Natural Computing in Finance: A Review

Anthony Brabazon[1,2], Jing Dang[1,2], Ian Dempsey[1,4], Michael O'Neill[1,3] and David Edelman[1,2]

[1] Natural Computing Research and Applications Group,
Complex Adaptive Systems Laboratory, University College Dublin, Ireland.
[2] School of Business, University College Dublin, Ireland.
[3] School of Computer Science and Informatics, University College Dublin, Ireland.
[4] Pipeline Financial Group Inc, New York, USA.
anthony.brabazon@ucd.ie; jing.dang@ucd.ie; ian.dempsey@gmail.com;
m.oneill@ucd.ie; davide@ucd.ie

Summary. The field of Natural Computing (NC) has advanced rapidly over the past decade. One significant offshoot of this progress has been the application of NC methods in finance. This chapter provides an introduction to a wide range of financial problems to which NC methods have been usefully applied. The chapter also identifies open issues and suggests future directions for the application of NC methods in finance.

1 Introduction

Recent years have seen the application of multiple Natural Computing (NC) algorithms (defined in this chapter as computer algorithms whose design draws inspiration from phenomena in the natural world) for the purposes of financial modelling [13]. Particular features of financial markets including their dynamic and interconnected characteristics bear parallel with processes in the natural world and prima facie, this makes NC methods 'interesting' for financial modelling applications. Another feature of both natural and financial environments is the phenomenon of emergence, or the activities of multiple individual agents combining to co-evolve their own environment.

The scale of NC applications in finance is illustrated by Chen & Kuo [22] 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 underscoring the continued growth in this application area (see also [14, 15, 21, 112, 122] for additional examples of NC applications in finance).

Some of the major areas of financial applications using NC methods are: forecasting, algorithmic trading, portfolio management, risk management, derivatives modelling and market modelling. In this chapter, we describe the

utility of NC methods within each of these areas wherein their usage can be broadly categorised as optimisation, model induction and agent-based modelling.

Optimisation

A wide variety of NC methodologies including genetic algorithms, evolutionary strategies, differential evolution and particle swarm optimisation have been applied for optimisation purposes in finance. A particular advantage of these methodologies is that, if applied properly, they can cope with 'difficult' search spaces. Examples of the use of optimisation techniques in finance, include optimal asset allocation, stock selection, risk management, pricing and hedging of options, and asset liability management [126].

Model Induction

While optimisation applications of natural computing are important, the underlying model or data generating process is not known in many real-world financial applications. Hence, the task is often to 'recover' or discover an underlying model from a dataset. This is usually a difficult task as both the model structure and associated parameters must be uncovered.

Financial markets are affected by a myriad of interacting economic, political and social events. The relationship between these factors and financial asset prices is not well understood and, moreover, is not stationary over time. Most theoretical financial asset pricing models are based on strong assumptions which are often not met in real-world asset markets. This offers opportunities for the application of model induction methodologies in order to recover the underlying data generating processes. These methods can be applied, for example, to financial forecasting, credit risk assessment and derivatives pricing.

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 differing expectations to future outcomes, in contrast to traditional assumptions of investor homogeneity and rational expectations. ABM attempts to explain market behaviour, replicate documented features of real-world markets, and allows us to gain insight into the likely outcomes of different regulatory policy choices.

A growing community of researchers are engaged in the application of natural computing methodologies in finance as illustrated by the number of conferences, workshops and special sessions in this area. Examples of these

include the annual track on Evolutionary Computation in Finance and 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*.

1.1 Structure of Chapter

The rest of this chapter is organised as follows. Section 2 provides a concise overview of a number of key families of natural computing methods. Section 3 introduces various financial applications of natural computing methods and shows how these methodologies can add value in those applications. Section 4 concludes this chapter, suggesting multiple avenues of future work at the intersection of finance and natural computing.

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 that are relevant for finance applications illustrated in this chapter are Neurocomputing, Evolutionary Computing, Social Computing, Immunocomputing, Physical Computing, and Developmental & Grammatical Computing (see Fig. 1).

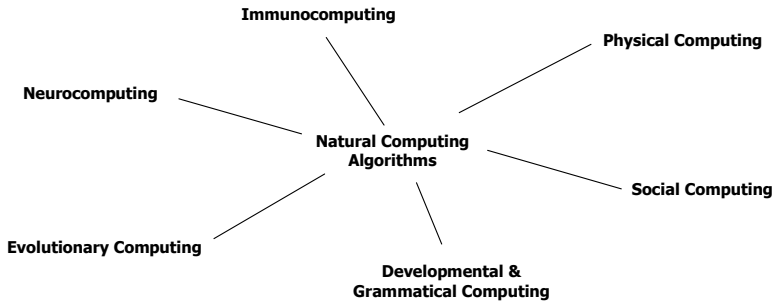


Fig. 1. An overview of Natural Computing Algorithms

Neurocomputing (or neural networks, NNs) typically draws inspiration from simplified models of the workings of the human brain or the nervous system. From a design perspective, neural networks can be characterised by a set of neurons (or nodes), the network structure which describes the pattern of connectivity between neurons, and the learning (or training) approach used. The

predominant neurocomputing paradigms include *feedforward networks*, *recurrent networks*, *self-organising networks*, *radial basis function networks*, *support vector machines* [127, 25], etc. Financial firms worldwide are employing NNs to tackle difficult tasks involving intuitive judgment or requiring the detection of data patterns which elude conventional analytic techniques. For example, NNs are already being used to trade the securities markets, to forecast the economy and to analyse credit risk. NNs were among the earliest NC methodologies to see widespread applications in finance (see, for example, Trippi et al. [110]) but they do suffer from the practical drawback that their black box nature makes their internal workings opaque to the user.

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* (GA), *evolution strategies* (ES), *evolutionary programming* (EP) and *genetic programming* (GP). These methods have broad applications for optimisation and model induction purposes. Recent extensions of the literature on EC to encompass dynamic, multi-objective and constrained optimisation problems has greatly increased the practical utility of these algorithms in finance.

Social Computing adopts a swarm metaphor and includes algorithms inspired by the flocking and schooling behaviour of birds and fish. It also includes algorithms inspired by behaviours observed in social insects such as ants. These social systems exhibit a number of characteristics facilitating self-organisation, flexibility, robustness, and direct or indirect communication among members of the population. Some examples of social computing include *ant colony*, *particle swarm* and *bacterial foraging* algorithms. These algorithms are population-based like their evolutionary computation counterparts, and they operate by allowing the population of problem-solvers to communicate their relative success in solving the problem to each other, thereby biasing the future actions of the individuals in the population.

Immunocomputing encompasses a family of algorithms which turn to the complex and adaptive biological immune system of vertebrates to inspire their design. 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 in the body. These properties are especially useful for tasks such as classification and optimisation. Practical applications of immunocomputing include financial pattern-recognition such as the identification of potentially fraudulent credit card transactions, the identification of financially at-risk companies and the identification of market 'state'.

Physical Computing draws inspiration from the physical processes of the natural world to design computational algorithms. These algorithms draw inspiration from phenomena such as simulated annealing and quantum mechanics. A claimed benefit of the quantum-inspired algorithms is that because they use a quantum representation, they can maintain a good balance between exploration and exploitation. It is also suggested that they offer computational efficiencies as use of a quantum representation can allow the use of smaller population sizes than typical evolutionary algorithms. Computational efficiency is important in many financial applications such as real-time trading where systems have to deal with large data flows and a dynamic environment. Consequently, there is a continuing demand for optimisation algorithms which can potentially offer efficiency gains.

Developmental and Grammatical Computing borrows from both a developmental and a grammar metaphor. Grammatical computing refers to algorithms which adopt concepts from linguistic grammars and are dominated by the generative form of grammars. Generative grammars are used to construct a 'sentence' in the language specified by the grammar, and this generative process is metaphorically similar to the developmental process in biology in which 'rules' govern the production of a complex, multi-cellular organism from a single embryonic cell. Generative grammars have been used in natural computing as a convenient representation by which developmental systems can be realised in-silico. The implementations of developmental & grammatical computing, such as *grammatical evolution* (GE) [88, 89] (a grammatical variant of GP) may also embed an evolutionary algorithm typically used to drive the search process. GE has been already successfully applied to financial forecasting, credit rating assessment, and other financial applications.

These families of NC algorithms provide a rich set of tools for the development of quality optimisation, model induction and agent-based modelling applications, and all have seen application in finance. Readers requiring detailed information on these algorithms are referred to earlier chapters in this book. A review of these methods can also be found in [13, 31, 57].

3 Financial Applications

In this section we introduce the application of NC methods across a range of financial areas including forecasting, algorithmic trading, portfolio management, risk management, derivatives modelling and agent-based market modelling.

3.1 Forecasting

Financial forecasting applications may involve the prediction of future values of macroeconomic variables, individual stock, market indices, commodity

futures, the volatility of some financial products, etc. From an optimisation perspective, NC methods can be applied either for variable identification or for parameter estimation (optimisation). For example, NC methods can be used to select the explanatory variables from a large pool of candidates, which are then incorporated into the forecasting model. Even where the modeler knows the appropriate set of explanatory variables and model form, the selection of appropriate model parameters (co-efficients) can be a difficult task, particularly for complex, non-linear model structures. In the more difficult problem where the model form is not known, model induction methodologies such as GP or NNs can be used. This is potentially of considerable importance in financial applications, as many theoretical forecasting models are based on assumptions which are not met in real-world financial markets. This offers opportunities for the application of NC methods as model induction tools in order to 'recover' the underlying data-generating processes directly from the data.

One family of NC methods which has seen extensive application for financial forecasting is NNs (e.g., [121, 56, 2, 17, 42]). They offer particular advantages due to their ability to identify non-linear models, handle noisy data and embed a memory (recurrent NNs). A simple case of index prediction using a basic feedforward Multi-layer Perceptrons (MLP) to construct a financial prediction model is illustrated in [13], where the MLP model is employed to predict the five-day percentage change in the value of the FTSE 100 Index. Ten inputs selected from a range of technical, fundamental and intermarket data were included in the final model:

1. 5-day lagged percentage change of the FTSE 100 index
2. 20-day lagged percentage change of the FTSE 100 index
3. Ratio of the 10 vs 5-day moving average of the FTSE 100 index
4. Ratio of the 20 vs 10-day moving average of the FTSE 100 index
5. Bank of England Sterling index
6. S&P 500 composite $\text{index}_{(t)-(t-5)}$
7. LIBOR 1-month deposit rate
8. LIBOR 1-year deposit rate
9. Aluminium ($ per tonne)
10. Oil ($ per barrel)

In developing the final MLP models, a 11:6:1 structure was utilised, as illustrated in Fig. 2

$$y_t = L\left(\sum_{j=0}^{5} w_j L\left(\sum_{i=0}^{10} b_i w_{ij}\right)\right)$$

where b_i represents $input_i$ (b_0 is a bias node), w_{ij} represents the weight between input node_i and hidden node_j, w_j represents the weight between hidden $node_j$ and the output node, and L represents the hyperbolic tangent function. Generally, NNs is developed through a trial and error approach guided by heuristics, the process is time-consuming, and there is no guarantee that the

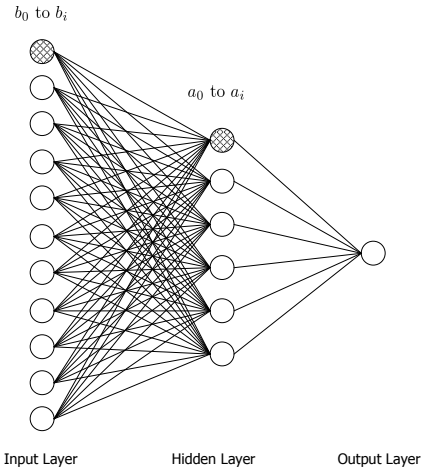


Fig. 2. Index Prediction using MLP model, with 11 input nodes, 6 hidden nodes and a single output node

final network structure is optimal. One approach is to automate the construction of the NN using evolutionary approaches (e.g., [1, 7, 18]).

GP can also be applied for forecasting. One of the best-known examples, EDDIE (which stands for *Evolutionary Dynamic Data Investment Evaluator*), was developed as an interactive decision tool [111, 70]. It is a *genetic programming* based system for channelling expert knowledge into forecasting. Given a set of variables, EDDIE attempts to find interactions among variables and discover non-linear functions.

Other examples of forecasting in the financial literature include the prediction of take-over targets [51, 11, 91, 94, 52], the prediction of auditor qualification of the financial statements of a company [83, 37, 108], the prediction of earnings and the prediction of IPO underpricing. In earnings prediction, [109] uses a GA to select explanatory variables from financial statements in order to predict corporate earnings. In the prediction of IPO underpricing, [93] employs GA for rule-based prediction. A less common, but nonetheless important, application in the literature is the prediction of volatility. Neely [85] uses GP to predict exchange rate volatility, where GP is applied for producing forecasting rules of the out-of-sample daily volatility in the foreign exchange market.

Typically, studies applying NNs, GA or GP methods for financial time series forecasting use measures of goodness of fit drawn from statistics such as *mean squared error*, *sum of squared error*, *mean absolute percentage error*, etc. as their error (or fitness) function. The aim is to uncover or train a model using historical data, which 'fits' that data well. Unsurprisingly, the

choice of fitness function usually has a critical impact on the behaviour of resulting model, hence a model constructed using one fitness metric would not necessarily perform well on another. While many forecasting studies applying NC methods to financial data indicate that models could be constructed to fit historic data fairly well, a common finding is that the quality of the forecasts diminishes, over time, out of sample.

3.2 Algorithmic Trading

Algorithmic trading is defined here as the use of computer programs to assist with any aspect of the trading of financial assets. It can therefore encompass systems which decide on certain aspects of the order such as the timing, price, or even the final quantity of the order. Hence algorithmic trading can be combined with any investment strategy. Below we illustrate a few related processes of algorithmic trading where NC methods can be applied, namely, investment analysis, arbitrage and trade execution.

Investment Analysis

Financial trading has seen a large number of applications of NC methods. Typically these studies take one of two approaches of investment analysis, using either fundamental data (*fundamental analysis*) or market data, primarily price and volume (*technical analysis*).

Fundamental Analysis

Taking the example of investing in stocks, fundamental investment concentrates on the use of accounting information about the company, as well as industry and macroeconomic data, in order to identify companies which are mispriced by the market. In other words, the objective is to identify stocks which are good value (underpriced by the market), or stocks which are overpriced by the market (and therefore are candidates for 'shorting'). In this approach, the investor needs to develop stock screening rules in order to decide which stocks to invest in. These rules were formulated manually in decades before computers. With a natural computing algorithm such as the GA, a large range of stock filter rules can be searched efficiently in order to find the highest-quality rules. In this approach, each individual in the population corresponds to a potential stock filter rule. The utility of these rules are tested using historical data, with the best rule (or set of rules) then being used for investment purposes (Fig. 3). More generally, GP methods can be applied to evolve the structure of the filter rules.

Technical Analysis

In contrast to investors using a fundamental investment approach, technical analysts attempt to identify imbalances in the supply and demand for

High sales growth relative to industry average?	High debt level relative to industry average?	High level of cash flow from operations relative to industry average?	High level of liquidity relative to industry average?	High profit level relative to industry average?

Fig. 3. String encoding of a number of fundamental indicators. Each indicator can be coded as a 0 (no) or 1 (yes)

a financial asset using information from the time-series of the asset's trading (such as historical price, volume and volatility data). Usually, investors who adopt a technical analysis approach look to combine technical indicators (pre-processed price and volume time series data about a financial asset), in order to produce a 'trading signal'. For example, a 'technical indicator' could be the *moving average convergence-divergence* (MACD) oscillator, calculated by taking the difference of a short-run and a long-run moving average. If the difference is positive, it may indicate that the market is trending upward. For example a buy signal could be generated when the shorter moving average crosses the longer moving average in an upward direction. A sell signal could be generated in a reverse case. Hence a sample MACD trading rule could be:

$$IF\ x\text{-day MA of price} \geq y\text{-day MA of price}$$

$$THEN\ Go\ Long\ ELSE\ Go\ Short$$

where $x < y$. The optimal value of x and y can be evolved through a genetic algorithm, in order to maximise the trading profit (the fitness measure). A candidate solution encoded as a binary string of length 8 is illustrated below.

x:	0	0	0	0	1	0	1	0
y:	0	0	1	1	0	0	1	0

This solution indicates that $x = 10$ and $y = 50$. The MACD oscillator is a crude band-pass filter, removing both high-frequency price movements and certain low-frequency price movements, depending on the precise moving average lags selected. In essence, the choice of the two lags produces a filter which is sensitive to particular price-change frequencies. In a recursive fashion, more complex combinations of moving averages of values calculated from a MACD oscillator can themselves be used to generate trading rules. In past decades, the search for apparently useful technical indicators (or combinations of these) was undertaken manually by investors who back tested various indicators on historical financial data. GP allows the automation of this process, with the concurrent vast expansion of the search space which can be feasibly searched [33].

A trading rule should specify the entry, profit-taking, and stop-loss or exit strategy. NC methods can be used for rule optimisation (to find optimal

parameters for a fixed rule) or rule induction (to find optimal combination of diversified rules). Early applications of EC to uncover trading rules include [10, 84, 5, 47]. The flexibility to implement different fitness functions is one particular advantage of EC approaches. However, there are also some issues related to the fitness functions as to include transactions costs and to consider different types of risks [92]. While markets exhibit periods in which a static trading rule can work, it is hard to find evidence of rules which are successful over long time periods. Of course, as financial markets comprise a dynamic system, the utility of any static trading system can be expected to degrade over time [34]. One basic way of examining the characteristics of a trading system is to use an equity curve (Fig. 4). The use of an adaptive trading strategy seems more plausible [49, 27].

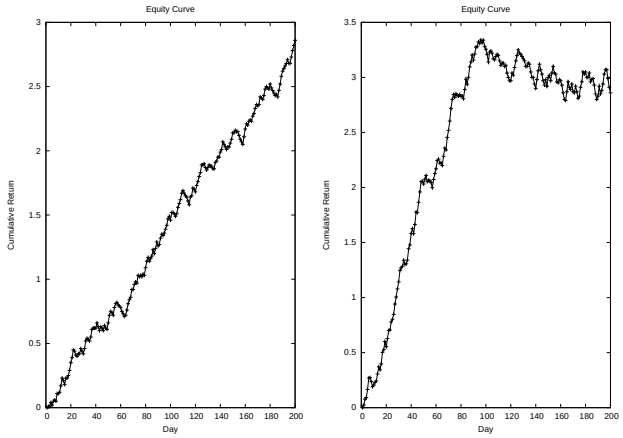


Fig. 4. Sample equity curves showing cumulative returns on the y axis and time on the x axis. The left-hand graph exhibits gradual return accumulation, whereas the right-hand graph suggests that the model is working less well on the second half of the time period

Recent work has seen a broadening of the information sources used as inputs in trading models. Instead of typical data drawn from the market, financial statements or macroeconomic data, for example, [107, 65] used text data drawn from either internet message boards / the financial press in the creation of trading rules.

A wide range of forecasting approaches have also been employed to support making trade decisions, such as support vector machines [41], and hybrid methods like neuro-fuzzy hybrids [125], neuro-genetic hybrids, geno-fuzzy [49] and ensemble methods (combining multiple models) [64].

Arbitrage

Arbitrage trading can be defined in a variety of ways, but broadly speaking, these trades seek to make profits by exploiting price differences of identical or similar financial instruments, on different markets or in different forms (for example, buying a share at \$23.78 on one exchange, and selling it immediately on another for \$23.82, thereby exploiting a pricing difference for the same asset between the two exchanges). As would be expected, arbitrage opportunities tend to be closed very quickly and transactions costs can negate apparent arbitrage possibilities.

A simple example of an arbitrage play based on *Put-Call Parity* is illustrated in [113]. In essence, the concept underlying this trade is that the price of a 'long' position on an asset and an associated put option (the right to sell that asset in the future at a specified price, see Section 3.5 for a detailed description of options) must be equal to the price of a long call option on the same asset and a long position in a risk-free bond. Specifically, for example, for a European option:

$$S_0 + P_E(S_0, T, K) = C_E(S_0, T, K) + K(1 + r)^{-T}$$

where S_0 is the current price of the underlying asset, P_E is the current price of a European put, C_E is the current price of a European call, r is the risk free rate of return, K is the strike price for both options, and T is the time to maturity of the options. If either the put or call option are mispriced the investor can, in theory, make a risk-free gain by constructing a portfolio of the four financial instruments.

The above example, describes an arbitrage opportunity between the cash market (for the asset) and the option market. More generally, arbitrage opportunities can also exist between cash and futures markets and between futures and options markets. In its purest form, arbitrage is a risk-free transaction but in reality, most arbitrage trades are exposed to some risk such as liquidity risk or credit risk, and more significantly, execution risk where prices move before all elements of the trade can be completed.

An alternative approach that wait for arbitrage opportunities to emerge and then try to trade on them, is to anticipate or forecast opportunities in advance of the actual mispricing occurring. Markose, Tsang and Er [76] adopted this approach and developed a GP model to predict arbitrage opportunities between the FTSE 100 index futures and options market up to ten minutes in advance of the arbitrage opportunity arising. Another example of the application of computational intelligence for uncovering arbitrage opportunities is provided in [113] who used self-organising, fuzzy NNs to identify mispriced American style currency options between the USD and the GBP, and then use a Delta hedge trading strategy to execute the arbitrage play.

Trade Execution

An important issue in trading financial assets is the efficient execution of large institutional size orders, and applications of EC for this task have begun to emerge in recent years. Typically, orders to buy or sell a stock can be either *market orders* (the transaction is undertaken immediately in the market at current prices) or limit orders (the purchase / sale must occur at a price which is no greater than / or less than a specific price). When a market order is placed, the customer does not have control over the final price, and in a limit order, while the customer has some price control, there is no guarantee that the order will actually be executed). So for example, if a customer places a limit order to buy a stock at $25 per share the transaction will only take place if the market price falls to $25 or less.

Over the last number of years, trading algorithms have been executing an ever-increasing number of trades on markets. In the U.S. their rise has been brought about through a series of technological and regulatory changes. Since 2001 with the move to decimalization of the U.S. equity markets, and the widespread acceptance of electronic market places, the average trade size has declined from 1,200 shares per transaction in 2000 to 300 shares in 2008 (NYSE Euronext). This in turn has led to an explosion in the number of trades executed and a narrowing of spreads, with large institutional orders taking longer to execute. As a result, investors wishing to trade large blocks face tradeoffs in balancing the risk of tipping their hand and providing information to the marketplace, thereby suffering market risk as the trade is executed. Trading algorithms seek to optimally execute these orders, using the vast amounts of data produced by the market place and submitting appropriately sized smaller orders to various destinations with the aim of achieving best execution. In reaching this goal an entire ecology of different trading algorithms have been designed to perform under different market conditions, with recent innovations intelligently switching between these algorithms depending on current market conditions.

When trading shares, particularly when an investor is looking to buy or sell a large quantity of stock, the problem of *market impact* arises. Market impact occurs when the actions of an investor start to move the price adversely against themselves. Hence, market impact is the difference between a transaction price and what the market price would have been in the absence of the transaction. For example, the order may be executed as quickly as possible through sweeping any orders posted to the limit-order book, however this would incur significant cost and drive the price of the stock against the investor. In this case the investor avoids market risk but, by demanding instantaneous liquidity, incurs significant market impact costs. The obvious strategy to minimise market impact is to break up the order up into smaller lots and spread it over several purchases. While this will reduce the market impact, it incurs the risk of suffering *opportunity cost*, that market prices may start moving against you during the multiple purchases. Added to this, the

steady flow of small orders over time will inform other market participants of the presence of a large institutional order and so encourage competitors to run ahead of the investor. Hence, the design of trade execution strategies is intended to balance out the total cost of market impact and opportunity cost while maintaining a tight control over information leakage.

In selecting a trade execution strategy, the investor must not only balance her preferences but must also be prepared to adapt and change strategy as market conditions evolve. NC techniques provide ample scope to assist in uncovering information that can help optimise trading algorithm selection and/or adaptation. Various rules and heuristics can be evolved and adapted using (for example) GAs that can provide a trading strategy with predictive capability [102] with the aim of selecting best trading tactic under current market conditions. For institutional sized orders, which can be on the order of millions of shares, the reduction in average price by a couple of pennies can lead to significant savings.

Despite the importance of optimising trade execution, there has been relatively little attention paid in the literature to the application of evolutionary methodologies for this task. One interesting exception is Lim and Coggins [71] who used a GA to evolve a trading strategy in order to optimise trade execution performance using order book data from the Australian Stock Exchange. In this study, the approach taken was to initially split each trade into a series of N equal sized orders, and the objective was to evolve the timing strategy for the execution of each of these N orders during a single trading day. Each order was submitted as a limit order at the best ask or bid prevailing at the time the order was submitted, depending on whether the investor was seeking to buy or sell shares. A simple traditional trading strategy could be to submit one of these orders every ten minutes. However, there is no guarantee that a ten minute order spacing would produce good results in terms of minimising market impact. Lim and Coggins [71] used a GA to uncover good quality timings for each order by evolving a chromosome of N genes, where each gene encoded the maximum lifetime that the order would remain on the order book (if it had not already been executed) before it was automatically ticked over the spread (for example, a limit buy order being repriced to the current ask) to close out the trade. Any uncompleted trades at the end of the day were closed out the same way. Hence, the GA evolved the maximum time that each order would be exposed to the market before being crossed over the spread.

A variety of fitness functions could be designed to drive the evolution of the trading strategy but a common metric of trade execution performance is its *Volume Weighted Average Price* (VWAP):

$$VWAP = \frac{\sum(Price \cdot Volume)}{\sum(Volume)}$$

The VWAP of a strategy can be calculated and benchmarked against (for example) the overall VWAP for that share during the period of the trad-

ing strategy's execution. The aim is to evolve a strategy which produces as competitive a VWAP as possible.

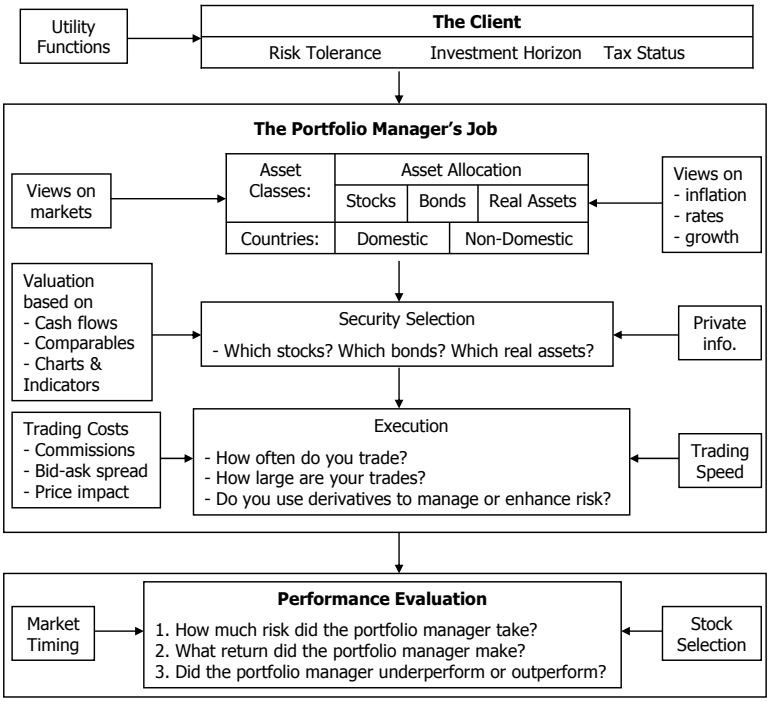
In the above approach, the basic structure of the execution rule is determined in advance (number of trades etc.) and the task of GA is to parameterise the rule. Another approach which could be applied is to use GP to evolve the structure of the execution rule, as well as its parameters.

In real-world trading a number of interesting additional issues arise. While the order book provides an indication of the current state of supply and demand for a share, it does not always present a true reflection of investor's trading intentions. For example, market participants can attempt to 'game' the order book by placing limit orders which are subsequently cancelled or amended, and (as described above) the order book may contain *iceberg orders* (a large order which has been split into several smaller orders in order to disguise the investor's trading intent). The dynamic nature of the order book suggests that an agent-based modelling approach could be used in order to uncover robust trade execution strategies.

3.3 Portfolio Management

In finance a *portfolio* refers to a grouping of financial assets such as stocks, bonds and cash equivalents. Portfolio management involves the art and science of making decisions about investment mix and policy, matching investments to objectives, asset allocation, and balancing risk and return. An overview of the portfolio management process is illustrated in Fig. 5. For a portfolio manager, the first and foremost part of the investment process is understanding the client's needs, the client's tax status and his or her risk preferences. The next part of the process is the actual construction of the portfolio, which involves asset class allocation and security selection decisions. Asset class allocation refers to the allocation of the portfolio across different asset classes defined broadly as equities, fixed income securities and real assets (such as real estate, commodities and other assets). The security selection decision refers to the selection of specific securities under each asset class. The final component of portfolio management is trade execution, the efficient purchase or sale of the relevant assets in the marketplace. An important and open question is how best to measure the performance of the resulting portfolio.

'Optimization is the engineering part of portfolio construction', as mentioned by Fabozzi et al [44] in their recent survey for quantitative equity portfolio management: *'Most portfolio construction problems can be cast in an optimization framework, where optimization is applied to obtain the desired optimal risk-return profile'.* Multiple elements of the portfolio management process, such as asset allocation, security selection, index tracking, etc. where optimisation is crucial, are amenable to NC methodologies. Below we illustrate applications of NC methods for the purposes of asset allocation and index tracking.



Source: Investment Philosophies (2003), by Damodaran, A. [29]

Fig. 5. An overview of the portfolio management process

Asset Allocation

Asset allocation is the selection of a portfolio of investments where each component is an asset class rather than an individual security. The aim of asset allocation is to invest a fixed amount of money in a diverse set of assets so as to maximise return while minimising a risk measure. The solution to this is a Pareto frontier (usually referred to as the *efficient frontier*) shown in Fig. 6, as for a given level of risk, there should not be a portfolio with a higher rate of return, or for a given level of return, there should not be a portfolio with a lower level of risk. Once the frontier is uncovered, the final choice of portfolio is determined by the individual investor's risk preference.

A classical approach used for asset allocation is the Markowitz mean-variance model [77, 78]. It assumes that investors wish to maximise their return (measured as mean or expected return) and minimise their risk (measured as variance or the standard deviation of their return). This produces the risk return trade-off. The goal of the Markowitz model is therefore to find an optimal portfolio p of N assets, each with a weighting w_i (in percentage),

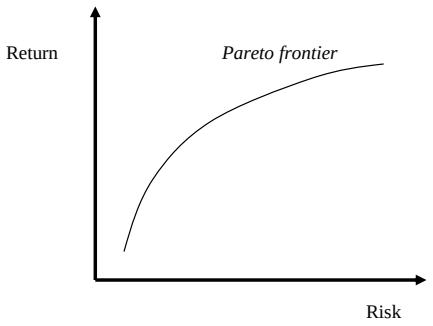


Fig. 6. Pareto frontier. The points that correspond to the risk-return of the set of portfolios that are Pareto optimal

such that the return E_p is maximised:

$$E_p = \sum_{i=1}^{N} w_i \cdot \mu_i$$

while minimising the variance of the return:

$$V_p = \sum_{i=1}^{N} \sum_{\substack{j=1 \\ i \neq j}}^{N} w_i \cdot w_j \cdot \sigma_{ij}$$

subject to

$$\sum_{j=1}^{N} w_i = 1$$
$$w_i \geq 0; \; i = 1, \ldots, N$$

where σ_{ij} is the covariance of return between asset i and j, the constraints are used to ensure that all the money is invested and all investments are positive (assuming that short selling is not allowed). It produces a multi-objective optimisation problem (maximise return, minimise risk) and there are two equivalent formulations which are the dual of each other (firstly fix a value of expected return and find the portfolio that minimises the risk, secondly select a level of risk, and find the portfolio that maximises the expected return). Either of the formulations produces a quadratic programming problem and several algorithms exist which can be applied to uncover good quality portfolios.

Adding Real-world Constraints

Quadratic programming relies on a number of assumptions, including a single quadratic objective function, linear constraints, and the existence of a positive definite covariance matrix between the asset returns. However, these assumptions are typically breached in the real-world. For example, the constraints can include *cardinality constraints* (a limit on the number of assets which can be held in the portfolio), i.e. $\sum_{i=1}^{N} sign(w_i) = K$, or there may be *threshold limits* on the amount of investment in any single asset, i.e. $w_i \leq m_i,\ i = 1, \ldots, N$, where m_i is the threshold amount for asset i. Other constraints may include industry or sector (or *concentration*) holding constraints, round lot constraints, and transactions costs may have both fixed and non-linear variable cost elements. These constraints can lead to non-convex, non-differential models. In addition, some constraints may be hard and others may be soft. Hence, real-world portfolio selection can present a difficult, high-dimensional, constrained optimisation problem, which is beyond the capabilities of traditional optimisation methods. In this setting, heuristic approaches such as evolutionary computing methods are of particular interest because of their ability to find good solutions even if optimality is not assured.

MOEA and Portfolio Selection

An extensive literature on Multi-Objective Evolutionary Algorithms (MOEA) has developed over the past twenty years (see [95, 32, 26] for a detailed review). MOEA have an advantage of maintaining a population of solutions and therefore offer the potential to uncover multiple points on the Pareto frontier. A wide range of approaches have been offered to deal with different types of constraints including penalty function approaches, repair mechanisms, the design of appropriate representations and diversity-generation operators. A stream of literature also exists which has used hybrid Evolutionary Algorithms or local search techniques for MOEA. Many of these approaches have been applied for portfolio selection.

The earliest papers to apply EAs for portfolio selection include [8, 72, 99, 116]. In the case of [8] multiple GA populations were used to identify the Pareto frontier. Rather than use the standard Markowitz model, the authors used a downside risk measure. The multi-objective problem was converted into a single objective using a trade-off function, with each population using a different trade-off coefficient and therefore producing a different portion of the Pareto frontier. The formulation of the portfolio problem includes cardinality and buy-in constraints, and a repair mechanism was applied in order to ensure that generated solutions were feasible. The utility of differing crossover operators and differing genotypic representations for the portfolio selection problem was examined by [103, 104], and the application of EC hybrids was examined by [106] and [105]. The impact of cardinality constraints was examined in [46] and [82] (the latter also adopted an EC hybrid approach). A number of other

applications including stock ranking, and credit portfolio optimisation have been reported in the literature (see [16, 96] for a review of MOEA applications in finance).

In practical settings, investment managers are concerned with a variety of risk and return measures, not just expected return and its variance. For example, *value at risk* (VaR), the risk that a portfolio could lose a significant amount of its value over a defined time window (more precisely, VaR at level (1-α) is the α-quantile of the loss distribution), has gained importance especially for regulatory purposes. VaR is typically non-linear and non-convex, making optimisation in models which use this metric difficult. More generally, a portfolio manager may be concerned with more than one risk constraint. A variety of papers have applied MOEA to non-Markowitz risk metrics, including [73] which uses a compound risk metric. Hochreiter [53] introduces an evolutionary stochastic portfolio optimisation methodology and illustrates its application using a set of structurally different risk measures, which include, Standard Deviation, Mean-absolute Downside Semi Deviation, Value-at-Risk, and Expected Shortfall. Recent work has also seen the application of co-evolutionary MOEAs for portfolio optimisation [38].

Index Tracking

There are two common types of portfolio management strategies: passive and active. Active portfolio management consists of picking assets which are expected to outperform the market. In contrast, passive management simply tracks a market index, where the objective is to form a portfolio that replicates the performance of an index as closely as possible.

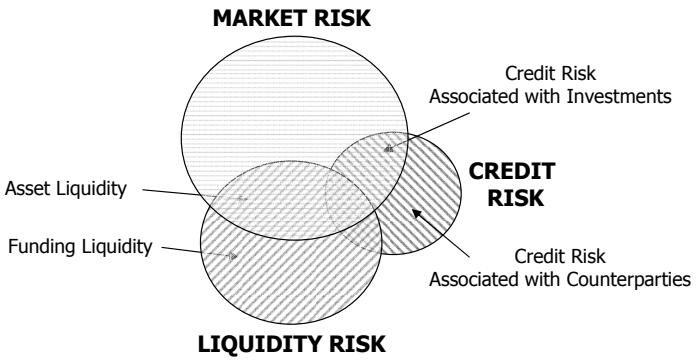
Passive portfolio management strategies have become very common in recent decades. Just like general portfolio optimisation, the construction of an index tracking portfolio is a constrained optimisation problem, where the objective is to minimise a measure of *tracking error* (or difference between the return to the portfolio and the return to the index), subject to a variety of constraints, similar to those in the general portfolio optimisation problem. The solution space is non-convex suggesting a useful role for population-based, global optimisation heuristics.

At first glance, the construction of an index tracking portfolio appears trivial, merely requiring the purchase of the same basket of assets that make up the index, using the same weight that each asset has in the index. However, the creation of a perfect replica portfolio is difficult for several reasons including a requirement for frequent portfolio re-balancing (with associated transactions costs), integer constraints on asset purchases, mandate limits on the maximum holding in any individual asset etc. Another practical issue is that not all assets making up market indices have equal liquidity. Hence, there may be good reason to seek to track the performance of a broad market index using a portfolio which comprises of a subset (rather than all) of the assets making

up the index. Another feature of this problem is that investor's risk attitudes to tracking errors are not symmetric. While investors will not wish to underperform the index, they will not object if the portfolio outperforms the index. EC applications to the index tracking problem include [98], [105], [90] and [75] which also examines the impact of investor loss aversion preferences on tracking portfolio construction.

3.4 Risk Management

Risk management is a critical aspect of investing. Some of the main types of risks faced by investors are illustrated in Fig. 7 below.



Source: Sound Practices for Hedge Fund Managers (2000) [100]

Fig. 7. Risk illustration

Market Risk Computation

Market risk refers to the risk faced by an investor arising from changes in financial market prices. The degree of risk of loss faced by an investor will vary depending on the price volatility of the assets they hold. Not all assets will have the same degree of volatility. There are various techniques to measure market risk [79], applications of NC methodology include the calculation of Value-at-Risk (VaR) and the sensitivities. The VaR approach measures the worst expected loss under normal market conditions over a specific time interval. The loss distribution is usually assumed to be normal when calculating VaR. Evolutionary algorithms do not need to embed this assumption and can incorporate any preferred loss distribution type [114]. NNs have also been used for other market risk measures, such as Conditional VaR estimates [67], and expected shortfall [35]. The sensitivity analysis approach measures how much the portfolio's (or a specific financial instrument's) value is expected to

change if there is a small change in one of the market-risk factors, such as interest rates, equity prices, commodity prices etc. The sensitivities are the so-called *greeks* for derivatives; *duration* and *convexity* for fixed income products. Keber [60] has used GP to calculate greeks of the American-type put options (see Section 3.5 for an explanation of derivative products).

Credit Risk Assessment

Credit risk is the risk that a counterparty to a deal fails to perform their obligations. Credit risk assessment is an important component of the lending decision of financial institutions and other commercial companies. Examples of decisions of where credit scoring could be useful include, decisions such as should a loan be extended to a firm or to an individual, should a customer be allowed to purchase goods on credit, or what credit limit should be offered to a customer on their credit card?

Over the past several decades an extensive literature has amassed on modelling creditworthiness and default risk. Traditional statistical methods including linear discriminant analysis [6] and logit [87] have been applied. In all of these applications, the objective is to develop a model which will provide a metric of creditworthiness from a series of explanatory variables. Typically, in assessing corporate creditworthiness, explanatory variables can include numbers drawn from the financial statements of the firm, from financial markets, general macro-economic variables, and non-financial, firm-specific information). In assessing personal consumer creditworthiness explanatory variables can include income, age, occupation, current employment status, past borrowing record etc. [118, 124, 69]. Closely associated streams of academic literature include corporate failure or bankruptcy prediction [6], and the reverse engineering of the bond-rating models used by rating firms such as Standard & Poor's (S&P), Moody's, Fitches' or Dominion Bond Rating Service [43, 39, 48]. Assessments of credit default probability could also form a useful input into a stock or bond trading model. A practical problem in constructing risk-assessment models is that there is no clear theoretical framework for guiding the choice of explanatory variables or model form. In the absence of an underlying theory, most published work on credit rating employs a data-inductive modelling approach. This produces a high-dimensional combinatorial problem, as the modeler is attempting to uncover a good set of explanatory variables and model form.

An illustration of an early credit risk assessment model is provided by Altman's [6] classic study in which five ratios were combined to produce a linear discriminant classification model for corporate bankruptcy. A Z score was calculated for each company, and this value determined whether the company was classified as bankrupt or solvent:

$$Z = 0.012X_1 + 0.014X_2 + 0.033X_3 + 0.006X_4 + 0.999X_5$$

where

X_1 = working capital to total assets
X_2 = retained earnings to total assets
X_3 = earnings before interest and taxes to total assets
X_4 = market value of equity to book value of total debt
X_5 = sales to total assets

As the range of NC techniques have expanded over the past twenty years, each new technique has been applied to credit scoring and corporate failure prediction. Examples include feedforward NNs [120, 9], self-organising maps [97, 62], GAs [63, 115], Ant models [117, 13], GP and GE [80, 13, 88, 4]. The domain offers particular potential for evolutionary automatic programming methodologies such as GP or GE as these methods can produce human-readable credit decision rules. This can be important in some countries where lenders can be required to justify decisions not to grant loans. Another advantage of GP and GE is that the rule-evolution process can be seeded using domain knowledge.

Another closely-related application is the prediction of bank failure [68], with many regulatory authorities using risk models in order to assess which financial institutions require the closest scrutiny. Obviously, for these applications it is important that the regulatory authority can verify the correctness of the underlying prediction model, hence methodologies which can incorporate expert knowledge and produce interpretable decision rules, such as fuzzy systems and GP, are of particular interest.

Of course, there are other types of risks which need to be quantified in practice, such as liquidity risk and operational risk (arising due to poor or inadequate management control systems or due to human error). However, as yet, there is little literature concerning the application of NC methods in these areas.

3.5 Derivatives Modelling

Derivatives are contracts whose value is derived from the value of the underlying assets, such as equities, interest rates, currencies, market indices, commodities etc. Two of the best known forms of derivative are futures and options. A future is an agreement to buy or sell goods, currency or securities on an agreed future date and for a price fixed in advance. An option is a financial instrument simply gives the holder (buyer) the right, but not the obligation, to buy (a call option), or sell (a put option), a specified underlying asset at a pre-agreed price on or before a given date. A European style option refers to an option that may only be exercised on expiration; while

an American style option can be exercised on any trading day on or before expiration.

The key issue for investors wishing to trade in derivatives is the determination of the fair price for the derivative. For some standard derivatives (based on specific assumptions such as continuous time finance theory), closed-form pricing equations have been determined (e.g. the Black - Scholes model [12, 81] for pricing European options, the Cox et al. binominal model [28] etc.). The traditional approach to pricing a derivative is [86]:

- specify a stochastic process for the underlying asset(s),
- derive the pricing equation for the derivative (using a no-arbitrage argument), and
- price the derivative by solving the pricing equation.

Of course, this approach can be difficult to implement, as the relevant stochastic process may be imperfectly understood, and the pricing equation may be too difficult to solve analytically. In the latter case, there is scope to use tools such as Monte Carlo (MC) simulation to estimate the expected payoff and the associated payoff risk for the derivative. In valuing a complex derivative using MC, the typical approach is to randomly generate a set of independent price paths for each security underpinning the derivative, then compute the present value of the payoff to the derivative under each set of these price paths. The simulation process is repeated multiple times and the distribution of the payoffs is considered to characterise the derivative. An example of this approach is illustrated in [61]. A critical issue in applying a MC approach is the correct design of the theoretical pricing model.

There have been two main avenues of application of NC methods in pricing financial derivatives, namely,

- model calibration, and
- model induction.

In model calibration, the objective is to estimate the parameters of (or 'calibrate') a theoretical pricing model. The parameters are estimated by fitting the model to the relevant returns time series. Typically the pricing model will have a complex, non-linear structure with multiple parameters. Hence, global search heuristics such as the genetic algorithm can have utility in uncovering a high-quality set of parameters. Examples of the use of NC algorithms for model calibration include [30, 45].

In model induction, NC approaches such as GP would have particular utility when little is known about the underlying asset pricing dynamics as both the structure and the parameters of the pricing model are estimated directly from the data, thereby extracting the pricing model implicitly. Even where theory does exist, model induction methodologies allow us to investigate whether other plausible theories may exist to explain observed prices. Applications of such methods include NNs [74, 54, 119], self-organising, fuzzy NNs [113], or GP [19, 23, 58, 24, 123] to recover a proxy for the price-generating model directly from the data.

Below we illustrate an example of using GP to generate an option pricing model. One advantage noted by [24] is that the GP process can be seeded with the Black-Scholes equation, with the final resulting model being an adaptation of the Black-Scholes equation for conditions which violate its underlying assumptions. For example, in the Black-Scholes setting for a non-dividend paying European call option there are five factors that affect the price of the option (assuming no dividends):

1. S_0 - the underlying asset price
2. K - the exercise price of the option
3. T - the time to maturity (of the option)
4. r - the risk free rate of return (of the underlying asset)
5. σ - the expected volatility of the asset price

Items 1 and 2 can be combined to give $(S_0 - K)$ or a measure of the *moneyness* of the option. An option is said to be 'in the money' when this value is greater than zero, and 'out of the money' when it is less than zero. In developing a pricing model for options from these factors the Black-Scholes model embeds several critical assumptions. It is assumed that the stock price undergoes a diffusion process that is log normally distributed, with an instantaneous drift and volatility given by μ and σ respectively. The volatility σ and the risk-free rate r are also assumed to be constant during option's life. This implies that:

$$\ln \frac{s_T}{s_0} \sim N\left((\mu - \frac{\sigma^2}{2})T, \sigma\sqrt{T}\right)$$

and in turn this leads to:

$$S_T = S_0 e^{\eta\ T}, \eta \sim N\left((\mu - \frac{\sigma^2}{2}), \frac{\sigma}{\sqrt{T}}\right)$$

where μ is the instantaneous expected return on the stock, σ is the instantaneous volatility of stock price return, S_T is the stock price at a future time T, S_0 is the stock price at time zero, $N(m, s)$ denotes the normal density function with mean m and standard deviation s and η is defined as the continuously compounded rate of return per annum realised between time zero and T. The Black-Scholes formula for the price at time zero of an European call option on a non-dividend paying stock is therefore:

$$C_0 = S_0 N(d_1) - K e^{-rT} N(d_2)$$

where

$$d_1 = \frac{\ln(S_0/K) + (r + \sigma^2/2)T}{\sigma\sqrt{T}}$$

$$d_2 = d_1 - \sigma\sqrt{T}$$

$N(\chi)$ is the cumulative probability distribution function for a standardised normal distribution, C_0 is the price of the European call option at time 0.

Of course, assuming that the Black-Scholes model did precisely value options, model induction techniques such as neural networks or GP could be used to recover the structure of the option pricing model directly from a historical time series of option prices (C_0) and the five factors that influence option prices. Fig. 8 illustrates a tree representation of the Black-Scholes Model which could (potentially) be recovered by GP.

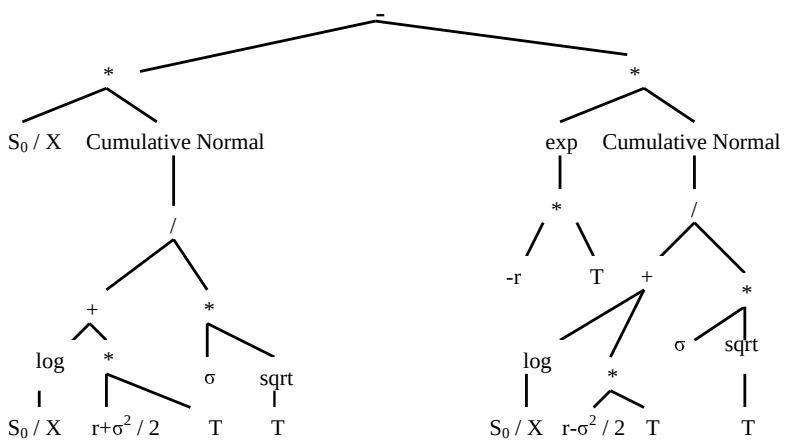


Fig. 8. Stylised lllustration of a tree representation of the Black-Scholes model (not all sub-trees are shown)

In reality, some of the key assumptions in the Black-Scholes model do not hold in real-world option markets, and hence the model does not explain observed option prices correctly. For example, prices can experience discontinuous jumps, the distribution of price changes has fatter tails than those implied by a log normal distribution, and asset price volatility changes over time.[5] The latter issue can be easily seen if market prices are substituted into the Black-Scholes model, in order to calculate the 'implied volatility' for that option. If the assumptions underlying the Black-Scholes option pricing model were correct, the implied volatilities for options on the same underlying asset would be constant for different strike prices and maturities. However in practice the Black-Scholes implied volatilities are varying over strike price and

[5] There is a long line of literature which examines alternative (non-normal) stock return models including Poisson jump-diffusion return processes and GARCH processes. However, closed-form solutions for the option price cannot be obtained for all these models.

maturity. Keber [59] has used GP to generate formulae for determining the implied volatility based on American put options.

3.6 Agent-based Market Modelling

The essence of Agent-based Modelling (ABM) lies in the notion of autonomous agents whose behaviour evolves endogenously leading to complex, emergent, system dynamics which are not predictable from the properties of the individual agents. ABM is an exciting new tool for exploring behaviour in financial markets that are far from traditional notions of equilibrium, and where agents exhibit behaviour that is less than fully rational at times. Financial markets are particularly appealing applications for agent-based methods especially considering the following: issues of price and information aggregation tend to be sharper in financial settings where agent objectives tend to be clearer; financial markets are rich in data sets (such as price and volume data) at many different frequencies, that can be used for testing and calibrating agent-based models; financial markets are well organised, centralised, and trade homogeneous products in a generally efficient fashion relative to markets for other goods and services; there are continuing developments in the area of experimental financial markets which give carefully controlled environments which can be compared with agent-based experiments; the key debates in finance about market efficiency and rationality are still unresolved; many puzzles of the financial time series (such as the volatility persistence) are still not well understood. In designing ABMs of financial markets, modellers face a daunting list of design choices which can critically impact on the system's behaviour. Important design questions include:

- Representation and structure of the actual trading agents. Agents can vary from simple budget constrained *zero intelligence agents*[6] as in Gode & Sunder [50] to sophisticated learning agents as in Chen & Yeh [20].
- The actual mechanism that governs the trading of assets. Ways of designing this include assuming a simple price response to excess demand, building the market such that a kind of local equilibrium price can be found easily, or explicitly modelling the dynamics of trading to mimic the continuous trading of real-world markets.
- Types of securities to be incorporated into the agent-based market model, where typically simple securities (such as stocks) are considered.

In designing agent-based models (ABMs) of financial markets, NC methods can be used to model the information processing and storage by agents, the process of adaptive learning by agents, or to model the trading mechanism. One example of the use of ABM to simulate a financial market is provided by LeBaron [66] in building the well-known Santa Fe Artificial Stock Market.

[6] Refers to a type of trader that randomly makes price bids (offers to buy) and/or price asks (offers to sell) subject only to a budget constraint.

This model simulates price generation and trading in a market made up of artificial adaptive agents. This is a prototype example of a complex system and is thought to illustrate the benefits of simulation modelling. The Santa Fe Artificial Stock Market consists of a central computational market and a number of artificially-intelligent agents. The agents choose between investing in a stock and leaving their money in the bank, which pays a fixed interest rate. The stock pays a stochastic dividend and has a price which fluctuates according to agent demand. The agents make their investment decisions by attempting to forecast the future return on the stock, using GA to generate, test, and evolve predictive rules. Other applications of ABM include the simulation of a foreign exchange market [55], the modelling of an artificial stock option market [40] and the modelling of an artificial payment card market [3].

A key output from the ABM literature on financial markets is that it illustrates that complex market behaviour can arise from the interaction of quite simple agents. Carefully constructed, ABM can help increase our understanding of market processes and can potentially provide insights for policy makers and regulators, where unlike laboratory sciences, we cannot re-run a real market under different regulations 'to see what would happen.' Of course, issues of model validation are important in all ABM applications including those in financial markets.

4 The Future

Though a plethora of academic literature on NC applications in finance exists, it is notable that many papers have concerned proof of concept rather than robust, industry-strength, applications. While NC methods offer potential in multiple areas in finance, the maturing of their application requires future work focusing on complex real-world problems. This will require the construction of multi-disciplinary research teams, drawing academic expertise as necessary from finance, computer science, mathematics, biology, etc. and combining this with industrial collaborators. For example, quality work on trading systems requires deep knowledge of market micro-structure, the regulatory environment, available financial instruments, and the technology available to traders. Below we indicate some promising future directions for research at the nexus of natural computing and finance.

- ***Forecasting***: While forecasting models applying NC methods can typically be constructed to fit historical data fairly well, a common finding is that the quality of the out-of-sample forecasts degrades over time. Hence we can expect to see increased use of more sophisticated methods for preprocessing the raw time-series inputs, and for the adaptation of the resulting models in response to changing environmental conditions. Another area of growing interest is the incorporation of information from text mining (e.g. from the financial press) into forecasting models.

- ***Algorithmic Trading***: While many published papers have focused on the development of simplified trading systems, successful real-world applications have often focused on the support of specific elements of the investment process. In the medium term, we can expect to see a notable increase in the rigor of published work in this area as computer scientists form teams with finance academics and practitioners, incorporating realistic models of market microstructure. The area of trade execution has seen relatively little published application of NC methodologies, despite its real-world significance. Model induction tools such as GP offer interesting potential here, as do agent-based modelling approaches. The latter could be used to uncover robust trade execution strategies.
- ***Portfolio Optimisation***: There has already been an extensive application of NC methods for portfolio optimisation, but we still require further systematic investigation of portfolio constraints for special sub-application areas including hedge funds, pension funds, and insurance. The extension of dynamic portfolio optimisation needs further development, which may involve the simulation or forecasting of extreme market situations, and the solution of multi-stage constraint optimisation problems.
- ***Risk Management***: Recent events on the financial markets have underscored the importance of risk management, and the weakness of existing theoretical models in this area. It is interesting to note that applications of NC methods in risk management have not attracted as much attention as might be expected in the literature and this remains an open research area. For example, NC methods could be applied to assist in the development of enterprise-wide risk management systems, and to improve the flexibility and efficiency of large scale multi-stage *asset liability management* (ALM) models.
- ***Derivatives Modelling***: In spite of the vast array of derivatives products available, and the weakness of financial theory once we move beyond vanilla products, there have only been a relatively limited number of applications of NC for model calibration or induction in this area. Possibilities also exist to hybridise NC methods with traditional numerical methods, and to develop dynamic derivative pricing models.
- ***Agent-based Market Modelling***: The field of ABM is attracting significant attention with the increasing questioning of agent homogeneity which underlies classical financial economics. ABM allows us to examine the effect of differing forms of market structure on market behaviour. Doubtless, the next few years will see increased focus on this given the failures of market regulation during the recent financial crisis. The co-evolutionary element of markets lends itself well to NC approaches in terms of modelling of agent behaviour and strategy adaptation.

A practical issue that arises in the application of NC to finance is that the underlying algorithms are themselves undergoing a process of maturation. Recent years have seen extensive research in order to extend canonical NC algorithms into high-dimensional environments (enhancing algorithmic scalability), to develop efficient algorithms for constrained optimisation, and to develop practical application of the algorithms in dynamic problem environments. Meanwhile, we have also seen developments in computer hardware, and in our ability to implement parallel versions of NC algorithms (e.g. using *graphics processing unit* (GPU) implementations). These two strands of development are creating an ever more powerful toolbox of NC algorithms for financial modellers.

References

1. Aiken, M. and Bsat, M. (1999). Forecasting market trends with neural networks. Information Systems Management, 16 (4): 42-48.
2. Aiken, M. (2000). Forecasting the United States gross domestic product with a neural network. Journal of International Information Management, 9 (1): 11-21.
3. Alexandrova-Kabadjova, B., Tsang, E. and Krause A. (2008). Evolutionary Learning of the Optimal Pricing Strategy in an Artificial Payment Card Market. In: Brabazon,A., O'Neill,M.(eds.) Natural Computing in Computational Finance, 233-251, Springer Berlin.
4. Alfaro-Cid, E., Cuesta-Canada, A., Sharman, K. and Esparcia-Alcazar, A. (2008). Stong typing, variable reduction and bloat control for solving the bankruptcy prediction problem using genetic programming. In: Brabazon,A., O'Neill,M.(eds.) Natural Computing in Computational Finance, 161-186, Springer Berlin.
5. Allen, F. and Karjalainen, R. (1999). Using genetic algorithms to find technical trading rules. Journal of Financial Economics, 51:245-271.
6. Altman, E. (1968). Financial ratios, discriminant analysis and the prediction of corporate bankruptcy. Journal of Finance, 23:589-609.
7. Armano, G, Marchesi, M. and Murru, A. (2005). A hybrid genetic-neural architecture for stock indexes forecasting. Information Sciences, 170: 3-33.
8. Arone, S., Loraschi, A. and Tettamanzi, A. (1993). A genetic approach to portfolio selection. Neural Network World, International Journal on Neural and Mass-Parallel Computing and Information Systems, 3:597-604.
9. Atiya, A. (2001). Bankruptcy prediction for credit risk using neural networks: a survey and new results. IEEE Trans. on Neural Networks, 12(4):929-935.
10. Bauer R. (1994). Genetic Algorithms and Investment Strategies, Wiley, New York.
11. Belkaoui, A. (1978). Financial ratios as predictors of Canadian takeovers. Journal of Business Finance & Accounting, 5(5): 93-108.
12. Black, F. and Scholes, M. (1973). The pricing of options and corporate liabilities. Journal of Political Economy, 81:637-659.
13. Brabazon, A. and O'Neill, M. (2006). Biologically Inspired Algorithms for Financial Modelling, Springer, Berlin.

14. Brabazon, A. and O'Neill, M. (eds.) (2008). Natural Computing in Computational Finance, Springer, Berlin.
15. Brabazon, A. and O'Neill, M. (eds.) (2009). Natural Computing in Computational Finance (Volume II), Springer, Berlin.
16. Castillo Tapia, M. G. and Coello Coello, C. (2007). Applications of multi-objective evolutionary algorithms in economics and finance: a survey. Proceedings of the IEEE International Conference on Evolutionary Computation (CEC 2007), 532-539, IEEE Press.
17. Cao, L.J., Tay, F.E.H.(2003). Support vector machine with adaptive parameters in financial time series forecasting. IEEE Trans. on Neural Networks, 14(6): 1506-1518.
18. Cao, L.J., Tay, F.E.H.(2003). A hybrid neurogenetic approach for stock forecasting. IEEE Trans. on Neural Networks, 18(3): 851-864.
19. Chen, S-H., Lee, W-C. and Yeh, C-H. (1999). Hedging derivative securities with genetic programming. International Journal of Intelligent Systems in Accounting, Finance and Management, 8(4):237-251.
20. Chen, S. H. and Yeh, C. H. (2001). Evolving traders and the business school with genetic programming: a new architecture of the agent-based stock market. Journal of Economic Dynamics and Control, 25 (3-4): 363-393.
21. Chen, S-H. (eds.) (2002). Genetic Algorithms and Genetic Programming in Computational Finance, Kluwer Academic Publishers.
22. Chen, S-H. and Kuo, T-W. (2002). Evolutionary computation in economics and finance: a bibliography. In: Chen, S-H. (eds.), Evolutionary Computation in Economics and Finance, Physica-Verlag.
23. Chidambaran, N., Lee, C. and Trigueros, J. (1998). Adapting Black-Scholes to a non-Black-Scholes environment via genetic programming. Proceedings of the IEEE/IAFE/INFORMS 1998 Conference on Computational Intelligence for Financial Engineering (CIFEr), 197-211, IEEE Press.
24. Chidambaran, N. (2003). Genetic programming with Monte Carlo simulation for option pricing. Proceedings of the 2003 IEEE Winter Simulation Conference, 285-292, IEEE Press.
25. Cristianini, N. and Shawe-Taylor, J. (2000). An Introduction to Support Vector Machines and Other Kernel-based Learning Methods. Cambridge University Press.
26. Coello, C. and Van Veldhuizen, D. and Lamont, G.(2002). Evolutionary Algorithms for Solving Multi-Objective Problems, Kluwer Academic Publishers.
27. da Costa Pereira, C. and Tettamanzi, A. (2008). Fuzzy-evolutionary modeling for single-position day trading. In: Brabazon, A. and O'Neill, M. (eds.) Natural Computing in Computational Finance, 131-159, Springer, Berlin.
28. Cox, J., Ross, S. and Rubinstein, M. (1979). Option Pricing: a simplified approach. Journal of Financial Economics, 7:229-264.
29. Damodaran, A. (2003). Investment Philosophies: Successful Investment Philosophies and the Greatest Investors Who Made Them Work, pp. 8, John Wiley and Sons.
30. Dang, J., Brabazon, A., O'Neill, M. and Edelman, D. (2008). Option model calibration using a bacterial foraging optimisation algorithm. Proceedings of the 2^{nd} European Workshop on Evolutionary Computation in Finance and Economics (EvoFin 2008), LNCS 4974, pp. 133-143, Springer.
31. de Castro, L.N. (2007). Fundamentals of natural computing: an overview. Physics of Life Reviews, 4(1):1-36.

32. Deb, K. (2001). Multi-objective Optimization using Evolutionary Algorithms, John Wiley and Sons.
33. Dempster, M. and Jones, C. (2001). A real-time adaptive trading system using genetic programming. Quantitative Finance, 1:397-413.
34. Dempsey, I., O'Neill, M. and Brabazon, A. (2009). Foundations in Grammatical Evolution for Dynamic Environments, Springer.
35. Diagne, M. (2002). Financial risk management and portfolio optimization using neural networks and extreme value theory, PhD thesis, Univ. of Kaiserslautern.
36. Drijver, S.(2005) Asset liability management for pension funds using multistage mixed-integer stochastic programming, PhD thesis, University of Groningen, The Netherlands.
37. Dopuch, N., Holthausen, R.W., and Leftwich R.W. (1987). Predicting audit qualifications with financial and market variables. The Accounting Review, LXII(3):431-454.
38. Dreżewski, R. and Siwik, L. (2008). Co-Evolutionary Multi-Agent System for Portfolio Optimization. In; Brabazon, A. and O'Neill, M. (eds.) Natural Computing in Computational Finance, 271-299, Springer, Berlin.
39. Dutta, S. and Shekhar, S. (1988). Bond rating: a non-conservative application of neural networks. Proceedings of IEEE International Conference on Neural Networks, II, 443-450.
40. Ecca, S., Marchesi, M and Setzu, A. (2008). Modeling and simulation of an artificial stock option market. Computational Economics, 32(1):37-53.
41. Edelman, D. and Davy, P. (2004). Adaptive technical analysis in the financial markets using machine learning: a statistical view. In: Fulcher, J. & Jain, L.C (eds.) Applied Intelligent Systems New Directions Series Studies in Fuzziness and Soft Computing.
42. Edelman, D. (2007). Adapting support vector machine methods for horserace odds prediction. Annals OR 151(1): 325-336.
43. Ederington, H. (1985). Classification models and bond ratings. Financial Review, 20(4):237-262.
44. Fabozzi, F.J. et al.(2007). Trends in quantitative equity management: survey results. Quantitative Finance, 7(2):115-122.
45. Fan, K., Brabazon, A., O'Sullivan, C. and O'Neill, M. (2007). Quantum-inspired evolutionary algorithms for calibration of the VG option pricing model. Proceedings of the 1st European Workshop on Evolutionary Computation in Finance and Economics (EvoFin 2007), LNCS 4447, 186-195, Springer.
46. Fieldsend, J., Matatko, J. and Peng, M. (2004). Cardinality constrained portfolio optimisation. Intelligent Data Engineering and Automated Learning (IDEAL 2004), LNCS 3177, 788-793, Springer.
47. Fyfe, C., Marney, J. and Tarbert, H. (1999). Technical analysis versus market efficiency - a genetic programming approach. Applied Financial Economics, 9(2):183-191.
48. Gentry, J., Whitford, D. and Newbold, P. (1988). Predicting industrial bond ratings with a probit model and funds flow components. Financial Review, 23(3):269-286.
49. Ghandar, A., Michalewicz, Z., Schmidt, M, Tô, T-D. and Zurbrugg, R. (2008). Computational Intelligence for Evolving Trading Rules. IEEE Trans. on Evolutionary Computation, 13(1):71-86..

50. Gode, D.K. and Sunder, S. (1993). Allocative efficiency of markets with zero intelligence traders. Journal of Political Economy, 101:119-37.
51. Harris, R., Stewart, J., Guilkey, D, & Carleton, W. (1982). Characteristics of acquired firms: fixed and random coefficients probit analyses. Southern Economic Journal, 49(1):164-184.
52. Hickey, R, Little, E. and Brabazon, A. (2006). Identifying merger and takeover targets using a self-organising map. Proceedings of the 2006 International Conference on Artificial Intelligence (ICAI '06), CSEA Press.
53. Hochreiter, R. (2008). Evolutionary stochastic portfolio optimization. In: Brabazon,A., O'Neill,M.(eds.) Natural Computing in Computational Finance, 67-87, Springer Berlin.
54. Hutchinson, J., Lo, A. and Poggio, T. (1994). A non-parametric approach to pricing and hedging derivative securities via learning networks. Journal of Finance, 851-889.
55. Izumi, K.(1999). An artificial market model of a foreign exchange market, PhD Dissertation, Tokyo University.
56. Kaastra, I. and Boyd, M. (1996). Designing a neural network for forecasting financial and economic time series. Neurocomputing, 10:215-236.
57. Kari, L. and Rozenberg, G. (2008). The many facets of natural computing. Communications of the ACM, 51(10):72-83.
58. Keber, C. (2000). Option Valuation with the Genetic Programming Approach. Proceedings of the sixth international conference, 689-703, Cambridge, MA: MIT Press.
59. Keber, C. (2002a). Evolutionary computation in option pricing: determining implied volatilities based on American put options. In: Chen, S-H. (eds.) Evolutionary Computation in Economics and Finance, 399-415, Physica-Verlag.
60. Keber, C. and Schuster, M. (2001). Evolutionary computation and the vega risk of American put options. IEEE Trans. on Neural Networks. 12(4): 704-715, Physica-Verlag.
61. Kim, J. and Byun, S. (2005). A parallel Monte Carlo simulation on cluster systems for financial derivatives pricing. Proceedings of the IEEE International Conference on Evolutionary Computation (CEC 2005), 1040-1044, IEEE Press.
62. Kiviluoto, K. and Bergius, P. (1998). Maps for analysing failures of small and medium-sized enterprises. In: Deboeck, G. and Kohonen, T. (eds.) Visual Explorations in Finance with self-organizing maps, 59-71, Springer-Verlag Berlin.
63. Kumar, N., Krovi, R. and Rajagopalan, B. (1997). Financial decision support with hybrid genetic and neural based modeling tools. European Journal of Operational Research, 103(2):339-349.
64. Kwon, Y-K. and Moon, B-R. (2004). Evolutionary ensemble for stock prediction. Proceedings of the Genetic and Evolutionary Computation Conference (GECCO 2004), LNCS 3103, 1102-1113, Springer.
65. Larkin, F. and Ryan C. (2008). Good news: using news feeds with genetic programming to predict stock prices. In: O'Neill, M. et al (eds.), Proceedings of the 11^{th} European Conference on Genetic Programming (EuroGP 2008), pp. 49-60, LNCS 4971, Springer.
66. LeBaron, B. (2002). Building the Santa Fe artificial stock market, Working Paper, Brandeis University, Jun. 2002.

67. Lee, H., Lee, J., Yoon, Y. and Kim, S. (2005) Coherent risk measure using feedfoward neural networks. ISNN 2005, LNCS 3497, 904-909, Springer Berlin.
68. Lee, C, Quek, C and Maskell, D. (2006). A brain-inspired fuzzy neuro-predictor for bank failure analysis. Proceedings of the IEEE International Conference on Evolutionary Computation (CEC 2006), 7927-7934, IEEE Press.
69. Leung, K., Cheong, F. and Cheong, C. (2007). Consumer credit scoring using an artificial immune system algorithm. Proceedings of the IEEE International Conference on Evolutionary Computation (CEC 2007), 3377-3384, IEEE Press.
70. Li, J. (2001). FGP: a genetic programming based tool for financial forecasting. PhD Thesis, University of Essex, UK.
71. Lim, M. and Coggins, R. (2005). Optimal trade execution: an evolutionary approach. Proceedings of the IEEE International Conference on Evolutionary Computation (CEC 2005), 1045-1052, IEEE Press.
72. Loraschi, A. and Tettamanzi, A., Tomassini, M. and Verda, P. (1995). Distributed genetic algorithms with an application to portfolio selection problems. In: Pearson,D., Steele, N. and Albrecht, R. (eds), Artificial Neural Networks and Genetic Algorithms, pp. 384-387, Springer.
73. Lipinski, P. (2008). Evolutionary strategies for building risk-optimal portfolios. In: Brabazon,A., O'Neill,M.(eds.) Natural Computing in Computational Finance, 53-65, Springer Berlin.
74. Malliaris, M. and Salchenberger, L. (1993). A neural network model for estimating option prices. Applied Intelligence, 3(3):193-206.
75. Maringer, D. (2008). Constrained index tracking under loss aversion using differential evolution. In: Brabazon,A., O'Neill,M.(eds.) Natural Computing in Computational Finance, 7-24, Springer Berlin.
76. Markose, S., Tsang, E. and Er, H. (2002). Evolutionary decision trees for stock index options and futures arbitrage. In: Chen, S-H.(eds), Genetic Algorithms and Genetic Programming in Computational Finance, 281-308, Kluwer Academic Publishers.
77. Markowitz, H. (1952). Portfolio selection. Journal of Finance, 1(7):77-91.
78. Markowitz, H. (1959). Portfolio Selection: Efficient Diversification of Investments, John Wiley and Sons.
79. Marrison, C. (2002) The Fundamentals of Risk Measurement. New York: McGraw Hill.
80. McKee, T. and Lensberg, T. (2002). Genetic programming and rough sets: a hybrid approach to bankruptcy classification. European Journal of Operational Research, 138:436-451.
81. Merton, R. (1973). Rational theory of option pricing. Bell Journal of Economics and Management Science, 4:141-183.
82. Moral-Escudero, R., Ruiz-Torrubiano, R. and Suarez, A. (2006). Selection of optimal investment portfolios with cardinality constraints. Proceedings of the IEEE International Conference on Evolutionary Computation (CEC 2006), 8551-8557, IEEE Press.
83. Mutchler, J.F. (1985). A multivariate analysis of the auditor's going-concern opinion decision, *Journal of Accounting Research*, 23(2):668-682.

84. Neely, C., Weller P. and Dittmar, R. (1997). Is technical analysis in the foreign exchange market profitable? A genetic programming approach. Journal of Financial and Quantitative Analysis, 32(4):405-428.
85. Neely, C. and Weller P. (2002). Using a genetic program to predict exchange rate volatility. In: Chen S-H. (eds.), Genetic Algorithms and Genetic Programming in Computational Finance, 263-278, Kluwer Academic Publishers.
86. Noe, T. and Wang, J. (2002). The Self-evolving Logic of Financial claim prices. In: Chen S-H.(eds.), Genetic Algorithms and Genetic Programming in Computational Finance, 249-262, Kluwer Academic Publishers.
87. Ohlson, J. (1980). Financial ratios and the probabilistic prediction of bankruptcy. Journal of Accounting Research, 18(1):109-131.
88. O'Neill, M. and Ryan, C. (2001) Grammatical Evolution, IEEE Trans. Evolutionary Computation, 5(4):349-358.
89. O'Neill, M., and Ryan, C. (2003). Grammatical Evolution: Evolutionary Automatic Programming in an Arbitrary Language, Kluwer Academic Publishers.
90. Orito, Y., Takeda, M., Iimura, K. and Yamazaki, G. (2007). Evaluating the efficiency if index fund selections over the fund's future period. In: Chen, S-H, Wang, P., Kuo, T-W. (eds.) Computational Intelligence in Economics and Finance, 157-168, Springer.
91. Palepu, K. (1986). Predicting takeover targets: a methodological and empirical analysis. Journal of Accounting and Economics, 8:3-25.
92. Pavlidis, N., Pavlidis, E., Epitropakis, M., Plagianakos, V. and Vrahatis, M. (2007). Computational intelligence algorithms for risk-adjusted trading strategies. Proceedings of the 2007 Congress on Evolutionary Computation (CEC 2007), 540-547, IEEE Press.
93. Quintana, D., Luque, C. and Isasi, P. (2005). Evolutionary rule-based system for IPO underpricing prediction. Proceedings of the Genetic and Evolutionary Computation Conference (GECCO 2005), 983-989, ACM.
94. Rege, U., (1984). Accounting ratios to locate take-over targets. Journal of Business Finance and Accounting, 11(3):301-311.
95. Schaffer, J. (1984). Multiple objective optimization with vector evaluated genetic algorithms, PhD Thesis, Vanderbilt University, Nashville, TN.
96. Schlottmann, F. and Seese, D. (2004). Financial applications of multi-objective evolutionary algorithms: recent developments and future research directions. In: C. Coelle Coello and G. Lamont (eds.), Applications of Multi-Objective Evolutionary Algorithms, 627-652, Singapore, World Scientific.
97. Serrano-Cina, C. (1996). Self organizing neural networks for financial diagnosis. Decision Support Systems, 17(3):227-238.
98. Shapcott, J. (1992). Index tracking: genetic algorithms for investment portfolio selection. Technical report, EPCC-SS92-24, Edinburgh Parallel Computing Centre.
99. Shoaf J. and Foster, J. (1998). The efficient set GA for stock portfolios. Proceedings of the IEEE International Conference on Evolutionary Computation (CEC 98), 354-359, IEEE Press.
100. Sound Practices for Hedge Fund Managers (2000), the Managed Funds Association (MFA), pp. 16, New York.
101. Stanley, K. and Miikkulainen, R. (2002). Evolving neural networks through augmenting topologies. Evolutionary Computation, 10(2):99-127.

102. Stephens, C. R. and Sukumar R. (2006). An Introduction to Datamining. In: Grover, R. and Vriens, M. (eds.), The Handbook of Market Research Do's and Don'ts, Sage Publications.
103. Streichert, F., Ulmer, H. and Zell, A. (2004a). Evaluating a hybrid encoding and three crossover operators on the constrained portfolio selection problem. Proceedings of the IEEE International Conference on Evolutionary Computation (CEC 2004), 932-939, IEEE Press.
104. Streichert, F., Ulmer, H. and Zell, A. (2004b). Comparing discrete and continuous genotypes on the constrained portfolio selection problem. Proceedings of the Genetic and Evolutionary Computation Conference (GECCO 2004), 1239-1250, Springer.
105. Streichert, F. and Tanaka-Yamawaki, M. (2006). The effect of local search on the constrained portfolio selection problem. Proceedings of the IEEE International Conference on Evolutionary Computation (CEC 2006), 8537-8543, IEEE Press.
106. Subbu, R., Bonissone, P., Eklund, N., Bollapragada, S. and Chalermkraivuth, K. (2005). Multiobjective financial portfolio design: a hybrid evolutionary approach. Proceedings of the IEEE International Conference on Evolutionary Computation (CEC 2005), 2429-2436, IEEE Press.
107. Thomas, J. and Sycara, K. (2002). GP and the predictive power of internet message traffic. In: Chen S-H. (eds.), Genetic Algorithms and Genetic Programming in Computational Finance, 81-102, Kluwer Academic Publishers.
108. Thompson, D., Thompson, S. and Brabazon, A. (2007). Predicting going concern audit qualification using neural networks. Proceedings of the 2007 International Conference on Artificial Intelligence (ICAI '07), CSEA Press
109. Trigueros, J. (1999). Extracting earnings information from financial statements via genetic algorithms. Proceedings of the 1999 IEEE International Conference on Computational Intelligence for Financial Engineering, pp. 281-296, IEEE Press.
110. Trippi, R.R. and Turban, E. (1993). Neural Networks in Finance and Investing.
111. Tsang, E. and Li, J. (2002). EDDIE for Financial Forecasting. In: Chen, S-H. (eds.), Genetic Algorithms and Genetic Programming in Computational Finance, 161-174, Kluwer Academic Publishers.
112. Tsang, E. and Martinez-Jaramillo, S. (2004). Computational finance. IEEE Computational Intelligence Society Newsletter, 8-13, August 2004.
113. Tung, W. and Quek, C. (2005). GenSoOPATS: a brain-inspired dynamically evolving option pricing model and arbitrage system. Proceedings of the IEEE International Conference on Evolutionary Computation (CEC 2005), 1722-1729, IEEE Press.
114. Uludag, et al (2007). Comparison of evolutionary techniques for value-at-risk calculation. Proceedings of EvoWorkshops 2007, LNCS 4448, pp.218-227.
115. Varetto, F. (1998). Genetic algorithms in the analysis of insolvency risk. Journal of Banking and Finance, 22(10):1421-1439.
116. Vedarajan, G., Chan, L. and Goldberg, D. (1997). Investment portfolio optimization using genetic algorithms. In: Koza, J. (eds.), Late Breaking Papers at the Genetic Programming 1997, 256-263.
117. Wang, C., Zhao, X. and Kang, Li (2004). Business failure prediction using modified ants algorithm. In: Chen, S-H., Wang, P., Kuo, T-W. (eds.), Computational Intelligence in Economics and Finance, Springer.

118. West, D. (2000). Neural network credit scoring models. Computers and Operations Research, 27:1131-1152.
119. White, A. (1998). A genetic adaptive neural network approach to pricing options: a simulation analysis. Journal of Computational Intelligence in Finance, 6(2):13-23.
120. Wilson, N., Chong, K. and Peel, M. (1995). Neural network simulation and the prediction of corporate outcomes: some empirical findings. International Journal of the Economics of Business, 2(1):31-50.
121. Wong, B.K., Selvi, Y. (1998). Neural network applications in finance: a review and analysis of literature. Information and Management, 34(3):129C140.
122. Wong, B., Lai, V. and Lam, J. (2000). A bibliography of neural network business applications research: 1994-1998. Computers and Operations Research, 27:1045-1076.
123. Yin, Z., Brabazon, A. and O'Sullivan, C. (2007). Adaptive genetic programming for option pricing. Proceedings of the Genetic and Evolutionary Computation Conference (GECCO 2007), 2588-2594, ACM Press.
124. Yobas, M, Crook, J. and Ross, P. (2000). Credit scoring using neural and evolutionary techniques. IMA Journal of Mathematics Applied in Business and Industry, 11:111-125.
125. Zaiyi, G., Quek, C. and Maskell, D. (2006). FCMAC-AARS: a novel FNN architecture for stock market prediction and trading. Proceedings of the IEEE International Conference on Evolutionary Computation (CEC 2006), 8544-8550, IEEE Press.
126. Zenios, S.A. (2008). Practical financial optimization, Wiley-Blackwell, 2008.
127. Vapnik, V. (1995). The Nature of Statistical Learning Theory. Springer-Verlag.