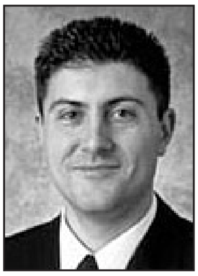
Financial survival of the fittest

Using genetic algorithms for investment decision-making

Genetic algorithms are a class of advanced computational methods that are extremely useful for solving complex optimisation problems. ROBERT PEREIRA explains how these algorithms work and suggests potential applications to financial market trading and investment management.



ROBERT PEREIRA SIA (Aff) is a quantitative analyst at Merrill Lynch Investment Managers. This paper is based on his PhD thesis on genetic algorithms applied to financial market trading rules, which was under-taken at the University of Melbourne.

Artificial intelligence or machine-based learning techniques, such as neural networks, genetic algorithms and fuzzy logic, are achieving more acceptance because of significant advances in information technology. Until recently, these computer algorithms were considered too time-consuming and computationally hungry to be practical. During the past decade, however, a number of financial institutions have successfully applied these methods to the solving of problems in banking, finance and investment (see Deboeck 1994).

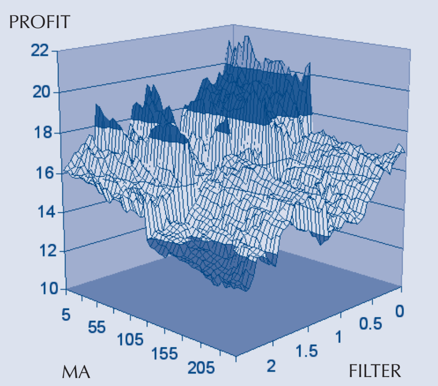
Computers are of crucial importance in pricing securities, developing trading strategies, financial analysis and portfolio optimisation because of the increasing complexity of these and other issues in finance. This complexity arises for three main reasons:

- financial market data is multidimensional and typically exhibits non-linearity, non-normality and non-stationarity;
- some financial optimisation problems have non-differentiable objective functions and are characterised by multiple optima; and
- an increasing number of problems involve very large amounts of data.

Standard numerical optimisation based on derivatives, for example Newton's methods, are ineffective since they typically get stuck at sub-optimal local solutions. Thus, there is a need for a method that is immune to these difficulties and able to deliver better results. Robust techniques, such as genetic algorithms, simulated annealing and random adaptive search, are required.

The importance of robustness can be appreciated by considering the problem of searching for the optimal parameters for a trading rule applied to financial markets. In general, this sort of problem does not have a differentiable objective function. Further, the data sets used for optimisation can be very large, especially if tick data are used, and tend to display the features mentioned above. This is illustrated in Figure 1, which displays the profit surface for a trading rule employing a moving average (MA) with a filter applied to the Australian sharemarket. The profit surface for this example is

FIGURE 1 Profit surface for a trading rule



characterised by multiple optima, non-linearity and non-stationarity.

Relative to computers, humans have limited computational and cognitive ability. In addition, Kahneman and Tversky (1982) have shown that humans tend to have certain biases that can result in inconsistent decision-making. But even computers can be severely limited when faced with problems that have a seemingly limitless number of different combinations or possibilities. Numerous problems in finance — for example, searching through many possible trading rules — have an extremely large number of possible solutions that require a powerful search technique. Genetic algorithms are not only robust, but also allow for the rapid evaluation of many different possibilities.

Genetic algorithms

Genetic algorithms, first proposed by Holland (1975), are biologically inspired optimisation techniques that are both extremely effective and efficient at searching large solution spaces. By using genetic operations they are able to simulate an evolutionary process by which an initial population of potential solutions to a difficult problem can develop through successive generations into optimal or near-optimal solutions.

In the standard algorithm, potential or candidate solutions to a problem are represented using vectors consisting of binary digits. An initial set (referred to as a population) of candidates is created randomly. This initial population can be evolved into a population consisting of better solutions through the genetic operations of crossover and mutation. These operations are analogous to those that act on the chromosomes in living organisms. Crossover ensures that better-performing candidates are evolved over time.

In general (as in the theory of natural selection or survival of the fittest), better-performing candidates have a higher probability of surviving and reproducing than the poorer-performing candidates, which eventually are eliminated from the population. The performance of each candidate is assessed according to an appropriate objective function. A selection process based on performance is applied to determine which of the candidates should participate in crossover, and thereby pass on their favourable traits to future generations. Diversity in the population is maintained through mutation, which introduces randomness into the search process. Similar to what happens in nature, it is important to maintain a diverse population that is able to adapt to a changing environment. This evolutionary process continues until the best (or better) performing individual(s), consisting of the optimal or near-optimal solutions, dominate the population. This process can be summarised in a number of steps (see Table 1).

Computational aspects

The three main operations that drive the evolutionary process of the genetic algorithm are selection, crossover and mutation. Selection determines which candidates are to participate in crossover and undergo possible mutation. The original method of crossover, known as roulette wheel selection, involves selecting candidates according to their fitness or performance. Each candidate is allocated a slice of the roulette wheel, with its size determined by relative fitness. Therefore candidates with a higher level of fitness have a correspondingly greater chance of being selected. For example, in Figure 2, candidate 1 is fitter than candidate 2, which in turn is fitter than candidate 3.

Pairs of candidates are selected for crossover using this roulette wheel method. To illustrate the process of crossover, assume that the two candidates A and B have been chosen for crossover. Crossover is performed by randomly partitioning or breaking two vectors A and B at a particular point — assume for the purpose of this example that this occurs between the second and third elements of each vector. Two new candidates, represented by vectors C and D, can be created by exchanging the binary digits to the left of the break point between the two vectors A and B. Crossover can be represented mathematically as

$$A = [\,1\ 0 : 1\ 0\ 0\,] \quad \& \quad B = [\,0\ 1 : 0\ 1\ 0\,] \quad \rightarrow$$
$$C = [\,1\ 0 : 0\ 1\ 0\,] \quad \& \quad D = [\,0\ 1 : 1\ 0\ 0\,].$$

At the completion of crossover, the new candidates replace the old candidates and are subjected to possible mutation. Through mutation, new genetic material can be introduced into the population. This increases the diversity in the population and, unlike crossover, randomly redirects the search procedure into new areas of the solution space, which may or may not be beneficial. This action underpins the genetic algorithm's ability to find novel or

FIGURE 2 Roulette wheel selection

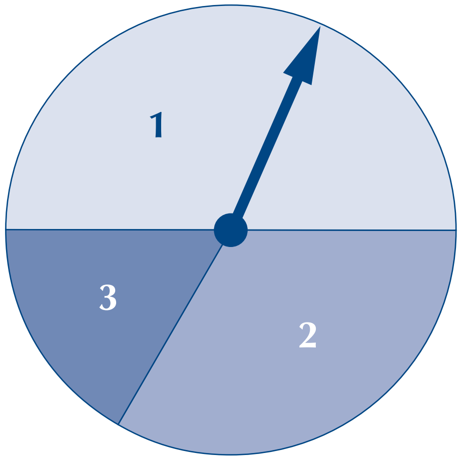


TABLE 1 The genetic algorithm process

Steps	Action
1	Represent potential solutions in terms of their binary representations
2	Create an initial population of candidates randomly
3	Calculate the performance of each candidate in the initial population
4	Perform selection to determine the restricted population
5	Apply crossover to the restricted population
6	Apply mutation to the restricted population
7	Calculate the performance of the candidates in the new generation
8	Return to Step 3 unless a termination criterion is satisfied

inconspicuous solutions and to avoid getting anchored at local sub-optimal solutions. Typically mutation occurs with low probability, in order not to unduly disrupt the search process. Mutation occurs by switching a binary digit from either a one to a zero or vice versa. To illustrate mutation, assume that for candidate C no element other than the third undergoes mutation.

B = [1 0 0 1 0] $\rightarrow$ C = [1 0 1 1 0]

Advantages and limitations

The main advantages are robustness in the presence of multiple optima and efficiency relative to other robust approaches such as simulated annealing. The efficiency of genetic algorithms relative to simulated annealing and other similar techniques is due to the genetic algorithm employing a multi-directed search process. This is achieved through maintaining a population of potential solutions, unlike simulated annealing, which concentrates its search on one particular point in the search space. A multi-directed search process is a form of parallel processing found in supercomputers and also employed by neural networks.

The main limitations of genetic algorithms are premature convergence and failure to converge in heavily constrained or highly nonlinear problems. Thus, genetic algorithms are classified as a weak optimisation technique, in the sense that convergence is not guaranteed. However, this might not be of crucial importance when the objective is to find profitable trading rules quickly rather than the most profitable rule, which may come at the cost of a greater amount of time necessary to obtain the global optimum. Dorsey and Mayer (1995) have shown that genetic algorithms might find only near-optimal solutions rather than the true global optimum. But typically, these solutions are closer to the global optimum compared with other approaches such as indirect methods, if these methods are applicable. This is because even though convergence is guaranteed for indirect methods, in complex problems indirect methods have a higher probability of converging to local optima than genetic algorithms or other direct methods.

Although not much can be done about failure of convergence, especially when the nature of the problem is the cause, premature convergence can be some what avoided by slight modifications to the standard genetic algorithm. One of the more common modifications is the introduction of a form of elitism. This ensures that the best or highest-performing candidate over the entire genetic algorithm trial is always maintained in the current population. As a result, the performance of the best candidate is strictly non-decreasing over successive iterations. However, if the number of elite candidates is set too high, this could lead to deterioration in the exploration of the search space.

FINANCIAL APPLICATIONS

Genetic algorithms are a valid approach to many practical problems in finance that are complex and thus require the use of an efficient and robust optimisation technique. There are many examples of genetic algorithm applications to investment and financial markets. Some of these are discussed below. There have also been applications of genetic algorithms in banking — mainly in credit evaluation and fraud detection.

Stock selection

Genetic algorithms can be used in deciding which securities to buy and which to short-sell. Advanced Investment Technology, an investment management firm in Florida, uses genetic algorithms and neural networks to predict the relative returns for individual shares. Empirical testing reveals that the genetic algorithms and neural networks demonstrate significant forecasting skill, compared with linear methods, in identifying which shares will outperform or underperform the market.

Asset allocation

Some fund managers attempt to generate added value to their portfolios through tactical asset allocation (TAA). This involves shifting allocations between the different asset classes of shares, bonds, property and cash. First Quadrant, an investment management firm in California, has successfully applied genetic algorithms to their domestic and global TAA models, as well as their tactical currency management models.

Portfolio construction

An important function in investment management is to determine the appropriate weights to give individual securities in a portfolio. This process is typically referred to as portfolio construction. An asset adviser in Milan has applied a genetic algorithm to find the optimal weights for a portfolio of securities by minimising downside risk.[1] This approach has been found useful when dealing with a large solution space characterised by multiple optima.

Efficient indexing

Given the recent popularity of index funds, cost-effective index replication based on an optimised sampling approach has become an important issue for passive managers. This problem is especially relevant for managers attempting to match the performance of a broad-based index consisting of thousands of securities; for example, the Russell 3000, Wilshire 5000 and the various MSCI indexes. A German consultant has developed a genetic algorithm application for tracking a market index effectively using the smallest possible subset of stocks from the index.

Trading rule discovery

Genetic algorithms have also been used to discover profitable trading rules. Bauer (1994) uses a genetic algorithm to develop market-timing trading rules for the US share and bond markets. These rules are formulated using macroeconomic data to uncover relationships between financial markets and the economy. This methodology can also be extended to the foreign-exchange market.

Genetic programming has been used by Citibank to discover profitable rules for trading the foreign exchange market. This technique is similar to a genetic algorithm but is less restrictive since it does not fix the length of the binary representation of the solutions. The idea is to represent trading rules using binary tree classifications and then evolve these rules into more profitable rules.

Optimisation of trading rules

The application of trading rules based on technical indicators to financial market trading requires the selection of appropriate parameter values. In practice, traders usually choose these parameters in a subjective process largely based on intuition and experience. Also, numerous studies examining technical trading-rule profitability have ignored the issue of parameter optimisation or have used parameter values determined *ex post*. This practice can lead to a data-snooping bias and also possibly introduce a subtle form of survivorship bias into the performance study.

A more objective and valid approach to the problem of parameter selection involves the use of historical data. In order to conduct a valid evaluation of technical trading rule performance free from data-snooping bias, it is necessary to choose parameter values *ex ante*. This can be achieved by using an in-sample period to determine the optimal *ex ante* parameter values. The performance of these optimal rules can then be evaluated out-of-sample. A genetic algorithm can be used to select the parameter values for technical trading rules. This optimisation method is relevant to this type of problem because of its robustness in the presence of multiple equilibria and non-linearity of the profit surface and because of its efficiency in searching very large parameter spaces.

NOTE

1 Downside risk is measured by calculating the variance or standard deviation of returns that fail to meet a specified level.

REFERENCES

Bauer, R.J. Jr., 1994, *Genetic Algorithms and Investment Strategies*, Wiley Finance Editions, John Wiley and Sons, New York.

Deboeck, G.J., 1994, Trading On the Edge: *Neural, Genetic, and Fuzzy Systems for Chaotic Financial Markets*, John Wiley and Sons, New York.

Dorsey, R.E., and W.J. Mayer, 1995, "Genetic Algorithms for Estimation Problems with Multiple Optima, Nondifferentiability, and Other Irregular Features", *Journal of Business and Economic Statistics*, 13 (1), pp. 53-66.

Holland, J.H., 1975, *Adaptation in Natural and Artificial Systems*, University of Michigan Press, Ann Arbor, Michigan.

Kahneman, D., and A. Tversky, 1982, "Intuitive Prediction: Biases and Corrective Procedures", in D. Kahneman, P. Slovic and A. Tversky (eds.), *Judgement Under Uncertainty: Heuristics and Biases*, Cambridge University Press, New York, pp. 414-21.