ELSEVIER

Journal of Economic Dynamics and Control
21 (1997) 1043-1063

JOURNAL OF
Economic
Dynamics
& Control

Toward a computable approach to the efficient market hypothesis: An application of genetic programming

Shu-Heng Chen [a,*], Chia-Hsuan Yeh [b]

[a] *Department of Economics, National Chengchi University, Taipei, Taiwan 11623, ROC*
[b] *Department of Economics, National Chengchi University, Taipei, Taiwan 11623, ROC*

Abstract

From a computation-theoretic standpoint, this paper formalizes the notion of unpredictability in the efficient market hypothesis (EMH) by a biological-based search program, i.e., genetic programming (GP). This formalization differs from the traditional notion based on probabilistic independence in its treatment of *search*. Compared with the traditional notion, a GP-based search provides an explicit and efficient search program upon which an objective measure for predictability can be formalized in terms of search intensity and chance of success in the search. This will be illustrated by an example of applying GP to predict chaotic time series. Then the EMH based on this notion will be exemplified by an application to the Taiwan and US stock market. A short-term sample of TAIEX and S&P 500 with the highest complexity defined by Rissanen's minimum description length principle (MDLP) is chosen and tested. It is found that, while linear models cannot predict better than the random walk, a GP-based search can beat random walk by 50%. It, therefore, confirms the belief that while the short-term nonlinear regularities might still exist, the search costs of discovering them might be too high to make the exploitation of these regularities profitable, hence the efficient market hypothesis is sustained.

Keywords: Genetic programming; Evolutionary computation; Minimum description length principle; Mean absolute percentage error; Efficient market hypothesis
JEL classification: C63; G14

* Corresponding author.

The authors are grateful for the research support from NSC grant No. 85-2415-H-004-001. They have benefitted from comments at the Thirty-First EFA (Eastern Finance Association) Annual Conference and the First International Conference of the Society for Computational Economics. They would also like to thank the anonymous referee for very helpful suggestions.

1. Introduction

Despite its long history,[1] the efficient market hypothesis (EMH) has remained at the heart of much of the contemporary debate in financial economics. Throughout its entire history, the EMH was mainly formalized and modified based on the concept of *probabilistic independence*. Technically speaking, it shows that the σ-algebra generated by the history of the rates of return will tell us nothing about the present or future rates of return. In other words, the rate of return R_t should be independent of any Borel functions of R_s $(s < t)$.[2] Up to the present, this seems to have been the only way to mathematize the intuitive meaning behind the EMH, i.e., *unpredictability*, but that is not to say that it is scientifically sound. The major problem of this formalization is that there is no effective algorithm such that we can construct the evidence of independence by trying *all* Borel functions of R_s $(s < t)$. Therefore, *the EMH based on this notion of unpredictability is practically uncomputable*, and in practice, the test gradually proceeds from *linear independence* to *nonlinear independence* such as the BDS test.[3] While many recent studies based on nonlinear tests[4] suggest that there might exist nonlinear dependence in the financial data, these tests alone tell us nothing about the predictability of the the rates of return;[5] nor do they indicate whether we can profit from a better prediction. For example, the rejection of nonlinear tests might suggest that there exist nonlinear regularities to be exploited; however, the search costs to find such nonlinear regularities may be so high that the net profits of using this nonlinear regularity might be negative. In this case, rejecting the non-existence of nonlinear dependence does not mean the rejection of the EMH. Hence, what we can truly learn from these nonliearity tests is not clear. In other words, there is a gap between the result of nonlinearity tests and its implication for the efficient market hypothesis.

This paper contributes to bridging the gap between nonlinearity tests and the EMH by providing an approach where both the issues of *predictability* and *profitability* are taken into account. Thus, if the EMH is rejected in our approach, it means what it should mean. We shall call this approach a *computable* approach. We start this construction by *reconsidering the meaning of unpredictability from a computation-theoretic perspective*. In computation theory, the theorem that the halting problem of the *universal Turing machine* is uncomputable implies that

[1] It goes back to Bachelier (1900).

[2] In the literature, this is called *weak-form efficiency*.

[3] See Brock, Dechert and Scheinkman (1987).

[4] See Savit (1988, 1989), Hinich and Patterson (1989), Hsieh (1989), Frank, Gencay and Stengos (1988), Scheinkman and Lebaron (1989), Peters (1991), and Willey (1992).

[5] For example, the tests alone cannot tell us whether low-dimensional chaos is easier to predict than high-dimensional chaos or whether deterministic chaos is easier to predict than stochastic nonlinearity.

the decision about whether any sequence is predictable is, in general, not computable (or decidable). Consequently, this paper does not consider formalizing unpredictability in its absolute sense but rather in its weaker sense as exemplified by the expressions of *hard to predict* and *very hard to predict*.

The intuitive meaning of *hard to predict* or *very hard to predict* can be considered equivalent to (*very*) *hard to find a rule from past experience under intensive search which can predict the future better than a random walk* (*RW*). If this is the case, to capture the technical meaning of unpredictability and the EMH, we only need *an explicit search program* in which *the intensity of the search* denoted by a vector x, $x \in R^m$, and *the chance of success in the search* denoted by π can be formalized. Then π as a function of x, i.e., $\pi(x)$ can be considered an objective measure of unpredictability, i.e., an indicator of showing how hard it is to predict. Given the same level of x, the lower the π, the harder it is to predict; or given π, the higher the level required, the harder it is to predict.

The explicit search program considered in this paper is the *genetic programming paradigm* developed by Koza (1992). A simple introduction to genetic programming is given in the appendix. Genetic programming (GP) extends the genetic algorithm to the domain of *computer programs* and is probably the most general style in *evolutionary computation*, a biologically based approach to computation. Computation can be regarded as a process of search for solutions. The search may become extremely difficult when the search space is too large and an exhaustive search is practically infeasible. Genetic programming, with its application of Darwin's '*survival of the fittest*' principle, has been shown as a very efficient paradigm to implement search in such cases.[6]

The search intensity of GP can be revealed by the set of chosen parameters in running genetic programming. For example, by increasing the '*population size*' from 500 to 1000 and/or the '*number of generations*' from 1000 to 2000 in Table 1, we are increasing the search intensity. As to the chance of success, it crucially depends on what we mean by success. We suggest that, whatever the definition of success, the *criterion of fitness* used in GP should be consistent with that definition. For example, if we consider that mean absolute percentage error (MAPE) should be used to measure the accuracy of a forecast, then MAPE also should be chosen as the *criterion of fitness* to run GP. Once the fitness of criterion is given, various definitions of success can be developed. For example, success means MAPE $= 0$, or MAPE < 1, and so on. In Section 2, this formalization of unpredictability is illustrated by examples of predicting chaotic

[6] Roughly speaking, many examples show that a GP-based search performs more efficiently than a blind random search. For a delicate discussion of this comparison, we refer to Chapter 9 of Koza (1992).

Table 1
Tableau for predicting chaotic dynamic systems

Population size	500
The number of trees created by complete growth	50
The number of trees created by partial growth	50
Functional set	$\{+, -, \times, \%\}$
Terminal set	$\{X_t, R\}$
The number of trees generated by reproduction	50
The number of new lives	50
The number of trees generated by mutation	100
The probability of mutation	0.2
The maximum length of the tree	17
The probability of leaf selection under crossover	0.5
The number of generations	1000
The maximum number in the domain of Exp	1700
Criterion of fitness	SSE

dynamic systems. Sections 3 and 4 are devoted to the demonstration of how this notion can be used to test the EMH.

2. An illustration: Predicting chaotic dynamic systems by genetic programming

> ... whether a chaotic process or a complex nonlinear process generates a pattern is irrelevant unless one knows the exact functional form generating the process. Even if tests indicate a high embedding dimension, forecasting is impossible unless a specific functional form is assumed (Fogler, 1995, p.16).

In this section we will illustrate our formalization of unpredictability by using genetic programming to predict chaotic dynamic systems.[7] This section also attempts to show that if the rates of return are generated by a *simple* deterministic dynamic system, then GP might actually discover it. If we represent each chaotic dynamic system by the LISP S-expression depicted as a rooted, point-labeled tree (GP-tree), the term 'simple' refers to the *depth* of the GP-tree and has nothing to do with the embedding dimension. This section will also show that knowing the existence of chaos or nonlinearity might not be as irrelevant as claimed by Folger. In fact, GP can be considered a second step of the data analysis within a complete framework. In the first step, we test whether nonlinearity exists. If it does, we can then initiate GP to see whether we can find it. To illustrate these points, the following three choatic dynamic systems with the same embedding

[7] The details of this section can be found in Chen and Yeh (1995).

Table 2
Number of generations required to make SSE = 0; n^*

Chaotic series	Depth of the GP-tree	n^*
1	4	7, 12, 14, 19
2	5	29, 37, 37, 70
3	6	151, NA, NA, NA

dimension are chosen from Devaney (1989) and the depth of their GP-tree is 4, 5, and 6, respectively.

$$x_{t+1} = 4x_t(1 - x_t), \quad x_t \in [0,1] \ \forall t, \tag{1}$$

$$x_{t+1} = 4x_t^3 - 3x_t, \quad x_t \in [-1,1] \ \forall t, \tag{2}$$

$$x_{t+1} = 8x_t^4 - 8x_t^2 + 1, \quad x_t \in [-1,1] \ \forall t. \tag{3}$$

By setting the initial value $x_0 = 0.213$, a time series composed of fifty observations is generated for Eqs. (1)–(3) respectively, i.e., Time Series 1, 2, and 3, and is shown in Fig. 1. The chosen parameters to run the GP-based search are given in Table 1. Here, sum of squared errors (SSE) is chosen as the criterion of fitness; π is the chance of SSE being 0. The search intensity is measured by the parameter *number of generations*, denoted by n. So, the degree of complexity in predicting will be measured by $\pi(n)$.

Four simulations were implemented for Eqs. (1)–(3). The number of generations required to learn each equation is shown in Table 2. While GP succeeded in all simulations in predicting the future from the past by discovering the underlying model for Time Series 1 and 2, the latter seems to be harder to predict than the former. This is because the number of generations required to learn Time Series 1 *to a* 100% *precision*, i.e., SSE = 0, is 7, 12, 14 and 19, while it is 29, 37, 37, and 70 for Time Series 2. Even though the sample $\pi(100)$ is 1 in both cases, the sample $\pi(n)$ is different for small n. When n is small, $\pi(n)$ for Time Series 2 is expected to be smaller than that for Time Series 1. So, if we set the chance of success to be equal, the search intensity required for Time Series 2 will be higher than that for Time Series 1. On the other hand, if we set the search intensity to be equal, the chance of success for Time Series 2 will be lower than that for Time Series 1. It is in this sense that we say that Time Series 2 is more difficult to predict than Time Series 1. As to Time Series 3, three out of four simulations failed to rediscover the underlying system. The only one that succeeded took 151 generations to rediscover Eq. (3). There is little doubt that Time Series 3 is extremely difficult to predict as opposed to Time Series 1 and 2. *These simple examples illustrate how genetic programming can provide us with an explicit search program upon which an objective measure for predictability can be constructed.*

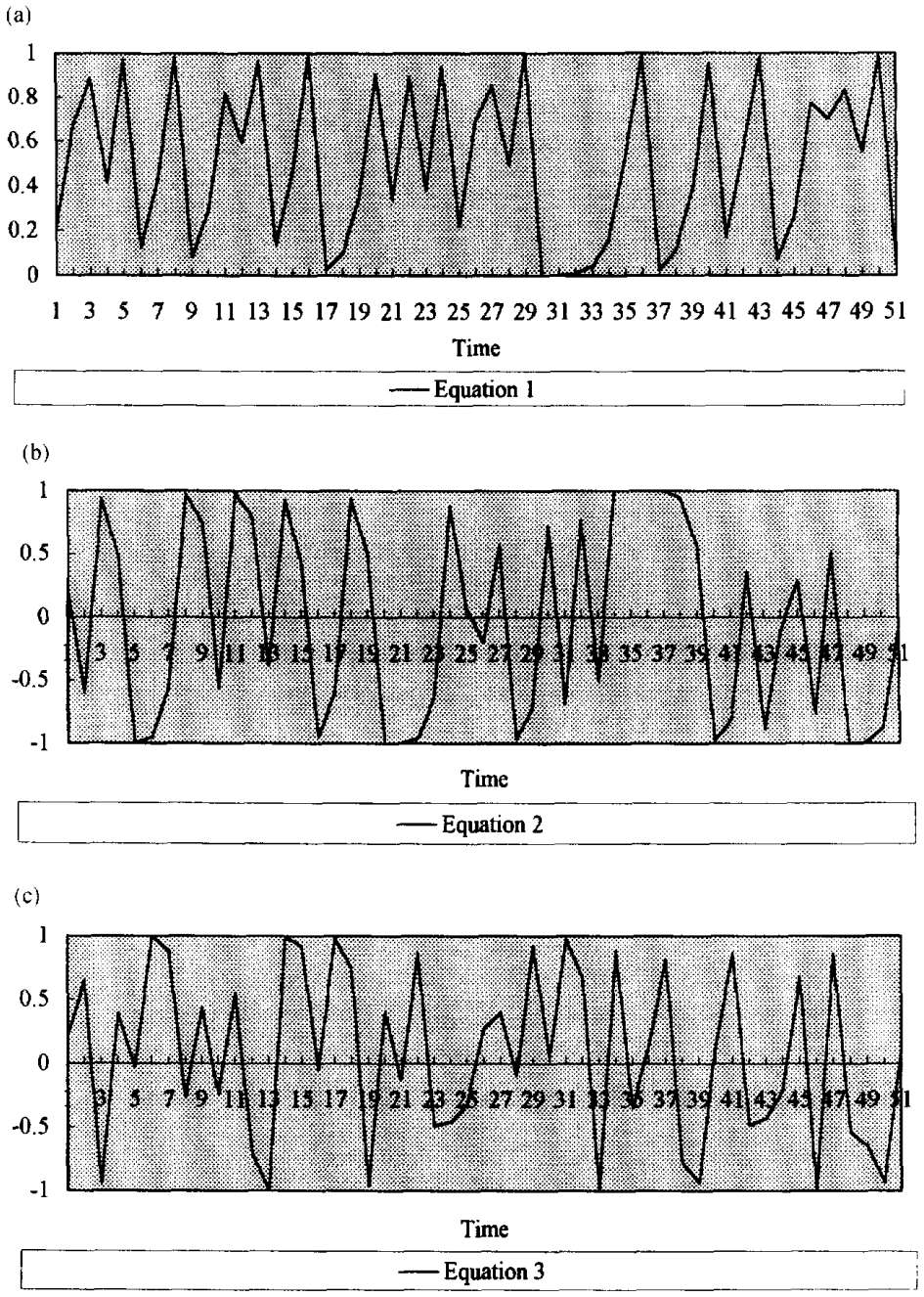


Fig. 1. Time series of chaotic dynamic systems: (a)1; (b)2; and (c)3.

3. Choosing data set by the MDL principle

We will now give two examples to demonstrate how this notion of unpredictability can be used to test the EMH. These examples are based on the data concerning the daily rate of return of the Taiwan Stock Price Index (TAIEX) and S&P 500 Index, which are available from the EPS database. From 1/5/71 to 1/27/94, there are 6677 observations in the Taiwan dataset and 5831 in the S&P 500 dataset. Are we going to use all of them? While the efficient market hypothesis puts no restriction on the sample size, the application of GP to different sizes of sample does require thought. This is because GP aims at finding the existence of potential nonlinear regularities. The requirement for the sample size varies with different periodicities. For the time-invariant long-term nonlinear relation, a large sample size is needed. However, the recent studies of the time series of stock prices, such as LeBaron (1992), seem to indicate that, even though nonlinear regularities might exist, they are not stable over time. Therefore, suppose that the stock market encounters a sequence of short-term time-variant nonlinear relations, a large sample size may average out all these relations. In this case, a smaller sample size is desirable, and the choice of a small sample size in this paper is justified by this consideration.[8]

How small should it be then? We do not have a definite answer to this question.[9] Nevertheless, since this example only attempts to demonstrate how to apply the GP-based search to the test of the EMH, the issue might not be that important and the length of the training series is arbitrarily set to be 50. However, out of these 6677 and 5831 observations, there are 6628 and 5782 series with the length of 50. Testing all of them is too time-consuming and also unnecessary by the following *rules of thumb*. Firstly, the series without any turning points, in general, might be less *complex* than those with many turning points. Secondly, the series which has regular patterns of turning points might also be less complex than those with irregular patterns. In fact, the possibility that not all time periods are equally hard to predict has been noticed by the recent studies of stock prices (e.g., LeBaron, 1992). For small sample size such as 50, this inequality might be prevalent. However, to take advantage of this inequality, the rules of thumb to judge the degree of complexity must be associated with an objective measure, and this paper will use *Rissanen's stochastic complexity* as the measure.

Rissanen's minimum description length (MDL) is an approximation for *Kolmogorov complexity* which measures the complexity of a set of data by the

[8] Similar example of the preference for the short data series can also be found in Oakley (1994).

[9] The answer Oakley (1994) is based on a rule established by Ruelle (1990), i.e., if the observations are generated by a chaotic dynamic system, then the expected minimum series from which useful information could be extracted can be determined by the *correlation dimension*. In his case, since the target function to be rediscovered is already known and is Mackey–Glass equation; therefore, Ruelle's crietrion can be applied in his case but not ours.

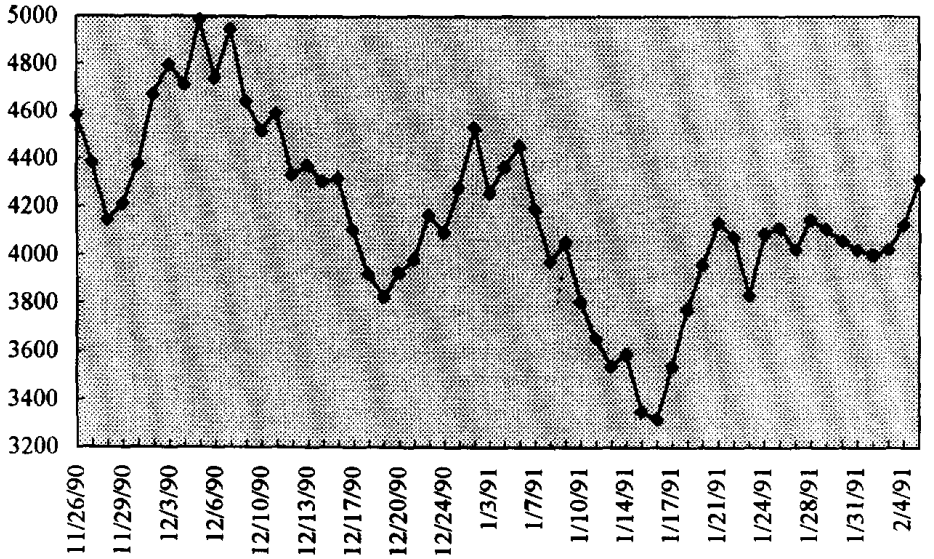


Fig. 2. The series of stock index with Max MDL(TAIEX).

length of the shortest universal Turing machine program that will generate the data. The measure is well-defined, but not practically computable. The MDL developed by Rissanen (1982) is a way to approximate this uncomputable measure by replacing the universal Turing machine with a class of probabilistic models. *This paper applies MDL to pick out the most complex 50-series observations, i.e., the sample period with the highest MDL, as our data set.*

A detailed description of this procedure and its meaning can be found in Chen and Tan (1995). Briefly speaking, we first transform the orginal sequence of $\{R_t\}$ from 1/5/71 to 1/27/94 into a 0-and-1 sequence based on the sign of R_t. Then MDL is computed for each of the 50 consecutive observations in the 0-and-1 sequence by choosing the *Bernoulli class* and *Markov class* as our model classes. By this criterion, we used the TAIEX from 11/27/90 to 1/30/91 (shown in Fig. 2)[10] and used the S&P 500 index from 5/6/77 to 7/19/77 (shown in Fig. 4).[11]

Once the length of in-sample data series is given, the next issue is how to determine the length of post-sample series.[12] Since we only consider the

[10] The MDL for this period is 37.807. The lowest MDL, which is 12.126, is observed in the period from 5/23/74 to 6/1/74 and the 50-day price indexes starting backward from 6/1/74 are shown in Fig. 3.

[11] The MDL for this period is 37.807. The lowest MDL, which is 30.95, is observed on 10/3/74, and the 50-day price indexes starting backward from 10/3/74 are shown in Fig. 5.

[12] For the prediction of chaotic time series, this question might be unimportant. Because, in the case when GP can successfully rediscover the underlying model which generates the chaotic series, the accuracy of prediction will not depend on how far we would like to predict.

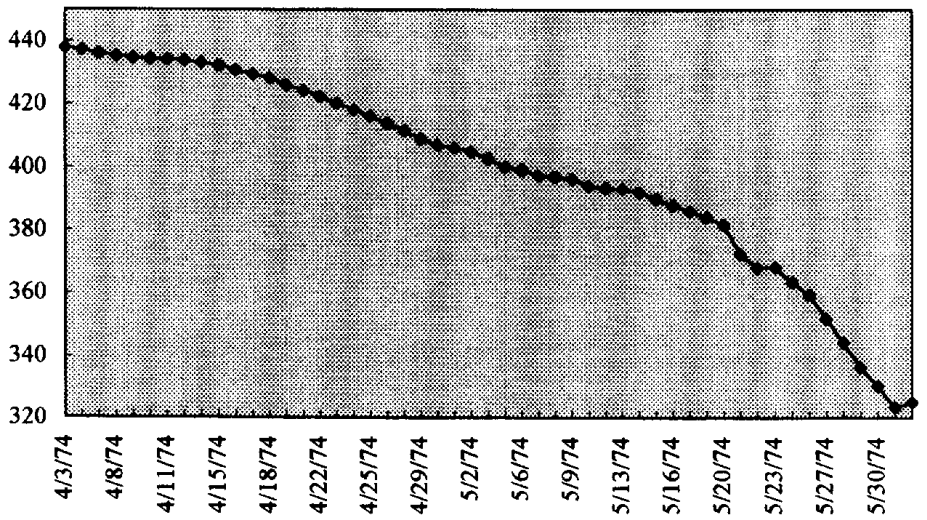


Fig. 3. The series of stock index with Min MDL (TAIEX).

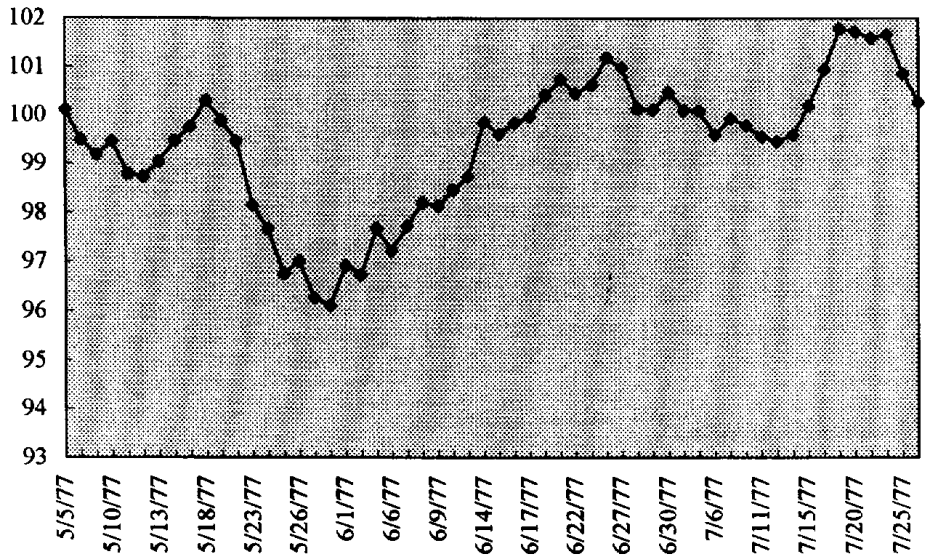


Fig. 4. The series of stock index with Max MDL (S&P500).

capability of GP to discover the possible existence of short-term nonlinearities, it is natural not to test its performance by using too many post-sample observations. We therefore arbitrarily set the in-sample data to be ten times the size of the post-sample data. So, for the TAIEX, the post-sample period is from 1/31/91

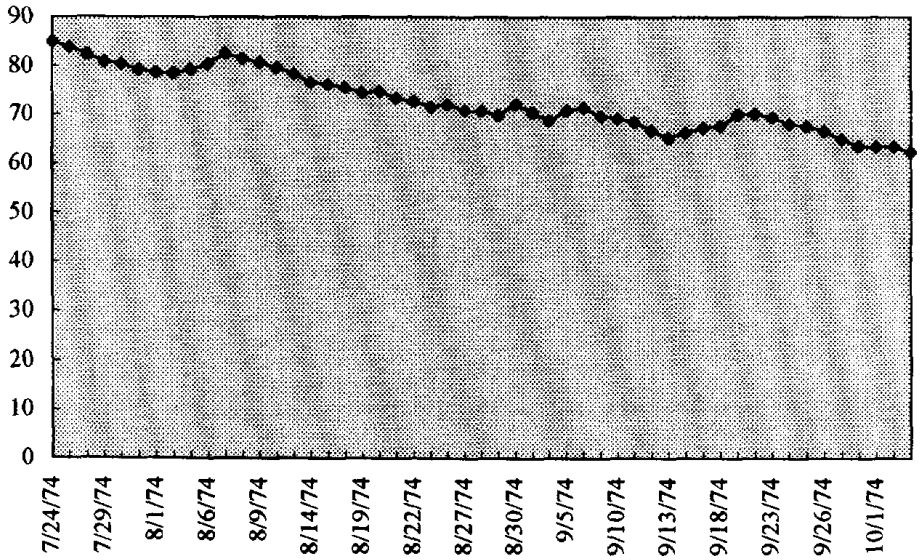


Fig. 5. The series of stock index with Min MDL (S&P500).

Table 3
Tableau of the GP-based search

Functional set	$\{+, -, \times, \%, \sin, \cos, \text{EXP}, R\,\text{LOG}\}$
Terminal set	$\{R_{t-1}, R_{t-2}, \ldots, R_{t-10}\}$
The number of generations	200
Criterion of fitness	MAPE

to 2/5/91 (shown in Fig. 2.). For the S&P 500 index, the post-sample period is from 7/20/77 to 7/26/77 (shown in Fig. 4).

4. The empirical results

To implement genetic programming, the program GP-Pascal is written in Pascal 4.0 by following the instruction given in Koza (1992). A detailed description of this program can be found in Chen et al. (1995). The chosen parameters to run GP-Pascal are the same as those in Table 1 except for those changes indicated in Table 3. One of the changes is the fitness function. The fact that SSE is replaced by MAPE is attributed to Makridakis (1993), who suggested a modified form of MAPE as the most appropriate measure satisfying both theoretical and practical concerns while allowing meaningful relative comparisons. Based on these parameters, 72 simulations were executed for TAIEX and S&P 500.

For each of the simulation, the MAPE is calculated for the in-sample period and the post-sample period. The in-sample MAPEs of the best model chosen by

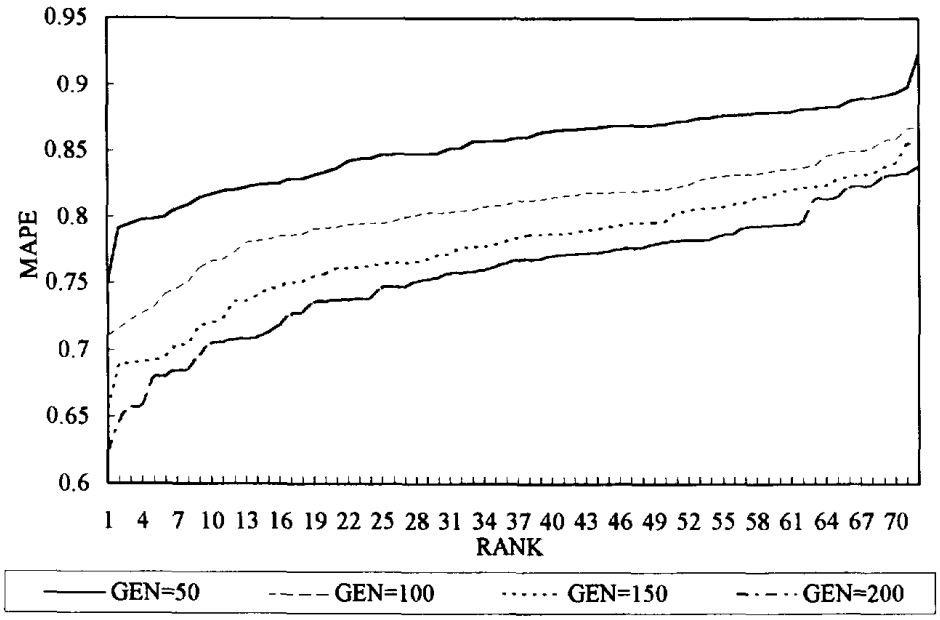


Fig. 6. In-sample MAPEs (TAIEX).

each simulation in Generation (Gen) 50, 100, 150 and 200 are ranked from the lowest to the highest and are shown in Figs. 6 and 7. Since the MAPE of the RW is 1, GP is said to beat the RW if its MAPE is less than 1. Let $\pi_1(n)$ be the probability that GP can beat the RW in Generation n. Figs. 6 and 7 show that the sample $\pi_1(n)$ are 100% in Generations 50, 100, 150 and 200 for both TAIEX and S&P 500, and when evolution takes longer, improvement can always be made. For example, for TAIEX, the MAPE in the best and worst case of these simulations in Generation 50 is 0.749 and 0.924, respectively, and in Generation 200, it is improved to 0.622 and 0.838. For S&P 500, the MAPE in the best and worst case of these simulations in Generation 50 is 0.792 and 0.934, respectively, and in Generation 200, it is improved to 0.691 and 0.898. To make a comparsion, the in-sample MAPES of linear autoregressive models of order p ($p = 1, 2, ...10$) are shown in Table 4. For both TAIEX and S&P 500, no linear AR model can beat RW.

After observing these improvements, one cannot help wondering whether the MAPE will go down to 0 by evolving longer. Can the GP-based search discover the true underlying model just like the previous illustration in Section 2? The answers seem to be negative because by examining Figs. 6 and 7 more carefully, we notice that the rate of improvement is decreasing, and hence, further improvement might be very limited.

If the MAPE(n) is unlikely to reach zero when n is large and the underlying regularities cannot be discovered, then the *overfitting problem* commonly seen in

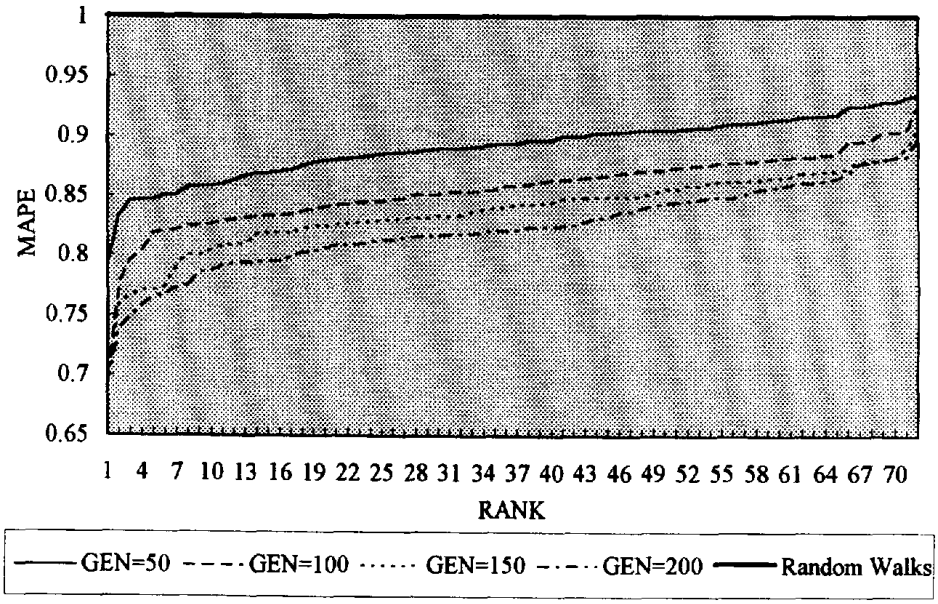


Fig. 7. In-sample MAPEs (S&P500).

Table 4
The MAPEs of in-sample and post-sample in TAIEX and S&P 500: Linear autoregressive models

Model	TAIEX		S&P 500	
	In-sample	Post-sample	In-sample	Post-sample
AR(1)	1.039	0.904	1.076	1.567
AR(2)	1.035	0.906	1.435	2.671
AR(3)	1.124	1.174	1.437	2.671
AR(4)	1.108	1.274	1.410	2.655
AR(5)	1.230	1.339	1.386	2.559
AR(6)	1.210	1.145	1.901	3.137
AR(7)	1.194	1.096	1.899	3.194
AR(8)	1.064	2.148	1.980	3.664
AR(9)	1.088	1.954	1.795	3.562
AR(10)	1.116	1.866	1.827	4.160

inductive inference could happen. In that case, the regularities detected by the GP-based search cannot be taken too seriously, and hence, no victory can be claimed by GP over random walks. To overcome this problem, the technique of *cross validation* is used. The post-sample MAPEs of the best model chosen by each simulation in Gen (generation) 50, 100, 150 and 200 are also ranked from the lowest to the highest and are shown in Figs. 8 and 9. Let $\pi_2(n)$ be the probability that GP can beat RWs in Generation n. Table 5 shows that the chance of beating RWs is about 50% for both TAIEX and S&P 500 in Generations 50 and 100.

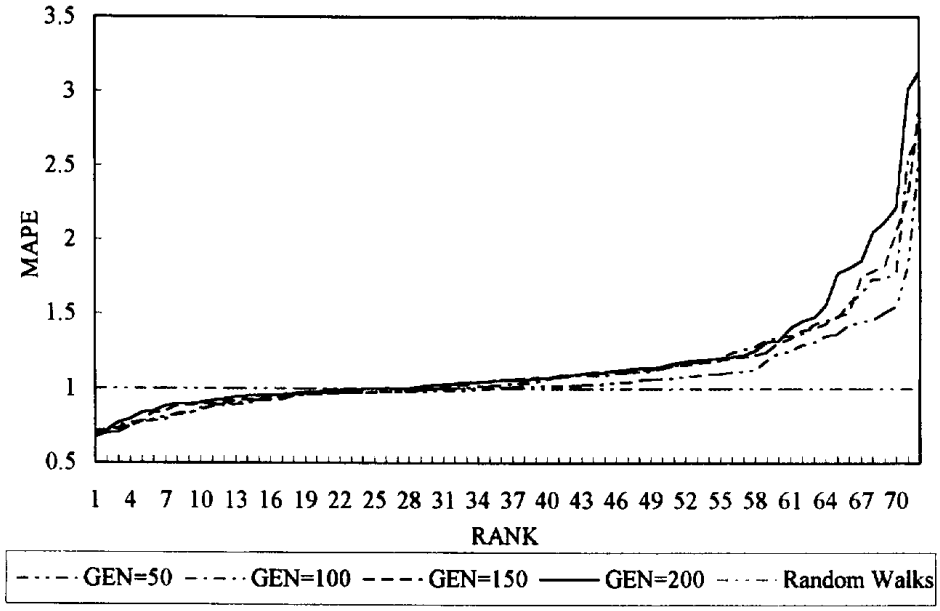


Fig. 8. Post-sample MAPEs (TAIEX).

Table 5
The chance of beating RWs by GP-based search: π_2

	TAIEX	S&P 500
$\pi_2(50)$	$\frac{35}{72}$	$\frac{41}{72}$
$\pi_2(100)$	$\frac{30}{72}$	$\frac{41}{72}$
$\pi_2(150)$	$\frac{27}{72}^a$	$\frac{34}{72}$
$\pi_2(200)$	$\frac{25}{72}^a$	$\frac{29}{72}$

[a] Statistically significantly different from 0.5 at the 5% level.

By this ratio, beating RWs is not as hard as is claimed in the current literature [13] and is much easier as opposed to the linear AR models. [14]. However, our results cannot be directly compared with the existing literature without some caution.

First of all, most of the existing tests for the EMH do not have an explicit search program such that the search intensity can be objectively measured. In our case, however, the maximum number of models each generation could possibly generate is 500, and in a simulation with 50 generations, this number can be as

[13] See Diebold and Nason (1990).

[14] See Table 5, there are only two linear models that can beat RWs in TAIEX and none can beat RWs in S&P 500.

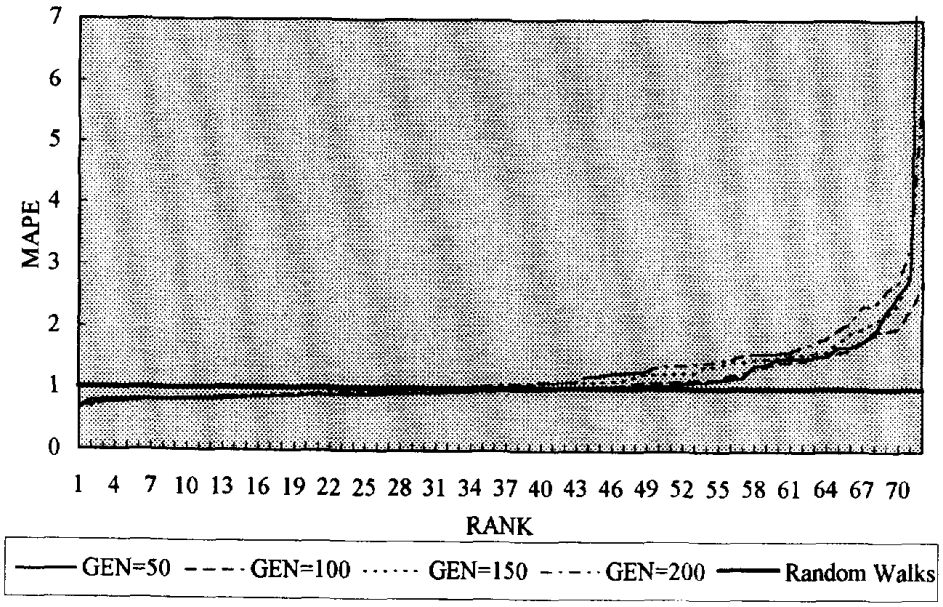


Fig. 9. Post-sample MAPEs (S&P500).

large as 25,000. Therefore, in each simulation, the best model in Generation 50 is the model that is the best among 25,000 candidates, and this number might be too overwhelming for most existing tests. Secondly, nonlinear regularities might exist but are not stable over time. Therefore, when different periods of data are given, the GP-based search will automatically be renewed. Forecasting in this recursive manner will certainly increase the intensity of search, but at the same time it might also increase the chance of beating the RW even in the long run. However, as LeBaron (1992) stated, '... measuring global forecast improvements may be misleading and not give a complete picture of how well a forecasting method is working (p. 392)'. Traditional forecasts comparison based on the global standard might easily underestimate the chance of beating RWs. *Nevertheless, whether the chance of beating the RW has been underestimated is not the concern of this paper. What we are trying to point out is that the chance of beating the RW cannot be objectively evaluated without explicitly taking search intensity into account.* Last but not least, since the search directed by genetic programming is *random*, what matters is the probability of finding models better than RWs. If that probability is only 50% or less, the GP-based search still cannot be considered more efficient than RWs.

What is the significance of beating RWs? Does that imply that regularities have been discovered? To answer this question, a simple correlation between the in-sample MAPE and post-sample MAPE in Generations 50, 100, 150 and 200 are run. The result is shown in Table 6. For most of the cases, the correlation is negative. This signifies more or less the overfitting problem mentioned before. In

Table 6
The correlation between the in-sample and post-sample MAPE

	TAIEX	S&P 500
ρ_{50}	−0.28[a]	−0.25[a]
ρ_{100}	−0.15[a]	0.00
ρ_{150}	−0.08	−0.07
ρ_{200}	−0.03	−0.26[a]

[a] Statistically significant at the 5% level.

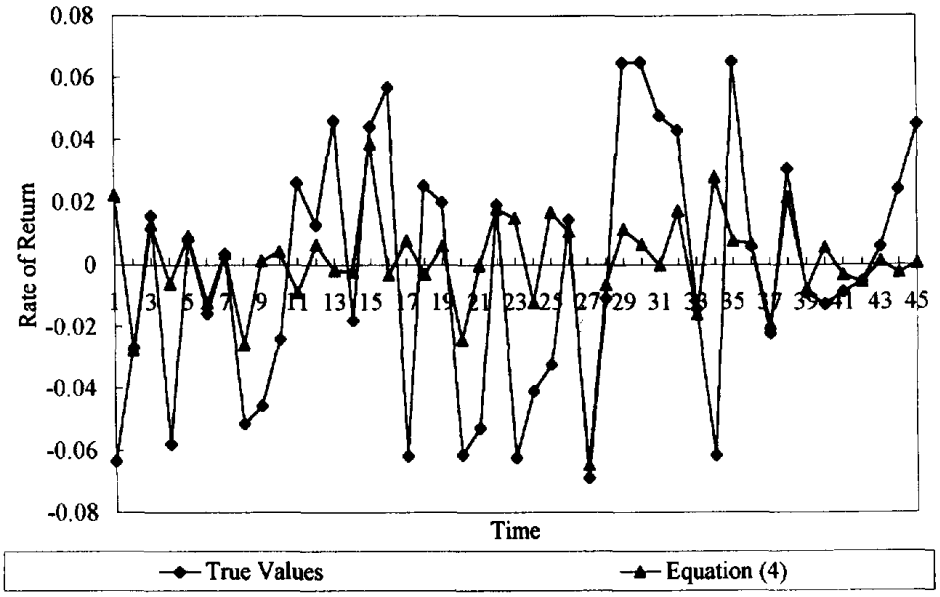


Fig. 10. The actual and predicted rate of return (TAIEX).

this case, it would be premature to treat 'beating RWs' and 'knowledge discovery' as two sides of the same coin.

Of course, since the relation is weakly negative, there are still many simulations that have superior performance in both the in-sample and post-sample period, and the model or the function chosen from these simulations can be considered a kind of regularity. For example, the best models for TAIEX chosen from simulations 21 and 24 are written in Eqs. (4) and (5) and the best for S&P 500 chosen from simulations 32 and 43 are written in Eqs. (6) and (7). The plots of the actual R_t and the predicted R_t calculated from Eqs. (4)–(7) are drawn in Figs. 10–13.

$$R_t = (((R_{t-4} + R_{t-2} * (R_{t-7} * \mathrm{Log}(((R_{t-4} + (R_{t-1} - (R_{t-2} + R_{t-7}))) * ((R_{t-6} - R_{t-7}) * \mathrm{Log}(R_{t-4} \% \mathrm{Log}\, R_{t-8}))) + (((R_{t-6} * (R_{t-6} + R_{t-4})) + R_{t-2}) * R_{t-10})))) + (R_{t-3} * (R_{t-6} + R_{t-6}))). \quad (4)$$

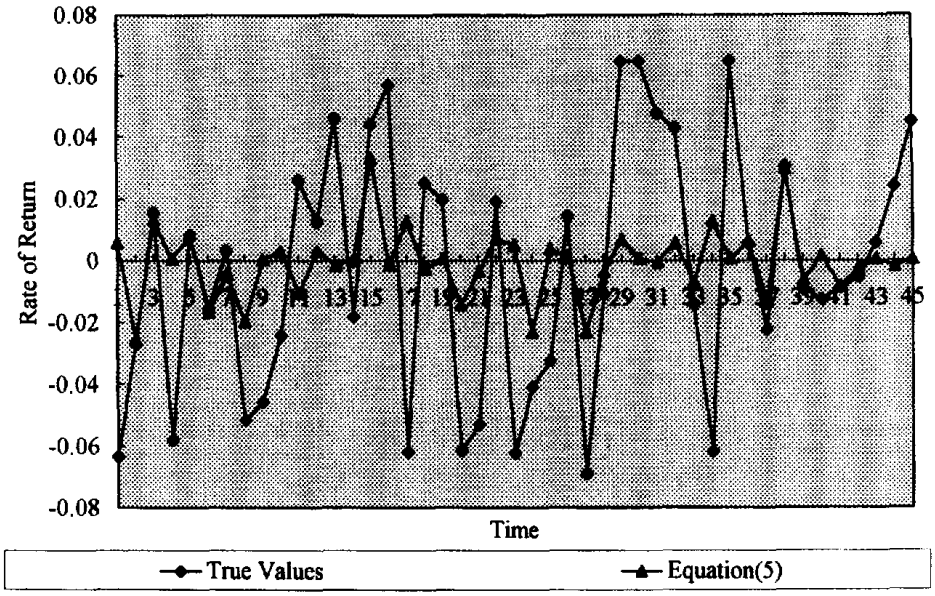


Fig. 11. The actual and predicted rate of return (TAIEX).

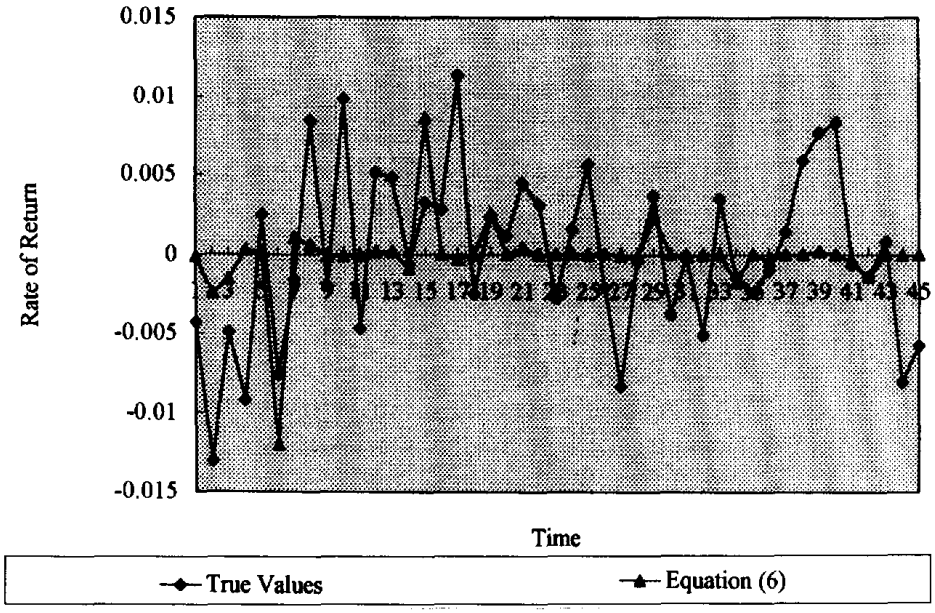


Fig. 12. The actual and predicted rate of return (S&P500).

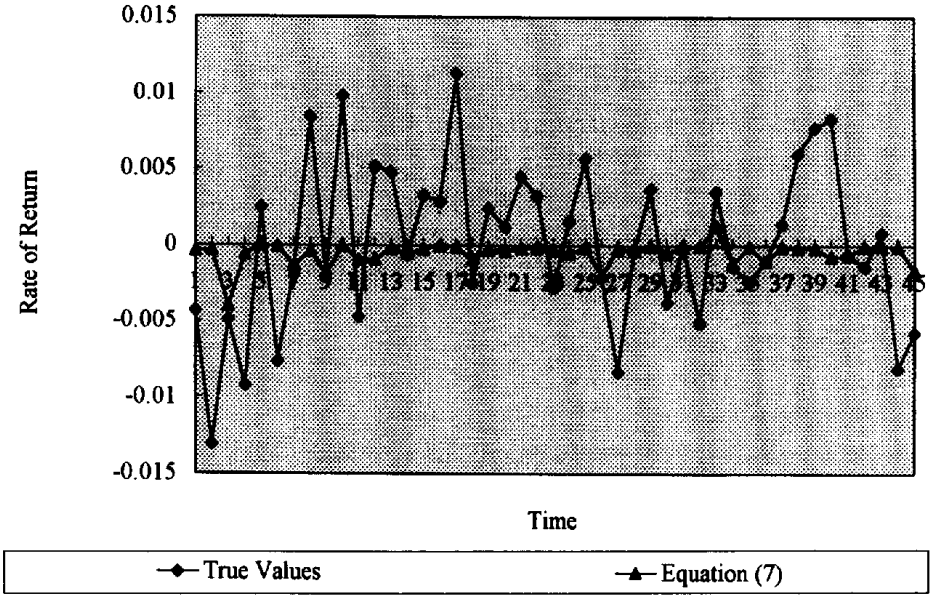


Fig. 13. The actual and predicted rate of return (S&P500).

$$R_t = (R_{t-7} * (\mathrm{Log}((((\mathrm{Log\,Sin\,Log\,Sin}R_{t-10} + R_{t-10}) + (R_{t-3} - ((\mathrm{Cos\,Log\,Sin}$$
$$R_{t-10} * (R_{t-8} * 8.317340)) - R_{t-5}))) - (((R_{t-7}\%(\mathrm{Cos}((((R_{t-5} *$$
$$(R_{t-9} * (R_{t-6} + R_{t-2}))) * R_{t-9}) + R_{t-9}) - R_{t-8}) * (R_{t-7} * R_{t-2}))) *$$
$$(R_{t-8} * R_{t-5})) * ((R_{t-3} - R_{t-4}) * \mathrm{Log}(R_{t-7} * R_{t-6})))) * R_{t-5}) *$$
$$(R_{t-4} + R_{t-2}))) \tag{5}$$

$$R_t = (R_{t-9} * ((R_{t-4} * (R_{t-2} * -8.448746))\%((R_{t-8} * (R_{t-10} + R_{t-6}))$$
$$\%\, R_{t-2}))) \tag{6}$$

$$R_t = (((\mathrm{Log}(((((R_{t-3} + R_{t-4}) * 3.034796) + R_{t-3}) * (R_{t-4} + R_{t-8})) * ((R_{t-8}$$
$$+R_{t-4}) * 3.034796)) + (((R_{t-8} + (R_{t-3} * ((R_{t-4} + ((((R_{t-3} + (R_{t-9}$$
$$*(R_{t-9} * R_{t-7}))) * (R_{t-1} * R_{t-5})) + \mathrm{Log\,Log}((R_{t-9} - R_{t-7}) * R_{t-1}))$$
$$*R_{t-5})) - (R_{t-3} - R_{t-9})))) - ((R_{t-4} + R_{t-8})\%R_{t-2})) - \mathrm{Exp}((((R_{t-3}$$
$$+R_{t-4}) * 3.034796) - R_{t-6}) * (R_{t-5}\%R_{t-8})))) + (((\mathrm{Log}((((R_{t-3}$$
$$+R_{t-4}) * 3.034796) + R_{t-3}) * (R_{t-6} - (((R_{t-3} + R_{t-4}) * 3.034796)$$
$$+(R_{t-8} * (R_{t-5} * ((R_{t-8} + R_{t-4}) * 3.034796)))))) - R_{t-2})$$
$$+R_{t-9}) + R_{t-10})) * \mathrm{Exp\,Log}((R_{t-9} - R_{t-7}) * R_{t-1})). \tag{7}$$

5. Conclusion

This paper explicitly considers the role search intensity could play in establishing a well-founded understanding of the EMH. This does not mean that traditional notions based on probabilistic independence neglect the importance of search. However, due to the lack of a search program, the test of the EMH has proceeded in an order which almost runs parallel with the development of time series analysis, e.g., from linear to nonlinear and from stochastic to chaotic. To some extent, this process can be considered a search, but such a search is not explicit enough for a measure of search intensity to be objectively constructed. Consequently, it is difficult to see how search costs and profits can be formalized in a practical way.

In his article, 'Efficient Market Hypothesis', Malkiei (1987) discussed the *weak form of the EMH*,

> Thus, investors cannot devise an investment strategy to yield abnormal profits on the basis of an analysis of past price patterns. (p.127)

By our formalization, search intensity implies search costs. A highly intensive search naturally implies that computation resources are highly involved. Hence, the costs and the associated profits are explicitly seen in the GP-based search. Thus, this paper contributes to a better understanding of Malkei's statement and the essence of the EMH.

Appendix A. simple description of genetic programming

A.1. The structure of a program

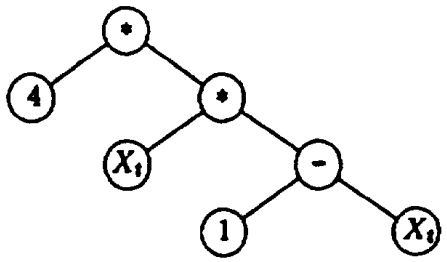
Genetic programming is basically composed of two parts: one, the *initialization* to randomly generate an initial population of programs, GP_0, and the other, the *dynamics* which gives the population of programs for n , i.e., $\{GP_n\}$, $n = 1,2,3....$ The dynamics of GP works in a pretty similar fashion to that of genetic algorithms (GAs). They are all done by applying the operation of Darwinian selection, crossover and mutation. However, there is an essential difference between GAs and GP which makes GP a generalization of GAs. Unlike GAs whose population is composed of *fixed-length binary strings*, the population of GP is composed of *programs*. In particular, each program in GP_t is written in its *LISP S-Expression* and can be depicted as a *parse tree*. For a nice introduction on how this actually works, we refer to Abelson and Sussman (1985). Briefly speaking, by appropriately defining the *terminal set* and *functional set*, each program can be written in terms of LISP S-Expression. For example, consider the logistic map

$$x_{t+1} = 4x_t(1 - x_t).$$

Define the terminal set to be $\{R, x_t\}$ where R is a constant and define the functional set to be { +, -, *, % }. Then the S-expression of the logistic map is

$$(* \ 4 \ (* \ x_t \ (-1 \ x_t))),$$

and the parse tree for this S-expression can be depicted as follows:



A.2. Genetic operators

A.2.1. Reproduction

Reproduction makes the copies of individual parse trees. The criterion used in copying is the normalized fitness value $\pi_{i,t}$. If $gp_{i,t}$ is an individual in the population GP_t with the normalized fitness value $\pi_{i,t}$, it will be copied into the next generation with probability $\pi_{i,t}$. The operation of reproduction does not create anything new in the population and the offsprings generated by reproduction constitute only part of the population GP_{t+1}. As specified in Table 1, reproduction is performed on only 10% (50 out of 500) of the population. The rest of the offsprings are generated by the other operators, such as *crossover* and *mutation*.

A.2.2. Crossover

The crossover operation for the genetic programming paradigm is a sexual operation that starts with two parental parse trees which are randomly selected from population GP_t in accordance with the normalized fitness described above. Next, by exchanging the parts of these parents, two offsprings are produced. This exchange begins by randomly and independently selecting one point in each parental parse tree using a uniform distribution described below.

By the syntax of LISP, each point (atom) of a parse tree could be either a *leaf* (terminal) or a *root* (function). Therefore, the point (atom) selected could either be a leaf or a root. As specified in Table 1, the probability that the crossover point is a root or a leaf is the same, i.e., one-half. Given that a root or a leaf is to be the point chosen for crossover, the probability that any root or leaf is chosen as the crossover point is uniformly distributed. For example, if the crossover point is to be a root, and then there are three roots in the parse tree, the probability that any one of the three roots is chosen for the crossover point is one-third (1/3). Unlike reproduction, the crossover operation creats new

individuals in populations. As specified in Table 1, 60% (300 out of 500) of the new-generation population is created in this way.

A.2.3. Mutation

The operation of mutation also allows new individuals to be created. It begins by selecting a parse tree $gp_{i,t}$ from the population GP_t based on $\pi_{i,t}$. Once a particular $gp_{i,t}$ is selected, mutation is a process of a random change of the value of a point (atom) within $gp_{i,t}$. Each point (atom) has a small probability of being altered by mutation, which is independent of other points (atoms). As specified in Table 1, the probability used throughout this paper is 0.2. The altered individual is then copied into the next generation of the population. 10% (50 out of 500) of the new-generation population is created in this way.

A.2.4. Immigration

The rest 10% of the offspring are immigrants that are created by randomly generating 50 parse trees. The motivation behind using the operator immigration is similar to that of using the operator mutation. They are all designed to equip the system with enough adaptability to avoid being trapped into a local optimum. The difference between mutation and immigration is that the latter will function completely independent of the ancestors while the former does not. Hence, immigration allows the system to be even more flexible.

References

Abelson, H., Sussman, G.J., Sussman, J., 1985. Structure and Interpretation of Computer Programs. MIT, Cambridge.

Bachelier, L., 1990. Théorier de la speculation. Annales Scientifiques de l'Ecole Normale Supérieure 17, 21–88.

Brock, W., Dechert, W., Scheinkman, J., 1987. A Test for Independence Based on the Correlation Dimension. Working paper, University of Wisconsin at Madison, University of Houston, and University of Chicago.

Chen, S., Lin, C., Yeh, C., 1995. On the model selection and its stability of the natural vacancy rate of housing: an application of genetic programming. Journal of Housing Studies 3, 73–98.

Chen, S., Yeh, C., 1995. Predicting chaotic dynamic systems with genetic programming. Proceedings of the 50th Session of the International Statistical Institute, Beijing.

Chen, S., Tan, C., 1995. On the stochastic complexity of Taiwan stock returns: An Application of Rissanen's MDL Principle. Paper presented at the 1st Conference on the Theory and Practice of Probability and Statistics, Department of Applied Mathematics, National Chengchi University, Taipei, Taiwan, R.O.C..

Diebold, F.X., Nason, J.A., 1990. Nonparametric exchange rate prediction? Journal of International Economics 28, 315–332.

Devaney, R., 1989. An Introduction to Chaotic Dynamical Systems. Addison-Wesley, Redwood City, 2nd edition.

Frank, M.Z., Gencay R., Stengos, T., 1988. International chaos? European Economic Review 32, 1569–1584.

Fogler, H.R., 1995. Investment analysis and new quantitative tools. Paper presented at the 1995 Eatern Finance Association Meeting in Hilton Head, South Carolina.

Hsieh, D.A., 1989. Testing for nonlinear dependence in daily foreign exchange rates. Journal of Business 62, 339–368.

Hinich, M.J., Patterson, D.M., 1985. Evidence of nonlinearity in daily stock returns. Journal of Business and Economic Statistics, Vol. 3, No. 1, 69–77.

Koza, J., 1992. Genetic Programming: On the Programming of Computers by Means of Natural Selection. MIT, Cambridge.

LeBaron, B., 1992. Nonlinear Forecasts for the S& P Stock Index. In: M. Casdagli, S. Eubank (eds.), Nonlinear Modeling and Forecasting. Addison-Wesley, New York, pp. 381–393.

Makridakis, S., 1993. Accuracy measures: theoretical and practical concerns. International Journal of Forecasting 9, 527–529.

Malkiei, B. G., 1987. Efficient market hypothesis. In: J. Eatwell, M. Milgate, P. Newman (Eds.), The New Palgrave: Finance, Norton, London, pp. 127–134.

Oakley, E. H. N., 1994. The application of genetic programming to the investigation of short, noisy, chaotic data series, In: T.C. Fogarty (Ed.), Evolutionary Computing, Springer, Berlin, pp. 320–332.

Peters, E. E., 1991. A chaotic attractors for the S&P 500. Financial Analysis Journal, March–April, 55–62.

Rissanen, J., 1982. A universal prior for integers and estimation by minimum description length. Annals of Statistics 11, 416–431.

Ruelle, D., 1990. Deterministic Chaos: The Science and the Fiction. Proceedings of Royal Society London A 427, 241–248.

Savit, R., 1988. When random is not random: An introduction to chaos in market prices. Journal of Futures Markets 8, 271–290.

Savit, R., 1989. Nonlinearities and chaotic effects in option prices. Journal of Futures Markets 9, 507–518.

Scheinkman, J. A., LeBaron, B., 1989. Nonlinear dynamics and stock returns. Journal of Business 3, 311–337.

Willey, T., 1992. Testing for nonlinear dependence in daily stock indices. Journal of Economics and Business 44, 63–74.