
Evolving Trading Rules

Adam Ghandar¹, Zbigniew Michalewicz², Martin Schmidt³, Thuy-Duong Tô⁴, and Ralf Zurbrugg⁵

¹ School of Computer Science, University of Adelaide, Adelaide, SA 5005, Australia, adam.ghandar@adelaide.edu.au

² School of Computer Science, University of Adelaide, Adelaide, SA 5005, Australia; also at Institute of Computer Science, Polish Academy of Sciences, ul. Ordonia 21, 01-237 Warsaw, and Polish-Japanese Institute of Information Technology, ul. Koszykowa 86, 02-008 Warsaw, Poland, zbyszek@cs.adelaide.edu.au

³ SolveIT Software Pty Ltd, Level 12, 90 Kings William, Adelaide 5000, Australia, martin.schmidt@solveitsoftware.com

⁴ School of Commerce, University of Adelaide, Adelaide, SA 5005, Australia, td.to@adelaide.edu.au

⁵ School of Commerce, University of Adelaide, Adelaide, SA 5005, Australia, ralf.zurbrugg@adelaide.edu.au

1 Overview

This chapter describes a computational intelligence system for learning trading rules and provides a comparison of the relative performance of portfolios managed by the system in the Australian Stock Exchange. Using only price and volume data the system determines rules to buy and sell stocks periodically. The rules adapt as market conditions change over time.

The system combines portfolio construction and management activities to build and manage a portfolio of stocks over time. The integrated process for stock selection and portfolio management allows for a search specification resulting in a highly adaptive and dynamic rule base system. Solutions are represented using fuzzy logic rule bases and a search for the best rule bases is implemented using an evolutionary process. The system produces rule bases that are adapted to market conditions using feedback from performance by adapting the length of the training data window and the action of a repair operator used in the evolutionary search.

The result section describes the application of portfolio evaluation tools to give a detailed assessment of the system's performance. A portfolio managed by the system is compared to the ASX200 index and an alternative portfolio constructed using a non-adaptive rule base.

The rest of this chapter is organized as follows: Section 2 provides background information. Section 3 explains the approach to constructing the com-

putational intelligence system in detail. Section 4 gives an evaluation of the systems performance and describes the methods for financial portfolio performance evaluation. Section 5 provides concluding remarks.

2 Background

There are many papers which describe various applications of nature-inspired algorithms to financial modeling; in this section we survey some of this work.

One of the possible financial applications has been in the area of developing trading rules to signal when investors should buy or sell various financial instruments. Research in this area has received greater attention over recent years as an appreciation for the ease by which computational algorithms can develop and evolve complex trading strategies are further realized. Research such as described in [13, 1] highlight the possibilities for evolutionary computation to provide trading strategies, based on pattern recognition, to profit from equity market trading. Published research in academic finance journals primarily focuses on examining how well genetic algorithms can develop specific trading rules using historical prices to test, *ex post*, their profitability.

This type of research is also directly related to the study of market efficiency. In an efficient capital market it would not be possible for traders to make a profit from past data as all relevant information for pricing a security today would be incorporated in today's price. Therefore, many finance papers (see [12], [14], and [8] for example) inter-relate the issue of market efficiency with the ability for genetic algorithms to literally "beat the market". Results are somewhat mixed. Although there is general consensus that financial markets do sometimes exhibit periods where certain trading rules work (see [6]), it is hard to find clear evidence that a single trading rule can function over an extended period of time. This is probably due to the fact that financial markets are ever-evolving, and in fact given the number of technical analysts that are employed in all the major financial trading institutions, when a trading rule is found to work it would not take long before it is exploited until it no longer yields a significant profit. It is therefore possibly more interesting to see if trading rules can be constructed that also continually evolve as the markets change. An adaptive trading strategy seems to be more promising than static approaches.

As well as genetic algorithms other nature-inspired search techniques have been applied to financial problems. Artificial neural networks have attracted a lot of interest over the past decade. A selection includes: [21] presents an index forecasting approach; [9] applies an ANN to currency exchange rate prediction by anticipating the direction of price change using signal processing methods for series with high noise and small sample sizes; [4] describes a neural evolutionary approach to find models of correlation between financial derivatives; [3] discusses assessing credit risk and predicting using ANNs; and [2] discusses a neural network for option pricing. More recently, numerous applications of

evolutionary computation have been published: [15] describes a dynamic stock selection system in which a model optimized using evolutionary computation determines optimal portfolio weights given trade recommendations; a genetic programming approach for combining trading rules in autonomous agents so that the rules compliment each other is given in [18]; [19] presents a linear genetic programming system for trading that uses intraday data; grammatical evolution for evolving human readable trading rules is extensively discussed in [5]; finally an application of genetic programming for discovering trading rules that are applicable in the short term is given in [16]. A significant benefit of genetic programming in expressing trading rules is the grammatical structure of phenotypes which enables expression of rules combining several input in a form that is able to be readily understood and applied, a feature shared with the fuzzy rule representation .

The system described in this chapter forms trading rules using price and volume history of stock prices and adapts the rules to changing market conditions. The approach is essentially referred to as technical analysis — rather than using fundamental accounting and macroeconomic data to determine which stocks to buy or sell, trading rules are developed solely applying historical data series from the previous trades of these stocks. In particular, moving average and volume indicators are employed for this purpose.

3 System Design

This section describes the design of the computational intelligence system. The system produces and applies trading rules taking into account evolving market conditions. The trading rules are represented using fuzzy logic rule bases and an evolutionary process facilitates the search for the best fuzzy logic rule bases with respect to the training data. In the following subsections we describe the the fuzzy rules, the linguistic variables used to construct these rules, the evolutionary process, the evaluation function and methods to implement adaptability.

3.1 Fuzzy Rule Base Representation

Fuzzy representation enables intuitive natural language interpretation of trading signals and implies a search space of possible rules that corresponds to trading rules a human trader could construct. An example of a typical technical trading rule such as “buy when the price of a stock X ’s price becomes higher than the single moving average of the stock X ’s price for the last, say, 20 days” (indicating a possible upward trend) could be encoded using a fuzzy logic rule such as “If *Single Moving Average Buy Signal* is *High* then *rating* is 1”; conversely we could have a trading rule such as “sell stocks with high price movement when the portfolio value is relatively low” encoded by a fuzzy rule:

“If *Price Change* is *High* and *Portfolio Value* is *Extremely Low* then *rating* is 0.1”.

Each fuzzy rule base consists of a set of “if - then” rules where the “If” part specifies properties of technical indicators and the “then” part specifies a rating with 10 discrete levels given a stock with these properties. The rule inputs are termed linguistic variables in the fuzzy logic component. Clearly, at least one linguistic variable must be defined to construct rules. Detailed definitions of the linguistic variables used in the system are given below. To construct the rules used to obtain the results presented in this chapter we used $V = 34$. The construction of the linguistic variables is described in section 4. The output is interpreted as a rating of the strength of a buy recommendation given fulfillment of the If part. It is possible for the If part of a rule to refer to any combination of the technical indicators the system uses to give one output rating. A rule base may contain at least one and no more than $O = 30$ rules.

The value of each linguistic variable is described by one of a possible seven fuzzy membership sets. These are defined describing the relative magnitude of a particular observation: *Extremely Low* (*EL*), *Very Low* (*VL*), *Low* (*L*), *Medium* (*M*), *High* (*H*), *Very High* (*VH*), and *Extremely High* (*EH*). Membership functions map crisp data observations to degrees of membership of these fuzzy sets.

Figure 1 shows a visualization of the membership functions for a linguistic variable. For the triangular membership functions, the mapping from an observation to a degree of membership for each membership function is fully defined by specifying a minimum, center and maximum value where the min and max values refer to the lowest and highest linguistic variable observations at the edges of the triangle that belong to the membership set to the least degree and the center belongs to the membership set to the highest degree (the top of the triangle). The ramp membership functions are specified by a minimum and maximum value, any values greater than or equal to the maximum are classified as having full membership.

To construct the membership functions for each linguistic variable a series of historical data observations (of the variable) is sorted from lowest to highest and duplicates are removed. This series is then divided into M ordered sets of equal size where each set corresponds to a membership set, the sets overlap. The extreme low and high membership sets take as their maximum the lowest and highest data observations of the sets that contain the lowest and highest observations. For the middle membership functions the lowest and highest members of each set are the minimum and maximum values that belong to the set and the center is found by taking the mean of all the observations that belong to the set. When new data is input to the system this procedure is repeated.

Any “if” part may include up to V linguistic variables. The output for each rule gives one of B different ratings; there can be up to $O = 30$ rules in each rule base.

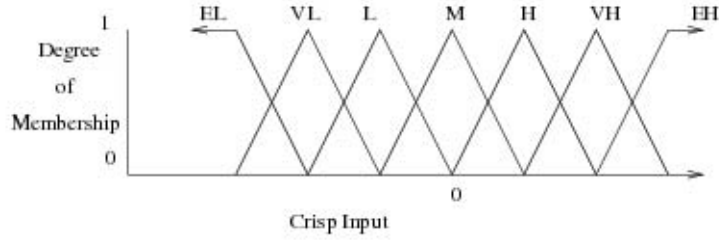


Fig. 1. Membership functions of the Fuzzy Sets *Extremely Low*, ... *Extremely High* for a linguistic variable. The middle membership functions (*Very Low* to *Very High*) are triangular and the two extreme membership functions are ramp membership functions. Note that the size of each function with respect to the crisp input that is mapped to a positive membership depends on the distribution of the data observations.

To estimate the total number of possible phenotypes, we may first estimate the number of possible phenotypes which consist of a single rule. Note that if a single rule has one linguistic variable present, then there are $M \times V \times B$ such phenotypes. If a single rule has two linguistic variables present, then there are $M^2 \times V \times B$ such phenotypes (M possible combinations of 2 linguistic variables out of available V , M^2 possible values for a pair of linguistic variables, B possible outcomes). In general, the number of possible phenotypes for a single rule (with one, two, ..., V linguistic variables present) is:

$$p = 10 \times \sum_{i=1}^V 7^i \times \binom{V}{i}.$$

The total number of phenotypes with k rules can be estimated as p^k .

An example of a phenotype rule base that could be produced by the system is given below. It consists of three rules:

- If *Single Moving Average Buy Signal* is *Extremely Low* then *rating* = 0.9.
- If *Price Change* is *High* and *Double Moving Average Sell* is *Very High* then *rating* = 0.4.
- If *On Balance Volume Indicator* is *Extremely High* and *Single Moving Average Buy Signal* is *Medium* and *Portfolio Value* is *Medium* then *rating* = 0.5.

Internally, each rule is represented using a sequence of slots. The columns in figure 2 have the following meanings. Column 1 contains a Boolean value to indicate whether the rule is active. Columns 2 through to 10 represent the rule inputs (each corresponds to a linguistic variable) and contain (a) a Boolean value indicating whether or not the linguistic variable is active, and (b) a number from 1 to 7 representing a membership function for the variable (1 corresponds to *extremely low* and 7 to *extremely high*). Finally, column 11 indicates the rule output rating and contains a single floating point value

	V_1	V_2	V_3	V_4	V_5	V_6	V_7	...	V_{33}			
B	B	I	B	I	B	I	B	I	...	B	I	F
B	B	I	B	I	B	I	B	I	...	B	I	F
B	B	I	B	I	B	I	B	I	...	B	I	F
B	B	I	B	I	B	I	B	I	...	B	I	F
B	B	I	B	I	B	I	B	I	...	B	I	F

Fig. 2. Internal rule base representation for a rule base with 5 rules ($O = 5$) and 33 linguistic variables ($V = 33$). B indicates a boolean value: $B \in \{T, F\}$; I an integer: $I \in \{1, 2, 3, 4, 5, 6, 7\}$; and F a float: $F \in \{0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0\}$.

from the set $\{0.1, 0.2, \dots, 1.0\}$). The internal representation for a rule base is simply a 30×11 matrix (note that columns 2 – 10 contain a Boolean and an integer).

The number of possible genotypes for rule bases is

$$2^O \times 2^{V \times O} \times 7^{V \times O} \times 10^O,$$

as there are $2^{V \times O}$ possible truth assignments for the Boolean variable in column 1 of the matrix, $2^{V \times O}$ possible truth assignments for all Boolean variables in columns 2 – 33 of the matrix and $7^{V \times O}$ possible assignments for integer variables in columns 2–10 of the matrix, and 10^O possible assignments for the variable in the 11-th column of the matrix as there are 10 possible output ratings.

As an example see figure 3. This rule base consists of a single active rule (in the first row) indicated by the true value in the first column and this rule has one active linguistic variable — V_2 . This rule would be interpreted in the phenotype representation (with reference to table 3.1 as: If *Price Change* is *Extremely High* then *rating* = 0.9.

	V_1	V_2	V_3	V_4	V_5	V_6	V_7	...	V_{33}									
T	F	2	T	1	F	7	F	1	F	6	F	1	F	7	...	F	1	0.9
F	F	4	T	2	F	2	T	3	T	4	F	5	F	1	...	F	2	0.3
F	T	1	F	3	F	2	F	2	F	2	T	4	F	4	...	F	4	0.7
F	T	4	F	5	F	1	F	4	F	2	T	5	F	2	...	F	5	0.4
F	F	6	T	3	F	3	T	3	F	7	F	3	T	7	...	F	3	0.5

Fig. 3. Example of the internal rule base representation. The first column indicates if a rule is used or not, the last column indicates the output rating give full satisfaction of that rule. The central columns (V_1 to V_{33}) show whether a linguistic variable is used and the membership function required for that variable in the rule. The order of the middle columns is important and corresponds to the particular linguistic variables.

Linguistic Variables

In this subsection we describe the linguistic variables used in the fuzzy rules as described above. The linguistic variables are based on technical indicators used by finance practitioners that are calculated using close price and volume data and in some cases Index and interest rate data.

Table 3.1 lists the definitions of the linguistic variables used in the system, table 3.1 lists conditions that must be met for the corresponding linguistic variable formulae in 3.1 to apply. The abbreviations used in the tables have the following expansions: SMA, single moving average; DMA, double moving average; PPO, price percentage oscillator; OBV, on balance volume indicator; SD, standard deviation; RSI, relative strength index; MFI, money flow index; Bol, bollinger band; Vol. DMA, volume double moving average; PVO, percentage volume oscillator; DMI, directional movement index; %R, percent R.

For the moving average variables the value ϵ refers to the period in days between the signal trigger (when the shorter moving average becomes lower or higher than the longer) and the day t for a signal to occur, this is clarified in figure 4. In the OBV linguistic variable the value obv_t for each day t is calculated from historical data using the algorithm:

Initially at $t = 0$ $obv_0 = v_0$, then for each subsequent day t of historical data observations obv_t is calculated as follows:

- if $p_t > p_{t-20}$ then $obv_t = obv_{t-1} + v_t$,
- if $p_t < p_{t-20}$ then $obv_t = obv_{t-1} - v_t$,
- if $p_t > p_{t-20}$ then $obv_t = obv_{t-1}$,

3.2 Evolutionary Process

The fuzzy rule bases undergo an evolutionary process. An initial population of rule bases (genotypes) are selected at random and may be seeded with some rule bases that correspond to accepted technical trading strategies.

The evolutionary algorithm used in our asset allocation system is summarized by the following sequence of steps:

1. Initialize population P of n solutions (each solution RB_i is a rule base):

$$P = \langle RB_1, RB_2, \dots, RB_n \rangle,$$

2. Evaluate each solution: calculate $eval(RB_i)$ for $i = 1, \dots, n$,
3. Identify the best solution found so far (*best*),
4. Alter the population by applying a few variation operators (tournament selection of size 2 is used),
5. Apply a repair operator to each offspring; this operator controls diversity of offspring with respect to the best solution $best_{previous}$ from the previous generation (elitism is not used),

No.	Name	Formula	Restrictions
1	Price Change 1	$\ln\left(\frac{p_t}{p_{t-\delta}}\right)$	$\delta = 20$
2	Price Change 2		$\delta = 50$
3	Price Change 3		$\delta = 100$
3	SMA Buy	$\frac{p_t}{ma_t}$	$len_{ma} \in \{10, 20, 30\}$
4	SMA Sell	$\frac{ma_t}{p_t}$	$len_{ma} \in \{10, 20, 30\}$
5	DMA Buy 1	$\frac{ma1_t}{ma2_t}$	$len_{ma2} \in \{10, 20, 30\}$
			$len_{ma2} \in \{40, 50, 60\}$
6	DMA Buy 2		$len_{ma1} \in \{60, 70, \dots, 120\}$
			$len_{ma2} \in \{130, 140, \dots, 240\}$
7	DMA Buy 3	$len_{ma1} \in \{60, 70, \dots, 120\}$	
		$len_{ma2} \in \{130, 140, \dots, 240\}$	
8	DMA Sell 1	$\frac{ma2_t}{ma1_t}$	$len_{ma2} \in \{10, 20, 30\}$
			$len_{ma2} \in \{40, 50, 60\}$
9	DMA Sell 2		$len_{ma1} \in \{60, 70, \dots, 120\}$
			$len_{ma2} \in \{130, 140, \dots, 240\}$
10	DMA Sell 3	$len_{ma1} \in \{60, 70, \dots, 120\}$	
		$len_{ma2} \in \{130, 140, \dots, 240\}$	
11	PPO 1	$\frac{ma1_t - ma2_t}{ma1_t} \times 100$	$len_{ma2} \in \{10, 20, 30\}$
12	PPO 2		$len_{ma2} \in \{40, 50, 60\}$
			$len_{ma1} \in \{60, 70, \dots, 120\}$
13	PPO 3	$len_{ma2} \in \{130, 140, \dots, 240\}$	
		$len_{ma1} \in \{60, 70, \dots, 120\}$	
		$len_{ma2} \in \{130, 140, \dots, 240\}$	
14	OBV Buy	$\frac{(p_t - \max[p_{t-1}, \dots, p_{t-n}])}{p_t} + \frac{(\max[obv_{t-1}, \dots, obv_{t-n}] - obv_t)}{obv_t}$	
15	OBV Sell	$\frac{(\min[p_{t-1}, \dots, p_{t-n}] - p_t)}{p_t} + \frac{(obv_t - \min[obv_{t-1}, \dots, obv_{t-n}])}{obv_t}$	
20	SD 1	$sd(\ln(\frac{P_t}{P_{t-1}}), \dots, \ln(\frac{P_t}{P_{t-\delta}}))$	$\delta = 20$
21	SD 2		$\delta = 50$
22	SD 3		$\delta = 100$
23	RSI	$RSI = 100 - \frac{100}{1+RS}$ $RS = \frac{total\ gains}{total\ losses}$	
24	MFI	$MFI = 100 - \frac{100}{1+MR}$ $MR = \sum \frac{MF^+}{MF^-}$ $MF^+ = p_i \times v_t, \text{ where } p_i > p_{i-1}, \text{ and}$ $MF^- = p_i \times v_t, \text{ where } p_i < p_{i-1}$	
25	Bol 1	$Bol = \frac{p_t - ma_t}{2 \times sd(P_t, \dots, P_{t-\delta})}$	$\delta = 20$
26	Bol 2		$\delta = 50$
27	Vol. DMA Buy 1	$\frac{vma1_t}{vma2_t}$	$len_{vma1} = 5, len_{vma2} = 20$
28	Vol. DMA Buy 2		$len_{vma1} = 20, len_{vma2} = 100$
29	Vol. DMA Sell 1	$\frac{vma2_t}{vma1_t}$	$len_{vma1} = 5, len_{vma2} = 20$
30	Vol. DMA Sell 2		$len_{vma1} = 20, len_{vma2} = 100$
30	PVO 1	$\frac{ma1_t - ma2_t}{SM_t} \times 100$	$len_{ma1} = 5, len_{ma2} = 20$
31	PVO 2		$len_{ma1} = 20, len_{ma2} = 100$
32	DMI	see [20]	
33	%R	$\%R = \frac{p_t - \min[p_{t-1}, \dots, p_{t-10}]}{\max[p_{t-1}, \dots, p_{t-10}] - \min[p_{t-1}, \dots, p_{t-10}]}$	

Table 1. Linguistic variables used in the system and calculation. The period length of the moving average variables is subject to evolution and may take values within the restrictions given in the third column.

No.	Conditions
4	$(p_{t-\epsilon} < sma_{t-\epsilon})$ and $(p_t > sma_t)$
5	$(p_{t-\epsilon} > sma_{t-\epsilon})$ and $(p_t < sma_t)$
6-9	$(ma1_{t-\epsilon} < ma2_{t-\epsilon})$ and $(ma1_t > ma2_t)$
10-13	$(ma1_{t-\epsilon} > ma2_{t-\epsilon})$ and $(ma1_t < ma2_t)$
14	$(p_t > \max[p_{t-1}, p_{t-2} \dots p_{t-n}])$ and $(obv_t \leq \max[obv_{t-1}, \dots, obv_{t-n}])$
15	$(p_t < \min[p_{t-1}, p_{t-2} \dots p_{t-n}])$ and $(obv_t \geq \min[obv_{t-1}, \dots, obv_{t-n}])$
23,24	$(vma1_{t-\epsilon} < vma2_{t-\epsilon})$ and $(vma1_t > vma2_t)$
25,26	$(vma1_{t-\epsilon} > vma2_{t-\epsilon})$ and $(vma1_t < vma2_t)$

Table 2. Conditions for the linguistic variables in 3.1 to be defined. ϵ is an interval in days to control the period some of the variables take values, it was fixed to 5 days in all tests described in this paper.

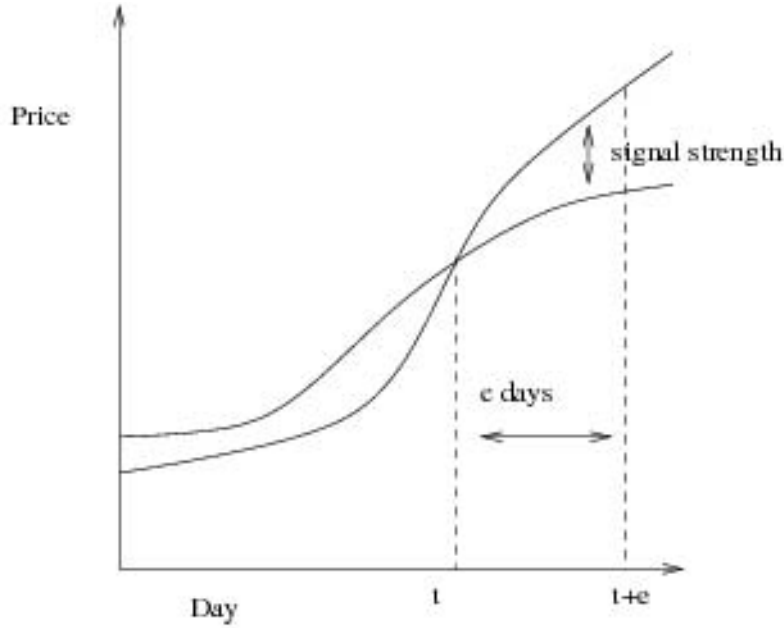


Fig. 4. Moving average signal design. The period “e” is the period during which the indicator emits a positive signal, and “signal strength” is the magnitude of the signal.

6. Repeat steps 2-5 successively for N generations,
7. The best solution after N generations represents the final solution.

Three variation operators (one mutation and two crossovers) and one repair operator are used in the process, the probabilities of applying a particular operator are evolved. We discuss each operator in turn below.

The mutation operator works by possibly modifying each gene of a single parent rule base in the process of producing an offspring. The type of gene remains the same: for instance a Boolean value cannot become an integer used to represent a membership function nor a decimal used to represent an output rating. If a gene is Boolean it is flipped. Otherwise if it is an integer or float, one of three events occur with equal probabilities:

1. The corresponding gene in the parent is incremented or decremented (equal probability for either) by a small amount, δ , to derive the offspring gene: for floats $\delta = 0.1$ and for integers $\delta = 1$. Since integers represent membership sets the change corresponds to a shift of one degree of membership (for example from low to very low).
2. The gene in the offspring is assigned a new value at random. For an integer gene the new value is selected from the domain $1, 2, \dots, 7$ and for a float from the domain $0.1, 0.2, \dots, 1.0$.
3. The corresponding gene in the parent is passed unaltered to the offspring.

The two crossover operators combine genes from two parents to produce a single offspring. The first one, uniform crossover, assigns each gene in the offspring the value of a gene selected from one of the parents (the parent that provides the gene value is selected with equal probability). The second crossover operator assigns the rows of the offspring matrix by selecting — with equal probability — rows from both parents. In other words, the effect of this operator is to build a new rule base by choosing complete rules from each parent.

The last operator used in the system is a repair operator. It is used to maintain stability between generations. It is a binary operator with two rule bases as arguments and its effect is to modify the first genotype in such a way that it is no more than p percent different from the second genotype, no more than p percent different from the second genotype, which is best before we started the most recent adaptation. The number p is a parameter of the method. We found this parameter to be very important in controlling the type of rules generated, in section 4 the values we used for p are given.

3.3 Evaluation of a Fuzzy Rule Base

The evaluation process comprises of three stages: in the first stage individual stocks are evaluated according to a rule base (section 3.3); in the second stage, the overall rule base's performance is evaluated (section 3.3). The return on investment (ROI) is adjusted in the final stage of the evaluation process (section 3.3).

Rating of individual stocks

In this section the procedure to assign a rating to stocks with respect to a rule base is explained. For any stock X a rating $RB(X)$ is defined. This mapping from stock data to stock rating will be described using an example. Consider a rule base as follows:

1. If *Single Moving Average Buy Signal* is *High* then *rating* = 0.7.
2. If *Price Change is High* and *Volume Change is Very High* then *rating* = 0.4.

On a particular day t the following observations are made of technical indicators for stock X :

1. *Volume Change* = 0.5
2. *Single Moving Average Buy Signal* = 0.95
3. *Price Change* = 0.2

Initially each rule in the rule base is processed separately. The first and only “If” part of the first rule is: If *Single Moving Average Buy Signal* is *High*. It was observed that for stock X on day t the *Single Moving Average Buy Signal* had value 0.95. The membership function for *High* is defined by its min, center and max which are, in this case, 0.12, 0.97 and 3.88 respectively. Using Equation 1, below, a membership function defined by these values maps the observed value 0.95 to a degree of membership of 0.97 in *High* or 97% *High*, a visualization of this procedure is given in Figure 5.

$$m(x) = \begin{cases} \frac{x-min}{center-min} & , \text{ if } min \leq x \leq center \\ 1 & , \text{ if } x = center \\ \frac{x-max}{center-max} & , \text{ if } center \leq x \leq max \\ 0 & , \text{ otherwise} \end{cases} \quad (1)$$

We now move to the output rating part of the first rule: then *rating* = 0.7. As the rule fulfilled the “If” part of the rule to the degree of 0.97 we adjust the output rating: $0.97 \times 0.7 = 0.679$.

The system looks at each rule in turn, the second rule in this example has two inputs: If *Price Change* is *High* and *Volume Change* is *Very High*. Each conjunction is processed in turn in the same way as the single conjunction in first rule. By this procedure, it is determined that the observation *Price Change* = 0.2 implies membership in the fuzzy set *High Price Change* = 0.5; and that *Volume Change* = 0.5 implies membership in *Very High Volume Change* = 1.0. These two values are combined by multiplying the membership degrees for each part resulting in a combined membership degree for the second rule: $0.5 \times 1 = 0.5$. The output rating is then adjusted using this combined membership degree: $0.5 \times 0.4 = 0.2$.

Finally an output rating with respect to the whole rule base is calculated. The final rating combines the sub-ratings from each rule to give a rating for

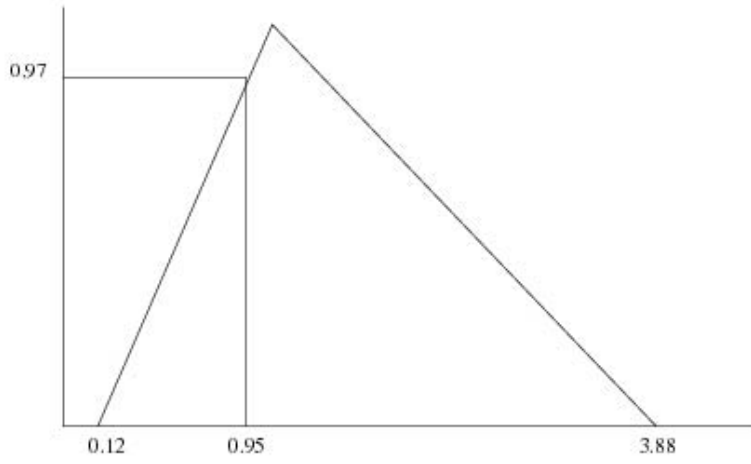


Fig. 5. Finding the degree of membership of observed single moving average buy signal 0.95 for stock X is "High" with a degree of 0.97.

stock X given some input data, we term this rating $RB(X)$. $RB(X)$ is defined as the center of mass of the sub-ratings from each rule in a rule-base:

$$RB(X) = \frac{\sum o_i}{\sum r_i}, \quad (2)$$

where o_i is the output of rule i for stock X , and r_i is the rating of rule i .

Recall that for the first rule the sub-rating was 0.679 and for the second it was 0.200. Therefore the combined rating is: $(0.679 + 0.2) \div (0.7 + 0.4) = 0.799$.

Evaluation of rule base performance

The procedure described in the previous section is applied to each stock in the market. This enables the definition of a ranking of stocks in a market M ordered by rating:

$$R(M) = \langle X_{i1}, X_{i2}, \dots, X_{in} \rangle, \quad (3)$$

where $M = \{X_1, X_2, \dots, X_n\}$ and $RB(X_{ik}) \geq RB(X_{ik+1})$.

The performance of a rule base is measured by analysis of simulated trading on a historical data window. A decoder (Figure 6) defines the interpretation of the ranking to make decisions for portfolio construction. In the simulated scenario an initial capital is used to construct a portfolio on day 1 of the simulation period, this portfolio is then rebalanced over the rest of the period depending on the signals generated by the rule base under evaluation.

In the system, a portfolio P_t is defined as a vector of holdings of stocks in $M = (X_1, \dots, X_n)$ at time t :

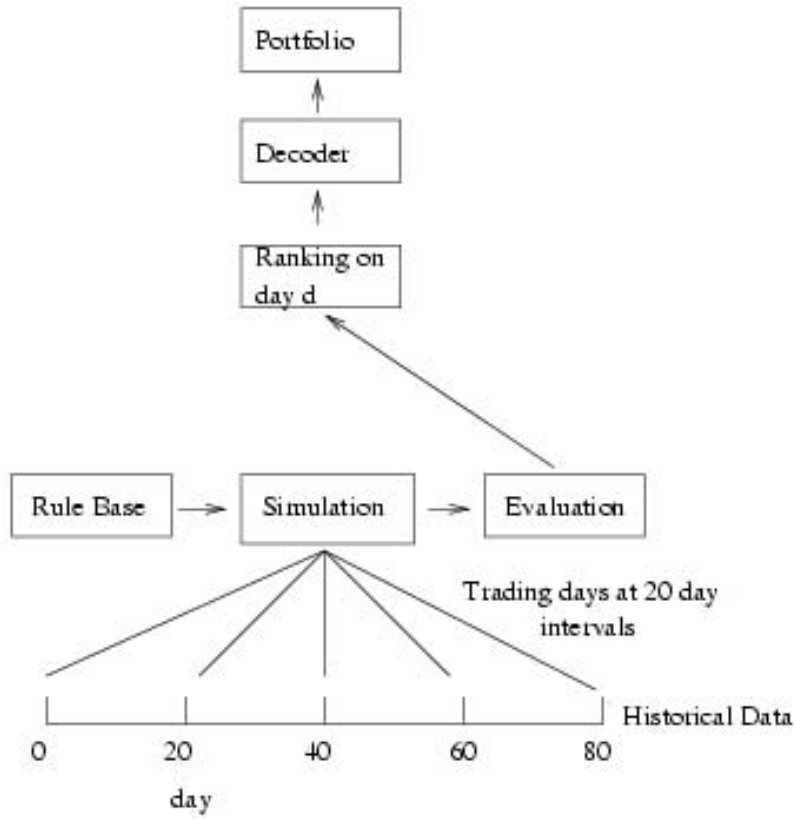


Fig. 6. The decoder takes a ranking and recommends a portfolio.

$$P = [a_1 X_{i1}, \dots, a_k X_{im}], \quad (4)$$

where a_1, \dots, a_k are natural numbers, $\{X_{i1}, \dots, X_{im}\} \subseteq M$, and $Value(P, t) = \sum a_j \times price(X_{ij})$, for $j = 1, \dots, m$.

The decoder interprets the ranking in terms of two parameters: *buy best stocks percentage* (BBS) and *sell Worst Stocks Percentage* (SWS). BBS is the percentage of stocks to select from top of the ranking and SWS is a percentage to sell from the bottom of the ranking. At set intervals $dist = 20$ during the simulation period stocks are sold according the ranking on the trading day according to SWS and then bought according to BBS. In this way the initial portfolio P_1 is updated to get a new portfolio P_2 and so on until the end of the simulation period is reached.

The measure used for evaluation of portfolio performance is Return on Investment (ROI) over the simulation period (see Equation 5).

$$ROI = \frac{\ln(V_{t_1}) - \ln(V_{t_0})}{t_1 - t_0}, \quad (5)$$

where V =Portfolio Value, t_1 = End Time and t_0 = Start Time.

The result of the simulation is the return on investment over the whole simulation period for RB_X : $ROI(RB_X)$. To compare RB_X to another rule base RB_Y it is the case that if $ROI(RB_X) > ROI(RB_Y)$ then RB_X is better than RB_Y . The basic ROI criteria is supplemented by consideration of a few additional characteristics of performance, this takes place in the final evaluation and are described in the following section.

Final Evaluation

The ROI is the basic measure for rule base evaluation. However to further guide the search for optimal performing rule bases ROI is adjusted using penalty functions. The final evaluation value equals the ROI from the simulation minus penalties. There are two penalties applied to modify ROI, and they are:

1. Portfolio loss penalty
2. Ockham's razor penalty

Let us discuss each penalty in turn starting with the portfolio loss penalty.

The portfolio loss penalty implements a capacity for reducing risk. In simulation we measure the portfolio gain or loss on each trading day (see section 3.3). Solutions that result in a reduction of portfolio value are penalized if they result in losses on any trading day even if at the end of the simulation period the return was high. The penalty becomes progressively higher for large losses.

The loss penalty is calculated using the formula:

$$Penalty_1 = \sum_{t=d, 2d, \dots, \frac{n}{2}d} m(t),$$

where d is the interval between trading days and $m(t)$ is calculated as follows,

$$m(t) = \begin{cases} 0.01, & \text{if } P_t - P_{t-d} \leq -5 \\ 0.1, & \text{if } -5 \leq P_t - P_{t-d} \leq -10 \\ 10, & \text{if } P_t - P_{t-d} \leq -10 \end{cases}$$

The second penalty, Ockham's razor, is calculated as follows:

$$Penalty_2 = r \times k_1,$$

where r is the number of active rules and k_1 is a constant which was set to 0.1 for the results given here. Ockam's razor reduces the fitness of solutions

with many rules. The magnitude of k_1 controls a balance where the return produced using a candidate rule base must be justified by the number of rules. For example, setting the value to 0.1 causes the evolutionary process to produce rules where each rule must contribute at least 0.1 to the fitness. A rule base with fewer rules that results in similar return on investment to one with a greater number of rules is preferred. Rule bases with fewer rules are also less likely to be over-fitted to the training data.

The penalties are added together to get an overall value for a rule base and this value is deducted from the ROI for that rule base. Figure 7 gives an overview of the process used to determine a fitness value — the penalized ROI. Adjusting the ROI using the penalty functions promotes the discovery of rule bases which when applied give high return with less risk and generalize well outside training data.

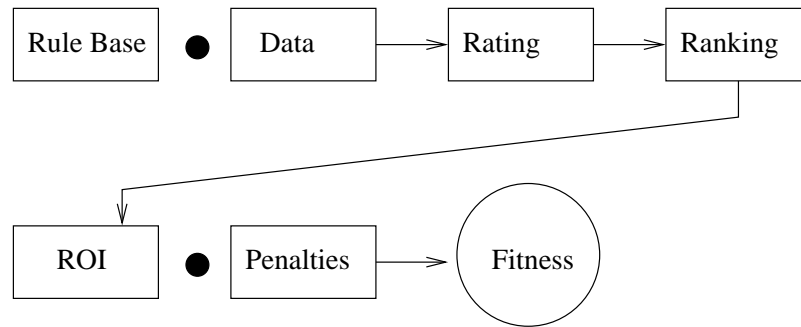


Fig. 7. Complete evaluation process.

3.4 Adaptation

In this section methods to cause rule bases to adapt to market conditions are discussed. Approaches towards this are through selection of data windows and by controlling the search. During the search process the performance of rule bases is evaluated based on data as described in the previous section. We first discuss the methods to select and alter the historical data used to evaluate the rules and then describe a method used to influence the ability of the evolutionary search to use information previously discovered across training windows.

Selection of Training Data Window

The selection of the training data window controls the period of training data used when generating rule bases. Three methods for selecting a data window are able to be used by the system:

1. Static window,
2. Sliding window,
3. Variable length sliding window.

The static window method uses a single initial period to evolve a rule base and then the rules from this period are used for all future trading. The sliding window uses a recent historical time windows for evaluation optimizing the rule base to recent periods.

The system also allows the length of the period of training data used for each sliding window to vary according to performance measured using a Sharpe ratio [17] which relates benchmark returns to the return of a real portfolio found from applying the rule base and decreases the window length when performance is worse than the benchmark. The results given in this chapter used the variable length sliding window method and the window length was controlled using the formula:

$$windowlength = e^{k \times \ln(sharpe)} \times maxLength,$$

where k is constant set to control the sensitivity of the window change which was set to 2, $sharpe$ is a Sharpe ratio $sharpe = \frac{(r_p - r_f)}{\sigma_p}$ and $maxLength$ is the maximum window length. The new window date is never set earlier than the previous window start date. In the case that the the window length is calculated as earlier than this, the previous start date is used.

Adapting the action of the repair operator

The repair operator is used to maintain stability between generations. It is a binary operator with two rule bases as arguments and its effect is to modify the first genotype in such a way that it is no more than p percent different from a second genotype. On initialization the second genotype is the solution from the previous window and during the search it is the best solution found at the current generation.

The parameter p is adjusted depending on the performance of a real portfolio in relation to the index which serves as a benchmark. It is reduced when performance is worse than the benchmark and increased if the real portfolio is out performing the benchmark. The rationale is to focus the search close to solutions while they give good performance and to broaden the search when this performance decays. p is increased or decreased according to the formula:

$$p = \begin{cases} sharpe \times 0.5 \times (1 + k), & \text{if } P_t^{real} < P_{t-d}^{real} \\ sharpe \times 0.5 \times (1 - k), & \text{if } P_t^{real} > P_{t-d}^{real} \\ 0, & \text{otherwise} \end{cases}$$

where the $sharpe$ ratio $sharpe = \frac{(r_p - r_f)}{\sigma_p}$ and k is a constant set in the configuration file to control the sensitivity of p to variation in portfolio performance.

The sharpe ratio combines measurement of both risk and return with respect to the benchmark to make the adjustment of p (controlling the level of restriction on the search) dependant not only on the difference between returns between the current and previous month, but also relative to benchmark returns which are expected to be achievable.

4 Performance of the System

The system described in the previous sections is applied to form portfolios from stocks traded on the Australian Stock Exchange (hereafter ASX). Every month during the period of August 2001 to December 2006 two different portfolios are formed. The first one is created using a full adaptive evolutionary process, referred to as the “Adaptive EA” portfolio. The second one is created using a static evolutionary process, ie. a rule base is generated for the first window and then used for the rest of the simulation. This is called the “Static EA” portfolio and serves as a comparison benchmark for the advantages provided by adapting the system. Finally we also compare the performance of our EA portfolios to the ASX index, which reflects the performance of the market as a whole.

Fig. 8. Portfolio values from 08/2001 to 12/2006. Each portfolio starts at a value of 1000 in 08/2001.

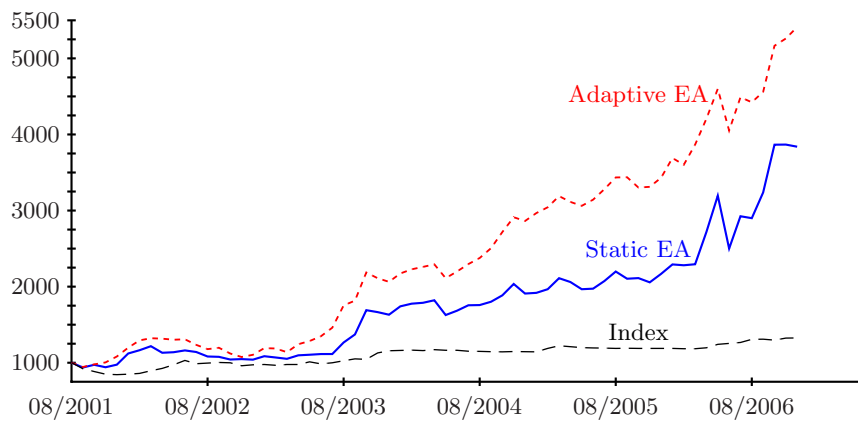


Figure 8 shows the portfolio values over the whole trading period, where the starting points have all been standardized to 1000. Both of the EA portfolios perform very well, clearly dominating the index. The total increase in portfolio value over the whole investment period is measured by “holding period return” reported in Table 3. The whole market during the investment

period improves slightly with the Index's holding period return of only 123% over a period of more than 5 years. During the same time, the Static EA portfolio value increases by 310% and the Adaptive EA portfolio value increases by 474%. Adapting the system fully to the market conditions results in a higher value of the Adaptive EA portfolio than that of the Static EA portfolio by 1.5 times. The annualized return, using either arithmetic or geometric method, is very impressive for the EA portfolios given the market conditions. However, it should be noted that returns of the EA portfolios are more volatile than the market. Further analysis is required to confirm whether investors are rewarded sufficient returns for the risk they bear.

Table 3. Portfolio Returns.

Panel 1. Raw portfolio value			
	ASX Index	Static EA	Adaptive EA
Holding Period Returns	123.34%	310.06%	473.99%
Annualized Arithmetic Returns	14.31%	28.59%	34.06%
Annualized Geometric Returns	13.67%	25.48%	32.08%
Annualized Volatility	11.39%	25.51%	20.45%

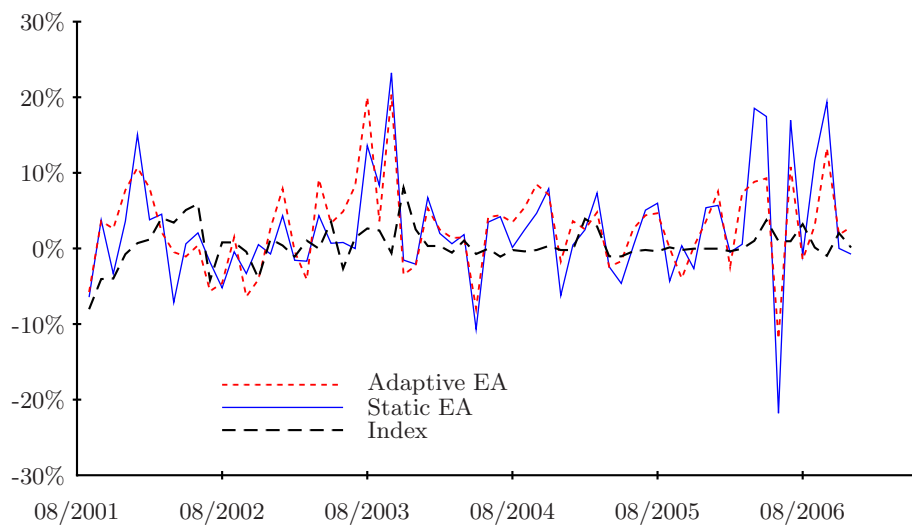
Panel 2. Excess portfolio value			
	ASX Index	Static EA	Adaptive EA
Holding Period Returns	117.99%	304.7%	468.63%
Annualized Arithmetic Returns	8.96 %	23.25%	28.71 %
Annualized Geometric Returns	8.32 %	20.13%	26.72 %
Annualized Volatility	11.36 %	25.49%	20.45 %

Panel 2 of Table 3 reports statistics for excess portfolio values, ie. portfolio values above the risk free return. It can be argued that over time, even if investors hold risk free assets, such as Treasury securities, they are rewarded with financial returns. Therefore it is more relevant to investors to check the performance of the excess returns. Due to the increase in risk free returns over time, the excess returns of all portfolios are slightly lower than the raw returns. However, the relationship between different portfolios does not change qualitatively.

The monthly returns of each portfolio are illustrated in Figure 9 while Table 4 further reports some characteristics of the return distributions. More than half of the time the Adaptive EA portfolio has a monthly return greater than 2.8%, whereas a half of the time the Static EA portfolio and the Index Portfolio have a return smaller than 2.4% and 0.8% respectively. Compared to the Index portfolio, the EA portfolios occasionally do experience higher loss. The largest negative returns for the two EA portfolios are 21.8% and 11.9%, whereas the largest loss for an index portfolio is only 8%. The index portfolio

does not experience any loss greater than 10% in any given month, compared to the occurrence frequency of 3.15% and 1.56% for the two EA portfolios. This result may be indicative of the penalty function not sufficiently discarding the choice of stocks that are more likely to experience a large decline. However, both of the EA portfolios exhibit positive skewness, which is what investors prefer. A positive skewness portfolio has a high probability of large returns. It should be noted that the EA portfolios also have lower kurtosis, which implies less regular and smaller swing away from the mean in investor returns. Overall, the Adaptive EA portfolio has better return potential compared to the Static EA portfolio while maintaining a lower level of risk.

Fig. 9. Portfolio monthly returns from 08/2001 to 12/2006.



We have noted that our EA portfolios have much better return potential than the Index, and at the same time are more volatile. The Sharpe ratio [17] measures how much excess returns (portfolio return r_p above the risk free rate r_f) investors are awarded for each unit of volatility, ie.

$$\text{Sharpe} = \frac{r_p - r_f}{\sigma_p}.$$

As can be seen in Table 5, the Sharpe ratio for the Index is only 0.79, whereas that for the Static EA portfolio is 0.91. Adaptive EA portfolio has the best Sharpe ratio of 1.4, nearly double the reward investors receive for holding

Table 4. Portfolio return characteristics.

	ASX Index	Static EA	Adaptive EA
Average monthly return	0.470%	2.383%	2.838%
Median monthly return	0.078%	0.685%	2.791%
Largest positive return	8.071%	23.257%	20.367%
Largest negative return	-8.008%	-21.769%	-11.930%
Skewness	-0.0829	0.3759	0.4356
Kurtosis	2.7062	2.1657	1.2516
Frequency of gain greater than 10%	0.000%	12.500%	7.813%
Frequency of loss greater than 10%	0.000%	3.125%	1.563%

a passive index portfolio. The improvement in return potential for a fully adaptive system has been well above some additional level of risk for investors.

The Sharpe ratio focuses on portfolio volatility which measures total risk of the portfolio. Modern portfolio theory further decomposes volatility into systematic risk and unsystematic risk. The systematic risk component reflects how the changes in market conditions affect portfolio values, whereas the unsystematic risk component is unique to each portfolio. The constraints under which we form portfolios, such as the maximum stocks in each country or each sector, are in fact constraints to build well diversified portfolios. A well diversified portfolio should have return awarded to compensate for the systematic risk component only. Denote $r_{m,t}$ the returns at time t of the market, the systematic risk β_p of portfolio p is determined by the Capital Asset Pricing Model (CAPM) equation:

$$r_{p,t} - r_{f,t} = \alpha_p + \beta_p(r_{m,t} - r_{f,t}) + e_{i,t}. \quad (6)$$

Since the excess return $r_{p,t} - r_{f,t}$ of any portfolio should be fully explained by its level of systematic risk β_p and the market risk premium $r_{m,t} - r_{f,t}$, in an efficient market the alpha value of the portfolio, α_p , should be zero. If it is not and in fact there is a positive value then the portfolio is outperforming relative to its level of systematic risk and the performance of benchmark index. The higher the alpha value, the better the portfolio is to hold. Both of the EA portfolios have positive alpha, indicating a superior performance (see Table 5). The Adaptive EA portfolio has a value of alpha 1.5 times larger than that of the Static EA portfolio.

The robustness of alpha values can also be measured through the information performance rank, sometimes also known as the appraisal ratio. Essentially, it evaluates the active stock-picking skills of the strategy, once unsystematic risk generated from the investment process is accounted for. As we are comparing each of our portfolio's with the ASX Index, the information ratio is calculated as:

$$\text{Annualized Information Ratio} = \frac{\sqrt{T}\alpha}{\sigma_e}, \quad (7)$$

Table 5. Standard portfolio performance measures.

	ASX Index	Static EA	Adaptive EA
Sharpe ratio	0.789	0.912	1.404
Alpha	NA	0.124	0.184
Information Performance Rank	NA	0.572	1.160
Selectivity	NA	0.124	0.184
Net Selectivity	NA	0.031	0.126

where T is the period multiple to annualize the ratio and σ_e is the standard error of equation 6. Grinold and Kahn, [11], have argued good information ratios should be between 0.5 and 1, with 1 being excellent. Goodwin [10] examined over 200 professional equity and fixed income managers over a ten year period and found that although the median information ratio was positive, it never exceeded 0.5. The information ratio of the Static EA portfolio is 0.57, indicating a very good performance. The Adaptive EA portfolio has an information rank of 1.16, double that of the Static EA portfolio. The information rank very close to 1 is indicative of a very strong and consistent performance.

Finally Fama's Net Selectivity measure [7] provides a slightly more refined method to analyze overall performance for an actively managed fund. Overall performance, measured as the excess returns of the portfolio over the risk-free rate, can be decomposed into the level of risk-taking behavior of the strategy and security selection skill. This security selection skill, or Selectivity, can be measured as a function of the actual return of the portfolio minus the return that the benchmark portfolio would earn if it had the same level of systematic risk. Both of the EA portfolios have a positive Selectivity, and the Adaptive EA portfolio still have much stronger performance than the Static EA one.

The Selectivity value, however, can be broken down still further to calculate Net Selectivity. Given that a portfolio's strategy may not be limited to simply track the benchmark portfolio, which would be the case for our portfolios under examination, it is also necessary to take into account the fact that the portfolios are not fully diversified, relative to the chosen benchmark. In fact, the number of stocks in our EA portfolios is much smaller than in the ASX index. To account for this, net selectivity is the value of selectivity that the strategy adds to the portfolio minus the added return required to justify the loss of diversification from the portfolio moving away from the benchmark. This effectively means any returns that the portfolio earns above the risk free rate must be adjusted for *both* the returns that the benchmark portfolio would earn if it had the same level of systematic risk and the same level of total risk to the benchmark. After accounting for the difference in total risk, the Static EA portfolio still maintain a positive Net Selectivity, but quite marginal. On the other hand, the Adaptive EA portfolio has a very substantial positive Net Selectivity of 0.13, again indicating a very strong performance.

To track the performance of our portfolios over time, rolling Alphas and rolling Information Ratios are calculated. Each month new Alpha and Information Ratio are calculated based on the data for the previous year. Figure 10 and 11 graph these two series. Even though both of the EA portfolios have positive overall Alphas and larger than 0.5 Information Ranks as shown in Table 5, only the Adaptive EA portfolio is able to maintain a consistent level of strong performance over time. This clearly shows the advantages of adapting the portfolio to changing market conditions.

Fig. 10. Rolling Alphas.

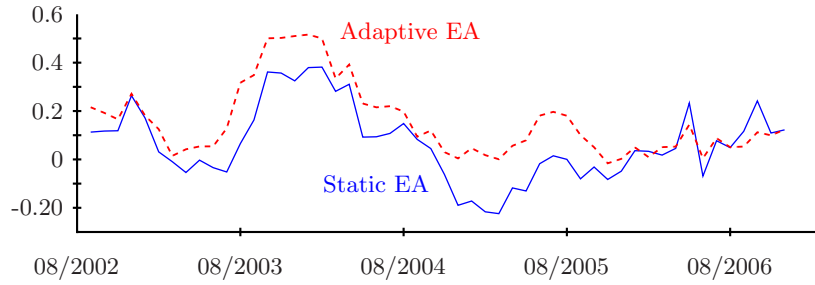
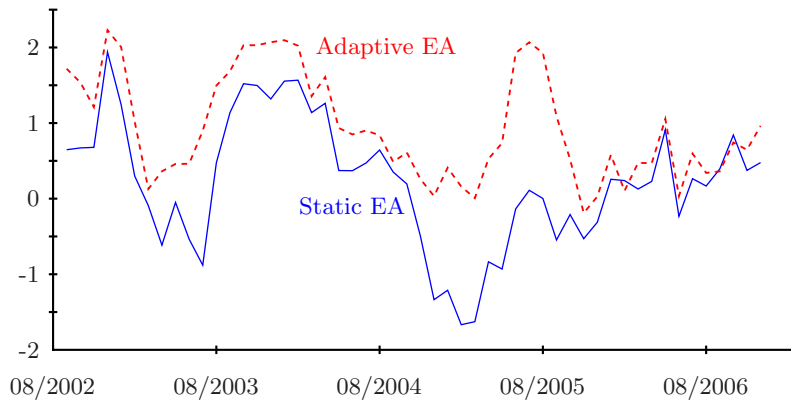


Fig. 11. Rolling Information Ranks.



5 Concluding Remarks

This chapter has provided a discussion of the framework in which an evolutionary algorithm manages a portfolio of selected stocks chosen using only price and volume information of individual company shares. Fuzzy logic rules are designed to optimize towards a specific fitness function, which is a simple return on investment measure. A specific financial penalty function is also incorporated to penalize solutions that select a portfolio of stocks that experiences significant losses. Effectively, it is a penalty for downside risk.

The computational intelligence system described in this chapter is tested on historical data of stocks traded on the Australian Stock Exchange during the period of August 2001 to December 2006. The empirical results show that portfolios constructed using evolutionary algorithm beat the market index by all standard portfolio performance measures. The clear dominance of the Adaptive EA portfolio over the Static EA portfolio is a result of the system being dynamic and adaptive to changing market conditions. Given that we impose both costs to trading and restrictions on how trades can occur, the EA portfolio performance is a relatively impressive result. This is even more particularly true when considering that only price and volume information is used to generate trading signals.

References

1. F. Allen and R. Karjalainen. Using genetic algorithms to find technical trading rules. *Journal of Financial Economics*, 51:245–271, 1999.
2. Panayiotis C. Andreou, Chris Charalambous, and Spiros H. Martzoukos. Robust artificial neural networks for pricing of european options. *Comput. Econ.*, 27(2-3):329–351, 2006.
3. Amir Atiya. Bankruptcy prediction for credit risk using neural networks: a survey and new results. *IEEE Transactions on Neural Networks*, 12(4):929–935, 2001.
4. Antonia Azzini and Andrea G.B. Tettamanzi. A neural evolutionary approach to financial modeling. In *GECCO '06: Proceedings of the 8th annual conference on Genetic and evolutionary computation*, pages 1605–1612, New York, NY, USA, 2006. ACM Press.
5. Anthony Brabazon and Michael O'Neill. *Biologically Inspired Algorithms for Financial Modelling (Natural Computing Series)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006.
6. W. Brock, Lakonishok, J., and B. LeBaron. Simple technical trading rules and the stochastic properties of stock returns. *Journal of Finance*, 47:1731–1764, 1992.
7. E. F. Fama. Components of Investment Performance. *Journal of Finance*, 27(3):551–567, 1972.
8. C. Fyfe, J. Marney, and H. Tarbert. Technical analysis versus market efficiency a genetic programming approach. *Applied Financial Economics*, 9:183–191, 1999.

9. C. Lee Giles, Steve Lawrence, and Ah Chung Tsoi. Noisy time series prediction using recurrent neural networks and grammatical inference. *Mach. Learn.*, 44(1-2):161–183, 2001.
10. T. H. Goodwin. The Information Ratio. *Financial Analysts Journal*, 54(4):34–43, 1998.
11. R.C. Grinold and R.N.Kahn. *Active Portfolio Management*. McGraw-Hill: New York, 2 edition, 2000.
12. P. Hsu and C. Kuan. Reexamining the profitability of technical analysis with data snooping checks. *Journal of Financial Econometrics*, 3(4):606–628, 2005.
13. A.W. Lo, Mamaysky, H., and J. Wang. Foundations of technical analysis: Computational algorithms, statistical inference, and empirical implementation. *Journal of Finance*, 55(4):1705–1765, 2000.
14. C. Neely, Weller, P., and R. Dittmar. Is technical analysis in the foreign exchange market profitable? a genetic programming approach. *Journal of Financial and Quantitative Analysis*, 32(4):405–426, 1997.
15. Jangmin O, Jongwoo Lee, Jae Won Lee, and Byoung-Tak Zhang. Dynamic asset allocation for stock trading optimized by evolutionary computation. *IEICE - Trans. Inf. Syst.*, E88-D(6):1217–1223, 2005.
16. Jean-Yves Potvin, Patrick Soriano, and Maxime Valle. Generating trading rules on the stock markets with genetic programming. *Comput. Oper. Res.*, 31(7):1033–1047, 2004.
17. W.F. Sharpe. Mutual fund performance. *Journal of Business*, 39(1):119–138, 1966.
18. Harish Subramanian, Subramanian Ramamoorthy, Peter Stone, and Benjamin J. Kuipers. Designing safe, profitable automated stock trading agents using evolutionary algorithms. In *GECCO '06: Proceedings of the 8th annual conference on Genetic and evolutionary computation*, pages 1777–1784, New York, NY, USA, 2006. ACM Press.
19. Nils Svargard, Stefan Lloyd, Peter Nordin, and Clas Wihlborg. Evolving short-term trading strategies using genetic programming. In David B. Fogel, Mohamed A. El-Sharkawi, Xin Yao, Garry Greenwood, Hitoshi Iba, Paul Marrow, and Mark Shackleton, editors, *Proceedings of the 2002 Congress on Evolutionary Computation CEC2002*, pages 2006–2010. IEEE Press, 2002.
20. J Welles Wilder. *New Concepts in Technical Trading Systems*. Trend Research, 1978.
21. Shaun-Inn Wu and Ruey-Pyng Lu. Combining artificial neural networks and statistics for stock-market forecasting. In *CSC '93: Proceedings of the 1993 ACM conference on Computer science*, pages 257–264, New York, NY, USA, 1993. ACM Press.