32] have developed a parallel algorithm for multidimensional option pricing problem. Huang has extended these studies to Asian options [21, 22]. A list of other related works using a binomial lattice approach can be obtained from the references of these papers. Other approaches for pricing options include Monte-Carlo simulation, fast Fourier transform, neural networks etc.

4.3 Ant Colony Optimization (ACO)

Swarm Intelligence is an approach to solve complex problems based on the social behaviors of some insects and animals. Ant Colony Optimization (ACO) [9, 11] falls under the banner of natural computing and is a guided, stochastic, search technique. The ACO approach is based on the foraging behavior of some species of ants. Many studies have shown interest in finding how ants are able to find the shortest paths between their nest and food sources. Research in this area has discovered that ants use indirect communication to communicate with other ants. This indirect communication involves modifying their environment, and is called *stigmergic* communication. In stigmergic communication, the ants move to a food source away from their nest, depositing on the path (ground) a chemical substance excreted from their body known as *pheromone*. Other ants can sense and perceive these pheromones and tend to follow the path with highest pheromone concentration. Using this communication, ants are able to forage and collect their food in an efficient way.

ACO involves a number of artificial ants to build solutions to an optimization problem. These ants exchange information about their own solutions to other ants using a communication scheme similar to the one used by real ants [14].

4.3.1 Foraging Behavior of Ants

In Fig. 4.3a the ants move from node A (the nest) to node E (the food source). We introduce an obstacle which obstructs the path between A and E. Ants have to make a decision whether to turn right or left at position B or D depending on whether they are coming from A to E or E to A respectively (Fig. 4.3b). The decision or choice of an ant is based on the intensity of the pheromone on the trails. The first ant reaching

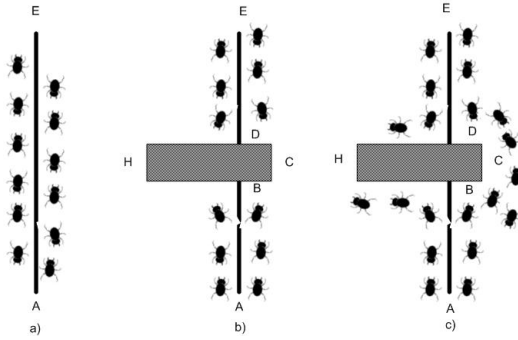


Fig. 4.3. Ants facing obstacle in their path

point B or D has the same probability to turn right or left, as there is no pheromone on either of the two alternative paths. The ants following the shorter path will travel to and from the food source more quickly than ants following the longer path. In the example, the path BCD is shorter than the path BHD (Fig. 4.3c). Therefore, the number of ants following the path BCD per unit time will be higher than the number of ants following BHD assuming all ants move at the same speed. This causes the pheromone levels on the path BCD to be higher than on path BHD. The ants approaching point B choose path BCD because of higher pheromone levels, thereby, following a shorter path to the food source E [12]. As time progresses, the pheromone along the longer paths evaporates leaving the ants to converge along shorter paths.

The objective of ACO in most applications is to find the shortest path. However, in option pricing, the primary interest is not in finding the shortest path, rather in finding the best node that allows the investor to exercise the option. Therefore, the general purpose ACO algorithm has to be altered to handle this problem.

4.3.2 Various ACO Algorithms

In the literature many types of ACO algorithms have been proposed. The first algorithm based on the behavior of ants was Ant System (AS) [12]. This paper introduced an optimization technique based on ant behavior and proposed a method for solving the Traveling Salesman Problem (TSP). In the TSP ACO implementation, an ant is placed at each city and it travels from current city, visiting other cities only once and returning to the original or initial city at the end of the tour. The ants communicate by depositing pheromones on edges connecting the cities. Eventually, the pheromone concentration on the shortest path increases due to more tours by ants. The pheromone on unused edges gradually evaporates completely in absence of any reinforcement of pheromone. Ant System work led to a development of number of ACO algorithms with good results in many applications.

4.3.3 Applications of ACO

Dorigo [9, 12] proposed ACO which involves distributed computation, positive feedback and a greedy heuristic to solve a given problem. Distributed computing helps in

searching a wide area of the problem domain, positive feedback helps in finding good solutions and a greedy heuristic is needed to find solutions in early stages. ACO has found many applications and successful implementation has been applied to a number of different combinatorial optimization problems. Some examples of ACO being applied to static optimization problems are: Traveling Salesman Problem, Quadratic Assignment Problem, Job-shop Scheduling Problem, Shortest Common Super-sequence Problem, Graph Coloring Problem, Vehicle Routing Problem, Routing in ad hoc networks and Sequential Ordering Problem. The following section provides a brief review of the related work on ACO in finance.

4.3.4 Applications of ACO in Finance

To the best of our knowledge, there is no work reported in the literature on the use of ACO for the option pricing problem. In this section we briefly outline the literature related to the use of ACO on other problems and applications, including finance though not option pricing. In finance, knowing the volatility of a financial instrument is important for many financial engineering strategies. However, predicting the volatility of an instrument, say a stock price, is a daunting task and is a research area by itself. Researchers and practitioners, using historical data have developed methods for predicting historical volatility (see [28] for a discussion on volatility). By its nature, historical volatility only reflects past price events and the accuracy of stock prices predicted based on historic volatility is questionable. There have been considerable efforts to develop quality methods to measure volatility accurately. Implied volatility measures the intrinsic dependence of past stock prices not only with time but with other factors affecting the price over a period of time. Keber and Schuster [24] used generalized ant programming to derive analytical approximations to determine the implied volatility for American put options. They used experimental data and validation data sets for computing the implied volatility. Their results outperformed any other approximations. Generalized ant programming is a new method inspired by genetic programming approach (introduced by Koza [25]) and Ant Colony System (ACS) [10, 18].

4.4 ACO for Option Pricing

In this section we develop two algorithms based on ACO for the option pricing problem. We use the following terminologies in this work. Our first algorithm is known as sub-optimal path generation algorithm spans the solution space like the binomial tree approach. Our second algorithm is known as the dynamic iterative algorithm where we shed the natural progression of the spanning of the solution space. Nodes in this algorithm are more random than in the sub-optimal path generation algorithm. A node in the graph is a possible price that the underlying asset could take. A path is a route that ants could take in moving towards an optimum solution (node). Moving from a node to another node means the ants progressing in time towards an optimum solution.

4.4.1 Sub-optimal Path Generation Algorithm

There are several components in this algorithm. (1) The algorithm starts by injecting ants from the valuation date (root). Ants can explore any path based on random

behavior. (2) Individual ants compute the payoff at each node based on an expression $V_{def} \leq X - S \leq V_{opt}$. As soon as an ant finds a value between the predefined values in the expression it updates (increasing) the pheromone density leading to the node. (3) Updating the pheromone on the path which has a good node (that satisfies the expression in the previous step) helps in making the path more attractive for other ants to explore more in the neighboring areas. However, ants still keep going on other paths to explore the whole search space. (4) If ants find a better node then pheromone values are updated to make the path to the newly found node more attractive. (5) The process continues until the optimum node is found or until the constraint how far (how close to expiration date) the user wants to search in a graph is exceeded.

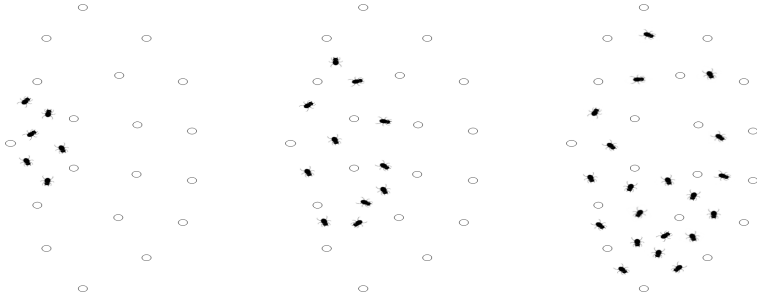


Fig. 4.4. The figures capture the ant movement at various times of the execution of the algorithm

4.4.2 Dynamic Iterative Algorithm

The elements of this algorithm are as follows. (1) In the solution space the source is the current date and the destination can be any node in the solution space. Destination is how far (future date) ants will search. The destination can be updated with a better node after some time to help us explore the whole search space. Here, “some time” refers to the computational time needed to find the optimal node for a source and a destination. (2) The algorithm starts by injecting ants at the source, and the ants explore random paths to reach the destination. (3) Each ant while traveling to the destination identifies the best node throughout the journey which is computed using a predetermined expression (for example, in case of put option $X - S$, where X is the strike price and S is the asset price). (4). The ant updates the pheromone density (locally) on the path. The local pheromone update is done while looking for solutions. On the other hand, the global pheromone update is done after all ants have found a solution. The purpose of the local pheromone update rule is to make the visited nodes less and less attractive as they are visited by ants, indirectly favoring the exploration of not yet visited nodes. The purpose of the global pheromone update rule is to promote ants to search for nodes in the neighborhood of the best node found so far. (5) Once all ants have updated their local pheromone values, the best nodes from all the paths explored by the ants are compared and the best node is identified. Now pheromone density is updated globally so that more ants follow the path which leads to the best node. The reason is to attract more ants so we can explore all possibilities from the identified node. However, to avoid a local

minima, pheromone gets exhausted (by local update and evaporation) after some time which forces ants to explore other areas in the solution space. Here some time refers to the computational time taken by ants to slowly lower the pheromone value to initial value. (6) We keep sending more ants from the source. Ants will follow the paths that have higher concentrations of pheromones compared to random behavior, that was seen initially. (7) The algorithm keeps finding better solutions until the best node is found.

We generate a random acyclic graph. A certain number of vertices (nodes) are connected to each other using edges (paths) randomly. The reason for not using a cyclic graph is that in real world it is not possible for investor to go back in time and have the same choices again. Ants can wander on these paths moving from node to node. Each node stores an asset price and each edge represents the transition from one stock value to another.

In the algorithm, ants deposit pheromone on the paths while walking and follow paths based on probability of pheromones deposited previously. Initially, all the paths have initial pheromone τ_0 on them so ants choose random paths. After a brief transitory time, the difference between the amounts of pheromones will differ. So the new ants coming from the nest will prefer in probability to choose the path with higher pheromone. We have taken the probability expression from ACS [10] and modified to it suit the application. The following expression gives the probability an ant k at node r chooses to go to node s

$$P_k(r, s) = \begin{cases} \frac{[\tau(r, s)][\eta(r, s)]^\beta}{\sum_{u \in J_k(r)} [\tau(r, u)][\eta(r, u)]^\beta} & \text{if } s \in J_k(r) \\ 0 & \text{otherwise} \end{cases} \quad (4.1)$$

where τ is the pheromone, η is the difference between the stock values of (r, s) , $J_k(r)$ is the set of nodes that ant k has still to visit and β is a measure to determine relative importance of pheromone versus difference between the stock values. In this algorithm, the ant with the best value globally deposits the pheromone. The probability equations are intended to make search more directed meaning ants mostly search in the neighborhood of the best node found by the algorithm. Global updates are performed after all ants have reported their own best node. The best ant (globally) is identified based on the best values reported by all ants. The pheromone level is updated by this globally best ant by update rule

$$\tau(r, s) \leftarrow (1 - \alpha)\tau(r, s) + \alpha\Delta\tau(r, s) \quad (4.2)$$

$$\text{where } \Delta\tau(r, s) = \begin{cases} (V_{gb})^{-1} & \text{if } (r, s) \in \text{global best node} \\ 0 & \text{otherwise} \end{cases} \quad (4.3)$$

where α ($0 < \alpha < 1$) is the pheromone decay parameter and V_{gb} is the difference between the initial stock price and the best node globally. This is intended to provide greater amount of pheromone for more profitable nodes. While searching for the solution, the ants visit paths and alter their pheromone by applying a local pheromone update

$$\tau(r, s) \leftarrow (1 - \rho)\tau(r, s) + \rho\Delta\tau(r, s) \quad (4.4)$$

where $0 < \rho < 1$.

Some Observations

The ants in the natural world try to find a shortest path from nest to a food source. The idea from natural world put forth in ACO algorithm is improved for the finance application as follows

- The option pricing application requires finding the best time for exercising the option. That means the ‘food’ consumption is going to happen only once (exercise of option). That single best node would become the destination where the option is exercised. In other words, the first major modification is that we relax the requirement of the shortest path for the ants to follow.
- The local optimum values are used to direct more ants to explore further in the solution space from the local optimum node.
- ACO has not been used to price options. Designing and implementing naive algorithm acted as a feasibility study for the current work. ACO only generates and evaluates a subset of the paths unlike binomial lattice model where all paths and nodes are exhaustively evaluated. This would address memory issues and explore the best time to exercise the option in an efficient way.

4.5 Implementation Details

In the literature on parallelizing combinatorial optimization problems such as TSP [29] and scheduling problems [7], where the graph is static, the implementation has been done on parallel computers using the distributed memory machine [19] or shared memory machines [5]. When the problem is communication intensive, shared memory machines are preferred. In ACO, the ants move from one node to another node, which may translate to moving from one processor to another, thereby, increasing communication. Shared memory machines have shown [8] to produce better performance results for ACO algorithms. In applications such as mobile ad hoc networks, where the graph is dynamic and mobility needs to be incorporated, a simulator has been used for the implementation. Since, the option pricing problem considers a static graph (solution space), we implement our algorithms on a shared memory machine using OpenMP.

With *call* and *put* options we expect the underlying asset’s price to go one direction, up for *call* and down for *put*. We can utilize this fact in channeling the ants in a particular direction towards the best node in the solution space. That is, when an initial set of ants move in one direction more ants follows in this direction. These ants are *reactive* ants, which we use in our sub-optimal path algorithm.

In another scenario, ants can explore the entire solution space for a best node both for *call* and *put* options. This is useful when there are both styles of options issued for the same underlying asset. The ants explore the solution space proactively on their own without any direction by the investor. While few of the *proactive* ants explore the solution space independently, others follow earlier ants just like in reactive case.

In our implementation, ants are agents for a single investor. In the current scenario all the agents are working for a single investor in finding a solution for the option prices that helps the investor in making an informed decision for entering the option contract. A node is a price point at a given time. Moving from a node to a node in the future

time, there could be various possible prices for the asset. The future time could be the next second, minute, hour, day or week. Each thread in the system represents an ant which can randomly move around the solution space. For ease of implementation, we have limited the number of nodes. We limit the number of nodes for implementation because of memory constraints. The primary goal of the algorithms is to find any good node (right time to exercise the option) rather than finding the shortest path leading to that node in minimum time. More nodes means more sampling, which is better, in general. However, sampling by minutes or every 10 minutes or even every 30 minutes may not yield much better results (in terms of accuracy) in pricing. Further, higher frequency sampling would result in a heavier computational load and hence a higher computational cost. Hence, we restrict the solution space to a reasonable number of steps. However, for some options it might be necessary for a very small step size for accuracy purposes in which case the number of nodes in the solution space would increase exponentially and finding the best node to exercise the option becomes a daunting task. Parallel computing will be useful for the problem in such a situation. In this study, we are not exploring very large graph, still we employ parallel computing with future problem size in mind.

4.5.1 Sub-optimal Path Generation Implementation

We compare the results obtained from the sub-optimal path generation algorithm with binomial lattice results with same set of parameters (strike price, asset price, etc.) used in the sub-optimal path generation algorithm implementation. We use synthetic test data to compare the proposed algorithms with the binomial model. The synthetic data used can easily be mapped to a real world stock.

The algorithm starts by sending ants randomly on each path (in the solution space) from an initial node. Once an ant reaches the next node it computes the local pay-off for a *put* option as $(X - S)$, where X is the strike price and S is the asset price at the new node. Once the solution satisfies either of the boundary conditions

$$X - S \geq Y \quad (4.5)$$

or

$$(n * \Delta t) > a \text{ predefined time during contract period} \quad (4.6)$$

where Y and *predefined time* are user-defined parameters, the option price computed is the optimum value. We have conducted experiments to price American *put* option using the proposed algorithms. We have experimented by varying the parametric conditions: initial stock price, strike price, volatility, time to maturity and number of time steps. The algorithm provides the optimal solution based on the user defined boundary conditions. The primary goal is to find the optimum node rather than finding it in shortest path or in minimum time. The measure of comparison is the pay-off an investor will get by exercising the option.

4.5.2 Dynamic Iterative Implementation

As an example, Figs 4.5 (Graph 1) and 4.6 (Graph 2) show two random acyclic graphs used in the experiments with 15 and 30 nodes respectively. Here, $V_x : A$, refers to vertex

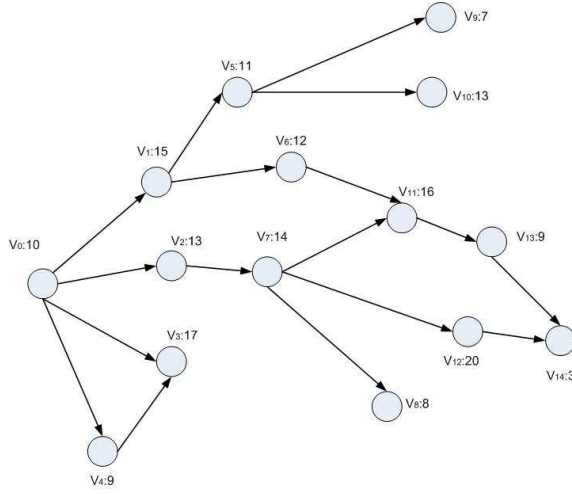


Fig. 4.5. Graph 1 (15 nodes)

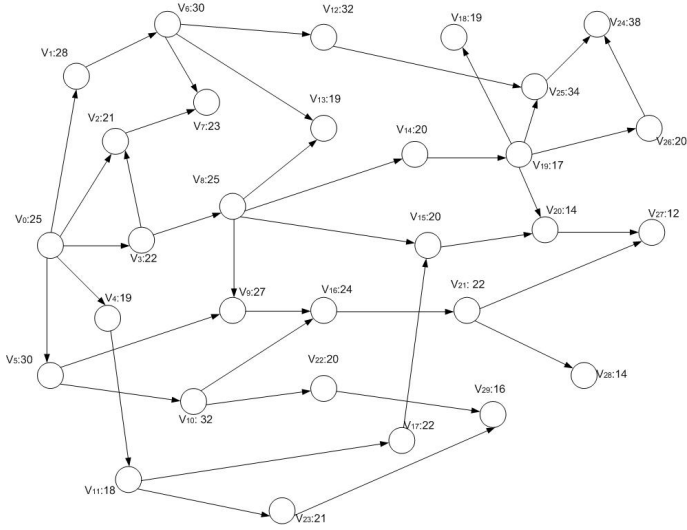


Fig. 4.6. Graph 2 (30 nodes)

V_x representing asset price A . For example in Fig. 4.5, $V_7 : 15$ refers to vertex 7 with asset price 15. We applied the dynamic iterative algorithm to these random graphs. The measure of comparison is the pay-off an investor will get by exercising the option. In other words, the time to exercise the option which gives us the highest profit. In all our experiments the numeric parameters are set to the following values: $\beta = 2, \alpha = 0.1, \rho = 0.1$ and $\tau_0 = 0.1$.

4.6 Results and Discussion

The experiments were done on an eight node shared memory machine with memory and hard disk of 7.5 GB and 36 GB respectively. To compare and validate the results obtained from the ACO we independently implemented the binomial lattice algorithm for different data sets used in the ACO study and gathered both timing and pricing results. The pricing results from both the algorithms agree with results from binomial method.

4.6.1 Sub-optimal Path Generation Algorithm

We use a structured graph for the implementation of this algorithm, The graph is analogous to a binomial or trinomial tree. The contract period for our experiments is six months and the number of time steps varies between 2000 (amounts to about 30 minutes interval) to 5000 (amounts to about 11 minutes interval). That is, in some experiments ants compute option prices for changes in the asset price happening every 11 minutes and in other experiments ants compute option prices for changes in the asset price happening every 30 minutes. In Table 4.2 and in Fig. 4.7, we set the desired profit level between \$10 and \$25 at \$5 intervals to determine how quickly one can achieve such profits. We set the number of time steps to be 2000. As the profit level increases, the execution time also increases. The reason for this is that for better profitability the ants would have to search more solution space and do more computation, hence increasing the execution time.

Table 4.3 and Fig. 4.8 show the execution time for various time steps at a desired profit of \$15. As the time steps increase, with constant profit level, the execution time also increases. This is because increasing the number of time steps implies increasing

Table 4.2. Desired Profit vs Execution Time with Time Steps 2000

Parameters	Value1	Value2	Value3	Value4
Desired Profit	10	15	20	25
Execution Time (secs)	0.049	0.106	0.515	1.269

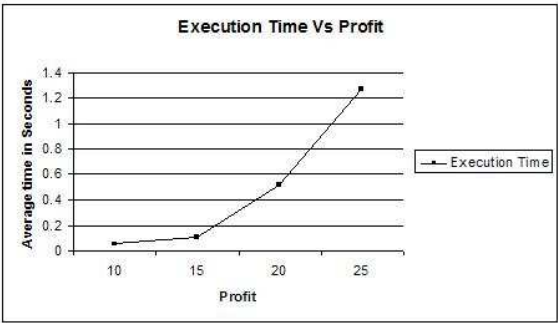
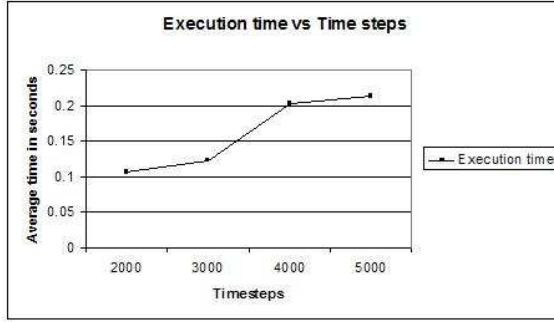


Fig. 4.7. Chart for Desired Profit vs Execution Time with 2000 Time Steps

Table 4.3. Time Steps vs. Execution Time with \$15 Profit

Parameters	Value1	Value2	Value3	Value4
Time Steps	2000	3000	4000	5000
Execution Time	0.106	0.123	0.203	0.213

**Fig. 4.8.** Chart for Time Steps vs. Execution Time with \$15 Profit**Table 4.4.** Comparison between Binomial method and sub-optimal path generation algorithm

Timesteps	Binomial Method (Secs)	Sub-optimal Path Generation (Secs)
10000	3	0.37
20000	13	1.44
40000	67	2.38
70000	220	7.46
100000	451	11.71

the number of nodes in the solution space. Since the solution space is analogous to a binomial tree, the number of nodes can be of the order of 2^L (for a binomial tree), where L is the number of time steps. However, as can be seen from Table 4.3, the increase in time is negligible. We compared the sub-optimal algorithm to the binomial lattice method (Table 4.4). Our algorithm performed better than the binomial lattice method in terms of speed. The performance of sub-optimal algorithm is better because in the ACO algorithm we do not generate all the nodes at each time steps as in binomial lattice method. Our algorithm only generates and computes price on nodes which are needed to price the option.

Since the ants search the solution space in various directions we observed a proportional increase of the execution time with higher desired profit. This happens due to structured search by the leading ants. This proportionality caught our attention to have a close look at the algorithm. Though the ants were allowed to search the solution space, there was a controlled exploration. We allowed only a limited number of ants in a smaller region of the solution space to reach a sub-optimal solution at a node from which more ants were allowed to search in an orderly fashion. This is the nature of the algorithm. One set of ants searching and dragging more ants to the sub-optimal

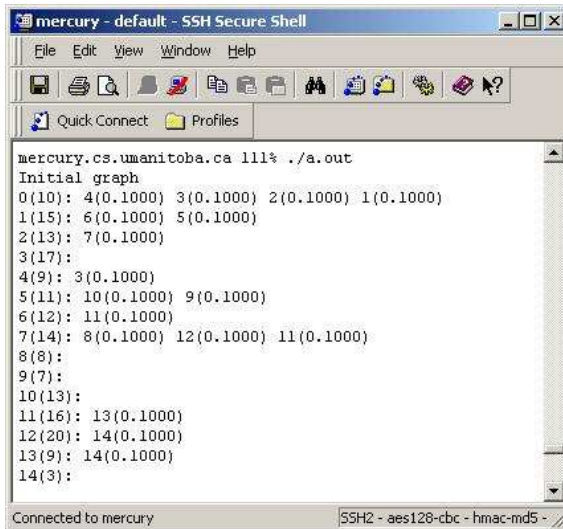
solution. That is, most of the ants are reacting to the first few ants. In other words, the *reactive ants* are exploiting the paths generated by few leading ants and hence they are more exploiting than exploring. Therefore, it is expected that with *proactive ants*, which explore the solution space, we would expect better results, which is studied in the dynamic iterative algorithm.

Also, note that in the sub-optimal algorithm, we have a structured graph. The reason for this is that we initially wanted a fair comparison between the binomial lattice method and our ACO method. In the dynamic iterative algorithm, we do not restrict the structure of the graph. The data structure for the solution space is not necessarily trees. They are random graphs used to capture real market movements.

4.6.2 Dynamic Iterative Algorithm

We applied the dynamic iterative algorithm to both the graphs shown in Figs. 4.5 and 4.6. Fig. 4.9 gives the initial graph for Fig. 4.5 (Graph 1). The graph is represented by an adjacency list. For example, $V_x(A) : V_y(p_1)V_z(p_2)$ means vertex (V_x) has an asset price of A and is connected to vertex V_y with pheromone p_1 and vertex V_z with pheromone p_2 . In Fig. 4.9, 1(15): 6(0.1000) 5(0.1000) represents vertex V_1 with asset price 15 connected to vertex V_6 and vertex V_5 with pheromone level 0.1 at both nodes. Applying ACO to the initial graph shown in Fig. 4.5 (Graph 1), we reach the final graph shown in Fig. 4.10. The best vertex or node to exercise the option is V_{14} with an asset price of 3. The path to the best node is $V_0 \rightarrow V_1 \rightarrow V_6 \rightarrow V_{11} \rightarrow V_{13} \rightarrow V_{14}$. Also, note the changed pheromone levels by which ants are guided to the best solution.

Similar results for Fig. 4.6 (Graph 2) are also shown in Figs. 4.11 and 4.12. Fig. 4.11 shows the list of vertices and how they are connected to other vertices in the graph. It



```

mercury.cs.umanitoba.ca 111% ./a.out
Initial graph
0(10): 4(0.1000) 3(0.1000) 2(0.1000) 1(0.1000)
1(15): 6(0.1000) 5(0.1000)
2(13): 7(0.1000)
3(17):
4(9): 3(0.1000)
5(11): 10(0.1000) 9(0.1000)
6(12): 11(0.1000)
7(14): 8(0.1000) 12(0.1000) 11(0.1000)
8(8):
9(7):
10(13):
11(16): 13(0.1000)
12(20): 14(0.1000)
13(9): 14(0.1000)
14(3):
  
```

Fig. 4.9. Initial Graph 1

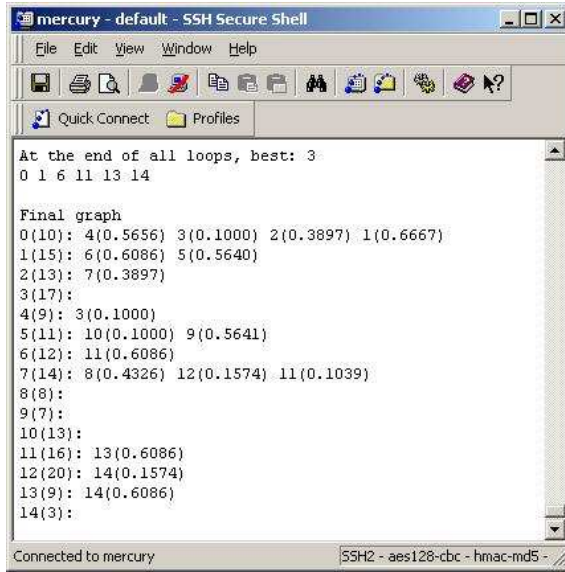


Fig. 4.10. Final Graph 1

also captures the initial pheromone level (i.e. 0.1) on the edges between the vertices. We apply our algorithm to Graph 2 and determine that the best node is vertex V_{27} with asset price 12. The algorithm also computes the path of the ants followed to reach this best node. The path is $V_0 \rightarrow V_4 \rightarrow V_{11} \rightarrow V_{17} \rightarrow V_{15} \rightarrow V_{20} \rightarrow V_{27}$.

For verification of our pricing results, we have done the following during our implementation. We opened a counter to store the asset prices from nodes when we generate the graph. We compared the asset price from a node as we generate the node with the asset value in the counter. For a *put* option, the counter price value is replaced with the asset price at a new node if the price at the new node is smaller than the counter price value (for a *call* option, the counter price is replaced with a larger node price). This information tells us beforehand where to expect the best option price among various nodes generated. We use this to verify the best node computed by the dynamic iterative algorithm. This is only for verification purposes. In all our experiments, dynamic iterative algorithm worked well in computing the best node to exercise the option.

Table 4.5, shows that the execution time increases as the number of nodes is increased. Note that unlike the sub-optimal algorithm we do not fix a constant profit because the graph is random and we do not restrict the limit on the gain. In other words, we make the boundary open. From Table 4.5, it can be seen for 10 nodes, the execution time is 3 seconds while for a larger graph (10000) it is 9.14 seconds. The overhead incurred in local and global pheromone updates, is predominantly attributed to the larger time needed for a smaller graph.

The market is full of uncertainties. We generally do not know what is going to happen tomorrow. In the binomial lattice algorithm the price change happens at a given node only in two different ways: up or down, and generally by a fixed factor. Similarly in

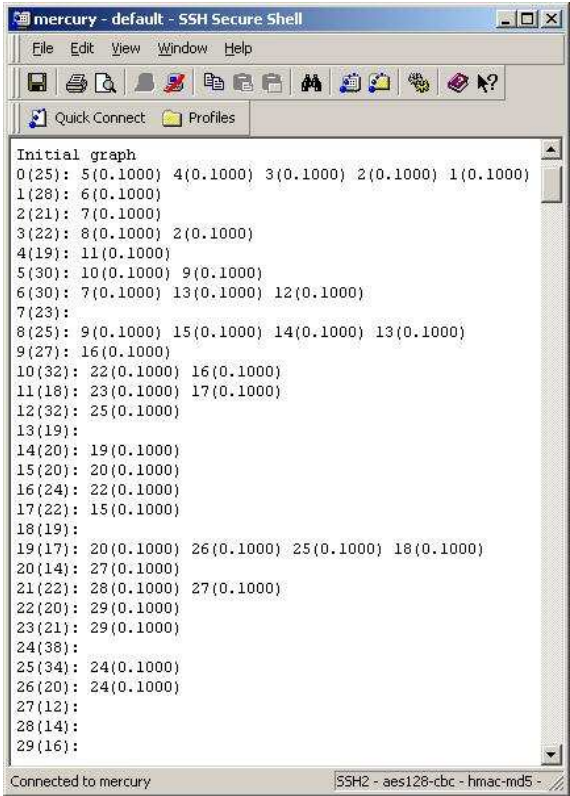


Fig. 4.11. Initial Graph 2

Table 4.5. Number of Nodes vs Execution Time

Parameters	Value1	Value2	Value3	Value4
Number of Nodes	10	100	1000	10000
Execution Time (in secs)	3.06	4.90	7.01	9.14

trinomial lattice it happens three different ways. One advantage of using ants for finding paths is that we can relax the restriction on price movements by letting the ants explore many different possible paths naturally. This physically means ants can capture day-to-day changes of the volatility in the market place. Therefore, unlike the other numerical approaches we do not have to specify volatility of the underlying asset, and therefore we have one less parameter to handle in the implementation. For simulation purposes, we restrict the number of different possible links that an ant can have. In our implementations, we have case studies with number of links between 5 and 20 as shown in Table 4.6. A single link between a node in a given time step to a node in the future (next time step) means the asset is stable. An experiment with 5 links means that an ant can have a maximum of five links going from one time step (node) to the nodes in the

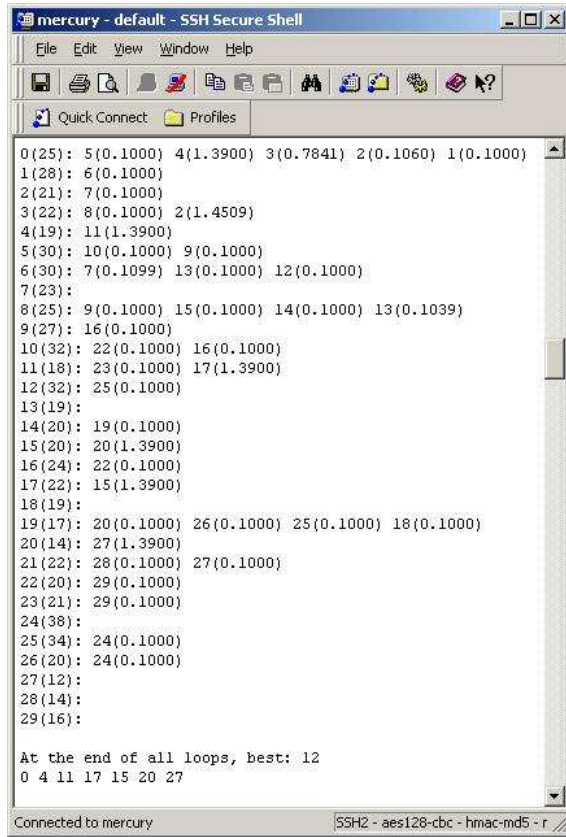


Fig. 4.12. Final Graph 2

Table 4.6. Number of Maximum Links vs. Execution Time

Parameters	Value1	Value2	Value3	Value4
Number of Maximum Links	5	10	15	20
Execution Time (in secs)	4.48	7.01	9.13	12.3

next time steps. That is, five different possible price changes are captured going from one node to the next. It need not have all five links. If there are all five links present from a node to the next time step, it implies that the underlying asset is highly volatile. In other words, we are able to capture the volatility of the underlying asset. Similarly, the presence of 20 links at a node implies that the asset price is highly volatile. In Table 4.6 it can be seen that as we increase the maximum number of links (or volatility), the execution time increases. This is because an increase in volatility increases the computational intensity of the problem.

4.6.3 Various Features of Sub-optimal and Dynamic Iterative Algorithm

The performance of the sub-optimal algorithm is far better than the dynamic iterative algorithm in terms of speed. This is due to the structure of the graph, we use graphs that are analogous to binomial & trinomial trees. In one of our experiments, we compared the execution time for both the algorithms for 10,000 nodes. The sub-optimal algorithm took less than one millisecond whereas the dynamic took 4.5 seconds (for maximum of 5 links). However, it should be noted that the sub-optimal algorithm will not be able to find a good pricing solution in a dynamic environment in which dynamic iterative algorithm works. This is because we do not use the parameters such as evaporation criteria and local pheromone update that are used in the dynamic iterative algorithm. The evaporation criteria allows ACO to converge towards a better solution by providing a means of exploring many different good paths, while at the same time eliminating the paths leading to bad nodes. The sub-optimal algorithm produces faster results because we are exploiting the solution space generated by few initial ants. The algorithm mainly relies on the exploitation of already-discovered paths and nodes. This is good for a given style of option, *call* or *put* by channeling the ants and it is expected that both local and global optimal solution will be available in the same neighborhood. Dynamic iterative algorithm is exploration as well as exploitation. Since it is advantageous to both styles of options *call* and *put*, it spends more time exploring. Also, by doing so, it finds the global optimal solution. Dynamic iterative algorithm converges more slowly than the static algorithm because of the use of local pheromone update and evaporation.

4.6.4 Comparison between Sub-optimal and Dynamic Iterative Algorithms with Binomial Lattice

We compared our algorithms to the binomial lattice model. The sub-optimal algorithm and binomial lattice model gave us the same pricing results, that is the best time to exercise the option. The sub-optimal algorithm is faster than the binomial model because the binomial model exhaustively prices option for all the nodes in the tree, whereas the sub-optimal algorithm does not have to price options at all the nodes. The dynamic iterative algorithm is not compared with the binomial model as the dynamic iterative algorithm works in a more complex and volatile environment than the environment in which binomial and sub-optimal algorithms work. For the dynamic iterative algorithm, the structure of the solution space is random and the volatility parameter need not be specified unlike the binomial and sub-optimal algorithm. In other words, this algorithm could handle underlying assets that are represented with varying volatility models. In sub-optimal and binomial lattice algorithms these volatility models would pose large computational challenges.

4.7 Conclusions

Pricing of options is a challenging problem. This work proposed a novel idea of using a nature inspired meta-heuristic algorithm called Ant Colony Optimization (ACO) to price options. We designed and implemented two new ACO-based algorithms to apply to a derivative pricing problem in computational finance. The first algorithm, named

sub-optimal path generation, generates various paths and identifies the best node in the solution space for exercising the option. In this algorithm, ants follow the paths generated by some leading ants to find better solutions as we search the solution space leading to an exploitation technique. The sub-optimal path generation algorithm outperformed the binomial lattice model. The second algorithm named the dynamic iterative algorithm, where few ants explore the solution space incrementally dragging more ants on the better path and eventually reaching the best node to exercise the option. This algorithm captures the real market place and finds the best time to exercise the option using exploration and exploitation techniques. Though the dynamic iterative algorithm converges more slowly than the sub-optimal path generation algorithm, the dynamic iterative algorithm can be executed on any random graph. The dynamic iterative algorithm is a better choice when dealing with the dynamic and highly volatile market place.

4.7.1 Future Work

In this chapter our study was limited to vanilla options. In future work, we would like to apply ACO to exotic options such as the pricing of Asian options. In our dynamic iterative algorithm, the ants keep track of the price values at each node along a path. We can easily use this price information to compute an average price so that we can apply this algorithm to price an Asian option. We intend to this in the near future. Similarly, the nature of our algorithms help us to price barrier option, where it is required to find the first stopping time, that is, earliest node to exercise the option. This can also be extended to Bermudan option.

We also intend to look into the possibility of using digital pheromones. Digital pheromones are data structures inspired by the insect model. In our application, we can allow ants to communicate more information amongst each other such as asset price, length of the path etc. using digital pheromones.

Bio-inspired algorithms such as ACO are gaining importance in many areas of finance such as to evolve trading rules, diagnosis of company's future etc. This work is the forerunner for more research to be done in future in financial applications using ACO algorithms.

Acknowledgments

The first author acknowledges the award of the University of Manitoba Graduate Fellowship and Manitoba Graduate Scholarship during his graduate studies. The last two authors acknowledge the partial support from Natural Sciences and Engineering Research Council (NSERC) of Canada and the Research Grant from University of Manitoba.

References

- [1] Black, F., Scholes, M.: The pricing of options and corporate liabilities. *Journal of Political Economy* 81, 637–654 (1973)
- [2] Brabazon, A., O'Neill, M.: *Biologically Inspired Algorithms for Financial Modelling*. Springer, New York (2006)

- [3] Caro, G.D., Dorigo, M.: AntNet: Distributed stigmergetic control for communications networks. *Journal of Artificial Intelligence Research* 9, 317–365 (1998)
- [4] Caro, G.D., Ducatelle, F., Gambardella, L.M.: AntHocNet: An adaptive nature-inspired algorithm for routing in mobile ad hoc networks. *European Transactions on Telecommunications (ETT), Special Issue on Self Organization in Mobile Networking* 16(5), 443–455 (2005)
- [5] Chandra, R., Dagum, L., Kohr, D., Maydan, D., McDonald, J., Menon, R.: *Parallel Programming in OpenMP*. Morgan Kaufmann, San Francisco (2001)
- [6] Cox, J.C., Ross, S.A., Rubinstein, M.: Options pricing: a simplified approach. *Journal of Financial Economics* 7, 229–263 (1979)
- [7] Delisle, P., Krackecki, M., Gravel, M., Gagné, C.: Parallel implementation of an ant colony optimization metaheuristic with OpenMP. In: *International Conference of Parallel Architectures and Compilation Techniques, Proceedings of the Third European Workshop on OpenMP, Barcelona, Spain*, pp. 8–12 (2001)
- [8] Delisle, P., Gravel, M., Krackecki, M., Gagné, C., Price, W.L.: Comparing parallelization of an ACO: Message passing vs. shared memory. In: Blesa, M.J., Blum, C., Roli, A., Sampels, M. (eds.) *HM 2005. LNCS*, vol. 3636, pp. 1–11. Springer, Heidelberg (2005)
- [9] Dorigo, M.: Optimization, learning and natural algorithms. PhD thesis, DEI, Politecnico di Milano, Italy [in Italian] (1992)
- [10] Dorigo, M., Gambardella, L.M.: Ant colony system: A cooperative learning approach to the traveling salesman problem. *IEEE Transactions on Evolutionary Computation* 1(1), 53–66 (1997)
- [11] Dorigo, M., Stützle, T.: *Ant Colony Optimization*. MIT Press, Cambridge (2004)
- [12] Dorigo, M., Maniezzo, V., Colomi, A.: The Ant System: optimization by a colony of co-operating agents. *IEEE Transactions on Systems, Man, and Cybernetics Part B: Cybernetics* 26(1), 29–41 (1996)
- [13] Dorigo, M., Bonabeau, E., Theraulaz, G.: *Swarm Intelligence: From natural to artificial systems*. Oxford University Press, New York (1999)
- [14] Dorigo, M., Birattari, M., Stutzle, T.: Ant colony optimization: artificial ants as a computational intelligence technique. *IEEE Comput. Intell. Mag.* 1(4), 28–39 (2006), <http://dx.doi.org/10.1109/CI-M.2006.248054>
- [15] Duan, J.C.: The GARCH option pricing model. *Mathematical Finance* 5, 13–32 (1995)
- [16] Duan, J.C.: Term structure and bond option pricing under GARCH. McGill University (unpublished manuscript) (1996), citeseer.ist.psu.edu/duan96term.html
- [17] Engle, R.: Autoregressive Conditional Heteroskedasticity with estimates of the variance of U.K. inflation. *Econometrica* 50(9), 987–1008 (1982)
- [18] Gambardella, L.M., Dorigo, M.: Solving symmetric and asymmetric TSPs by ant colonies. In: *International Conference on Evolutionary Computation, Nayoya University, Japan*, pp. 622–627 (1996)
- [19] Gropp, W., Lusk, A., Skjellum, A.: *USING MPI: Portable Parallel Programming with the Message-Passing Interface*. MIT Press, Cambridge (1994)
- [20] Gunes, M., Sorges, U., Bouazzi, I.: ARA – the ant-colony based routing algorithm for MANETs. In: *Proceedings of the international conference on parallel processing workshops (ICPPW 2002)*, Vancouver, B.C., pp. 79–85 (2002)
- [21] Huang, K.: A parallel algorithm to price Asian options with multi-dimensional assets. Master's thesis, Department of Computer Science, University of Manitoba, Winnipeg, MB, CA (2005)

- [22] Huang, K., Thulasiram, R.K.: Parallel algorithm for pricing American Asian options with multi-dimensional assets. In: Proc. (CD-RoM) 19th Intl. Symp. High Performance Computing Systems and Applications (HPCS), Guelph, ON, Canada, pp. 177–185 (May 2005)
- [23] Hull, J.C.: Options, futures, and other derivative securities. Prentice-Hall, Englewood Cliffs (2006)
- [24] Keber, C., Schuster, M.G.: Generalized ant programming in option pricing: Determining implied volatilities based on American put options. In: Proceedings of the IEEE International Conference on Computational Intelligence for Financial Engineering, Hong Kong Convention and Exhibition Centre, Hong Kong, pp. 123–130 (March 2003)
- [25] Koza, J.R.: Genetic Programming: On the Programming of Computers by Means of Natural Selection. MIT Press, Cambridge (1992)
- [26] Merton, R.C.: Theory of rational option pricing. *Bell Journal of Economics* 4, 141–183 (1973)
- [27] Prisman, E.Z.: Pricing Derivative Securities: An Interactive Dynamic Environment with Maple V and MATLAB with Cdrom. Morgan Kaufmann Publishers Inc., San Francisco (2000)
- [28] Rahmail, S., Shiller, I., Thulasiram, R.K.: Different estimators of the underlying asset's volatility and option pricing errors: parallel Monte-Carlo simulation. In: Proceedings of the International Conference on Computational Finance and its Applications (ICCF), Bologna, Italy, pp. 121–131 (2004)
- [29] Randall, M., Lewis, A.: A parallel implementation of ant colony optimization. *Journal of Parallel and Distributed Computing* 62(9), 1421–1432 (2002), <http://dx.doi.org/10.1006/jpdc.2002.1854>
- [30] Roth, M., Wicker, S.: Termite: ad-hoc networking with stigmergy. In: Proceedings of IEEE Global Telecommunications Conference (Globecom 2003), San Francisco, USA, pp. 2937–2941 (2003)
- [31] Seeley, T.D.: The Wisdom of the Hive: The Social Physiology of Honey Bee Colonies. Harvard University Press, Cambridge (1995)
- [32] Thulasiram, R.K., Bondarenko, D.: Performance evaluation of parallel algorithms for pricing multidimensional financial derivatives. In: IEEE Computer Society Proceedings of the Fourth International Workshop on High Performance Scientific and Engineering Computing with Applications, Vancouver, BC, Canada, pp. 306–313 (2002)
- [33] Thulasiram, R.K., Litov, L., Nojumi, H., Downing, C., Gao, G.: Multithreaded algorithms for pricing a class of complex options. In: Proceedings (CD-RoM) of the IEEE/ACM International Parallel and Distributed Processing Symposium (IPDPS), San Francisco, CA (2001)
- [34] Wedde, H.F., Farooq, M., Pannenbaecker, T., Vogel, B., Mueller, C., Meth, J., Jeruschkat, R.: BeeAdHOC: An energy efficient routing algorithm for mobile ad hoc networks inspired by bee behavior. In: Proceedings of Genetic and Evolutionary Computation Conference, Washington, DC, pp. 153–160 (2005)

A Neuro-Evolutionary Approach for Interest Rate Modelling

Robert Bradley^{1,2}, Anthony Brabazon^{1,2}, and Michael O'Neill^{1,3}

¹ Natural Computing Research and Applications Group,
Complex & Adaptive Systems Laboratory
University College Dublin, Ireland

robert.bradley@ucdconnect.ie, anthony.brabazon@ucd.ie, m.oneill@ucd.ie

² School of Business, University College Dublin, Ireland

³ School of Computer Science and Informatics, University College Dublin, Ireland

Summary. This chapter presents a novel application of a neuro-evolutionary methodology for the purposes of intraday trading of German government bond futures. This market is very liquid, is the focus of substantial trading activity, and hence, quality trading systems for this market have considerable practical implication. To date, few studies have examined the potential utility of computational intelligence methodologies for the purpose of trading on this market and none have adopted a neuro-evolutionary approach. Our results suggest that structure in the time series of bond futures prices can be uncovered and that this information can be used to trade with some success. A number of future extensions of this study are also indicated.

5.1 Introduction

Bond markets play a significant role in capital allocation in developed economies. As an illustration of the scale of these markets, the total value of outstanding marketable bond debt in the US was approximately \$29.2 trillion [1] as at the 30 September 2007. In comparison, the total amount of marketable equity of all US companies on the same date was approximately \$22.3 trillion [1]. Multiple issuers use bond markets to raise funding including companies, financial institutions, government agencies and central government. Given the scale and liquidity of bond markets they attract substantial trading interest.

Germany is regarded as a stable and powerful economy making German government debt an attractive investment. The demand for this debt has been particularly evident in recent times as the *sub prime* financial crisis has seen portfolio managers move capital into the relative safety of the highest quality bonds such as those issued by the German government (a “*flight to quality*”). The demand for German government bonds carries through to the Eurex derivatives market where the *Euro-Schatz*, *Euro-Bobl*, and *Euro-Bund* German Bond Futures are the most heavily traded fixed income futures in the world.

The construction of an intraday trading system for bond futures is a difficult task as the time series of prices from this market is quite volatile (see Fig. 5.5) and we do not

have strong theory to describe the exact nature of the price-generation process. This suggests that a model-induction methodology such as a multi-layer perceptron may have utility in uncovering this process from the underlying data. In this chapter, we test whether this is in fact the case and assess the utility of the resulting model by examining whether it can be used to successfully trade bond futures.

A practical difficulty in developing a multi-layer perceptron is that the creation of a quality network for a specific application can be time-consuming. Hence, there has been significant interest in the possibility of automating some or all of this development process by means of evolving neural net structures. A key issue in doing this efficiently is the matching of the design of the diversity-generating operator(s) to the choice of representation for the neural network. This is not a trivial task and a diverse range of approaches have been suggested. In this study we adopt the NEAT methodology [4–6] which adopts a principled approach to this issue.

5.1.1 Structure of Chapter

The rest of this chapter is organised as follows. The next section provides a concise overview of the bond futures market followed by a short introduction to the NEAT methodology. We then outline the experimental methodology adopted and the results obtained. We conclude this chapter by suggesting a number of avenues for future work.

5.2 Trading Bond Futures

A futures contract is a standardised agreement between two counter-parties where the buyer (seller) agrees to take delivery of (deliver) a specific quantity of the *underlying* (for example, a financial security, a foreign currency or a commodity etc.) on a specific date in the future (called the *maturity date*), for a price agreed upon now. Futures can, for example, be used by producers and consumers in order to hedge their respective future income and exposures by providing assurance as to the future price they will receive/pay for some item. This is essential for businesses when trying to manage projected cash flows and associated risks. Traders in financial markets can also use futures to hedge certain exposures in their portfolio. For example, a Fixed Income dealer might want to protect a portfolio of government bonds from adverse changes in interest rates by purchasing, or selling, bond futures. Futures can also be used by speculators who want to express a view on the direction of the market. In the case of fixed income futures the underlying is a fixed income product such as a government bond. The future's market price is quoted in the same way as the underlying bond, i.e., as a percentage of the face value of the bond.

German Government Bond Futures

In this chapter we look at three particular fixed income futures which derive their value from German government bonds and which are traded on Eurex; the *Euro-Schatz* (FGBS), *Euro-Bobl* (FGBM), and *Euro-Bund* (FGBL) Bond Futures. All three futures trade the next 4 maturities in a quarterly (March, June, September, December) delivery cycle and expire on the 10th day of the delivery month. At maturity of the future, the party which is required to deliver the bond can opt to deliver *any* German federal bond

Table 5.1. Eurex Bond Futures

Contract	Nominal value	Tick	Maturity Window	Coupon (%)	Eurex code
Euro-Schatz	EUR 100,000	0.005	1.75 to 2.25 years	6.00	FGBS
Euro-Bobl	EUR 100,000	0.005	4.5 to 5.5 years	6.00	FGBM
Euro-Bund	EUR 100,000	0.010	8.5 to 10.5 years	6.00	FGBL
Euro-Buxl	EUR 100,000	0.020	24 to 35 years	6.00	FGBX

which satisfies certain constraints. For example, the holder of a short Euro-Schatz position at maturity must deliver a German bond which has a remaining time to maturity in the range 1.75 to 2.25 years.

Table 5.1 shows the contract specifications for four German bond futures. The nominal value is the sum that the holder of the bond redeems when the bond matures. The tick size is the minimum move a bond future price can make in the market. The Euro-Bund trades full ticks, where a tick is a one basis point move. For example, a one tick up in the Euro-Bund, from 99.65 to 99.66 for example, is worth 10 euro per contract, which is calculated by multiplying the nominal value by the move, which is $.01 \times 100,000$ in this case. The Euro-Schatz, and Euro-Bobl futures are priced similarly, although they move in half ticks, with a value of 5 euro per future, i.e., $.005 \times 100,000$. When a bond is issued a coupon rate is decided upon based on the market rates at the time of issue. This coupon rate decides on the percentage of the nominal value that is to be paid annually to the holder on the bond.

The bond futures price quoted in the market is called the *clean price* and trades at a percentage of the face value of the contract. This clean price does not include accrued interest from the last coupon payment. The future is said to be trading at par value if it is at 100, at a premium if the price is above 100, and at a discount if it is below. This price quotation system is the same as that used in the underlying cash bond market.

Deliverable Bonds

At the maturity of a futures contract any bond which has a remaining time to maturity which falls inside the maturity window as specified in Table 5.1 is deliverable. During the lifetime of the future, the party on the short side of the position must disclose the basket of deliverable bonds which satisfy the aforementioned constraints. When trading on Eurex your counter-party is the Eurex Clearing House and the Eurex website maintains a list of deliverable bonds for all maturities currently being traded. The components of these baskets depend on various parameters and can change during the future's lifetime. At delivery the holder of the short side of the position can choose which of the deliverable bonds to deliver. As expected, they will choose the bond from the set of deliverable bonds which is cheapest for them to deliver. The actual delivery price is calculated by multiplying the final settlement price of the future by a conversion factor (described below) and adding any accrued interest since the last coupon payment on the underlying bond.

The components of the basket of deliverable bonds may, and usually will, have different values for their respective parameters including coupon rate and time to maturity. As the Eurex Bond Futures are based on a notional bond with a 6% coupon, a calculation is

Table 5.2. Euro-Schatz deliverable bonds

ISIN	Coupon (%)	Maturity Date	Conversion Factor
DE0001137230	4.00	10/09/2010	0.967456
DE0001141471	2.50	08/10/2010	0.940976
DE0001135168	5.25	04/01/2011	0.985709

necessary to adjust the futures price to correct this. Eurex displays a conversion factor for each deliverable bond. Table 5.2 shows the deliverable bonds for the Euro-Schatz December 08 expiry as of 23/09/2008, including their conversion factors.

The theoretical future price (TFP) is calculated as the underlying bonds market price, plus the financing costs, minus the income received on the cash bond position (Eq. 5.1)

$$TFP = \frac{1}{CF} \left[C_t + \left(C_t + c \frac{t-t_0}{365} \right) \times {}_t r_c \times \frac{T-t}{360} - c \times \frac{T-t}{365} \right] \quad (5.1)$$

where CF is the conversion factor, C_t is the market price of the bond, c is the coupon rate, $T-t$ is the time to expiry on the futures contract, t_0 is the coupon date, t is the value date, and ${}_t r_c$ is the 12 month Euribor money market rate.

An Example

The following demonstrates the calculation of the theoretical future price for the Euro-Schatz December 08 expiring on Monday 22nd September 2008 (data is taken from Table 5.3).

$$TFP = \frac{1}{0.967456} \left[100.03 + (100.03 + 0.13) \times 0.0544 \times \frac{79}{360} - 4.00 \times \frac{79}{365} \right] \quad (5.2)$$

$$= 103.386$$

At maturity of the future contract the actual delivery price is calculated as follows

$$\begin{aligned} \text{Delivery price} &= \text{Future's Final Settlement Price} \\ &\times \text{Conversion factor} + \text{accrued interest} \end{aligned} \quad (5.3)$$

The final delivery price is calculated for each bond in the basket of deliverable bonds. The conversion factor makes unrealistic assumptions about the shape of the yield curve,

Table 5.3. Theoretical future price example

Value date	22/09/2008
Bond	4.00% coupon, maturity 10/09/2010
Bond price	100.03
Future delivery date	10/12/2008
Accrued interest	$4.00 \times (12/365) = 0.13$
Conversion factor	0.967456
Euribor 12m	5.44%

and this creates a bias toward certain bonds in the basket of deliverable bonds. At maturity the deliverable bond which is Cheapest To Deliver (CTD) is actually delivered.

5.3 NEAT

In this study we employ the neuro-evolution of augmenting topologies (NEAT) methodology to evolve multi-layer perceptrons (MLP) for the purposes of developing a trading system for the German bond market. NEAT was developed in 2002 by Stanley and Miikkulainen [4–6] and attempts to overcome the problems of evolving MLPs using a direct encoding of an MLP structure. NEAT simultaneously evolves both MLP topology and weights. Proponents of NEAT claim that it

1. applies a principled method of crossover (i.e., attempts to overcome the permutation problem),
2. protects structural innovations (i.e., attempts to overcome noisy fitness problem), and
3. grows MLPs from a minimal structure (i.e., attempts to ensure that final MLP is no more structurally complex than necessary).

In order to achieve this NEAT uses three mechanisms, a novel MLP encoding which allows for ‘sensible crossover’, a speciation concept which offers some protection for structural innovations, and a seeding process whereby all initial MLP structures are of minimal size. Algorithm 5.1 provides an overview of NEAT. The workings of the algorithm are described in more detail in the following sections.

Algorithm 5.1. NEAT Algorithm

```

Generate an initial population of  $n$  (identical) genotypes of minimal structure;
Decode one genotype into a MLP structure and assess fitness of all members of the
population;
repeat
    Select a random member of each species to represent that species;
    for Each genotype in turn do
        | Calculate shared fitness of genotype;
    end
    for Each species in turn do
        Calculate number of members of that species in the next generation;
        Insert best current member of that species automatically into the next generation;
        Select best  $x\%$  of each species for mating pool;
        repeat
            Randomly select parents from the mating pool and apply crossover to obtain a
            child;
            Apply mutation to newly-created child;
            Assign new child to an existing or a new species;
        until required number of children are generated ;
    end
until terminating condition ;

```

5.3.1 Representation in NEAT

The genetic encoding used in NEAT is illustrated in Fig. 5.1. Each gene encodes an individual connection between two nodes, therefore the genotype encodes an entire MLP structure and associated weights. Each *connection gene* has four pieces of information, the index number of the input and output nodes for that connection, the connection's real-valued weight, an indicator as to whether that connection is 'enabled' (on) or 'disabled' (off), and the *innovation number* for the connection. Innovation numbers are assigned in sequence when a new connection is first created in the population of genotypes (when two nodes are linked for the first time), and *all* subsequent instances of that connection in the population have the same innovation number. Hence, a connection gene between (say) node 1 and 4, will have the same innovation number for all members of the population of genotypes. The innovation number helps ensure that genotypes can be lined up sensibly during crossover.

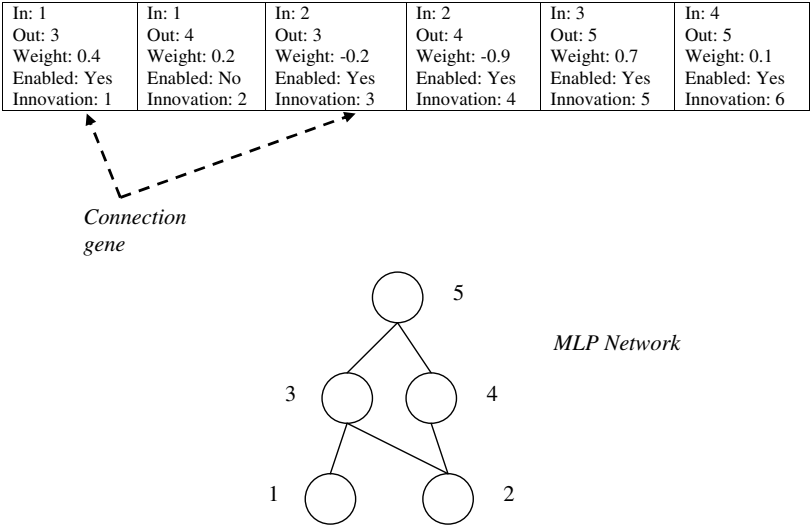


Fig. 5.1. A genotype in NEAT. This genotype has six connection genes (one of which is disabled) and encodes an MLP with two hidden layer nodes.

5.3.2 Diversity Generation in NEAT

As in canonical ECs, new generations of genotypes are created using a select, reproduce and replace cycle. Each of these processes are described in the following sections.

Crossover in NEAT

The crossover operation in NEAT is based on the idea that nature only allows sensible, not random, crossover during reproduction. In contrast to the random crossover process in primitive neuro-evolutionary algorithms, NEAT encodes and uses information on the

genotype's historical development (via its innovation numbers) in order to allow proper pairing of parent genotypes for crossover of their MLP structures.

Each connection gene representing a connection between two specified nodes will have the same innovation number for all genotypes in the population. Hence, genotypes can be lined up, and their common connection genes identified. The parents may also have disjoint and / or excess genes (genes which occur either inside or outside the overlapping range of the two parent's sets of innovation numbers), where one parent has a

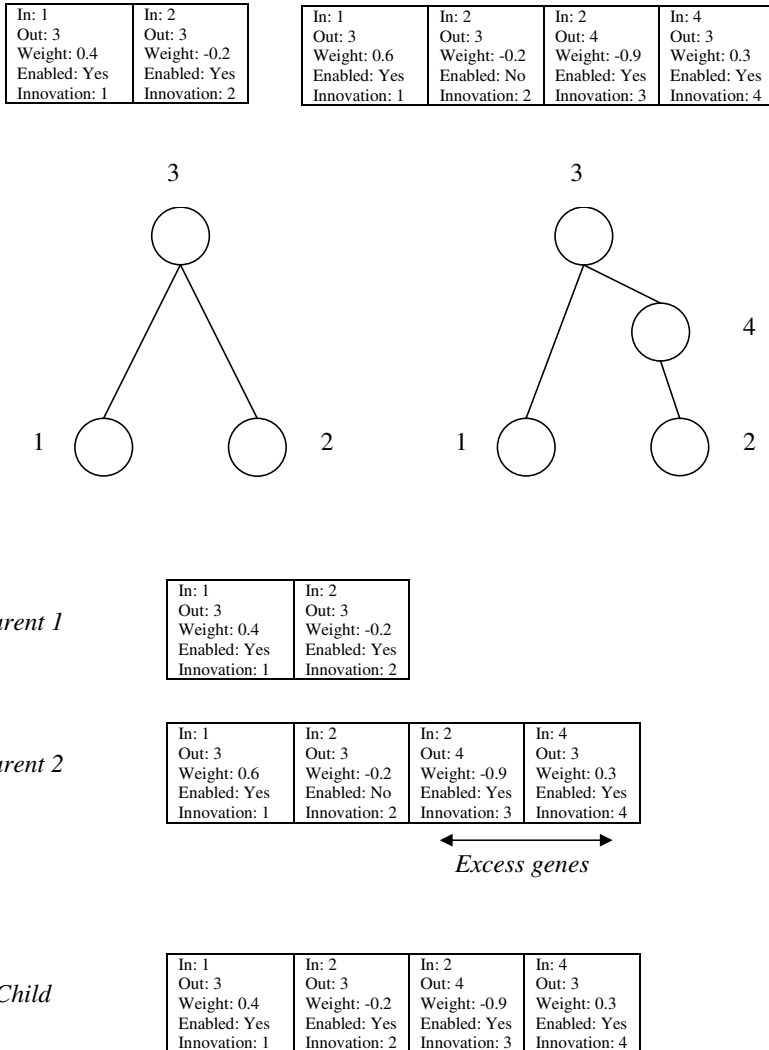


Fig. 5.2. Crossover in NEAT. Parent 2 is assumed to be the fitter, hence all its excess genes are inherited by the child. Stochastically, the connection between node 2 and 3 is turned on in the child, despite it being disabled in parent 2.

connection gene which the other does not. Fig. 5.2 provides an illustration of the matching process highlighting matching and excess genes between two parent genotypes.

During the crossover process, the child genotype inherits all connection genes which are common to both parents with the weight value being randomly selected from one of the parents. Disjoint and excess genes are inherited from the fitter parent. If a gene is enabled in one parent and disabled in the other, it is stochastically enabled/disabled in the child. A relatively strong selection pressure is applied in NEAT whereby the top 40% in each species (see Sect. 5.3.3) are placed in a mating pool with random selection from this pool in order to select parents. In generating children, a 25% cloning and a 75% crossover rate is suggested by [4].

One item of note in the operation of crossover in NEAT is that it does not usually play as significant a role as mutation in generating structural diversity in the population. As children inherit all the disjoint and excess genes from the fitter parent, their basic structure will resemble that of the fitter parent. Hence, crossover primarily acts as a search of weight space around fitter parent. In contrast, crossover in “traditional” neuro-evolution, while running into the issue of competing conventions, does generate substantial structural diversity (akin to macro-mutation). Consequently, the true importance of crossover in NEAT is unclear as mutation does much of the work in generating structural diversity.

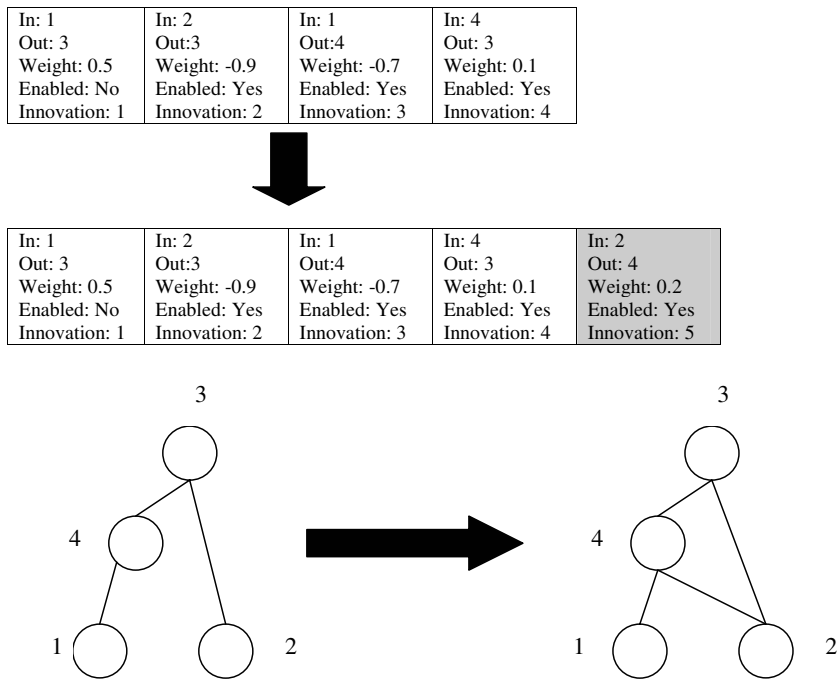


Fig. 5.3. An add connection mutation. Here a new connection is added between nodes 2 and 3 and a corresponding connection gene is inserted in the genotype.

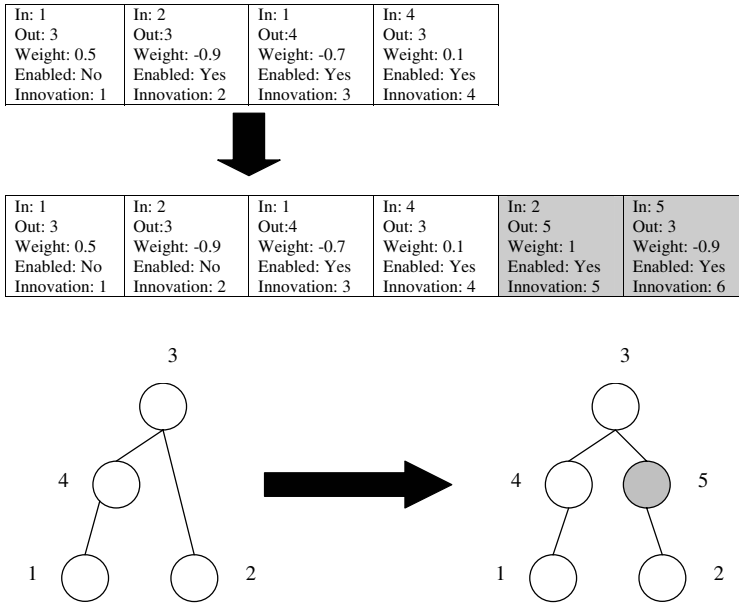


Fig. 5.4. An add node mutation. Here a new node is inserted between nodes 2 and 4 resulting in the addition of two new connection genes to the genotype.

Mutation in NEAT

Three types of mutation are possible in NEAT, mutation of a real-valued connection weight, an ‘add connection’ mutation and an ‘add node’ mutation. In an add connection mutation, a new connection gene is appended to a genotype which creates a connection between two existing nodes in a network. The weight for this new connection is generated randomly (Fig. 5.3). In the case of an add node mutation, a randomly selected existing connection is broken and a new node is placed at the break point. The connection into this new node is given a weight of 1 and the connection out of the node retains the old weight from the broken connection (Fig. 5.4). Typically, the weight mutation rate is higher than that for connection or node additions. Over time, the mutation process will tend to produce longer genotypes and hence more complex MLP structures.

5.3.3 Speciation

In evolving MLP structures, newly created topologies will often have limited fitness as their connection weights are not optimised when the new structure is initially created. Hence the new structures may be promptly deselected before they have a chance to optimise.

To overcome this problem, NEAT uses speciation [3] in order to help protect new structures. The concept is similar to that of speciation in nature where species compete in different ecological niches and the most direct competition for resources is between members of the same species. In NEAT speciation is undertaken using the structural

information contained in each genotype. The object is to determine which genotypes are most “similar” and therefore should be considered to belong to the same species. The degree of similarity between two genotypes is determined by looking at their connection genes, how many genes match, how many excess and disjoint genes are there, and what is the degree of similarity in the connection weights on the matching genes? A metric is calculated using

$$\frac{c_1 E}{N} + \frac{c_2 D}{N} + c_3 \overline{W} \quad (5.4)$$

where N is the number of genes in larger of two chromosomes being compared, E and D are the number of excess and disjoint genes respectively, \overline{W} is average weight differences of matching genes, and c_1, c_2, c_3 are the relative weights on each element of the metric.

When a new child genotype is created, the value of the above metric is calculated by comparing the similarity of the new genotype to a randomly chosen member of each species in the last generation of the NEAT algorithm. The new child is then assigned to the first species where the calculated distance is within a predetermined threshold value δ . If no species is found to be within this threshold distance, the genotype finds a new species.

Fitness-sharing

In the selection step in NEAT, a fitness-sharing mechanism is used in order to encourage diversity in the population of genotypes. The use of fitness-sharing is driven by the observation that individuals within a species compete for the same resources and each species occupies a niche in the wider ecological environment. Shared fitness is calculated using

$$f'(i) = \frac{f(i)}{\sum_{j=1}^n s(d(i, j))} \quad (5.5)$$

where $f(i)$ represents the original (unadjusted) fitness of genotype i and $f'(i)$ represents the shared (reduced) fitness of genotype i . The *sharing function* s provides a measure of the *density* of other species members within a given neighbourhood of a specific genotype i . For any pair of genotypes (i, j) , the sharing function returns a value of ‘0’ if (i, j) are more than a specified distance (t') apart. In NEAT a simple neighbourhood definition is used, being the number of members of the species to which genotype i belongs. Hence, the shared fitness of a species member is its original fitness divided by the population size of that species. The use of fitness-sharing makes it difficult for any species to “take over” the population and helps protect and promote structural diversity.

The size of individual species alter over time depending on whether the adjusted fitness of individuals in that species are higher or lower than the populational average fitness. The number of individuals in each species changes from one generation to the next according to

$$N'_j = \frac{\sum_{i=1}^{N_j} f_{ij}}{\overline{f}} \quad (5.6)$$

where N'_j and N_j are the new and old number of individuals in species j , f_{ij} is the adjusted fitness of individual i in species j and \bar{f} is the average adjusted fitness of the entire population. An elitist selection process is also used whereby the best individual in each species survives into the next generation, once the population has at least five members. Species can become extinct, either where the the number of new individuals in the species falls below 1, or if there is no change in the fitness of the best member of the species over multiple generations.

5.3.4 Incremental Evolution

NEAT begins with a uniform population of networks which have no hidden layer nodes and where all inputs are connected to the output node(s). Additional complexity in the form of hidden layer nodes or new connections is introduced as necessary via structural mutations. This approach implies that in earlier generations of the evolutionary process, search and optimisation is performed on small MLP structures which makes weight optimisation within those structures easier.

Of course, a variety of other methods of initialisation could be used and the above approach will need to be altered for cases where the problem has a large number of input and/or output nodes. For example, using the above approach, a problem with 40 inputs and 10 output nodes would require 400 connections, resulting in a high-dimensional search space. In contrast, initialisation of the population using say a hidden layer with 3 nodes would reduce the number of connections to 150.

5.4 Experiments

In this section, we describe the experimental methodology employed in our study. Initially we describe the dataset and explore some features of the data.

5.4.1 Data

The dataset used in this study consists of 5,000 five minute bars of intraday data for the three German bond futures mentioned in Sect. 5.2; the Euro-Schatz, Euro-Bobl, and Euro-Bund. A bar contains a value for the open, high, low, and close prices for the interval, and also the number of contracts traded in the 5 minute period (i.e. volume).

Fig. 5.5 shows the time series of closing prices for the three futures. The series exhibit a variety of price behaviours, including bullish, bearish, and choppy periods. This varied behaviour poses a difficult learning environment for NEAT.

The Cheapest To Deliver (CTD) bonds for the three futures have varying values for coupon rate, time to maturity, and price. To compare the return on these bonds we can calculate the yield to maturity (YTM) for each bond, which gives the percentage return the holder would receive if (s)he held the bond until maturity. The yield curve (cash curve), which is a plot of the YTM values, can be plotted against the yield curve derived from the three futures. The future yield curve can be seen as the market's estimate as to where the cash curve will be at maturity of the future contracts. Fig. 5.6 illustrates the relationship between the yield curve derived from the CTD bonds and the bond futures.

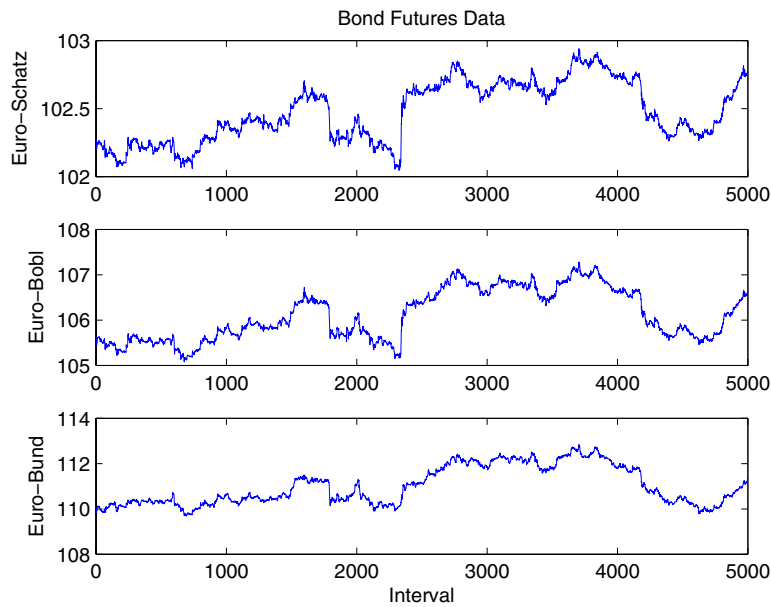


Fig. 5.5. Time series of closing futures price for each bond for each five-minute data bar

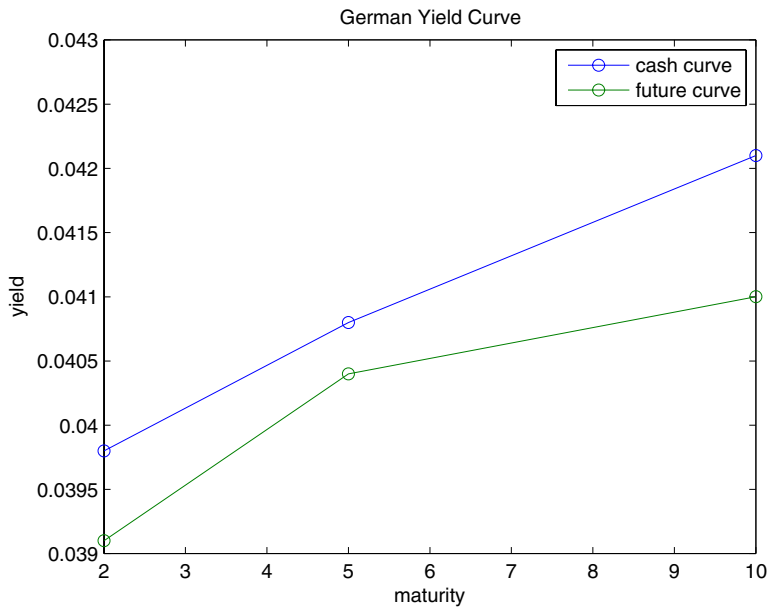


Fig. 5.6. Yield Curve for CTD bonds and bond futures

Table 5.4. Volume statistics measured in number of contracts traded per 5 min block

	Schatz	Bobl	Bund
Total	15,186,286	14,432,952	24,057,734
Mean	3,037	2,887	4,812
Std	3,817	3,146	5,367
Max	59,738	40,240	57,818
Min	1	1	1

Table 5.5. First difference of closing price in five minute blocks

	Schatz	Bobl	Bund
Mean	0.0099	0.0208	0.0274
Std. Dev	1.1825	2.6142	3.5879
Max Up	11.5000	24.5000	22.0000
Max Down	-8.5000	-24.0000	-34.0000

It is evident from the graph that the futures market is expecting yields to fall before the maturity of the futures contracts.

The bond futures market is one of the most active fixed income markets. Table 5.4 shows a number of basic statistics on the volume of activity (volume of contracts traded) on the market for the period 13/06/2008 to 25/07/2008. The Bund tends to be more volatile, and more heavily traded, than the Schatz and Bobl. Total volume for the given period of trading in the Schatz is over 15 million contracts, compared to the Bund where over 24 million contracts were traded. The average number of contracts per trade ranges from 2,887 (Bobl) to 4,812 (Bund). Each of these corresponds to a bond with a face value of Euro 100,000 and hence represents a substantial ‘gross’ position in the underlying (bond) instrument.

A move in a bond future price from 99.31 to 99.32 corresponds to a full ‘tick’. As already noted, the Bund moves in full ticks, but the Schatz and Bobl contracts move in half ticks. Table 5.5 describes the behavior of the first differences of the closing price data. The volatility varies across contracts, with the Bund being the most volatile with an average move of 0.274 of a tick between each 5 minute block. As can be seen, the standard deviation of the number of tick changes between five minute blocks is quite high relative to the mean, illustrating the volatile nature of price changes in even short time periods for these futures.

5.4.2 Experimental Parameters

In setting the parameters for the NEAT system used to generate the MLPs, we considered parameter settings reported in prior applications of NEAT and supplemented this with some trial and error experimentation. We have not attempted to optimise the parameter settings but experimentation indicated that the results obtained were not hypersensitive to small changes in these settings.

Table 5.6 lists the key parameter settings we employed. The elitism proportion parameter was set to 20% which means that the top 20% of the population are passed on

Table 5.6. Experimental Parameters

Parameter	Value
Pop size	500
Mutate weights	.1
Add neuron	.01
Add connection	.05
Crossover	.9
Elitism Proportion	.2
Min species threshold	6
Max species threshold	10
Complexity threshold	100

to the next generation without being altered by genetic operators. This ensures that the population does not lose high-performing individuals from one generation to the next. Speciation, as described in section 5.3.3, allows networks to evolve within a sub population. This ensures that offspring are given a chance to optimise without being killed off prematurely by more mature individuals. The “min/max species threshold” settings allow the user to keep the number of species within a certain range. The upper limit was set to 10 and the lower limit was set to 6.

As the population evolves the number of neurons and connections in the average network increases according to the mutation probabilities in Table 5.6. Over time the structural complexity of MLPs in the population can increase to a level which results in significant computational overhead. To combat this problem the average level of structural complexity is tracked in the population, and once it exceeds a predetermined threshold, a pruning process is initiated. Pruning reduces the population’s average complexity until it falls below the specified threshold (100 in our case). This is achieved by randomly removing nodes and edges from more complex individuals.

5.4.3 Trading Simulator

In this study we concentrate on inputs which are developed from the time series of the futures’ prices. A total of 15 inputs are available for use by the MLPs being evolved. These are the open, high, low, close, and volume first differences (between two succeeding five minute blocks) for each of the three bond futures.

The networks are all initialised with a bias node and one randomly chosen input from the 15 available, connected to a single output node, as in Fig. 5.7. Thus, the population of MLPs are initialised with minimal complexity. This leaves evolution to decide which of the other inputs should be switched on and the number of hidden nodes that should be added.

The evolved MLPs output a value between 0 and 1 which is post-processed using

$$y = \begin{cases} 1 & \text{if } a \geq 0.6 \\ 0 & \text{if } 0.3 < a < 0.6 \\ -1 & \text{if } a \leq 0.3 \end{cases} \quad (5.7)$$

where y is the final network output and a is the MLP output before postprocessing. The resulting value of y is then input to the trading simulator. The trading simulator is used to evaluate the performance of a network over a given dataset. The simulator accepts three signals; buy (1), sell (-1), and do nothing (0). At any point in time the system is either long or short 1 future contract. If the trade position is long and we get a sell signal, the long position is closed out at the current market price, and a short position is initiated. Conversely, if we are short and get a buy signal we close out the short position and initiate a long position. This behaviour results in a simple trading strategy which is always in the market. The maximum position is limited to a single future to make post trade analysis of results easier. Table 5.7 outlines the logic of this signalling system.

The trading simulator records the state of the system at the end of each five minute interval. A number of state variables are recorded including the realised profit and loss,

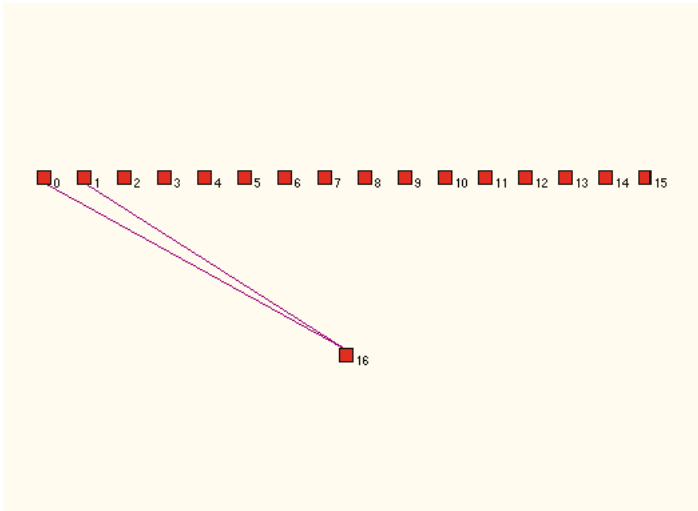


Fig. 5.7. Initial basic network

Table 5.7. Trading Signal Logic

Signal	Previous State	New State
0	Flat	Flat
0	Long	Long
0	Short	Short
1	Flat	Long
1	Long	Long
1	Short	Long
-1	Flat	Short
-1	Long	Short
-1	Short	Short

the position (long/short), and any trade executed. Upon reaching the end of the dataset the simulator prints the state vector to a csv file for further analysis.

In training the MLPs, a moving window approach is adopted. This allows the population of MLPs to adapt over time, allowing the uncovering of new relationships in the data (i.e. the data generating process is not assumed to be static). The moving window is implemented as follows. The population is trained for G generations on the first W observations of the dataset. The best individual is then tested out of sample on the next S observations. The training window of size W is then incremented by S . The population is then trained for $g < G$ generations which gives the population a chance to incorporate the new data. The best individual is tested out of sample on the next S intervals, and the window is moved forward by S again. This process is repeated until the moving window reaches $D - S$, where D is the size of the dataset. The total number of generations in a run can be calculated using Eq. 5.8

$$Gens = G + \left(\frac{D - W - S}{S} \times g \right) \quad (5.8)$$

where G is the number of generations over the initial training period, D is the size of the dataset, W is the size of the training window, S is the size of the window increments, and g is the number of generations at each window increment. The window retraining system is desirable for a number of reasons. Firstly, each window shift results in a partially new environment which challenges the population's ability to generalise and efficiently incorporate new information. Secondly, the moving window system yields out of sample results for the entire dataset (apart from the initial training period).

5.5 Results

5.5.1 In-sample

Fig. 5.8 shows the in-sample fitness value for the population's average and best networks over the entire training run of 450 generations. A window shift occurs at generations 100, 150, 200, 250, 300, 350, and 400. As expected, mean populational fitness falls each time the window is shifted. However, the population shows an ability to adapt to the new data environment with performance improvements occurring after the window is moved. Thus, NEAT shows promise in a dynamic, financial, environment. Fig. 5.8 also suggests that the evolutionary process discovers a network with good generalisation capabilities early on and only replaces this network a small number of times over the run.

One of the nice features of the NEAT algorithm is gradual complexification of the population of MLPs being evolved. The idea is that it is desirable to limit the search space and exhaust the search for an optimal solution at each level of network complexity before increasing complexity. This results in simpler networks which theoretically should be better at generalising than very complex networks, which may be prone to overfit. Fig. 5.9 illustrates this process. The number of neurons and connections increases steadily for the first 160 generations of the training period. At this point the complexity threshold parameter is breached, triggering the algorithm to switch to pruning mode. This results in a pull back in the average complexity below the threshold

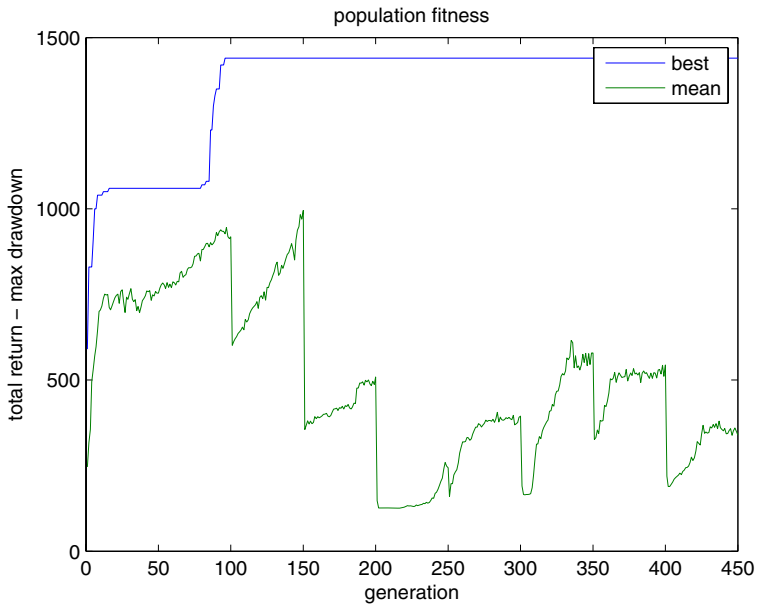


Fig. 5.8. Mean and best populational fitness (in-sample)

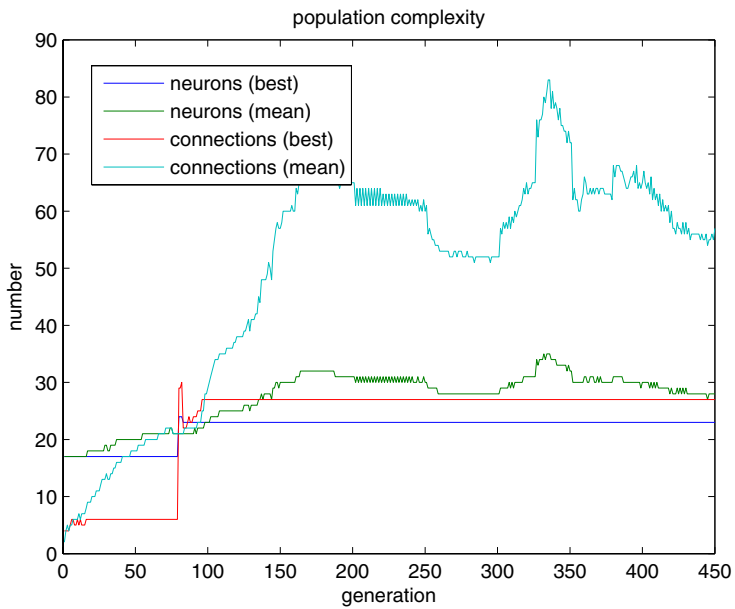


Fig. 5.9. Network complexity

and a return to complexification mode. The threshold is breached again at generation 345, again triggering the pruning process. This system of pruning keeps the average complexity in a tight range.

5.5.2 Out of Sample

The best network, which can change throughout the training period, is tested out of sample on the futures price dataset. Fig. 5.10 shows the performance (including realistic transaction costs) plotted against a random buy/sell strategy. The random series is an average of 30 simulations where a future was randomly bought/sold at each interval.

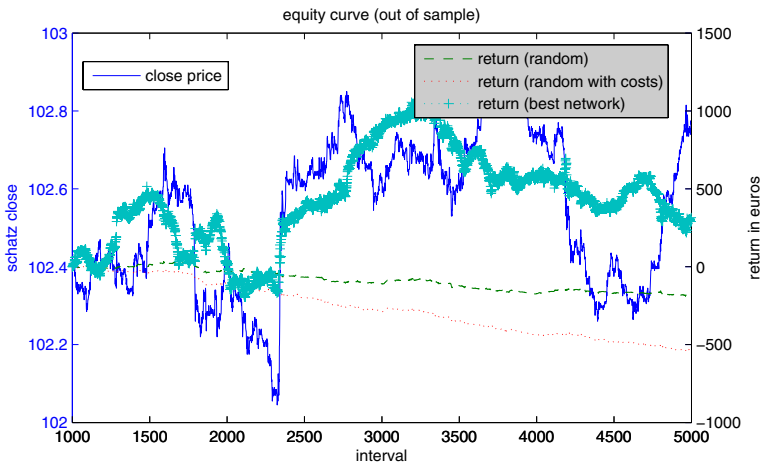


Fig. 5.10. Out of sample equity curve vs equity curve from random trading strategy

5.6 Conclusions

This chapter presents a novel application of a neuro-evolutionary methodology for the purposes of the intraday trading of German government bond futures. To date few studies have examined the potential utility of computational intelligence methodologies for the purpose of trading on this market and none have adopted a neuro-evolutionary approach.

In spite of restricting attention to a very concise set of potential inputs, the results suggest that the resulting model is capable of uncovering structure in the time series of bond futures prices and is capable of using this information to trade with some success. Of course, no conclusive assessment of the utility of this methodology can be made on the basis of the limited experiments we have undertaken in this initial study and we intend to pursue multiple avenues to further extend this work. For example, the current trading simulator only uses a limited range of inputs and its power could be further enhanced by use of established filter rules for price data such as technical indicators. We also note that we adopt a simple trading strategy in this study, whereby the simulator can

only go long or short one contract in response to a buy/sell signal. We intend to refine this in order to allow the system to build up a larger long / short position in response to successive buy/sell signals and also to allow the system to buy/sell varying numbers of contracts depending on the strength of the signal generated by the system. At present, the fitness of the trading system is assessed using a simple return-risk metric (total return - maximum drawdown). Future work will explore the application of a range of fitness metrics in order to examine the impact of choice of fitness metric on the trading characteristics of the resulting system. We also intend to investigate the application of grammar-based GP for trading this market.

References

- [1] Securities Industry and Financial Markets Association - Research Quarterly (November 2007),
<http://www.sifma.org/research/pdf/Research-Quarterly-1107.pdf>
- [2] Brabazon, A., O'Neill, M.: *Biologically Inspired Algorithms for Financial Modelling*. Springer, Berlin (2006)
- [3] Mahfoud, S.: *Niching Methods for Genetic Algorithms*. Unpublished PhD Thesis Department of General Engineering, University of Illinois at Urbana-Champaign (1995)
- [4] Stanley, K., Miikkulainen, R.: Evolving neural networks through augmenting topologies. *Evolutionary Computation* 10(2), 99–127 (2002)
- [5] Stanley, K., Miikkulainen, R.: Efficient evolution of neural network topologies. In: *Proceedings of the 2002 Congress on Evolutionary Computation (CEC 2002)*, pp. 1757–1762. IEEE Press, Los Alamitos (2002)
- [6] Stanley, K., Miikkulainen, R.: Efficient Reinforcement Learning Through Evolving Neural Network Topologies. In: *Proceedings of the 2002 Genetic and Evolutionary Computation Conference (GECCO 2002)*, pp. 569–577. Morgan Kaufmann, San Francisco (2002)

Who's Smart and Who's Lucky? Inferring Trading Strategy, Learning and Adaptation in Financial Markets through Data Mining

Christopher R. Stephens¹, José Luis Gordillo², and Enrique Martínez Miranda³

¹ C₃ - Centro de Ciencias de la Complejidad and Instituto de Ciencias Nucleares
Universidad Nacional Autónoma de México, A. Postal 70-543 México D.F. 04510, Mexico
stephens@nucleares.unam.mx

² Dept. de Supercomputo, DGSCA, Universidad Nacional Autónoma de México,
A. Postal 70-543, México D.F. 04510, Mexico

³ Facultad de Ciencias, Universidad Nacional Autónoma de México,
A. Postal 70-543 México D.F. 04510, Mexico

Summary. Trading “profits” can be obtained by luck or by the implementation of a superior trading strategy. In this chapter we discuss the difficulties of distinguishing between the two. First, a suitable characterization of profit that distinguishes between trading gains and market gains is required. Secondly, one needs to be able to characterize trading “strategies”. To achieve this, we introduce the notion of a genotype-phenotype map to finance, where the genotype is associated with the information set and associated decision rules that lead to a given set of trading decisions for a given trader, while the phenotype is described by the set of observable trading decisions themselves. In AI based systems, such as agent-based markets, a strategy is implemented algorithmically and so the genotype is explicitly known. In real markets however, the genotypic trading strategy of one trader is hidden from the rest. The phenotype however, is, in principle, observable. A microscopic description at the level of the set of individual trades, however, is not sufficient to understand or characterize the strategies at a more macroscopic and intuitive one. By introducing a set of coarse grained variables that can be used to classify strategy types, we show how these variables can then be data mined to understand what differs between an intelligent and a lucky strategy. We show that these variables can be used to distinguish between different strategy types and can be further used to infer the presence of learning and adaptation in the market. We illustrate all of the above using data from an experimental political market.

6.1 Introduction

For many people, including quite a number of scientists, predicting financial markets is a modern day equivalent of the medieval alchemist’s quest to turn base metal into gold. Why do some scientists think they can make money by predicting markets? There is probably some bias in this that stems from thinking of a market as a “mechanical” system, whereby if we only but knew the mechanical laws that govern it then it would be possible to predict its behaviour. In this case it would not matter if the system were stochastic or not, as this just implies that there is an underlying probability distribution that governs the dynamics. Computer scientists as opposed to physicists probably think

of the market as “algorithmic” rather than mechanical. The underlying bias is similar however, in that if we but understood the algorithm we’d be able to predict.

Markets, however, are not mechanical, nor algorithmic, at least not in the sense of a fixed algorithm. Evolution, adaptation and learning are much more appropriate paradigms [1, 2, 3] for financial markets. Indeed, these paradigms underlie the construction of agent-based artificial financial markets [4, 5, 6]. Such markets, as with real markets, are based on the concept of a trading strategy, and in this artificial setting these strategies are represented algorithmically.

The question then is one of whether or not in an ecology of competing strategies there are “winning” ones that preferentially survive as being the fittest. This requires two elements: a notion of a strategy and another of winning. In the context of artificial markets, the strategy element is fairly clear, as only strategies that are algorithmically representable can be entertained. However, this is not the case in a real market. Who knows what strategy the proprietary traders of Goldman-Sachs are using? The other subtlety is what does it mean to make money in a financial market? This might seem like a stupid question at first sight. The question is perhaps better put as: How does one know a strategy is winning through being more intelligent as opposed to just dumb luck?

These issues are, in fact, at the heart of the debate about market efficiency [7, 8] as the Efficient Markets hypothesis implies that in an efficient market there cannot exist trading strategies that make excess profits systematically. This begs the question, of course, of excess relative to what? As we will see, it is, in fact, possible for even zero-intelligence traders to make profits relative to a benchmark such as a risk free interest market. But this is by luck. To distinguish skill from luck it is necessary to be able to distinguish between market gains, those that accrue from general market trends, and trading gains - those due to a particular trading strategy [9] in the market. An empirical measure that achieves that [10, 11] allows one to determine market efficiency empirically by determining if there is any trading strategy that is making profits in excess of a dynamic buy and hold portfolio, i.e., one that gets recalibrated every trade.

So, one may identify a winning strategy if one knows the strategy used in order to be able to follow its performance. As mentioned, unfortunately, in a real market we do not know what strategies are being used. All we could hope to see is the imprint of that strategy in the pattern of buys and sells from a particular trader. This begs the question: Is it possible to characterize a strategy only from this pattern of buys and sells? This is analogous to trying to characterize an organism and/or its survival strategy through the phenotype rather than the genotype and must be done by statistical inference. An appropriate vehicle for such inference is data mining and so, in this chapter, we show how a data mining philosophy can be used to characterize winning trading strategies and build a predictive model based on their “phenotypic” profiles.

The content of this chapter will be as follows: First we will discuss the problem of measuring profits, discussing a measure that makes a clean separation between trading gains and market gains. In Sect. 6.3 we will then discuss the problem of relating profits to a trading strategy, introducing the concept of a genotype-phenotype map into finance. We will also discuss the problem of realizing profits using a particular trading strategy. In Sect. 6.4 we will show how data mining techniques may be used to characterize

trading strategies. We illustrate all these points in the context of a particular experimental market [12]. In Sect. 6.5 we present the results of this study showing first, how the market is inefficient, in that certain traders make systematic, excess profits. We then use data mining techniques to characterize the strategies of these successful traders and then show how these techniques allow one to infer the existence of learning and adaptation in the market. Finally, in Sect. 6.6 we draw some conclusions.

6.2 The Problem of Measuring Profits

The standard way to evaluate if you are making “profits” or not is to measure the value of your portfolio against some benchmark. The idea then is that you are winning if your profits are “excess” relative to this benchmark. A standard one is the risk-free interest rate. Really, what we are doing there is to compare two strategies over a certain period of time. Your actual strategy of buying and selling versus just leaving all your money in a “risk-free” asset such as a treasury bill. A quantitative measure of this is the well known Sharpe ratio

$$S(X) = \frac{(R(X) - R_f)}{\sigma(X)} \quad (6.1)$$

where $R(X)$ is the average return on an investment X , R_f is the risk-free interest rate and $\sigma(X)$ is the standard deviation of the returns. These returns can be measured with respect to different frequencies - daily, weekly, monthly, annually etc. The Sharpe ratio not only takes into account the rate of return of your investment but also how risky it is, as proxied by the volatility of the returns. Most people would accept this measure as being a very fair one.

In Figs. 6.1 and 6.2 we see a subset of trader portfolio values and the price series from a pair of simulated markets. In both cases the economic universe consists of one risky asset and one risk free asset, and that is all. Each market consisted of 20 traders executing trades according to a particular trading strategy. Only the portfolio values of ten randomly chosen traders are shown for legibility. In both cases the net proportion of traders with positive Sharpe ratio, where a risk free annual interest rate of 4% was chosen, is greater than 50%. Clearly, markets, in this sense, are not a zero sum game.

What were the successful, and therefore by implication “intelligent”, trading strategies that were used by the successful agents? What went into these markets? Well, Fig. 6.1 shows the results for a random subgroup of ten traders from a group of 20 “zero-intelligence” traders that flipped a fair coin to decide whether to buy or sell. So, how come they’re all making money and have a positive Sharp ratio? Luck! The coin tossing realises a stochastic process. Over a given period of time, fluctuations in supply and demand and the limit prices set by the traders sometimes cause the market to go up and sometimes to go down. In this particular realisation it went up. In another it might well have gone down. However, a real financial market is a single realisation of a stochastic process. We cannot rerun the market many times over to see if someone who is successful with a given strategy will be successful again given the same conditions. Normally, we try to get around this by considering the time dimension: Are someone’s profits “systematic”? i.e., is the person making excess profits over a long period of time.

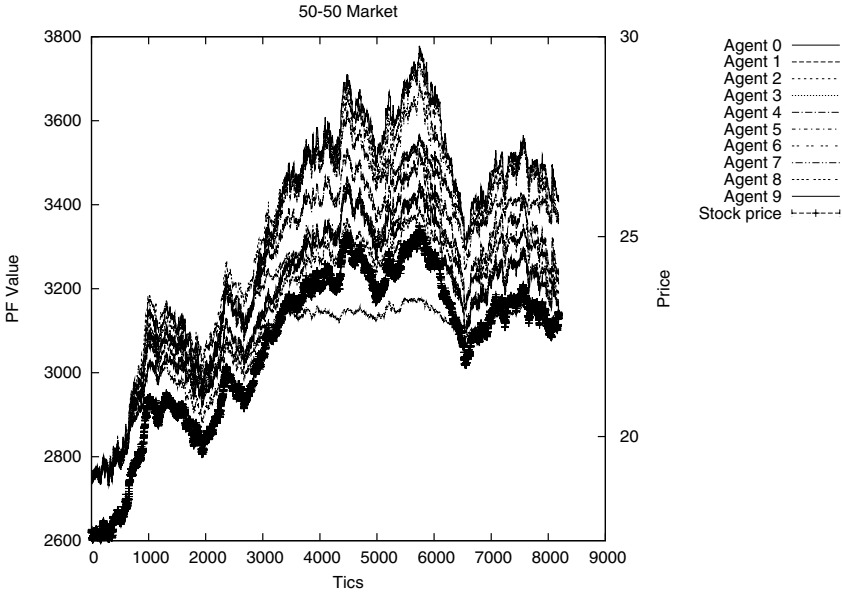


Fig. 6.1. Graph of portfolio value in terms of excess trading profits as a function of time for ten zero-intelligence traders

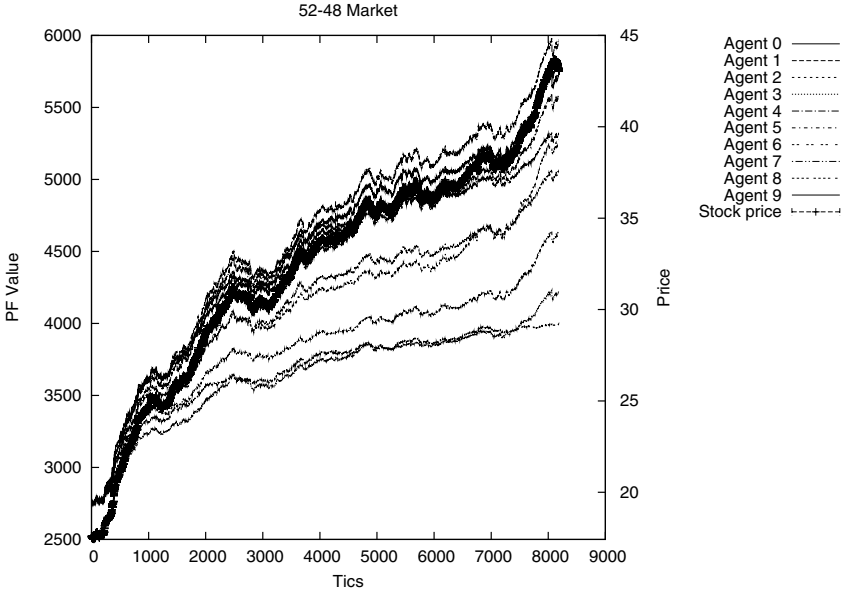


Fig. 6.2. Graph of portfolio value in terms of excess trading profits as a function of time for ten biased traders

For the zero-intelligence traders, if we single out a particular one and follow it in time we will see that the net returns over a long period of time are, indeed, vanishingly small.

So, what about Fig. 6.2? There the traders also tossed a coin, but, this time, a biased one, where the probability to buy was 52% instead of 50%. The same biased coin was used for all. This means that there is an excess demand as more want to buy than sell and so the price tends to go up. Now, *all* of the traders have significant profits with corresponding Sharpe ratios of about 4-5! Fantastic, now a particular trader can think about cashing in his stock after making a handsome profit and head to the shops. However, this is a one asset economic universe. There are no other assets, such as Ferraris, diamonds, Van Goghs etc. In this sense, by trying to cash in their stock the trader is simply penalising himself by changing to an asset (cash) whose value is static while that of the other asset keeps on increasing due to the avid buying of the other traders, and so the trader would lose by cashing in. On the other hand, if the trader keeps on trying to buy with the same probability as the rest then, although on average he won't lose *relative* to the others, neither will he win.

These two examples give one pause in thinking what does it mean to make money. How does one distinguish between making money by luck and making it through an intelligent trading strategy? Also, what sense does it make to say you're winning against a benchmark when you're losing relative to everyone else in the market? In other words, how does one distinguish between trading returns, associated with an intelligent trading strategy, versus market returns, associated with general movement in the market.

So, we can ask: Is there some other measure for making money that overcomes these problems? Yes, it is called "excess trading profit" [10, 11]. The idea is that you're only as good as your last trading decision. In other words, each trading decision is measured relative to a benchmark of not having done anything. Thus, between one trade and another we can ask how much your portfolio value increased due to that particular trading decision relative to not having made that decision. So, for example, at a particular time, t_1 , you buy 5,000 shares of Federal Screw. At any subsequent time you can evaluate how much your portfolio value has increased compared to having done nothing, i.e., "buy and hold". You might think this is standard procedure, to measure relative to a buy and hold strategy. However, here, the buy and hold benchmark is reset every time a trade is carried out. So, if you start off with 10,000 shares at t_0 , buy 5,000 at t_1 and buy another 5,000 at t_2 your buy and hold reference point for the trade at t_2 is 15,000 shares not 10,000. This simple mechanism of measuring profit relative to a moving target when considered in the context of a suitable statistic solves both the above problems of deciding whether a strategy is intelligent or not and also of getting confused between "absolute" and relative profit.

Specifically, one defines the excess trading profit for an agent j using a trading strategy i at time t associated with a transaction at time t' as

$$e_{ij}(t, t') = \Delta p(t, t') \Delta n(t, t') \quad (6.2)$$

where $\Delta n(t, t')$ is the change in portfolio holding due to the transaction and $\Delta p(t, t')$ is the change in price since the transaction. So, unless there is a change in price *and* you change holding there is no excess profit. In this way only active decisions count. The excess trading profit associated with a series of trading decisions is then

$$E_{ij}(t, t') = \sum_{k=t'+1}^{k=t} e_{ij}(k, k-1) \quad (6.3)$$

If several agents, N_j , all use the same trading strategy, then the average profit for that strategy is $E_i(t, t') = (1/N_j) \sum_j E_{ij}(t, t')$.

As excess profits are taken to be a sign of an inefficient market our two examples above would also seem to be at first sight inefficient. To test this, the empirical measure we use for inefficiency, based on the concept of excess trading profits, is

$$I_{ij}(t, t') = \frac{(E_i(t, t') - E_j(t, t'))}{\left(\frac{\sigma_i^2(t, t')}{N_i} + \frac{\sigma_j^2(t, t')}{N_j} \right)^{1/2}} \quad (6.4)$$

where $\sigma_i^2(t, t')$ is the variance in the excess trading profits of the i th trading strategy/group. Essentially, I_{ij} is a t-test for the difference in mean excess trading profits of two trading strategies/groups. In the denominator is the standard error, so, this measure informs us of the statistical significance of the difference of the means and therefore on the probability of whether it is luck or an intelligent strategy that leads to such a profit distribution. Typically, if $|I_{ij}| > 2$ then the relative Inefficiency between strategies i and j is statistically significant. Using this measure one can determine that both our examples above are, in fact, “efficient” [10] in that no trader is making excess profits relative to another. This is, in fact, a hallmark of markets with homogeneous trading strategies.

6.3 The Problem of Relating Profits to Strategies

Having decided how to measure profits in a way that distinguishes between luck and intelligence we now encounter the problem of trying to characterise a trading strategy so that we can apply our profits measure to it. What is a trading strategy in the first place? Abstractly, it is a rule, F , that maps a set of input variables, $\mathbf{X} = (X_1, X_2, \dots, X_N)$, to a trading decision, D , such as buy 5,000 shares of Federal Screw with a market order, or place a limit order to sell at a price of \$13.33 etc. Thus, a strategy is $D = F(\mathbf{X})$. Sounds conceptually simple doesn't it? It's not. How does one establish the input set \mathbf{X} for instance? What sounds logical is when a rule is very algorithmic, such as IF(price > 1 month moving average) THEN (Buy). In fact, this is how artificial financial markets work, and also how trading, using AI type tools, such as GP or GAs, becomes a rules-based algorithm. The problem is that human beings aren't algorithms. One can imagine, for instance, that traders who were involved in particularly stressful personal relationships or in financial difficulties had poorer trading performance than those who weren't. Does that mean that X_{568} should represent relationship difficulties = YES/NO?

Just like finger prints, no two humans will have exactly the same trading strategy F . There are just too many variables that enter decisions. Worse, many of the variables can't be put into a form that could be modelled algorithmically. This is, in fact, a deep problem of efficient markets theory when thought of in terms of information. Each trader has a unique information set and we don't know how to map it to a trading strategy. Basically, efficient markets theory tries to circumvent this by assuming that information is common

property, the type of information that is common depending on which formulation of the efficient markets hypothesis one takes - Weak, Semi-strong or Strong - and that investors are rational and therefore will act on this information rationally.

6.3.1 The Genotype-Phenotype Map in Finance

To use an analogy familiar in Evolutionary Computation we can place the above discussion in the context of imagining that a given trader has a “genotype” and a corresponding “phenotype”. The genotype in this sense corresponds to the trading strategy of the trader as characterized by the map between the information set of the trader and the actual set of buys and sells. Of course, in the real world this genotype is hidden, although efficient markets theory would claim that the genotype, at least for a rational investor, leaves its mark on price and therefore, in principal at least, can be inferred. Inferred through what information? Instead of trying to infer the genotype we will here think more of trying to see how the phenotype of the strategy manifests itself in the market. By phenotype we mean a set of phenotypic characteristics that should be observable.

What would these characteristics be? Well, let's think of some simple examples: a momentum-based strategy, for instance, should manifest itself in the fact that transactions are carried out in accordance with the formation of tendencies in the price. A contrarian strategy on the other hand might manifest itself through buys when a stock is low and sells when it is high. A stock picking strategy could manifest itself by buys/sells only in particular stocks using a portfolio that is quite distinct to the market portfolio. A market making strategy on the other hand might be identified by frequent changes in position at or near the best prices in the market as the trader tries to make money trading the spread.

In deciding which phenotypic characteristics are the most useful, observationally we will assume that we have access to $\{D\}$, the set of trading decision, on which to construct them. This will mean that we have access to the individual trades and quotes of each trader. Of course, the actual available information set may be less than this. For example, the identity of the trading parties is usually confidential. However, in principle one has a data set that contains the price, type and identity (which could be anonymized) of every order placed in the market. In principle, one could take $\{D\}$ as the phenotype. Obviously, the genotype-phenotype map in this case may be degenerate, as there might be many different genotypic trading strategies that lead to the same set of trading decisions, especially if the set is small. For instance, there are many possible trading strategies that lead someone to buy 1,000 shares of Federal Screw at a given time. On the other hand the full set of trading decisions $\{D\}$ will be too fine grained and not allow one to see commonalities. For instance, two sets of trading decisions may be distinct but both be associated with a momentum strategy on a particular stock. What is required is a set of coarse grained variables that summarize different dimensions of the fine grained $\{D\}$. In Sect. 6.4 we will see just such a possible set.

6.3.2 The Problem of Realizing Profits Using a Trading Strategy

Most work done in applying AI techniques and in particular EC to algorithmic trading is essentially “paper trading”; i.e., past data is taken, a model created and the model then “tested” on another, separate set of past data. What “tested” means here is that

the trading decision of the algorithm is derived and then it is *assumed* that the trading decision is implementable and, moreover, that the net profit of the decision can be read off from the subsequent evolution of the market. These assumptions have two potential pitfalls: firstly, it may be that the trade was not realizable; and, secondly, it may be that the trade affected the subsequent evolution of the market so that the real profit if the decision had been implemented would have been quite different to the expected profit in the approximation that the trade did not affect the subsequent evolution of the market. The first assumption is essentially an assumption about liquidity available in the market, while the second is associated with the problem of market impact. Essentially, the assumptions will be less problematic in the limit that the implemented trade is in a very liquid situation and does not move the market much. For instance, it's a good bet that you can buy 100 shares of Microsoft at any time and without moving the price. Buying 100,000 shares of a stock with an Average Daily Volume of 50,000 shares is a completely different matter.

6.4 Inferring the Trading Strategy Phenotype by Data Mining

Faced with the problem of not being able to characterize the trading strategy genotypically we have to decide how to characterize it phenotypically. Of course, to be useful in this sense, the phenotypic traits of the strategy must be observable. The question then is what set of observable traits to use? If we think of the transactions and quotes that a trader participates in, one has available the type of order placed, the price of the order or transaction, the size of the order or transaction, in what asset and at what time. From those fundamental degrees of freedom: order type, price, volume, time and asset one can of course construct a large number of derived characteristics. Below, we list a useful subset that we will later use in a concrete demonstration of how these phenotypic variables can be used to characterize trading strategies.

- Number of transactions
- Number of buys
- Percentage buys
- Number of sells
- Percentage sells
- Number of position changes (buy to sell or vice versa)
- Percentage of position changes
- Number of stocks transacted
- Number of counterparties
- Total volume
- Volatility of volume
- Average volume per transaction
- Total excess trading profits
- Volatility of excess profits
- Average profit per transaction
- Sharpe ratio for excess trading profits
- Percentage of correct decisions
- Total number of transactions with profits/losses

Of course, we not only need to determine a potential set of phenotypic traits on which to base a comparison but we also need to be able to determine if the difference between two sequences of trading decisions is statistically significant in terms of any one of these variables. There are various statistics that are adequate to the task. Later, we will illustrate the comparison using the following statistic

$$\varepsilon' = \frac{\langle x \rangle_1 - \langle x \rangle_2}{\left(\frac{\sigma_1^2}{n_1} + \frac{\sigma_2^2}{n_2} \right)^{1/2}} \quad (6.5)$$

where $\langle x \rangle_i$ is the mean value of the variable x for trading strategy i and n_i is the number of traders associated with that strategy. The denominator is just the standard error so the statistic is just determining to what extent the difference in means for a given phenotypic trait for the two strategies is just due to chance. If we imagined that the two strategies are drawn from two different normal distributions then standard hypothesis testing would tell us that if $|\varepsilon'| > 2$ then the two strategies are distinguishable at the 95% confidence level.

How might different trading strategies manifest themselves in terms of these phenotypic characteristics? Well, with the examples mentioned in the previous section, a momentum-based strategy, for instance, should manifest itself in the fact that there is an excess of buys or sells in accordance with the formation of tendencies in the price. A stock picking strategy could manifest itself by buys/sells only in particular stocks using a portfolio that is quite distinct to the market portfolio. A market making strategy on the other hand might be identified by frequent changes in position at or near the best prices in the market and trading with many counter-parties.

As an example of this thinking we will consider the case of a “political” experimental market [12] that ran between the middle of August and the middle of September 2004 and was based on parliamentary elections in the German state of Brandenburg. In this case the “stocks” are eight political parties and the price of the stock is the expected share of the vote for that party. There were 108 participants in the market and each participant started off with a fixed amount of money and shares.¹ The market structure was that of a continuous double auction where traders could place limit orders of market orders. The order book for the market was partially open as the traders had access to the three best buy and sell quotes.

6.5 Results

We wish to determine first of all whether there are any traders that systematically make excess trading profits. We then wish to determine if they, or, rather, their trading strategy, can be characterized phenotypically. In terms of determining whether they are making excess profits systematically we first divide the market temporally into two periods with roughly the same number of trades in each period. We then rank the agents according to their excess trading profits in the first period of the market. From this ranked list of 108 agents we form 10 groups, starting from the agent with the most excess trading

¹ There was no investment of real money in the market.

profits: 3 groups of 11 agents; 3 groups of 10 (that did not trade in the first period); 3 groups of 11 and, finally, one group of 12 corresponding to those traders with the biggest systematic losses.

6.5.1 Is It an Efficient Market?

In Figs. 6.3 and 6.4 we see the relative Inefficiency between different trading groups for the first and second periods of the market respectively. In Fig. 6.3, the solid curve is the relative Inefficiency of the group with highest profits in the first period relative to that of the rest of the market. The dotted curve is the excess profits of the highest profits group in the first period against the lowest. Both these curves show that group 1 is clearly making a statistically significant amount of excess profits relative to the rest of the market in the first period and, even more so, relative to the group of biggest losers. The dashed curve in Fig. 6.3 is the relative Inefficiency in the first period of the market between two groups of 11 traders chosen randomly. What can be seen, as is to be expected, is that there is no statistically significant Inefficiency between these two groups as having been chosen randomly they cannot be associated with any common trading behaviour and hence cannot share the same strategy.

The three corresponding curves in Fig. 6.4 represent the relative inefficiencies between the same groups as just mentioned, but now considering the excess trading profits for these groups in the second period of the market. Once again, group 1 is making excess trading profits relative to the rest of the market as a whole and with respect to group 10, though, now, the relative inefficiency with group 10 is at the same level as with the rest of the market. So, we can establish that the market has inefficiencies with the associated inference that one group is making profits in a way that cannot be

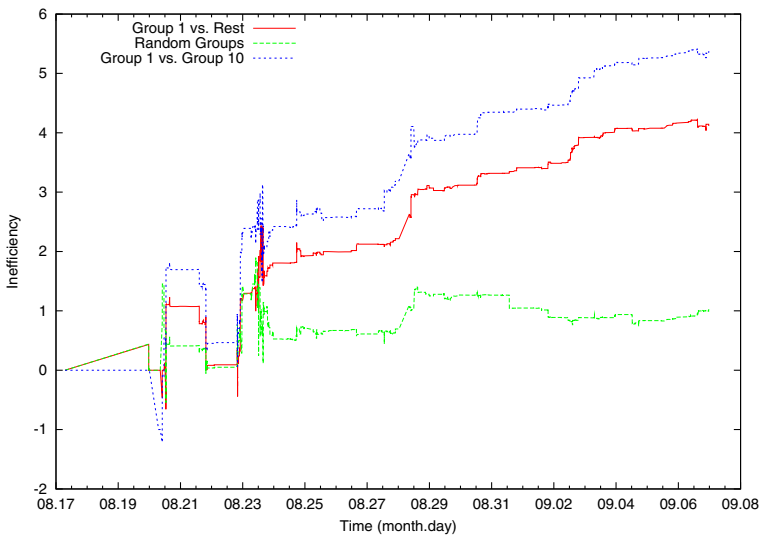


Fig. 6.3. Graph of Inefficiency against time for different trader groups for first period of the market

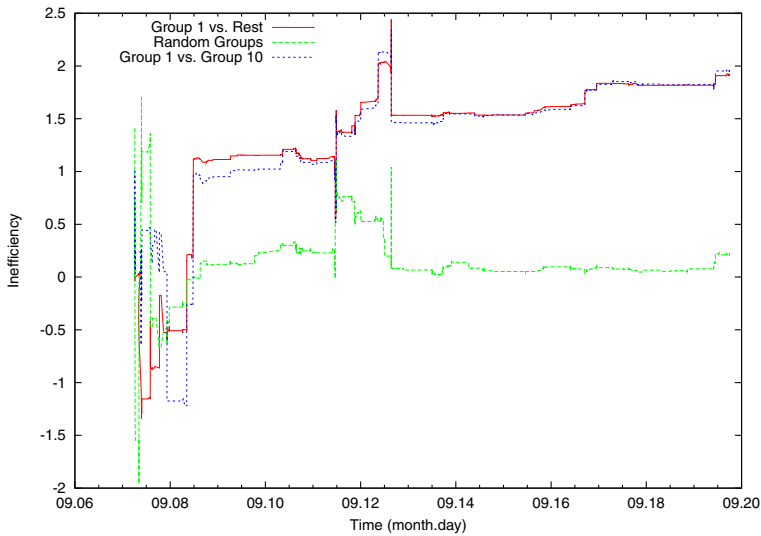


Fig. 6.4. Graph of Inefficiency against time for different trader groups for second period of the market

attributed to luck. However, we see that the market was much more inefficient in the first period than in the second.

6.5.2 Why Is It Inefficient?

So, we can ask: what are the winners doing that makes them so successful? A first question is: Who are they trading with? Are they making profits across all trader groups or are they having more success with some traders versus others? The second question is: Why is the market much more efficient in the second period? Did the intelligent/unintelligent traders stop being intelligent/unintelligent? In Table 6.1 we see the excess trading profits of the different trader groups in the two halves of the market. As a matrix this table is antisymmetric as the net profit of a group with itself has to be zero. Note that in the first period more than 60% of group 1's profits come from trading with group 10 while over 70% of group 10's losses come from trades with group 1. Given that the market was anonymous however, how did the winning traders that enter in group 1 identify losing traders from group 10?

In the second period of the market, when compared to the first period, we see a radical change. Group 1 continues to make a significant amount of excess trading profits, but now the percentage of their profits that come from group 10 is only about 10%, in stark contrast to the 60% figure from the first period. In fact, about 60% of group 1's profits now come from trading with three groups - 4, 5 and 6 - that did not even participate in the first period. Notice also that the total profits of group 1 falls from about 25 million in the first period to about 18 in the second. On the contrary, the losses of group 10, which amounted to about 22 million in the first period were reduced to only 3 million in the second!

Table 6.1. Profits and losses in millions between trader groups for the two periods of the market

First Period										
Group	1	2	3	4	5	6	7	8	9	10
1	0	0.054	0.857	0	0	0	0.061	1.145	7.332	16.455
2	-0.054	0	-0.043	0	0	0	0.355	0.033	0.721	4.460
3	-0.857	0.043	0	0	0	0	0.724	0.671	0.562	0.326
4	0	0	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0	0	0
7	-0.061	-0.355	-0.724	0	0	0	0	-0.162	0.475	0.228
8	-1.145	-0.033	-0.671	0	0	0	0.162	0	0.054	-0.052
9	-7.332	-0.721	-0.562	0	0	0	-0.475	-0.054	0	0.826
10	-16.455	-4.460	-0.326	0	0	0	-0.228	0.052	-0.826	0
Second Period										
Group	1	2	3	4	5	6	7	8	9	10
1	0	-0.068	0.544	10.616	1.737	1.042	2.281	-1.020	1.430	2.236
2	0.068	0	0.008	0.000	0.062	0.173	-0.100	-0.019	0.005	-0.017
3	-0.544	-0.008	0	0.053	0.117	-0.097	0.181	0.007	-0.089	0.300
4	-10.616	0.000	-0.053	0	0.173	-0.295	0.143	0.027	0.107	0.229
5	-1.737	-0.062	-0.117	-0.173	0	0.156	0.016	-0.227	0.193	0.148
6	-1.042	-0.173	0.097	0.295	-0.156	0	0.139	0.616	-0.151	-0.045
7	-2.281	0.100	-0.181	-0.143	-0.016	-0.139	0	0.272	0.040	-0.059
8	1.020	0.019	-0.007	-0.027	0.227	-0.616	-0.272	0	0.677	0.083
9	-1.430	-0.005	0.089	-0.107	-0.193	0.151	-0.040	-0.677	0	-0.022
10	-2.236	0.017	-0.300	-0.229	-0.148	0.045	0.059	-0.083	0.022	0

In Table 6.2 we see a table that represents a matrix whose elements are the percentage of trades transacted between one group and another. We see that almost 50% of group 1's trades in the first half were with traders from group 10. In fact, both groups constitute more than 60% of the total liquidity in the market in the first period. However, just because the two groups are responsible for most of the trading does not imply that one should be systematically winning against the other. In contrast, in the second period only 15% of group 1's trades are with group 10 traders.

So what else is going on? In Fig. 6.5 and we see a list of all the trades carried out by group 1 in the first period ranked by excess trading profit. What is immediately noticeable is that basically all profits come from only a relatively small number of trades. In fact, more than 90% of group 1's profits come from only about 5% of trades. On the contrary, the vast majority of trades lead to very small profits/losses. In fact, about 40% of trades led to no profit/loss at all due to the fact that there was no price movement relative to the previous trade. In contrast, we can see in Fig. 6.5 that for group 10 about 90% of their losses comes from about 5% of trades. The extreme heterogeneity in the losses of group 10 is, in fact, a mirror of the profits of group 1. As with group 1, with about 40% of trades there was no price movement and hence no profit/loss.

With these results in hand we can start to see how and why group 1 is winning. As mentioned, given that the market is anonymous, group 1 traders cannot identify group 10 traders per se. However, what is clear is that they are "intelligent" enough to identify good trading opportunities. These trading opportunities are such that a trader, principally from group 10, places a limit order at a very exaggerated price - very low for a sell and very high for a buy - in a limit order book with no other orders, otherwise the order would cross the spread and execute against an already existing limit order on the

Table 6.2. Percentage of transactions executed between different trader groups in the two periods

First Period										
Group	1	2	3	4	5	6	7	8	9	10
1	17.12	4.36	9.96	0	0	0	3.42	5.81	10.63	48.70
2	23.60	4.49	16.01	0	0	0	4.78	6.18	11.52	33.43
3	32.27	9.58	9.75	0	0	0	7.23	6.72	5.38	29.08
4	0	0	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0	0	0
7	25.10	6.46	16.35	0	0	0	3.04	7.22	11.41	30.42
8	23.48	4.61	8.39	0	0	0	3.98	10.06	6.92	42.56
9	30.42	6.08	4.75	0	0	0	4.45	4.90	7.42	41.99
10	37.31	4.73	6.87	0	0	0	3.18	8.07	11.24	28.61
Second Period										
1	18.47	3.54	9.36	5.57	10.94	9.93	10.94	9.87	5.88	15.50
2	32.75	1.17	11.11	2.92	5.26	9.94	8.77	8.19	2.34	17.54
3	20.76	2.66	14.59	4.77	10.66	8.70	8.56	11.08	4.49	13.74
4	22.22	1.26	8.59	7.58	8.08	9.34	13.13	7.58	6.06	16.16
5	29.08	1.51	12.77	5.38	6.05	13.11	5.04	11.93	1.85	13.28
6	23.64	2.56	9.34	5.57	11.75	13.25	9.04	8.58	3.16	13.10
7	28.36	2.46	10.00	8.52	4.92	9.84	3.28	14.10	3.44	15.08
8	23.89	2.14	12.10	4.59	10.87	8.73	13.17	5.51	2.60	16.39
9	35.09	1.51	12.08	9.06	4.15	7.92	7.92	6.42	3.02	12.83
10	26.34	3.23	10.54	6.88	8.49	9.35	9.89	11.51	3.66	10.11

other side. An alert trader from group 1 notices the opportunity and places a market order which executes against the limit order. After the trade, limit orders are placed at prices that are more in line with the past history of the asset and then only relatively small profits/losses are again available. In other words, group 1 traders are alert entrepreneurs that time their trades according to how appropriate market conditions are - in this case an inappropriately placed limit order at variance with previous market expectations. The number of these badly placed orders, that are far out of line with previous expectations of the price, is relatively few. However, they are predominantly placed by group 10 traders, while group 1 traders are the most adept at taking advantage of them.

Considering the same curves in the second period in Figs. 6.7 and 6.8 we see that, although, as in the first period, there is a great deal of heterogeneity in the relative profits of the different trades, it is now much reduced for group 10. The biggest loss from a single trade for group 10 is about 0.7 million, compared to 2.6 million in the first period. This indicates that group 10's price expectations are much more in line with those of the rest of the market in the second period. Interestingly, group 1 seems to show even more heterogeneity in the second period. As we can see though this is due to one extremely large trade that was three times bigger than any trade in the first period. Apart from that

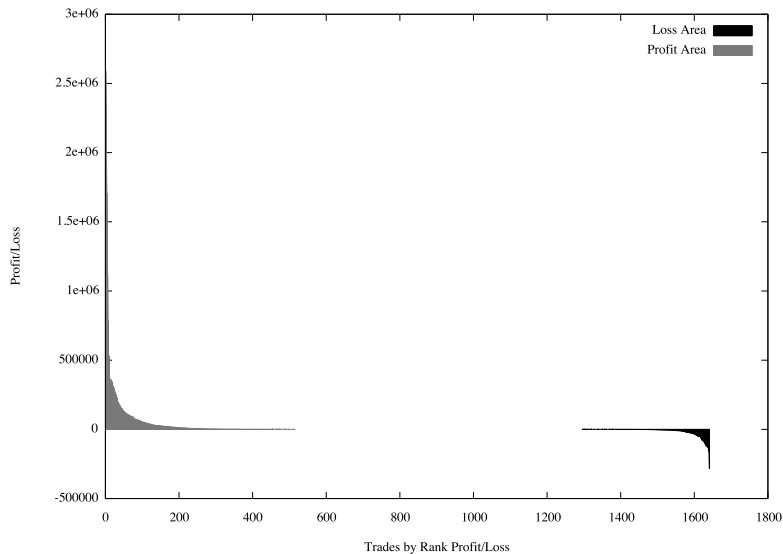


Fig. 6.5. Ranked list of transactions by excess trading profits for Group 1 in the first period

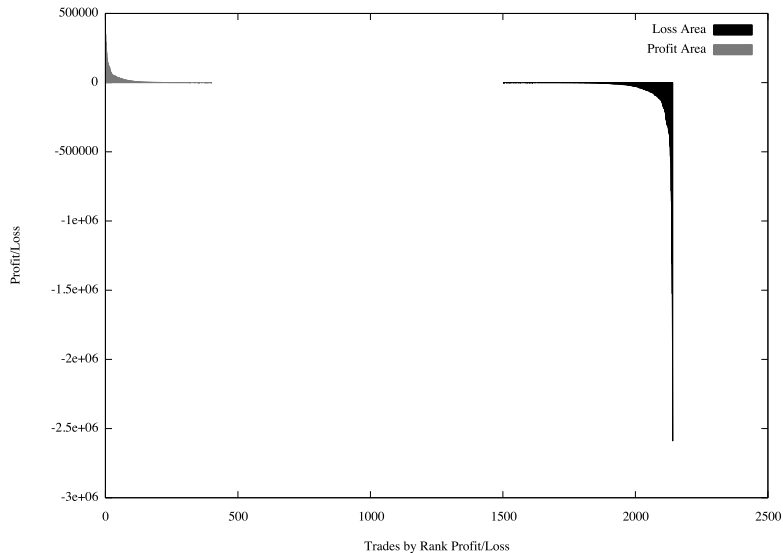


Fig. 6.6. Ranked list of transactions by excess trading profit for Group 10 in the first period

one single trade, which was with a counter-party from group 4 that, remember, had no trades in the first period, the degree of heterogeneity is substantially less than in the first period.

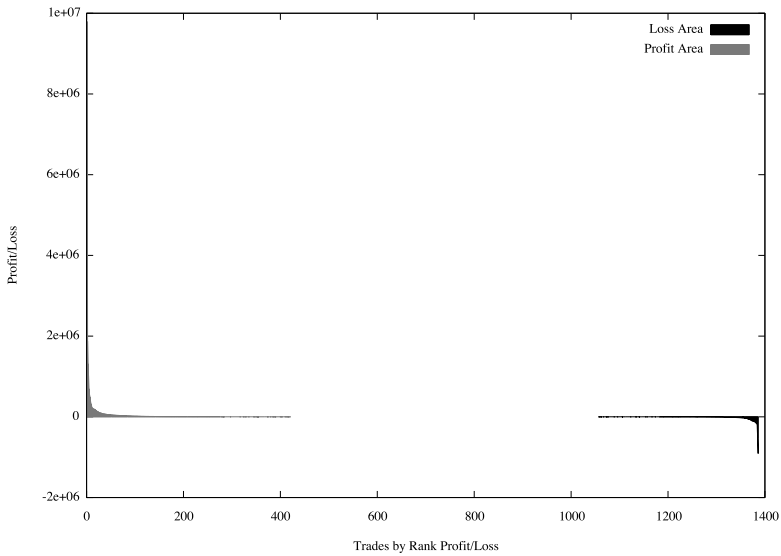


Fig. 6.7. Ranked list of transactions by excess trading profits for Group 1 in the second period

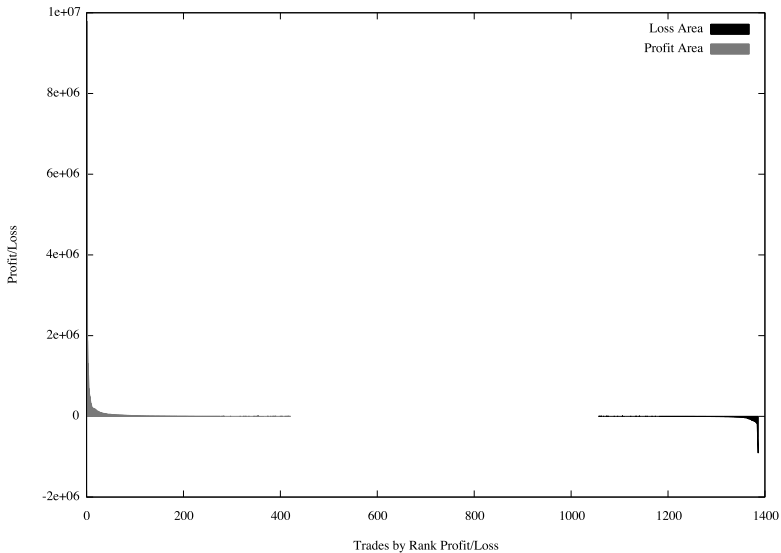


Fig. 6.8. Ranked list of transactions by excess trading profit for Group 10 in the second period

6.5.3 Characterizing Trading Strategies

So, we see that there are significant differences between different trader groups in terms of their profits. We have also seen that these differences are related to who is trading

Table 6.3. Values of ε' for feature values by group

Group 1 vs.	Trades	Buys	Sells	D	Partner	Volume	Volume Standard Deviation	Mean Volume per Trade
First Period								
2	3.60	3.72	3.19	1.76	3.69	4.25	-0.16	-0.50
3	2.69	2.60	2.61	3.35	3.09	3.51	1.64	-0.39
7	3.83	3.92	3.50	0.63	4.64	4.05	0.20	-0.48
8	3.25	3.42	2.78	2.09	3.09	2.95	1.50	0.83
9	2.62	2.40	2.56	0.76	2.77	2.98	-0.21	-0.51
10	-0.44	0.47	-0.75	-0.37	-0.39	0.32	1.00	0.53
Second Period								
2	2.36	2.30	1.81	2.91	3.12	2.84	3.04	2.13
3	1.17	1.09	1.02	2.87	1.62	1.49	2.40	1.60
4	1.73	1.67	1.45	-0.10	2.03	2.04	0.28	0.16
5	1.23	1.71	0.44	0.98	1.76	1.38	-0.36	-0.04
6	1.33	1.67	0.45	-0.22	1.04	1.44	1.53	1.00
7	1.49	1.33	1.39	1.83	1.86	1.83	2.23	1.71
8	1.43	1.72	0.54	0.85	1.57	1.72	2.36	1.62
9	2.21	2.18	1.66	-1.01	2.55	2.29	0.30	0.11
10	1.10	0.99	0.94	-0.39	1.08	1.53	1.35	0.69

with who, with the group of biggest winners directly taking advantage of a group of traders that have inferior trading strategies in that their expectations about price are at times in violent disagreement with the rest of the market. The question now is: Can we characterize the trading strategies of the different groups as outlined in section 6.4? In Table 6.3 we see the values of ε' , introduced in Sect. 6.4, where strategy 1 refers to group 1 and strategy 2 refers to one of the other nine groups. In other words, ε' is determining if there is any statistically distinguishable difference in the strategy of group 1 relative to those of other groups in terms of the phenotypic traits considered. Considering the phenotypic profiles from the first period, we see that group 1's strategy is very distinguishable from those of group 3, 7, 8 and 9 (groups 4, 5 and 6 are omitted as they did not trade in the first period). Group 1's strategy is distinguishable from these others in that group 1 transacted more trades, both buys and sells, traded with more counter-parties and operated significantly larger volumes.

Comparing with group 10 however, we see that group 1's strategy is not that distinguishable from that of group 10, with no statistically significant differences between any of the measured phenotypic traits. What this is telling us is that the small subset of phenotypic traits we are considering is enough to distinguish between those who win/lose a lot versus those who do not win/lose much but not enough to distinguish between large profits versus large losses. This, of course, is not overly surprising. If it were so easy to distinguish a winning versus a losing strategy using such simple variables then we would be using it to trade and not publishing it in an academic forum!

Considering now the second period, we can see that the statistical signal that differentiates group 1 from other groups is now much weaker. Group 1 is still associated with more trading, both buy and sell, but now the statistical significance of the differences is substantially less. In particular we can see that the profile of group 1 relative to that of group 10 is different in the second period with the latter being now more distinguishable from the former.

Table 6.4. Trader groups with associated Sharpe Ratio and Score rankings in the two periods

Group	Agents	First Period		Second Period	
		Sharpe Ratio Rank	Score Rank	Sharpe Ratio Rank	Score Rank
1	mammutfam	1	1	1	4
	hanni1982	2	5	34	16
	famfan	3	6	14	34
	saladin	4	4	5	1
	gruener	5	3	12	12
	zigzag	6	7	4	10
	Wahlalal	7	22	19	75
	rapper	8	20	39	88
	kaufunger	9	16	21	40
	truck676	10	30	10	18
	cezanne	11	23	37	56
10	Traurig	97	26	82	37
	Geoman	98	41	81	107
	Tishimdorf	99	21	108	19
	Progno	100	19	17	38
	briutt	101	18	75	45
	Tob11	102	43	33	56
	Camporesi	103	14	74	65
	fischmob	104	8	6	9
	Rodrigues	105	17	96	24
	Mauritius	106	11	86	15
	BAYERNP	107	2	102	11
	Angelo	108	12	69	105

An important aspect of this identification of phenotypic traits that characterize a particular trading group is that it allows for the creation of a predictive model based on these features. A robust and useful model is to use the Naive Bayes approximation [13] on these traits to create a score model where the chosen class is that of the top decile of traders with respect to the Sharpe ratio of excess trading profits, i.e., group 1. The idea is that the first period is used to specify a score $S(C|\mathbf{X})$, where C is the top decile of traders ranked with regard to the Sharpe ratio and \mathbf{X} is a feature vector, which serves as a predictive model with which to phenotypically characterize the trading strategy of this winning group. In the second period this model is then applied to the traders and one observes to what extent this model is capable of predicting the most successful traders. In Table 6.4 we see the relative ranking of the traders associated with groups 1 and 10.

What can we glean from this table? Clearly there is a high degree of correlation (coefficient of correlation 0.88) between the score and Sharpe ratio ranks in the first period for group 1. This is not overly surprising given that the model was created on the data of period 1 to predict the class of group 1 traders. For group 10 there is a strong anticorrelation. This is because group 10 traders also have a high score as the model is predominantly picking up predictability of big winners/losers versus small winners/losers. In other words the chief source of false positives for predicting membership of the group with the most profits is precisely the group with the biggest losses. In the second half there is also a substantial correlation (coefficient of correlation 0.71) between Sharpe ratio and score ranks. This is gratifying that it shows that, out of sample, the model is predicting well in that score is a good predictor of Sharpe ratio rank. The most interesting aspect though is to consider the change between the first and second periods. The average Sharpe ratio rank for group 1 in the first period is 6, while

the average score rank is 12. In the second period, the average Sharpe ratio rank of the traders who gained the most in the first period is 18. In other words, performance has deteriorated substantially. The average score rank in the second period is 32. What this implies is that the profile of the traders and the consequent score is a good predictor of relative performance, in that a decrease in the score is associated with a decrease in performance.

6.5.4 Adaptation and Learning

We now turn to the question of what changes between the first and second period that gives rise to such noticeable differences in performance of the traders who were in groups 1 and 10 in the first period? Remember, that in the first period, we characterized group 1 traders as alert entrepreneurs who identify the important trading opportunities when they arise that are being offered by group 10 traders. We saw that these trading opportunities were relatively few but led to the vast majority of profits of group 1, and also to the vast majority of losses of group 10. Both sets of traders had similar phenotypic profiles, the main difference between them being due to the price and order type associated with those trades that led to large profits/losses.

In the second period we saw that the trading characteristics of the traders of groups 1 and 10 were quite different to those exhibited in the first period. This was seen at the level of their profits and losses and in terms of the percentage of trades carried out between these two groups. Given that the scores for the two groups are quite distinct between the first and second periods then, taking the score components and the associated phenotypic traits as a proxy for the trading strategy used, it is clear that the groups are using quite distinct trading strategies in the second period compared to the first.

Why is this? Why would group 1 change a winning strategy? The answer is that they did not have any option. Their strategy was changed for them by the actions of others. In the first period, groups 1 and 10 were involved in 1,928 and 2,517 trades respectively, while in the second period these numbers were 1,581 trades for group 1 and only 930 for group 10! Group 10 traders incurred large losses using the trading strategy that they implemented in period one and so, learning from their mistakes, adapted. This adaptation took the form of a withdrawal from the market. If you don't trade you can't lose. There was also learning in the sense that the number of orders placed at prices that were far from market expectations was smaller. This learning and adaptation in the strategy of group 10 manifested itself in the fact that the score components diminished in the second period. In this sense, adaptation and learning are manifest in the score and the associated phenotypic profile as representations of the trading strategy.

How does this answer why group 1 changed their strategy? The strategy of group 1 would clearly be to keep on identifying profitable trading opportunities. The total number of trades they carried out was not so different in the second period compared to the first. However, who they traded with was quite different. In the first period group 10 offered group 1, anonymously of course, profitable trading opportunities. Group 1 traders, as alert entrepreneurs, gladly accepted them. In the second period, these trading opportunities from group 10 dried up to a large extent. Group 1 therefore had to look for new opportunities by trading, once again anonymously, with other traders. However, these other trader groups did not generally offer the same profit opportunities, being

pegged to more realistic price scenarios. Group 4 was the exception to this rule, but group 4 was a “novice” group in the second period not having traded in the first.

6.6 Conclusions

Financial markets can be appropriately thought of as consisting of ecologies of competing trading strategies. In fact, artificial agent-based financial markets are predicated on this concept. A goal then is to understand which strategies are the most appropriate at a given time, leading to the most profits. In this chapter we have shown that there are important and deep subtleties associated with determining what a strategy is and, furthermore, whether it is winning by luck or by skill. Normally, in artificial financial markets, or in applications of Evolutionary Computation techniques to markets, a trading strategy is defined algorithmically, usually in a very simple fashion. In this sense, there is no ambiguity. With an appropriate definition of trading profits one can then determine whether such a strategy is superior as opposed to just lucky. However, such strategies are almost inevitably tested in paper trading where neither the feasibility of the strategy is tested nor its impact in the market.

As markets are intrinsically adaptive and traders learn this is a very strong caveat when trying to evaluate potential performance in real market trading. A particular trading strategy has to compete against others. One that performs well may well have the undesired effect of causing others to change their strategy so that the profits disappear. This is, in fact, the conceptual underpinning of efficient markets theory. So, it is important to be able to have an idea for how a given strategy might influence others. However, we do not have access to the trading strategy of others.

To circumvent this problem we introduced the notion of a genotype-phenotype map to finance, wherein the hidden “genotype” of the strategy was proxied using a set of coarse grained variables associated with the “phenotype”, i.e., the observable set of buys and sells of the strategy. We showed how data mining techniques could be used to characterize trading strategies and also provide a predictive model for winning strategies. We illustrated all of this in the context of a controlled experimental market. We showed that the market was inefficient, in that certain traders made excess trading profits at the expense of others in a way that was not attributable to luck. We then characterized the winning trading strategy and created a predictive model. We saw that losing traders in the market learned from their experience and adapted by changing their trading strategy accordingly. This was manifest in the fact that the phenotypic characterization of their strategy changed radically as a function of time. We also saw that this had a knock-on effect in that the winning strategy of the most profitable traders was now no longer implementable as they could no longer rely on the cooperation of the counter-parties from whom they had previously been making profits. It takes two to tango!

We believe that the potential for reverse engineering trading strategies in financial markets as we have shown here is an important potential application for data mining techniques. We also believe that these results should be taken as a cautionary tale for those who would like to believe that finding winning trading strategies by paper trading is an indication of how well the strategy will perform in a real market.

Acknowledgments

Christopher R. Stephens, José Luis Gordillo, and Enrique Martínez Miranda acknowledge support from the UNAM Macroproyecto MTUIC and from Conacyt.

References

- [1] Farmer, J.D., Lo, A.: *Frontiers of Science: Evolution and efficient markets*. Proceedings of the National Academy of Sciences 96, 9991–9992 (1999)
- [2] Farmer, J.D.: *Market Force, Ecology and Evolution*. Santa Fe Working Paper 98-12-117E Santa Fe Institute (1998)
- [3] LeBaron, B.: *Evolution and Time Horizons in an Agent-Based Stock Market*. *Macroeconomic Dynamics* 5, 225–254 (2001)
- [4] Palmer, R.G., Arthur, W.B., Holland, J.H., Lebaron, B., Tayler, P.: *Artificial Economic Life: A Simple Model of a Stock Market*. *Physica D* 73 (1994)
- [5] Gordillo, J.L., Stephens, C.R.: *Analysis of Financial Markets using Artificial Intelligence Techniques*. *Computación y Sistemas* 6, 253–273 (2003)
- [6] LeBaron, B.: *Agent-based Computational Finance*. In: Tesfatsion, L., Judd, K.L. (eds.) *Handbook of Computational Economics*, vol. 2, ch. 24 (2006)
- [7] Fama, E.F.: *Efficient Capital Markets: A Review of Theory and Empirical Work*. *Journal of Finance* 25, 383–417 (1970)
- [8] Fama, E.F.: *Efficient Capital Markets: II*. *Journal of Finance* 46, 1575–1617 (1991)
- [9] Bagehot, W.: [pseud], *The Only Game in Town* *Financial Analyst's Journal* 27, 12–14 (1971)
- [10] Benink, H.A., Gordillo, J.L., Pardo, J.P., Stephens, C.R.: *A Study of Neo-Austrian Economics Using an Artificial Stock Market*, EFA 2004 Maastricht Meetings Paper No. 3218 (March 2004) (submitted to *Journal of Empirical Finance*), <http://ssrn.com/abstract=567125>
- [11] Stephens, C.R., Benink, H.A., Gordillo, J.L., Pardo-Guerra, J.P.: *A New Measure of Market Inefficiency* (August 24, 2007), <http://ssrn.com/abstract=1009669>
- [12] Hauser, F.: *Die Presse Online-Wahlbörse 2002 - Eine finanzwirtschaftliche Betrachtung*. Thesis, University of Innsbruck (2003)
- [13] Hand, D., Mannila, H., Smyth, P.: *Principles of Data Mining*. MIT Press, Cambridge (2001)

Financial Bubbles: A Learning Effect Modelling Approach

Tsung-Han Hsieh^{1,2}, Youwei Li¹, and Donal G. McKillop¹

¹ School of Management, Queen's University Belfast, 25 University Square, Belfast, BT7 1NN, United Kingdom
{thsieh01,y.li,dg.mckillop}@qub.ac.uk

² Tajen University, Taiwan

Summary. This chapter studies financial bubbles by incorporating a learning effect into the coordination game model which was articulated by Ozdenoren and Yuan [36]. Monte Carlo simulation is then utilised to analyse how the addition of a learning effect impacts upon the investment decision of informed investors as well as the formation of the aggregate investment. The simulation exercise demonstrates that both the learning effect and the feedback effect contribute to price multiplicity with price multiplicity observed when informed investors have more precise private information. The analysis emphasises that the learning effect is stronger in situations where informed investors act counter to the price signal and the actions of uninformed investors.

7.1 Introduction

A financial bubble (or bubble, hereafter) occurs when a set of assets are traded at higher prices than their fundamental values and in larger volume for a period of time with a price and volume crash then ensuing. An example of a bubble occurred in the Internet sector during the 1998 to 2000 period. NASDAQ listed Internet stocks rose by a factor of 3 between January 1999 and February 2000, volume also increased three-fold during this period. Internet stocks crashed in February 2000 with volume and prices falling by 30 percent in a few days. From start to finish the Internet bubble lasted approximately 14 months. Not unexpectedly such phenomena attract many studies seeking to explain what contributes to these price and volume patterns and how factors interplay with each other.

Kindleberger and Aliber [28, pp. 33–46] argue that bubbles occur due to the excessive optimism of speculative investors. In the past this excessive optimism is fuelled by the self-fulfilling effects of investors' beliefs (Flood and Garber [17]). The start of a bubble is brought about by optimistic investors believing in a prosperous future due to the discovery of a new market, the invention of a new technology or other similar positive information suggesting that significantly higher returns can be made relative to that available from contemporary investments. Such positive expectations encourage increases in both price and volume. When prices subsequently falter a mass exodus from the market ensues resulting in both a price and trading volume crash. Kindleberger and Aliber [28] suggest that bubbles tend to be associated with laxity of credit standards

and ethical restraints and ‘the implosion of an asset price bubble always leads to the discovery of frauds and swindles’. This then begs the question why do bubbles emerge and grow without investor fear?

Morris and Shin [30] took the view that self-fulfilling beliefs’ studies were of limited benefit to policy makers as this explanation of financial bubbles does not provide an answer to the question of how beliefs are formulated. They argue that price-multiplicity, which happens whenever multiple equilibrium prices occur with a given market supply, in traditional equilibrium models is too simple as it does not explain the role played by information in forming the crises. On the other hand, coordination game theory in which the participants benefit most when taking the same actions or not at all, might be helpful as an explanation of such circumstances (see Cooper [9]). Morris and Shin [30], too, address the issue of price multiplicity and combine findings from the coordination of market participants by Carlsson and van Damme [7] and the rational expectation equilibrium model of Grossman and Stiglitz [20] and Hellwig [22] to formulate a global game theoretical model. Studies, including Angeletos and Werning [2] and Ozdenoren and Yuan [36], have applied and extended this model to analyse bubble-like investor behaviours in different situations, such as the foreign exchange markets and in the case of bank runs.

A general observation in markets is that informed investors tend to take the actions and indeed manipulate the actions of uninformed investors when the former make investment decisions. Such manipulation may lead uninformed investors to overestimate market sentiment and make incorrect investments. A situation such as this may occur in scenarios in which insider trading or other facets of corporate misconduct are prevalent. By augmenting the baseline model formulated by Ozdenoren and Yuan [36], we demonstrate how investor decision making and market equilibrium can be affected by corporate misconduct.

From our simulation analysis of investing behaviour, we find that the market is more volatile when informed investors are more pessimistic about the investment intentions of uninformed investors and also in the situation where informed investors are more confident about the precision of their information.

The remainder of this chapter is organised as follows. In Sect. 2, the finance literature pertaining to bubbles is discussed. In Sect. 3 the proposed modelling approach with rational expectation equilibrium (REE) is explained. In this section we also present a rational expectation model with a learning effect extension (REE-LE) where informed investors learn from the behaviour of other investors. Based upon the baseline model of Ozdenoren and Yuan [36] we develop a new model which describes how informed investor’s rational guess at the demand of uninformed investors affects market demand. In Sect. 4 the simulation results of the learning effect extension are presented. Sect. 5 concludes the chapter.

7.2 Literature Review

Several attempts at defining financial bubbles have been made. For instance, Kindleberger [27, p.16] defines a bubble as a rising price movement over a period of time which then explodes. Shiller [40] concludes that the term is best thought of as referring to

a situation in which news of price increases spurs investor enthusiasm which spreads by psychological contagion from person to person, in the process amplifying stories that might justify the price increase and bringing in a larger and larger class of investors, who, despite doubts about the real value of the investment, are drawn to it partly though envy of other's successes and partly through a gambler's excitement.

However, empirical research on bubbles tends to focus upon a certain bubble or disaster and in consequence a precise definition may not be critical. Duckenfield [15] and Kindleberger and Aliber [28] provide an overview on a wide range of bubbles and subsequent crashes from a historical perspective.

Various explanations are put forward as to the cause of bubbles. For example, rational expectation bubbles (Blandhard and Watson [3], and Flood and Garber [16]), sunspot (or external uncertainty) (Cass and Shell [8]), irrational speculation (or famously 'irrational exuberance') (Shiller [37, 40]), risk-shifting between financial institutions and agents (Allen and Gale [1]), government conspiracy (Thompson and Hickson [42]), self-fulfilling effects (Cutler et al. [10], Flood and Hodrick [18], Subrahmanyam and Titman [41], and Morris and Shin [30]), herd behaviour (Camerer [5], De Long et al. [14], and Shiller [39]), feedback effects from price to return (Shiller [38, 40]), variation in stock supply (Hong et al. [25] and Ofek and Richardson [35]), information asymmetry (Grossman [19] and Grossman and Stiglitz [20]), and coordination (Angeletos and Werning [2], Camerer [5], Morris and Shin [30, 33], and Ozdenoren and Yuan [36]). It must, however, be stressed that each explanation as to the cause of bubbles tends to have resonance for specific kinds of bubbles and may not offer a general explanation for all. Camerer [5] classifies bubbles under three headings

1. fundamental bubbles,
2. fads, and
3. information bubbles.

The fundamental viewpoint, as articulated by Blanchard and Watson [3], Flood and Garber [16], and Hahn [21], analysed bubbles by assuming that assets and agents have unlimited life. However, in the real world, the life time of assets and agents are not infinite. Sunspot theory attempted to explain fundamental bubbles in terms of external uncertainties. Hong et al. [25] and Ofek and Richardson [35] argue that the Internet bubble was an example of a fundamental bubble and was caused by variation in stock supply. They maintain that the expiry of the insider lockup period of Internet companies flooded NASDAQ with excess stocks from insiders. It must be emphasised, however, that this branch of research tends to be focused on a specific bubble. More specifically, stock supply variation can help explain the Internet bubble but may not have general applicability for all bubbles. Hong and Stein [26] take the view that the Internet bubble might well be better classified as a 'fad'. They suggest that the increase supply of Internet stocks occurred because of a change in the confidence of investors which to some extent was triggered by their heterogeneous beliefs. Information bubbles may also be caused by the heterogeneous beliefs of investors which in turn may cause prices to deviate from their fundamental value. One example of an information bubble in action is

that of a rumour significantly impacting upon price with price eventually returning to a fundamental value when the rumour is proved unfounded.

In response to the causes of fads and information bubbles feedback effect, that is where the expected cash flows are affected by its market price (Subrahmanyam and Titman [41]), attracts significant attention. Feedback effects are seen in economics, for example, between income and anticipated saving (Campbell [6]), Keynes' general theory assumes this relationship to be linear. Feedback effects have also been demonstrated to exist in stock markets (De Long et al. [14], Hirshleifer et al. [24], and Subrahmanyam and Titman [41]), and in currency and futures markets (Cutler et al. [10]). The feedback effect has been analysed in terms of the role of speculative informed investors, uninformed investors and irrational investors.

In stock markets, the feedback effect contributes to high volatility (Shiller [40]) and forms a backward-bending demand curve (Yuan [44]). Moreover, feedback usually occurs in conjunction with herd behaviour (Camerer [5]). Subrahmanyam and Titman [41] regard the feedback effect as a network externality or complementarity (Bulow et al. [4]) among stakeholders, because higher return can be attained when more investors are involved. Moreover, self-fulfilling beliefs, which lead to coordination, between investors are attached to the formation of the feedback effect (Cutler et al. [10], Morris and Shin [30, 32], and Subrahmanyam and Titman [41]). It is intuitively easy to understand that people act based on their beliefs and the investment decision is the best indicator of the beliefs of investors. With network externalities, the more investors coordinate the more they benefit from their investment (Camerer [5], De Long et al. [14], and Subrahmanyam and Titman [41]) and thus in our examination of feedback effects we should concentrate on the coordination behaviour of investors.

With insight from coordination game theory, in which players in a game benefit most from investment simultaneously, Carlsson and van Damme [7] developed global game theory by introducing a noise term into the payoff structure. The noise in their model is a random variable drawn from a given set of incidents and it seems that the situation is more relevant to the real world. Moreover, Morris and Shin [31, 32, 33] argue that the uncertainty of beliefs and multi-equilibrium can be seen as a result of two modelling assumptions in traditional models. The two assumptions are a common knowledge of fundamentals and that economic agents are certain about the behaviour of other agents in equilibrium. In this way, Morris and Shin extend the global game by making the incomplete information spread over a distribution, for instance, the normal distribution. This body of work has been further developed and extended by authors such as Angeletos and Werning [2], Morris and Shin [34], and Ozdenoren and Yuan [36].

An important element to the feedback effect is that of a learning effect. Observation suggests that informed investors may learn or guess the aggregate demand of uninformed investors according to experience (Tversky and Kahneman [43]) or learning (Morris [29]) from historical information. The learning effect in conjunction with the feedback effect within the coordination game can cause price to deviate from its fundamental value as in Daniel et al. [11]. In this study, the lagged response was due to an irrational investment decision centred around a failure to respond immediately to price changes. Taking these factors together, our objective is to answer the question '*to what extent does investor behaviour affect the feedback effect and the equilibrium in a bubble*'.

7.3 Rational Expectation Equilibrium with Learning Effect (REE-LE)

In this section, by applying the feedback modelling approach with the coordination game of Ozdenoren and Yuan [36] we propose a Rational Expectation Equilibrium model with learning effect (or REE-LE, hereafter) to describe how informed investors' rational guess at the demand of uninformed investors affects the market demand. In this model, uninformed investors behave innocently, while the informed determine their investment based on their private information, the fundamental value of the innovation, and an expectation of the demand of the uninformed.

Based on the Rational Expectation Equilibrium (REE) model of Grossman and Stiglitz [20] and Hellwig [22] as well as the global game theory developed by Carlsson and van Damme [7] and Morris and Shin [30, 31, 32, 33], Ozdenoren and Yuan [36] create a one-stage model to explain how information affects the equilibrium of a market composed of informed and uninformed investors. In the model setting of REE, two kinds of assets and two types of traders are introduced. The informed investors who have private information about the return on the risky asset which then enables them to have an advantage over uninformed investors. The information technology, i.e. private information, used by the informed traders comes with noise and makes the price of the risky asset follow a normal distribution over a certain range. Investors are assumed to maximise their expected trading utility which is assumed to have a coefficient of absolute risk aversion identified as ρ (see, for instance, Hirshleifer and Riley [23, pp. 83–88]). On the other hand, uninformed investors receive public information, i.e. the market price, as their investment guidance.

In a global game, the decision making of players is correlated which makes the pay-offs of every player depend on the decision of other participants (Carlsson and van Damme [7]). Morris and Shin improve the global game model by injecting a noise into the game and make the payoff of each participant in a game distribute over a range. Through calculating the indifference curve, or the signal function, of the decision makers, the individual and market demand curves can be derived.

The signal setting of Ozdenoren and Yuan [36] is based on the work of Angeletos and Werning [2] and Morris and Shin [34]. Angeletos and Werning [2] have introduced an endogenous public signal into a global game model and find price multiplicity is robust in a separate market. Morris and Shin [34] find that private signals carry more information than public signals. However, Ozdenoren and Yuan [36] found that the trade-off between cost and incentive of coordination moderates the strategic complementarities. The effect of strategic complementarities is further debilitated by the prediction behaviour of informed investors in our model.

We introduce the investor learning effect as an adjustment to the baseline model of Ozdenoren and Yuan [36]. Whenever an informed investor has a motivation to earn a learning premium (Morris [29]), she would like to guess or predict the behaviour of uninformed investors before making any decision. So, it makes sense to model the informed investor as having a 'rational guess' about the demand of the uninformed and the aggregate demand can be determined by adding the demand of the two parties. Our simulation results show that no matter whether the prediction is correct or not, this behaviour reduces both price multiplicity and strategic complexities.

7.3.1 Assets

Consider an economy with one risk-free bond and one risky asset. The risk-free bond works as a numéraire and its price is always one and the risk-free rate is zero. The risky asset has an aggregate supply of $M > 0$ and its risky terminal payoff is

$$\tilde{V} + f(X, \tilde{\theta}).$$

In the payoff function, \tilde{V} stands for the payoff from the regular operation of the company without R&D. We let \tilde{V} equal $\bar{V} + \sigma_v \tilde{\epsilon}_v$, where σ_v is a positive constant, and $\tilde{\epsilon}_v$ is standard normal (with zero mean and unit standard deviation). In this study, we define $\bar{V} = 0$ for convenience. It should be noted that our results can be extended to cases whenever $\bar{V} \neq 0$. $f(X, \tilde{\theta})$ denotes the stochastic payoff from innovation, where X is the amount invested by informed investors in the risky asset and $\tilde{\theta}$ is the fundamental value of the innovation with a uniform distribution over a range of real numbers. We assume $f(X, \tilde{\theta})$ is positively related to X and θ .

The feedback effect from the demand to the terminal value is captured in $f(X, \tilde{\theta})$. As highlighted by Ozdenoren and Yuan [36] ‘if managers learn from informed investors in making real investment decisions, then their decisions, and in turn the terminal value of the risky asset, will be affected by the investment from informed investors, X , which aggregates heterogenous private information from informed investors’.

7.3.2 Investors

There are two types of investors, namely, informed and uninformed investors. Informed investors belong to a measure-one continuum, indexed by $i \in [0, 1]$. They employ an information technology: $\tilde{s}_i = \tilde{\theta} + \beta E[L] + \sigma_s \epsilon_i$, where \tilde{s}_i is the noisy private signal being observed at time 0 about $\tilde{\theta}$, β is the learning factor informed investors rationally guess of the demand of uninformed investors, which is denoted by L , and $\tilde{\epsilon}_i$ is assumed to be uniformly distributed on $[-1, 1]$ with density h , and σ_s is a standard normal distribution with mean 0 and precision 1. The learning effect is assumed to be linear to signal for simplicity. Conditional on $\tilde{\theta}$, noisy private signals, \tilde{s}_i , are independently identically distributed across informed investors.

The amounts traded of the informed investment is assumed to be restricted to $x(i) \in [0, z]$, where z is a fixed number and $z \geq 1$. This position limit is adapted to represent the fact that informed investors may face capital or borrowing constraints. The utility of informed investors from buying $k \in [0, z]$ units of the asset is $k(f(X, \tilde{\theta}) - P)$, where $f(X, \tilde{\theta})$ is the dividend payoff from the asset and P is the price of the asset. As informed investors are risk neutral, they invest either up to the position limit of z or zero. The aggregate demand of the informed investor, X , is the sum of the position of all informed investors, that is: $X = \int_0^1 x(i) di$. Uninformed investors belong to a measure- w continuum and apply a mean-variance strategy with risk aversion parameter, ρ . Their demand is

$$\begin{aligned} L(P) &= \frac{w(E[\tilde{V}] - P)}{\rho \text{Var}(\tilde{V})} \\ &= \frac{E[\tilde{V}] - P}{\lambda}, \end{aligned} \tag{7.1}$$

where λ is the slope of this uninformed demand curve. Finally, the noisy demand is assumed to be $\sigma_y \tilde{y}$ and represents a noisy shock in the market. This shock represents the information aggregation process and prevents the market-clearing price from fully revealing the fundamentals (Grossman and Stiglitz [20]). In the noisy shock, $\sigma_y > 0$ and \tilde{y} is a standard normal random variable independent of $\tilde{\epsilon}_i$, $\tilde{\theta}$, and $\tilde{\epsilon}_i$ for every i .

7.3.3 Equilibrium

The market equilibrium is defined as:

Definition 7.1 (Equilibrium). *An equilibrium includes: (1) a price function: $P(\tilde{\theta}, \tilde{y})$, (2) strategies: $\pi(\tilde{s}_i, P) : \mathbf{R} \rightarrow [0, 1]$, and (3) the corresponding aggregate demands, $L(P)$ and $X(P, \tilde{\theta})$, such that:*

- *Informed investor i : $\pi(\tilde{s}_i, P) \in \arg \max_{\pi} z \pi E[f(X(P, \tilde{\theta}), \tilde{\theta}) - P | \tilde{s}_i = s_i, P]$*
- *Uninformed investor: $L(P) = \frac{\omega(E[\tilde{V}] - P)}{\rho \text{Var}(\tilde{V})} = \frac{E[\tilde{V}] - P}{\lambda}$*
- *The market clearing condition: $X(P, \tilde{\theta}) + L(P) + \sigma_y \tilde{y} = M$*

The information technology which enables informed investors to learn is $\tilde{s}_i = \tilde{\theta} + \beta E(L) + \sigma_s \tilde{\epsilon}_i$, where $\tilde{\epsilon}_i$ is uniformly distributed on $[-1, 1]$. Informed investors react to the potential aggregate demand of uninformed investors by increasing their threshold by $\beta E(L)$, where $E(L)$ is the estimate of L by informed investors adjusted by a parameter β . The monotone equilibrium with cutoff strategies is met when $\pi(\tilde{s}_i, P) = 1$ if $\tilde{s}_i \geq g(P)$ for some function $g(P)$, and $\pi(\tilde{s}_i, P) = 0$ otherwise. The investment of the informed investor i occurs only when her private signal, \tilde{s}_i is greater than or equal to a specific threshold, $g(P)$, and vice versa. M is the supply of the risky asset.

Assuming that all informed investors conform to a cutoff strategy; in other words, they purchase iff $\tilde{s}_i = \tilde{\theta} + \beta E(L) + \sigma_s \tilde{\epsilon}_i \geq g(P)$, or equivalently, $\tilde{\epsilon}_i \geq (g(P) - \tilde{\theta} - \beta E(L))/\sigma_s$. According to the relative strength of \tilde{s}_i and $g(P)$, there are three possible positions an informed investor might hold. First, all informed investors will hold a position z when the signal received is definitely stronger than their cutoff strategy, i.e. $\tilde{\theta} + \beta E(L) > g(P) + \sigma_s$. Second, investors will withdraw when the signal is too weak, when $\tilde{\theta} + \beta E(L) < g(P) - \sigma_s$. Finally, when $g(P) - \sigma_s \leq \tilde{\theta} + \beta E(L) \leq g(P) + \sigma_s$, the proportion of invested investors with position z is $(1 - (g(P) - \tilde{\theta} - \beta E(L))/\sigma_s)/2$. The monotone equilibrium of informed investors with learning is

$$X(P, \tilde{\theta}) = \begin{cases} 0 & \text{if } \tilde{\theta} + \beta E(L) < g(P) - \sigma_s \\ \frac{z}{2} \left(1 - \frac{g(P) - \tilde{\theta} - \beta E(L)}{\sigma_s} \right) & \text{if } g(P) - \sigma_s \leq \tilde{\theta} + \beta E(L) \leq g(P) + \sigma_s \\ z & \text{if } \tilde{\theta} + \beta E(L) > g(P) + \sigma_s \end{cases} \quad (7.2)$$

Uninformed investors' aggregate demand $L(P)$ is substituted into the market clearing condition. Under the assumption of REE, $E(L) = L$, the price becomes

$$P = \lambda X + \lambda \sigma_y \tilde{y} - \lambda M, \quad (7.3)$$

where $\lambda = (\rho \text{Var}(\tilde{V}))/w$ is the price impact of a marginal aggregate demand change. Substitute Eq. 7.3 into Eq. 7.2, and the market clearing prices become

$$P = \begin{cases} \lambda\sigma_y\tilde{y} - \lambda M & \text{if } \tilde{\theta} < g(P) - \sigma_s - \beta L \\ \frac{2\sigma_s}{z\beta + 2\sigma_s} \left(\frac{z\lambda}{2} \left(1 - \frac{g(P) - \tilde{\theta}}{\sigma_s} \right) + \lambda\sigma_y\tilde{y} - \lambda M \right) & \text{if } g(P) - \beta L - \sigma_s \leq \tilde{\theta} \leq g(P) - \beta L + \sigma_s \\ \lambda z + \lambda\sigma_y\tilde{y} - \lambda M & \text{if } g(P) + \sigma_s - \beta L < \tilde{\theta} \end{cases} \quad (7.4)$$

In the second part of Eq. 7.4, a statistic of $\tilde{\theta}$ can be derived

$$\tau \equiv \left(\frac{2\sigma_s + \beta}{z\lambda} P + \frac{2\sigma_s}{z} M + g(P) - \sigma_s \right) = \tilde{\theta} + \frac{2\sigma_y\sigma_s}{z} \tilde{y}, \quad (7.5)$$

which is a normal distribution with mean θ and standard deviation $2\sigma_y\sigma_s/y$.

The statistic τ is a public signal the same as the statistics in Ozdenoren and Yuan [36] with the same precision and mean. This means that τ is endogenous and is not affected by private information or the expectation of informed investors. With τ and the following monotone equilibrium condition, we can derive the cutoff strategy of informed investors. The cutoff strategy can be characterised by the following proposition as in Ozdenoren and Yuan [36].

Proposition 7.1 (The Monotone Equilibrium of Informed Investors). *The equilibrium described in Definition 1 can be characterised as:*

- *informed participants' unique monotone equilibrium strategy: there is a unique function $g : \mathbf{R} \rightarrow \mathbf{R}$ such that informed investors' equilibrium strategies are given by $\pi(\tilde{s}_i, P) = 1$ if $s_i \geq g(P)$*
- *informed investors' aggregate demand, $X(P, \tilde{\theta})$, can be uniquely determined by Eq. 7.2 and uninformed investors' demand can be uniquely defined by $L(P) = (E[\tilde{V}] - P)/\lambda$, given a market clearing price P and any monotone equilibrium*
- *the market equilibrium price $P(\tilde{\theta}, \tilde{y})$ satisfies Eq. 7.4.*

Proposition 1 identifies the uniqueness of the demand at a given price from informed investors based on a given fundamental situation. The multiplicity of equilibrium prices only occurs when Eq. 7.4 has multiple solutions at a given demand investment. This region is the backward-bending region, in which higher stock price attracts more investment, in Ozdenoren and Yuan [36]. The next subsection details the equilibrium of the cutoff strategy and the aggregate demand of a linear payoff-demand-risk relationship.

7.3.4 The Cutoff Strategy and the Aggregate Demand

The cutoff strategy function must be determined before the aggregate demand can be calculated. To demonstrate the effect of the learning factor on the equilibrium demand and on the feedback effect without losing generality, the dividend function with feedback effect is defined as $f(X, \tilde{\theta}) = \alpha X + \theta$. The following lemma, as detailed in Ozdenoren and Yuan [36] describes informed investors' equilibrium strategy.

Lemma 7.1 (Equilibrium Cutoff Strategy of Informed Investors). *The equilibrium cutoff strategy, $g(P)$, is unique in the region when the fundamental cannot determine the equilibrium with dividend payoff function $f(X, \tilde{\theta}) = \alpha X + \theta$, is:*

$$g(P) = P + \sigma_s - \left(\alpha + \frac{2\sigma_s}{z}\right) \left(\frac{P}{\lambda} + \frac{(1-z)\beta P}{2\lambda\sigma_s} + M - \sigma_y E\left[u \left| \frac{\frac{P}{\lambda} + M + \frac{(1-z)\beta P}{2\lambda\sigma_s} - z}{\sigma_y} \leq u \leq \frac{\frac{P}{\lambda} + M + \frac{(1-z)\beta P}{2\lambda\sigma_s}}{\sigma_y} \right| \right]\right). \quad (7.6)$$

From Eq. 7.6, the aggregate demand can be detailed as follows.

Lemma 7.2 (Aggregate Demand with Learning Effect). *The aggregate demand with signal in Lemma 1 can be calculated by adding the aggregate demands of informed investors, $X(P, \tilde{\theta})$, and uninformed investors, $L(P)$, as:*

$$AD(P, \tilde{\theta}) = \begin{cases} -\frac{P}{\lambda} & \text{if } \tilde{\theta} + \beta E(L) < g(P) - \sigma_s \\ \frac{z}{2} \left(1 - \frac{g(P) - \tilde{\theta} + \frac{\beta P}{\lambda}}{\sigma_s}\right) - \frac{P}{\lambda} & \text{if } g(P) - \sigma_s \leq \tilde{\theta} + \beta E(L) \leq g(P) + \sigma_s \\ \frac{P}{z - \frac{P}{\lambda}} & \text{if } \tilde{\theta} + \beta E(L) > g(P) + \sigma_s \end{cases} \quad (7.7)$$

7.4 The Simulation Results

This section presents the simulation results of the learning effect extension. The learning effect simulations are conducted using the Monte Carlo technique with both 50 and 200 repetitions. Simulation results are obtained for both and are similar, hence we only report result for 50 repetitions.

The Learning Effect Equilibrium

Ozdenoren and Yuan [36] define the cut-off strategy as the situation where the informed investor is indifferent between making the investment decision and deciding not to invest. Based on Eq. 7.6 and Eq. 7.7, the cutoff functions and the aggregate demands with learning effect are simulated. The experiment results of cutoff functions are depicted in Fig. 7.1 and the aggregate demands in Fig. 7.2.

In Fig. 7.1, the valid range of the cutoff function expands and becomes flatter as β increases. This implies that the more confident informed investors are that the uninformed will follow their investment strategy, then the smaller is the required signal variation to encourage informed investors to make the investment decision. In essence, this emphasises that informed investors are more optimistic about market outcomes and take more aggressive investment decisions when they expect more uninformed investors to follow the same strategy. Of course, the contrary also applies with informed investors being more cautious and needing more precise information when they expect the uninformed not to follow their actions, i.e. β is smaller.

As depicted in the signal function, aggregate demand decreases and price falls as β increases. In Fig. 7.2 the aggregate demand shrinks and becomes more elastic with higher values of β . When the expectation of informed investors is that demand by uninformed investors is likely to be higher then the aggregate demand for the stock will be lower at each and every price. Having made this point, it is also evident from Fig. 7.2 that the price multiplicity range is largest when β is negative. This implies that when

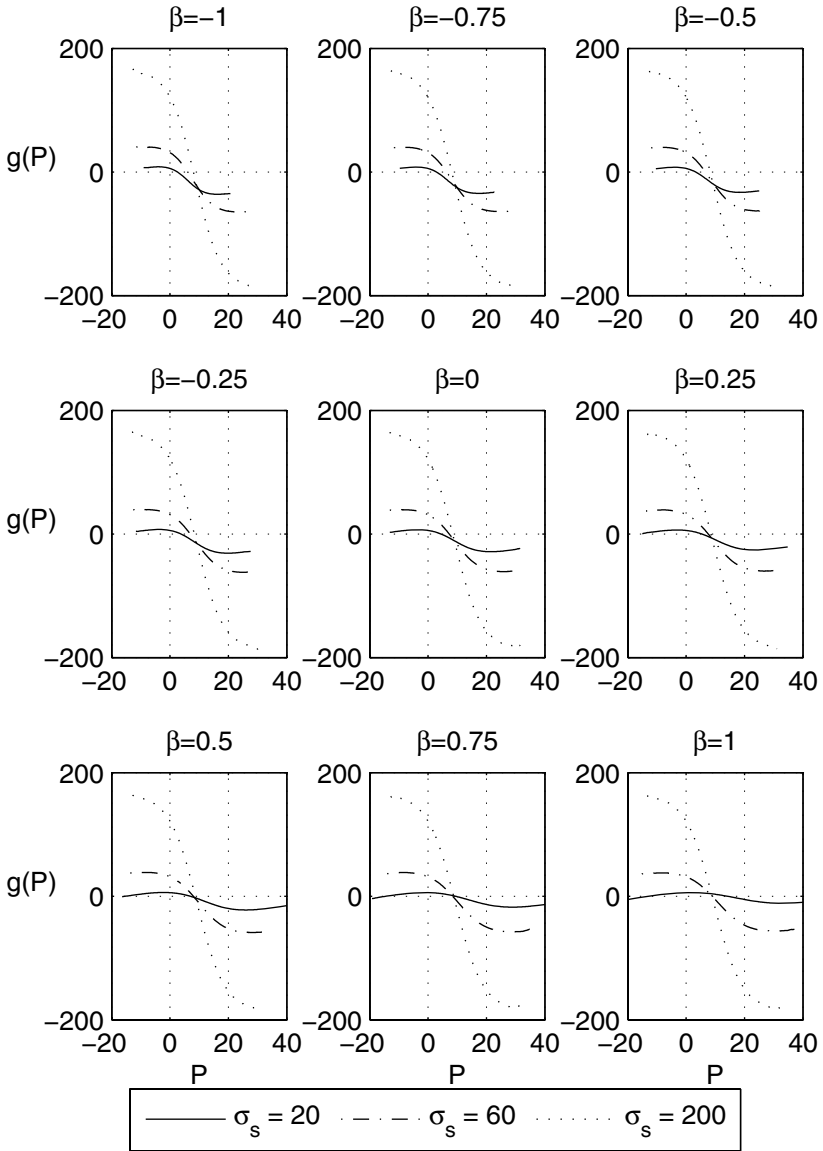


Fig. 7.1. Equilibrium cutoff strategies with learning effect. This figure shows that the cutoff strategy values with different private signals under $\beta \in \{-1, -0.75, -0.5, -0.25, 0, 0.25, 0.5, 0.75, 1\}$. The signals of informed investor are more precise ($\sigma_s = 20$), less precise ($\sigma_s = 60$), and very noisy ($\sigma_s = 600$), respectively. The parameters are $\sigma_y = 4$, $\alpha = 2$, $z = 20$, $\lambda = 1$, and $M = 1$. The above diagrammatic representations illustrate that the cutoff strategy curve expands as β gets larger. The signal strength ranges of the cut-off strategy are more condensed as the precision of private information increases. The X-axes represent the price of the risky asset and the Y-axes are the signal strength. Note that the signal range is from -200 to 200.

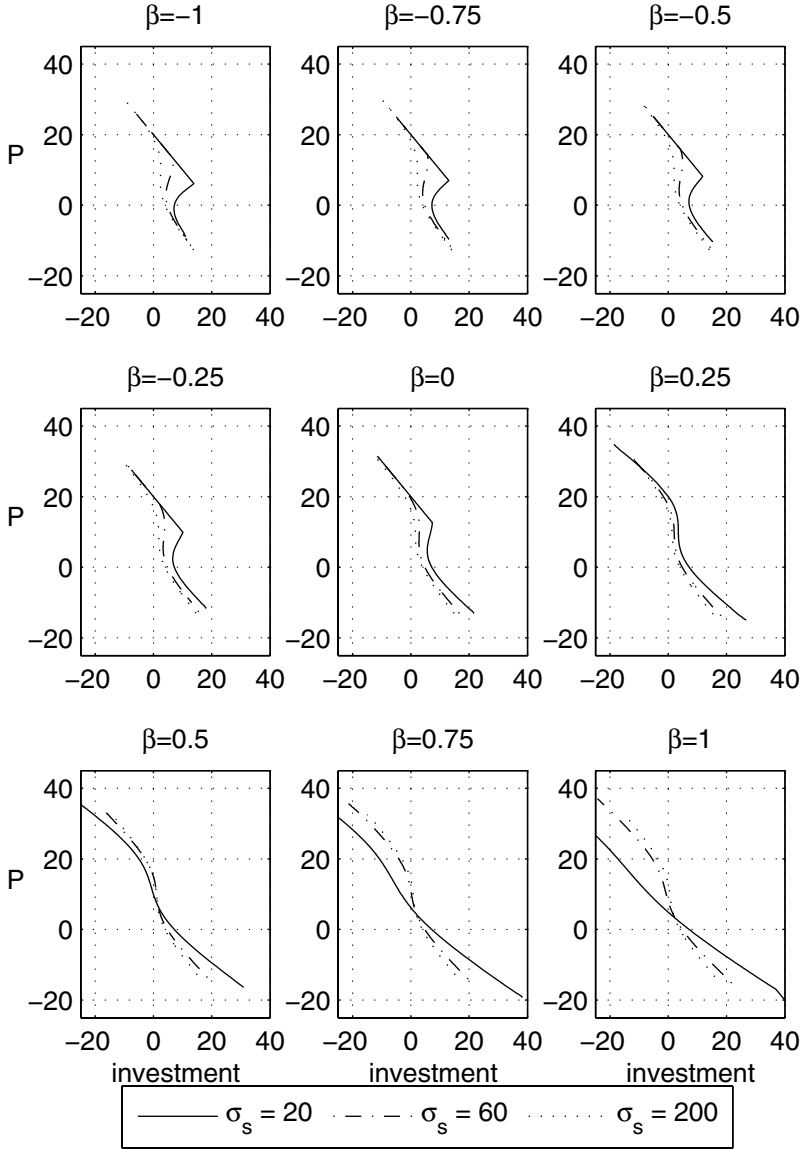


Fig. 7.2. Aggregate demand with learning effect. The simulated demand curves are dependent upon the learning factor $\beta \in \{-1, -0.75, -0.5, -0.25, 0, 0.25, 0.5, 0.75, 1\}$ and three different precision values of the private signal available to informed investors, that is more precise ($\sigma_s = 20$), less precise ($\sigma_s = 60$), and very noisy ($\sigma_s = 600$), respectively. The parameters are $\sigma_y = 4, \alpha = 2, z = 20, \lambda = 1$, and $M = 1$. The diagrammatic representation demonstrates that price multiplicity is more apparent the smaller the value of β . The demand curves also expands and becomes flatter as β increases.

informed investors are less confident of the actions of uninformed investors there is a greater spread of potential equilibrium prices. It is also the case that when the private signal, \tilde{s}_i , is more precise, price multiplicity is more pronounced. This finding shows that a bubble, which happens when the economy is more prosperous with price multiplicity, is more prominent when informed investors act counter to uninformed investors.

The Feedback Effect

On the other hand, the range of cutoff signals expands as σ_s increases as less precise private information makes it difficult for the informed investor to make a decision based on private information. Ozdenoren and Yuan [36] argue that when private information is precise, the information effect on the stock price will be more pronounced. This makes market demand more volatile and in consequence there are multiple market equilibrium prices for any given level of investment.

7.5 Conclusion

Financial bubbles exist when demand in the market is high and there is a high probability of price multiplicity. From the above discussion, we find that both the learning effect and the feedback effect contribute to price multiplicity. The manner in which they impact upon price multiplicity is, however, different. Price multiplicity is observed when informed investors have a higher feedback effect, that is when informed investors have more precise private information. However, the learning effect characterised by β is strong when informed investors act counter to the price signal. An example is the situation where informed investors endeavour to profit by manipulating price in an opposite direction to that suggested by the private information available to them, uninformed investors follow their actions, prices rise and the informed offload the stock and make profit at the expense of the uninformed investor. On the other hand, the aggregate demand is higher when private information is more precise and the learning effect is negative.

The experiments also show that the learning effect makes the market depressed when the learning effect is very prominent, especially when the private information obtained by informed investors is with higher precision. This could be due to a stronger substitute effect mitigating the learning effects and the feedback effects (Ozdenoren and Yuan [36]). Another explanation for the less persuasive impact of the learning effect might be that of investor coordination where all members can profit by adopting the same investment strategy. This is similar to the scenario where all investors can make a profit in a period of high confidence and prosperity. However, as the market supply is not elastic in our model, the aggregate demand is not able to reflect the corresponding coordination effect.

The cross-effect between the learning effect and the feedback effect is not considered in this analysis, however, this may be an important future development as it may provide insight to phenomena such as the relationship between feedback and herd behaviour. A further extension of the model may be found within the general area of behavioural finance. For instance, Shiller [39] maintained that group thinking and irrational behaviour may cause bubbles. De Bondt [12, 13] suggest that under- and over-reaction contributes

to information bubbles. De Bondt argues that investors tend to overreact when making a decision based on their own belief but under-react to immediate and large price fluctuations. Incorporating insights from the work of these authors in behavioural finance coupled with the learning effect introduced in the present model may result in improved models and consequently a better understanding of financial bubbles.

References

- [1] Franklin Allen, F., Gale, D.: Bubble and crisis. *Economic Journal* 110, 236–255 (2000)
- [2] Angeletos, G.M., Werning, I.: Crises and prices – information aggregation, multiplicity and volatility. *The American Economic Review* 96, 1720–1736 (2006)
- [3] Blanchard, O., Watson, M.: Bubbles, Rational Expectations and Financial Markets. In: *Crises in the Economic and Financial Structure: Bubbles, Bursts and Shocks*, Lexington (1982)
- [4] Bulow, J., Geanakoplos, J., Klemperer, P.: Multimarket oligopoly: strategic substitutes and complements. *The Journal of Political Economy* 93(3), 488–511 (1985)
- [5] Camerer, C.: Bubbles and fads in asset prices. *Journal of Economic Surveys* 3, 3–41 (1989)
- [6] Campbell, J.: Does saving anticipate declining labor income? an alternative test of the permanent income hypothesis. *Econometrica* 55, 1249–1273 (1987)
- [7] Carlsson, H., van Damme, E.: Global games and equilibrium selection. *Econometrica* 61, 989–1018 (1993)
- [8] Cass, D., Shell, K.: Do sunspots matter? *The Journal of Political Economy* 91, 193–227
- [9] Cooper, R.: *Coordination Games: Complementarities and Macroeconomics*. Cambridge University Press, Cambridge (1999)
- [10] Cutler, D., Poterba, J., Summers, L.: Speculative dynamics and role of feedback traders. *The American Economic Review* 80, 63–68 (1990)
- [11] Daniel, K., Hirshleifer, D., Subrahmanyam, A.: Investor psychology and security market under- and overreactions. *The Journal of Finance* 53(6), 1839–1885 (1998)
- [12] De Bondt, W.: The Psychology of Underreaction and Overreaction in World Equity Markets. In: De Bondt, W. (ed.) *Security Market Imperfections in World Wide Equity Markets*, pp. 65–89. Cambridge University Press, Cambridge (2000)
- [13] De Bondt, W.: Bubble Psychology. In: *Asset Price Bubbles*, pp. 205–216. MIT Press, Cambridge (2003)
- [14] De Long, J., Shleifer, A., Summers, L., Waldmann, R.: Positive feedback investment strategies and destabilizing rational speculation. *The Journal of Finance* 45(2), 379–395 (1990)
- [15] Duckenfield, M.: *History of financial disasters 1763–1995*. Pickering & Chatto (2006)
- [16] Flood, R., Garber, P.: Market fundamentals versus price-level bubbles: The first tests. *The Journal of Political Economy* 88, 745–770 (1980)
- [17] Flood, R., Garber, P.: Gold monetization and gold discipline. *The Journal of Political Economy* 92(1), 90–107 (1984)
- [18] Flood, R., Hodrick, R.: On testing for speculative bubbles. *The Journal of Economic Perspectives* 4(2), 85–101 (1990)
- [19] Grossman, S.: On the efficiency of competitive stock markets where traders have diverse information. *The Journal of Finance* 31, 573–585 (1976)
- [20] Grossman, S., Stiglitz, J.: On the impossibility of informationally efficient markets. *The American Economic Review* 70(3), 393–408 (1980)

- [21] Hahn, F.: Equilibrium dynamics with heterogenous capital goods. *The Quarterly Journal of Economics* 80, 633–646 (1966)
- [22] Hellwig, M.: On the aggregation of information in competitive markets. *Journal of Economic Theory* 22, 477–498 (1980)
- [23] Hirshleifer, J., Riley, J.: *The analytics of uncertainty and information*. Cambridge University Press, Cambridge (1992)
- [24] Hirshleifer, D., Subrahmanyam, A., Titman, S.: Feedback and the success of irrational investors. *Journal of Financial Economics* 81, 311–338 (2006)
- [25] Hong, H., Scheinkman, J., Xiong, W.: Asset float and speculative bubbles. *The Journal of Finance* 61(3), 1073–1117 (2006)
- [26] Hong, H., Stein, J.: Differences of opinion, short-sales constraints, and market crashes. *The Review of Financial Studies* 16(2), 487–525 (2003)
- [27] Kindleberger, C.: *Manias, panics, and crashes: a history of financial crises*. Macmillan, Basingstoke (1978)
- [28] Kindleberger, C., Aliber, R.: *Manias, Panics, and Crashes: A History of Financial Crises*, 5th edn. John Wiley & Sons Ltd., Chichester (2005)
- [29] Morris, S.: Speculative investor behavior and learning. *The Quarterly Journal of Economics* 111(4), 1111–1133 (1996)
- [30] Morris, S., Shin, H.-S.: Unique equilibrium in a model of self-fulfilling currency attacks. *The American Economic Review* 88(3), 587–597 (1998)
- [31] Morris, S., Shin, H.-S.: Rethinking multiple equilibria in macroeconomic modelling. Technical report, Yale University & Oxford University (2000)
- [32] Morris, S., Shin, H.-S.: Social value of public information. *The American Economic Review* 102, 1521–1534 (2002)
- [33] Morris, S., Shin, H.-S.: Global Games: Theory and Applications. In: *Advances in Economics and Econometrics*, pp. 56–114. Cambridge University Press, Cambridge (2003)
- [34] Morris, S., Shin, H.-S.: Endogenous public signals and coordination. *Levine's bibliography*, UCLA Department of Economics (April 2006), <http://ideas.repec.org/p/cla/levrem/1222470000000001309.html>
- [35] Ofek, E., Richardson, M.: Dotcom mania: The rise and fall of internet stock prices. *The Journal of Finance* 58(3), 1113–1137 (2003)
- [36] Ozdenoren, E., Yuan, K.: Feedback effects and asset prices. *The Journal of Finance* (forthcoming, 2008)
- [37] Shiller, R.: Do stock prices move too much to be justified by subsequent changes in dividends? *The American Economic Review* 71, 421–436 (1981)
- [38] Shiller, R.: Market volatility and investor behavior. *The American Economic Review* 91, 58–62 (1990)
- [39] Shiller, R.: Conversation, information, and herd behavior. *The American Economic Review* 85, 181–185 (1995)
- [40] Shiller, R.: *Irrational Exuberance*, 2nd edn. Princeton University Press, Princeton (2005)
- [41] Subrahmanyam, A., Titman, S.: Feedback from stock prices to cash flows. *The Journal of Finance* 56(6), 2389–2413 (2001)
- [42] Thompson, E., Hickson, C.: Predicting bubbles and bubble-substitutes. *Global Business and Economics Review* 8(3–4), 217–246 (2006)
- [43] Tversky, A., Kahneman, D.: Judgement under uncertainty: Heuristics and biases. *Science* 185, 1124–1131 (1974)
- [44] Yuan, K.: Asymmetric price movements and borrowing constraints: A rational expectations equilibrium model of crises, contagion, and confusion. *The Journal of Finance* 60(1), 379–411 (2005)

A Appendix

A.1 Proof of Proposition 1

In the proof of this proposition, the notation θ refers to $\tilde{\theta}$, and L to $L(P)$. To check the monotonicity of the informed strategy, the informed conditional expectation with learning of the risky asset's dividend payoff is expressed as the equation

$$\begin{aligned} & \int_{-\infty}^{\infty} f(X(P, \theta), \theta) h(\theta|s, P) d\theta = \\ & \int_{-\infty}^{g(P) - \sigma_s - \beta L} f(X(P, \theta), \theta) h(\theta|s, P) d\theta + \\ & \int_{g(P) - \sigma_s - \beta L}^{g(P) + \sigma_s - \beta L} f(X(P, \theta), \theta) h(\theta|s, P) d\tilde{\theta} + \\ & \int_{g(P) + \sigma_s - \beta L}^{\infty} f(X(P, \theta), \theta) h(\theta|s, P) d\theta, \end{aligned} \quad (7.8)$$

where s is the private signal with learning parameter β , price P , and the density function $h(\theta|s, P)$ of $\tilde{\theta}$ conditional on $\tilde{s} = s$ and P . First, the first term on the right hand side of the Eq. 7.8 can be broken into the following equation by Bayes rules (i.e. $f(x, y) = f(x|y)f(y)$)

$$\begin{aligned} & \int_{-\infty}^{g(P) - \sigma_s - \beta L} f(X(P, \theta), \theta) h(\theta|s, P) d\theta \\ &= \Pr(\theta < g(P) - \sigma_s - \beta L | s, P) \\ & \int_{-\infty}^{g(P) - \sigma_s - \beta L} f(X(P, \theta), \theta) h(\theta|s, P, \theta < g(P) - \sigma_s - \beta L) d\theta. \end{aligned}$$

As $\theta \in [s - \sigma_s, s + \sigma_s]$,

$$\Pr(\theta < g(P) - \sigma_s - \beta L | s, P) = \frac{\min\{s + \sigma_s, g(P) - \sigma_s - \beta L\} - (s - \sigma_s)}{2\sigma_s},$$

under the assumption that $g(P) > s - \sigma_s$. Furthermore, price is uninformative about θ in this range by Eq. 7.4, the posterior is uniform, and the pdf is

$$h(\theta|s, P, \theta < g(P) - \sigma_s - \beta L) = \frac{1}{\min\{s + \sigma_s, g(P) - \sigma_s - \beta L\} - (s - \sigma_s)}$$

Then,

$$\begin{aligned} & \int_{-\infty}^{g(P) - \sigma_s - \beta L} f(X(P, \theta), \theta) h(\theta|s, P) d\theta \\ &= \frac{1}{2\sigma_s} \int_{s - \sigma_s}^{\min\{s + \sigma_s, g(P) - \sigma_s - \beta L\}} f(X(P, \theta), \theta) d\theta. \end{aligned}$$

With the same rule, the second term in Eq. 7.8 can be expressed as

$$\begin{aligned}
& \int_{g(P)-\sigma_s-\beta L}^{g(P)+\sigma_s-\beta L} f(X(P, \theta), \theta) h(\theta|s, P) d\theta \\
&= \Pr(g(P) - \sigma_s - \beta L \leq \theta \leq g(P) + \sigma_s - \beta L | s, P) \\
& \int_{g(P)-\sigma_s-\beta L}^{g(P)+\sigma_s-\beta L} f(X(P, \theta), \theta) h(\theta|s, P, g(P) - \sigma_s - \beta L \leq \theta \leq g(P) + \sigma_s - \beta L) d\theta.
\end{aligned}$$

Let

$$\tau \equiv \left(\frac{2\sigma_s + \beta}{z\lambda} P + \frac{2\sigma_s}{z} M + g(P) - \sigma_s \right).$$

As in Eq. 7.5, $\tau = \theta + (2\sigma_s \sigma_y / z)y$ is a sufficient statistic for the clearing price, P . Then the density function in this area is

$$\begin{aligned}
& h(\theta|s, P, g(P) - \sigma_s - \beta L \leq \theta \leq g(P) + \sigma_s - \beta L) \\
&= h(\theta|s, \tau, g(P) - \sigma_s - \beta L \leq \theta \leq g(P) + \sigma_s - \beta L) \\
&= \begin{cases} \frac{\phi(\frac{\theta-\tau}{2\sigma_s \sigma_y / z}) \frac{z}{2\sigma_s \sigma_y}}{\int_{[s-\sigma_s, s+\sigma_s] \cap [g(P)-\sigma_s-\beta L, g(P)+\sigma_s-\beta L]} \phi(\frac{\theta-\tau}{2\sigma_s \sigma_y / z}) \frac{z}{2\sigma_s \sigma_y} d\theta} & \text{if } \theta \in [s - \sigma_s, s + \sigma_s] \cap \\ & [g(P) - \sigma_s - \beta L, g(P) + \sigma_s - \beta L] \\ 0 & \text{otherwise} \end{cases}
\end{aligned}$$

So,

$$\begin{aligned}
& \int_{g(P)-\sigma_s-\beta L}^{g(P)+\sigma_s-\beta L} f(X(P, \theta), \theta) h(\theta|s, P) d\theta \\
&= \frac{||[s-\sigma_s, s+\sigma_s] \cap [g(P)-\sigma_s-\beta L, g(P)+\sigma_s-\beta L]||}{2\sigma_s} \\
& \frac{\int_{[s-\sigma_s, s+\sigma_s] \cap [g(P)-\sigma_s-\beta L, g(P)+\sigma_s-\beta L]} f(X, \theta) \phi(\frac{\theta-\tau}{2\sigma_s \sigma_y / z}) (\frac{z}{2\sigma_s \sigma_y}) d\theta}{\int_{[s-\sigma_s, s+\sigma_s] \cap [g(P)-\sigma_s-\beta L, g(P)+\sigma_s-\beta L]} \phi(\frac{\theta-\tau}{2\sigma_s \sigma_y / z}) (\frac{z}{2\sigma_s \sigma_y}) d\theta}.
\end{aligned}$$

Finally, the third term on the right hand side of Eq. 7.8 is

$$\begin{aligned}
& \int_{g(P)+\sigma_s-\beta L}^{\infty} f(X(P, \theta), \theta) h(\theta|s, P) d\theta \\
&= \frac{1}{2\sigma_s} \int_{\max(s-\sigma_s, g(P)+\sigma_s-\beta L)}^{s+\sigma_s} f(X(P, \theta), \theta) d\theta.
\end{aligned}$$

To solve the cutoff strategy, we assume that $s = g(P)$ is the cutoff signal a agent feels indifferent in investing or not. To this agent, the first term in the right hand side of Eq. 7.8 is

$$-\frac{1}{2\sigma_s} \int_{s-\sigma_s-\beta L}^{s-\sigma_s} f(X(P, \theta), \theta) d\theta, \quad (7.9)$$

and the third term in the same equation is

$$\frac{1}{2\sigma_s} \int_{s+\sigma_s-\beta L}^{s+\sigma_s} f(X(P, \theta), \theta) d\theta. \quad (7.10)$$

Since risky returns should be positive so that the dividend payoff (f) is monotonic increasing, and the summation of Eq. 7.9 and 7.10 is positively related to βL . However, fundamental value largely determines the investment strategy in these two areas such that Eq. 7.9 and 7.10 is small and can be neglected.

The indifference condition in the second term in Eq. 7.8 becomes

$$P' = \frac{\int_{g(P)-\sigma_s-\beta L}^{g(P)+\sigma_s-\beta L} f\left(\frac{z}{2}\left(1 - \frac{g(P)-\theta-\beta L}{\sigma_s}\right), \theta\right) \phi\left(\frac{\theta-\tau}{2\sigma_s\sigma_y/z}\right) d\theta}{\int_{g(P)-\sigma_s-\beta L}^{g(P)+\sigma_s-\beta L} \phi\left(\frac{\theta-\tau}{2\sigma_s\sigma_y/z}\right) d\theta},$$

where $P' \simeq P$ when the magnitude of Eq. 7.9 and 7.10 is small and they are monotonic. Assume that a value of $\kappa = g(P) - \beta L$ satisfies the indifference condition, such that

$$P' = \frac{\int_{\kappa-\sigma_s}^{\kappa+\sigma_s} f\left(\frac{z}{2}\left(1 - \frac{\kappa-\theta}{\sigma_s}\right), \theta\right) \phi\left(\frac{\theta-\kappa-\frac{2\sigma_s+\beta}{\lambda z}P - \frac{2\sigma_s}{z}M - \beta L + \sigma_s}{2\sigma_s\sigma_y/z}\right) d\theta}{\int_{\kappa-\sigma_s}^{\kappa+\sigma_s} \phi\left(\frac{\theta-\kappa-\frac{2\sigma_s+\beta}{\lambda z}P - \frac{2\sigma_s}{z}M - \beta L + \sigma_s}{2\sigma_s\sigma_y/z}\right) d\theta}.$$

With a change of variables, $x = \frac{\theta-\kappa}{\sigma_s}$, the above equation becomes

$$P' = \frac{\int_{-1}^1 f\left(\frac{z}{2}(1+x), \kappa + \sigma_s x\right) \phi\left(\frac{x - \frac{2P}{\lambda z} - \frac{\beta P}{\lambda z \sigma_s} - \frac{2M}{z} - \frac{\beta L}{\sigma_s} + 1}{2\sigma_y/z}\right) dx}{\int_{-1}^1 \phi\left(\frac{x - \frac{2P}{\lambda z} - \frac{\beta P}{\lambda z \sigma_s} - \frac{2M}{z} - \frac{\beta L}{\sigma_s} + 1}{2\sigma_y/z}\right) dx}. \quad (7.11)$$

According to the definition of the payoff function, $f(X, \theta)$ is increasing in θ . This function is also increasing in κ for a given P , such that the right hand side of the Eq. 7.11 is increasing in κ if only if

$$\int_{-1}^1 \phi\left(\frac{x - \frac{2P}{\lambda z} - \frac{\beta P}{\lambda z \sigma_s} - \frac{2M}{z} - \frac{\beta L}{\sigma_s} + 1}{2\sigma_y/z}\right) dx > 0,$$

which is definitely true according to the definition of a normal distribution. We have shown that in the uncertain region of the investing strategy, for a given P , there exists a unique signal, κ , which makes investors indifferent between investment or not. Furthermore, as the payoff function, $f(X, \theta)$, is increasing in the fundamental value, θ , f is strictly positive when $s > \kappa$ and strictly negative if $s < \kappa$. Hence, an informed investor purchase only when she receives a signal exceeding $g(P)$.

A.2 Proof of Lemma 1

At first, we ignore the first and the third region in Eq. 7.8 and ignore the difference between P and P' in the first stage for simplicity, then take these two conditions into consideration afterwards. Substitute $f(X, \theta) = \alpha X + \theta = (\frac{\alpha z}{2} + \sigma_s)x + \frac{\alpha z}{2} + \kappa$ as the same change of variable in Eq. 7.11, we obtain

$$P' = \frac{\int_{-1}^1 \left(\frac{\alpha z}{2} + \sigma_s\right) x \phi\left(\frac{x - \frac{2P}{\lambda z} - \frac{\beta P}{\lambda z \sigma_s} - \frac{2M}{z} - \frac{\beta L}{\sigma_s} + 1}{2\sigma_y/z}\right) dx}{\int_{-1}^1 \phi\left(\frac{x - \frac{2P}{\lambda z} - \frac{\beta P}{\lambda z \sigma_s} - \frac{2M}{z} - \frac{\beta L}{\sigma_s} + 1}{2\sigma_y/z}\right) dx} + \left(\frac{\alpha z}{2} + \kappa\right).$$

After rearranging the terms, the cutoff strategy function becomes

$$g(P) = \kappa + \beta L = P' - \frac{\alpha z}{2} + \beta L - \frac{\int_{-1}^1 (\frac{\alpha z}{2} + \sigma_s) x \phi(\frac{x - \frac{2P}{\lambda z} - \frac{\beta P'}{\lambda z \sigma_s} - \frac{2M}{z} - \frac{\beta L}{\sigma_s} + 1)}{2\sigma_y/z} dx}{\int_{-1}^1 \phi(\frac{x - \frac{2P}{\lambda z} - \frac{\beta P'}{\lambda z \sigma_s} - \frac{2M}{z} - \frac{\beta L}{\sigma_s} + 1)}{2\sigma_y/z} dx}.$$

With a change of variables $u = \frac{\frac{2P'}{\lambda} + \frac{\beta P'}{\lambda \sigma_s} + \frac{2M}{z} + \frac{\beta L}{\sigma_s} - 1 - x}{2\sigma_y/z}$, and rearrange the above equation to obtain the cutoff strategy

$$g(P) = P' - \frac{\alpha z}{2} + \beta L - \quad (7.12)$$

$$\frac{\int \frac{\frac{P'}{\lambda} + \frac{\beta P'}{2\lambda \sigma_s} + M + \frac{\beta z L}{2\sigma_s}}{\sigma_y} (\frac{\alpha z}{2} + \sigma_s) (\frac{2P'}{\lambda z} + \frac{\beta P'}{\lambda z \sigma_s} + \frac{2M}{z} + \frac{\beta L}{\sigma_s} - 1 - \frac{2u\sigma_y}{z}) \phi(u) du}{\frac{\frac{P'}{\lambda} + \frac{\beta P'}{2\lambda \sigma_s} + M + \frac{\beta z L}{2\sigma_s} - z}{\sigma_y}}$$

$$\frac{\int \frac{\frac{P'}{\lambda} + \frac{\beta P'}{2\lambda \sigma_s} + M + \frac{\beta z L}{2\sigma_s}}{\sigma_y} \phi(u) du}{\frac{\frac{P'}{\lambda} + \frac{\beta P'}{2\lambda \sigma_s} + M + \frac{\beta z L}{2\sigma_s} - z}{\sigma_y}}$$

$$= P' + \sigma_s + \beta L - (\frac{\alpha z}{2} + \sigma_s) \quad (7.13)$$

$$\begin{aligned} & \frac{\int \frac{\frac{P'}{\lambda} + \frac{\beta P'}{2\lambda \sigma_s} + M + \frac{\beta z L}{2\sigma_s}}{\sigma_y} \frac{2u\sigma_y}{z} \phi(u) du}{\frac{\frac{P'}{\lambda} + \frac{\beta P'}{2\lambda \sigma_s} + M + \frac{\beta z L}{2\sigma_s} - z}{\sigma_y}} \\ & (\frac{2P'}{\lambda z} + \frac{\beta P'}{\lambda z \sigma_s} + \frac{2M}{z} + \frac{\beta L}{\sigma_s} - \frac{\int \frac{\frac{P'}{\lambda} + \frac{\beta P'}{2\lambda \sigma_s} + M + \frac{\beta z L}{2\sigma_s}}{\sigma_y} \phi(u) du}{\int \frac{\frac{P'}{\lambda} + \frac{\beta P'}{2\lambda \sigma_s} + M + \frac{\beta z L}{2\sigma_s} - z}{\sigma_y}}) \end{aligned}$$

$$= P' + \sigma_s + \beta L - (\alpha + \frac{2\sigma_s}{z}) \quad (7.14)$$

$$\begin{aligned} & (\frac{P'}{\lambda} + \frac{\beta P'}{2\lambda \sigma_s} + M + \frac{\beta z L}{2\sigma_s} - \sigma_y \frac{\int \frac{\frac{P'}{\lambda} + \frac{\beta P'}{2\lambda \sigma_s} + M + \frac{\beta z L}{2\sigma_s}}{\sigma_y} u \phi(u) du}{\int \frac{\frac{P'}{\lambda} + \frac{\beta P'}{2\lambda \sigma_s} + M + \frac{\beta z L}{2\sigma_s} - z}{\sigma_y} \phi(u) du}) \\ & \int \frac{\frac{P'}{\lambda} + \frac{\beta P'}{2\lambda \sigma_s} + M + \frac{\beta z L}{2\sigma_s} - z}{\sigma_y} \phi(u) du \end{aligned}$$

$$\begin{aligned} & = P' + \sigma_s + \beta L - (\alpha + \frac{2\sigma_s}{z}) (\frac{P'}{\lambda} + \frac{\beta P'}{2\lambda \sigma_s} + M + \frac{\beta z L}{2\sigma_s} - \\ & \sigma_y E[u | \frac{\frac{P'}{\lambda} + M + \frac{\beta P'}{2\lambda \sigma_s} + \frac{\beta z L}{2\sigma_s} - z}{\sigma_y} \leq u \leq \frac{\frac{P'}{\lambda} + M + \frac{\beta P'}{2\lambda \sigma_s} + \frac{\beta z L}{2\sigma_s}}{\sigma_y}]). \end{aligned} \quad (7.15)$$

If the two extreme regions are taken into consideration, with the assumption of rational expectation that market equilibrium equals the asset's dividend payoff and Eq. 7.8 becomes

$$P = \frac{1}{2\sigma_s} \left(\int_{s+\sigma_s-\beta L}^{s+\sigma_s} (\alpha X + \theta) d\theta - \int_{s-\sigma_s-\beta L}^{s-\sigma_s} (\alpha X + \theta) d\theta \right) + P', \quad (7.16)$$

when $f(X(P, \theta), \theta) = \alpha X + \theta$. Then Eq. 7.16 is rearranged as

$$\begin{aligned} P &= \frac{1}{2\sigma_s} (\beta L \alpha X - \beta L \alpha X + \frac{1}{2} \theta^2|_{s+\sigma_s-\beta L}^{s+\sigma_s} - \frac{1}{2} \theta^2|_{s-\sigma_s-\beta L}^{s-\sigma_s}) + P' \\ &= \frac{1}{4\sigma_s} ((s + \sigma_s)^2 - (s + \sigma_s - \beta L)^2 - (s - \sigma_s)^2 + (s - \sigma_s - \beta L)^2) + P' \\ &= \frac{1}{4\sigma_s} ((s + \sigma_s - s - \sigma_s + \beta L)(s + \sigma_s + s + \sigma_s - \beta L) - \\ &\quad (s - \sigma_s - s + \sigma_s + \beta L)(s - \sigma_s + s - \sigma_s - \beta L)) + P' \\ &= \frac{1}{4\sigma_s} (4\sigma_s \beta L) + P' \\ &= \beta L + P' \end{aligned} \quad (7.17)$$

After adjusting Eq. 7.15 by applying Eq. 7.17, a change of variables $u = \frac{\frac{2P}{\lambda z} + \frac{\beta P}{\lambda \sigma_s z} + \frac{2M}{z} + \frac{\beta L}{\sigma_s} - 1 - x}{2\sigma_y/z}$, and the equilibrium condition $L = -\frac{P}{\lambda}$, we obtain

$$\begin{aligned} g(P) &= P \left(\frac{\lambda + \beta}{\lambda} \right) - \frac{\alpha z}{2} - \frac{\beta P}{\lambda} - \\ &\quad \frac{\frac{\frac{P}{\lambda} + \frac{\beta P}{2\lambda \sigma_s} + M + \frac{\beta z L}{2\sigma_s}}{\sigma_y}}{\frac{\frac{P}{\lambda} + \frac{\beta P}{2\lambda \sigma_s} + M + \frac{\beta z L}{2\sigma_s} - z}} \left(\frac{\alpha z}{2} + \sigma_s \right) \left(\frac{2P}{\lambda z} + \frac{\beta P}{\lambda z \sigma_s} + \frac{2M}{z} + \frac{\beta L}{\sigma_s} - 1 - \frac{2u\sigma_y}{z} \right) \phi(u) du \\ &\quad \frac{\int \frac{\frac{\frac{P}{\lambda} + \frac{\beta P}{2\lambda \sigma_s} + M + \frac{\beta z L}{2\sigma_s}}{\sigma_y}}{\frac{\frac{P}{\lambda} + \frac{\beta P}{2\lambda \sigma_s} + M + \frac{\beta z L}{2\sigma_s} - z}} \phi(u) du}{\sigma_y} \\ &= P + \sigma_s - \left(\alpha + \frac{2\sigma_s}{z} \right) \left(\frac{P}{\lambda} + \frac{(1-z)\beta P}{2\lambda \sigma_s} + M - \right. \\ &\quad \left. \sigma_y E \left[u \left| \frac{\frac{P}{\lambda} + M + \frac{(1-z)\beta P}{2\lambda \sigma_s} - z}{\sigma_y} \leq u \leq \frac{\frac{P}{\lambda} + M + \frac{(1-z)\beta P}{2\lambda \sigma_s}}{\sigma_y} \right] \right) \right). \end{aligned} \quad (7.18)$$

Evolutionary Computation and Artificial Financial Markets

Serafin Martinez-Jaramillo¹ and Edward P.K. Tsang^{2,3}

¹ Banco de Mexico, Av. 5 de mayo No. 1, Col. Centro, C. P. 06059, Mexico D. F.
smartin@banxico.org.mx

² Centre for Computational Finance and Economic Agents (CCFEA), University of Essex,
Wivenhoe Park, Colchester CO4 3SQ, Great Britain
edward@essex.ac.uk

³ Department of Computing and Electronic Systems, University of Essex, Wivenhoe Park,
Colchester CO4 3SQ, Great Britain

Summary. It is essential not only for investors but for regulators to understand the mechanisms that govern financial markets. However, financial markets are constantly evolving and are becoming more complex and as a consequence more difficult to analyze and understand. Traditional analytical methods cannot explain some of the phenomena which are present in real markets and some of the assumptions that had to be made for the sake of tractability in such models are over-simplistic. This opens the field to alternative methods that allow us to relax some of the most unrealistic assumptions in order to gain a better understanding of such complex systems. Agent-based computational economics (ACE) offers a suitable alternative for the study of financial markets. In this chapter we develop a software platform called Co-evolutionary, Heterogeneous Artificial Stock Market (CHASM); which allows us to perform a series of experiments with the purpose of identifying the aspects that could be responsible for the statistical properties (stylized facts) of financial prices. In CHASM, we model different types of traders: technical, fundamental and noise traders. However, we focus our research on technical traders represented as genetic programming (GP) based agents which co-evolve in the market forecasting price changes on the basis of technical indicators. We perform a detailed exploration of the market's features in order to identify the conditions under which the stylized facts emerge. Moreover, we develop a behavioral constraint inspired by the Red Queen evolutionary principle to model endogenously the competitive pressure of the market.

8.1 Introduction

Financial markets are essential for financial systems. Such markets represent one of the most efficient ways to allocate financial resources into companies. However, bubbles and crashes are recurrent phenomena and have enormous repercussions for the global economy. In fact, nowadays we can see as never before that a crash in one market could lead to a worldwide slump on most of the remaining stock markets. Moreover, crises in financial markets can directly affect other aspects of the (real) economy; for example, interest rates, inflation, unemployment, etc. These effects in turn could cause even more instability in financial markets.

Financial markets are a very important part of our everyday lives. For example, everybody suffers the consequences of a stock market crash, like the international market crash in 1987. Moreover, this phenomena (market crashes) occur with a higher frequency than is predicted by standard economic theory.

One of the most important research issues in financial markets is the explanation of the process that determines asset prices, and as a result the rate of return. There are many models that can be used to explain this process, including the Capital Asset Pricing Model (CAPM), the Arbitrage Pricing Theory (APT) and the Black-Scholes Option Pricing model.

The complexity of financial markets represents a big challenge to specialists in the area. The traditional way of analysing such markets is via analytical models. However, these models present some difficulties and this has led to the development of alternative methods for the analysis of such markets. The emerging fields of Agent-based Computational Economics (ACE) [101] and Computational Finance [102], provide some means to tackle some of the limitations of analytical models in economics and finance. The ACE approach has been successfully applied in several economic studies, varying from macroeconomic models and payment cards markets [1, 5, 7, 54, 55]. The study of financial markets using ACE has caught the attention of an increasing number of researchers.

Agent-based financial markets of different characteristics have been developed for the study of such markets in the last decade since the influential Santa Fe Artificial Market¹ [11, 63]. Some of them differ from the original Santa Fe market in the type of agents used [23, 46, 78, 99, 110] or in the design of their market mechanisms [12, 46, 99, 110]. Other studies borrow ideas from statistical mechanics [65, 66, 72]. Some important research has been done modeling stock markets inspired on the Minority Game² including [17] and [20]. There are financial simulated markets in which several stocks are traded such as [26]. However, there are some criticisms of this approach due to the problem of calibration of the numerous parameters needed for the simulation program, the complexity of the simulation, etc.

The contradictions between existing theory and empirical properties of the stock market returns are the main driving force for some researchers to develop and use different approaches to study financial markets. An additional aspect on the study of financial markets is the complexity of the analytical models of such markets. Previous to the development of some new simulation techniques, very important simplifying (unrealistic) assumptions had to be made in order to ensure the tractability of developed theoretical models.

Artificial intelligence, and in particular evolutionary computation, has been used in the past to study financial and economic problems. However, the development of a well established community of what is now known as the Agent-based Computational

¹ The Santa Fe Artificial Stock Market is a simulated stock market developed at the Santa Fe Institute by a multi-disciplinary group of researchers including Brian Arthur, John Holland, Blake Lebaron, Richard G. Palmer and Paul Talyer.

² The Minority Game was first proposed by Yi-Cheng Zhang and Damien Challet [19] inspired by El Farol bar problem introduced by Brian Arthur [10].

Economics community facilitates the study of phenomena in financial markets that was not possible in the past.

The influential work of Arthur [11] and previously the development of the concept of bounded rationality [8, 96, 97, 98] and [10], changed the way in which we think of economic agents. No longer do we think of agents as being fully rational or as being homogeneous in their expectations and information symmetry.

Our approach to the modeling of artificial stock markets is different to the above mentioned approaches mainly with regard to the strategic behavior of the agents. We use a very simple market mechanism and sophisticated agents, given that our aim is to study the co-evolution of a group of genetic programming-based agents and the consequences of price changes on the agents' strategic behavior. We are also interested in finding the conditions under which the statistical behavior of the endogenously generated price series resembles the behavior of real market prices. The market reported in this work is composed of different types of traders: technical traders, fundamental traders, and noise traders. Additionally, with the purpose of investigating the role of heterogeneity in artificial financial markets, we have developed a flexible software platform with sophisticated traders and an evolutionary-inspired constraint.

The rest of the chapter is organized as follows: Sect. 8.2 deals with the definition of some basic concepts. Sect. 8.3 provides a review of the state of the art on artificial financial markets. In Sect. 8.4 some of the main criticisms of the agent-based approach in economics and finance are outlined. Sect. 8.5 gives all the details of our simulated stock market and reviews the most relevant features of the model. Sect. 8.6 describes in detail the forecasting mechanism of the technical traders and provides examples of how learning benefits the agents during the trading phase of the simulation. Sect. 8.7 provides the details of the simulation and gives a full account of the different parameters of the model. In Sect. 8.8 we provide the results for the first set of experiments, exploring the different aspects of the model without learning taking place. Sect. 8.9 introduces the Red Queen principle, and presents the results of a series of experiments designed to investigate the impact that this principle has on prices. Sect. 8.10 presents the conclusions and outlines possible avenues of future research.

8.2 Preliminary Concepts

In this section we review briefly some of the basic concepts which are necessary for the understanding of artificial financial markets. Concepts like market efficiency and the statistical properties of stock returns are described later in this section.

8.2.1 Market Efficiency

"I'd be a bum in the street with a tin cup if the markets were efficient." (Warren Buffett)

The concept of market efficiency [35, 36] has ruled the research agenda in Financial Economics in previous decades. It is considered one of the most important concepts in finance and for some years has been at the centre of a debate. There exist different forms of efficiency. However, we are interested just in what is known as *informational efficiency*. Markets are said to be informationally efficient if prices fully reflect available

information [35]. One must be careful to interpret such a definition of efficiency, the last part of the phrase refers to the “available information”. This implies that the definition of market efficiency depends on the information set. Furthermore, we can now derive an alternative definition of market efficiency. A market is efficient with respect to a particular information set ϕ if it is impossible to make abnormal profits by using this set of information to formulate buying and selling decisions. It is helpful to link the type of efficiency with the information set in the following definitions of different market efficiencies

- *Weak Form Efficiency* in this case, the relevant information set comprises all current and past prices.
- *Semi-Strong Form Efficiency* asserts that the asset market is efficient relative to all publicly available information.
- *Strong Form Efficiency* asserts that the market for an asset is efficient relative to all information including private information.

Market efficiency has been associated in the past with the concept of “*random walk*” [76]. However, it is now a well known fact that the price changes do not follow a random walk [69]. Moreover, it is commonly accepted that there are certain variables that possess some predictability power in respect of future price changes [37]. Beyond the debate of market efficiency or the joint hypothesis test³ our work pretends to shed some light on the way in which such efficiency is achieved or at least to explain the origins of the behaviour of financial prices.

8.2.2 Statistical Properties of Stock Prices

The statistical analysis of the price time series is usually performed on the continuously compounded return or log return. The log returns are defined in the following way

$$r_t \equiv \log \frac{P_t}{P_{t-1}} = p_t - p_{t-1} \quad (8.1)$$

where $p_t \equiv \log P_t$. Some of the advantages of such returns are first, that the continuously compounded multi-period return is the sum of continuously compounded single period returns, and second, that it is easier to derive the time-series properties of additive processes than multiplicative processes.

Time series of stock returns exhibit interesting statistical features which seem to be common to a wide range of markets and time-periods. Such statistical properties are known as “*stylized facts*” and have been reported for several types of financial data, and their presence seems to be ubiquitous in all sorts of financial markets [29, 72, 73]. The “stylized facts” have become a very important benchmark for the researchers of artificial financial markets. They are often seen as the first verification criteria when

³ In [18] the authors state: “First, any test of efficiency must assume an equilibrium model that defines normal security returns. If efficiency is rejected, this could be because the market is truly inefficient or because an incorrect equilibrium model has been assumed. This joint hypothesis problem means that market efficiency as such can never be rejected.”

building a simulated financial market [62]. Moreover, some artificial markets try to explain the origins of such stylized facts [71].

We will not report all of the stylized facts in our experiments, mainly because of the frequency of our generated prices (which we will interpret as daily closing prices). Therefore, we will describe briefly the facts that we will be reporting in later sections, as described by Cont in [29]. The stylized facts that we are going to test are the following

1. Lack of autocorrelations: (linear) autocorrelations of returns are usually insignificant. However, this is not true for small intra-day time scales.
2. Volatility clustering: different measures of volatility display a positive autocorrelation over several days, which quantifies the fact that high-volatility events tend to cluster in time. As noted by Mandelbrot in [77], “large changes tend to be followed by large changes, of either sign, and small changes tend to be followed by small changes”.
3. Slow decay of autocorrelation in absolute returns: the autocorrelation function of absolute returns decays slowly as a function of the time lag, roughly as a power law with an exponent $\beta \in [0.2, 0.4]$. This is sometimes interpreted as a sign of long-range dependence.
4. Heavy tails: The distribution of daily and higher frequency returns displays a heavy tail with positive excess kurtosis. The tail index is finite, higher than two and less than five for most assets, exchange rates and indexes.
5. Conditional heavy tails: even after correcting returns for volatility clustering (e.g. via GARCH-type models), the residual time series still exhibit heavy tails. However, the tails are less heavy than in the unconditional distribution of returns.
6. Non Gaussianity: the stock returns on a weekly, daily and higher frequencies fail to be normally distributed.

Fig. 8.1 illustrates the daily closing prices 1(a) and log returns 1(b) for the FTSE 100 index and for Barclays bank’s share 1(c) and 1(d) from the 2nd of January 1998 to the 31st of December 2004.

In order to verify that our endogenously generated price series mimics some of the statistical properties above described, we will perform different sorts of tests. For the first property, we will report the autocorrelations of the log returns, the absolute log returns and the squared log returns for different time lags. The autocorrelation of the absolute and squared log returns will allow us to investigate the phenomenon known as volatility clustering [77]. Empirical studies in various stock indices and stock prices have shown that the autocorrelation function of the squared returns remains positive and decays slowly over several days. The autocorrelation function can be defined as

$$C(\tau) = \text{corr}(r_t, r_{t+\tau}) \quad (8.2)$$

where τ is the time lag. If the first property holds, it should be observed that the log returns’ autocorrelations for different lags should be around zero. In Fig. 8.2 we can observe that the log returns’ autocorrelation is effectively around zero for the FTSE 100 index and Barclays bank’s share. However, we can see in the same figure, that such lack of autocorrelations does not happen for the absolute or squared log returns, which is a quantitative signature of the phenomenon known as volatility clustering (property number two).

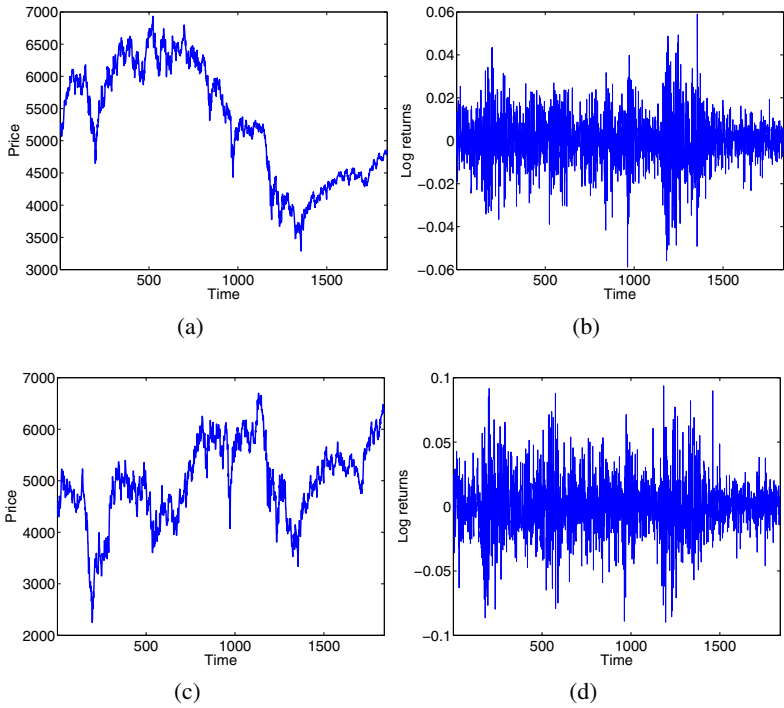


Fig. 8.1. Price and log returns for the FTSE 100 (a) and (b); and Barclays bank's share (c) and (d)

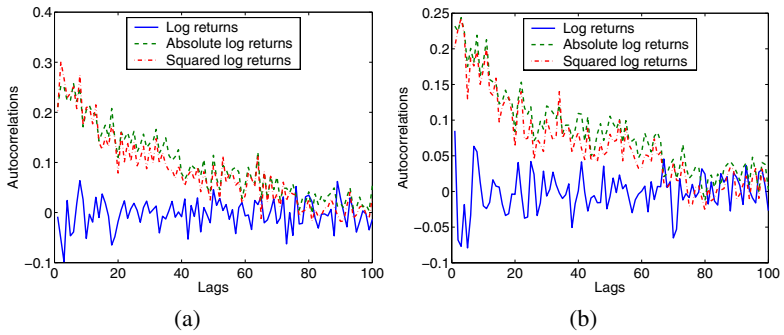


Fig. 8.2. Autocorrelations for different lags of the log returns, absolute log returns and squared log returns on the FTSE 100 (a) and on Barclays bank's share (b)

Property number three can be also verified in Fig. 8.2. We can see there that the autocorrelation of the absolute and squared log returns decays until it is practically zero for lags larger than eighty days.

The distribution of financial time series displays “fat tails”. The term “fat tails” refers to higher density on the tails of a distribution in comparison to the tails’ density under

Table 8.1. Statistics for the log returns FTSE100 and Barclays

Statistics	FTSE100	Barclays
Mean	-0.0000340558	0.00020352
Median	0	0
Minimum	-0.0588534	-0.0898057
Maximum	0.0590256	0.0937403
Std. Dev.	0.0124598	0.0233416
GARCH coefficient	0.899999	0.899536
ARCH coefficient	0.0895644	0.0899882
Skewness	-0.133266	0.113409
Kurtosis	5.13829	4.62582
J-B Test H value	1	1
Corr. coefficient	-0.00831591	0.085004
AlphaHill 1 %	5.05533	6.7913
AlphaHill 2.5 %	3.96377	4.89301
AlphaHill 5 %	3.24536	3.35794
AlphaHill 10 %	2.61432	2.26481
AlphaHill 15 %	1.97705	1.94232

the normal distribution. In order to be able to determine the shape of the tail one must estimate the shape parameter (α) or the tail index (τ). The Hill tail index [48] is an estimator of the α parameter and it could be considered as the standard tool for the study of tail behaviour of economic data due to its good performance and simplicity. However, one of the main problems on the application of this index is that it is necessary to define a priori the size of the tail. To overcome such limitation, the fourth property is going to be tested by calculating and reporting the Hill tail index for different tail sizes (0.1%, 0.5%, 1%, 2.5%, 5%, 10% and 15%). Additionally, we will report the returns' kurtosis. For a normal distribution the kurtosis is three. However, it has been found that in financial data kurtosis is typically larger than three. This phenomenon is known as excess kurtosis and is an indication of fat tails.

The fifth property is going to be tested by reporting the ARCH and GARCH coefficients. Both coefficients should be less than one. Property number six is examined using the Jacque-Bera test, which indicates whether the sampled data is drawn from a Normal distribution or not. Table 8.1, shows some basic statistics (including, GARCH and ARCH coefficients, skewness, kurtosis, the Jacque-Bera H value, the correlation coefficient with lag one and different Hill tail indexes for various tail sizes) for the FTSE 100 and Barclays. Sample kurtosis is a valuable indicator of departure from normality (it has a value of three for the Normal distribution). Typical values for sample kurtosis in exchange rates, indexes and high frequency data are much larger than three.

We can observe in Table 8.1 values for kurtosis of 5.13829 and 4.62582 respectively. The values for the Hill tail index vary from 6.7913 to 1.94232. Another interesting value is that of the Jacque-Bera test, which in the two reported cases rejects the null hypothesis that the sampled data is drawn from a normal distribution.

8.3 Survey of Artificial Financial Markets Research

The area of Artificial Financial Markets has witnessed a sustained increase in the number of papers published related to this field. We can see all different sorts of artificial markets made by researchers from very dissimilar disciplines like Economics, Finance, Computer Science, Physics, Psychology, etc. Although they all differ in the assumptions made, methodology and tools; these markets share the same essence: the macro behaviour of the market (usually the price) emerges endogenously as a result of the micro-interactions of the (heterogeneous) market participants. This approach is in contrast with the traditional techniques being used in Economics and Finance. Moreover, in [74] Lux and Ausloos declare

“Unfortunately, standard modeling practices in economics have rather tried to avoid heterogeneity and interaction of agents as far as possible. Instead, one often restricted attention to the thorough theoretical analysis of the decisions of one (or a few) *representative* agents”

In [53], Kirman criticizes the *representative individual* approach in economics. Despite the existence of different works around the same time (early 1990s), we can assert that the most influential work is the Santa Fe Artificial Stock Market [11]. This market was developed by a number highly reputed researchers, among them John Holland, the inventor of genetic algorithms [49].

In order to understand the different approaches of the variety of artificial (simulated) financial markets we will describe the different types of markets on the basis of the framework proposed in [58]. The main design issues identified in [58] are

- Agents
- Market Mechanism
- Assets
- Learning
- Calibration
- Time

In addition to the description of the different approaches in artificial financial markets by using the above described framework, we will describe some other related works that were not commented in LeBaron’s survey. Although, we consider such framework sufficient for our needs, there is a fairly detailed extension of it in [47] that is worth looking at. In this work the basic design issues proposed in [58] are extended and detailed. The main goal that we pursue in this section of the chapter is to describe some of the classic works on Artificial Financial Markets within the frameworks of [58] and [47].

8.3.1 Agents’ Trading Strategies: From “Zero Intelligence” to Artificial Intelligence

The decision of which type of agents will be used in the artificial financial market is by far the most important one and the different options that exist range from basic zero intelligence agents [26, 46, 99] to genetic programming based agents [23, 24, 32, 81]. There are several design issues regarding the agents as described in [47]: the decision

making process, the objective function that drives their decision making, heterogeneity and learning. Learning is discussed in more detail below.

The Decision-making Process

In relation to the decision making process, there are several routes to follow when we face the decision of the design of the agents that will participate in the market. The most obvious approach is to model the agents to reflect the strategies that are used in real life. However, these rule based agents are strongly criticized by traditional economists in the sense that such agents might lack of a well defined objective (utility) function. Another criticism is related to the dynamics of interactions. In real life people can change their mind, and as a consequence, change their investment strategies.

Examples of static ruled based agents are [46, 99]. Gode and Sunder propose what they called “*zero intelligence*” agents. Such agents’ behaviour is ruled by a simple budget constraint. Nevertheless, such agents generate efficient trading behaviour when they are placed in a realistic market mechanism. This research aims to demonstrate the importance of the market mechanism. These agents fit into the same category as the agents previously mentioned regarding the learning mechanism, this means that these agents do not learn. There are agents that are able to adapt to the new market conditions by changing their investment rules such as [6, 11, 23, 78, 81]. This class of agents use a variety of artificial intelligence techniques to model the constant change of the strategies. Some related problems to this approach are

1. the complexity of the resulting model that incorporates sophisticated agents,
2. the limitations on the search space caused by the assumptions and design of the agents, and
3. the complexity of the evolved strategies.

In [47] in addition to the rule-based agents, there is a further classification of agents to include forecasting agents. Within such classification, the agents can be divided into econometric based forecasters and forecasters based on cognitive systems. Examples of the former can be found in [59, 110]. Examples of the later include [6, 11, 23, 32, 78, 81, 111]. The main difference between the two approaches is that forecasting agents do not consider the semantic specifications of the cognitive process. This raises an important aspect of the agents’ modelling as it was stated in [34]: it is important to represent the cognitive process in a way that helps to understand the things that we model, we should be careful in the selection of the technique that we use to represent the agents’ behaviour and avoid just taking an algorithm designed with another original purpose.

A different approach to the design of the agents, is the one followed in works like [2, 3, 12, 30, 54, 71, 72]. In these models the agents make their decisions taking into account the decisions made by the other agents. The purpose of such works is to model herding behaviour among the agents. Such phenomena is thought to be present in financial markets and it is believed that such behaviour could partially be the responsible for the appearance of the heavy tails in the distribution of stock price changes.

The Objective Function

In relation to the objective function, there are two main ways to design this element of the agents. The objective function could be modeled implicitly or explicitly on the agents decision making process. In the case of an implicit objective function, the decision making incorporates indirectly the objective of the agents. For example, in the case where the agents are equipped with real life trading strategies, the goal of such strategies is profit maximization; however, it is not explicitly incorporated into the agent's trading strategy. In [39] we can find an example of this type of objective function.

The case of an explicitly modeled objective function is present in the majority of agents that participate in simulated markets. Within this type of objective function, utility maximization and profit maximization objective functions can be found. The first type of objective function can be found in works that base the agents' decision on the concept of Constant Absolute Risk Aversion (CARA). These utility functions can be found in [11, 23, 60, 84, 110]. Modifications or enhancements of the CARA concept can be found in works like [15, 25]. The second form of objective function can be found in works like [78, 81, 112].

Other Relevant Aspects on the Design of Agents

There are several sources of heterogeneity on the design of agents for artificial financial markets. There could be heterogeneity on the type of agents used for the simulation (fundamental, chartists, noise traders, etc.). The information and the parameter settings could be also a source of heterogeneity. Finally, there could also be heterogeneity on the learning capabilities and on type of adaptation and learning used by the agents (ANN, GA, GP, LCS, etc.). The discussion regarding the adaptation and learning mechanism is a whole topic in itself and is going to be discussed later in this chapter. Summarizing, the design of the agents is the single most important aspect on the modeling of agent-based markets. Some of the most important design issues include

- decision making (rule based, econometric forecasters and cognitive based agents)
- objective function (explicit, implicit, utility maximization and profit maximization)
- heterogeneity (types of agents, information basis, parameter settings and learning)
- learning (from zero intelligence to genetic programming)

Grothmann [47] provides a detailed description of each of the above mentioned design issues.

8.3.2 Market Mechanism

This is the second most important design decision that the researcher must make in order to build up an artificial financial market. Again, there are a wide range of possibilities on this front. There are three main ways to solve this design problem: the easiest is to create a simple price response to the excess demand with a simple clearing mechanism, an alternative is to create a simple market where a local equilibrium can easily be found and last, a continuous double auction-like mechanism can be explicitly implemented. One of the main advantages of this area of research is that different market mechanisms can be compared in order to contrast them in specific issues that the researcher might be interested to study.

In the first type of market mechanism we can find an earlier version of the Santa Fe Artificial Stock Market [84]. Additionally, there are markets using a similar market mechanism, some examples of such models are [23, 30, 39, 78, 81]. In addition to the above described mechanism, there are some works that incorporate a market maker to deal with the excess demand [38].

The second type of mechanism defines a market structure that allows the discovery of a temporal equilibrium price. This method requires the definition of a market with more economic structure than in the previous case. Among some examples of markets with such type of mechanism are [11, 66].

The third way of modeling the market mechanism is to implement a fairly realistic market. For example, a continuous double auction mechanism with limit orders and some other realistic features. Some examples of such models are [22, 46, 110]. More recently, some new research has been done in modeling the decision making process made by a market maker by means of reinforcement learning [21].

An alternative to all of the above mentioned market mechanisms is to model a stylized mechanism that does not resemble any real trading mechanism. However, such mechanism should bring some advantages, like in the case of the Minority Game (MG), as the basis for the modeling of a simulated financial market. For such game, there exists an analytical solution to the problem and can be used to gain understanding of its rich collective behaviour. There is a fairly important group of researchers working on the MG as the basic framework to model financial markets, like [17, 20, 51, 79].

8.3.3 Assets

The assets that are going to be traded on the market are an important aspect of every artificial financial market. Modeling this aspect of the market can be separated into three different decisions concerning the: number of assets, types of assets and asset properties.

Regarding the first case, the vast majority of the research done in this area so far has involved the trading of two different assets: a risk free asset and a risky asset. This has been done for the sake of tractability, otherwise the complexity involved in the simulation and the analysis of a multiple assets market could be impossible. Nevertheless, some recent works have carried out such challenging task and have produced markets in which multiple assets are traded [26, 108, 112].

The aspect related to the types of assets traded in the market refers to the different possibilities of financial instruments that are going to be considered available to the agents. Again, in the majority of the available markets, the agents can choose from a risk free asset (mostly cash or bonds) or a risky asset (a stock, a currency or a security). Some more interesting extensions along this line could include markets of financial derivatives.

In relation to the third aspect of the traded assets, we can observe that in the majority of the markets, the agents can choose between a risk free asset or a risky one. The properties of each of the assets may vary in the following way: the risk free asset could be cash or a bond that pays an interest rate in each period of the simulation; on the other hand, the risky asset could be linked to a fundamental value that could be modeled by an exogenous fundamental process like in [39, 81], a stream of dividends like in [23, 84, 85] or even

a constant value like in [72]. Another way of modeling the fundamental process can be found in [68], where each fundamental trader perceives an individual fundamental value. Finally, let us consider the case of the approach in [22], where the authors propose a market in which there is a stream of dividends for each individual agent.

8.3.4 Learning

Learning is a crucial element of the design of an Artificial Financial Market. Several design issues arise when deciding the way in which the agents will update their trading strategies. One of these issues is the rule generation mechanism, the rule transformation over time (if any) and the fitness evaluation criteria for these rules. In [14], Brenner provides an overview of the learning models that have been used by the economists in recent decades. Additionally, in this paper there is an extensive discussion on how to design the learning process for economic models and he provides some very useful advice.

On the design of the learning mechanism it is possible to follow the line originally developed in [46] in which they make use of zero-intelligence agents with a budget constraint in a double auction like market. Despite the low level of intelligence, they are able to get a remarkable allocation efficiency that could be comparable with the efficiency obtained in experiments with humans. In this study Gode and Sunder state

“Adam Smith’s invisible hand may be more powerful than some may have thought; it can generate aggregate rationality not only from individual rationality but also from individual irrationality.”

In [40] Farmer, Patelli and Zovko by using zero intelligence agents, arrive to almost the same conclusion of Gode and Sunder: ... *it appears that the price formation mechanism strongly constrains the statistical properties of the market, playing a more important role than the strategic behavior of agents.*

Our position regarding such conclusions is that, in our opinion, “intelligence” and adaptation are very important mechanisms in the modeling of the agents’ behaviour. The experimental results we obtain in this study, point in a different direction to the above mentioned research. We believe that, as it has been clearly explained in [27], the results in [46] are a consequence of the experimental setting and more sophistication is needed to test the behaviour of humans in different conditions and markets.

The following discussed agents’ adaptation mechanism is considered to be similar to reinforcement learning and it is a very interesting mechanism that has been intensively used by some researchers in modeling of financial markets. We are referring to the so called “Minority Game” a simplification of the problem introduced by Brian Arthur in [9]. Such problem was inspired by the bar El Farol in Santa Fe and the basic idea in such problem is to attend to the bar when there is less than 60% of the possible attendees. There are some works that are based on the Minority Game to develop simulated financial markets, including [17, 20, 51, 79, 80]. Despite the limitations of the MG as the basic element of an artificial market, its analytical tractability makes it very attractive for the study of financial markets as complex adaptive systems of many interacting units.

Another possibility in designing the agents’ learning mechanism is to model the agents behaviour by using Artificial Intelligence techniques, like genetic algorithms [7];

or as it has been proposed in [50] by Holland and Miller, by using learning classifier systems [11, 63, 91, 92]; or by using artificial neural networks [110, 111]; or genetic programming like in [23, 24, 32, 33, 81]; or using reinforcement learning [21].

From the soup of different Artificial Intelligence techniques to select from, some questions arise like

- Which technique to use?
- Which one is the best?
- Which one is most realistic?
- Which technique is most efficient in computational terms

The answer to all these questions obviously depends on the type of agent being modelled. Another important issue is the computational effort that is needed to implement each of the above listed techniques. Depending on the computational resources available, one specific technique might be better than another because of its computational efficiency. In addition to the reasons outlined in [32], in our opinion GP is a suitable computational technique to model agent behaviour for the following reasons

- It is a technique that has proven to be successful in financial forecasting.
- It is a very flexible technique.
- Its expressional power can allow the researcher to build up very sophisticated behaviours.
- There are several techniques that allow complexity control on the evolved individuals.
- GP shares with other similar techniques the possibility of making it easy to embed domain knowledge and information processing mechanisms.

Despite all the advantages on using artificial intelligence techniques in the design of the agents trading strategy; is important to never forget that decisions should be made considering the modeling of a realistic behaviour and cognition process [34].

8.3.5 Calibration and Validation

Validation is an important issue in Agent-Based Computational Economics and in particular in Artificial Financial Markets [45, 58, 60]. Validation is a basic demand to the agent-based markets as it is pointed out in [60]. Simulated markets should be able to replicate realistic quantitative features of the real market with a reasonable calibration. Another important aspect that we would expect from simulated markets is that the modeling of bounded rationality should be modeled in a fairly simple and transparent way. Finally, the trading mechanism should be a truthful representation of a real trading situation.

Due to the way in which simulated markets are designed, multiple parameters need to be user-defined and that gives the researcher several degrees of freedom. Despite this issue, the researchers in this area can perform several things in order to overcome this problem [58]. Among such things, one is to create a useful benchmark in which the behaviour of the market is well defined. Another approach is to use parameters in the simulated market derived from experimental or real markets.

The selection of the market parameters is by no means a simple task. However, it is very important to explore the changes on the behaviour of the market due to changes on such parameters. This has been recognized in [58] by LeBaron

“... understanding exactly where the parameter boundaries are between simple and complex behaviors is crucial to understanding the mechanisms that drive agent based markets.”

We can find some interesting works in which an analysis of such parameters has been made, like [15, 25, 44, 72]. The approach of incorporating parameters borrowed from either, real or experimental markets, can be found in works like [13, 60, 81, 111, 112].

8.3.6 Time

The timing in the context of artificial financial markets refers to the order in which the relevant events take place (synchronization), the length of the past history considered by the agents and the frequency in which such agents update their behavioural rules.

The vast majority of agent-based financial markets have synchronous models of trading. The most important assumption in these models is that trading happens between two discrete points in time. Examples of such trading synchronization are: [11, 39, 66, 78, 81, 111]. This sort of synchronization mechanism is one of the main criticisms to this methodology as clearly such assumption is unrealistic. More effort is needed to implement realistic asynchronous agent-based models. There are some models that contemplate (even if in a limited way) asynchronous trading. Examples of a more realistic simulation synchronicity can be found in [88].

In relation to the memory span of the agents in [59, 63], LeBaron studies the changes on the statistical properties of the price due to changes on the memory of the agents.

Regarding the frequency of the updating of the agents' behavioural rules, there are essentially three options: updating with a fixed periodicity, like in [11, 59], updating with a notion of rank, like in [23] or updating in an endogenous way, as it is proposed in [75, 78, 81].

8.4 Limitations

It is always a challenge to convince a person that uses or believes in the traditional methods in economics and finance of the validity of the Agent-Based approach to study financial markets. Surely, many of us have faced such difficulties when talking to some of our fellow researchers. Particularly, the mention of the lack of realism of some of the most important assumptions in economics (like homogeneous expectations or full rationality) generates always a debate with passionate positions on each side [35, 36, 84, 93, 94].

In this section we will mention some of the most important and frequent criticisms that the Agent-based approach to the study of financial markets faces. We think that some of such criticisms are justifiable and the people that believe in the validity of this field should work in trying to tackle them in order to gain further acceptance from the critics of this approach.

One of the main criticisms to the ACE approach and by extension to the Artificial Financial Markets field is the calibration of the models and the necessary tuning of the parameters' constellation [45, 58, 60, 61, 109]. Some of the critics of this approach argue that a lot of work is needed to choose the right parameters for the simulation to make sense. Moreover, how could we justify the values taken by some of such parameters?

Another important criticism is that some of such artificial markets lack of a rational, optimizer, utility maximizer representative economic agent. The more traditional economists are very reluctant to accept an approach in which there is not a rational expectations type of agent, where instead there are inductive boundedly rational heterogeneous agents [9, 96, 97]. Nevertheless, we are convinced that people have *bounded rationality*. To justify our opinion we will cite Herbert Simon: "*boundedly rational agents experience limits in formulating and solving complex problems and in processing (receiving, storing, retrieving, transmitting) information*"

More objections to the Agent-Based models come from the complexity of such simulations. There exist many artificial financial markets that achieve realistic prices that reproduce the stylized facts present in financial time series. However, due to the complexity of the simulations, it is not clear which aspect is responsible of the generation of such statistical properties.

The assumption of the synchronicity of some of the events in the artificial financial markets and more generally in the agent-based models is one of the most unrealistic assumptions as it was seen in the previous section. Although, some progress has been made more work is needed in order to implement more realistic models on the synchronization of the trading events.

8.5 CHASM

The Co-evolutionary Heterogeneous Artificial Stock Market (CHASM) is a software platform that allows a user to experiment with different market scenarios. In CHASM, a user may create markets with different combinations of traders, including fundamental, technical, noise or hybrid traders. In this section we provide an overview of the model and its main characteristics. For a more detailed description of the model please refer to [81].

8.5.1 Overview of the Model

The market is populated by traders that interact with each other by buying and selling some assets. Any market participant i will be able to hold at any time t two different types of assets

- a risky asset, denoted by $h_i(t)$ or
- cash, denoted by $c_i(t)$.

The market is composed of technical, fundamental and noise traders. We define N_T as the number of technical traders, N_F as the number of fundamental traders, N_N as the number of noise traders and N as the total number of traders in the market. The stock price at time t will be denoted by $P(t)$.

At the beginning, all the agents are endowed with a certain number of shares and a certain quantity of cash, both specified by the investigator. Their position on each of the assets might change as a result of the agents' decision to sell or buy a certain quantity of the risky asset.

The market mechanism is described in detail in [81], we will not give the full details of the model in order to avoid making this chapter too long.

Defining all the different sorts of traders that intervene in a financial market is a complex task. Nevertheless, there are some well accepted classes of traders that are commonly used in the literature. In this work we will limit such classes to three basic types: technical traders, fundamental traders, and noise traders. None of them follows rational expectations and their means of interaction will be through the price. In the artificial financial markets literature we will find mostly these three types of traders, although the specific mechanisms and implementations can vary widely.

8.5.2 Important Features in CHASM

In this section we will describe briefly some of the most important features of CHASM.⁴ The implications of changes in such important aspects of our model are explored through experimentation and the results are reported below.

Market and Limit Orders

When a person, a professional trader, a market maker or a corporation is trading on a stock market, there are different ways of doing so. After the decision-making process of any of such entities, an order must be submitted to a broker (or the representant that is trading on behalf of them). There are two main types of orders⁵

1. Market orders
2. Limit orders.

Fundamental trading

In addition to the incorporation into the market of fundamental traders, CHASM allows us to incorporate fundamental-like behavior on top of the technical traders. This characteristic of the technical traders in our model can be justified by arguing that in real life some traders do use technical analysis in conjunction with fundamental analysis. These traders know that the price of a certain stock is well beyond a reasonable value (fundamental value); however, they still follow the trend a little longer (short time horizon) in order to make a profit out of it. In [100] the authors report that more than 90 percent of dealers in the foreign exchange market use some form of technical analysis, and in short time horizons technical analysis predominates over fundamental analysis.

⁴ More detail is provided in [81].

⁵ There is a number of other types of orders but it is beyond the scope of this paper to give a full account of them.

Indicators

The indicators used by the technical traders to forecast increases or decreases in the price are a very important aspect of our market. Such indicators can make a substantial difference to the behavior of the endogenously generated price.

The indicators used for the current work consist of technical, momentum and volatility indicators. The indicators that were used and their periods are: The price moving average delta of the last 12 and 50 days, the trading breakout rule of the last 5 and 50 days, the filter rule of the last 5 and 63 days, the price volatility of the last 12 and 50 days, the momentum of the last 10 and 60 days and the momentum-moving average of the last 10 and 60 days. We used both short and long horizon indicators because that is the way in which they are used by practitioners of technical analysis. We chose such indicators mainly because they proved to be useful in forecasting rises and drops of the price in previous work such as [16, 67, 103, 104, 105, 106], and [42]. Nothing prevents us from using other information such as more sophisticated technical indicators, information from the limit order book, market microstructure information, fundamental information, etc. We performed a form of normalization on the data in order to limit their ranges, which in turn limit the size of the search space.

Desired Return and Time Horizon

The technical traders will be organized in groups that will share some common characteristics, as previously described. Among such characteristics we have the desired rate of return and the corresponding time horizon to achieve such a rate of return. These two characteristics (parameters) prove to be of central importance in the behavior of the market, since our GP agents work as classifiers.

The majority of the simulated markets possess agents that do not consider multi-period preferences, and, furthermore, the agents share the same planning, forecasting and decision-making horizon [58]. CHASM is different to most of the previously designed models in the forecasting mechanism and the heterogeneity of time horizons.

The GP forecasting mechanism of our agents works by classifying the training cases into three different classes: buy, sell or hold. Such classification depends on the time horizon provided, as the mechanism will verify for each data point if, in the near future (time horizon), effectively there was a rise (drop) in the price by more (less) than the desired rate of return.

If the selection of such quantities is unreasonable, it will cause the classifier to be biased towards a certain class, Thus creating unrealistic (unreasonable) investment rules and having an unrealistic price and statistical properties of the returns as a result.

Trading Proportion

The trading proportion is a parameter of the market that controls the proportion of the asset or cash that the traders would commit on each of the operations that they will perform during the trading rounds. The trading proportion is a quantity that we can use to model the degree of cautiousness of the agents in our market.

The trading proportion proved to be of central importance in our simulations as the prices exhibited different behavior for different values of g . The implications of this

important feature of our market will be tested experimentally and described in the next section.

8.6 Learning to Forecast Investment Opportunities

In this section we explain the main forecasting mechanism, which is the core of the decision-making process of the technical traders. Genetic programming is at the heart of this mechanism and it has been used in the past to perform technical analysis by several research groups, for example [31, 42, 82]. The modeling of the learning process by the agents is a central part of our research agenda. With regard to the agents' learning process, we consider of extreme importance what Lucas wrote in [70]

In general terms, we view or model an individual as a collection of decision rules (rules that dictate the action to be taken in given situations) and a set of preferences used to evaluate the outcomes arising from particular situation-action combinations. These decision rules are continuously under review and revision; new decision rules are tried and tested against experience, and rules that produce desirable outcomes supplant those that do not.

For the modeling of the learning process described above we will use genetic programming. This technique has been previously described as a suitable way to model economic learning in [14]. The learning process that we used to model our agents' behavior will be described further in this section.

8.6.1 Forecasting with EDDIE

We use the architecture for EDDIE, explained in [103] and [67], for the elaboration of the agents' decision rules which recommend whether to buy, hold or sell. As is standard with genetic programming, index genetic programming each agent is assigned an initial population of randomly generated decision rules. These include well known fundamentals-based forecasting rules or trend-following, moving average-based technical rules. Candidate individuals are selected randomly, biased by their fitness, to generate members of the next generation. General mechanisms (referred to as *genetic operators*, e.g. selection, crossover, mutation) are used to combine or change the selected candidate individuals to generate offspring, which will form the population in the next generation.

In EDDIE, an individual is represented by a decision tree. The basic elements of such decision trees are *rules* and *forecast values*. A single rule consists of one useful indicator for prediction, one relational operator such as "greater than", or "less than", etc, and a threshold (real value). Such a single rule interacts with other rules in one decision tree through logic operators such as "Or", "And", "Not", and "If-Then-Else". Forecast values in this model are directions of price movements – either a positive trend (suggesting that positive x percentage return within a specified time interval can be achievable) or a negative trend (suggesting that negative x percentage return within a specified time interval can be achievable).

Fig. 8.3 shows an example of one possible decision tree. In this figure we can see that the root node is always an If-Then-Else node, the left child an If-Then-Else node

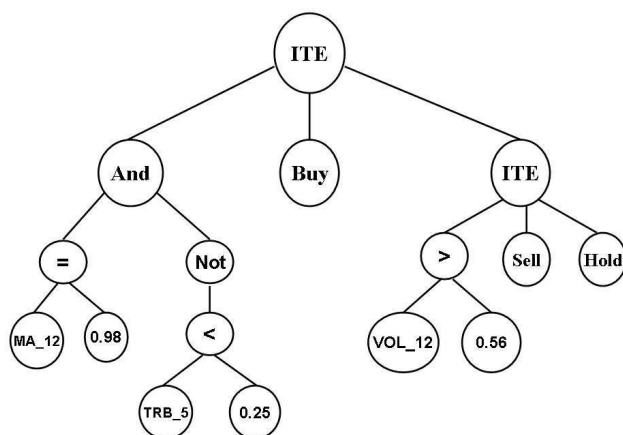


Fig. 8.3. Example of a decision tree

```

1  If ((MA_12 = 0.98)AND(NOT(TRB_5 < 0.25))) Then
2    Buy
3  Else
4    If(VOL_12 > 0.56) Then
5      Sell
6    Else
7      Hold
8    End if
9  End if

```

Fig. 8.4. Example of a decision rule interpreted from a decision tree

is a “condition” node. Additionally, there are two right children which could be either a “decision” node or another If-Then-Else node.

One of the conditions of the previous expression that specifies that the moving average of the past twelve days should be equal to 0.98 in fact is verified in a range around such value. It would be extremely unlikely that a randomly generated number could be matched exactly by the evolution process and for that reason a range is used instead.

The type of node is going to become relevant when applying the crossover and mutation genetic operators because it is precise to maintain the tree consistency. For example, in the case that a mutation operation is going to take place and the selected node where such mutation is going to happen is a “condition” node; then, the randomly generated mutation must be a “condition” like node. The same should happen for the crossover operation: there must be compatibility between the subtrees that are going to be exchanged by the parents.

Recommendation to BUY at t follows from the prediction of a price rise (positive trend) over a given period, recommendation to do nothing (HOLD) follows from the

Table 8.2. A contingency table for three-class classification/prediction problem

	Predicted price rise (PBs) BUY	Predicted no inf. (PHs) HOLD	Predicted price drop (PSs) SELL
Actual price rise (ABs) BUY	# of True Buys (TB)	# of Actual Buy Predicted Hold (ABPH)	# Actual Buy Predicted Sell (ABPS)
Actual no inf. (AHs) HOLD	# of Actual Hold Predicted Buy (AHPB)	# of True Holds (TH)	# of Actual Hold Predicted Sell (AHPS)
Actual price drop (ASs) SELL	# of Actual Sell Predicted Buy (ASPB)	# of Actual Sell Predicted Hold (ASPH)	# of True Sells (TS)

fact that there is no evidence of a price rise or a price drop and recommendation to SELL follows from the prediction of a price fall ($x\%$ negative trend). Note different returns thresholds and horizons exist for different classes of traders. Since decision trees are used to predict directions of price changes and make recommendations for trade, the success or failure of recommendations can be categorized as a three-class classification problem. Each prediction point for every decision tree can be classified into either a positive position, a holding position or a negative position. For each decision tree, we define RC (Rate of Correctness), and RF (Rate of Failure) as its prediction performance criteria. Formula for each criterion is given through a contingency table in Table 8.2. Let's define: RC as the Rate of Correctness; and RF as the Rate of Failure.

$$RC = \frac{TB + TH + TS}{ABs + AHs + ASs} \quad (8.3)$$

$$RF = \frac{AHPB + ASPB}{PBs} + \frac{ABPH + ASPH}{PHs} + \frac{ABPS + AHPS}{PSs} \quad (8.4)$$

Each agent selects the decision tree which constitute trading strategy to buy or sell that maximizes the fitness function

$$f_{(1)} = \varphi(rc)RC - \varphi(rf)RF \quad (8.5)$$

The fitness function involves two performance values, i.e. RC and RF, each of which is assigned a different weight $\varphi(rc)$ or $\varphi(rf)$ respectively. While the fitness function can guard against loss making positions, the population of decision trees from which agents conduct their search may lead to investment income under-performance. One important advantage of genetic programming is that we can bias the search mechanism by using different values for such weights. However, we used a value of zero for the

weight $\varphi(rf)$. In other words, we used just the rate of correctness as the performance criteria to drive the evolutionary mechanism.

We implemented a very efficient method to avoid bloat and to speed up our simulation (see [87]). Bloat happens during the evolutionary process when the trees grow in size but there is no improvement in fitness (see [57]). Essentially, this means that there are some branches of the trees that are redundant or, even worse, they reduce the fitness of the individual. We considered the control of bloat to be important because it was necessary for us to generate realistic investment rules. It would have been very difficult for us, or indeed anyone, to justify very complex rules being used by the agents.

8.6.2 Learning

Learning is a key factor in our simulation and as we were investigating the repercussions of the type of learning frequency on the endogenously generated price, we had to verify that the learning process was beneficial. To be more precise, learning should enable individuals to improve their wealth in relation to other traders. Technical traders are the only type of traders that are able to learn during the simulation of the market.

To investigate this issue, we performed several experiments as follows: in each experiment we replicated one trader (technical) and followed both traders' wealth (the original and the replicated one) throughout the whole simulation. We set the replicated trader to be able to retrain every 1000 steps of the simulation. We can see clearly in Fig. 8.5 that after the execution of the GP mechanism with the endogenously generated price, the replicated trader improved its wealth in comparison with the original trader. We executed these experiments several times with different traders from different groups and in

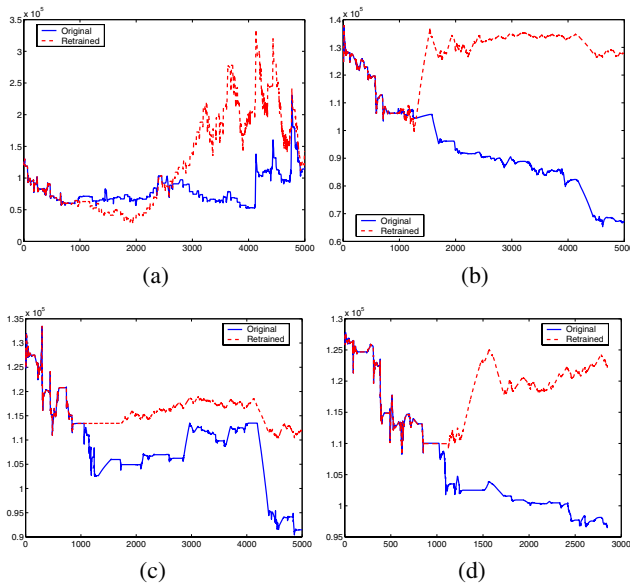


Fig. 8.5. Examples of wealth evolution with and without learning

the vast majority of cases the replicated traders did better than the original ones. We allowed the replicated trader to retrain every 1000 periods of trading because we wanted to give the original trader the opportunity to improve during that period and even to perform better than the retrained agent, as we can see in Fig. 8.5(a). Additionally, we wanted to avoid changing too many things simultaneously, which would make analysis impossible.

During the retraining phase, the GP mechanism is executed with the same conditions as per the initial training. This is done with the objective of preserving what we perceive as a realistic and competent forecasting mechanism. It is worth mentioning that the rate of correctness of the agents during the initial trading is above sixty percent.

It is important to note that, in our case, the fitness measure used to drive the evolution process was the rate of correctness. Other fitness criteria, such as the profitability of the agents' trading strategies, could have been used. The rate of correctness does not equate to profitability. However, it has a direct impact on the agent's wealth.

8.7 The Simulation

The market will operate as if each trading round is one day. This is due to the fact that our technical traders were trained with daily closing prices. However, there is nothing to prevent us from interpreting the time in another scale or from training the agents with high frequency data. The market participants will be able to trade on every single round of the market, with some exceptions to be explained next.

Noise traders will take a decision to buy, sell or do nothing based on the probabilities assigned for each decision. They will be able to participate in the market on every single iteration of the simulation program.

The fundamentalists enter on a buying (selling) cycle if there is a difference between the stock's price and the fundamental value beyond certain threshold value T . They will stay in this cycle until the difference is smaller than another threshold value τ . After the return of the stock's price to the fundamental value, this type of traders will review again whether there exists a difference between the market price and fundamental value.

The technical traders can change their position on each trading round based on their forecasts, unless they have some pending limit orders to execute. Once they have completed the round trip (the investment decision and the exit strategy), they can take another decision based on the generated rules.

Once all the market participants make their bids and offers, we calculate the excess demand and then the price can be updated. Afterwards, each agent's orders are partially satisfied by a proportion that clears the market considering the new price. Finally, the holdings of the risky asset and cash are updated for each of the traders that participated in the trading round.

After all the above steps are executed, each technical trader reviews its retraining condition. In the model there exist two types of conditions for retraining: in fixed time intervals and in an endogenous way, known as *Red Queen* retraining.

In the case that the retraining periodicity is set as fixed time intervals, the trader launches the GP mechanism creating the initial population of rules with half of their current population and the other half randomly generated.

The case of the Red Queen retraining is more complex: retraining for a certain agent will take place as soon as the agent's wealth falls below the average wealth. This agent will not be retrained until it gets enough evidence that its investment rule fails to achieve above-average wealth. The agent's initial population for each retraining process is generated in the same way as was described for retraining with fixed periodicity.

8.7.1 Parameters

We have created a flexible model in which has a large number of parameters for us to explore enabling us to analyze different phenomena in financial markets. These parameters are going to be divided mainly into two different classes: market parameters and traders' parameters. For more details regarding the parameters of the simulation please refer to [81].

8.8 Experiments

This section describes the experiments performed in CHASM with the aim of generating realistic price series. We seek to discover the minimal conditions under which "stylized facts" emerge. It is worth emphasizing that finding such conditions in a complex system is non-trivial. To our knowledge, this is the first successful attempt in identifying minimal sets of conditions in an endogenous model under which realistic price dynamics emerge. The complexity of some artificial markets usually prevents researchers from knowing which aspects of their model are responsible for the emergence of stylized facts.

With regard to the exploration of the parameters and main features, we performed a full exploration of each of the parameters involved on the simulation. For example, in the case of the trading proportion parameter we scaled it from one percent to one hundred percent. In the case of the limit orders, their exploration was easier as there are only eight possible variations on this feature. In the case of the fundamental behavior, we turned this on and off for every single group of the technical traders. We proceeded in the same fashion for all the mentioned parameters and features. Nevertheless, we will only present the results of the most relevant cases.

8.8.1 Parameters and Feature Exploration

In the previous section we described the most important features and parameters of CHASM. Due to the many possible combinations of such features and parameters, a systematic approach is necessary to search for conditions under which the price dynamics exhibit the desirable properties. We will first describe the characteristics of what we call the Base Case, and then describe the experimental results of changes on the other features and parameters of the model.

8.8.2 The Base Case

A large number of controlled experiments were conducted before obtaining a price series that resembled the dynamics present in real prices. In this section, we present the

results of a parameter setting which reproduces statistical properties of stock returns. This setting will be used as a base case for studying the effects of changing the model along the individual dimensions listed in the previous section. The Base Case has the following parameters and characteristics

- Seven different groups of technical traders.
- The groups have different indicators.
- The groups share the same desired rate of return (5.5%) and time horizon (14 days).
- The agents trade 8% of their current holdings or use 8% of their cash to buy more shares.
- The agents generate both types of limit orders.
- The groups have the same computing power.
- There is no learning taking place.
- Group number seven of technical traders behave like value traders under certain circumstances.

Besides the set of indicators used by each group of traders and the fundamental behavior exhibited just by one group, the remaining conditions are the same for all the different groups. The indicators were assigned in the following way: group one was assigned the two moving average delta indicators, group two was able to use the trading breakout indicators, group three had the filter indicators, group four used the volatility, group five used the momentum indicator, group six used a moving average of the momentum indicator and finally group seven used all of the indicators. For the experiments reported in this paper, the number of agents in each group is typically three.

In Fig. 8.6 we can see the price and the log returns of the Base Case. We can see that the price resembles the dynamics of the prices in real markets and the log returns capture the well known phenomenon of volatility clustering. This phenomenon can be investigated quantitatively and in Fig. 8.7 it is possible to see the autocorrelation for different lags of the log returns, the absolute log returns and the squared log returns.

We can observe in Fig. 8.7 that the autocorrelation of log returns is around zero, as it should be. Additionally, we can see that for the absolute and squared log returns there is a positive autocorrelation that decays slowly but remains positive even for lags larger than eighty. However, such positive autocorrelation is never close to zero as we saw in the case of the FTSE 100, Fig. 8.2.

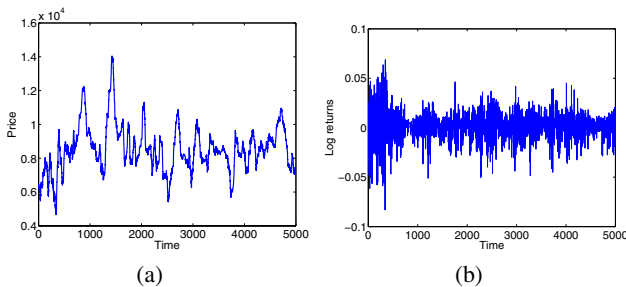


Fig. 8.6. Price and log returns for the Base Case

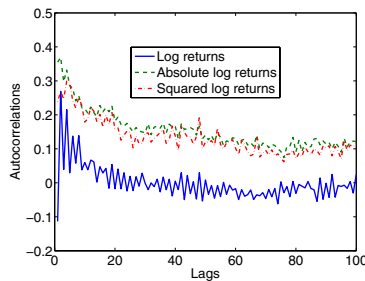


Fig. 8.7. Autocorrelations for different lags of the log returns, absolute log returns and squared log returns on the Base Case

8.8.3 Limit Orders

Our agents were equipped with a powerful forecasting mechanism (EDDIE). However, even if EDDIE is reliable, all it does is to help the agent that uses it to pick up investment opportunities. Profit is only realized when the agent sells the assets. Therefore, we decided to equip the agents in CHASM with a more complete investment strategy, which includes an exit strategy. This exit strategy will be modeled by two different types of limit orders: *profit-taking* limit orders and *stop-loss* limit orders. In order to test the relevance and the impact of the inclusion of limit orders on the agents' trading strategies, we designed some experiments in which we turned on and off the use of either type of limit order. We departed from the base case by not generating both limit orders but everything else remained the same.

8.8.4 Fundamental Trading

Fundamental behavior is modeled in most Artificial Financial Markets. It is an important mechanism, in our experience, in order to prevent the price from behaving in a simplistic fashion. Before including such characteristic in our market, we had prices that were either always increasing or decreasing until they collapsed. We modeled the fundamental behavior as it was designed in [39], on top of the technical trading behavior. This means that the agents will behave like a technical traders until the price departs from what they consider to be the fundamental value of the risky asset by a given threshold.

In order to test the impact of triggering the fundamental behavior, we performed some experiments in which we activated or deactivated such behavior in just one of the groups. Departing from the Base Case, we turned on and off the fundamental-like behavior in all the groups except for group seven. We did not impose such a mechanism on all the groups for two reasons: first, we wanted to have certain heterogeneity and second, we wanted to prevent the endogenously generated price following too closely the exogenous fundamental value.

8.8.5 Indicators

The indicator set used by each of the different groups of agents is one of the most relevant factors that we identified in order to reproduce stylized facts. We used the indicator set to model information asymmetries on the different groups of agents.

We took two different possible approaches: firstly, as described for the Base Case, we equipped each group of traders with a subset of indicators, except one group which is equipped with all the indicators. Secondly, every group is given all the available indicators for building their decision rules. The former is taken as the Base Case.

8.8.6 Desired Return and Time Horizon

The desired rate of return and the time horizon are two very important aspects of our model. In fact, these two parameters will rule the creation of investment rules during the evolutionary process.

In order to test the impact of such parameters on the behavior of the price, we performed several experiments in which we either had homogeneity among the different groups (like in the Base Case) or we had heterogeneity in those two parameters. We have to stress the importance of a careful selection of both parameters. For example, if, for a particular group, we programmed them to ask for a large desired return on their investment, then it would be unreasonable to assign them a small time horizon. Such unreasonable selection of both parameters could lead to unreasonable behavior of the agents, like agents that buy all the time or agents that do nothing most of the time. In other words, let's assume that we want to get a 20% return. Clearly this is very difficult to observe in real markets. However, if we want to achieve this in five days it is almost impossible to observe. The result of such selection would generate training data in which the class that would have the majority is the "Do Nothing" class.⁶

8.8.7 Trading Proportion

The trading proportion parameter could be used to model how cautious or aggressive the agents are in trading. If we want to model rather conservative agents, the trading proportion parameter would be close to 1%. On the other hand, if we want to model aggressive trading, the parameter would be close to 100%.

In the base case the trading proportion is 8%. From there we varied the trading proportion parameter in order to test its impact on the market dynamics. When this parameter is reduced below 8%, no difference was observed in the price dynamics. Consequently, our analysis focusses on increases in this value. However, we must stress that changes in this parameter are closely related to the parameter market depth⁷ on the price determination equation.

⁶ Remember that the agents are trying to classify each price point to belong to the classes: "Buy", "Sell" and "Do Nothing".

⁷ Market depth is related to the size of an order that is necessary to change the price in the market.

8.8.8 Discussion

In this section we describe the conditions under which some of the statistical properties of the endogenously generated price resembles those of the real financial markets. In CHASM we can change several parameters and features of the model. A systematic experimental investigation of changes in such parameters and characteristics was necessary. In this section we report the results of such experiments from which we can extract the following main conclusions

- The generation of limit orders proved to be an important factor to obtain realistic price dynamics. Such limit orders were incorporated as part of a more complete investment strategy of the agents.
- The inclusion of the fundamental behaviour in one of the groups of agents was an important factor to obtain a realistic price. We must stress that even when just one of the groups has activated this behaviour, the obtained price improves dramatically in terms of the desirable properties that we were looking for.
- The indicators and the heterogeneity on their assignment to the different groups of agents is an important factor in our aim to obtain stylized facts. The assignment of such indicators allowed us to model asymmetries on the information used by the agents.
- Heterogeneity in the computational capability assigned to the groups of agents proved to be of no utility in our aim to get realistic price dynamics. We attribute this to the fact that the sort of investment rules generated by stupid agents are far from realistic and as consequence, the behaviour of such agents is unrealistic as well.
- The desired rate of return and time horizon parameters, proved a key factor first to generate realistic investment rules within the agent's minds and second the heterogeneity on the assignment of such parameters favored the behaviour of the endogenously generated price.
- The trading proportion parameter can be considered as a parameter that possesses an important impact on the resulting price. Nevertheless, we must stress that changes in such parameter should be accompanied by reasonable changes on the market depth parameter on the price determination equation.

Additionally, with the objective of making a clear distinction of which scenarios and the reported experiments are the ones that bear most similarity to real prices, we will present a resume in a tabular form. The criteria that we used for our experiments to determine if a price is or not (in our opinion) realistic are the following ones

- C1) GARCH and ARCH coefficients. Both coefficients and their addition must be less than one.
- C2) Kurtosis. The kurtosis of the price generated on the reported experiment should be larger than three, which is the kurtosis of a normal distribution.
- C3) Jacque-Bera Test H value. It is also necessary that the hypothesis that the generated price is drawn from a normal distribution is rejected.
- C4) Alpha Hill estimator. It is required that the Hill estimator is between five and two for different tail sizes.

- C5) Autocorrelation I. The autocorrelation of the logarithmic returns should be around zero, even for different lags.
- C6) Autocorrelation II. A positive autocorrelation for lags close to zero and autocorrelation decay for larger lags should be observed for the absolute and squared logarithmic returns.
- C7) Atypical behaviour. The prices generated under the evaluated scenario must not show cyclical or another forms of atypical behaviour like huge changes or monotonicity.

In Table 8.3 we report whether the experiment passes or fails each individual criterion. In the table, on each row we report the analysis for a price generated under one of the scenarios described on this section and each column represents one of the criteria. We can see that the two most successful experiments are: the one in which we provided the groups with different time horizons and expected returns, and the one in which we used a trading proportion of 30%. Interestingly, criterion number six seems to be the most difficult to satisfy and none of our experiments was able to reproduce this behaviour.

From the controlled exploration of the relevant aspects of our simulation model, we have found promising areas of the market parameters where the endogenously generated price reproduces to a certain extent the statistical properties of real financial markets. We have explored aspects such as the generation of limit orders, the inclusion of fundamental behavior on the traders' investment strategies, homogeneity or heterogeneity on the information set, etc. Nevertheless, there is still another dimension to explore in our artificial market.

The experiments described in this section do not contemplate adaptation (learning) to the new conditions of the market by the agents. The decision on the type of learning mechanism used for the agents is by no means a simple one. Furthermore, once this decision has been taken, there is another very important decision to make: How frequently

Table 8.3. Evaluation criteria on stock prices for each group of experiments

	C1	C2	C3	C4	C5	C6	C7
Base Case	✓	✓	✓	✓	✓	✗	✗
Limit Orders	✓	✓	✓	✓	✗	✗	✗
No Fundamental	✓	✓	✓	✓	✓	✗	✗
Homo. Information	✓	✓	✓	✗	✓	✗	✗
Heter. Computation	✓	✓	✓	✓	✓	✗	✗
Heter. Time and Return	✓	✓	✓	✓	✓	✗	✓
Trading Proportion 10%	✓	✓	✓	✗	-	-	-
Trading Proportion 20%	✓	✓	✓	✓	-	-	✗
Trading Proportion 30%	✓	✓	✓	✓	✓	✗	✓
Trading Proportion 40%	✓	✓	✓	✓	-	-	✗
Trading Proportion 50%	✓	✓	✓	✗	-	-	✗
Trading Proportion 60%	✓	✓	✓	✗	-	-	✗
Trading Proportion 70%	✓	✓	✓	✓	-	-	✗
Trading Proportion 80%	✓	✓	✓	✗	-	-	✗
Trading Proportion 90%	✓	✓	✓	✓	✓	✗	✗
Trading Proportion 100%	✓	✓	✓	✗	✓	✗	✗

should the agents review their belief systems? Lucas [70] suggested that human traders revise their beliefs when such beliefs do not match reality. Performance evaluation on financial markets crucially involves the other traders' performance [58]. However, it is very unlikely that the endogenous revision mechanism is as complicated as the one proposed in [23]. Inspired mainly by the above mentioned works, we propose a simple and plausible endogenous mechanism for the agents to update their belief systems. In the next section we will report the results of four groups of controlled experiments under three different scenarios of the adaptation (learning) process

1. No learning taking place for all the agents
2. Learning with fixed periodicity
3. Learning under the Red Queen constraint

8.9 Co-evolution and the Red Queen principle in CHASM

Co-evolution is said to take place when two or more lineages have an impact on each other's selection mechanism and cause changes to each other that increase fitness. This mutual adaptation happens whenever certain ecological interaction has fitness effects for all the participants. Co-evolution in Evolutionary Computation, and in particular in GP, is an important area of study in Computer Science and has been used for modeling independent agents in the past [56]. Moreover, the notion of competitively evolving populations, and in particular decision trees, has been intensively studied in [4, 52] and [95].

We aim to model traders in real markets. Therefore, we consider co-evolution to be of central importance in our work because in real life traders certainly have an influence on each other's trading strategies. Additionally, it is common for them to change their strategy if they are not performing well in relation to other market participants. Therefore, in our opinion it is necessary to include in our market model the co-evolution of the agents' trading strategies. In our market the co-evolution of such strategies will be modeled by using two different mechanisms

1. The endogenous generation of the risky asset's price.
2. The regeneration of the technical trader's "strategies". In CHASM, this can be done in a fixed or endogenous way.

In our market, the population of investment rules of each trader co-evolves through the price. The effectiveness of each rule (its forecasting precision) determines the probability of it being selected to be part of the population of the next generation during the evolutionary process. The precision of a certain rule belonging to a particular trader depends on the rules of the other traders because they will have to change their rules if their performance on the market is not good. Therefore, it affects the trader's fitness.

With regard to the learning process, in CHASM the user can model the agents so that they are able to adapt to the new conditions of the market, either periodically (with a user defined periodicity) or with a periodicity that is endogenously generated by a behavioral constraint known as the Red Queen constraint. CHASM allows those two different conditions to trigger learning because we want to contrast both ways, although we believe that learning in an endogenous way is more realistic.

Adaptation of the market participants with fixed periodicity has been used in previous works, eg. [11]. However, we consider it unrealistic since in real life the traders do not change their investment strategies (at least not the totality) in a strictly fixed way. Moreover, the findings reported in [63] are very revealing about the importance of the learning periodicity. In [63], the authors reported the differences in the price properties due to changes in the frequency of the retraining procedure and the memory length of the agents.

The condition to trigger the learning process endogenously has been modeled in previous works. For example, the notion of self realization was modeled in [23] which is based on a notion of rank. However, we believe that the traders' decision to change their strategy must be motivated by their performance in the market in terms of wealth. It is very likely that this decision is strongly related to the other market participants' performance. This was recognized as an important aspect in artificial financial markets by LeBaron in [58].

To summarize, we believe that the modeling of co-evolution in simulated markets is important. Therefore, we make it a key feature in CHASM. It is possible to model the way in which learning is triggered in two different ways: with fixed periodicity or endogenously. In CHASM we are able to model such forms of adaptation and we will report the results of some experiments later in this section.

8.9.1 The Red Queen in CHASM

In our research we have a market populated by a co-evolving population of agents, each attempting to enhance their fitness relative to others. This is inspired by the Red Queen principle, based on the observation made to Alice by the Red Queen in Lewis Carroll's *Through the Looking Glass*: "in this place it takes all the running you can do to keep in the same place". The Red Queen principle was originally proposed by the evolutionary biologist Leigh van Valen in [107] as a metaphor for a co-evolutionary arms race between species. In cases where the competition for scarce resources rules the behavior of the participants, the important performance measure is relative to the other individuals involved in the arms race.

The Red Queen principle has been studied in Computer Science, more specifically in competitive co-evolution [83, 86]. In particular, in [28] the authors propose some means to measure the progress in computer simulated co-evolution. The Red Queen effect has also been studied in the context of Economics. In [89] the author claims that the evolution of intelligence itself is hypothesized to arise as a Red Queen-type arms race giving rise to Machiavellian behavior in social interactions. In [90] the author describes the relation between the evolution of complex organisms, the reasons behind sexual reproduction, the emergence of high intelligence and the Red Queen effect. In competitive co-evolution, the Red Queen principle entails constraints on performance enhancement of all individuals if each is to maintain *status quo* in relative fitness.

In CHASM, the Red Queen principle will be modeled through the Red Queen constraint. Such a behavioral constraint will force traders to search for new investment rules whenever they are being "left behind". More specifically, a trader will launch

a GP mechanism considering the most recent information (the most recent price history and the relevant indicators) whenever its wealth is below the population's average wealth. As recognized in [58] by LeBaron: *"A trader's performance depends critically on the behavior of others"*.

8.9.2 Experimental Design

To experimentally test the impact of the Red Queen principle, we designed a set of experiments in which the agents retrained in a fixed or endogenous way. Then we observed the differences in the statistical properties of the stock returns and the agents' wealth distribution. In these experiments we defined some study cases which we considered to be interesting. This was taking into account the experience gained from the experiments described in the previous section. These study cases were different variations of the factors that we identified as important in the previous phase of experimentation.

We first executed the simulation with the parameters that we consider to be appropriate to get realistic statistical behavior of the log returns. Afterwards, we observed the price behavior and we performed a series of statistical tests to identify if the price presented stylized facts. In case the stylized facts were not replicated, we tried another parameter constellation and so on until we got the desired properties. After obtaining the appropriate parameters, we executed the market with the same setting but allowing the agents to learn with fixed periodicity. Next, we reported the statistics of this execution. For the experiments reported in this chapter, learning with fixed periodicity was triggered every one thousand trading periods.

Using the same configuration of the two previous experiments, we allowed the agents to retrain in an endogenous way, i.e. we enabled the Red Queen constraint. Then we observed the price and the statistical properties of the logarithmic returns. For the experiments, the Red Queen constraint was implemented by retraining the agents whenever their wealth was below the total population's average wealth.

8.9.3 Case Studies and Results

To explore the model described in the previous section, we developed a series of experiments by changing key features and parameters that were identified to be important. We had different base cases in which the statistical properties of the price were reported for different parameters. After tuning the parameters under each case of study to obtain interesting prices and statistics, the simulation was repeated with learning taking place. This allowed us to observe the changes on the statistical properties of price due to the different types of learning.

In the previous phase of experimentation, the following three features were found to be important for producing stylized facts: (i) Computing Power, (ii) Information, and (iii) Time Horizon and desired Rate of Return. With each of these three features being either homogeneous or heterogeneous, there are eight possible combinations, one of which is our base case

- Homogeneous Computing Power, Heterogeneous Information and Homogeneous Time Horizon and Return.

We studied three out of the remaining seven combinations. The excluded cases were as follows

- Heterogeneous Computing Power, Heterogeneous Information and Heterogeneous Time Horizon and Return.
- Heterogeneous Computing Power, Heterogeneous Information and Homogeneous Time Horizon and Return.
- Heterogeneous Computing Power, Homogeneous Information and Homogeneous Time Horizon and Return.
- Homogeneous Computing Power, Homogeneous Information and Homogeneous Time Horizon and Return.

The reason for the exclusion of the first three cases is that our previous results showed that heterogeneity on the computational capabilities proved to be counterproductive in our quest for reproducing stylized facts. In all the experiments that we performed with heterogeneous computational capabilities, the results were similar in the sense that the price did not resemble by any means prices in real financial markets. The reason for the exclusion of the last case is that we detected that complete homogeneity was not good. The results of the last case (not reported here) showed a monotonous price (always increasing or always decreasing), i.e., there was a sort of consensus and a self-fulfilling behavior of the price. We believe that this was mainly caused by the homogeneity of agents in the important features that we detected.

For each of the experiments' log returns we will report: basic descriptive statistics, the result of the Jarque-Bera Test, the GARCH and ARCH coefficients, the skewness and kurtosis, the correlation coefficient, and the Hill estimator for the 0.1%, 0.5%, 1%, 2.5%, 5%, 10% and 15% most extreme log returns of the experiment's respective data series. The basic statistics reported here are the mean, median, minimum, maximum and the standard deviation.

Case 1: Heterogeneous Computing Power, Homogeneous Information and Heterogeneous Time Horizon and Return

In this study case we have seven groups of technical traders with heterogeneous computing capabilities, homogeneous information, heterogeneous desired return and time horizon.

The purpose of testing and analyzing this specific case was to verify the impact that the asymmetry in computational power (in terms of the GP mechanism) has on the market. In other words, heterogeneity in computing power refers to two of the GP parameters assigned to the agents: population size and number of generations. We identified these two parameters as the most relevant ones for measuring the computational capabilities of the agents. In this experiment, the first group of agents was provided with the smallest number of generations and population size (10 and 50 respectively), the second group had a bigger number of generations and population size than the first group (50 and 100 respectively), the third had bigger numbers (75 and 250 respectively), the fourth was the best endowed of all the groups (100 generations and 500 population size) and then we started to reduce the population size and number of generations for the remaining three groups.

The obtained prices were the least successful in replicating the stylized facts in comparison to the other cases. We expected such results to a certain extent because it is very important that the behavior of the agents is closer to reality and thus we need competent traders to participate in our market. Otherwise, the price and its statistical properties do not resemble real data. We found very unrealistic behavior of the price even with learning taking place; in fact the price behavior is worst when learning happens.

Case 2: Homogeneous Computing Power, Heterogeneous Information and Homogeneous Time Horizon and Return

In this case we have seven groups of technical traders with homogeneous computing capabilities, heterogeneous information, homogeneous desired return and time horizon. This configuration is the same as for the base case reported in subsection 8.8.2.

The purpose of setting up this case was to study the role that the information, modeled here by the different indicators, plays with regard to the price formation and wealth distribution. For this purpose, the computational capabilities of the agents will be homogeneous and the same will happen with the desired return and time horizon. Therefore, the only difference between the trader groups is the information set they will use to create the investment rules. For example, the agents in the first group will use just the moving average delta indicators to create decision rules, the second group of agents will use the trading breaking rules to do the same, the third group will use filter rules, the fourth group will use volatility, the fifth will use a momentum indicator, the sixth a moving average indicator based on the momentum and the last group will use all the indicators.

Case 3: Homogeneous Computing Power, Homogeneous Information and Heterogeneous Time Horizon and Return

In this case we have seven groups of technical traders with homogeneous computing capabilities, homogeneous information and heterogeneous desired return and time horizon.

The purpose of this experiment was to verify the importance of the heterogeneity in the agents' time horizon and desired rate of return in reproducing the statistical properties in real financial markets. In this experiment, for the first group of agents, we chose a small time horizon and a desired rate of return that made sense for such a time horizon (otherwise we would have experienced the same problems that we had at the beginning of our research). For the second group we chose a slightly longer time horizon and adjusted the rate of return to an achievable level within such a time horizon. We increased the time horizon and chose the respective desired rate of return for the remaining five groups. With this organization, we had the first groups with small time horizon and rate of return and the last groups with the parameters taking bigger values.

Case 4: Homogeneous Computing Power, Heterogeneous Information and Heterogeneous Time Horizon and Return

In this case we have seven groups of technical traders with homogeneous computing capabilities, heterogeneous information and heterogeneous desired return and time horizon.

The purpose of this experiment is to explore the impact on the price of the heterogeneity in information, desired return and time horizon. This case would allow us to clarify the importance of the heterogeneity in our market. We expected the price dynamics in this case to be the best of all of our basic cases. Benefitting from the experience of the previous cases, we decided not to make the traders different in terms of the GP mechanism (computational capability). The sources of heterogeneity were: the indicators, the desired rate of return and the time horizon. We assigned the indicators to the different groups in the same way that was described in Case 2. In the case of the time horizon and desired rate of return we proceeded in the same way that was described in the study Case 3.

Discussion

In this section we observed the implications that learning has on the statistical properties of the endogenously generated price and the repercussions of learning on the wealth distribution of the agents and on the different groups of traders.

We described the way in which we implemented what we called the Red Queen Constraint, inspired on previous works describing the relevance of the Red Queen Principle in Economics. Before we experimentally tested our interpretation of such principle, we performed a series of experiments in which the agents retrained with fixed periodicity determined in an exogenous way. But which is the right periodicity? Moreover, how do we justify such decision? We argue that the agents' decision to retrain should be taken endogenously and for that reason we propose the Red Queen Constraint as the way in which the traders should update their beliefs. As a result, we observed that the price dynamics and the statistical properties of the log returns were closer to the dynamics and properties of the real financial time series.

We observed that the wealth distribution of different groups of traders was different in different executions of CHASM: Generally speaking, the traders that take decisions with higher frequency end up better off than those that do not. Interestingly, asking the agents to review their beliefs in an endogenous way, resulted in a very unequal distribution of wealth in the final trading round of the market. This result is a very interesting one, since we are just asking to the population of traders to do the best they can to preserve their status quo in the market. Nevertheless, it seems that such arms race causes the majority of the traders to end up with a small amount of wealth, while very few agents end up with most of the population's wealth. Additionally, we observed that heterogeneity does help in general terms to obtain realistic statistical properties. However, having heterogeneity in computing capability between the agents proved to be harmful to our intentions of reproducing realistic market dynamics (Case 1). This point is important for us due to the fact that we believe that simulations of Artificial Financial Markets should be done with agents capable enough to imitate human behavior. In our experience, having agents with not good enough investment decision rules, creates a very different type of market dynamics.

Information proved to be an interesting source of heterogeneity in our market (Case 2). The impact of the time horizon and desired rate of return was beneficial (Case 3). The combination of heterogeneity in both information and investment preferences did not lead to the best results in terms of reproducing the stylized facts (Case 4).

Table 8.4. Evaluation criteria on stock prices for each group of experiments

	C1	C2	C3	C4	C5	C6	C7
Case 1 with no learning	✓	✓	✓	✗	✓	✗	✗
Case 1 with fixed learning	✓	✓	✓	✗	✗	✗	✗
Case 1 with Red Queen constraint	✓	✓	✓	✓	✗	✗	✗
Case 2 with no learning	✓	✓	✓	✓	✓	✗	✗
Case 2 with fixed learning	✓	✓	✓	✓	✓	✗	✗
Case 2 with Red Queen constraint	✓	✓	✓	✓	✓	✗	✗
Case 3 with no learning	✓	✓	✓	✓	✓	✗	✓
Case 3 with fixed learning	✓	✓	✓	✓	✓	✗	✓
Case 3 with Red Queen constraint	✓	✓	✓	✓	✓	✓	✗
Case 4 with no learning	✓	✓	✓	✓	✓	✗	✓
Case 4 with fixed learning	✓	✓	✓	✓	✓	✗	✗
Case 4 with Red Queen constraint	✓	✓	✓	✓	✓	✗	✗

In real life, the traders are likely to have different time horizons (due to budget constraints or simple preferences), return considerations and they tend to look in different ways the same information (or even they might use different sources of information). Nevertheless, it is unlikely that there is a big difference in computing capability among the main players (small investors and the rest of us are just noise).

We should emphasize at this point that despite the fact that our market is capable of modelling fundamental behavior, in this four study cases we are not making use of it. This means that we are able to mimic to a certain extent the statistical properties of log returns without an explicit-exogenous fundamental mechanism. We consider this to be very important for the study of Artificial Financial Markets because so far we have always seen a sort of fundamental mechanism in any of the important works in this field. The exception to this rule are the works of [17, 41] and [43]. In [17] the authors arrive at a conclusion that is very much related to ours:

“This suggests that the statistics we observe in real markets is mainly due to the interaction among speculators trading on technical grounds, regardless of economic fundamentals”

Finally, we assume in the same way as it was done on Section 8.8, the criteria that we used to differentiate the results of the experiments reported on this section. We use the same criteria also defined on the previous section. In Table 8.4, we can observe which of the individual criteria are satisfied by each study case. We can see that the cases that satisfy the most of the criteria are: the three experiments of Case 3 and the experiment in which learning does not take place of the Case 4. Case 3 is the most successful of all the study cases and it is the only case in which the price satisfies C6. In Case 4 we can observe that in the experiment with the Red Queen Constraint the non-linear autocorrelations decay to zero. However, in the case of the absolute log returns, such autocorrelation becomes negative for lags close to 100.

8.10 Conclusions

Agent-based economic models in general, and artificial financial markets in particular, seem to have achieved acceptance as a method to understand and analyze economic systems and financial markets respectively. However, the future success of the field depends on how closely such models can reproduce what happens in real life and on the simplicity of them. From our experience and from the evidence gathered in this work we can formulate the following conclusions.

It is very important to pay attention to the way in which the agents' initial population of rules are generated. We could have highly unbalanced training and testing data which in turn could cause biased behavior of the agents.

The learning mechanism should be tested in order to prove that it is doing properly the task which was designed for: improving the agent's wealth on the market. Our results show that the learning mechanism that we designed improves the agent's wealth even in a changing environment. Additionally, in the environment that we defined, the rate of correctness of the GP mechanism is a suitable fitness function which derives into an improvement of the agent's wealth.

Heterogeneity is an important feature of our market in order to obtain realistic prices. We tested different sources of heterogeneity like information, computational capability, desired return and time horizon, etc. Our results show that heterogeneity has an important impact on the statistical properties of the endogenously generated price.

The periodicity in which the agents' update their beliefs has enormous repercussions on the price. The decision on how frequently the agents should update their beliefs is not an easy one and for that purpose we propose an endogenous behavioral constraint inspired on the Red Queen evolutionary principle.

8.10.1 Future Work

One way in which this work can be extended is to compare different (probably more complex) market mechanisms. Despite the success that we had on reproducing some of the statistical properties of financial prices, there are some other statistical properties which we did not test like the gain/loss asymmetry.

Market microstructure is another possible area of future work in which we could answer some of the most relevant questions in finance. Additionally, a richer set of market participants could be incorporated like market makers.

It is not clear yet which is the best way to model economic learning and very important questions are waiting to be answered. Despite the fact that GP seems to be an appropriate tool to model economic learning, there is still a passionate debate on rationality and on the way in which economic agents are being modelled by the agent-based community. However, we are convinced that GP is a suitable flexible and transparent tool to be used in such work.

Financial markets are probably the most competitive, dynamic and complex of all the markets. Automatic trading has increased dramatically in recent years and we still need to know the implications of such intensive trading on the prices and more importantly on the likelihood of financial crashes. We are convinced that there is a lot of relevant

research waiting to be done which will be impossible to perform only by analytical means or as it was put by Markowitz on the introduction of the book [64] by Levy et al:

“Levy, Solomon and Levy’s Microscopic Simulation of Financial Markets points us towards the future of financial economics. If we restrict ourselves to models which can be solved analytically, we will be modeling for our mutual entertainment, not to maximize explanatory or predictive power.”

Evolutionary computation is at the center of such efforts to overcome the limitations of analytical methods as it is a powerful, expressive, flexible and transparent paradigm in the artificial intelligence area.

Acknowledgements

The author acknowledges financial support from Banco de Mexico and CONACYT. The authors are grateful to the anonymous referees for their helpful comments. Special credit is given to Dietmar Maringer and Bruce Edmonds for their valuable input to this work.

References

- [1] Alexandrova-Kabadjova, B., Tsang, E., Krause, A.: Market structure and information in payment card markets. Working Paper 06, Centre Computational Finance and Economic Agents (2006)
- [2] Alfarano, S., Wagner, F., Lux, T.: A minimal noise trader model with realistic time series properties. Economics Working Paper 2003–15, University of Kiel, Germany (2003)
- [3] Alfarano, S., Wagner, F., Lux, T.: Estimation of agent-based models: the case of an asymmetric herding model. Tech. rep., University of Kiel, Germany (2004)
- [4] Angeline, P.: Genetic Programming and Emergent Intelligence. In: Kinnear Jr., K.E. (ed.) *Advances in Genetic Programming*, ch. 4, pp. 75–98. MIT Press, Cambridge (1994)
- [5] Arifovic, J.: Genetic algorithm learning and the cobweb model. *Journal of Economic Dynamics & Control* 18, 3–28 (1994)
- [6] Arifovic, J.: The behavior of the exchange rate in the genetic algorithm and experimental economics. *Journal of Political Economy* 104, 510–541 (1996)
- [7] Arifovic, J.: Evolutionary Dynamics of Currency Substitution. *Journal of Economic Dynamics & Control* 25, 395–417 (2001)
- [8] Arthur, W.B.: Designing Economic Agents that Act Like Human Agents: A Behavioral Approach to Bounded Rationality. *American Economic Review* 81, 353–359 (1991)
- [9] Arthur, W.B.: On learning and adaptation in the economy. Working paper 92-07-038, Santa Fe Institute (1992)
- [10] Arthur, W.B.: Inductive reasoning and bounded rationality: The El Farol problem. *American Economic Review* 84, 406–411 (1994), citeseer.ist.psu.edu/arthur94inductive.html
- [11] Arthur, W.B., Holland, J.H., LeBaron, B., Palmer, R.G., Talyer, P.: Asset pricing under endogenous expectations in an artificial stock market. In: Arthur, W.B., Durlauf, S., Lane, D. (eds.) *The Economy as an Evolving Complex System II*. Addison-Wesley, Reading (1997)

- [12] Bak, P., Paczuski, M., Shubik, M.: Price variations in a stock market with many agents. *Physica A* 246, 430–453 (1997)
- [13] Boswijk, H.P., Hommes, C.H., Manzan, S.: Behavioral heterogeneity in stock prices. In: IFAC symposium Computational Economics, Finance and Engineering, Seattle, USA (2003)
- [14] Brenner, T.: Agent learning representation: advice in modelling economic learning. In: Judd, K., Tesfatsion, L. (eds.) *Handbook of Computational Economics. Agent-Based Computational Economics*, vol. 2. Elsevier Science B.V., Amsterdam (2005)
- [15] Brock, W., Hommes, C.H.: A rational route to randomness. *Econometrica* 65, 1059–1095 (1997)
- [16] Brock, W., Lakonishok, J., LeBaron, B.: Simple technical trading rules and the stochastic properties of stock returns. *Journal of Finance* 47, 1731–1764 (1992)
- [17] Caldarelli, G., Marsili, M., Zhang, Y.C.: A prototype model of stock exchange. *Europhysics Letters* 40, 479–484 (1997)
- [18] Campbell, J.Y., Lo, A.W., MacKinlay, A.C.: *The Econometrics of Financial Markets*. Princeton University Press, Princeton (1997)
- [19] Challet, D., Zhang, Y.C.: Emergence of cooperation and organization in an evolutionary game. *Physica A* 246, 407 (1997)
- [20] Challet, D., Marsili, M., Zhang, Y.C.: Modeling market mechanism with minority game. *Physica A* 276, 284–315 (2000)
- [21] Chan, N.T., Shelton, C.R.: An electronic market maker. Technical Report 200-005, MIT Artificial Intelligence Laboratory (2001)
- [22] Chan, N.T., LeBaron, B., Lo, A.W., Poggio, T.: Agent-based models of financial markets: A comparison with experimental markets. MIT Sloan Working Paper 4195-01, Massachusetts Institute of Technology (2001)
- [23] Chen, S.H., Yeh, C.H.: 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(3–4), 363–393 (2001)
- [24] Chen, S.H., Yeh, C.H.: On the emergent properties of artificial stock markets: the efficient market hypothesis and the rational expectations hypothesis. *Journal of Economic Behavior & Organization* 49(2), 217–239 (2002), <http://www.sciencedirect.com/science/article/B6V8F-45F900X-8/2/c034ae35c111ca061a11cae1df4b2cd5>
- [25] Chiarella, C., He, X.Z.: Asset price and wealth dynamics under heterogeneous expectations. *Quantitative Finance* 1, 509–526 (2001), citeseer.ist.psu.edu/chiarella01asset.html
- [26] Cincotti, S., Ponta, L., Raberto, M.: A multi-assets artificial stock market with zero-intelligence traders. In: WEHIA 2005, Essex, United Kingdom, June 13–15 (2005)
- [27] Cliff, D., Bruten, J.: Zero is not enough: On the lower limit of agent intelligence for continuous double auction markets. In: First Hewlett Packard International Workshop on Interacting Software Agents, Bristol, United Kingdom (1997)
- [28] Cliff, D., Miller, G.F.: Tracking the Red Queen: Measurements of Adaptive Progress in Co-Evolutionary Simulations. In: *Proceedings of the Third European Conference on Advances in Artificial Life*, pp. 200–218. Springer, London (1995)
- [29] Cont, R.: Empirical properties of asset returns: stylized facts and statistical issues. *Quantitative Finance* 1(2), 223–236 (2001), citeseer.ist.psu.edu/cont01empirical.html
- [30] Cont, R., Bouchaud, J.P.: Herd behavior and aggregate fluctuations in financial markets. *Macroeconomic Dynamics* 4, 170–196 (2000)

- [31] Dempster, M.A.H., Payne, T.W., Romahi, Y., Thompson, G.W.P.: Computational learning techniques for intraday FX trading using popular technical indicators. *IEEE Transactions on Neural Networks* 12(4), 744–754 (2001), <http://mahd-pc.jbs.cam.ac.uk/archive/PAPERS/2000/ieeetrading.pdf>
- [32] Edmonds, B.: Modelling bounded rationality in agent-based simulations using the evolution of mental models. In: Brenner, T. (ed.) *Computational Techniques for Modelling Learning in Economics*, pp. 305–332. Kluwer, Dordrecht (1999), citeseer.ist.psu.edu/edmonds99modelling.html
- [33] Edmonds, B., Moss, S.: Modelling bounded rationality using evolutionary techniques. In: *Evolutionary Computing, AISB Workshop*, pp. 31–42 (1997), citeseer.ist.psu.edu/edmonds97modelling.html
- [34] Edmonds, B., Moss, S.: The importance of representing cognitive processes in multi-agent models. In: Dorffner, G., Bischof, H., Hornik, K. (eds.) *ICANN 2001. LNCS*, vol. 2130, pp. 759–766. Springer, Heidelberg (2001)
- [35] Fama, E.F.: Efficient capital markets: A review of theory and empirical work. *The Journal of Finance* 23, 383–417 (1970)
- [36] Fama, E.F.: Efficient capital markets II. *The Journal of Finance* 46, 1575–1617 (1991)
- [37] Fama, E.F., French, K.: Size and book-to-market factors in earnings and stock returns. *The Journal of Finance* 50, 131–155 (1995)
- [38] Farmer, J., Joshi, S.: The price dynamics of common trading strategies. SFI Working Paper 00-12-069, Santa Fe Institute (2000), citeseer.ist.psu.edu/farmer00price.html
- [39] Farmer, J.D.: Market force, ecology, and evolution. *Industrial and Corporate Change* 11, 895–953 (1998), citeseer.ist.psu.edu/farmer98market.html
- [40] Farmer, J.D., Patelli, P., Zovko II: The predictive power of zero intelligence in financial markets. *Proceedings of the National Academy of Sciences of the United States of America* 102, 2254–2259 (2005)
- [41] Franci, F., Matassini, L.: Life in the stockmarket - a realistic model for trading (2000)
- [42] Garcia-Almanza, A.L., Tsang, E.P.: The repository method for chance discovery in financial forecasting. In: Gabrys, B., Howlett, R.J., Jain, L.C. (eds.) *KES 2006. LNCS*, vol. 4253, pp. 30–37. Springer, Heidelberg (2006)
- [43] Ghoulmie, F., Cont, R., Nadal, J.P.: Heterogeneity and feedback in an agent-based market model. *Journal of Physics: Condensed Matter* 17, S1259–S1268 (2005)
- [44] Giardina, I., Bouchaud, J.P.: Bubbles, crashes and intermittency in agent based market models. *The European Physical Journal B - Condensed Matter* 31, 421–437 (2003)
- [45] Gilli, M., Winker, P.: A global optimization heuristic for estimating agent based models. *Computational Statistics & Data Analysis* 42, 299–312 (2003)
- [46] Gode, D.K., Sunder, S.: Allocative efficiency of markets with zero intelligence (z1) traders: Market as a partial substitute for individual rationality. *GSIA Working Papers 1992-16*, Carnegie Mellon University, Tepper School of Business (1991)
- [47] Grothmann, R.: Multi-agent market modeling based on neural networks. PhD thesis, Faculty of Economics, University of Bremen (2002)
- [48] Hill, B.M.: A simple general approach to inference about the tail of a distribution. *Annals of Statistics* 3, 1163–1173 (1975)
- [49] Holland, J.H.: *Adaptation in Natural and Artificial Systems*. University of Michigan Press, Ann Arbor (1975)
- [50] Holland, J.H., Miller, J.H.: Artificial adaptive agents in economic theory. *The American Economic Review* 81, 365–370 (1991)
- [51] Jefferies, P., Hart, M., Hui, P., Johnson, N.: From market games to real-world markets. *The European Physical Journal B - Condensed Matter and Complex Systems* 20, 493–501 (April)

- [52] Juille, H., Pollack, J.B.: Dynamics of co-evolutionary learning. In: Maes, P., Mataric, M.J., Meyer, J.A., Pollack, J., Wilson, S.W. (eds.) *Proceedings of the Fourth International Conference on Simulation of Adaptive Behavior: From animals to animats 4*, pp. 526–534. MIT Press, Cambridge (1996), <http://www.demo.cs.brandeis.edu/papers/sab96b.pdf>
- [53] Kirman, A.P.: Whom or What Does the Representative Individual Represents? *The Journal of Economic Perspectives* 6, 117–136 (1992)
- [54] Kirman, A.P.: Ants, rationality and recruitment. *The Quarterly Journal of Economics* 108, 137–156 (1993)
- [55] Kirman, A.P., Vriend, N.J.: Evolving market structure: An ace model of price dispersion and loyalty. *Journal of Economic Dynamics & Control* 25, 459–502 (2001)
- [56] Koza, J.R.: Evolution and co-evolution of computer programs to control independent-acting agents. In: Meyer, J.A., Wilson, S.W. (eds.) *From Animals to Animats: Proceedings of the First International Conference on Simulation of Adaptive Behavior*, September 24–28, 1990, pp. 366–375. MIT Press, Paris (1991)
- [57] Langdon, W.B., Poli, R.: Fitness causes bloat. In: Chawdhry, P.K., Roy, R., Pant, R.K. (eds.) *Soft Computing in Engineering Design and Manufacturing*, pp. 13–22. Springer, London (1997)
- [58] LeBaron, B.: A builder's guide to agent based financial markets. *Quantitative Finance* 1, 254–261 (2001)
- [59] LeBaron, B.: Evolution and time horizons in an agent-based stock market. *Macroeconomic Dynamics* 5, 225–254 (2001)
- [60] LeBaron, B.: Calibrating an agent-based financial market. Working paper, Graduate School of International Economics and Finance, Brandeis University (2003)
- [61] LeBaron, B.: Agent-based computational finance. In: Judd, K., Tesfatsion, L. (eds.) *Handbook of Computational Economics. Agent-Based Computational Economics*, vol. 2. Elsevier Science B.V., Amsterdam (2005)
- [62] LeBaron, B.: Agent-based financial markets: Matching stylized facts with style. In: Colander, D.C. (ed.) *Post Walrasian Macroeconomics Beyond the Dynamic Stochastic General Equilibrium Model*. Cambridge University Press, Cambridge (2006)
- [63] LeBaron, B., Arthur, W.B., Palmer, R.G.: Time series properties of an artificial stock market. *Journal of Economic Dynamics & Control* 23, 1487–1516 (1999)
- [64] Levy, H., Levy, M., Solomon, S.: *Microscopic Simulation of Financial Markets: From Investor Behavior to Market Phenomena*. Academic Press, London (2000)
- [65] Levy, M., Solomon, S.: Dynamical explanation for the emergence of power law in a stock market model. *International Journal of Modern Physics C* 7, 65–72 (1996)
- [66] Levy, M., Levy, H., Solomon, S.: A microscopic model of the stock market: cycles, booms and crashes. *Economics Letters* 45, 103–111 (1994)
- [67] Li, J., Tsang, E.P.K.: Investment decision making using FGP: A case study. In: Angeline, P.J., Michalewicz, Z., Schoenauer, M., Yao, X., Zalzal, A. (eds.) *Proceedings of the Congress on Evolutionary Computation*, vol. 2, pp. 1253–1259. IEEE Press, Washington (1999), <http://www.cs.bham.ac.uk/~jxl/cercialink/web/publication/CEC99.pdf>
- [68] LiCalzi, M., Pellizzari, P.: Fundamentalists clashing over the book: a study of order-driven stock markets. *Quantitative Finance* 3, 1–11 (2003)
- [69] Lo, A., Mackinlay, A.C.: Stock market prices do not follow random walks: Evidence from a simple specification test. *Review of Financial Studies* 1, 41–66 (1988), citeseer.ist.psu.edu/1088stock.html

- [70] Lucas, R.E.: Adaptive behavior and economic theory. In: Hogarth, R.M., Reder, M.W. (eds.) *Rational Choice: The Contrast between Economics and Psychology*, pp. 217–242. University of Chicago Press (1986)
- [71] Lux, T.: Herd behaviour, bubbles and crashes. *The Economic Journal* 105, 881–896 (1995)
- [72] Lux, T.: 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 (1998)
- [73] Lux, T.: The limiting extremal behaviour of speculative returns: an analysis of intra-daily data from the frankfurt stock exchange. *Applied Financial Economics* 11, 299–315 (2001)
- [74] Lux, T., Ausloos, M.: Market fluctuations i: Scaling, multiscaling and their possible origins. In: Bunde, A., Kropp, J., Schellnhuber, H.J. (eds.) *Theories of Disaster - Scaling Laws Governing Weather, Body, and Stock Market Dynamics*, pp. 373–409. Springer, Heidelberg (2002)
- [75] Lux, T., Marchesi, M.: Scaling and criticality in a stochastic multi-agent model of a financial market. *Nature* 397, 498–500 (1999)
- [76] Malkiel, B.G.: *A Random Walk Down Wall Street*, 1st edn. W. W. Norton & Co., New York (1973)
- [77] Mandelbrot, B.B.: The variation of certain speculative prices. *Journal of Business* 36, 394–419 (1963)
- [78] Markose, S., Tsang, E.P.K., Martinez-Jaramillo, S.: The Red Queen principle and the emergence of efficient financial markets: an agent based approach. In: Angeline, P.J., Michalewicz, Z., Schoenauer, M., Yao, X., Zalzalá, A. (eds.) *8th Workshop on economics and heterogeneous interacting agents (WEHIA)*, vol. 2, pp. 1253–1259. Springer, Germany (2003)
- [79] Marsili, M.: Toy models of markets with heterogeneous interacting agents in economics with heterogeneous interacting agents. In: Kirman, A., Zimmerman, J.B. (eds.) *Economics With Heterogeneous Interacting Agents*, vol. 503, p. 161. Springer, Heidelberg (2001), citeseer.ist.psu.edu/marsili01toy.html
- [80] Marsili, M., Challet, D.: Trading behavior and excess volatility in toy markets. *Adv. Complex Systems* 1, 1–14 (2001)
- [81] Martinez-Jaramillo, S., Tsang, E.P.K.: An heterogeneous, endogenous and co-evolutionary gp-based financial market. *IEEE Transactions on Evolutionary Computation* (forthcoming)
- [82] Neely, C.J., Weller, P., Dittmar, R.: Is technical analysis in the foreign exchange market profitable? a genetic programming approach. *Financial and Quantitative Analysis* 32, 405–426 (1997)
- [83] Pagie, L., Hogeweg, P.: Information Integration and Red Queen Dynamics in Coevolutionary Optimization. In: *Proceedings of the 2000 Congress on Evolutionary Computation CEC 2000*, pp. 1260–1267. IEEE Press, Los Alamitos (2000), citeseer.ist.psu.edu/pagie00information.html
- [84] Palmer, R.G., Arthur, W.B., Holland, J.H., LeBaron, B., Tyler, P.: Artificial economic life: A simple model of a stock market. *Physica D* 75, 264–274 (1994)
- [85] Palmer, R.G., Arthur, W.B., Holland, J.H., LeBaron, B.: An artificial stock market. *Artificial Life and Robotics* 3, 27–31 (1999)
- [86] Paredis, J.: Coevolving Cellular Automata: Be Aware of the Red Queen? In: Baeck, T. (ed.) *Proceedings of the 7th Int. Conference on Genetic Algorithms (ICGA 1997)*. Morgan Kaufmann Publishers, San Francisco (1997)
- [87] Poli, R.: A simple but theoretically-motivated method to control bloat in genetic programming. In: *Proceedings of the 6th European Conference*, pp. 204–217. Springer, Heidelberg (2003)

- [88] Raberto, M., Cincotti, S.: Modeling and simulation of a double auction artificial financial market. *Physica A: Statistical Mechanics and its Applications* 355(1), 34–45 (2005)
- [89] Robson, A.J.: The evolution of rationality and the Red Queen. *Journal of Economic Theory* 111(1), 1–22 (2003)
- [90] Robson, A.J.: Complex Evolutionary Systems and the Red Queen. *Economic Journal* 115(504), F211–F224 (2005)
- [91] Schulenburg, S., Ross, P.: Strength and money: An LCS approach to increasing returns. In: Lanzi, P.L., Stolzmann, W., Wilson, S.W. (eds.) *IWLCS 2000*. LNCS, vol. 1996, p. 114. Springer, Heidelberg (2001), citeseer.ist.psu.edu/schulenburg01strength.html
- [92] Schulenburg, S., Ross, P.: Explorations in lcs models of stock trading. In: Lanzi, P.L., Stolzmann, W., Wilson, S.W. (eds.) *IWLCS 2001*. LNCS, vol. 2321, pp. 150–179. Springer, Heidelberg (2002), citeseer.ist.psu.edu/673677.html
- [93] Shiller, R.J.: From efficient market theory to behavioral finance. *Journal of Economic Perspectives* 17, 83–104 (2003)
- [94] Shleifer, A.: *Inefficient Markets: An Introduction to Behavioral Finance*. Clarendon Lectures in Economics. Oxford University Press, Oxford (2000)
- [95] Siegel, E.V.: Competitively evolving decision trees against fixed training cases for natural language processing. In: Kinnear Jr., K.E. (ed.) *Advances in Genetic Programming*, vol. 19, pp. 409–423. MIT Press, Cambridge (1994)
- [96] Simon, H.A.: A behavioral model of rational choice. *The Quarterly Journal of Economics* 69, 99–118 (1955)
- [97] Simon, H.A.: *Models of Bounded Rationality*. MIT Press, Cambridge (1982)
- [98] Simon, H.A.: A mechanism for social selection and successful altruism. *Science* 250(4988), 1665–1668 (1990), <http://www.sciencemag.org/cgi/content/abstract/250/4988/1665>, <http://www.sciencemag.org/cgi/reprint/250/4988/1665.pdf>
- [99] Sunder, S.: Market as an artifact aggregate efficiency from zero intelligence traders. In: Augier, M.E., March, J.G. (eds.) *Models of a Man: Essays in Memory of Herbert A. Simon*, pp. 501–519. MIT Press, Cambridge (2004)
- [100] Taylor, M.P., Allen, H.: The use of technical analysis in the foreign exchange market. *Journal of International Money and Finance*, 304–314 (1992)
- [101] Tesfatsion, L.: Agent-based computational economics: Growing economies from the bottom up. *Artificial Life* 8, 55–82 (2002), citeseer.ist.psu.edu/article/tesfatsion02agentbased.html
- [102] Tsang, E.P.K., Martinez-Jaramillo, S.: Computational finance. In: *IEEE Computational Intelligence Society Newsletter*, pp. 3–8. IEEE Press, Los Alamitos (2004)
- [103] Tsang, E.P.K., Li, J., Butler, J.M.: EDDIE beats the bookies. *Software: Practice and Experience* 28(10), 1033–1043 (1998), <http://www3.interscience.wiley.com/cgi-bin/abstract/10007354/%START>
- [104] Tsang, E.P.K., Li, J., Markose, S., Er, H., Salhi, A., Iori, G.: EDDIE in financial decision making. *Journal of Management and Economics* (2000), <http://privatewww.essex.ac.uk/~scher/EDDIE%20PROJ/Tsang-Eddie%-JMgtEcon2000.doc>
- [105] Tsang, E.P.K., Yung, P., Li, J.: EDDIE-automation, a decision support tool for financial forecasting. *Decision Support Systems*, Special Issue on Data Mining for Financial Decision Making 37(4), 559–565 (2004), <http://www.sciencedirect.com/science/article/B6V8S-4903GV9-1/%2/d6ba531a46ce45526ff9015e4447409a>

- [106] Tsang, E.P.K., Markose, S., Er, H.: Chance discovery in stock index option and future arbitrage. In: *New Mathematics and Natural Computation*, vol. 1, pp. 435–447. World Scientific, Singapore (2005)
- [107] van Valen, L.: A new evolutionary law. *Evolutionary Theory* 1, 1–30 (1973)
- [108] Westerhoff, F.H.: Multi-asset market dynamics. *Macroeconomic Dynamics* 8, 596–616 (2004)
- [109] Winker, P., Gilli, M.: Indirect estimation of the parameters of agent based models of financial markets. *Fame research paper*, University of Geneva (2001)
- [110] Yang, J.: The efficiency of an artificial double auction stock market with neural learning agents. *Evolutionary Computation in Economics and Finance*, 85–106 (2002)
- [111] Zimmermann, H., Neuneier, R., Grothmann, R.: An approach of multi-agent FX-market modelling based on cognitive systems. In: *Proceedings of the International Conference on Artificial Neural Networks (ICANN)*, Viena (2001)
- [112] Zimmermann, H., Neuneier, R., Grothmann, R.: Multi-agent modeling of multiple FX-markets by neural networks. *IEEE Transactions on Neural Networks*, Special issue 12, 735–743 (2001)

Classical and Agent-Based Evolutionary Algorithms for Investment Strategies Generation

Rafał Dreżewski, Jan Sepielak, and Leszek Siwik

Department of Computer Science

AGH University of Science and Technology, Kraków, Poland

{drezew, siwik}@agh.edu.pl

Summary. In this chapter an evolutionary system for generating investment strategies is presented. The algorithms used in the system (evolutionary algorithm, co-evolutionary algorithm, and agent-based co-evolutionary algorithm) are verified and compared on the basis of the results coming from experiments carried out with the use of real-life stock data. The conclusions drawn from the results of experiments are such that co-evolutionary and agent-based co-evolutionary techniques better maintain population diversity and generate more general investment strategies than evolutionary algorithms.

9.1 Introduction

Investing on the stock market requires the analysis of a great number of possible strategies (which securities should be chosen, when they should be bought and when sold). The majority of investment decisions are based on analyzing present and historical data since it allows for predicting future trends. The problem however is that the anticipation of future trends depends on many assumptions, parameters and conditions. As a result, the investor or the analyst is able to analyze only the small subset of all possible strategies, so the optimal investment strategy is usually not found [15].

The set of the strategies which consists of indicator function is infinite because the complexity of the strategy can be unlimited. Formulas of the given strategy are functions of hundreds (or thousands) of parameters. Complexity of the problem makes it impossible to use direct search methods and instead of it a heuristic approach has to be used. For instance *evolutionary algorithms* can be applied here.

Evolutionary algorithms are optimization and search techniques, which are based on the Darwinian model of evolutionary processes [2]. One of the branches of evolutionary algorithms are *co-evolutionary algorithms* [12]. The most important difference between them is the way in which the fitness of the individual is evaluated. In the case of evolutionary algorithms the fitness of the individual depends only on how “valuable” is the solution of the given problem encoded within its genotype. In the case of co-evolutionary algorithms the fitness of the individual depends on the values of other individuals’ fitness. The value of fitness is usually based on the results of tournaments, in which the given individual and some other individuals from the population are engaged. Co-evolutionary algorithms are generally applicable in the cases in which it is difficult or even impossible to formulate an explicit fitness function. Co-evolutionary

interactions between individuals have also other positive effects—for example maintaining population diversity [6]. Population diversity is especially important in the case of multi-modal optimization problems, multi-objective optimization problems, and dynamic environments.

Agent-based (co-)evolutionary algorithms have been proposed as the result of research on decentralized models of evolutionary computations. The basic idea of such an approach is the realization of evolutionary processes within a multi-agent system, which leads to very interesting class of systems: (co-)evolutionary multi-agent systems—(Co)EMAS [5]. Such systems have some features which radically differ them from “classical” evolutionary algorithms. The most important of them are the following: synchronization constraints of the computations are relaxed because the evolutionary processes are decentralized (individuals are agents), there exists the possibility of developing hybrid systems using many different soft-computing techniques within one single, coherent agent architecture, and finally there is the possibility of introducing new evolutionary and social mechanisms, which were hard or even impossible to introduce in the case of classical evolutionary algorithms. (Co)EMAS systems have been already applied to solving multi-modal and multi-objective optimization problems [7]. Another area of applications is the modeling and simulation of social and economical phenomena.

In the case of financial and economical computations (and also financial and economical modeling and simulation) we have to deal with dynamic environments and competing or co-operating economical and social agents. As it was said before, co-evolutionary techniques help maintaining population diversity, and agent-based co-evolutionary systems maintain the diversity even better. What is more, agent-based approach allows us to easily model social and economical agents and relations between them.

In the chapter the component-based system for generating investment strategies is presented. In the system three algorithms were implemented: a “classical” evolutionary algorithm, a co-evolutionary algorithm, and an agent-based co-evolutionary algorithm. These algorithms were assessed and compared during the series of experiments and these results are reported in this chapter. The chapter is then concluded with some avenues of future work being suggested.

9.2 Previous Research on Evolutionary Algorithms for Generating Investment Strategies

In recent years there has been a growing interest in applying biologically inspired algorithms to solving economic and financial problems [3, 4]. Below, a sample of applications of evolutionary algorithms in systems supporting investment-related decision making are presented. To the best of the authors’ knowledge, there have been no prior attempt to apply agent-based co-evolutionary algorithms in such systems.

S. K. Kasscieh, T. L. Paez and G. Vora used the genetic algorithm in constructing an investment decision-support system [10]. The tasks of the algorithm included selecting which companies to invest in. The time series of the considered companies were given. In their system some logical operations were carried out on the data. The genetic

algorithm was used to determine, which logical operators should be applied in a given situation.

O. V. Pictet, M. M. Dacorogna, R. D. Dave, B. Chopard, R. Schirru and M. Tomassini [11] presented the genetic algorithm for the automatic generation of trading models which used financial indicators. Three algorithms were implemented: a genetic algorithm (it converged to local minima and revealed a poor generalization capability), a genetic algorithm with fitness sharing [16] (it explored the search space more effectively and revealed better abilities to find diverse optima), and a genetic algorithm with a fitness sharing technique developed by authors themselves in order to prevent the concentration of the individuals around “peaks” of fitness function (it revealed the best capability of generalization). The proposed algorithms selected parameters for technical indicators and combined them to create new, more complex, ones.

F. Allen and R. Karjalainen [1] used genetic programming to finding trading rules for the S&P 500 index. Their algorithm was able to select the structures and parameters for rules. Each rule was composed of a function organized into a tree and a returned value (signal), which indicated whether stocks should be bought or sold at a given price. Components of the rules were the following: functions operating on historical data, numerical or logical constants, logical functions which allowed for the combination of individual blocks in order to build more complicated rules. The root function always returned a logical value which ensured the correctness of the strategy and the fitness measure was based on excess return from the buy-and-hold strategy. However the return did not take into consideration the transaction cost.

9.3 Evolutionary System for Generating Investment Strategies

In this section both the architecture of the component-based system for generating investment strategies as well as three algorithms (evolutionary algorithm, co-evolutionary algorithm, and agent-based co-evolutionary algorithm) used as computational components are presented and discussed.

9.3.1 The Architecture of the System

In Fig. 9.1 the high-level architecture of the system is presented. As can be seen, the system consists of the following basic components

- *DataSource*—this component supplies the data to the strategy generator. Historical stock data is used.
- *Functions*—it contains all classes which are necessary for creating formulas of strategies. It includes constructs which can carry out basic operations on formulas i.e.: initialization, exchange of single functions, adding new functions or removing existing ones. Formulas can be tested on data and, in such a case, the results will be returned.
- *SystemTester*—this component allows for testing of proposed strategies. It is able to prepare reports concerning the transactions and containing the information about the gained profit. It is used by the generation algorithms to estimate the fitness of the developed trading strategies.

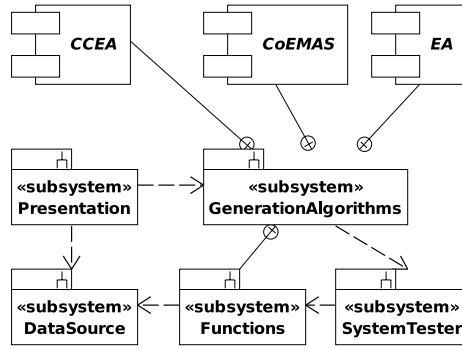


Fig. 9.1. The architecture of the system used in experiments

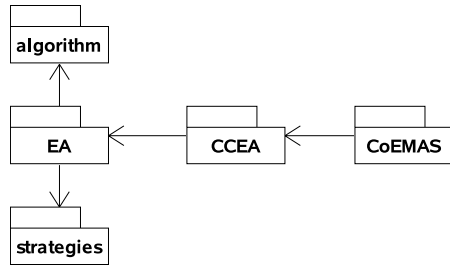


Fig. 9.2. Package diagram for generation strategies algorithms

- *GenerationAlgorithms*—contains the implementation of three algorithms responsible for generating strategies: evolutionary algorithm (EA), co-evolutionary algorithm (CCEA) and co-evolutionary multi-agent system (CoEMAS). This component includes the mutation and recombination operators as well as the fitness estimation mechanisms. Implementation of this subsystem was divided into packages shown in Fig. 9.2:
 - *algorithm* package—it is the most general package containing classes, which are the basis for implementation of other algorithms responsible for investment strategy generation.
 - *EA* package—it contains implementation of evolutionary algorithm.
 - *CCEA* package—it contains implementation of co-evolutionary algorithm.
 - *CoEMAS* package—it contains implementation of agent-based co-evolutionary algorithm.
- *Presentation*—it contains definitions of GUI forms responsible for instance for results presentation, algorithm monitoring etc.

9.3.2 Data Representation

In all three algorithms implemented in this chapter, a strategy consists of a pair of formulae. The first formula indicates when one should enter the market and the second

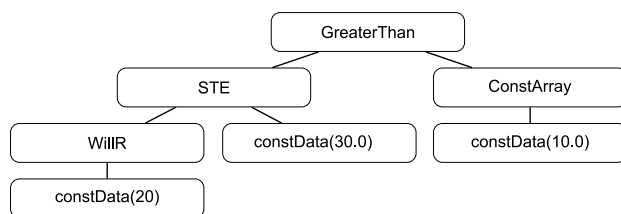


Fig. 9.3. Tree of sample formula

indicates when one should exit the market. Each formula can be represented as a tree in which the leaves and nodes are functions performing some operations. Each tree has a root and a number of child nodes. The root of the tree always returns a logical value. Fig. 9.3 shows the tree corresponding to the formula: $STE(WillR(20), 30) > 10.0$. When treating this expression as entry formula, the system will generate “buy” signal when the value of Standard Error indicator (STE) from 30 days and for data from Williams’ %R indicator for percentage period equal to 20, is greater than 10.0.

A formula tree is represented in memory as a tree. The root node is an object containing references to the functions and references to parameters. These parameters are also the same objects as the root. Leaves of the tree are objects which do not contain parameters. When a formula is executed recursive calls occur. In the beginning, the root requires values of all parameters needed to invoke its function. Then the control flows to objects of the parameters. Objects representing parameters behave in the same way as the root object. Leaves do not contain parameters, so they can return the value required by the parent node. Functions (which formulas are composed of) were divided into four categories

- Functions returning data arrays (see Table 9.1). There are 6 such functions.
- Mathematical functions (for example *Abs*—each value in the returned data array is absolute value of corresponding element from the data array passed as an argument, *Cos*—calculates cosine for values from the data array passed as an argument, etc.) There are 40 such functions.
- Tool functions (for example *ProjBandBot*—returns the data array with values of the bottom Projection Band, *STEBandBot*—returns the data array with values of the bottom Standard Error Band, etc.) There are 33 of them.
- Indicator functions (see Table 9.2). There are 14 of them.

There are 93 functions altogether.

Implemented functions accept the following types of parameters: constants (*integer*, *float* or *enum*), arrays of constant float values, and values returned by other function (arrays of logical values or arrays of float values).

Classes containing implementations of the above-mentioned functions are presented in Fig. 9.4. A *FunctionBase* class contains meta-data concerning all functions defined in descendent classes. To those meta-data belong for instance feasible values of functions arguments. *SecurityData* class contains session data of single stock. It supplies data to indicator functions. It also accumulates errors from formulas runs (e.g. division by zero, attempt of computing logarithm of negative value, etc.) and allows reporting of

Table 9.1. Data array functions

Function name	Description
Close	Returns closing prices.
ConstArray	Returns array of constants passed as argument.
High	Returns daily high prices.
Low	Returns daily low prices.
Open	Returns open prices.
Volume	Returns volumes.

Table 9.2. Indicator functions

Function name	Description
AD	Returns the data array with values of the Accumulation/Distribution indicator.
ADX	Returns the data array with values of Average Directional Movement indicator.
BBandBot	Returns the data array with values of the Bollinger Band Bottom indicator.
BBandTop	Returns the data array with values of the Bollinger Band Top indicator.
LinearReg	Returns the data array with values of the Linear Regression indicator.
Mov	Returns the data array with values of diverse moving averages.
ROC	Returns the data array with values of the Rate of Change indicator.
RSI	Returns the data array with values of the Relative Strength Index indicator.
STE	Returns the data array with values of the Standard Error indicator.
Stdev	Returns the data array with values of the Standard Deviation indicator.
TSF	Returns the data array with values of the Time Series Forecast indicator.
Var	Returns the data array with values of the Variance indicator.
WillR	Returns the data array with values of the Williams' %R indicator.
Zig	Returns the data array with values of the Zig Zag indicator.

these when it is required. Other classes contain implementation of indicator functions belonging to the categories mentioned earlier.

9.3.3 Evolutionary Algorithm (EA)

In the evolutionary algorithm, a genetic programming approach was used. The class diagram presented in Fig. 9.5 shows that each individual has simultaneously (in its genotype) two formula trees which represent formulas for entering and exiting the market. The process *computeFitness* is used to estimate fitness of an individual and this value is obtained using *getFitness*. It is also possible to get all components of fitness such as profit, average length of trades, and average complexity of formulas. An object of *FormulaTree* class contains the root of tree and the cached object of the formula, which is created from the tree using *createFormula*.

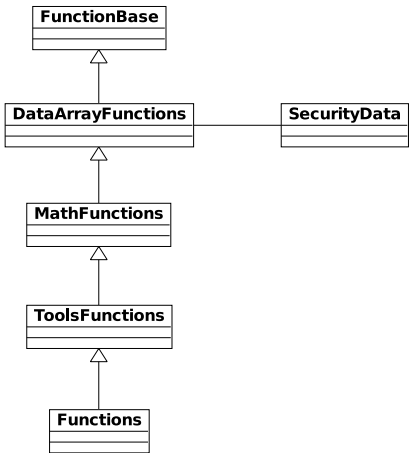


Fig. 9.4. Hierarchy of classes with functions used to build formulae

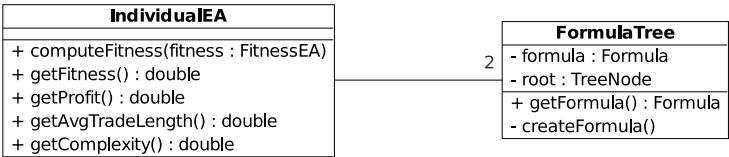


Fig. 9.5. Individual in EA and its genotype

Estimation of the fitness value of the strategies is carried out using historical stock price data. Two tables of logical values are created as a result of execution of the formulae. The first one relates to the purchase action (entering the market) and the second one relates to the sale action (exiting the market). The algorithm determines when a purchase and a sale occurs and computes the resulting profit/loss. Entering the market occurs when a system is outside the market and there is the value of “true” in the entry table. Exiting the market occurs when the system is in the market and there is value of “false” in the exit table. In other cases no operation is performed (hence, no short-selling is allowed). When applying this algorithm, the system cannot enter the market more than once. In other words, after a sale—a purchase action must take place, and after a buy—a sale must occur. The profit/loss from the transaction is estimated when exiting the market and it is accumulated. The cost of each transaction is included—the commission is calculated by subtracting a certain constant from the transaction value.

Apart from the profit/loss there are also other criteria which are included in the fitness estimation. The first one is formula complexity. Formulae which are too complex increase computational effort and can produce overfit. The complexity of the formulas is determined by summing up of all component functions. The second criteria is the length of the transaction. This depends on the preference of the investor.

Recombination

Three kinds of recombination operators are used: *returned value* recombination, *arguments* recombination, and *function* recombination. *Returned value* recombination is performed when there are two functions with different arguments but the same returned values within the formula tree. These functions are exchanged between individuals (functions are moved with their arguments).

For example, let us consider two formulae

- $ProjBandBot(34) < LLV(ProjBandBot(7), 7)$ (see Fig. 9.6), and
- $ProjBandTop(7) > STEBandBot(High(), 7, 7)$ (see Fig. 9.7),

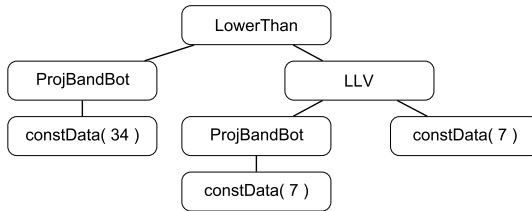


Fig. 9.6. Parent 1 for *returned value* recombination

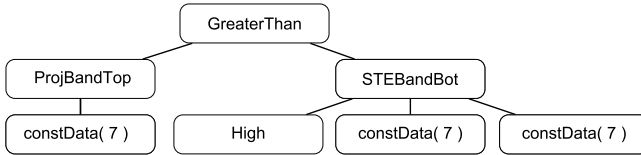


Fig. 9.7. Parent 2 for *returned value* recombination

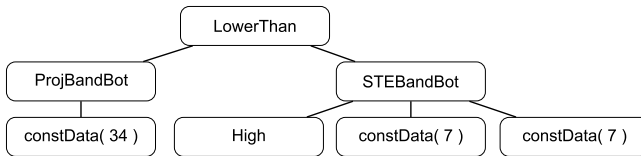


Fig. 9.8. Descendant 1 after *returned value* recombination

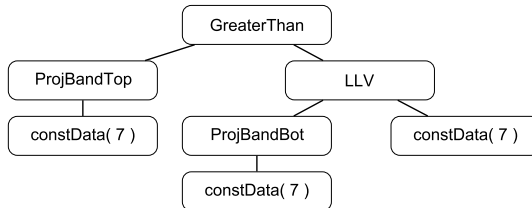


Fig. 9.9. Descendant 2 after *returned value* recombination

where *ProjBandBot* is function which calculates the bottom Projection Band indicator, *ProjBandTop* calculates the top Projection Band indicator, *LLV* (Lowest Low Value) is indicator, which calculates the lowest value in the data array over the preceding period, and *STEBandBot* calculates the bottom Standard Error Band indicator. Functions *ProjBandBot* and *STEBandBot* have different arguments, but the types of their return values are the same (they both return double values). As a result two new formulae will be created

- *ProjBandBot*(34) < *STEBandBot*(*High*(),7,7) (see Fig. 9.8), and
- *ProjBandTop*(7) > *LLV*(*ProjBandBot*(7),7) (see Fig. 9.9).

Arguments recombination occurs when there are two functions with the same arguments within the parents. These arguments are exchanged between individuals during the recombination. For example, let us consider two parents

- *BBandBot*(*Close*(),10,*Triangular*,4.3) (see Fig. 9.10), and
- *BBandTop*(*Open*(),3,*Simple*,2.9) (see Fig. 9.11),

where *BBandBot* and *BBandTop* calculate respectively bottom and top Bollinger Band indicator. Functions *BBandBot* and *BBandTop* have the same arguments. After exchanging these arguments two new descendants will be created

- *BBandBot*(*Open*(),3,*Simple*,2.9) (see Fig. 9.12), and
- *BBandTop*(*Close*(),10,*Triangular*,4.3) (see Fig. 9.13).

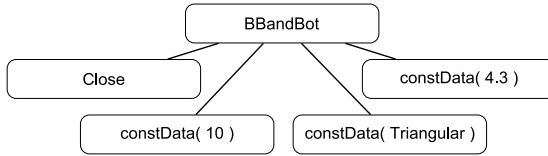


Fig. 9.10. Parent 1 for *arguments* recombination

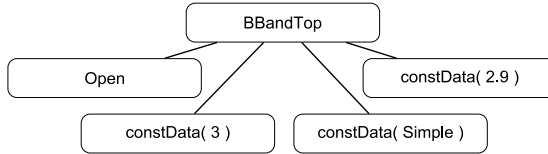


Fig. 9.11. Parent 2 for *arguments* recombination

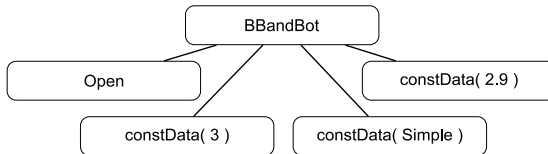


Fig. 9.12. Descendant 1 after *arguments* recombination

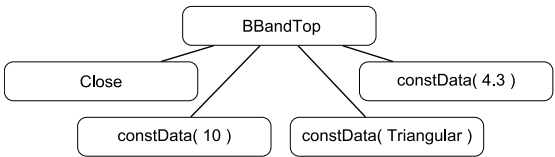


Fig. 9.13. Descendant 2 after *arguments* recombination

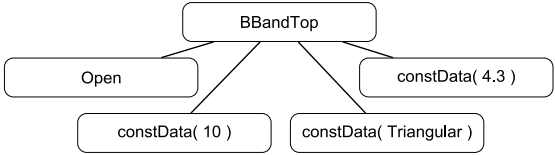


Fig. 9.14. Descendant 1 after *function* recombination

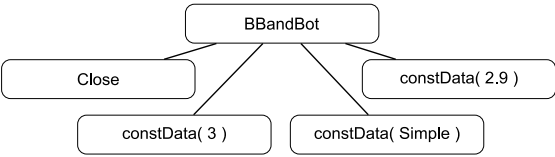


Fig. 9.15. Descendant 2 after *function* recombination

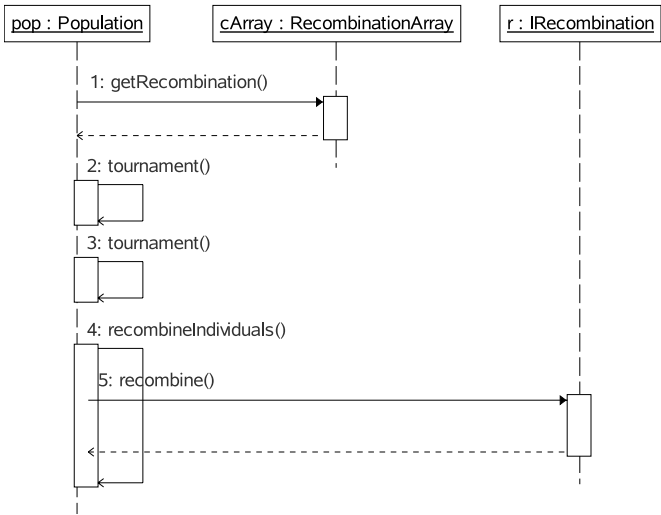


Fig. 9.16. Carrying out a single recombination

Function recombination can take place when two functions have the same arguments and the same returned values. Let us consider two parents from the previous example. Functions *BBandBot*, *BBandTop*, *Close* and *Open* fulfill the assumption which

requires return values and parameters to be the same. As a result of applying *function* recombination two new individuals will be created

- *BBandTop*(*Open*(), 10, *Triangular*, 4.3) (see Fig. 9.14), and
- *BBandBot*(*Close*(), 3, *Simple*, 2.9) (see Fig. 9.15).

Only one kind of recombination can be used to create an offspring in each iteration but the type of recombination (from the three possibilities) selected is governed by a probability distribution. The parameter *reproduction factor* determines how many offspring are generated. The size of population is multiplied by this coefficient. The outcome defines the number of offspring which are created.

During the recombination stage, in the first place, the type of recombination is chosen using *RecombinationArray* class. Next, parents are chosen using tournament selection. When two parents are selected, the selected type of recombination is performed. A sequence diagram for a single recombination is presented in Fig. 9.16.

Mutation

Two types of mutation were used: *function arguments* mutation and *function* mutation. In *function argument* mutation the argument of the function must be a constant value. This constant value is exchanged with the other one drawn from the allowed range. For instance, having a function *Add*(1, 2) operator can modify it to *Add*(5, 2). There are three variants of the *function* mutation

1. If a given function should be mutated, a list of the functions taking the same arguments is found. If such functions exist, the exchange is performed. If there are no such functions, mutation is not carried out. For instance *And*(*a*, *b*) can be changed to *Or*(*a*, *b*).
2. A given function can be exchanged with another, completely new one, regardless of arguments—but the returned types must match. Arguments of such function are created randomly. For instance:
 - *BBandTop*(*Month*(), 13, *E*, 4.714)—before mutation,
 - *BBandTop*(*HHV*(*Close*(), 3), 13, *E*, 4.714)—after mutation.

The *Month* function returns day of the month, which data came from, and the *HHV* function calculates the highest value in the data array over the preceding period. The function *Month* was exchanged for *HHV*, and its arguments were created randomly.

3. Similarly as in (2), but, if it is possible, parameters of the replaced function are copied to a new one. For instance, there is *Mod*(*Low*(), *AD*())—before mutation, and *STE*(*Low*(), 14)—after mutation. The *Mod* function (which calculates modulus) was exchanged for the *STE* function and the parameter *Low* was moved. The second parameter had to be created randomly because *STE* function requires an integer constant as an argument. It was named as the partial function mutation.

The probability of mutation depends on the depth in the formula tree. If it is closer to leaves the mutation probability is greater. Thanks to this, there is a greater chance that the mutation will cause small changes in the genotype (locality). Mutation factors are used while calculating mutation probability. These coefficients let the user specify

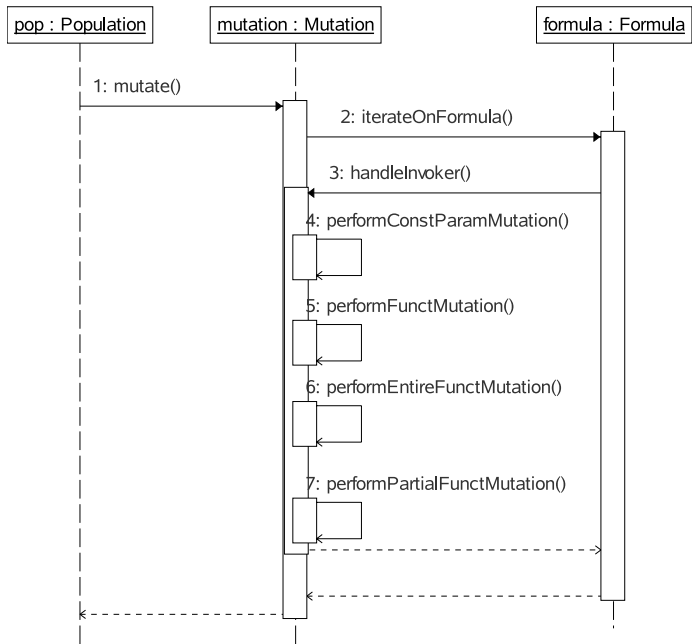


Fig. 9.17. Carrying out mutation of single formula element

how the likelihood of mutation changes as the depth in a formula of the mutated element changes. If such coefficient is greater than 1, then greater depth results in greater mutation probability. But if coefficient is in the range $[0, 1]$ then greater depth reduces mutation probability.

Fig. 9.17 shows the mutation sequence for a single formula element. The process *Mutate* from *Mutation* class is carried out on the genotype of an individual. Next, the *HandleInvoker* process is invoked on each function of genotype. Particular mutations are performed with a probability specified by the user.

Selection

Tournament selection ([2]) was used in the EA algorithm. In tournament selection, a group of N individuals ($N \geq 2$) is selected. The individual which has the highest fitness is chosen from this group.

After creating the offspring, it is added to the population of parents (the reinsertion mechanism). From such enlarged population the new base population was chosen also with the use of a tournament mechanism.

Algorithm 9.1 shows the scheme of the evolutionary algorithm. At the beginning the population is created randomly and evaluated. Next, reproduction, recombination and mutation are performed in each generation. When the individuals are created, their fitness value is estimated. Afterwards, a new population is selected by applying tournament reinsertion on the parent and child individuals.

Algorithm 9.1. Scheme of evolutionary algorithm (EA)

```

gen ← 0;
random initialization of population Pop(gen);
evaluate Pop(gen);
while not stop condition do
    Pop1(gen) ← reproduction Pop(gen);
    Pop2(gen) ← recombination Pop1(gen);
    Pop3(gen) ← mutation Pop2(gen);
    evaluate Pop3(gen);
    Pop(gen + 1) ← reinsertion (Pop3(gen) ∪ Pop(gen));
    gen ← gen + 1;
end

```

9.3.4 Co-Evolutionary Algorithm (CCEA)

Co-evolution is the process of the mutual adaptation of a set of individuals which interact with each other [8]. When an individual becomes better adapted, other individuals also have the opportunity to improve their fitness.

All evolutionary and co-evolutionary algorithms consist of the searching of a population of solutions in accordance with the concept of natural selection. Co-evolutionary algorithms however, differ from ordinary evolutionary algorithms. Individuals in the population may interact with other individuals. Partners of interactions may be members of the same (sub)population, or may belong to different (sub)populations, depending on the nature of the problem being solved.

Many real problems are difficult to solve using standard evolutionary techniques. However, such problems can sometimes be decomposed into many sub-problems, with the solution to these sub problems producing a solution to the initial problem. *Co-operative co-evolution* is designed to solve the problem X through co-evolving solutions for sub-problems which X was decomposed into [8].

While developing a co-evolutionary algorithm for generating investment strategies, a co-operative approach was proposed by M. A. Potter and K. A. De Jong ([14]). There are two species in the implemented algorithm: individuals representing entering the market strategies and individuals representing exiting the market strategies. Interactions between these species rely on co-operation.

The class diagram presented in Fig. 9.18 shows that each individual has in its genotype one formula tree representing the formula for entering or exiting the market depending on population to which the individual belongs. To estimate the fitness, the

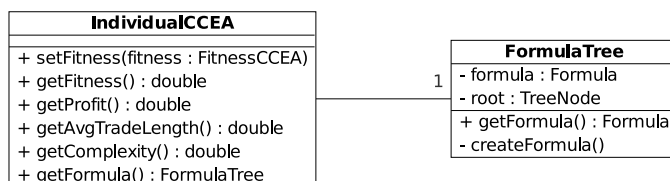


Fig. 9.18. Individual in CCEA and its genotype

formula is obtained using *getFormula* method and passed to an object of *FitnessCCEA* class. In *setFitness* method—fitness value and its elements are taken from the *FitnessCCEA* object.

During the fitness estimation process individuals are paired to form a complete solution. In the first generation, for each evaluated individuals from the first species a partner for co-operation from the second species is chosen randomly. For the complete solution created in this way, the fitness is computed and assigned to the individual that is being evaluated.

In the next generations, the best individual from the opposite species from the previous generation is chosen for the evaluated individual. For instance, if second generation has finished then at the beginning of third generation reproduction starts. After recombination and mutation new individuals are created and then they can be evaluated. To do this the best individual from the second generation of the *exit the market* population is retrieved. Then pairs of previously selected individual and each individual from third generation from the *enter the market* population are created. Fitnesses from such formed pairs are assigned to individuals which came from the enter the market population. In the similar manner fitnesses of individuals from the second population are established. The best individual from second generation of the enter the market population is selected. Pairs composed of this individual and each individual from the exit the market population (from third generation) are created. Estimated fitnesses are assigned to individuals from the exit the market population.

Algorithm 9.2 shows the scheme of the co-evolutionary algorithm. At the start of the algorithm, populations of two species are created randomly and fitness of each individual is estimated. Next, reproduction, recombination and mutation are applied on both species in each iteration. When offspring individuals are created, fitness estimation occurs. Both species interact at the stage of fitness computing. After that, reinsertion is performed and a single step of CCEA is finished. The pairs of individuals gaining the best profit are the final output of the algorithm.

Algorithm 9.2. Scheme of co-evolutionary algorithm (CCEA) [13]

```

gen ← 0;
foreach species s do
  random initialization of population  $Pop_s(gen)$ ;
  evaluate  $Pop_s(gen)$ ;
end
while not stop condition do
  foreach species s do
     $Pop_s^1(gen) \leftarrow$  reproduction  $Pop_s(gen)$ ;
     $Pop_s^2(gen) \leftarrow$  recombination  $Pop_s^1(gen)$ ;
     $Pop_s^3(gen) \leftarrow$  mutation  $Pop_s^2(gen)$ ;
    evaluate  $Pop_s^3(gen)$ ;
     $Pop_s(gen + 1) \leftarrow$  reinsertion ( $Pop_s^3(gen) \cup Pop_s(gen)$ );
  end
  gen ← gen + 1;
end

```


9.3.5 Co-evolutionary Multi-agent System (CoEMAS)

The Co-evolutionary multi-agent system used in this study is an agent-based realization of the co-evolutionary algorithm. Its general principles of functioning are in accordance with the general model of co-evolution in multi-agent systems [5]. The CoEMAS system is composed of the environment (which include computational nodes—*islands*—connected with paths) and agents, which can migrate within the environment. The selection mechanism is based on the resources, which are defined in the system. The general rule is such, that in the each time step the environment gives more resources to “better” agents and less resources to “worse” agents. The agents use the resources to perform each activity, like migration, reproduction, and so on. Each time step (the agents can live for more than one generation), individuals lose some constant amount of their resources, which is given back to the environment. The agents make all their decisions independently—especially those concerning reproduction and migration. They can also communicate with each other and observe the environment.

In the CoEMAS algorithm realized in the system for generating investment strategies, each co-evolutionary algorithm (CCEA) is an agent which is located on one of the islands. The population of each co-evolutionary algorithm also consists of the agents (there are two species of agents within each population, like in the case of CCEA).

Genetic operators and fitness estimation mechanism are the same as in the case of previously described algorithms. The selection mechanism is different—it works on the basis of resources (the agent possessing the greater amount of resource wins the tournament).

On the basis of the amount of the possessed resource, each individual decides whether it is ready for the reproduction. Reproduction occurs when the level of its resource is greater or equal to $r_{min}^{rep,\gamma}$ (see Algorithm 9.2). Parents are chosen using a tournament. At the mutation stage the amount of resource does not change. During the recombination process, parents give their offspring certain amount of their resources.

The tournament during the reinsertion phase is also based on the resources—the agent that possesses more resources wins the tournament. At the reinsertion stage better individuals receive more resources from the environment and the worse ones receive less. If an agent’s level of resources drops below r_{die}^γ , it dies.

The possibility of migration of agent-individuals from one population to another was added as well. During migration the resources possessed by a given agent are reduced by a constant amount.

The *reproduction factor* parameter in CoEMAS determines only the maximum number of offspring to be created. Practically, the amount of created individuals is smaller and depends on how many individuals have sufficient resources to reproduce. If it turns out that there are no such individuals, reproduction stops.

Individuals die when they possess too few resources. However, this approach is insufficient because after a few generations there will be many individuals in population which vegetate (do nothing and do not give back resource to environment) and therefore their existence is useless. For that reason, when individuals pass to new generation they give back a certain percent of their resources to environment.

The number of individuals in the population depends on the amount of resources in environment, and the initial size of population is specified by the user. If the

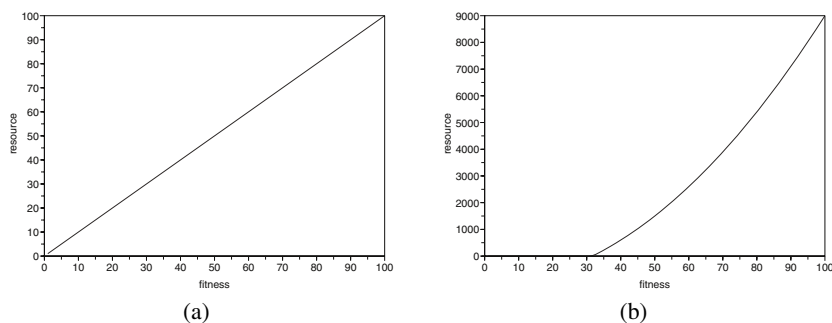


Fig. 9.19. Allocation of resource at initialization stage (a), and in consecutive generations (b)

environment has more resources, then more individuals can exist. If there is a small quantity of resources and these are shared between too many individuals, then many individuals will be killed at the reinsertion stage.

Each population lives in a certain environment (island). The environment possesses a specified amount of resources which circulates between the environment and agents. The resources of population can change only through migration (individuals which migrate take their resources to the another population). The amount of resources in the system is constant. In the first stage resources are allocated proportionally to fitness: the greater fitness the greater amount of resource allocated to an individual. The entire population receives resources (apart from individuals with non-positive fitness). Fig. 9.19a shows the manner of resource allocation at the initialization stage.

In the consecutive generations—in order to increase selection pressure—only the part of population receives resources. Resources are not allocated proportionally to the fitness value. The fitness value is raised to the power specified by the user and the resource is allocated on the basis of such modified fitness. The manner of resource allocation after initialization is shown in Fig. 9.19b.

9.4 The Experiments

In order to examine the generalization capabilities of the system and compare the proposed algorithms, strategies which earn the largest trading profit per year were sought. An attempt was made to determine which algorithm generalizes best and what quantity of stocks should be used for strategy generation so that the system would not overfit. It is also interesting to assess which algorithm generates the best strategies and has the smallest convergence.

9.4.1 Plan of the Experiments

The presented results of the experiments comparing the quality of the generated strategies and convergence properties of the considered algorithms are average values from 30 runs of each the algorithm. Each algorithm was run for 500 generations on the data of 10 randomly chosen stocks. The session stock data came from the WIG index ([9])

and the period of 5 years was chosen (from 2001-09-29 to 2006-09-29). The size of the population in all the algorithms was equal to about 40 individuals (the CoEMAS approach uses variable-size populations).

All experiments were made with the use of optimal values of the parameters. These values were found during consecutive experiments. The algorithms were run 10 times for each parameter value coming from the established range and average results were computed—on this basis the set of best parameter values were chosen. While comparing all algorithms three approaches were used

- on-line efficiency: $\frac{1}{T} \sum_{t=1}^T f(t)$, where T is the number of generations algorithm worked through, and $f(t)$ is fitness of the best individual in generation t .
- off-line efficiency: $\frac{1}{T} \sum_{t=1}^T f^*(t)$, where $f^*(t) = \max\{f(1), f(2), \dots, f(t)\}$
- the best value in the last generation: $\max_{i=1, \dots, N} f_i(T)$, N is the number of individuals in the last generation, and f_i is fitness of i -th individual.

While examining the generalization capabilities, each algorithm generated a trading strategy for for n random stocks (stage 1). Then n different stocks were chosen ten times at random and the profit was calculated using the best strategy obtained in the stage 1. Then, the average of these profits was counted. These calculations were carried out four times for $n = 3, 5, 7, 10$.

Like in the first type of experiments (when the quality of the solutions and the convergence properties were compared), populations had similar sizes in the case of all three algorithms. All experiments were carried out on a machine with one AMD Sempron 2600+ processor.

9.4.2 Parameter Value Selection

Table 9.3 and Table 9.4 show optimal parameter values which were determined using back to back experiments. Table 9.3 concerns all three algorithms. All algorithms have the same values of these parameters. Table 9.4 concerns only CoEMAS algorithm. Parameters in this table refer to migration and resources which occur only in CoEMAS.

In some cases, when changing values of parameters, no regularity of influence on efficiency changes was found. But there were some exceptions.

In Fig. 9.20 there is presented the influence of agent's minimal resource level on CoEMAS efficiency. In the beginning, with the growth of this parameter the value of efficiency slightly increases. But when resource amount exceeds 10 units it systematically decreases.

Fig. 9.21 presents the influence of the percent of resource obtained by a child after recombination in CoEMAS on its efficiency. It indicates that giving a lot of resource causes a decrease in efficiency of the algorithm. Whereas giving a small amount of resource causes the increase of system's efficiency.

Fig. 9.22 presents dependency between the amount of resource which individuals give back to environment at migration stage and CoEMAS performance. At the beginning when the amount of resource which is given back grows, efficiency grows too (the best individual achieved a fitness of about 430 units). Optimum is at about 14 units. Then, the increase in returned resource causes performance decrease.

Table 9.3. Optimal values of parameters for all three algorithms

Parameter	Parameter value
Fitness function	
Price for entering the market	Open
Price for exiting the market	Close
Entry commissions	0.0
Exit commissions	1.0
Kind of got commissions	Points (\$)
Transaction length weight	0.1
Profit weight	1.0
Formula complexity weight	0.2
Initialization	
Initial formula depth	4
Mutation	
Entire function mutation probability	0.3
Entire function mutation—if probability depends on depth	true
Entire function mutation factor	2.0
Partial function mutation probability	0.1
Partial function mutation probability—if probability depends on depth	true
Partial function mutation factor	1.5
Function mutation probability	0.03
Function arguments mutation probability	0.03
Formula depth change probability	0.01
Recombination	
Arguments recombination—usage probability	0.4
Arguments recombination—argument change probability	0.3
Function recombination—usage probability	0.4
Function recombination—function change probability	0.3
Return value recombination—usage probability	0.7
Return value recombination—function change probability	0.2
Population	
Population size	40
Reproduction	
Tournament size during reproduction	5
Reproduction factor	0.8
Reinsertion	
Tournament size during reinsertion	3

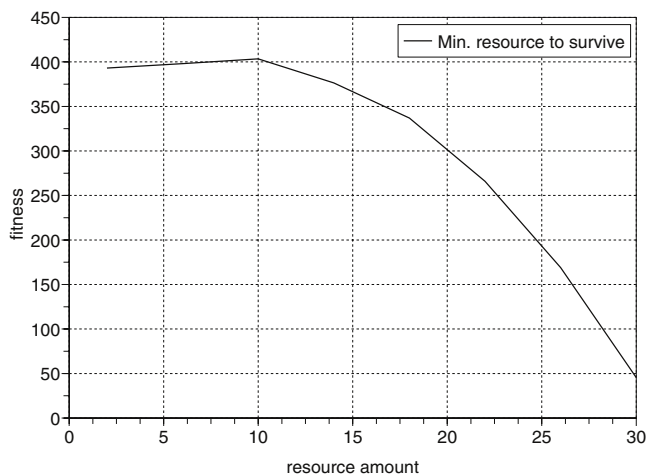
Fig. 9.23 presents influence of migration probability on CoEMAS efficiency. Only probability smaller than 0.15 causes high performance (the best individual achieved fitness of about 400 units). Efficiency decreases when probability is greater than 0.15.

9.4.3 Comparison of Algorithms

In EA and CCEA the size of populations during evolution is constant and is adjusted by the parameter specified by the user. Admittedly, the population size increases at

Table 9.4. Optimal values of additional parameters for CoEMAS

Parameter	Parameter value
Migration	
Number of generation from which mutation is started	5
Mutation probability	0.05
Resources	
Amount of resource required to reproduction	20.0
Amount of resource given back at migration	14.0
Minimal amount of agent's resource to survive	10.0
Initial amount of environment resource	980.0
Percent of resource received from parents	20.0
Percent of resource which environment loses at reinsertion	100.0
Percent of resource taken back at ageing	3.0
Percent of population which receives energy	20.0
Exponent of <i>pow</i> function at reinsertion	4.0

**Fig. 9.20.** The influence of minimal amount of resource for individuals to survive on results obtained by CoEMAS (each point is the average of 10 values)

recombination step, but after reinsertion becomes again the same as before the recombination. Different behavior can be observed in CoEMAS—as the size of the population varies during evolution.

Fig. 9.24 presents population size in CoEMAS during the evolutionary process. It shows that at the initial population is small (about 47 individuals) but after three steps it grows to 56 individuals. This is caused by the appearance of some very good individuals in populations. They receive a lot of resources from environment and create many equally good individuals in the reproduction stage. Because the quantity of the offspring is greater than the quantity of dead individuals, the size of population increases. When the ability of the algorithm to find better and better individuals diminishes, the

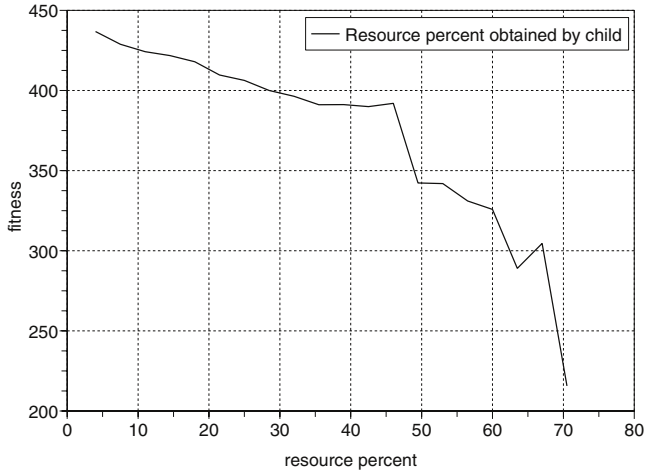


Fig. 9.21. The influence of the percent of resource obtained by a child after recombination on results obtained by CoEMAS (each point is the average of 10 values)

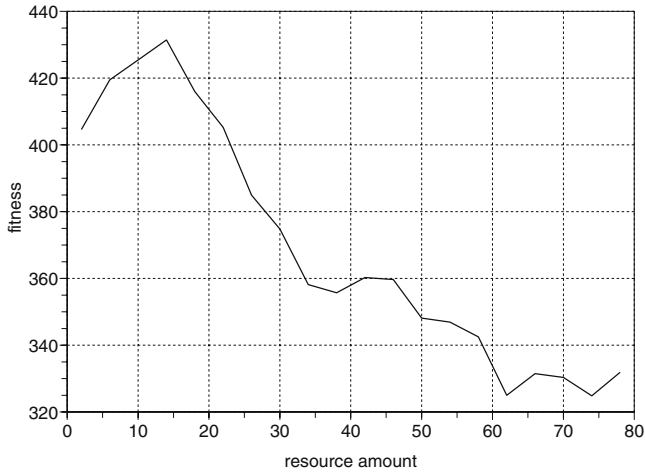


Fig. 9.22. The influence of the amount of resource given back at migration stage on results obtained by CoEMAS (each point is the average of 10 values)

size of population also decreases. When the algorithm no longer finds new and much better solutions (after about 100 generations) then, the population size does not change significantly.

Fig. 9.25 shows the average fitness (from 30 experiments) of the best individuals for each generation. The comparative results show that the evolutionary algorithm achieved the best results, outperforming the co-evolutionary algorithm. The quality of the solution generated by the CoEMAS was close to that of the CCEA.

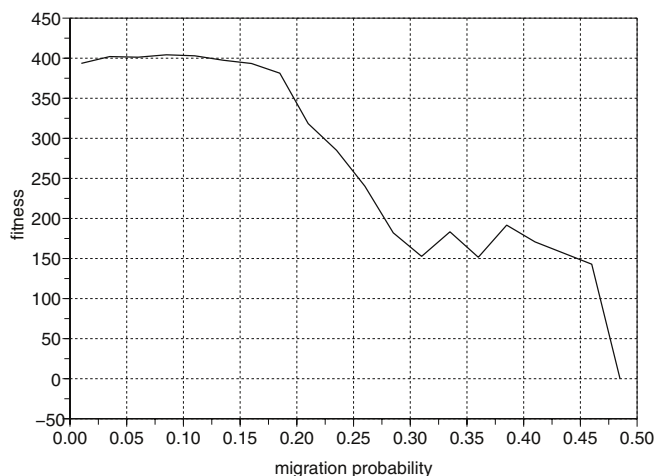


Fig. 9.23. The influence of the migration probability on results obtained by CoEMAS (each point is the average of 10 values)

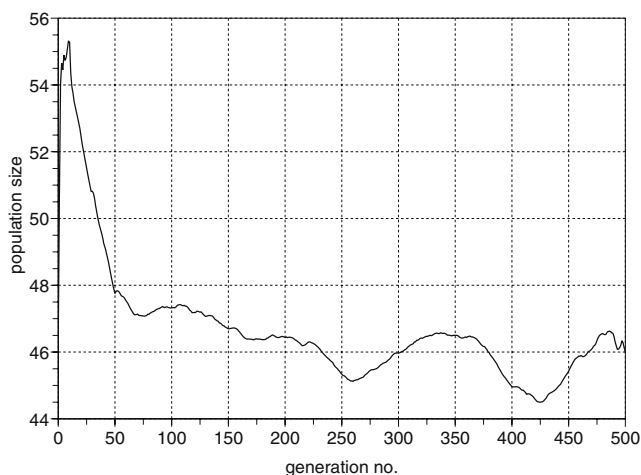


Fig. 9.24. Population size in CoEMAS during evolution

Fig. 9.26 presents a plot of solution convergence during the algorithm. It is manifested by the occurrence of multiple pairs of identical solution individuals in the population. In the case of convergence the evolutionary algorithm had the worst results. Convergence commenced early in the runs and from generation 200 it was typically in the range from 50% to 60%. The co-evolutionary algorithm exhibited less convergence, with CoEMAS displaying the least convergence.

Table 9.5 provides a comparison of the efficiency, run time and convergence of all three algorithms. The evolutionary algorithm had the best efficiency and the worst

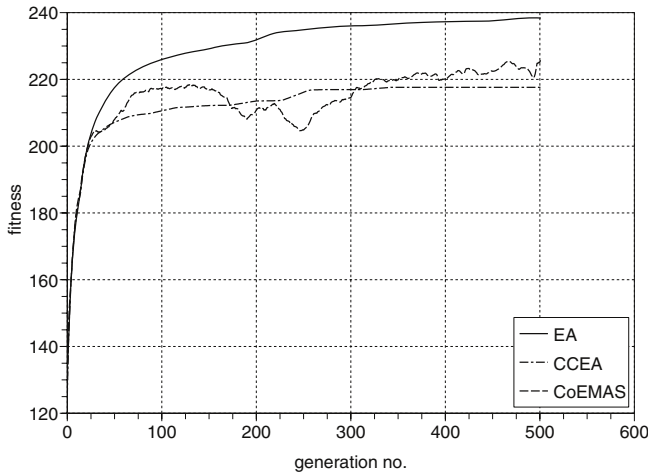


Fig. 9.25. Fitness of the best individuals (average values from 30 experiments)

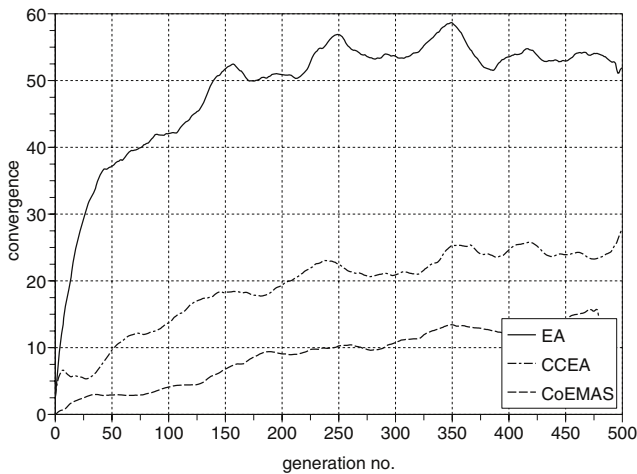


Fig. 9.26. Convergence values for three compared algorithms (average from 30 experiments)

results in the case of convergence, whereas the agent-based co-evolutionary algorithm was always better than the co-evolutionary algorithm with the exception of run time. The agent-based co-evolutionary algorithm had better off-line efficiency than the evolutionary algorithm because it found good solutions faster, however, those solutions were not good enough for the on-line efficiency to be better. Considering the run times of all algorithms, both evolutionary and co-evolutionary algorithms came off quite well. The co-evolutionary algorithm is somewhat worse because it processes two populations. The agent-based co-evolutionary algorithm has a run time six times longer than other algorithms. This arises as it is necessary to process two co-evolutionary algorithms in

Table 9.5. Comparison of efficiency, run time and convergence of all three algorithms

Algorithm	On-line ef- ficiency	Off-line ef- ficiency	The best fit- ness in last generation	On-line con- vergence	Convergence in last genera- tion	Algorithm run time (m:s)
EA	229.888	230.229	238.369	48.807	51.910	1:47.574
CCEA	213.069	213.071	217.629	19.145	27.147	1:50.213
CoEMAS	214.744	238.741	226.170	8.851	14.785	9:29.872

Table 9.6. Generalization capabilities of the algorithms

Algorithm	No. of stocks in the group	Profit (%) per year (stocks from the group)	Profit (%) per year for buy & hold strategy (stocks from the group)	Profit (%) per year (random stocks)	Profit (%) per year for buy & hold strategy (random stocks)
EA	3	133.24	86.24	8.79	34.81
	5	89.15	28.85	1.8	33.62
	7	68.32	21.43	13.74	23.72
	10	87.57	24.29	20.81	26.07
CCEA	3	115.63	46.12	3.29	21.07
	5	64.82	8.83	7.39	31.36
	7	85.75	45.24	14.69	24.64
	10	69.06	26.37	20.75	25.14
CoEMAS	3	87.39	12.64	3.97	30.28
	5	66.97	7.64	-1.93	20.72
	7	86.93	39.48	19.01	34.47
	10	46.67	18.67	24.6	24.98

two threads (computations were carried out on a machine with one processor). But the agent-based co-evolutionary algorithm can be easily distributed because of the decentralized nature of agent-based computations.

The generalizability of the strategies uncovered by each algorithm is examined in Table 9.6. The results show that while generating a strategy, at least 7 stocks should be used (see Table 9.6). If there are more stocks used during the strategy generation, the profit will be greater in the case of random stocks. For random stocks, when there was 3 or 5 of them in the group, the profits are unstable. For this reason it is difficult to compare implemented algorithms with a *buy and hold* strategy. It is not so, when the number of stocks in the group is 7 or 10. In the case of the random stocks, buy and hold strategy was always better (on average 2.67 times) than the strategies generated by all three evolutionary algorithms, but for the stocks from the learning set generated strategies were always better (on average 1.45 times) than a buy and hold strategy.

9.5 Summary and Conclusions

Generating investment strategies is a difficult problem because there exist many assumptions, parameters, conditions and objectives which impact on reported results. In the case of such problems finding the globally optimal solution is impossible in most cases and a sub-optimal solution is usually quite sufficient for the decision maker. In such cases some (meta-)heuristic algorithms like biologically inspired techniques and methods can be used. In this chapter a system for generating investment strategies which

uses three types of evolutionary algorithms was presented. The system can generate strategies with the use of a “classical” evolutionary algorithm, a co-evolutionary algorithm, and an agent-based co-evolutionary algorithm. These algorithms were tested using real-life data drawn from the WIG index.

The results obtained show that a classic evolutionary algorithm (GP) generated the individual (strategy) with the best fitness, followed by the best strategy generated by an agent-based co-evolutionary algorithm, and finally, the best strategy generated by a co-evolutionary algorithm. If population diversity (convergence) is examined, the results reverse: the best was CoEMAS, the second CCEA, and the worst results were reported in the case of EA. Such observations generally confirm that co-evolutionary and agent-based co-evolutionary algorithms maintain population diversity much better than “classical” evolutionary algorithms. This can lead to stronger abilities of such populations of solutions to “escape” from local minima. High population diversity is also a very desirable property in the case of dynamic environments (such as financial environments).

When we consider the generalization capabilities (profit gained from 7 and 10 random stocks during one year) of the strategies generated with the use of each evolutionary algorithm, the best results were obtained by CoEMAS (21.8% profit on the average), followed by CCEA (on the average 17.7%), and the worst results were obtained by the classic EA (17.3% on the average). The three algorithms provide better results than a buy and hold strategy for stocks from the learning set and worse results in the case of the random stocks.

Future research could concentrate on additional testing of the developed algorithms, and on the implementation and testing of other co-evolutionary mechanisms—especially in the case of the most promising technique: CoEMAS. Also, the implementation of the distributed version of the agent-based algorithm is included in future research plans.

References

- [1] Allen, F., Karjalainen, R.: Using genetic algorithms to find technical trading rules. *Journal of Financial Economics* 51(2), 245–271 (1999)
- [2] Bäck, T., Fogel, D., Michalewicz, Z. (eds.): *Handbook of Evolutionary Computation*. IOP Publishing and Oxford University Press, Oxford (1997)
- [3] Brabazon, A., O’Neill, M.: *Biologically Inspired Algorithms for Financial Modelling*. Springer, Heidelberg (2006)
- [4] Brabazon, A., O’Neill, M. (eds.): *Natural Computation in Computational Finance*. Springer, Heidelberg (2008)
- [5] Dreżewski, R.: A model of co-evolution in multi-agent system. In: Mařík, V., Müller, J.P., Pěchouček, M. (eds.) *CEEMAS 2003*. LNCS, vol. 2691, pp. 314–323. Springer, Heidelberg (2003)
- [6] Dreżewski, R., Siwik, L.: Techniques for maintaining population diversity in classical and agent-based multi-objective evolutionary algorithms. In: Shi, Y., van Albada, G.D., Dongarra, J., Sloot, P.M.A. (eds.) *ICCS 2007*. LNCS, vol. 4488, pp. 904–911. Springer, Heidelberg (2007)
- [7] Dreżewski, R., Siwik, L.: Co-evolutionary multi-agent system for portfolio optimization. In: [4], pp. 271–299 (2008)

- [8] Ficici, S.G.: Solution concepts in coevolutionary algorithms. PhD thesis, Brandeis University, Waltham, MA, USA (2004)
- [9] Historical stock data, http://www.parkiet.com/dane/dane_atxt.jsp
- [10] Kassicieh, S.K., Paez, T.L., Vora, G.: Investment decisions using genetic algorithms. In: Proceedings of the 30th Hawaii International Conference on System Sciences, vol. 5. IEEE Computer Society, Los Alamitos (1997)
- [11] Pictet, O.V., et al.: Genetic algorithms with collective sharing for robust optimization in financial applications. Tech. Rep. OVP.1995-02-06, Olsen & Associates (1995)
- [12] Paredis, J.: Coevolutionary algorithms. In: Bäck, T., Fogel, D., Michalewicz, Z. (eds.) Handbook of Evolutionary Computation, vol. (suppl. 1), IOP Publishing and Oxford University Press (1998)
- [13] Potter, M.A., De Jong, K.A.: A cooperative coevolutionary approach to function optimization. In: Davidor, Y., Männer, R., Schwefel, H.-P. (eds.) PPSN 1994. LNCS, vol. 866, pp. 249–257. Springer, Heidelberg (1994)
- [14] Potter, M.A., De Jong, K.A.: Cooperative coevolution: An architecture for evolving coadapted subcomponents. *Evolutionary Computation* 8(1), 1–29 (2000)
- [15] Tertitski, L.M., Goder, A.G.: Method and system for visual analysis of investment strategies. US Patent 6493681 (2002)
- [16] Yin, X.: A fast genetic algorithm with sharing scheme using cluster analysis methods in multimodal function optimization. In: Forrest, S. (ed.) Proceedings of the Fifth International Conference on Genetic Algorithms. Morgan Kaufman, San Francisco (1993)

Income Distribution and Lottery Expenditures in Taiwan: An Analysis Based on Agent-Based Simulation

Shu-Heng Chen¹, Bin-Tzong Chie¹, Hui-Fen Fan¹, and Tina Yu²

¹ National Chengchi University, Taipei, Taiwan 116
{chchen,g0258501,92921029}@nccu.edu.tw

² Memorial University of Newfoundland, St. John's, NL A1B 3X5 Canada
tinayu@cs.mun.ca

Summary. We investigate the impact of income distribution on lottery expenditures in Taiwan, using an agent-based model developed in [2, 3]. The agents in the model are potential lottery buyers, whose characteristics are described by three features: the percentage of income spent on the lottery, the preferences among lottery numbers selected and the aversion to regret. We used a genetic algorithm to drive the model simulation under agents with different incomes, based on household income data in Taiwan from 1979 to 2003. The simulation results indicated that the impact of income distribution on lottery sales is not significant. This might be due to the Taiwan economy having a minor degree of income variation which has a low effect on lottery expenditures.

10.1 Introduction

Lotteries are pervasive phenomena worldwide. Most modern lottery games are variations of the pari-mutuel “lotto” design: players select a set of numbers in a given range and win prizes according to how many numbers are guessed correctly. The winning prizes come from a proportion of the lottery sales (the pay-out) and the remaining portion (the take-out) is used to administer the game, pay the ticket outlets and cover tax liabilities and other charges. In many countries, lottery taxes have become a major source of charity and educational funds. In order to maximize tax revenues and minimize the negative social impact of gambling, governments consider socio-economic variables and demographic information in regulating lottery policies.

In Taiwan, the first lottery game, Lotto, was launched in January 2002, with a pay-out rate of 60%. The remaining 40% (take-out) is spent in two areas: 13.25% for administration expenses and 26.75% for government tax. Compared to the rest of the world, where lottery take-out rates vary between 68.4% and 40% (Data source: La Fleur’s Lottery World, U.S. Lotteries’ Unaudited FY00 Sales by Games. <http://www.lafleurs.com/>), Lotto has a relatively low take-out rate.

As the lottery market continues to grow, the Taiwanese government has become concerned about the determinants of the lottery expenditures, including income and socio-economic variables such as age, sex, and education, which might give rise to a “demographic” burden of the implicit lottery tax. As an initial effort to address these issues, this research investigates *income distribution* and its impact on lottery sales. This is an important subject as there have been many studies reporting the regressivity of lottery

tax: lotteries are disproportionately consumed by the poor, who become the heavy bearers of the implicit lottery tax [4, 5, 15]. The situation will become even worse if economic inequality also plays a role in lottery expenditures. In this research, we adopt an agent-based simulation approach to study this issue.

The lottery market is a dynamic entity whose macro behaviors are influenced by many factors, such as the size of jackpot rollover and the psychology of lottery players. While various mathematical models, such as multiple linear regression, have been used to study the socio-economic variables related to lottery expenditures [14, 15], these models do not capture the dynamic nature of the market. By contrast, agent-based models simulate the lottery market behaviors by designing agents as lottery players and implement various market scenarios in a systematic manner. The simulation results hence allow us to analyze the cause and effect of market phenomena that are difficult to trace using the traditional models. The agent-based simulation approach is becoming a promising alternative in conducting economic analysis [13].

The remainder of this chapter is organized as follows. Sect. 10.2 introduces the concept of income distribution and explains the lottery market's operations. Sect. 10.3 describes our design of a lottery market model using an agent-based system. In Sect. 10.4, the genetic algorithm used to drive the model's simulation is presented. We provide the experimental setups in Sect. 10.5. Based on the simulation results, we analyze the relationship between income distribution and lottery sales in Sect. 10.6. The tax revenues under different tax rates are evaluated in Sect. 10.7. In Sect. 10.8, we discuss rollover and its impact on lottery sales. Finally, Sect. 10.9 concludes the chapter with a brief outline of future work.

10.2 Background

Income distribution provides an indication of the economic inequality of a society. There are several metrics used by economists to measure income equality. This work has adopted two methods: the *5-range ratio* and the *Gini coefficient* [9]. In the 5-range ratio approach, the household income observations are divided into 5 groups from the lowest 20% to the highest 20%. The average income of the 80th percentile is then divided by the average income of the 20th percentile. A larger ratio means a higher degree of income inequality.

The Gini coefficient is derived from the Lorenz curve. To plot a Lorenz curve, the income observations are ranked from the lowest to the highest. The cumulative proportion of the *population* is plotted on the *x*-axis and the cumulative proportion of the *income* is plotted on the *y*-axis. Fig. 10.1 gives an example of a Lorenz curve. The diagonal line represents perfect equality. The greater the deviation of the Lorenz curve from this line, the greater the inequality. To calculate the Gini coefficient ratio, we have used the area between the Lorenz curve and the diagonal line as the numerator and the area under the diagonal line as the denominator. Thus, a low Gini coefficient ratio indicates high income equality. When the ratio is 0, it means perfect equality (everyone having exactly the same income). When the ratio is 1, it indicates perfect inequality (where one person has all the income, while everyone else has zero income). The Gini coefficient requires that no one has a negative net income or wealth.

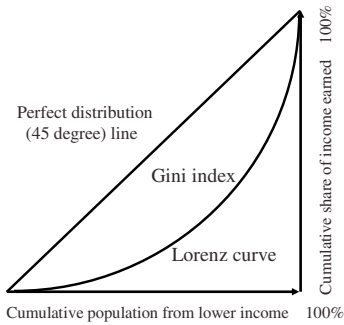


Fig. 10.1. Gini coefficient and Lorenz curve

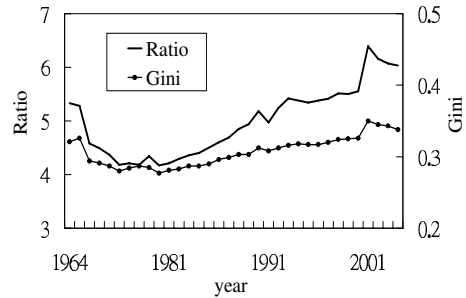


Fig. 10.2. Taiwan's income distribution

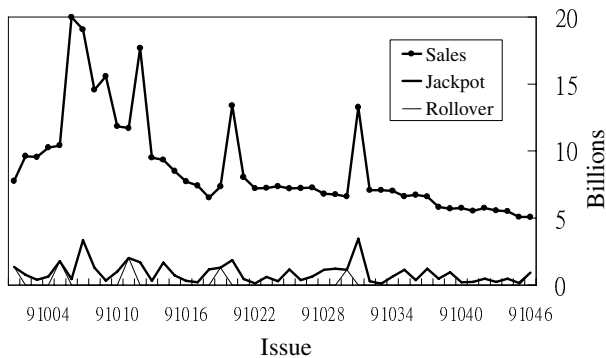


Fig. 10.3. Lotto sales volume (in \$NT) in 2002

Fig. 10.2 gives the household income distribution in Taiwan from 1964 to 2004, measured by the 5-range ratio and the *Gini coefficient* (Data source: <http://fies2.tpg.gov.tw/doc/result/93/211/Year04.doc>). The 5-range ratio from 1964 to 1971 ranged between 5.33 and 4.17. After that, the ratio first increased then started decreasing after 2001. The Gini coefficient index has a similar trend. The highest Gini index was 0.35 in 2001, which is still below the international standard of inequality threshold of 0.4. This indicates that the Taiwan economy has minor income variation. This research will study lottery sales in Taiwan based on this set of income data.

The two major lotteries sold in Taiwan are Lotto and Big Lotto. Lotto was first issued in January 2001. A Lotto player picks 6 out of 42 numbers to match the 6 winning numbers. If the picked numbers match all 6 winning numbers, the player wins the *jackpot*. If nobody wins the jackpot, the prize is rolled over to the next draw. In 2005, the rule was changed to pick 6 out of 38 numbers to promote sales. After the rule change, the odds of winning the jackpot was increased from 1.90629×10^{-7} to 3.62229×10^{-7} .

In 2004, a new game Big Lotto was introduced. This game has the same rules as that of the Lotto game except that the game player picks 6 out of 49 numbers. The odds of winning its jackpot is about 0.715×10^{-7} . When nobody wins the *jackpot*, the rollover frequently increases the sales of the next draw. Fig. 10.3 gives the Lotto sales volume

(in NT\$) in 2002. It is clear that during the rollover draws, such as issues 91006, 91012, 91020 and 91031, the sales increased dramatically. We will incorporate this market phenomenon to design the agents in our agent-based lottery market model, which is described in the following section.

10.3 An Agent-Based Model of Lottery Markets

An agent-based model typically has two parts: the environment and agent engineering. The environment is described by a set of rules governing the interactions of agents in the model while agent engineering involves the design of agents' representative characteristics. This research used the lottery market model built in [2, 3] to conduct the simulation. In this model, the following $x+4$ -tuple vector is used to describe the lottery market environment:

$$M = (x, X, \tau, s_0, \dots, s_x) \quad (10.1)$$

where x is the number of picks that a lottery player has to make from a total of X numbers. Depending on the number of matches between a player's picks and the winning numbers, different prizes are awarded.

Let y denote the number of matches, where $y = 0, 1, \dots, x$, and S_y is its prize. The prize with the highest amount is called the *Jackpot*. Each prize, S_y , is shared by all players who picked the numbers that matched the winning numbers. In the event when nobody wins a particular prize S_y , the amount is added to the next draw. When the *jackpot* prize is added to the next draw, it is referred to as a *rollover*. Rollovers usually attract more participants in the next draw, called the *rollover draw*.

The amount of monetary rewards to the lottery winners is governed by the lottery tax rate, τ . Let S be the total lottery sales, the *pay-out rate* is $1 - \tau$ and the total amount of monetary rewards is $(1 - \tau) \times S$. This amount is distributed among different prizes with rates $s_0 \dots s_x$, such that $\sum_{y=0}^x s_y = 1$. In other words, $S_y = s_y(1 - \tau) \times S$. Normally, s_y increases as y (the number of matches) increases.

The second part of the lottery market model is agent design. In a lottery market model, agents are potential lottery buyers. What motivates an individual to gamble? How much does one bet? We do not think there is a unique answer to these questions nor a single approach to address these issues. Among many possible ways of designing the agents, we focus on three features that capture the "stylized facts" of lottery markets: *participation level*, *conscious selection* and *aversion to regret*.

10.3.1 Participation Level

Participation level α is defined as the percentage of an individual's income I that is spent on the lottery. When considering the reasons why someone wants to buy lottery tickets, we perceive two types of possible influences: external and internal. The most noticeable external influence is the size of the jackpot. In Section 10.2, we have observed from the empirical data that the lottery sales increased during the rollover draws. We have therefore used the size of the jackpot as a factor that influences an agent's lottery participation. Another possible influence on individuals' decisions to purchase lottery tickets is their own internal subjective belief (probability) that they will win the jackpot.

We have also implemented this subjective belief of lottery winning as an alternative way to determine α . The simulation results from the two different implementations will be compared and discussed.

Lottomania and the Halo Effect

Lottomania refers to the phenomenon that sales following a *rollover* are higher than those of normal draws. Lottomania is mostly created by the media to arouse gamblers' desires to play the lottery. This effect may last for a few draws after the rollover draw [1]. Another related observation is that lottery sales are unexpectedly high right after a large jackpot prize is won. In the lottery industry, this is called the halo effect [7, 11]. The following equation defines an agent's lottery participation level based on the jackpot prize

$$\alpha = \rho(J) \quad (10.2)$$

where ρ is the participation function, α is the participation level, and J is the *jackpot* prize. This work assumes that agents base their decisions on some heuristics, and hence uses an if-then rule to represent ρ and approximate Equation (10.2). One example of a rule is "if the jackpot is unusually high, then I will spend 10% of my income to buy lottery tickets." The linguistic term *unusually high* in this rule is a common form of human reasoning. We have used fuzzy sets [16] to define linguistic terms and describe the value range of J in the if-then rule.

In this work, J is mapped into 4 linguistic terms (*Low*, *Medium*, *High* and *Huge*) by 4 different membership functions. A membership function decides the degree of membership of a value to a particular fuzzy set. For example, if the fuzzy set *High* and *Medium* are defined as

$$\text{Medium} = \{\text{jackpot} | 500,000 < \text{jackpot} < 1,200,000\}$$

$$\text{High} = \{\text{jackpot} | 1,000,000 < \text{jackpot} < 2,000,000\}$$

The jackpot prize of 1,500,000 belongs to *High* 100%, while the prize 900,000 belongs to *High*, maybe 90% and *Medium* 10%. The degree of membership of a value to a particular fuzzy set is decided by the membership function associated with that fuzzy set. In this study, we have adopted triangular-shape membership functions for simplicity. A triangular-shaped membership function is defined by two base points: left leg and right leg. They in turn give the peak of the triangle point (see Fig. 10.4).

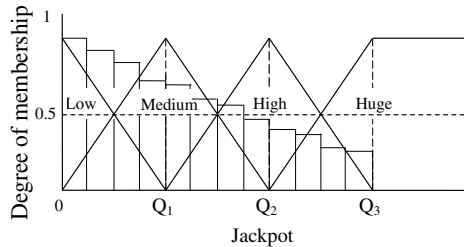


Fig. 10.4. Four triangular-shaped membership functions for the Jackpot prize

The four membership functions are not fixed but change over time depending on the historical jackpot prizes. In other words, the definitions of “Low”, “Medium”, “High” and “Huge” of a jackpot prize differ every day. On day t , the jackpot prizes from day 0 to day $t-1$ are analyzed based on their frequency under different prize ranges. The prize ranges with lower frequency normally have high values and are very motivating in terms of lottery purchasing. By contrast, those jackpot prizes with high frequency are normally with low values and are unattractive to lottery buyers. After arranging the prize ranges according to their frequency, we divided the entire jackpot prize ranges evenly into 3 partitions. (Q_1, Q_2, Q_3), which in turn give the base points of the four membership function. Fig. 10.4 shows this scheme of membership function design. The degree of membership to the four fuzzy sets is represented as a vector $\vec{\mu} = \{\mu_1, \mu_2, \mu_3, \mu_4\}$, where μ_1 is the degree of membership to “Low”, μ_2 is the degree of membership to “Medium”, μ_3 is the degree of membership to “High” and μ_4 is the degree of membership to “Huge”. For example, the jackpot prize 900,000 has $\vec{\mu} = \{0, 0.1, 0.9, 0.0\}$.

An individual view of the significance of a jackpot prize differs from one person to another. To customize the membership functions for each agent, an individual weight vector $\vec{a} = \{a_1, a_2, a_3, a_4\}$ is used, where each a_i is between 0 and 1. This vector is applied on the right-hand side of the if-then rule when calculating the participation level α_i for agent i

$$\alpha_i = a_{i1}\mu_1 + a_{i2}\mu_2 + a_{i3}\mu_3 + a_{i4}\mu_4 \quad (10.3)$$

This gives α_i a value between 0 and 1. While the value of $\vec{\mu}$ changes over time as the jackpot prizes change, each agent’s weight vector \vec{a} also evolves. The adaptation of \vec{a} is carried out by a genetic algorithm, which will be described in Sect. 10.4.

Subjective Belief

The second implementation of an agent’s lottery participation level is the agent’s subjective belief [3]. Let p_i be agent i ’s *subjective belief* (probability) that he/she will win the jackpot. Assume the agents are *risk-averse* with the log utility function $u(c) = \log c$. It can be shown that the optimal participation level α_i^* is

$$\alpha_i^* = \begin{cases} \frac{I_i - (\frac{I_i}{Jp_i})^{\frac{1}{1-p_i}}}{I}, & \text{if } \log I_i \leq p_i \log J + (1-p_i) \log(I_i - e), \\ 0, & \text{otherwise.} \end{cases} \quad (10.4)$$

where I_i is the income of agent i and e is the unit cost of the lottery ticket. Eq. (10.4) shows that both the size of jackpot (J) and the agent’s subjective belief (p_i) have a positive impact on the participation level α^* . However, the impact of income (I_i) on α^* , can not be analyzed easily. Do poor people spend a larger portion of their incomes on the lottery or it is the other way round? Research work on this issue has given rise to mixed reports [6, 14] and the answer to that question is inconclusive.

10.3.2 Conscious Selection

Despite the fact that the lottery winning numbers are generated randomly, lottery players tend to believe that one can predict the future winning numbers by analyzing the

winning numbers in the past, and hence choose numbers in a non-random manner. This is called *conscious selection* [8].

We implemented an agent's conscious selection using an X -dimensional vector \vec{b} whose elements take either "0" or "1" (recall X is the total number of possible selections that a lottery player can make). For a number z , where $1 \leq z \leq X$, if the z th element in \vec{b} is "1", the number z is consciously selected by the agent, while "0" indicates the opposite. Therefore, \vec{b} gives a list of numbers which are consciously selected by the agent. If \vec{b} has exactly x 1s, there is only one possible combination and that combination is used by the agent to purchase the lottery ticket(s). If \vec{b} has more than x 1s, there is more than one combination. The agent will randomly select one of them to purchase the ticket(s). Finally, if \vec{b} has less than x 1s, some random numbers will be generated to make up a total of x numbers for the lottery tickets.

10.3.3 Aversion to Regret

Regret aversion arises because of peoples' desire to avoid the pain of regret resulting from a poor decision. In the case of lottery purchases, the regret refers to the cost of not gambling after knowing somebody has won the lottery. Lottery promoters capitalize on the aversion to regret to encourage lottery buyers to keep on buying [12]. We have incorporated an agent's aversion to regret θ using the following equation

$$r_i = \begin{cases} -\theta_i I_i, & \text{if } \alpha_i^* = 0 \text{ and } N_x > 0, \\ \theta_i I_i, & \text{if } \alpha_i^* = 0 \text{ and } N_x = 0, \\ 0, & \text{otherwise.} \end{cases} \quad (10.5)$$

where θ_i denotes how regretful agent i , who did not purchase lottery ($\alpha_i^* = 0$), felt after knowing someone had won the jackpot, i.e., the number of winners of the jackpot is positive ($N_x > 0$). In the case when the jackpot is not drawn ($N_x = 0$), agent i may also derive pleasure from the gamblers' misfortune, and hence has a positive r_i value. If an agent did engage in lottery purchases, however, then there would be no such regretful effect, and r_i is 0. In summary, an agent i in the lottery market model is described by the following 3 characteristics

- *participation level*: α_i under lottomania implementation or α_i^* under subjective belief implementation.
- *conscious selection*: \vec{b}_i ;
- *aversion to regret*: θ_i ;

These agent properties evolve during the simulations using a genetic algorithm, which is explained in the following section.

10.4 Genetic Algorithms

Genetic algorithms (GAs) [10] start with a population of randomly generated individuals. With a defined fitness criterion, individuals are evaluated and selected for reproduction. The two most commonly used reproduction methods are crossover and mutation.

The generated offspring then form a new generation of population. This process of evaluation, selection and reproduction is repeated many times until a termination criterion is met. We will describe the implementation of our GA in the following sub-sections.

10.4.1 Representation

In the lottery market model, the population consists of a group of agents, each of which has the 3 characteristics, α/α^* , \vec{b} and θ , as described in the previous section. These three characteristics are represented using a linear chromosome. As given in Eq. (10.3), α is decided by two vectors \vec{u} , which is calculated from the historical jackpot prizes, and \vec{d} , which is an agent's individual weight vector. We encode \vec{d} in the GA representation using 4 binary bits for each of the four elements in the vector. The total number of bits representing \vec{d} is therefore 16. Each of the four binary bits is decoded to a real value between 0 and 1 using the following equation

$$a = \frac{\sum_{j=1}^4 c_j 2^{j-1}}{2^4 - 1} \quad (10.6)$$

where c_j is the j th bit value counted from the right. For example, 0011 is decoded as $\frac{2^0+2^1}{2^4-1} = \frac{3}{15} = 0.2$; 1001 is decoded as 0.6; 1100 is decoded as 0.8 and 1111 is decoded as 1.0. Assume $\vec{d} = \{0.2, 0.6, 0.8, 1.0\}$ and $\vec{\mu} = \{0, 0, 0.25, 0.75\}$. According to Eq. (10.3), the agent would invest $\alpha = 0.95$ of his income to purchase lottery tickets.

When the participation level is determined using subjective belief, the probability p is coded as a real number in the GA representation. This only takes one gene space. Based on the p , we can calculate α^* using Eq. (10.4). The coding of \vec{b} (conscious selection vector) is straightforward: it is a binary string of length X . Fig. 10.5 gives an example of the case where $X = 20$. The consciously selected numbers are 1, 6, 9, 11 and 12.

Aversion to regret, θ , is also a real number between 0 and 1. When the participation level (α) is decided using a fuzzy rule, θ is coded with a 4-bits binary string, so that the entire chromosome is a binary string, which is easier for genetic operation. Eq. (10.6) is used to decode the 4-bits binary string to a real number. In the case where the participation level (α^*) is decided by the agent's subjective belief (p), θ is coded as a real number taking one gene space in the chromosome. The value θ is used to calculate r using Equation (10.5), which is used to calculate the agent's fitness (see Section 10.4.2). In summary, when the agent's participation level is decided by a fuzzy rule, its representation contains $(\vec{d}, \vec{b}, \theta)$, which takes $20+X$ bits. If the agent's participation

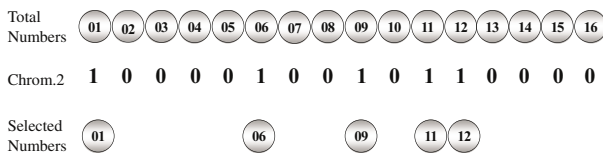


Fig. 10.5. An example of the selected lottery numbers

level is decided by its own subjective belief, its representation contains (p, \vec{b}, θ) , which has length $2+X$.

10.4.2 Fitness Function

Each agent is provided with different income, based on the Taiwan income data (see Table 10.2). The fitness of agent i is

$$F_i = I_i - G_i + \pi_i + r_i \quad (10.7)$$

where I_i is agent i 's income, G_i is the amount of money spent on the lottery (calculated from α_i), π_i is the prize won and r_i is the aversion to regret, per Eq. (10.5).

10.4.3 Genetic Operators

The genetic production starts from the selection of a mating pool. There are several different selection schemes in GA. However, to have a better focus, only tournament selection will be tried in this paper. By tournament selection, each individual in the mating pool is determined as follows. We first randomly select φ chromosomes without replacement, and then take the *best* two of them. The parameter φ is known as the *tournament size*, and it is also the mating pool size.

Given the mating, two genetic operators, *crossover* and *mutation*, are applied to the winning pair to generate two offspring. Since each chromosome contains three characteristics of an agent, crossover is restricted to swapping only the same characteristics between the two parent agents. We first randomly determine which one of the three characteristics of the crossover will take place. If it is on the bit-strings \vec{a} , \vec{b} or θ , the one-point crossover is applied with probability P_c . If it is on the real-valued p or θ , the arithmetic crossover is applied with the same probability P_c . The arithmetic crossover works by averaging the two parents' gene values as the gene value of the offspring.

After crossover, each of the two offspring has a probability of P_m to be mutated. For bit-strings \vec{a} , \vec{b} and θ , we apply bit mutation, i.e., 0 is flipped to 1 and 1 is flipped to 0. For the real-valued p and θ , we apply an arithmetic mutation, which is designed to be an equivalent of the bit-mutation shown in Eq. (10.8)

$$v^{new} = v^{old} + \sum_{i=1}^{16} B_{P_m} \left(\frac{1}{2}\right)^i \cdot (-1)^{B_{\frac{1}{2}}}, \quad (10.8)$$

where v^{old} and v^{new} are the gene values *before* and *after* the mutation. B_{P_m} and $B_{\frac{1}{2}}$ are the Bernoulli random variables with a success probability of P_m , and a mutation rate of one half, respectively. In this way, the arithmetic mutation size, denoted by σ , is $\sum_{i=1}^{16} B_{P_m} \left(\frac{1}{2}\right)^i \cdot (-1)^{B_{\frac{1}{2}}}$. When all offspring are produced, they replace the entire population and become the new generation for another evolutionary cycle [2].

10.5 Experimental Setup

The agent-based lottery market is defined by two sets of parameters; one is associated with the market (the top half of Table 10.1), and the other is associated with the GA

Table 10.1. Parameter values for the agent-based lottery market

Market parameters	
Pick x from X (x/X)	5/16
Lottery Tax Rate (τ)	10%,... 90%
Prize Distribution Rate ($s_0, s_1, \dots s_5$)	0%,0%,35%,15%,12%,38%
Drawing Period (w)	3 days
Number of Agents (N)	5,000
Agent Income (I)	see Table 10.2
GA parameters	
p_i value range	[0,0.003] (for α^*)
Crossover Rate (P_c)	90%
Mutation Rate (P_m)	0.1%
Arithmetic Mutation Size (σ)	in Equation (10.8)
Population Size	5,000
Tournament Size (φ)	200
Number of Generations	500
Number of runs (lottomania implementation)	25
Number of runs (subjective belief implementation)	50

(the bottom half of Table 10.1). In this lottery game, a player picks 5 out of 16 possible numbers. Players with 0 or 1 matching numbers will not receive any prize. 35% of the total prize pool is given to players with 3 matching numbers. The largest prize is the jackpot which receives 38% of the total prize pool. Note that the 2-matching-number winners receive a larger proportion of the total prize pool than the 3-matching-number and 4-matching-number winners. This is because the number of 2-matching-number winners is large, and hence a larger proportion of the prize pool is allocated to that prize. After the prize is divided among all winners, the individual prize for a 2-matching-number winner is still smaller than that for a 3-matching-number or a 4-matching-number winner.

The prize pool is the total lottery sales after the deduction of tax. Various lottery tax rates (τ) from 0% to 90% are implemented to investigate whether there is an optimal tax rate for the government to receive the maximum lottery tax revenue in this lottery market model.

The drawing period is the number of days between two draws. In this model, each period is 3 days long. On the first day, each agent is assigned with a different income (explained later in this section). During the 3-day period, all agents can purchase lottery tickets as desired. The lottery sales are added to the prize pool. At the end of the 3rd day, the winning numbers are drawn and the prizes are distributed. All prizes that are not won are rolled over to the next period.

After the prize distribution, the fitness of each agent in the population can be calculated using Eq. (10.7). Based on this fitness, selection and reproduction take place to generate a new generation of agents. This ends the current period and all new agents are allocated with new income to start a new period. This also means that each GA generation is equivalent to one period and is 3 days long.

Table 10.2. Average household income of Taiwan from 1979 to 2003

Year	First 20%	Second 20%	Third 20%	Fourth 20%	Highest 20%	The High-Low Ratio	Gini coefficient
1979-1983	87.12	137.55	175.66	227.33	372.34	4.27	0.2201
1984-1988	82.30	135.45	175.21	228.13	378.91	4.60	0.2286
1989-1993	74.81	132.66	175.64	231.83	385.07	5.15	0.2399
1994-1998	72.33	129.37	174.55	232.87	390.89	5.40	0.2469
1999-2003	68.05	124.96	171.82	232.15	403.01	5.92	0.2590
Fix Income	200	200	200	200	200	1.00	0.0000

Each agent is assigned with a different income using the data in Table 10.2. From the Taiwan income data, we selected 5 periods from 1979 to 2003 where income distributions have different degrees of variation. Each period is 5 years long. Their Gini coefficients lie between 0.22 to 0.26 and their 5-range ratios range from 4.27 to 5.92. For each of the periods, the household incomes are partitioned into 5 groups, from the lowest 20% to the highest 20%. The average incomes of each of the 5 groups are assigned to the agent population uniformly. We also conduct an extra set of experiments where all agents have the same income (200). This allows us to compare lottery sales under any income distribution that varies and with perfect distribution. For each of the 6 different income distribution setups, we conduct 25 simulation runs for the lottery market model under lottomania implementation and 50 runs for the lottery market model under subjective belief implementation. The average results are used in the analysis and discussion.

10.6 Lottery Sales vs. Income Distribution Analysis

We apply two statistical tests on the simulation data to determine whether the lottery sales are different under the 6 different income distributions. The first test is the Kolmogorov-Smirnov (KS) test and the second one is the Mann-Whitney-Wilcoxon (MWW) test. Both tests are non-parametric or distribution free methods as they do not assume that the data are drawn from a given probability distribution. The procedures for the two-sample KS-test are as follows

1. Calculate the cumulative frequency for the first data set $S_1(X)$;
2. Calculate the cumulative frequency for the second data set $S_2(X)$;
3. Find the greatest discrepancy between the two frequencies, which is called the “D-statistic”, $D_s = \max|S_1(X) - S_2(X)|$.
4. Compare this against the critical D-statistic (D) for that sample size.
5. If $D > D_s$, reject the null hypothesis that the two data sets are distributions of the same form.

The KS-test is more accurate when the sample size is small. Since our data series are collected from GA runs of 500 generations, the data size is considered to be large. We therefore use a second statistical test, MWW, to validate the first test results. The procedures for the MWW-test are as follows:

1. Combine the data from both data sets and rank each value;
2. Take the ranks for the first data set and sum them as W_1
3. Take the ranks for the second data set and sum them as W_2
4. Calculate $u_1 = n_1 n_2 + \frac{n_1(n_1+1)}{2} - W_1$; $u_2 = n_1 n_2 + \frac{n_2(n_2+1)}{2} - W_2$, where n_1 is the size of the first data set and n_2 is the size of the second data set.
5. $U = \min(u_1, u_2)$;
6. Calculate $Z = \frac{U - E(u)}{\sqrt{V(u)}}$, where $E(u) = \frac{n_1 n_2}{2}$ and $V(u) = \frac{n_1 n_2 (n_1 + n_2 + 1)}{12}$;
7. Compare Z against the critical Z -statistic (Z_s) for that sample size.
8. If $Z > Z_s$, reject the null hypothesis that the two data sets are distributions of the same form.

The 6 sets of simulation data from the 6 different income distributions setups under the 50% lottery tax rate ($\tau = 0.5$) are paired to carry out the 2 statistical tests. Table 10.3 gives the test results for the lottery market model under lottomania implementation and Table 10.4 gives the test results for the model with subjective belief implementation. The entries whose p-values are less than 0.05 are marked with an *.

The tests show that only income distribution in 1984 under the model with lottomania implementation results in lottery sales that are different from the sales under income with equal distribution (200): both of the p -values of the null hypothesis, based on D_s and Z , are almost nil. All other income distributions produce lottery sales that are not significantly different from each other. There are also a number of entries in the tables whose p-values are < 0.05 (with *). However, these entries do not demonstrate any consistent trend, and hence no conclusion can be drawn from them.

Table 10.3. Lottery market model with lottomania implementation

K-S test statistic ($\tau = 0.5$)						M-W test statistic ($\tau = 0.5$)				
Y	1979	1984	1989	1994	1999	1979	1984	1989	1994	1999
200	0.2370	0.0000*	0.1236	0.0258*	0.0590	0.4971	0.0000*	0.0232*	0.1073	0.1073
1979		0.0013*	0.2370	0.8774	0.4141		0.0041*	0.2216	0.4377	0.4263
1984			0.0258*	0.0258*	0.0104*			0.0625	0.0344*	0.0397*
1989				0.8774	0.6485				0.6554	0.4971
1994					0.8774					0.9536

Table 10.4. Lottery market model with subject belief implementation

K-S test statistic ($\tau = 0.5$)						M-W test statistic ($\tau = 0.5$)				
Y	1979	1984	1989	1994	1999	1979	1984	1989	1994	1999
200	0.3584	0.5077	0.3584	0.2408	0.0951	0.5649	0.4100	0.7020	0.4380	0.4628
1979		0.8409	0.8409	0.9541	0.6779		0.0076*	0.9862	0.7174	0.6766
1984			0.8409	0.9541	0.8409			0.7801	0.9478	0.9478
1989				0.8409	0.8409				0.7277	0.7485
1994					0.9541					0.9423

One possible reason why agents with different income distributions produce similar lottery sales is the low income variations of the data sets: the Gini indexes are very close to each other, between 0.22 and 0.26. We will perform simulations on data from other countries with a higher income inequality to verify this hypothesis.

10.6.1 Lottery Sales vs. Tax Rates

We also examine the lottery sales under different tax rates (τ) and income distributions. As shown in Figs. 10.6 and 10.7, the sales volume is the lowest when all agents have the same income of 200. In addition, the general trend is that the sales volume decreases as the tax rate (τ) increases. However, there are a couple of exceptions. For example, 1984 data have an increased sales volume when τ increased from 40% to 50%. Another important observation is that, under all income distribution data, the sales volume exhibits a sharp decline when the tax rates are higher than 80%. This indicates that lottery sales are strongly influenced by tax rates. We will analyze lottery tax revenues under different tax rates in the following section.

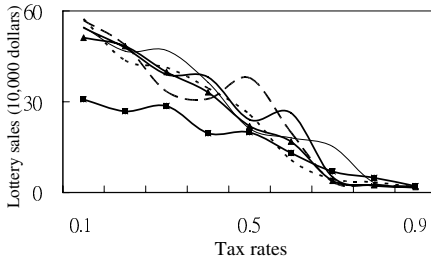


Fig. 10.6. Lottery sales vs. tax rates: lottomania implementation

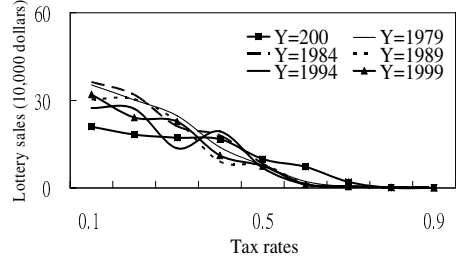


Fig. 10.7. Lottery sales vs. tax rates: subject belief implementation

10.7 Lottery Tax Revenue vs. Tax Rates Analysis

Lottery tax revenue (R) is affected by two factors: lottery sales volume (S) and the tax rate (τ) (see Eq. (10.9)). The two factors generate two counter-balancing forces, which are described in Eq. (10.10).

$$R = \tau S = \tau(\alpha I) \quad (10.9)$$

$$\frac{\partial TR}{\partial \tau} = \underbrace{S}_{+} + \tau \underbrace{\frac{\partial \alpha}{\partial \tau} I}_{-} \quad (10.10)$$

The positive force is characterized by the plus sign in Eq. (10.10), which says that given the sales volume S , the higher the lottery tax rate τ , the higher the tax revenue R . Meanwhile, we also expect a negative co-relation between lottery participation α and the tax rate τ , i.e., $\frac{\partial \alpha}{\partial \tau} < 0$. The second term in Eq. (10.10) hence has a negative value.

To plot the Laffer curves to depict the relationship between tax revenue (R) and the tax rate (τ), we first transform our simulation data in the following ways. For each of

the simulation runs, there are 500 generations, i.e., 500 lottery draws. At each draw t , tax revenue R_t is collected. The tax revenue for each simulation run is therefore a time series $\{R_t\}_{t=1}^{500}$. We normalize the revenue series by dividing R by the total income ($\sum_{i=0}^n I_i$, where n is the number of agents and I_i is the income of agent i) and call this new series $\{r_t\}_{t=1}^{500}$. Notice that the normalized tax revenue r_t can be interpreted as the *effective tax rate* or the *tax revenue*. To avoid the possible initialization biases, we removed the first 100 data points from the series and calculated the average of the rest of the 400 data points, \bar{r} . Since we made 25 runs for each tax rate, the median values are used to plot the Laffer curves in Figs. 10.8 and 10.9.

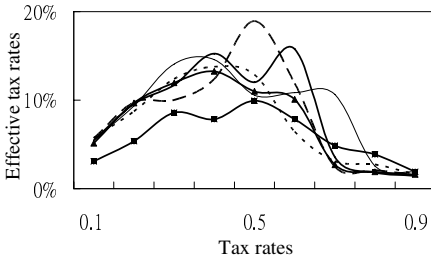


Fig. 10.8. Laffer curves: lottomania implementation

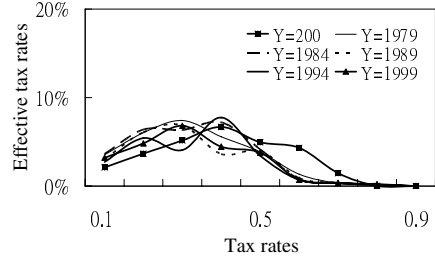


Fig. 10.9. Laffer curves: subjective belief implementation

As analyzed in Eqs. 10.9 and 10.10, the (normalized) tax revenue \bar{r} first increases with the lottery tax rate τ , and then decreases with it. The curves also show that under different income distributions, there are different optimal tax rates τ that give the optimal tax revenue \bar{r} . Under lottomania implementation, the optimal τ is between 40% and 60% which generates an \bar{r} of between 10% and 19%. Under subjective belief implementation, the optimal τ is between 30% and 40% which generates an \bar{r} of between 6% and 8%.

We also evaluate the uncertainty of \bar{r} using box-whisker plots. Since the plot pattern is similar for all simulation results from the 6 different income distribution data, we only show two of them, one for the lottomania implementation (Fig. 10.10) and one for the subjective belief implementation (Fig. 10.11). In a box-whisker plot, the box in the middle covers 50% of the simulated tax revenue. The longer the box, the more uncertain the tax revenue is. The two box-whisker plots show that the tax revenue is relatively low and stable when the tax rate is at its two extremes ($\tau = 10\%$, 90%). The box starts to inflate when the tax rate is moving toward the center, which signifies the growing uncertainty in tax revenue. The degree of uncertainty is further compounded by the enlarging whiskers, which extend the box to the frontier of the sample distribution. The high degree of uncertainty makes it unclear if there exists an optimum tax rate τ that gives the maximum revenues.

10.8 Discussion on Rollovers and Sales

Many researchers have reported that a large size of rollover would make the lottery more attractive and increase sales [1, 7]. A previous work applying statistical tests to lottery

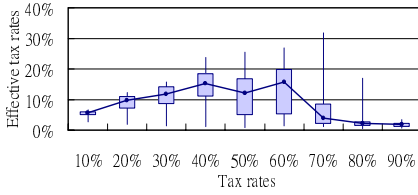


Fig. 10.10. Box-whisker plot: lottomania implementation

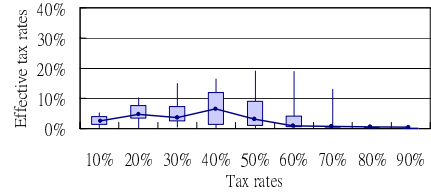


Fig. 10.11. Box-whisker plot: subject belief implementation

sales data from 7 countries also supports this proposition [2]. In this study, we have applied the same statistical tests to the simulation data generated from our experiments to examine if the same phenomenon appears in our lottery market model.

For each experimental run, we collected the sales from each draw. The data that were from rollover draws and from the regular draws were then separated into two data sets. Next, we conducted statistical tests to evaluate whether the two data sets were significantly different one from the other. Fig. 10.12 and Fig. 10.13 provide the t-statistics.

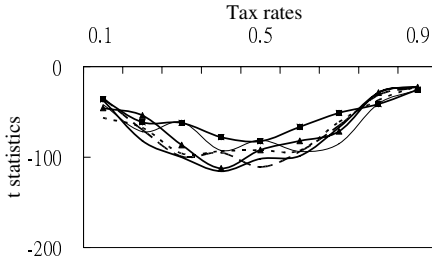


Fig. 10.12. Rollover vs. sales: lottomania implementation

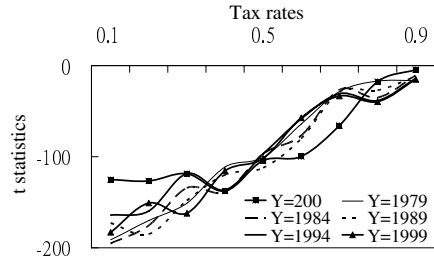


Fig. 10.13. Rollover vs. sales: subjective belief implementation

All t-statistics on data from the simulation runs under 6 different income distributions and 10 different tax rates have negative values. This means that the lottery sales for the rollover draws have a decreased volume, which is the opposite of what has been observed in the real lottery markets. A similar result was reported previously using the same agent-based modeling system under agents with equal income [2, 3]. The explanation presented there can be applied to our case.

In general, GA learning favors agents who win the lottery and who propagate those winning agents' characteristics (α_i , \vec{b}_i , θ_i) to the following generations. However, most gamblers do not win the prizes, and hence will end up with less money than the non-gamblers. In other words, the rank of the agents based on the money they have after the draws is as follows: winning gamblers, non-gamblers, and losing gamblers. During the rollover draw, there was no jackpot winner in the last draw. All winning gamblers won small prizes. The non-gamblers' agents hence have good chances to be propagated from

the last draw (generation) to the rollover draw (generation). With these non-gambler agents, the lottery sales in the current draw (rollover draw) are reduced. However, if there is a jackpot winner in the previous draw, the situation is completely reversed. The jackpot winner's characteristics would be greatly propagated to the current generation, since the winner has very high fitness (the amount of money after the draw). Consequently, the lottery sale in the current draw (the draw after the jackpot winning draw) is increased.

10.9 Concluding Remarks

While the lottery has been widely adopted in many countries to raise charity and educational funds, there are concerns about its side-effects, such as the regressivity of the lottery tax and the addiction to gambling. Various studies have been devoted to identifying these side-effects so that the government can regulate related policies.

This chapter presents a study on the impact of income distribution on lottery expenditures in Taiwan based on the simulation of an agent-based model. The simulation results show that the impact is not significant enough for the government to raise any concerns. Although the lottery market model is simple and does not reflect the real market perfectly, it provides a vehicle to study the issues that are difficult to investigate using traditional models such as regression. We will continue this research by improving the model with more sophisticated agent design and incorporating different algorithms to drive the simulation. Meanwhile, we plan to investigate other socio-economic issues related to lottery expenditures based on the newer model.

Acknowledgments

The authors are grateful to two anonymous referees for their very painstaking review of the paper. The research support in the form of NSC grant No. NSC. 95-2415-H-004-002- MY3 is gratefully acknowledged.

References

- [1] Beenstock, M., Haitovsky, Y.: Lottomania and other anomalies in the market for lotto. *Journal of Economic Psychology* 22, 721–744 (2001)
- [2] Chen, S.H., Chie, B.T.: Lottery markets design, micro-structure, and macro-behavior: An ACE approach. *Journal of Economic Behavior and Organization* 67, 463–480 (2008)
- [3] Chen, S.H., Chie, B.T.: Agent-based modeling of lottery markets: Policy-making from the bottom up. In: Dennard, L., Richardson, K.A., Morcol, G. (eds.) *Complexity and Policy Analysis-Tools and Concepts for Designing Robust Policies in a Complex World*. ISCE Publishing (2008)
- [4] Clotfelter, C., Cook, P.: *Selling Hope: State Lotteries in America*. Harvard University Press, Cambridge (1989)
- [5] Kitchen, H., Powells, S.: Lottery expenditures in Canada: A regional analysis of determinants and incidence. *Applied Economics* 23(2), 1845–1852 (1991)
- [6] Deboer, L.: Lottery taxes be too high. *Journal of Policy Analysis and Management* 5(3), 594–596 (1984)

- [7] Farrell, L., Morgenroth, E., Walker, I.: A time series analysis of U.K. lottery sales: Long and short run price elasticities. *Oxford Bulletin of Economics and Statistics* 61(4), 513–526 (1999)
- [8] Farrell, L., Hartley, R., Lanot, G., Walker, I.: The demand for lotto: The role of conscious selection. *Journal of Business and Economic Statistics* 18(2), 228–241 (2000)
- [9] Hall, R.E., Lieberman, M.: *Economics: Principles and Application*, updated 2nd edn., pp. 365–367 (2003)
- [10] Holland, J.H.: *Adaptation in Natural and Artificial Systems*. University of Michigan Press, Ann Arbor (1975); MIT Press (2nd edn.)
- [11] Matheson, V.A., Grote, K.: Examining the ‘Halo Effect’ in Lotto Games. *Applied Economics Letters* 14(4-6), 307–310 (2007)
- [12] Statman, M.: Lottery players / stock traders. *Financial Analysts Journal* 58(1), 17 (2002)
- [13] Tesfatsion, L.: Agent-based computational economics: A constructive approach to economic theory. In: Tesfatsion, L., Judd, K.L. (eds.) *Handbook of Computational Economics, Agent-Based Computational Economics*. Handbooks in Economics Series, vol. 2. North-Holland, Amsterdam (2006)
- [14] Vrooman, D.H.: An economic analysis of the New York state lottery. *National Tax Journal* 4, 482–490 (1976)
- [15] Worthington, A.C.: Finance in gambling expenditures: Australian evidence on socio-economic and demographic tax incidence. *Public Finance Review* 29(4), 326–342 (2001)
- [16] Zadeh, L.A.: Fuzzy sets. *Information and Control* 8, 338–353 (1965)

The Emergence of a Market: What Efforts Can Entrepreneurs Make?

Shuyuan Wu¹ and Anthony Brabazon^{2,3}

¹ Department of Management Faculty of Business, AUT University, Auckland, New Zealand
shuyuan.wu@aut.ac.nz

² Natural Computing Research and Applications Group, Complex & Adaptive Systems
Laboratory, University College Dublin, Ireland
anthony.brabazon@ucd.ie

³ School of Business, University College Dublin, Dublin, Ireland

Summary. This chapter, using a complex adaptive systems (CAS) approach, models how entrepreneurs create markets for a new, disruptive, technology through an effectuation process. Starting from dispersed knowledge components held by both the demand and supply sides, a market emerges from the interactive learning behaviours of entrepreneurs and potential customers. The CAS approach enables investigation of both system-level emergence and the process of dynamic co-evolution at the individual level. The results indicate that the process of market creation is significantly impacted by factors including exploration tendency, alertness, and participant prior knowledge.

11.1 Introduction

Complex and adaptive systems have interacting components whose individual behaviours and interactions lead to system-level emergent phenomena [49]. Economies, and individual firms within economies, present a rich ecology of interacting processes. Schumpeter [42] pictures an economy as a complex system with existing and emerging industries undergoing creative destruction leading to continual adaptation within the economy. Adopting a firm-level unit of analysis, Penrose [35] presents the firm as a system of wealth production and knowledge application, with the firm's productive resources being the components of this system. In a general sense, organisational scientists have long treated firm-level organizations as complex adaptive systems, the components of which are internal decision-making mechanisms [8, 54]. To investigate firm-level organisational changes, simulation models have been developed to treat strategic adaptation and punctuated equilibrium as results of subsystem interactions via basic learning processes [5, 26, 27]. This study takes a similar approach to examine the creation of markets.

Markets entail the interaction of multiple buyers and suppliers with each influencing the behaviour of the other. For a market to exist, the demand side has to perceive that there is value in the suppliers offering and in turn, this requires that the suppliers have some understanding of customers' needs. In this sense we can see a market as a body of knowledge, having converged from the two sides. The supply side configures a *value proposition*, a combination of technology and other components which promise

to satisfy customers' needs; on the demand side, *customer perceived value* (CPV), the customers' assessment of a solution's overall capacity to meet their needs, comes as a composite of multidimensional concerns and evaluations [25]. Knowledge components originate from both sides and evolve in an entangled way. For example, the creation of a telephone which is 'mobile' and 'wireless' was realised by supply side decisions on combining the enabling technology components - wireless communication, microelectronics, telephony, to name just a few. Meanwhile the concept of a 'mobile-phone' was perceived as providing value by customers as they evaluate and compromise between various issues in concern, including, for example, sound quality, the size and weight of the handset, battery life etc.

Knowledge '*never exists in concentrated or integrated form, but solely as the dispersed bits of incomplete and frequently contradictory knowledge which all the separate individuals possess*' ([20], p. 519). No matter which side initiates it, the innovation or market creation journey is a process by which the social needs and technological possibilities meet and shape each other to converge [3, 56]. From the perspective of a supply-side firm starting its market creation journey with a new technology, answering the question of what values to propose and for whom involves critical decisions under uncertainty. As what the end-solution should be is unknowable to both the supply and demand sides, the answer is to be agreed through an *effectuation* process, '*a process that continually transforms existing realities into possible markets*' ([40], p. 544). This process occurs through the commitments of individuals to networking [39] so that technical possibilities and needs are reconfigured and refined through social interactions. So in general, market creation is the collective learning process of solution-formation.

To combine dispersed and incomplete knowledge components, of what is needed with what is possible, into an end-solution a decision-maker from either the demand or supply side starts the learning process with what she currently knows and has, involves other stakeholders whose knowledge components are recognised as relevant, and with them (collectively) bounds the uncertainty out in the environment 'by deeming irrelevant a wide variety of information that may be available' ([40], p. 534). So individuals and firms from the demand and supply sides commit to the effectuation process, learning about and from each other in order to create a market as an institution of bounded cognition under uncertainty [40]. Such a community temporarily agrees on what needs are most relevant and how this need is to be satisfied [39]. The construct of such institutions or communities takes interactive learning and transformation.

Sarasvathy and Dew [40] describe the emergence of a new market as setting a thin interface (an artefact) between two hierarchical levels of complex adaptive systems ([40], p. 550)

The new market, however, gets fabricated, not through the designs of any one person, but as a chain of interactive commitments that form the interface between the inner environment of the effectual network [of committed members forming the community], and the outer environment ...

To create a 'market' as bounded cognition - an artefact to which the firm and its customers are committed, the effectuation process from the point of view of a supply-side firm is a learning process: learning to expand knowledge and to converge constraints

[40]. By involving and interacting with the demand side, an option of what value to create, and for whom to create it, is taken by suppliers.

This chapter investigates how a new technology competes with its older predecessor, and how a market is created for the new technology via an effectuation process involving potential customers. This process brings about a technological disruption when new effectuation networks emerge, producing a new system attractor, as suggested by Rosenkopf and Tushman ([38], p. 404)

This community of organizations evolves as new organizations introduce technological discontinuities, as coalitions form around technological substitutes, as incumbent organizations resist these efforts as substitution, and as interorganizational processes of compromise and accommodation affect a dominant design.

CAS provides a set of tools and frameworks for investigating emergent phenomena. Unlike for example, laboratory-based sciences, it is not possible to ‘rerun’ the creation of a real-world market under different circumstances in order to assess the influence of different variables or different market structures. Indeed, the emergent nature of the process of market creation would render such attempts at ‘understanding’ problematic. Hence, a simulation Agent-based Modelling (ABM) approach provides a good methodology in or attempts to understand the complex process of market creation.

We investigate the emergence new markets, by modelling the behaviour of a population of individual learning agents (suppliers and users). These agents can learn from each other and we can examine the outcome of these interactions - the emergence of a viable market. An artificial fifty dimensional (50D) problem space is generated within which the agents can interact. The traits and behaviours of the agents are modelled by synthesizing concepts from bounded cognition, individual learning processes and marketing. In the simulations we will observe how the agents can co-evolve so as to combine their knowledge components in new ways leading to a new (higher) customer perceived value (CPV). The constructed ABM simulator is used to test the influence of some fundamental factors on market emergence, including: individual learners’ exploitation and exploration tendency; their alertness in searching for relevant information; and their prior knowledge.

The remaining sections in the chapter are organized as follows. We initially review the literature of organisational learning and technology entrepreneurship to identify and synthesize key factors influencing the cognitive behaviours. The construction of our simulator and the implementation of individuals’ learning behaviours in this simulator are then reported. Next we provide the simulation results and discuss these. Finally, we conclude the chapter.

11.2 The Building Blocks and Propositions

A complex phenomenon at any level can only be explained by studying the entities and their interactions at one level down the hierarchy (components or subsystems). To understand how a machine or living body works, we study its component parts and investigate how they interact. Likewise, to understand how a market is created, we need

to see how learners from the potential demand-side and supply-side come to understand each other. To understand how the CPV of a radically new technology emerges, we need to study the sub-product level knowledge components to see how they are brought together.

From the point of view of individual actors or components, the future of any complex adaptive system is unpredictable as the path to the future is stochastic. However, system level order emerges from individuals' actions and interactions [52, 53]. An entrepreneurial firm with a new technology as its core resource, the customers whose needs are satisfied by what the firm offers, and competitors offering similar solutions constitute a market or technological community

The lags (temporal or otherwise) between any invention and the creation of new economic welfare enabled by it, require not only the ability and alertness to recognize, and the perception and perseverance to discover opportunities for the achievement of pre-determined goals such as increasing profits and larger market shares, but also necessitate decisions and actions based often only on human imagination, and human aspirations, that may or may not in time lead to new products, firms and markets. ([41], p. 159)

Widget X

A technology entrepreneurial firm begins its effectuation journey from the local reality of its initial conditions - its technology core - and some prior knowledge about established markets and the customers within them. Sarasvathy and Dew [40] illustrate the uncertainty of market creation for new technologies with Goodman's ([16]) *grue paradox*.¹

The effectuation process starts with some knowledge component(s) - or 'widget X'. In general, widget X can be any component of a future market including demand side elements (such as needs and wants), or supply side components (such as inventions, ideas about product and/or service, as well as institutional structures of a market such as channel, regulatory infrastructure, or standards bodies) ([40], p. 547). Using the 'grue' paradox as an analogy, future grass cannot be predicted to be green or blue. Widget X can be further developed into either green or blue (or for that matter, any other colours) end-products and thus 'the history of technological invention is full of unanticipated economic consequences' ([41], p. 142). To the extent that end product from (any) widget X is unformed and negotiable, the market is not to be 'discovered' but rather will *emerge* through transformation. The entire process is driven by interactions, with stakeholders learning about the existence of 'relevant' components and negotiating on what the end-product from widget X's should be like.

When an effectual network or technological community is being formed around providers of key components for the development of widget X, an opportunity is created

¹ The Grue paradox flows from the observation that multiple hypotheses could be supported by any set of empirical data. Goodman illustrated this paradox with the sample hypothesis that 'All emeralds are green.' A physical examination of a sample of emeralds will, of course, support this hypothesis. However, consider an alternative hypothesis that 'All emeralds are grue' where grue is defined as being green before (say) the year 2200 and blue after that date. Obviously, this hypothesis cannot be disproved by examining the colour of a sample of emeralds today!

and a market emerges. With the new artefact having been set, the behaviors of suppliers, customers and related institutions are 'boxed' within this inner environment, until the next gale of creative destruction enters to shake and reshape the system(s). The vital point of new artefact formation, as Simon ([48], p. 12) puts it

is the possession of relevant skill and knowledge, and at certain key periods in the history of science and of other domains, the relevant knowledge comes from a field other than the one to which it is applied

Critical to new market creation seeded by widget X, then, is the capability of capturing, evaluating, and utilizing 'outside' knowledge components. An entrepreneurial owner of a knowledge component (widget X) needs to learn about other relevant knowledge components, in order to make decisions on what and how to combine them into a commercially successful solution. Through the interactive learning activities of entrepreneurs and their potential customers, novel combinations of technical components are developed into vehicles of customer values, and thus new paradigms for wealth generation are set [11, 42].

To commit to and negotiate a green or blue widget X, the two parties (customers and entrepreneurs) have to sense the existence of each other, learn about the widget X's that each one carries, recognise one another as relevant or not, and commit to negotiations (interactive learning). Given the complex and uncertain environments that entrepreneurs are required to navigate through, they need to possess, at a minimum, essential individual characteristics that deal with and benefit from information asymmetries: prior knowledge endowments, the level of alertness to distant knowledge, and some tendency for exploration in face of uncertainties.

Prior Knowledge

Prior knowledge enables connections to unfamiliar domains and hence influences the generation and nature of the business ideas [43, 46]. Shane [43] suggests that prior knowledge about

1. the potential market(s),
2. the way to serve the market, and
3. customer problems, enables entrepreneurial alertness.

Shane and Khurana [45] hypothesized that prior knowledge accumulated through careers of entrepreneurs are important not only for forming social ties [18], but also as a means of learning. It provides a framework that can be used to process information [13, 21]. Cohen and Levinthal [9] emphasize that learning is self-reinforcing by nature, and thus the ability to absorb new knowledge is a function of the breadth of current knowledge stock. The broader the prior knowledge stored, the easier it is for a learner to evaluate and acquire new 'relevant' knowledge components. Newly acquired components may not be well utilized for a while, until the appropriate contextual knowledge is obtained. Simon [48] suggests that the possession of relevant knowledge 'chunks' is the precondition for learning, innovating, and problem-solving. These chunks give rise to insights or intuitions necessary for the evaluation and further application of new knowledge. Therefore we propose

H1. Prior knowledge equips technology entrepreneurs to capture new knowledge components and thus positively influences the emergence of a new market.

Alertness Coefficient

When seeking relative information to form solutions, alert learners are ‘quick’ in recognizing relevant knowledge components, and are quick in transforming and applying them. Within this context, scholars have postulated that the level of entrepreneurial activity within an organization is a function of available information and entrepreneurial alertness [12, 33, 45]. Ray and Cardozo [36] see alertness as a state of awareness or a propensity to notice and to be sensitive to information about objects, incidents, and patterns of behavior in the environment, with special sensitivity to problems, unmet needs and interests, and novel combinations of resources. Minniti [33] conceptualizes alertness as a parameter that controls how well different learners can take advantage of information asymmetry. In other words, alertness is the extent to which a learning entity makes use of its current knowledge endowment to acquire new knowledge. Equipped with the same prior knowledge, different learners may have different levels of alertness which leads to different learning performances. The different alertness coefficients of agents in a market underscores the proposition that all cannot be explained by prior knowledge endowments. Entrepreneurs not only need to possess the basic endowments of their prior knowledge but also need to take advantage of the information asymmetries [33]. The alertness coefficient thus may be an important predictor of the market creation performance of technology entrepreneurs

H2. Higher level of alertness leads to better market creation performance.

While prior knowledge and alertness combine to create alert learners, these learners need to be bold enough to commit to the learning journeys under uncertainties and this is considered in our next hypothesis.

Exploration vs. Exploitation

Market creation and development require aspiration for explorative learning. An organization or individual person with a tendency towards exploration searches for new ideas and conducts experimentation to deliver novelty, while exploitative learners focus on tweaking existing knowledge [30]. Of course, bold exploration in a sea of uncertainty may not produce profit, with March [31] noting that often, bold learners’ explorations are driven by ‘the heroism of fools and the blindness of true believers’. Similarly, [27] observe that the acquisition and processing of distant knowledge components ‘takes place in a relatively costly process of search, frequently conducted under conditions of ambiguity.’ ([27], p. 48). So, to achieve organisational changes, convergence and reorientation, exploration is necessary and through the process of experimentation the organisation recognises new goals or means to achieve goals, finds new ways of assembling responses or connecting stimuli to responses, and integrates ‘new constructs into existing cognitive structures.’ ([27], p. 49). In contrast, exploitation, or first-order learning is ‘a routine, incremental conservative process that serves to maintain stable relations and sustain existing rules’. We examine the significance of *Exploration Tendency* (ET) for the process of market creation.

H3. ET motivates commitments to new knowledge acquisition under uncertainty and positively influences market creation performance.

Hence, the above hypotheses are examining whether learning activities are initiated by alertness, informed by prior knowledge and motivated by ET. The hypotheses are tested using our simulation model. The following section explains the simulation model in detail.

11.3 Methodology

To simulate the behavioral processes leading to changes in complex systems, an ABM simulator [19] generates agents, endows them with various traits including specifying the simple rules their behaviours follow, and observes them interacting with one another. The behaviour of the system arises out of the interactions among these individual agents. For example, a modeller can specify the rules of behaviours of thousands of individual ants and then observe the resulted colony-level patterns. The emergence of any new colony-level structure (system level emergence) is not designed or programmed, but can be observed [37]. On observing the emergence of a new system structure, the researcher may check who among the individual agents initiated and/or benefited from the structural changes, and even trace the individuals' journeys to investigate what features and/or contingencies have led to such 'successes'. At this stage, statistical analysis can also be used to study whether there are factors significantly influencing the individual performance and system behavior. In this sense, an ABM simulator may also be seen as a special data generator for longitudinal case survey. When a theoretical focus is longitudinal, nonlinear, and processual (as is technological disruption, or the emergence of a new CPV), simulation modelling provides a robust method for theory development [10]. Particularly for studies on multiple interdependent processes operating simultaneously [19] collecting large scale empirical data may be impossible. In such cases, simulation may be a rigorous alternative to generate data for theory development through statistical analysis.

In this study, a simulation methodology is used for data generation to test the propositions that predict a pattern of market emergence. A set of agents representing individuals or organisations from both the demand and supply sides of the market are generated. Their activities are governed by simple rules of learning and differences among their (individual) features are governed by a probability distribution [27, 32].

11.3.1 Simulator Construction

In this model, new market(s) will emerge endogenously as a combination of the knowledge components initially owned (but not initially shared) by agents from the demand and supply sides. As knowledge is shared, a new CPV (or body of shared knowledge), is built and agreed concerning individuals' needs and how these needs can be satisfied. A CPV as shared knowledge emerges in the 50 dimensional space when a new effectuation network is being formed around a technology-based entrepreneur and potential customers committed to developing their individual 'widget X's'. Therefore, at the

system level, we expect to observe agents from different subpopulations converging together on their shared knowledge. The market creation performance of an entrepreneur is measured by counting the customers who are committed to further developing her widget X. Specific details of the simulator construction are reported below.

The Space for System-Level Emergence

Customers require a solution to 'get a job done' and a valuable solution has a set of specifications. It is believed that for any given job (e.g., transportation, cutting wood, or growing corn), customers collectively apply 50 to 150 metrics to measure the performance of a technical solution [55]. We model the 'knowledge space' for emergent customer-values as having 50 dimensions: supply-side technology possibilities and the demand side concerns on which customer values are to be perceived. In this 50D space of knowledge, an agent's coordinates represent its 'knowledge-profile', represented as a 50 dimensional vector bearing the agent's level of knowledge on each individual dimension. Individual agents initially have knowledge on their widget X's one core dimension but possess little information on the remaining dimensions. However, they will typically build up knowledge about other dimensions during the effectuation journey. In this sense, each of the agents is in an open and evolving world and this reflects the real-world impact of information asymmetries, or bounded rationality.

Agents carrying knowledge-components move around in the knowledge space, and can potentially sense the existence of other agents, decide whether other agent's knowledge is relevant and learn from/about each other. Knowledge about a new dimension, once captured, is taken into an agent's updated knowledge-profile (it's 50D vector). Dispersed knowledge components from various agents can therefore be integrated to finally create a market [20].

Among the 50 dimensions of possible future customer values, some are connected with others. So knowledge on one dimension may lead to the recognition of the existence of other dimensions. For example, the weight of a laptop computer, its memory capacity and its computation speed are all interconnected features and hence customers' perceived value about these features comes as a compromise. The interconnections among the dimensions actually make this 50D space a twisted torus, somewhat similar to an N-K landscape [5, 22, 28]. In other words, adding a new dimension to a possible solution may activate or change the agent's knowledge about another K dimensions. If customers, for example, start to believe that a mobile-phone handset should have a camera function as an improvement, they then expect new applications and specifications such as a large data storage space and on-line picture-sharing.

Scales on the Dimensions

Goldstein and Gigerenzer [15] suggest that there are three levels of knowledge. First, one may have no knowledge of an issue at all, so the existence of such a dimension has been ignored or unrecognized. At the second level, knowledge about some dimensions/issues is merely recognized based on prior knowledge. At the third level of knowledge, one can provide all sorts of additional information about an issue - a dimension on which one has deep expertise. An example of the latter in this simulation is the information that each agent initially possesses on their knowledge core (their widget X).

For this study, we use 0, 1, and 2 to denote respectively the three levels of knowledge (ignorance, some prior knowledge, and the knowledge core). At any point in time, the

location of an agent in the 50D space reveals its knowledge profile. At the starting point of learning, agents carry partial and dispersed knowledge components of some (potentially) enabling technologies (supply side) or demand-side customer problems/concerns (demand-side). Each agent's knowledge is limited to a few dimensions known to it. For example, an individual technology-based entrepreneur (denoted as a 'techie' in the rest of this chapter) ('techie 31' from the simulator) has a knowledge profile as (2 2 0 0 2 0 1 0 0 0 1 0 0 0 0 1 1 0 0 0 0 0 1 0). This means that this technology-based entrepreneur has expertise on dimensions 1, 2, and 5 (e.g., cooking expertise the cook possesses as her widget X [40]); prior knowledge about some potential market domains on dimensions 7, 11, 16, 17, and 23 (e.g., knowledge about a grocery store owned by a friend with whom this cook might start a deli business; or, about a popular media for whom she might produce cooking videos (Sarasvathy and Dew 2005)). In the meantime, other dimensions are unknown to this techie; in other words, her point of view is one from a little corner of the 50D space.

Taking an example from the demand side, one of the 200 customers, Customer 7, is initially located at (0 0 0 0 0 1 0 0 1 0 0 1 0 1 0 0 0 0 0 2 0 0 0 1 0 2 0 0 2 0 0 2 2 0 0 0 0 1 0 0 0 0 0 0 0 1 2 0 2 0 0) sharing little knowledge with techie 31.

An incumbent organization (denoted as *incum* in the remainder of this chapter) (for example, organisation 2) at (0 0 0 0 0 1 0 0 0 0 0 1 1 2 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 2 0 2 1 2 0 2 2 0) is an established supplier in an extant market, and it has more dimensions of expertise, some of which respond to customers' existing needs. Although initial conditions are critical to agents, none of the agents can predict *a priori* what/how the future is to be, which dimensions from the unknown may occur to them and to be agreed on. Yet the future is reachable through transformation and effectuation.

Boundary-Spanning Organizations

To make it possible for agents to learn distant knowledge, we randomly planted in the 50D space 50 boundary-spanning organizations (denoted as *Spanners*). For new technologies, boundary-spanning organizations could be 'research labs, patent agencies, regulatory bodies, professional societies, trade associations, consortia, and other types, depending on technological and political contexts' ([38], p. 411). Rosenkopf and Tushman [38, 39] believe that boundary-spanning actors (which are composed of representatives from multiple organizations) create cognitive linkages across organizations in different technological communities. Dosi [11] (p. 229) suggested that these bridging institutions may have a key influence on the early stage of innovations. From the supply-side point of view, [50] suggest that firms, in order to overcome the tyranny of served markets, build cognitive ties broadly with suppliers, businesses in different industries, consultants, universities, and government agencies. In this simulator, Spanners are set to have various levels of knowledge randomly, for example, one (Spanner 52) stands at the point (0 0 0 0 0 2 1 1 2 1 0 2 2 2 2 0 1 1 1 2 2 1 0 1 1 1 0 0 2 0 0 0 1 0 0 0 0 0 0 0 0 0 2 0 2 1 2 1 0). All the spanners are randomly planted as such in the knowledge space.

Behaviours of the Learning Agents

In the 50D knowledge space, an agent's location denotes its current knowledge-profile and its movements are learning activities as time goes by. At each time-step (tick) of

the algorithm, learners are displaced from their current positions by applying a velocity vector to them [23]

$$X_i(t+1) = X_i(t) + V_i(t) \quad (11.1)$$

The magnitude and direction of an agent's velocity at each step are determined by simple rules: whom a learner decides to move toward (learn from), and how large this step can be. The simple rules for each population of agents are as follows:

Techies: May move towards the closest spanner and/or a tentatively identified lead user (a customer close enough in the knowledge space to it, thanks to their prior knowledge on the same dimensions, or indirectly, through some bridging spanners);

Incums: If they are sufficiently 'explorative', they may learn from distant customers (customers whose needs have not been identified/served). If they are not explorative, they stay with committed customers to elaborate on current solutions based on shared knowledge;

Customers: Learn about a supplier, a techie or an incum, having sensed its existence; and/or having learnt from a close neighbor, if the neighbor is happier (having more dimensions of its needs served).

In a natural ecosystem, predators have to make foraging decisions with little, if any, knowledge of present resource distribution and availability. The likelihood of a learner to sense distant knowledge elements is similar to the encounter rates with prey in heterogeneous natural environments [47]. We model this likelihood as a decreasing exponential function of the distance [56]

$$C = Ae^{-bD} \quad (11.2)$$

where D is the Euclidean distance in the knowledge space between the locations of an agent and the source of the knowledge element to be recognized. The alertness coefficient, b , represents the extent to which such a distance obstructs the learning activity - in other words, the extent to which an agent can take advantage of information asymmetry. For an alert learner, $b < 1$. Since the 'intelligence was guided by will towards the solution of envisaged problems' ([40], p. 535), A is ET, the exploration tendency, or the 'will' of committing to new learning under uncertainty. For the implementation of ET concept, we adopt a 5 point scale, with 1 being the lowest in ET and 5 the highest. Combining Eq. (1) and (2), the learning behaviours of the three types of agents are expressed by the following equations

Techies:

$$\begin{aligned} V_i(t+1) &= A_i e^{-b_i D_i} (\text{leaduser}(t) - X_i(t)) + A_i e^{-b_i D_i^*} (P_{\text{closest}}(t) - X_i(t)) \\ D_i &= \text{EuclideanD}(\text{leaduser}(t), X_i(t)) \\ D_i^* &= \text{EuclideanD}(\text{Spanner}(\text{closest}), X_i(t)) \end{aligned} \quad (11.3)$$

Incums:

$$V_i(t+1) = (1 - A_i e^{-b_i D_i}) (\text{customer}(t) - X_i(t)) + A_i e^{-b_i D_i} (\text{leaduser}(t) - X_i(t)) \quad (11.4)$$

Customers:

$$V_i(t+1) = \alpha(lbest(t) - X_i(t)) + A_i e^{-b_i D_i} (heard(t) - X_i(t)) \quad (11.5)$$

where α is random number drawn from $U(0, 1)$ representing a customer's exploitation tendency (its tendency to learning only from close neighbours in the 50D space); *lbest* is a neighbour customer who is recognized as being happier (having more dimensions of need served); and *heard* is either a techie or an incum, whose existence has been recognized by this customer.

If an agent has '0' level of knowledge on a dimension before making a step of movement, it may after learning from others, flip the knowledge to '1'. This is governed by

$$Sigmoid(V) = \frac{1}{1 + e^{-|V|}} \quad (11.6)$$

When the sigmoid function of the velocity on a dimension is larger than a random number $U(0, 1)$, knowledge on that dimension flips from '0' to '1' [4]. After recognizing a dimension as relevant, the knowledge-gain along the dimension is cumulative from '1' to '2'. An agent can 'unlearn' (or forget) about a dimension by unloading its knowledge from '2' continuously down to '1', but not from '1' back to '0'. After recognizing the existence of a dimension, one cannot be ignorant of its existence any more. Taking into account the interconnectedness of dimensions, if a learner's knowledge level on one dimension is higher than 1, there is a chance for this learner to recognise the existence of other K dimensions.

If customers and supplier(s) (either incum or techie) have built sufficient shared knowledge to come close to each other, a new effectuation network emerges in the 50D space. Customers whose needs are satisfied by an incum will paint their shared patches green whereas those who are happy with a techie's widget X (and are willing to commit further to its development) paint their patches blue. Normally, as incums have initially more shared dimensions of knowledge with customers, some green patches emerge very early in the simulation. These become extant markets in the disruptive techies' eyes. Still, we are unable to predict the colour of future patches (therefore all markets are 'grue').² From running the simulation model, the emergence of green or blue patches can be observed. Simulation experiments are conducted to collect data for testing the hypotheses on individual learning behaviours of the agents and their market creation performance.

11.3.2 Model Validation

To simulate means to build a likeness. The validation issue of a simulation model addresses the question of how accurate that *likeness* is [24]. Although there are arguably diverse approaches through which a researcher can validate a simulation model [24], empirical validation- comparing generated data with longitudinal case studies is the most direct approach [10]. Alternatively, staying in accordance with 'expert opinion and professional acceptance can be as good validation' ([24], pp. 1089-1090). As we have to leave empirical validation for future studies, the current model construction

² The NetLogo code is available from the authors on request. Please contact Shuyuan Wu.

complies with the widely accepted principles for building agent-based models. Each agent has strategic choices in making its movements in the knowledge space for each time step. Specifically, to test whether the model renders a reasonably wide array of behaviors, we draw a sample of 104 technology-based learners randomly by running the simulator in various system conditions such as different levels of prior knowledge, different involvement with boundary spanning organizations, and different levels of explorative behaviours of the incumbent organizations.

11.4 Simulation Results

This section presents the simulation results. We initially describe the results of this simulator by demonstrating the system level emergence and secondly, we examine the behaviours and performance of individual agents.

System-Level Emergence: Markets are Grue

Fig. 11.1 shows a simulator snapshot of the development of green and blue patches within the system. It was captured after 300 ticks (iterations) of one run of the simulator. In this figure, green patches are technological communities around green solutions developed from the widget X's of incums. On the blue patches are effectuation networks created by techies together with their committed customers. White patches represent mature markets developed from green or blue patches). Yellow figures represent customers, red circles with dots are spanners, blue happy faces are techies, and grey pillars are incums. The picture shows the emergence of blue and green patches, as the results of the commitments of agents in their learning journeys.

Fig. 11.1 demonstrates the existence of blue or green patches that have emerged from the learning activities of the agents. From this figure, we can observe the results of the movements and commitments of the agents in creating a new markets for/from their 'widget X's'. Within this landscape, a variety of markets are created. We find that

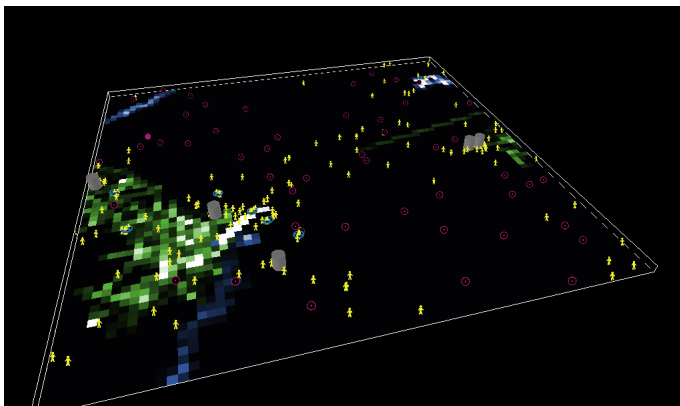


Fig. 11.1. Markets are Grue

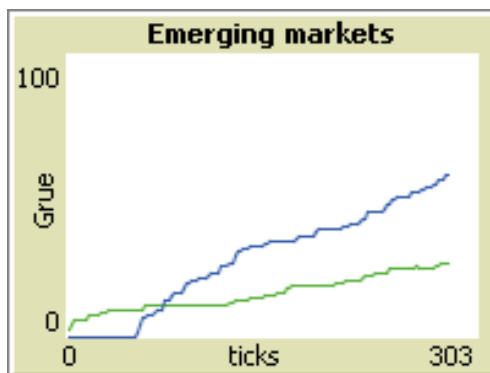


Fig. 11.2. The emergence of blue patches. The blue line crosses the green line after approx. 79 iterations and remains above the green line thereafter. The x-axis (ticks) are iterations of the simulation, whereas the y-axis denotes the number of green and blue patches respectively.

there are established markets (green patches which are set earlier by incums and their customers during the running of simulation, new markets (blue patches showing where previously unsatisfied customers have interacted with techies to create new markets for their new-to-the-world widget X's), and well developed markets (when the blue or green solutions have been well refined and accepted and those markets are maturing).

Fig. 11.2 shows dynamically the technological disruption. We can observe that green patches appear at iteration 5 (very early on in the running of simulation). Sharing more dimensions of knowledge with customers at the starting point, incums are able to interact and negotiate with customers to establish solutions to their needs quite quickly. However the green line up to 303 iterations shows that the number of green patches and mature markets from them (markets for the former technology) tend to stay fairly constant over the running of the simulation, indicating that extant markets of established solutions continually attract, retain, and sometimes lose customers. This can be contrasted with the emerging blue patches, which began to become noticeable at iteration 62. As can be seen from Fig. 11.2, the number of blue patches increases steadily after iteration 62. This indicates that more and more customers are committed to develop new solutions together with techies as their shared knowledge expands. At iteration 79, the number of blue patches overtakes the number of green patches indicating that the new markets have begun to overtake existing markets. This indicates the success of disruptive technologies [1, 6, 7]. In creating its markets, a new disruptive technology at its inception is inferior to mainstream solutions along the recognized dimensions of performance. Therefore at that stage their early development only serves niche segments which value their non-standard performance attributes, however, subsequently along their development, these technologies are able to raise the performance attributes such that they begin to involve more and more customers. Fig. 2 graphically displays the competition between green and blue solutions being developed, the result of which was the emergence and dominance of blue widgets (or blue markets being created). At iteration 303 (the end of the simulation experiment), customers with techies have created 82

blue patches, while there were 39 green patches in comparison (i.e. the new technology is dominating).

The system-level emergence demonstrates a number of issues of significance for theory advancement in market creation. Firstly, the results of the system-level emergence in the simulation provide evidence that markets are indeed *grue*. Technology entrepreneurs need not know the future in order for new markets to emerge. Rather, the collective learning and interactions of entrepreneurial entities, customers and others give rise to opportunities for market creation. The various agents within a system work through commitments to exchange and combine their knowledge components without a complete knowledge about the future. These results demonstrate the importance of learning and transformation (the accumulation and sharing of knowledge resources) that entrepreneurs and customers are required to commit to in order to create new CPV.

Our second theoretical insight is the emergence and dominance of disruptive innovations. While preliminary at this stage, the results of our simulation suggest that the success of disruptive innovations is due to the combination of knowledge, and interactive learning activities with customers [6, 14, 17]. However, in order to examine the complex and adaptive behaviours of this system level emergence, we need to examine the micro-level interactions of the entrepreneurial agents.

What Efforts Can a Techie Make?

Individual agent's movements are recorded automatically during the running of the simulator. With this data we can trace the learning journey of individual agents and also test the influence of each individual's traits on their performance. Longitudinal cases can be drawn from the datasets as abstract versions of market creation journeys. Each can be compared with market-creation case examples of technology-based entrepreneurs that began with new-to-the-world 'widget Xs' (for example, molecules such as Kevlar [51] and Surlyn [34] invented by Du Pont).

In general, the learning journey for individual techies is uncertain. They go through a stochastic process of expanding resources (the techie's knowledge, in this model) and converging on constraints constructed together by the supply and demand sides [40]. Most of the techies end up having no customers staying with them, even if they attracted customers at times during the simulation run. This parallels the high failure rate for real-world product development. Some succeed in having more and more customers committed to their widget X because of their strong wills to make use of what they know and push the boundary of the unknown. They 'move' actively, even after being frustrated during earlier time steps and they are alert to identify distant knowledge elements so that they expand knowledge resources to realise a shared body of CPV together with customers.

The journey to success is not smooth. It is difficult to sense the existences of potential customers and attract their attentions to new widget X': on average it takes more than 45 iterations to observe blue patches showing up. It seems even harder to keep customers' commitment because there are multiple competing widget X's being developed at the same time. For example, we traced an alert and extremely bold (exploratory) techie, 'techie 4' with $b = 0.5$ and $A = 5$. Being narrowly specialised in its expertise and having only 3 dimensions of prior knowledge, it had no potential customer within mind-sight range. It learned about a few dimensions from Spanner 225, then Spanner 241, and

hence after 3 iterations saw Customer 70. However, Customer 70, committed to a green widget X with Incum 10, did not pay any attention to Techie 4. Techie 4 continued expanding its knowledge, learning from Spanner 441 and identified Customer 137 as a lead-user at iteration 9, but this customer decided not to go together with it either. At iteration 15, Customer 70 who had been involved with Incum 10 and then Techie 2 updated its knowledge-profile (on what is important to satisfy her needs) and the update took it closer to Techie 4. However, the first 'transaction' between these two did not happen till iteration 38, when Customer 70 felt happy to paint its patch blue, after a negotiation lasting 23 iterations. As a result of these, customer 70's need-profile was updated to [1 1.23 1.48 1 1 1 2 1 2 1 1 1 1 1 1 1 1 1 1 0 0 0 1 1 1 1 1 1 0 0 1 1 1 0 0 0 0 0 1 1 0 1 1 1 1 1 1 0 1 1 1] and the knowledge of techie 4 arrived at [1 2 2 2 0 1 1.4 1 1.18 1 1 1 1 1 1 0 0 0 1 0 0 0 1 0 0 0 0 0 0 0 0 1 0 1 1 1 1 1 1 0 0 1 0 1 1].

Until iteration 63, techie 4 had only been trading with customer 70, without expanding its knowledge resources at all and unfortunately Customer 70 left it during iteration 63. Afterwards, techie 4 learned from Spanner 241, Spanner 215, and Customer 102 to recognize dimension 17 and 22 so that it involved Customer 102 and recaptured Customer 70. By iteration 66 customer number has increased to 5 after Techie 4 took into its knowledge-profile dimension 5 and dimension 34.

The market kept expanding as Techie 4 learnt more about the needs of lead users, Customer 70, 102, and 137, and meanwhile potential customers were learning from each other. By iteration 96 it had involved 10 committed customers and the (sub)market size increased to 35 customers at iteration 121. Within 10 iterations the number of customers had increased to 66. The count of committed customers as the market creation performance of techie 4 reached 95 (47.5% of the customer population) at iteration 134 when we ended our observation on it.

In order to examine the importance of the three parameters in our model, we collected data from the simulator by running it under a variety of conditions. This dataset includes 104 techies and their market creation performances (counts of customers who are committed together with individual techies to develop future solutions). Multiple regression analysis tested the relationship between the number of customers involved (Customers) with a focal techie as its performance and the techie's exploration tendency (ET), alertness coefficient (Alertness), and the breadth of prior knowledge (PriorKw). The results show that ET, alertness, PriorKw together explained 20.7% (the R^2) of the variance in market-creation performances. The ANOVA analysis confirms the significance of this model, with p-value smaller than 1%. This means that the success of market creation was not totally by chance, but can be attributed to the individual traits of each techie agent, even though future is not predictable from initial conditions. PriorKw was positively correlated with the performance, with the coefficient being 0.56 ($p < 0.00$). This is consistent with hypothesis 1 that prior knowledge plays a highly significant part in the creation and development of CPV. Support for hypothesis 2 is also found as the alertness coefficient, b , was found to be negatively correlated (due to the implementation of the alertness concept in the simulator, the higher b is, the less alert an agent is) with techies' market creation performances. The effect is strongly significant ($B = -1.542$, $p < 0.001$). ET has a correlation of 0.5 at the significance level of 95% ($p < 0.05$) with the market creation performance, suggesting support for hypothesis 3. The result from

our simulations supports an assertion that information asymmetries impact on the market creation process of technology entrepreneurs [43]. The ability to create markets is a function of the interaction of prior knowledge and alertness. Also ET, the aspiration to create future through commitments under uncertainties, is significantly important for the success.

The emergence of CPV is a function of the system in which individual agents interact and expand their knowledge. Entrepreneurs and customers act within their worlds of bounded cognition, partial knowledge and uncertainty [40]. During the effectuation process, stakeholders come together to commit to transforming extant realities into a new market. Therefore to a great extent, the creation of an effectuation network (and the eventual development of a market) is largely dependent on the interactions of who and what components have been seen as 'relevant' and hence has come on board during the process. In short, chance plays a large role.

11.5 Conclusions

This study shows that the interactive behavioral processes of market creation can be realized through computer simulation so that researchers can analyze both system-level behavior and the influence of individual factors on performance.

Taking the CAS approach, we simulate the market creation process for disruptive technologies. New markets emerge from the interactions between entrepreneurs and their potential customers. Starting with limited and dispersed knowledge components, these individual agents converge at artefacts of shared knowledge (CPV) on what is needed and how that need is to be satisfied. In expanding their knowledge profiles and negotiating their constraints, individual learners sense each other's existence, recognise relevant components, and learn about/from each other through commitments. The result showed that individual entrepreneurial traits including prior knowledge endowments, alertness, and exploration tendency are significantly influential in the market creation performance. However, because the effectuation process of individual agents is highly stochastic and complex, individuals' traits together explained only 21 percent of the variance of market creation performance.

It is not possible in a single set of simulation experiments to exhaustively examine every possible combination of settings for each parameter in the simulation model. Future work will extend the range of settings examined and will include further development of the simulator.

References

- [1] Adner, R.: When are technologies disruptive? A demand-based view of the emergence of competition. *Strategic Management Journal* 23, 667–688 (2002)
- [2] Ali, A.: Pioneering versus incremental innovation: review and research propositions. *Journal of Product Innovation Management* 11, 46–61 (1994)
- [3] Arthur, W.: The structure of invention. *Research Policy* 36, 274–287 (2007)
- [4] Brabazon, A., O'Neill, M.: *Biologically-inspired algorithms for financial modelling*. Springer, Heidelberg (2006)

- [5] Brabazon, A., Silva, A., Ferra De Sousa, T., O'Neill, M., Matthews, R., Costa, E.: Organizational strategic adaptation in the presence of inertia. *Advances in Complex Systems* 8, 497–519 (2005)
- [6] Christensen, C.: *The innovator's dilemma - when new technologies cause great firms to fail*. Harvard Business School Press (1997)
- [7] Christensen, C., Johnson, M., Rigby, D.: Foundations for growth: How to identify and build disruptive new businesses. *MIT Sloan Management Review* 43, 22–31 (2002)
- [8] Cohen, M., March, J., Olsen, J.: A garbage can model of organizational choice. *Administrative Science Quarterly* 17, 1–25 (1972)
- [9] Cohen, W., Levinthal, D.: Absorptive capacity: A new perspective on learning and innovation. *Administrative Science Quarterly* 35, 128–152 (1990)
- [10] Davis, J., Eisenhardt, K., Bingham, C.: Developing theory through simulation methods. *Academy of Management Review* 32, 480–499 (2007)
- [11] Dosi, G.: Technological paradigms and technological trajectories: A suggested interpretation of the determinants and directions of technical change. *Research Policy* 11, 147–162 (1982)
- [12] Eckhardt, J., Shane, S.: Opportunities and entrepreneurship. *Journal of Management* 29, 333–349 (2003)
- [13] Fiske, S., Taylor, S.: *Social cognition*. Addison-Wesley, Reading (1984)
- [14] Gatignon, H., Robertson, T.: Technology diffusion: An empirical test of competitive effects. *Journal of Marketing* 53, 35–49 (1989)
- [15] Goldstein, D., Gigerenzer, G.: Models of ecological rationality: The recognition heuristic. *Psychological Review* 109, 75–90 (2002)
- [16] Goodman, G.: *Ways of worldmaking*. Hackett (1978)
- [17] Govindarajan, V., Kopalle, P.: The usefulness of measuring disruptiveness of innovations ex post in making ex ante predictions. *Journal of product innovation management* 23, 12–18 (2006)
- [18] Granovetter, M.: *Getting a job: A study of contracts and careers*. Harvard University Press (1974)
- [19] Harrison, J., Lin, Z., Carroll, G., Carley, K.: Simulation modeling in organizational and management research. *Academy of Management Review* 32, 1245–1299 (2007)
- [20] Hayek, F.: The use of knowledge in society. *American Economic Review* 35, 519–530 (1945)
- [21] Hillerbrand, E.: Cognitive differences between experts and novices: Implications for group supervision. *Journal of Counselling and Development* 67, 293–296 (1989)
- [22] Kauffman, S., Levin, S.: Towards a General Theory of Adaptive Walks on Rugged Landscapes. *Journal of Theoretical Biology* 128(1), 11–45 (1987)
- [23] Kennedy, J., Eberhart, R., Shi, Y.: *Swarm Intelligence*. Morgan Kaufman, San Mateo (2001)
- [24] Kleindorfer, G., O'Neill, L., Ganeshan, R.: Validation in Simulation: Various Positions in the Philosophy of Science. *Management Science* 44(8), 1087–1099 (1998)
- [25] Kotler, P., Armstrong, G., Saunders, J., Wong, V.: *Principles of Marketing*. Pearson Education, London (2001)
- [26] Lant, T., Mezias, S.: Managing discontinuous change: A simulation study of organizational learning and entrepreneurship. *Strategic Management Journal* (1986-1998) 11, 147–179 (1990)
- [27] Lant, T., Mezias, S.: An organizational learning model of convergence and reorientation. *Organization Science* 3, 47–71 (1992)
- [28] Levinthal, D., Warglien, M.: Landscape Design: Designing for Local Action in Complex Worlds. *Organization Science* 10(3), 342–357 (1999)

- [29] Lynn, G., Morone, J., Paulson, A.: Marketing and Discontinuous Innovation: The Probe and Learn Process. *California Management Review* 38, 8–37 (1996)
- [30] March, J.: Exploration and Exploitation in Organisational Learning. *Organization Science* 2(1), 71–87 (1991)
- [31] March, J.: Rationality, foolishness, and adaptive intelligence. *Strategic Management Journal* 27, 201–214 (2006)
- [32] McKelvey, B.: Towards a complexity science of entrepreneurship. *Journal of Business Venturing* 19, 313–341 (2004)
- [33] Minniti, M.: Entrepreneurial alertness and asymmetric information in a spin-glass model. *Journal of Business Venturing* 19, 637–658 (2004)
- [34] Norling, P., Statz, R.: How discontinuous innovation really happens. *Research Technology Management* 41, 41–44 (1998)
- [35] Penrose, E.: *The Theory of the Growth of the Firm*. John Wiley, New York (1959)
- [36] Ray, S., Cardozo, R.: Sensitivity and creativity in entrepreneurial opportunity recognition: A framework for empirical investigation. Sixth Global Entrepreneurship Research Conference, Imperial College, London (1996)
- [37] Resnick, M.: *Turtles, Termites, and Traffic Jams*. MIT Press, Cambridge (2000)
- [38] Rosenkopf, L., Tushman, M.: The co-evolution of technology and organizations. In: Baum, J., Singh, J. (eds.) *Evolutionary Dynamics of Organizations*. Oxford University Press, Oxford (1994)
- [39] Rosenkopf, L., Tushman, M.: The coevolution of community networks and technology: Lessons from the flight simulation industry. *Industrial and Corporate Change* 7, 311–346 (1998)
- [40] Sarasvathy, S., Dew, N.: New market creation through transformation. *Journal of Evolutionary Economics* 15, 533–565 (2005)
- [41] Sarasvathy, S., Venkataraman, S., Dew, N., Velamuri, R.: Three views of entrepreneurial opportunity. In: Acs, Z., Audretsch, D. (eds.) *Handbook of Entrepreneurship Research: An interdisciplinary survey and introduction*. Kluwer Academic Publishers, Dordrecht (2003)
- [42] Schumpeter, J.: *The Theory of Economic Development: An inquiry into profits, capital and the business cycle*. Harvard Economic Studies, vol. XLVI. Harvard University Press (1934)
- [43] Shane, S.: Prior knowledge and the discovery of entrepreneurial opportunities. *Organization Science* 11, 448–469 (2000)
- [44] Shane, S., Khurana, R.: Bringing individuals back in: The effects of career experience on new firm founding. *Industrial and Corporate Change* 12, 519–543 (2003)
- [45] Shane, S., Venkataraman, S.: The promise of entrepreneurship as a field of research. *Academy of Management Review* 25, 217–226 (2000)
- [46] Shepherd, D., Detienne, D.: Prior knowledge, potential financial reward, and opportunity identification. *Entrepreneurship Theory and Practice* 29, 91–112 (2005)
- [47] Silva, A., Neves, A., Costa, E.: An empirical comparison of particle swarm and predator prey optimization. *Lecture Notes on Artificial Intelligence*, pp. 103–110. Springer, Heidelberg (2002)
- [48] Simon, H.: What we know about the creative process. *Frontiers in Creative and Innovative Management* 4, 3–22 (1985)
- [49] Simon, H.: *Sciences of the Artificial*, 3rd edn. MIT Press, Cambridge (1996)
- [50] Slater, S., Narver, J.: Market orientation and the learning organization. *Journal of Marketing* 59, 63–74 (1995)
- [51] Smith, L.: A miracle in search of a market. *Fortune* 102, 92–94 (1980)
- [52] Stacey, R.: The science of complexity: An alternative perspective for strategic change processes. *Strategic Management Journal* (1986–1998) 16, 477–495 (1995)

- [53] Stacey, R.: Strategic management and organizational dynamics. Pitman Publishing (1993)
- [54] Tushman, M., Romanelli, E.: Organizational evolution: A metamorphosis model of convergence and reorientation. In: Cummings, L., Staw, B. (eds.) *Research in Organizational Behavior*. JAI Press (1985)
- [55] Ulwick, A.: *What customers want: Using outcome-driven innovation to create breakthrough products and services*. McGraw-Hill, New York (2005)
- [56] Wu, S., Brabazon, A.: A garbage can model for Schumpeterian process: The network effects. *International Journal of Foresight and Innovation Policy* 4, 287–306 (2008)

Index

- adaptive markets hypothesis 9
- agent-based artificial markets 96
- agent-based co-evolutionary algorithm 183
- agent-based co-evolutionary techniques 181, 182
- agent-based computational economics (ACE) 138
- agent-based modelling 137, 227
- agent-based models 208
- agent based model 207, 210
- American option 52
- ant colony optimisation 51, 56
- ant system 57
- arbitrage 10
- arbitrage portfolio 10
- artificial financial markets 144

- barrier option 52
- benchmarks 20
- Bermudan option 52
- best matching unit 35
- Big Lotto 209
- binomial lattice option pricing 54
- biologically-inspired algorithms 182
- Black-Scholes-Merton (BSM) model 54
- bloat 157
- bloat control 37
- bond futures 75
- book losses prediction 31
- boundary-spanning organisations 233
- bounded rationality 139

- characterising a trading strategy 100
- characterising trading strategies 102, 109
- clean price 77
- co-evolution 139, 165, 181, 225
- co-evolutionary algorithm 181, 183, 193
- co-evolutionary heterogeneous artificial stock market (CHASM) 151
- co-evolutionary multi-agent system 182, 195
- complex adaptive system 225
- computational finance 51
- conditional heavy tails 141
- connection gene 80
- constant absolute risk aversion 146
- contrarian strategy 101
- coordination game theory 118
- coupon 77
- Cox-Ross-Rubinstein binomial model 54
- crossover 11, 36, 188, 215
- customer perceived value 226

- deliverable bond 77
- demographic burden 207
- disruptive technology 225
- diversity 84
- dual tree program 12
- dynamic environments 182

- EDDIE 154
- effectuation process 226
- efficient market hypothesis 9, 96
- euro-bobl bond 76

- euro-bund bond 76
- euro-schatz bond 76
- European option 52
- Euro Stoxx index 12
- evolutionary algorithm 181, 183
- excess trading profit 96, 99, 100

- fads 119
- financial bubble 117, 118
- financial distress prediction 31
- fitness sharing 84
- function set 12
- fundamental bubbles 119
- fundamental trading 152
- futures contract 76
- fuzzy set 211

- GARCH model 52
- genetic algorithm 11, 213
- genetic operators 36, 215
- genetic programming 9, 11, 31, 36, 139, 186
- genetic programming based agents 144
- genotype-phenotype map 101
- Gini coefficient 208
- global game 121
- government bond futures 76
- grammar 14
- grue paradox 228

- heavy tails 141
- herd behaviour 120
- heuristics 9
- Hill tail index 143

- income distribution 208
- indicators 153
- inferring trading strategy 102
- informational efficiency 139
- information bubbles 119
- informed investors 122
- Infotel database 32
- innovation number 80
- intelligent trading strategy 99
- intraday effects 12
- investment strategy 181

- jackpot 209
- Jacque-Bera test 143
- joint hypothesis test 140

- Kolmogorov-Smirnov test 217
- Kruskal-Wallis test 44

- Laffer curve 219
- lead-lag performance 22
- learning rate 35
- Lorenz curve 208
- lottery games 207
- lottery pay out 207
- lottery take out 207
- lottery tax 208
- Lotto 207, 209
- Lottomania 211

- Mann-Whitney-Wilcoxon test 217
- market creation 225
- market efficiency 139
- market equilibrium 123
- market impact 24, 102
- market order 15
- measuring profits 97
- membership function 211
- Monte Carlo simulation 125
- moving window 90
- multi-agent system 182
- multi-objective optimization 182
- mutation 11, 36, 191, 215

- neighbourhood function 35
- neuro-evolution of augmenting topologies (NEAT) 79
- noise traders 158

- objective function 14
- option 51

- paper trading 101
- pari-mutuel “lotto” 207
- performance statistics 17
- pheromone 56
- population diversity 181, 182
- price multiplicity 118
- prune and plant 38

- 5-range ratio 208
- rational expectation equilibrium model 121
- red queen 158, 166
- rollover 210
- Russian option 52

- Santa Fe artificial market 138
- selection 192
- self-organisation 34
- self-organising map 31, 34, 39
- sharing function 84
- Sharpe ratio 97
- similarity function 35
- stationary bootstrap 19
- statistical arbitrage 9, 10
- stigmergy 52, 56
- strike price 51
- strongly typed GP 38
- structural optimisation 36
- swarm intelligence 52

- technical traders 153
- terminal set 12
- theoretical future price 78

- tick size 77
- tournament selection 11
- trading cost 24
- travelling salesman problem 57
- tree structure 36

- U-Matrix 35, 39
- underlying 76
- uninformed investors 122

- validation 149
- value proposition 225
- volatility clustering 141
- volume weighted average prices 12

- zero-intelligence traders 97
- zero intelligence agents 144, 148

Author Index

- Alfaro-Cid, E. 31
- Brabazon, Anthony 1, 75, 225
- Bradley, Robert 75
- Chen, Shu-Heng 207
- Chie, Bin-Tzong 207
- Dreżewski, Rafał 181
- Esparcia-Alcázar, A.I. 31
- Fan, Hui-Fen 207
- Gordillo, José Luis 95
- Hsieh, Tsung-Han 117
- Kumar, Sameer 51
- Li, Youwei 117
- Maringer, Dietmar 9
- Martinez-Jaramillo, Serafin 137
- McKillop, Donal G. 117
- Merelo, J.J. 31
- Miranda, Enrique Martínéz 95
- Mora, A.M. 31
- O'Neill, Michael 1, 75
- Saks, Philip 9
- Sepielak, Jan 181
- Sharman, K. 31
- Siwik, Leszek 181
- Stephens, Christopher R. 95
- Thulasiram, Ruppia K. 51
- Thulasiraman, Parimala 51
- Tsang, Edward P.K. 137
- Wu, Shuyuan 225
- Yu, Tina 207