# Failure of Genetic-Programming Induced Trading Strategies: Distinguishing between Efficient Markets and Inefficient Algorithms

Shu-Heng Chen[1] and Nicolas Navet[1,2]

[1] AI-ECON Research Center, Department of Economics, National Chengchi University, Taipei, Taiwan 11623, chchen@nccu.edu.tw
[2] LORIA-INRIA, Campus-Scientifique, BP239, F-54506 Vandoeuvre, France, nnavet@loria.fr

Over the last decade, numerous papers have investigated the use of Genetic Programming (GP) for creating financial trading strategies. Typically, in the literature, the results are inconclusive but the investigators always suggest the possibility of further improvements, leaving the conclusion regarding the effectiveness of GP undecided. In this paper, we discuss a series of pretests aimed at giving more clear-cut answers as to whether GP can be effective with the training data at hand. Precisely, pretesting allows us to distinguish between a failure due to the market being efficient or due to GP being inefficient. The basic idea here is to compare GP with several variants of random searches and random trading behaviors having well-defined characteristics. In particular, if the outcomes of the pretests reveal no statistical evidence that GP possesses a predictive ability superior to a random search or a random trading behavior, then this suggests to us that there is no point in investing further resources in GP. The analysis is illustrated with GP-evolved strategies for nine markets exhibiting various trends.

## 1 Motivation and Introduction

Computational intelligence techniques such as genetic programming [1], with their continuous advancement, persistently bring us something positive to expect, and incessantly push the application domain to more challenging issues. However, sometimes, the costs and benefits of using these advanced CI techniques are uncertain. Usually the benefits are not assured, while the costs are immediate. On the one hand, the CI techniques are frequently used as intensive search algorithms, which are inevitably computationally demanding, and take up a great amount of computational

[1] Although, in this paper, we solely focus on genetic programming, the general ideas and some specific implementations should also be applicable to other computational intelligence techniques used to induce trading strategies.

resources. On the other hand, whether or not there is a needle in the haystack remains dubious. For example, in the financial application domain, the lack of such a needle may be due to the efficient markets hypothesis or the no-arbitrage condition. Certainly, if such a needle does not exist at all, then all efforts are made to no avail. Given this asymmetry between costs and benefits, it would be economical, at the first stage, to test for the existence of such a needle before a fully-fledged version of the search is applied. We refer to this procedure as a *pretest*.

The pretest procedure proposed here is in a sense similar to the pretests used in econometrics where the estimator of an unknown parameter is chosen on the basis of the outcome of a pretest [9]. Pretesting, also known as "data-snooping" in finance, classically serves to select the right model that will be used later on for forecasting purposes (see [5, 20]). More broadly, pretesting can be considered to be a practice of a sequential decision-making process, which is used when the decision involves a great deal of uncertainty, and the costs of making a wrong decision are huge. [2] In this case, at the first stage, we would like to expend some limited resources on probing into gaining some initial information, e.g., the distribution of a very uncertain environment, while in the later stages, we will make our decision based on the gauged distribution.

The reasoning behind prestesting is very intuitive, and [11] is the first to apply this idea to the financial application of genetic programming (GP). Reference [11] proposed a measure known as the $\eta$ statistic. The $\eta$ statistic is a measure of predictability obtained by comparing the predictions regarding the actual data and the shuffled data.[3] Basically, by using a simple (vanilla) version of GP, one can first gauge the predictability based on $\eta$. When $\eta$ is low or close to zero, it indicates that there is nothing to forecast. So, the use of fully-fledged GP is not advised. The virtue in doing this is to distinguish *two kinds of possibilities* when we see the failure of an initial attempt based on simple GP. First, the series itself has nothing to forecast; second, GP has not been used appropriately. Understanding this distinction can result in big differences in our second stage of the decision. In the former case, we may simply give up any further search to avoid wasting resources. In the latter case, we should keep on exploring different deliberations of GP to search for potential gains before a final conclusion can be reached. In either case, we have a clear-cut situation. However, when a pretest is absent, we become less conclusive: we are no longer sure whether the problem is due to the non-existence of the needle, or the improper use of GP.

[2] The problem of sequential decision making under incomplete knowledge has been studied by researchers in various fields, such as optimal control, psychology, economics, and game theory.

[3] The $\eta$ statistics make use of *surrogate data*, that is, the data sharing statistical properties of the data under study but not the property that is tested for. Here, the property investigated is temporal dependence and, thus, by shuffling the original time series, the temporal dependencies, if any, are lost. The interested reader might refer to [16] and [18] for a good starting point on the use of surrogate data.

Unfortunately, in most financial trading applications of GP, a pretest has been largely neglected.[4] We think that this negligence may give rise to many inconclusive results. Typically, what happens is that the results from using GP are not very convincing, but the investigators always suggest directions for further improvement, leaving the actual conclusion regarding the effectiveness of GP undecided. Therefore, this study attempts to provide a practical pretesting procedure aimed at reducing the number of cases where the conclusion is inconclusive.

Needless to say, there are various ways of implementing different types of pretesting. For example, the $\eta$ statistic mentioned above can be used as a pretest, as [11] did, but that is mainly applied to forecasting time series. That a series is to a certain extent predictable does not necessarily imply that we can develop profitable trading strategies. For example, the predictability horizon might be too short, the fluctuation might not be volatile enough to cover the round-trip transaction costs or, simply, the right trading instrument might not be available (e.g., no short selling in a downward oriented market) or else they are some regulation and rules (e.g., the "uptick rule" makes intraday trading with short selling more difficult). Consequently, the literature on forecasting with GP (e.g., [12, 17] and [6]) and the literature on trading with GP (e.g., [1, 14, 21] and [4]) are usually separated. Therefore, in this paper, we attempt to develop pretest procedures that are more suitable for trading purposes. However the correlation between the predictability[5] of a time series and the profitability of GP induced rules, and more generally of any trading strategies, is an intriguing and still open question, whose answer[6] constitutes, in our view, a major step towards efficient market timing decision tools.

More precisely, we will propose here several different styles of pretests, which when put together can help us decide whether there are hidden patterns to be discovered and whether GP is properly designed to do the job. The essential idea underlying all proposed pretests is to compare the performance of GP with random trading strategies or behavior. However, as we shall see in Sect. 2, just making trading strate-

[4] This may not be completely so. In fact, most earlier studies selected a risk-free investment (e.g., treasury bills) or, most often, the buy-and-hold strategy as the benchmark. However, the conclusion that "GP performs better than buy-and-hold in a bearish market and worse in a bullish market" is often found in the literature. However, nothing different can be expected since buy-and-hold is the worst possible strategy in a steadily decreasing market and the best possible strategy in a steadily increasing market. This shows the limits of choosing buy-and-hold as a benchmark.

[5] Numerous metrics, emerging from the fields of information theory, the study of dynamical systems and algorithmic complexity or statistics, have been devised to quantify the predictability of a system observed by the data it produces. One can mention the Lyapunov exponent, which is a measure of the rate of divergence of nearby trajectories and thus an indication of the short-term predictability, the Shannon entropy which measures the diversity of the data produced or the Grassberger-Crutchfield-Young statistical complexity which informs us of the amount of information which is relevant to the system's dynamic. The reader interested in predictability measures can refer to [2] and [19] for a comprehensive survey.

[6] Of particular interest is the work of [10] which is a significant step in that direction.

gies or trading behavior arbitrarily random is not sufficient to provide a fair and informative comparison. To do so, some constraints are expected, and the interesting point is how to impose these constraints properly.

The rest of the paper is organized as follows. Section 2 provides a detailed formulation of the four pretests. The first three are concerned with the trading strategies, whereas the last one is concerned with the trading behavior. Normally, trading behavior comes from trading strategies, and they cannot be separated; however, when randomness is introduced, differences between the two may arise. In particular, in the vein of algorithmic complexity, random trading strategies can imply trading behavior actually using knowledge, while random trading behavior presumably excludes such a possibility. We, therefore, intentionally distinguish between the two by referring to the former as *zero-intelligence strategies*, and the latter as *lottery trading*. Section 3 discusses how to use these proposed tests together to make a better judgment given the initial results we have. Section 4 illustrates the proposed pretests based on the real data and the experimental designs detailed in the appendix. Section 5 gives the concluding remarks.

## 2 Pretests: Description and Rationale

In this section, we describe a series of 4 pretests and discuss their purpose and implementation. Of the 4 pretests, we highlight 2 that are of particular interest and, as shown in Sect. 3, enable us to gain complementary knowledge on the data under study and on the efficiency of the GP's implementation. In the following, we consider GP with a validation stage before the actual testing of the out-of-sample data. Validation means that the best rules induced on the training interval are further selected on the unseen data, i.e., the validation period, before being applied out-of-sample. The validation step is a device to fight overfitting that has been widely used in earlier GP work (see for instance [1, 15]).[7] Note that our proposals, except for pretest 1 which explicitly requires validation, remain valid as they do for GP without the validation step.

### 2.1 GP versus Equivalent Intensity Random Search

The basic idea here is to compare the outcome of GP with an *equivalent intensity random search*. We say that two search algorithms are equivalent in terms of search intensity if their execution leads to the evaluation of the same number of *distinct* trading strategies for the training data. For instance, let us consider GP with the parameters chosen for this study: a population of 500 individuals evolved over 100 generations. In the first approximation, the equivalent random search (ERS) would consist of evaluating 50,000 randomly created solutions. In practice, search algorithms sometimes rediscover identical solutions over the course of their execution. This can

[7] The actual effectiveness of validation in this context is, however, still an open question. See [4] and [3].

simply be detected by keeping track of all created individuals since the beginning of the execution, and in doing so useless fitness evaluations can be skipped, which actually saves computing time when the fitness function is rather time-consuming. Since, computationally speaking, what is preponderant in our context is the fitness evaluation, we impose the restriction that our definition of equivalent search intensity only accounts for unique individuals, *i.e.,* individuals that require evaluation. We consider two solutions to be distinct if their expression is *syntactically different*[8], in our GP context, if the trees representing the programs are different.

The three following pretests compare GP with a random search both with and without a training and validation stage. In random search, the biologically inspired evolution process of GP is simply replaced by the creation of solutions at random. Since with random search the strategies do not benefit from the "intelligence" resulting from the evolution and learning process[9], we dub randomly created solutions as *zero-intelligence trading strategies.*

For each pretest $i$, we formulate the null hypothesis $\mathcal{H}_{i,0}$ that GP does not outperform the technique it is compared with at pretest $i$, where the alternative hypothesis is denoted by $\mathcal{H}_{i,1}$. The experiments will provide us with the answer to whether $\mathcal{H}_{i,0}$ should be rejected in favor of $\mathcal{H}_{i,1}$ or not. As usual, the chosen significance level of the test enables us to finely control the probability to falsely reject the null, that is in our case to come wrongly to the conclusion that GP is more effective than the technique it is compared with.

### Pretest 1: GP versus Equal Search Intensity Random Search - both with a Training and a Validation Stage.

The implementation of the random search strategy is straightforward: parameters of GP are set in such a way that only the initial generation, where individuals are created at random, is used. The size of the initial population is adjusted so that the resulting search intensity is identical to the one for the regular GP.

[8] Two individuals can be syntactically different while being equivalent in the sense that they always lead to the same trading decisions, and the equivalence could thus also be defined in terms of semantics. With symbolic simplification using rewriting rules and interval arithmetic on the function arguments, we could detect that some syntactically different individuals are in fact semantically identical. However, there is no way of making sure that all duplicates will be detected and the implementation of this procedure would be so complex and time consuming at run time that, in our opinion, a definition based on semantics would be of little practical interest. Alternatively, the equivalence in search intensity could be defined in terms of equivalent running time. However, there is such a difference in complexity between a fully-fledged GP implementation and random search that it is hard to imagine how we can ensure that the two implementations have been optimized in a similar manner, while a better implementation of GP may for instance may lead us to come to an opposite conclusion.

[9] Comparing GP with random search informs us regarding the effectiveness of the GP operators. Further meaningful information regarding this issue could be obtained by comparing regular GP with an implementation that would favor crossover among the less fit solutions ("breed-the-worst"), as suggested in [13].

- **Hypothesis $\mathcal{H}_{1,0}$ cannot be rejected:** the first explanation that can be envisaged is that, GP or not, there is nothing essential to be learned from the past. In that case GP would strongly "overfit" the training data, possibly explaining that in the same cases its out-of-sample performance is worse than that with a random search. This can be due to the market being efficient or because the training interval exhibits a time series pattern which is significantly different from the out-of-sample period.[10] Another explanation is that the GP machinery is not working properly, for instance due to a wrong choice in the composition of the function and terminal sets, because the parameters controlling the GP run are inappropriate (e.g., a search intensity that would be insufficient), or the genetic operators are unable to create better-than-random individuals.
- **Hypothesis $\mathcal{H}_{1,0}$ is rejected in favor of $\mathcal{H}_{1,1}$:** there may be something to learn from the past and GP, with the chosen parameters, may be effective in that task.

Rejecting $\mathcal{H}_{1,0}$ is of course a first indication of the efficiency of GP but, as we will see in Sect. 3, further investigation may provide additional information to answer that question and rule out mere luck.

### Pretest 2: GP versus Equal Search Intensity Random Search with a Training but without a Validation Stage.

Here, the best solutions found at random over the training interval are applied directly to the out-of-sample period. With regard to pretest 1, pretest 2 could give us some insight into how effective validation is as a device to fight against overfitting. However, since overfitting is unlikely to occur with random search, the rationale for using pretest 2 is unclear and it will not be further considered in this study. A more direct and effective way to evaluate the effect of the validation stage is simply to compare regular GP with and without validation.[11]

### Pretest 3: GP versus Equal Search Intensity Random Search both without a Training and without a Validation Stage

In pretest 3, the selection of the strategies for the training set is removed: a large number of random strategies are created and applied directly out-of-sample. The performance is evaluated as the average performance (e.g., average total return) over the set of random strategies. Comparing the outcome of pretest 3 with regard to pretest 1 and regular GP tells us something about how effective the selection process is, and the extent to which a top performing rule on the training and validation sets will keep on performing well out-of-sample. If strategies selected by GP or random search on the training and validation intervals have some predictive ability out-of-sample, this will provide us with evidence that there is something to learn from the past. It is

[10] In [4], experiments have consistently highlighted that when training and out-of-sample data sets are very "dissimilar", for instance if the market exhibits opposite trends, then there is little chance that GP will perform well out-of-sample.

[11] For instance, as in the case of [4] and [8].

worth pointing out that the randomness of the strategies here is constrained by the GP language: rules can only be made with GP functions/terminals organized according to the constructs of the language and its typing scheme. For instance, it may happen that the GP language is not sufficiently expressive to define a rule consisting of buying and selling every other period.[12] In the remainder of this study, we will consider pretest 4, presented in Sect. 2.2, that is similar in spirit to prestest 3, but is more random in the sense that it does not possess the bias in randomness induced by the GP language.

## 2.2 GP versus Lottery Trading

We refer to *lottery trading* as a strategy that would consist of making the investment decision randomly on the basis of the outcome of a random variable. In its simplest form, the random variable would follow a Bernoulli distribution where the parameter $p$ expresses the probability of taking a long position and $1 - p$ the probability of closing a long position or staying out of the market.

In our context, this requires refinement since we are interested in profitability and profitability takes into account transaction costs. Therefore, in order to allow a fair comparison with GP, we should make sure that the expected number of transactions for lottery trading is the same as for GP. We refer to the expected number of transactions per unit of time as the *frequency of a trading strategy*. Another important characteristic of a trading strategy is what we term its *intensity, i.e.* the number of periods where a position[13] "in the market" is held, over the length of the trading interval. We should also enforce lottery trading to have the same expected intensity as GP to avoid misleading results, for instance, in the case where, given its frequency, the intensity of lottery trading is not sufficient to cover the transaction costs with the volatility of the market under study.

We denote by $F_{GP}$ and $I_{GP}$, respectively, the average frequency and average intensity observed for the set of GP-evolved rules applied to the testing interval over all GP runs, and $N_{GP}$ is the number of transactions leading to $F_{GP}$. For the experiments made in the following, a sequence of investment decisions $S_{LT}$ resulting from lottery trading is generated at random according to the following procedure:

- the intensity for lottery trading, $I_{LT}$ , is uniformly chosen in $[I_{GP} \cdot (1 - \alpha), \min(1, I_{GP} \cdot (1 + \alpha))]$ where parameter $\alpha$ $(0 \leq \alpha \leq 1)$ introduces a con-

[12] Period refers to the granularity of time used for trading, for instance, one second or one day.

[13] Implicitly, we consider here the trading of a single instrument, e.g., an index, where two positions are possible at each point in time, i.e., be in or be out of the market if short selling is not possible, or with short selling as implemented in [4], holding a long position or a short position. These concepts can be extended to the case where there are three possibilities in each time period: holding a long position, holding a short position or staying out of the market. Similarly, intensity and the frequency of a strategy can be instantiated for each traded instrument.

trolled randomness.[14] In the first step, $S_{LT}$ is made of the '0' positions (*i.e.,* out of the market) followed by the block of '1' positions (*i.e.,* in the market) corresponding to $I_{LT}$,

- the number of transactions $N_{LT}$ is uniformly chosen in the set of integer values that are even[15] in interval $[N_{GP} \cdot (1 - \alpha), N_{GP} \cdot (1 + \alpha)]$. The block of '1' is subdivided at random in $N_{LT}/2$ sub-sequences and each sub-sequence is inserted at random inside the block of '0'. This design avoids the problem of overlapping among the '1' sub-sequences that may occur with other schemes.

We formulate the pretest comparing GP and lottery trading and denote by $\mathcal{H}_{4,0}$ the null hypothesis that GP does not outperform lottery trading while the alternative hypothesis is $\mathcal{H}_{4,1}$.

**Pretest 4: GP versus Lottery Trading**

Obviously, if GP is not able to outperform lottery trading, it gives strong evidence that GP will not be good at evolving effective trading strategies with the data at hand. In Sect. 3, we shall discuss this point in more detail.

## 3 What do the pretests tell us ?

The outcomes of the pretests provide us with answers to the following two questions: Is there something essential to learn on the training data that can be of interest for the out-of-sample period? Does the GP implementation show some evidence of effectiveness in that task? Clearly, before actually trading with GP evolved programs, these two questions must be answered with reasonable certainty; the rest of this section explains how pretests may help in that regard.

### 3.1 Question 1: Is there something to learn ?

The null hypothesis $\mathcal{H}_{4,0}$ corresponding to pretest 4 has been presented in Sect. 2.2. We introduce pretest 5 that will be used in conjunction with pretest 4.

[14] Parameter $\alpha$ is intended to reproduce the variability of intensity and frequency observed over the sample of GP runs that lottery trading is compared with. In the simplest form presented here, this is implemented as a parameter $\alpha$ which is unique for intensity and frequency. It is of course possible to refine this scheme by individualizing the parameter for intensity and frequency, or by drawing at random the values of $I_{LT}$ and $N_{LT}$ according to the empirical distributions of intensity and frequency encountered over the sample of GP runs. This latter procedure is especially meaningful when the empirical distributions of intensity and frequency in GP significantly differ from the uniform distribution that is implicitly assumed here.

[15] $N_{LT}$ has to be even since a "buy" transaction is followed by a sell transaction and no positions are left open.

### Pretest 5: Equivalent Intensity Random Search with Training and Validation versus Lottery Trading

Here, we compare lottery trading to a random search with training and validation, and a search intensity equivalent to the one used for GP in pretest 4. The null hypothesis $\mathcal{H}_{5,0}$ is that the equivalent intensity random search does not outperform lottery trading for the out-of-sample data. Depending on the validity of $\mathcal{H}_{4,0}$ and $\mathcal{H}_{5,0}$, we can draw the conclusions that are summarized in Table 1:

| | $\mathcal{H}_{4,0}$ | $\mathcal{H}_{5,0}$ | **Interpretation** |
|---|---|---|---|
| case 1 | $\neg$**R** | $\neg$**R** | there is evidence that there is nothing to learn |
| case 2 | **R** | $\neg$**R** | there may be something to learn (weak certainty) |
| case 3 | **R** | **R** | there is evidence that there is something to learn |
| case 4 | $\neg$**R** | **R** | there may be something to learn (weak certainty) |

**Table 1.** Information drawn from the outcomes of pretest 4 and pretest 5 ($\neg$**R** means that the null hypothesis $\mathcal{H}_{i,0}$ cannot be rejected while **R** means that the hypothesis is rejected in favor of the alternative hypothesis).

In case 1, the best solutions for the training intervals, obtained with 2 different search algorithms, do not perform better than lottery trading for the out-of-sample period. This suggests to us than there is nothing to learn. In case 2, GP outperforms lottery trading but random search does not; it is possible that there is something to learn, but that the selected random rules do not have a sufficient predictive ability. In any case, this leads us to a less certain conclusion than in case 3 where both search techniques outperform lottery trading. Finally case 4 is a special case where random search performs better than lottery trading but GP does not. The whole evolutionary process of GP has thus a detrimental effect and a possible explanation is that GP-induced solutions strongly overfit the training data despite the validation stage.

## 3.2 Question 2: Is the GP machinery working properly?

The second question we ought to ask is whether GP is effective. Of course, this cannot be answered with the data at hand if pretests 4 and 5 have shown that there is nothing to be learned (case 1 in Table 1). In addition, in case 4 of Table 1, we already know that GP is not efficient since, by transitivity, it is outperformed by the random search-based algorithm. Thus, the only two cases where one really needs to proceed to further examination are case 2 and case 3. The validity of the null hypothesis $\mathcal{H}_{1,0}$, which can be tested with pretest 1, gives a helpful insight into the answer: only if $\mathcal{H}_{1,0}$ should be rejected can we conclude that GP shows some real effectiveness. We would like to stress that rejecting $\mathcal{H}_{1,0}$ is far from implying profitability, but beating a mere random search algorithm on a difficult problem with an infinite search space is the bare minimum one can expect from GP.

## 4 Experiments

The aim of the experiments is to evaluate the extent to which the pretests proposed are reliable. The methodology adopted here is to check if the outcomes of the pretests are consistent with results already published in the literature. We call the software used in [4] GP1, which will constitute our benchmark, while GP2 is the GP implementation developed for this study. Although both programs have been developed by members of the AI-ECON Research Center, they have not been written by the same persons and do not share a single line of code. Furthermore, GP2, which is based on the *Open-Beagle* C++ library (see [7] and `http://beagle.gel.ulaval.ca/`), makes use of strongly-typed GP on the contrary to GP1. The GP2 control parameters, as close as possible to the ones used in [4] for GP1, are summarized in Table 1 (Appendix A).

The traded instruments are the indexes of 3 stock exchanges: the TSE 300 (Canada), the Nikkei Dow Jones (Japan) and the Capitalization Weighted Stock Index (Taiwan). They have been chosen among the 8 markets studied in [4] because they exhibit the main price evolution patterns that can be found in the set of 8 markets. The aim of GP is to induce the most profitable strategy, measured by the accumulated return, for trading the stock exchange index. The use of short selling is possible. We adopt what is done classically in the literature in terms of data-preprocessing and use normalized data that is obtained by dividing each day's price by a 250-day moving average.[16] In a way similar to what is usually done, we subdivide the whole data set into three sections: the *training*, *validation* and *out-of-sample* test periods. For each stock index considered, 3 different out-of-sample test periods of 2 years each (*i.e.,* 1999-2000, 2001-2002, 2003-2004) follow a 3-year validation and a 3-year training period. In the following, the term market refers to a stock exchange during a specific out-of-sample period. For instance, market Canada-1 (C1 for short) is the TSE 300 during the out-of-sample period 1999-2000. Hypothesis testing is performed with the *Student's t-test* at a 95% confidence level. The samples for statistics are made up of the results of 50 GP runs, 50 runs of equivalent search intensity random search with training and validation (ERS) and 100 runs of lottery trading (LT) with parameter $\alpha = 0.2$ (see §2.2 for the definition of $\alpha$). The following results were obtained with GP2:

- In 4 out of the 9 markets (*i.e.,* C3, J2, T1, T3), there is evidence that there is something to learn from the training data (case 3 in Table 1 - GP2 and ERS outperform Lottery Trading). This is consistent with [4] where GP1 performs outstandingly in these 4 markets (respective total return: 0.34, 0.17, 0.52, 0.27).
- In markets C1, J3 and T2, pretests 4 and 5 suggest to us that there is nothing to learn (case 1 in Table 1 - neither GP2 nor ERS outperform Lottery Trading). Except for C1, GP1 also performs poorly ($-0.18$ for J3 and $-0.05$ for T2).[17]

[16] See [4] for a discussion on how non-normalized data affects the performance of GP.

[17] The two markets that are not listed, i.e. C2 and J1, correspond to cases where "there may be something to learn." Precisely, they both belong to case 2 in Table 1, that GP beats LT but random search does not beat LT.

- Finally, in the 3 markets where GP2 is shown to beat ERS ($\mathcal{H}_{1,0}$ is rejected in favor of $\mathcal{H}_{1,1}$ for J1, J2 and T1), the GP results are very good: both GP1 and GP2 produce positive returns and outperform the buy-and-hold strategy.

Although more comprehensive tests are needed, the experiments conducted here on 9 markets show some preliminary evidence that the proposed pretests possess some predictive ability. Indeed, when the outcome is "nothing to learn," the two GP implementations perform very poorly (except in one case). On the contrary, when the pretests suggest that there is something to learn, at least one GP implementation does well.[18]

When pretests suggest to us that a market is efficient, we cannot conclude that there is no way of making consistent profits in this market, because the concept of efficiency is of course relative to the investors considered. What can be concluded is that a group of investors making their investment decisions by running GP2 on the past price time-series will not be able to consistently outperform the market. It is also worth noting that the efficiency of a market is variable over time; for instance, pretests suggest to us that T2 is efficient while T1 and T3 are not. As highlighted in [4], GP not being efficient is often due to the training interval exhibiting a time series pattern which is significantly different from the out-of-sample period (e.g., "bull" versus "bear", "sideways" versus "bull", . . . ). Thus, a first way of making improvements that can be investigated is to rethink the data division scheme.

In light of the pretests, we should also conclude that our GP implementation (*i.e.* GP2) is more efficient than random search (GP2 outperforms ERS in 3 markets while ERS never beats GP2 with statistical significance). However, in our experiments, searching trading rules at random, with the same set of functions and terminals as used in GP, is usually enough to come up with trading systems that outperform lottery trading when GP does as well. This suggests to us that GP2 may only be able to take advantage of "simple" regularities in the data.

## 5 Conclusions

The main purpose of this paper is to enrich the earlier research on Genetic Programming (GP) induced market-timing decisions by proposing pretests aiming to shed light on the GP results. In actual fact, in the literature, the results of applying GP for market-timing decisions are typically not very convincing, but the investigators always suggest the possibility of further improvements. If the investigators can first be convinced that there is something to learn and that GP is suitable for that task, then their conclusion would be less vague and uncertain. We propose here a series of

[18] In all 4 such cases (C3, J2, T1, T3), GP2 beats LT, but in 2 cases where the market is bullish (C3 and T3) the returns earned by GP2, which are 5% and -4%, respectively, are far less than those of Buy-and-Hold, which are more than 30% during the out-of-sample period. As a result, one cannot say that GP2 is performing superbly. However, in those 2 cases, GP1, which seems in general to be the best implementation, happens to be very efficient (only a few percent less than buy and hold).

pretests, where GP is tested against a random behavior (*lottery trading*) and against strategies created at random (*zero-intelligence strategies*) that aim to answer these two crucial questions. Of course there is the risk of getting a wrong pretest result and the possible reasons why GP may have failed should be thoroughly investigated before drawing a conclusion. But, in the end, analyzing the results in light of the pretests should help us to draw more fine-grained conclusions.

## Acknowledgements

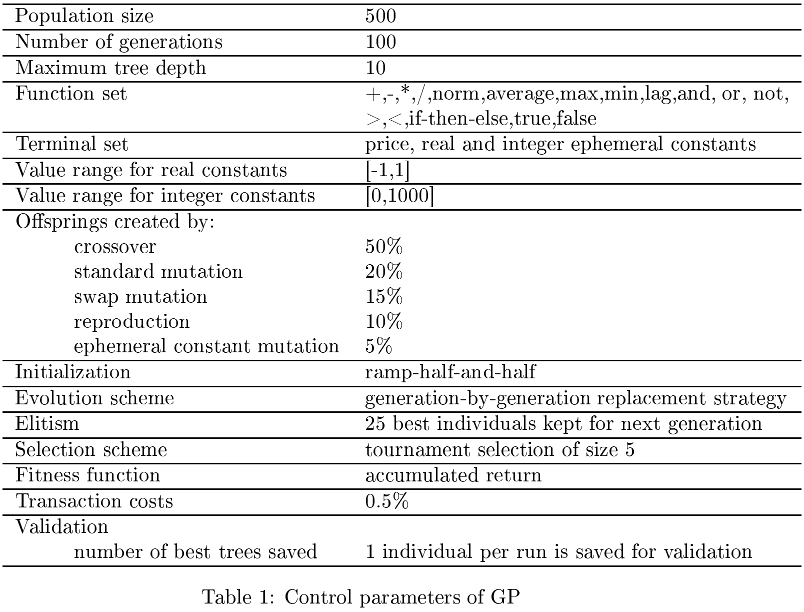
This research was conducted when the second author (Nicolas Navet) was a visiting researcher at the AI-ECON Research Center, National Chengchi University (NCCU), Taipei, Taiwan. The financial support from the AI-ECON Research Center, INRIA, as well as NCCU is greatly acknowledged. The authors would also like to acknowledge the grant from the National Science Concil #95-2415-H-004-002-MY3.

## A Genetic Programming Settings

Program GP2 implements strongly typed GP with the set of functions and terminals described in Table 1. The parameters here are basically identical to the ones in [4] (program GP1) except when fine-tuning GP2 has highlighted that better results may be obtained with different parameters. Precisely when we make use of more elitism, the size of the tournament selection is set to 5 and numerical mutation is implemented.

## References

1. Allen F, Karjalainen R (1999) Using genetic algorithms to find technical trading rules. Journal of Financial Economics 51:245–271
2. Boffetta G, Cencini M, Falcioni M, Vulpiani A (2002) Predictability: A way to characterize complexity. Physics Reports 356:367
3. Chen SH, Kuo TW (2003) Overfitting or poor learning: a critique of current financial applications of GP. In: Ryan C, Soule T, Keijzer M, Tsang E, Poli R, Costa E (eds) Proceedings of the sixth European conference on genetic programming. Springer-Verlag:34–46
4. Chen SH, Kuo TW, Hoi KM (2007) Genetic programming and financial trading: how much about "what we know"? In: Zopounidis C, Doumpos M, Pardalos PM (eds) Handbook of financial engineering. Springer. Forthcoming.
5. Danilov D, Magnus J (2004) Forecast accuracy after pretesting with an application to the stock market. Journal of Forecasting 23:251–274
6. del Arco-Calderón CL, Viñuela PI, Castro JCH (2004) Forecasting time series by means of evolutionary algorithms. In: PPSN:1061–1070
7. Gagné C, Parizeau M (2002) Open beagle: a new versatile c++ framework for evolutionary computations. In: Late breaking papers, genetic and evolutionary computing conference (GECCO):161–168



| | |
|---|---|
| Population size | 500 |
| Number of generations | 100 |
| Maximum tree depth | 10 |
| Function set | +,-,*,/,norm,average,max,min,lag,and, or, not, >,<,if-then-else,true,false |
| Terminal set | price, real and integer ephemeral constants |
| Value range for real constants | [-1,1] |
| Value range for integer constants | [0,1000] |
| Offsprings created by: | |
| crossover | 50% |
| standard mutation | 20% |
| swap mutation | 15% |
| reproduction | 10% |
| ephemeral constant mutation | 5% |
| Initialization | ramp-half-and-half |
| Evolution scheme | generation-by-generation replacement strategy |
| Elitism | 25 best individuals kept for next generation |
| Selection scheme | tournament selection of size 5 |
| Fitness function | accumulated return |
| Transaction costs | 0.5% |
| Validation | |
| number of best trees saved | 1 individual per run is saved for validation |

Table 1: Control parameters of GP

**Fig. 1.** GP control parameters

8. Gagné C, Schoenauer M, Parizeau M, Tomassini M (2006) Genetic programming, validation sets, and parsimony pressure. In: Collet P, Tomassini M, Ebner M, Gustafson S, Ekárt A (eds) Proceedings of the 9th European conference on genetic programming. Springer Verlag:109–120
9. Giles J, Giles D (1993) Pre-test estimation and testing in econometrics: recent developments. Journal of Economic Surveys 7(2):145–197
10. Hong J, Chung Y (2003) Are the directions of stock price changes predictable? Statistical theory and evidence. Technical report, Cornell University
11. Kaboudan MA (1999) A measure of time series' predictability using genetic programming applied to stock returns. Journal of Forecasting 18:345–357
12. Kaboudan MA (2000) Evaluation of forecasts produced by genetically evolved models. In: 6th international conference on computing in economics and finance. Society for Computational Economics
13. Langdon WB, Poli R (2002) Foundations of genetic programming. Springer-Verlag.
14. Li J, Tsang E (2000) Reducing failures in investment recommendations using genetic programming. In: 6th international conference on computing in economics and finance. Society for Computational Economics
15. Neely C, Weller P, Dittmar R (1997) Is technical analysis in the foreign exchange market profitable? A genetic programming approach. Journal of Financial and Quantitative Analysis 32(4):405–427
16. Palus M, Pecen L, Pivka D (1995) Estimating predictability: redundancy and surrogate data method. Working Paper 95-07-060, Santa Fe Institute. available at http://ideas.repec.org/p/wop/safiwp/95-07-060.html
17. Santini M, Tettamanzi A (2001) Genetic programming for financial time series prediction. In: Miller J, Tomassini M, Lanzi PL, Ryan C, Tettamanzi AGB, Langdon WB

(eds) Proceedings of the fourth European conference on genetic programming. Springer Verlag:361–370

18. Schreiber T, Schmitz A (2000) Surrogate time series. Phys. D 142(3-4):346–382
19. Shalizi CR (2006) Methods and techniques of complex systems science: an overview. In: Deisboeck T, Yasha K (eds) Complex systems science in biomedicine. Springer Verlag, New York:33–114
20. Sullivan R, Timmermann A, White H (1999) Data-snooping, technical trading rule performance, and the bootstrap. Journal of Finance 54:1647–1692
21. Zumbach G, Pictet O, Masutti O (2001) Genetic programming with syntactic restrictions applied to financial volatility forecasting. Technical Report GOZ.2000-07-28, Olsen & Associates.