Evolving Robust GP Solutions for Hedge Fund Stock Selection in Emerging Markets

Wei Yan
Department of Computer Science
University College London
Gower Street, London, WC1E 6BT
w.yan@cs.ucl.ac.uk

Christopher D. Clack
Financial Computing and Quantitative Finance
Department of Computer Science
University College London
Gower Street, London, WC1E 6BT
c.clack@cs.ucl.ac.uk

ABSTRACT

Stock selection for hedge fund portfolios is a challenging problem for Genetic Programming (GP) because the markets (the environment in which the GP solution must survive) are dynamic, unpredictable and unforgiving. How can GP be improved so that solutions are produced that are *robust* to non-trivial changes in the environment? We explore an approach that uses subsets of extreme environments during training.

Categories and Subject Descriptors

I.2.M [**Artificial Intelligence**]: Miscellaneous

General Terms

Algorithms, Experimentation

Keywords

Genetic Programming, Diversity, Phenotype, Finance, Adaptation, Dynamic Environment

1. INTRODUCTION

In May 1994, following an increase in US short-term interest rates, tumbling bond prices, and a knock-on effect on international currencies, the financial speculator George Soros lost $650,000,000 in just two days [18]. On 17th August 1998, Russia defaulted on its debts; three days later the financial markets across the world collapsed and in just one day, the hedge fund Long Term Capital Management lost $553,000,000 [18].

The financial markets are highly dynamic, unpredictable and unforgiving. If GP is used to evolve a solution to a financial trading or investment problem it must be robust to these time-varying disturbances in the markets.

It follows from the above examples that by "robust" we do not mean insensitivity of the fitness of an individual to perturbations resulting from the genetic operators (genotypic robustness [27, 28]); although this form of "robustness" favours broad plateaus to sharp peaks in the search space, it does not give much indication about how the best-of-run individual will perform when the fitness function itself changes (i.e. the surface of the search space fluctuates). Relevant, though insufficient, other previous definitions of robustness include the insensitivity of the fitness of an individual to small fluctuations in an individual's parameters (sometimes known as phenotypic robustness or generalizability [4, 29]) and the insensitivity to a noisy fitness function [5, 8, 19]. The problem with these latter two definitions is that all known work in the area assumes that the fluctuations or noise are drawn from a known and time-invariant distribution (typically uniform or Gaussian), and are small. By contrast, the financial markets undergo large, abrupt and time-varying changes.

Aragón et al [1] model a dynamic environment as a sequence of fitness functions, each defined by changes to the previous. The model uses occasional macro-mutation for radical genotoype shake-up ("recrudescence"), and assumes that all possible changes to the current fitness function are enumerable (and finite, and, in practice, few). It assumes that we can evolve continuously and wait several generations before adaptation to the new fitness function is achieved. Unfortunately, in the real world we cannot wait for the evolutionary system to learn from the new environment!

Our approach is substantially different to the prior work in this area, and is based on the assumption that examples exist of greatly differing extreme environments. If that assumption holds, then we propose to present these examples to the genetic population during training in order to select individuals that perform well in a variety of extreme training environments. The obvious research question this entails is whether, as a result of such exposure, trained individuals are more likely to be robust to large disturbances in the environment? [1] There are three ways we plan to measure this kind of robustness in the context of our finance application:

1. when exposed to an out-of-sample volatile validation data-set, a more robust solution will have a lower standard deviation of returns, while the returns themselves do not decrease; or

[1]An alternative approach is to look for an *adaptive* solution, i.e. one that detects changes in the environment and responds by modifying its internal structure and the way that it operates. However, a similar question arises: in an unforgiving environment, would it have *time* to adapt and survive without prior exposure to extreme environments?

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
GECCO'07, July 7–11, 2007, London, England, United Kingdom.
Copyright 2007 ACM 978-1-59593-697-4/07/0007 ...$5.00.

2. when exposed to an out-of-sample volatile validation data-set, a more robust solution will have higher returns while the standard deviation does not increase; or

3. when exposed to an out-of-sample volatile validation data-set, the mean return per unit of risk of a more robust solution will not significantly reduce from that measured during training.

2. RELATED WORK

2.1 Robustness

Robustness for a biological system is a property to allow a system to maintain its functionality despite internal and external perturbations [14, 30].

Robustness is a very broad theme and it is impossible to capture all its aspects by means of a single definition. Robustness is an ubiquitously observed property of biological systems. It is considered to be a fundamental feature of complex evolvable systems [14].

The definition of robustness in evolutionary systems varies from author to author, but in broad terms, it can divided into two categories:

1. Robust to internal changes (genotypic robustness)

   Robustness as the resistance to changes from variation operators such as crossover and mutation. Soule [27, 28] observes that the most outstanding evidence of pressure towards this type of robustness is the phenomenon of code growth (or bloat) in GP. Code bloat is a rapid increase in code size that does not result in fitness improvements. It is proposed that GP trees grow this extra code ("introns") as a means of protecting the useful code within good solutions. By adding introns the useful code is less likely to be affected by crossover or other similar operators. The robustness in this sense can be drawn parallel to gene redundancy in biosystems and to the existence of "neutral networks" [2] which enable a population to maintain a dominant phenotype required for adaptation despite random genotype changes during the evolution [11].

2. Robust to external changes (phenotypic robustness)

   (a) Robustness as the generalisation ability of the programs evolved using GP [2, 15, 16, 21, 23]. The concept of generalisation is originated from connectionist or symbolic learning research and it is defined as the desired successful performance of the solution when it is applied to an environment similar to the one it was evolved for. In the context of evolutionary systems, the ability to generalise is defined as "the predictive accuracy of the learner in mapping unseen input cases to outputs with a satisfactory degree of correction" [17]. In this respect, robustness is in line with though opposite to the definition of overfitting. Overfitting happens when the computational effort spent on obtaining a more precise fit of the sample results in an increased error on other data.

   (b) Robustness as the ability to cope with non-constant noise [13, 22]. Practical optimisation problems often require the evaluation of solutions through experimentation, stochastic simulation, sampling, or even interaction with the user. Thus, most practical problems involve noise. Jordanne et al. [13] investigated this particular aspect of robustness when noise is added to the deterministic objective function values.

   (c) Robustness as the sensitivity of performance quality in the presence of external environmental perturbations. For example, Hermann [10] defines robust solutions as the one that has the best worst-case performance.

   This aspect of robustness is the most consistent with phenotypic robustness in nature. Although a biological system exhibits robustness in terms of genes, structures etc, from an evolution point of view, ultimately robustness of only one feature matters: fitness is the ability to survive and reproduce (which in evolutionary systems means the performance quality of a solution).

   (d) Robustness as the ability for self-repair when subject to severe phenotypic damage [20, 3]. This behaviour is reminiscent of autonomous regeneration of the pond organism hydra, which can reform itself when its cells are dissociated and then re-aggregated in a centrifuge [7].

[2]Connected networks of RNA sequences with identical structure.

2.2 Structured Training Sets for Robustness

The way in which training data is presented to the population is central to our work yet has received little prior attention. The techniques that have been proposed are twofold: the use of random noise in the training data; and the use of randomly generated environments (fitness cases). However, the experimental methodology of the prior work is not entirely helpful in giving confidence that robust solutions have actually been evolved.

Ito et al and Reynolds [12, 24] use noise and modify initial conditions in order to promote robustness of the programs produced by GP — robustness both to changes in the initial conditions and to changes in the environmental stimuli. The use of noise can be helpful in reducing the brittleness of programs and increasing the likelihood of robustness [24]. In Ito et al. [12] both changing initial coordinates and the direction of the robot, together with the introduction of noisy sensors and actuators, are tried to produce robust programs for robot behaviour. Separate training and testing sets are used but there is no discussion of how training and test cases should be chosen. Furthermore, training and testing comparisons are done on a generation basis and the measure of robustness in testing on a different environment after the training (i.e., after evolution has stopped) is not reported. The results of the experiments do not make it clear whether a robust behaviour has been reached and if so, how it is reached.

Haynes and Wainwright [9] use GP to evolve an agent that can survive in a hostile environment. Rather than using a fixed environment for a particular run, a new environment is generated randomly at the end of each generation. In this way, it is hoped that the agent can handle "any" environment. Since the agent does not seem to be tested in a new environment after the evolution has stopped, the nature and the degree of robustness to new environments (that might have been obtained by variable training environment during the evolution) remains unexplained.

A good example of experiments attempting to reduce the brittleness of individuals generated by GP is presented in [21]. The system in this paper evolves optimised manœuvres for a pursuer/evader problem. The results of this study suggest that use of a fixed set of training cases may result in brittle solutions due to the fact that such fixed fitness case may not be representative of possible situations that the agent may encounter. It is shown that use of randomly generated fitness cases at the end of each generation can reduce the brittleness of the solution when the solution is tested against a set of large representative situations after the evolution. However, a proper selection method for training and testing cases is not provided.

Rosca [25] addresses issues of size and generality in GP; however, the degree of overlapping between training instances and the testing instances does not seem to be explicitly controlled. In such a case, an objective and direct comparison using a common basis between training and testing may be difficult.

3. DESCRIPTION OF THE ALGORITHM

We are concerned with not only the performance or fitness of the GP evolved solutions but also the performance volatility of the GP evolved solutions across a range of environment dynamics. For example, in a scenario where market prices are rising ("bull market"), a scenario where market prices are falling ("bear market"), and a scenario where market prices are fluctuating with large amplitude ("volatile market").

We therefore consider the training data to be a set of fitness cases — a vector of environments — representing a possible range of different environments and then adjust the fitness with its perceived volatility to obtain an whole picture of an individual's performance . Let S be the training environments vector and s_n be the n^{th} possible type of environment, which we call a "scenario"; then $S = \{s_1 \ldots, s_n\}$. We also hold a separate out-of-sample validation vector V.

We consider three ways in which the GP population should be exposed to these scenarios:

1. "Standard GP" (SGP): use the entire vector S, treated as a single unit, throughout all generations;

2. "Multiple-scenario Evaluation in the Last Generation" (MELG): use a variant of the three-dataset methodology [23, 6], where the entire vector S (treated as a single unit) is used for $n-1$ generations, and in the final generation individuals are tested against a subset of environments $\{s_i\}$ drawn from S. The "best-of-run" individual used in the validation on set V is that which has, in the final generation, the highest Volatility-Adjusted Fitness (see below);

3. "Multiple-scenario Evolution" (MEVO): in each generation, use a subset of environments $\{s_i\}$ drawn from S, and ascribe to each individual a Volatility-Adjusted Fitness (see below). Evolution proceeds as normal on the basis of this adjusted fitness measure. The "best-of-run" individual from the final generation is used in the out-of-sample validation on set V.

Volatility-Adjusted Fitness

We have previously introduced S as the training environments vector and s_n as the n^{th} possible scenario; hence $S = \{s_1 \ldots, s_n\}$. Now let $M = \{m_1 \ldots m_p\}; m_j \in S$ be a subset of S that is used for fitness evaluation.

Let I_i be an individual in the population, and $f_{I_i}^{m_j}$ be the fitness of individual I_i when evaluated on scenario m_j. Then F_{I_i} is the "fitness vector" of I_i when evaluated on a subset of scenarios, given by $F_{I_i} = \{f_{I_i}^{m_1}, \ldots, f_{I_i}^{m_p}\}$.

We use standard deviation to calculate the volatility of the fitness (performance) of the individual across this range of scenarios:

$$\sigma_{I_i} = \sqrt{\frac{1}{p}\sum_{j=1}^{p}(f_{I_i}^{m_j} - \overline{F_{I_i}})^2} \qquad (1)$$

where: $\overline{F_{I_i}}$ = mean of F_{I_i}, given by $\frac{1}{p}\sum_{j=1}^{p} f_{I_i}^{m_j}$

The "Volatility-Adjusted Fitness" (VaF) of an individual is now defined as the mean fitness divided by volatility:

$$VaF_{I_i} = \frac{\overline{F_{I_i}}}{\sigma_{I_i}} \qquad (2)$$

3.1 MELG

For MELG we use a variation of the three-dataset methodology as outlined in [23, 6]. In our version of this methodology, the *training set* is used to evaluate the fitnesses of individuals in $n-1$ generations; elitism ensures that the best-of-generation individuals survive to the last generation, and the individuals in the last (n^{th}) generation are tested against a different *in-sample volatility set*; a best-of-run individual is selected and its quality is assessed using yet another different *out-of-sample validation set.*

Note that a possible drawback of this methodology is that, where data samples are limited, either the training set or the out-of-sample validation set must be smaller than it would be in a two-dataset methodology. However, in our variation of the methodology the *in-sample volatility set* is a vector of subsets of the initial *training set.* We also choose to set the fitness vector to be $F_{I_i} = \{f_{I_i}^{m_0}, f_{I_i}^{m_1}, \ldots, f_{I_i}^{m_p}\}$, where $f_{I_i}^{m_0}$ is the fitness of the individual previously calculated in the $n-1^{th}$ generation — thus, the fitness vector contains information about fitness on the whole training set treated as a single unit, as well as fitness on each of the scenarios.

Our methodology permits a more direct comparison with SGP, since we know that both SGP and MELG have been given identical training data — what is different is the way in which that data is presented to the population.

3.2 MEVO

The MEVO algorithm differs from MELG in that it uses a two-dataset method: the *in-sample volatility set* used in MELG is used not only in the final generation, but in every generation. The second dataset is therefore the out-of-sample validation set V.

Thus, evolutionary selection is based on the volatility adjusted fitness VaF_{I_i}, which is calculated from $F_{I_i} = \{f_{I_n}^{m_1}, \ldots, f_{I_n}^{m_n}\}$ (see above). F_{I_i} can be thought of as an "intermediate" fitness vector for each individual, and VaF_{I_i} is the "real" fitness of an individual. Note that MEVO is *not* exposed to the entire training set ($f_{I_n}^{m_0}$); we only expose it to the extreme scenarios. This might be thought to put MEVO at a disadvantage because it does not have access to as much information as MELG or SGP, but in early trials we discovered that using the entire training set as another scenario led to poor results.

4. HEDGE FUND SIMULATION

To test the two new algorithms, we simulate a long/short market-neutral hedge fund of Malaysian equities. We choose the Malaysian market because it (in common with other emerging markets) is particularly volatile. The GP system evolves a non-linear equation that uses market data to determine whether a single stock should be selected to buy, or to sell.

4.1 System Overview

Our test system comprises a GP system coupled with an investment simulator. The coupling between the two is the fitness function — the investment simulator is called each time the GP system needs to determine the fitness of an individual, at which point the individual is used to control the simulation of an hedge fund of Malaysian stocks. The simulator is applied to training data giving monthly prices and other factors. Monthly returns on investment are calculated, and at the end of each simulated year the Sharpe ratio [26] is calculated.

Fitness

The fitness f for an individual is the Sharpe ratio [26], given by Equation 3.

$$Sharpe\ Ratio = \frac{\overline{x_i} - RFR_i}{\sigma_i} \tag{3}$$

In Equation 3, $\overline{x}_i$ is the average monthly Return on Investment (ROI) over the sub-period i, σ_i is the standard deviation of monthly ROIs over the sub-period i, and RFR_i is the average monthly Risk Free Rate for sub-period i. We set RFR_i to $0.00\dot{3}$ for all i (equivalent to 4% per annum).

Note that we have chosen *not* to use a multiple-objective approach to fitness evaluation. At an early stage we experimented with using two objectives (high ROI and low volatility) but the system performed poorly. In financial investment ROI and volatility are very closely linked (they are not properly independent objectives); the result was that the non-dominated set was very small and this adversely affected evolution, causing the system to converge on a local optimum with poorer performance than the solution found using the Sharpe Ratio as a single objective.

4.2 The Investment Simulator

We simulate a market-neutral long/short hedge fund of Malaysian equities. The fund focuses on a basket of 33 Malaysian stocks, which it can buy ("go long") or sell (even if it doesn't own any — "go short"). Since all 33 stocks are quite well correlated, the market-neutral strategy simply entails buying the profitable stocks and selling (short if necessary) those stocks that are performing poorly.[3]

The training data is monthly prices (and other technical and fundamental data) over a period of 71 months. All trading occurs at the beginning of each month and the resulting stock mix is held for the duration of the month. At the beginning of each month, the simulator uses the individual provided by the GP system as a stock selection model that quantitatively measures the attractiveness of each stock; this model is a non-linear combination of technical and fundamental factors to predict the return expectation for each stock over a 4-week forward horizon.

[3]A *contrarian* strategy might do the opposite — sell the high stocks and buy the low stocks, on the expectation that mean-reversion will occur and the high stocks will fall while the low stocks will rise.

Table 1: Description of Factors

1.	Closing stock price on 1st day of a month
2.	Closing stock price on last day of a month
3.	12-month MACD: moving average convergence and divergence
4.	capitalisation = (number of shares) × (stock price (c))
5.	ROE = (net income) ÷ (shareholders' equity)
6.	ROE(this year) − ROE(prev. year)
7.	((total debt) ÷ (common equity)) × 100
8.	(sum of last 12-months of cash dividends) ÷ (stock price (c))
9.	(last 6 months' trailing earnings per share - prev. last 6 trailing earning per share) ÷ (absolute prev. last 6 months trailing earning per share)
10.	as above (replace 6 with 12)
11.	as above (replace 12 with 36)
12.	The rate of change in the reported last 12-month earnings per share over the three year time interval terminating on the date of the last interim period for which earnings were announced
13.	(last 12-month trailing earnings per share) ÷ (closing market price)
14.	(historical book value per share) ÷ (closing monthly market price)
15.	(cash earnings per share) ÷ (closing market price)
16.	One month dollar price change
17.	One year dollar price change
18.	(current year's net sales or revenue - previous year's net sales or revenue) ÷ (previous year's net sales or revenue)
19.	(last 12-month trailing earnings per share - last 12-month dividend per share) ÷ (last year's book value per share)

For each month, we apply the stock selection model to the current month data — this is a table per stock with 19 factors (see Table 1) and 7,680 data points. A return prediction is assigned to each stock.

The stocks are grouped into 4 market sectors and within each sector all stocks are ranked according to the expected return. The portfolio simulator then makes the following fund management decisions:

- The long/short portfolio is both dollar neutral and sector neutral. Thus, at all times, 24 stocks are maintained in the portfolio with 12 long positions and 12 short positions equally distributed across all the sectors. According to the ranking, the top 3 stocks in each sector become the top fractile and the bottom 3 become the bottom fractile. The top fractile of each sector and the bottom fractile of each sector are chosen to hold long positions and short positions respectively in the portfolio.

- Sectors are equally weighted and each stock is given equal weight in the portfolio. Thus, each position accounts for approximately 4% of total portfolio value.

- CFDs (Contract for Differences) are used instead of conventional shares to trade on stocks. We assume 20% notional trading requirement (margin), 0.25% trading commission, and 5% financing rate.

Table 2: GP Parameter Settings

Population size (N)	1000
Method of generation	Ramped half and half
Function set	{+, -, *, /, Exp}
Terminal set	18 firm-specific factors
Selection scheme	Fitness proportionate selection
Criterion of fitness	Monthly Sharpe ratio
Trees generated by elitism	10 (1%)
Trees generated by crossover	950 (95%)
Trees generated by mutation	40 (4%))
Termination criterion	100-generation evolution
Max. depth initial generation	6

At the end of each month, all of the positions held in the portfolio are closed and the profit or loss of the portfolio during the month is calculated. At the beginning of the next monthly trading cycle, the simulator updates the expected return based on the new "current" data and a new desired long/short portfolio is formed.

5. EXPERIMENT

Our primary research question is: *"are the best-of-run individuals from the two new systems more robust than the best-of-run individual from SGP when exposed to a volatile and previously unseen environment?"*

Our experiment compares the performance of all three systems: SGP, MELG and MEVO. The basic GP parameter settings for the three systems are identical, as given in Table 2.

5.1 Data

All three systems use an Investment Simulator that has an investment universe of 33 Malaysian stocks. The training data for all three systems comprises time-series financial data for the 33 stocks taken from the period 31st January 1999 to 31st December 2004.

SGP and MELG use a training data set of financial time-series data taken from the period 31st January 1999 to 31st December 2004 (71 months).

For MELG (last generation only) and MEVO, the following three scenarios were chosen:

1. Bull market: 31/05/2003 to 31/12/2004 (19 months);
2. Bear market: 31/01/2000 to 31/05/2001 (16 months);
3. Volatile market: 31/01/1999 to 31/03/2000 (14 months).

Figure 1 shows the overall market index for Malaysian stocks, and a non-weighted portfolio index of the 33 investment stocks, for the overall period under study. It also indicates the three scenario periods (bull, bear and volatile) and the validation period.

5.2 Out-of-Sample Validation

All three systems are validated on a previously unseen "out of sample" data set, comprising time-series financial data for the 33 stocks taken from the period 31st July 1997 to 31st December 1998. During this period the Malaysia stock market suffered great volatility including both the highest and lowest monthly returns in the entire period under study. From May 1998 to October 1998, the stock index lost more than 42%. Then from November 1998 onwards, there was a remarkable performance from the market index, rising 23.3% in November.

We have deliberately chosen this period as a real test of robustness of individuals in a dynamic and hostile environment. One expects episodes of extreme volatility in world stock markets, and in emerging markets in particular. A successful hedge fund stock selection model must be robust — be able to perform in both (extreme) up and down markets.

For the out-of-sample validation, we performed 25 complete training runs (each run being seeded with a different random number) of each of the three systems (SGP, MELG, and MEVO) and the best-of-run individual was selected from the final generation of each run.

The selected individuals were then validated on the previously unseen data; the results of the 25 runs are discussed in the following section.

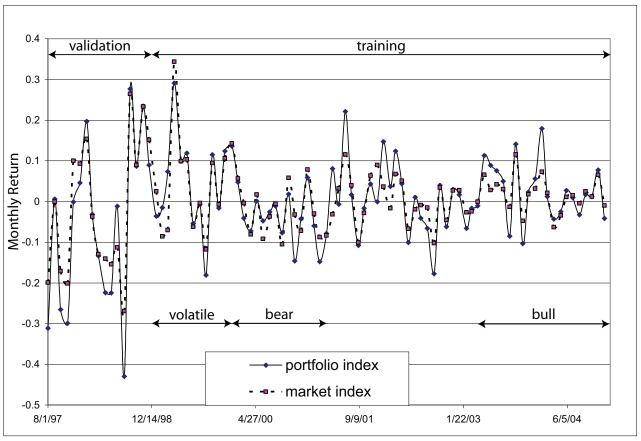

Figure 1: Market and portfolio indices (fractional monthly returns, 31st July 1997 to 31st December 2004), scenarios and validation period.

6. DISCUSSION OF RESULTS

Figure 2 shows the mean monthly returns (over 25 runs) on the validation data for all three systems (SGP, MELG and MEVO), together with the porfolio index returns. The portfolio index (constructed from the stocks in which our simulator invests) shows considerable volatility — it is more volatile than the overall market index seen in Figure 1, and so beneficial effects displayed by our GP system cannot be due solely to "cherry-picking" the least volatile stocks. SGP is not very volatile, but neither does it make much profit. Both MELG and MEVO appear to be adept at avoiding losses yet still able to make good gains in positive months. Figure 3 show vividly the difference between the large cumulative losses of the portfolio index compared with the cumulative gains of MELG and MEVO.

Figure 4 gives another view of the mean monthly returns by plotting the frequency distributions of returns in the validation period. The portfolio index (dashed) is very volatile, whereas all three GP systems are much less volatile (though with significant positive fat tails).

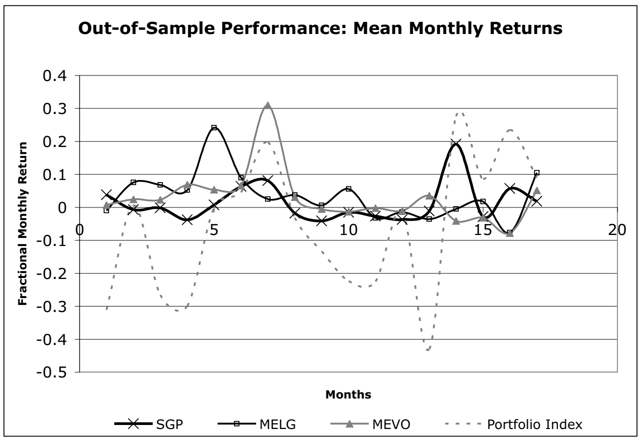

Figure 2: Mean monthly fractional returns.

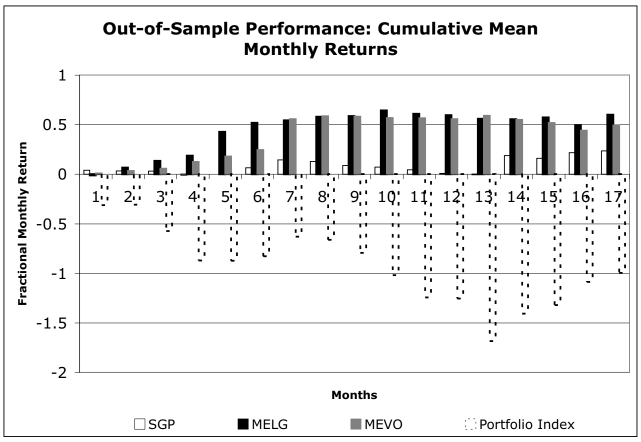

Figure 3: Cumulative monthly fractional returns.

6.1 Robustness

So what does this tell us about "robustness", and how do we measure it? Simplistically, we might take robustness to be synonymous with "low variance" — i.e. the performance of the individual does not alter much, despite the extreme volatility of the market environment. However, in practice we have a much more exacting requirement: it is not helpful to an investor to know that an individual robustly (i.e. with low variance) makes a loss regardless of the market! A much more helpful measure is to know that the individual combines two qualities of (i) high return on investment and (ii) low variance in the face of extreme volatility.

In Section 1 we stated our three measures of robustness:

1. when exposed to an out-of-sample volatile validation data-set, a more robust solution will have a lower standard deviation of returns, while the returns themselves do not decrease; or

2. when exposed to an out-of-sample volatile validation data-set, a more robust solution will have higher returns while the standard deviation does not increase; or

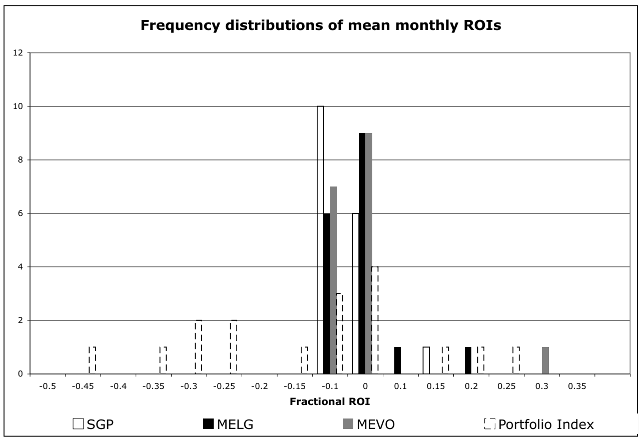

Figure 4: Frequency distributions of mean monthly fractional returns.

3. when exposed to an out-of-sample volatile validation data-set, the mean return per unit of risk of a more robust solution will not significantly reduce from that measured during training.

The performance of our three systems, using robustness measures 1 and 2 above, are illustrated in Figure 5, which shows standard deviation plotted against returns. Specifically, the figure plots returns in excess of the risk free rate, and we have added data for the portfolio index and for a popular non-genetic technical strategy (using Moving Average Convergence Divergence (MACD) to select stocks). The portfolio index is shown to be not at all satisfactory, with both low returns and high standard deviation; the MACD approach performs much better than the portfolio index, but not as well as any of the three GP systems. In terms of robustness:

1. The three GP systems and MACD all have similar standard deviations (a ranked T-test indicates no significant difference in the GP distributions) and so by this measure no one system is more robust than another.

2. By contrast, the three GP systems and MACD differ in their returns while their standard deviations do not differ, so by measure 2 they are not equally robust. In order (from least to most robust) we have MACD, then SGP, the MEVO and finally (the best) MELG. We further quantify this below.

Fund managers use a very similar approach to our robustness measures 1 and 2 — they use the Sharpe Ratio [26] (see Section 4.1) which determines the ROI (in excess of the risk free rate) per unit of risk (given by the standard deviation). Since the standard deviations in this case are the same, a Sharpe Ratio comparison also provides a quantitative comparison of returns and thus of our robustness measure 2. Therefore, we have calculated the Sharpe Ratios (across 25 runs) for each of the three GP systems.

Comparison of the Sharpe Ratio distributions shows that all three systems achieve substantially better results than the portfolio index (as expected from Figure 5) and a ranked

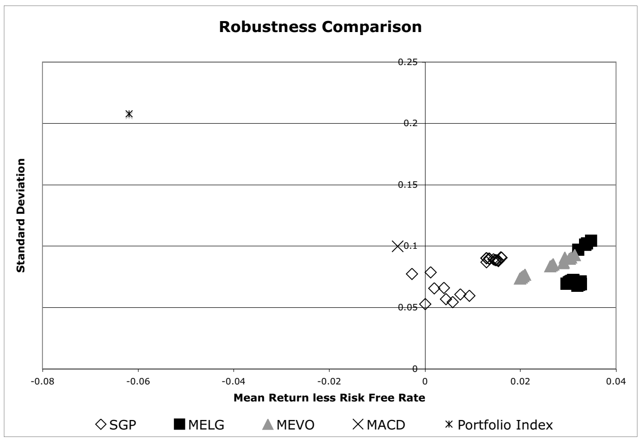

Figure 5: Robustness comparison.

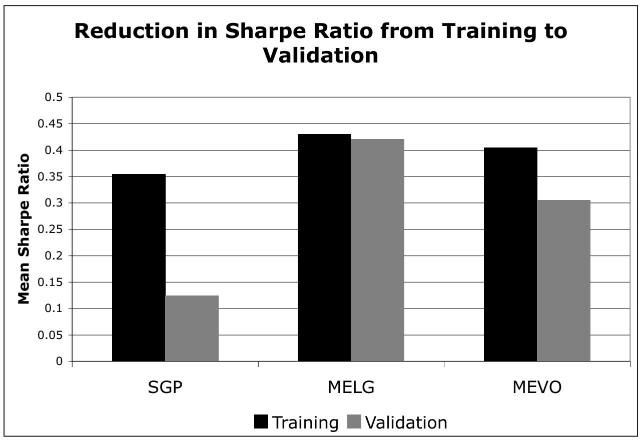

Figure 6: Robustness measure: drop in Sharpe Ratio.

T-test comparison of the Sharpe Ratios indicates a statistically significant difference between all three systems. The p-values (the probabilities that two compared distributions are from the same population) are presented in Table 3. The means of the Sharpe Ratio distributions are: 0.125 (SGP), 0.305 (MEVO), and 0.421 (MELG) — MELG is substantially the most robust system of the three.

Table 3: Summary of Ranked T-test (p-values)

Compare SGP with MELG:	4.01×10^{-16}
Compare SGP with MEVO	4.13×10^{-16}
Compare MEVO with MELG	4.62×10^{-9}

Our third measure of robustness determines how much the mean return per unit of risk reduces when moving from the training set to the validation set. This is shown in Figure 6. The percentage reductions in Sharpe Ratio (and associated p-values from a ranked T-test) were 65% for SGP (1.4×10^{-11}) , 25% for MEVO (3.1×10^{-8}) and just over 2% for MELG (0.92), indicating a substantial robustness advantage for MELG.

7. SUMMARY AND CONCLUSION

In a volatile and unforgiving financial environment, the use of carefully selected scenarios of extreme market behaviour during GP training can produce more robust trained individuals than those produced by a standard GP system.

We investigated two different mechanisms for presenting the extreme scenarios to the population:

1. the first (Multiple-scenario Evaluation in Last Generation — MELG) used a complete set of training data for $n-1$ generations and then used three extreme scenarios (taken from the complete training set) to test the individuals in the final generation;

2. the second (Multiple-scenario Evolution — MEVO) did not see the complete set of training data, but instead the three extreme scenarios were used to determine fitness throughout all generations.

In Section 3 we introduced Volatility-Adjusted Fitness as a mechanism for assessing the robustness of an individual when exposed to multiple training scenarios; this is the mean fitness across all scenarios divided by the root mean squared errors from the mean. The fitness of an individual assessed on a single scenario was given by the Sharpe ratio [26].

Our system used a GP system to evolve a non-linear factor model for stock-picking, coupled with an Investment Simulator that modelled a long-short, market-neutral, sector-neutral hedge fund trading Contracts for Difference (CFDs) in the highly volatile Malaysian stock market. Historical stock data (both technical and fundamental) was used from the period 1997–2004.

We introduced three practical measures of robustness: the first two compared volatility against returns on investment, and the third compared the Sharpe Ratio during training with the Sharpe Ratio during validation (see Section 1). Experiments were run on three GP systems (MELG, MEVO and a “standard” GP system — SGP) with 25 runs of each, and comparisons were made with both a portfolio index and a non-genetic simple technical analysis for stock picking.

Although robustness measure 1 showed no difference between the three GP systems, statistical analysis of measures 2 and 3 indicated overwhelmingly that MELG provides the most robust individual, with SGP being the least robust. All three GP systems were shown to have better performance than the non-genetic technical analysis, and this in turn performed very much better than the portfolio index.

Further work in this area includes extending the experiment to a larger universe of stocks; combining the scenarios mechanism with other robustness-enhancing techniques; and investigating better ways to present the extreme scenarios to the population. For example, we are trying to gain a better understanding of why using the complete training set as a separate scenario for MEVO did not give good results.

8. ACKNOWLEDGMENTS

The authors thank Dr Gerard Vila for suggestions and discussions, SIAM Capital for financial support, and Reuters for access to financial data. We also thank the anonymous referees for their helpful comments.

9. REFERENCES

[1] V. S. Aragon and S. C. Esquivel. An evolutionary algorithm to track changes of optimum value locations in dynamic environments. *Journal of Computer Science and Technology*, 4(3):127–134, 2004.

[2] T. F. Bersano-Begey and J. M. Daida. A discussion on generality and robustness and a framework for fitness set construction in genetic programming to promote robustness. In J. R. Koza, editor, *Late Breaking Papers at the 1997 Genetic Programming Conference*, pp. 11–18, Stanford Bookstore, 1997.

[3] C. P. Bowers. Formation of modules in a computational model of embryogeny. In *The 2005 IEEE Congress on Evolutionary Computation*, volume 1, pages 537–542, 2005.

[4] J. Branke. Creating robust solutions by means of evolutionary algorithms. In *PPSN V: Proceedings of the 5th International Conference on Parallel Problem Solving from Nature*, pp. 119–128, Springer, 1998.

[5] J. M. Fitzpatrick and J. J. Grefenstette. Genetic algorithms in noisy environments. *Machine Learning*, 3:101–120, 1988.

[6] C. Gagné, M. Schoenauer, M. Parizeau, and M. Tomassini. Genetic programming, validation sets, and parsimony pressure. In *Proceedings of the 9th European Conference on Genetic Programming*, LNCS 3905, pp. 109–120, Springer, 2006.

[7] A. Gierer, S. Berking, H. Bode, C. N. David, K. Flick, G. Hansmann, H. Schaller, and E. Trenkner. Regeneration of hydra from reaggregated cells. *Nature New Biology*, 239:98–101, 1972.

[8] U. Hammel and T. Bäck. Evolution strategies on noisy functions: How to improve convergence properties. In *PPSN III: Proceedings of the International Conference on Evolutionary Computation. The Third Conference on Parallel Problem Solving from Nature*, LNCS 866, pp. 159–168, Springer, 1994.

[9] T. Haynes, R. Wainwright, S. Sen, and D. Schoenefeld. Strongly typed genetic programming in evolving cooperation strategies. In L. Eshelman, editor, *Genetic Algorithms: Proceedings of the Sixth International Conference (ICGA95)*, pages 271–278, Pittsburgh, PA, USA, 15-19July 1995. Morgan Kaufmann.

[10] J. Herrmann. A genetic algorithm for minmax optimisation problems. volume 2, pp. 1099–1103, 1999.

[11] M. A. Huynen, P. F. Stadler and W. Fontana, Smoothness within ruggedness: The role of neutrality in adaptation. *Proceedings of the National Academy of Sciences of the United States of America (PNAS)*, 93(1):397–401, 1996.

[12] T. Ito, H. Iba, and M. Kimura. Robustness of robot programs generated by genetic programming. In *Genetic Programming 1996: Proceedings of the First Annual Conference*, MIT Press, 1996. 321–326.

[13] E. Jordaan, A. Kordon, L. Chiang, and G. Smits. Robust inferential sensors based on ensemble of predictors generated by genetic programming. In *Parallel Problem Solving from Nature - PPSN VIII*, LNCS 3242, pp. 522–531, Springer-Verlag, 2004.

[14] H. Kitamo. *Foundations of systems biology.* MIT Press, ISBN 0-262-11266-3, 2001.

[15] I. Kuscu. Promoting generalization of learned behaviours in genetic programming. In *Fifth International Conference on Parallel Problem Solving from Nature*, LNCS 1498, pp. 491–500, Springer-Verlag, 1998.

[16] I. Kuscu. Generalisation and domain specific functions in genetic programming. In *Proceedings of the 2000 Congress on Evolutionary Computation CEC00*, Vol. 2, pp. 1393–1400, IEEE Press, 2000.

[17] I. Kushchu. Genetic programming and evolutionary generalization. *IEEE Transactions on Evolutionary Computation*, 6(5):431–442, October 2002.

[18] R. Lowenstein. *When Genius Failed.* Fourth Estate, 2002.

[19] B. L. Miller and D. E. Goldberg. Genetic algorithms, selection scheme, and the varying effect of noise. *Evolutionary Computation*, 4(2):113–131, 1996.

[20] J. F. Miller. Evolving a self-repairing, self-regulating, french flag organism. pages 129–139, 2004.

[21] F. W. Moore and O. N. Gacia. A new methodology for reducing brittleness in genetic programming. In *Proceedings of the National Aerospace and Electronics 1997 Conferences, NAECON-97*, 1997.

[22] V. Nissen and J. Propach. On the robustness of population-based versus point-based optimisation in the presence of noise. *IEEE Transactions on Evolutionary Computation*, 2(3), 1998.

[23] L. Panait and S. Luke. Methods for evolving robust programs. In *Genetic and Evolutionary Computation — GECCO 2003*, volume 2724 of *LNCS*, pages 1740–1751. Springer, 2003.

[24] C. W. Reynolds. An evolved, vision-based behavioral model of coordinated group motion. In *From Animals to Animats (Proceedings of Simulation of Adaptive Behaviour).* MIT Press, 1992.

[25] J. Rosca. Generality versus size in genetic programming. In *Genetic Programming 1996: Proceedings of the First Annual Conference*, pp. 381–387. MIT Press, 1996.

[26] W. F. Sharpe. The sharpe ratio. *J. Portfolio Management*, 21:49–58, 1994.

[27] T. Soule. Operator choice and the evolution of robust solutions. In R. L. Riolo and B. Worzel, editors, *Genetic Programming Theory and Practise*, chapter 16, pages 257–270. Kluwer, 2003.

[28] T. Soule, R. B. Heckendorn, and J. Shen. Solution stability in evolutionary computation. In *ISCIS XVII International Symposium On Computer and Information Sciences*, pp. 237–241, CRC Press, 2002.

[29] S. Tsutsui and A. Ghosh. Genetic algorithms with a robust solution searching scheme. *IEEE Transactions on Evolutionary Computation*, 1(3):201–208, 1997.

[30] A. Wagner. *Robustness and Evolvability in Living Systems.* Princeton University Press, 2005.