A Measure of Time Series' Predictability Using Genetic Programming

Based on standard genetic programming (GP) paradigm, we introduce a new test of time series' predictability. It is an index computed as the ratio of two fitness values from GP runs when searching for a series' data generating process. One value belongs to the original series, while the other belongs to the same series after it is randomly shuffled. Theoretically, the index boundaries are between zero and 100, where zero characterizes stochastic processes while 100 typifies predictability. This test helps in reducing model search space and in producing more reliable forecast models.

KEY WORD: Complexity; Nonlinearity; Artificial intelligence; Search algorithms.

1. INTORDUCTION

Genetic programming is a search technique introduced by Koza (1992). It is useful in finding a symbolic structural model that characterizes the dynamical behavior of sequential data sets. The method has been rather successful in finding the underlying dynamical data generating process (or DGP) in many areas if properly used. Lensberg (1997), Fernandez and Evett (1997), Oussaidene et al. (1996), Chen and Yeh (1996), and Chen and Yeh (1997) used it to study financial markets for example. McKay et al. (1996) and Greeff and Aldrich (1997) applied it to industrial processes. Bettenhausen and Marenbach (1995), Hickey (1996), Hiden et al. (1997), and Willis et al. (1997) applied it to chemical and biological processes. Schoenauer et al. (1996) used it to find mechanical models. GP was also used to predict chaotic signals in Fogel and Fogel (1996), Jonsson and Barklund (1996), Mulloy et al. (1996), and in Oakly (1996).

We use the results from GP's search for dynamic structures to introduce a new test of time series' predictability. The aim is to minimize the search space when attempting to identify a series' data generating process. Implementing predictability or complexity of time series tests helps reduce high costs of search and specification error. Search space can be reduced if there are many independent variables that can possibly be included and some choices must be made. Perhaps such a test can help identify clusters of variables that best explain the dependent variable. This test also helps find the best possible specification and proper structural form. Linear (L), nonlinear (NL), and L-NL combination structures are all included in the search for the best specification. Therefore such a test may help delete extraneous information and to find the best if not true model specification.

There are many tests that apply when analyzing time series. Lee et al. (1993) compares a neural network test with five others that test for linearity. All tests in the comparison were sensitive to departures from linearity in mean and some may have power against series linear in mean with ARCH. These tests are only a first step toward analyzing methods capable of unambiguous detection of neglected nonlinearity in sequential data. Oakly (1996) used GP to test for chaos. He finds that chaotic signals (or nonlinear deterministic processes sensitive to initial conditions) yield richer GP processes. He did not introduce a specific measure that clearly identifies chaotic or other signals. Kaboudan (1998) proposed a test for complexity. It is based on the correlation dimension estimates of a series before and after randomly shuffling it. The test works well when the data set tested is sufficiently long (at least 1,000 observations). Thus there remains a need to discover a test that measures the complexity or predictability of scarce data. The test proposed in this paper caters to achieving this objective at the cost of computational time. While the Kaboudan (1998) test for complexity is data demanding, its computational time is relatively short when compared with the proposed new test.

We introduce the proposed new test (η) in Section 3 after reviewing the genetic programming paradigm and symbolic regressions used in the search for model structures in the next Section. Section 4 contains an evaluation of the test performance using experimental data. It also contains a comparison between the proposed predictability test and Kaboudan's previous complexity test. An application of the two tests to financial time series is in Section 5. The conclusion is in Section 6.

2. GENETIC PROGRAMMING

From a forecaster's point of view, GP is a form of computerized model specification. Koza (1992) created a special form of programming called *symbolic regression* that specializes in searching for the 'best' dynamical structural that predicts data series. To find that structure, a GP computer program is given the dependent variable, a host of many possible independent variables, and a set of operators. The independent ones may be lagged dependent variables in a time series model, or a set of possible other explanatory variables determined by the analyst searching for a behavioral model. The operators are mathematical functions including the basic arithmetic ones (+ ; - ; * ; ÷) as well as trigonometric, exponential, square root, and logarithmic operators. The program then randomly selects and combines independent variables and operators in search for a model specification that would satisfy some user-specified fitness function. A minimum tolerable error at all data points signals the program to terminate the search if it is reached. When a program is executed, it generates many combinations of independent variables and operators. Each combination is known as an individual (often referred to as a tree and sometimes as an equation). A maximum number of variables and operators in any equation is also specified by the user and are known as the tree depth. Each set of equations simultaneously evaluated is a population. The user also sets the number of individuals to include in a single population. A GP code is written to solve each individual equation to generate predicted values of the dependent variable. Each observation's error is computed and the sum of squared errors is obtained for each individual. That individual characterized as fittest or possessing the lowest SSE is kept in memory, and the program generates a new population of individuals. This is done using an evolutionary process where crossover and mutation produce the next generation. Crossover is a process involving exchanging part of one individual (equation) with part of another. Mutation is a process of simply replacing a part of an individual with new part(s). Here the fittest individuals get a higher chance of survival and producing new equations or individuals. The weaker individuals get a lower chance and ultimately die. This process of creating new "fitter" generations continues until a user-specified number of generations is reached. The program saves only the fittest individual for each generation and among all populations in a file containing the final results.

Although GP seems logical and may in fact - under proper circumstances - yield a 'best' equation to describe the dynamical process generating a time series, it is not free from problems. Here are three:

1. The algorithm is a search process in a huge space. Therefore, it is conceivable that the program gets trapped in a local minimum SSE in the search space. This means that there is no guarantee that the program will find that global minimum SSE. To avoid this problem it is essential to conduct the search a sufficiently large number of times, hoping that one of these will find that global minimum SSE equation or individual. Unfortunately, this problem also means that it is extremely difficult if not impossible to reproduce results. This is due to the random selection of the initial individuals' structures, and the randomness of crossover and mutation.

2. There are so many user-specified parameters that rely entirely on that user's experience and intuition. This problem is so vast; there are numerous studies that attempt to address only partials of it. It is sufficient here to mention only a few, especially that no attempt will be made in this study to resolve these issues, and since these problems have little to no impact on the development of the predictability test. The first, and perhaps of concern to practitioners, is the selection of sample size one uses to obtain the symbolic regression. There is no agreement in the literature on what may be optimal, but there is general agreement that a small sample size is usually sufficient. Small here is defined as less than 100 data points. Fernandez and Evett (1997) conducted a study just to evaluate

the effects of varying the sample size. They studied trading profitability using three different training periods. They found that profitability chances increased when the training period is medium (150 days), followed by small (50 days), and worst for large (5 years). The second perhaps more important problem is the selection of the population size and number of generations. Gathercole and Ross (1997) address the problem of selecting the appropriate population size and number of generations to run. While Koza (1992) favors larger population sizes, Gathercole and Ross find that small populations with many generations is better than the opposite. A third, and perhaps less serious problem, is the selection of crossover and mutation rates. Harries and Smith (1997) address this problem. For the development of the test proposed in this paper, the parameters selected to complete all runs are in Table 1, and are discussed later.

3. The efficacy of the search process is affected by the magnitudes of random constants in equations generating a series. One can only postulate the reason. It seems that the larger the random constant, the larger the search space the program has to go through to 'find' that constant. Investigations during this study suggest normalizing the data one is modeling to a reasonable range (such as ±1, and not more than ±10) prior to completing a GP run. Fortunately, this problem has little to no effect on the test proposed in this study either as shown in the next section. None of the experimental data needed normalization, but stock returns in Section 5 did.

There are many GP packages one can use to obtain symbolic regression results. We selected Andy Singleton's GPQuick (1995) in C++ after tailoring it to accept input files of time series and print the final output in two separate files. One output file contains the fittest equation in each generation and the final equation, and the other contains the measures of fitness, R^2 and SSE. Table 1 contains the parameters used for all runs in this study. They are basically the default ones the developer of the software specified except for a few: the population size, the number of generations, the sample size, the tolerance error, and the crossover and mutation rates. Sample size, population size, and number of generations were selected after completing a Monte Carlo study comparing the differences and gains from varying three parameters. The other parameters were arbitrarily selected based on what was used in other studies.

3. THE PREDICTABILITY TEST: η

The new test is a measure of percentage predictable information in a variable Y_t. SSE is one way to measure unpredictable information. If Y_t actually contains predictable information, SSE_Y should be reasonably low. Randomly shuffling the sequence of Y_t will ruin predictable information in the original sequence. Accordingly, Y_t has low while its randomly shuffled sequence S_t has high entropy. The proposed test is based on comparing two SSE values from estimating the two sets of data, series Y_t and that same series after it is randomly shuffled. Random shuffling is obtained using Efron's (1982) bootstrap method. The method randomly samples data with replacement. This yields a new scrambled series with very similar statistical characteristics. The comparison between the two SSE values is then feasible and logical given that the variables' units of measurement affect these values.

GP is used to find the best fit equation and compute SSE for a given series Y_t representing a sample of the variable Y collected over t = 1,…,T time periods. The prediction error is:

$$SSE_Y = \sum_{t=1}^{T}(Y_t - \overset{\wedge}{Y_t})^2, \qquad (1)$$

where $\hat{Y}$ are the predicted values of Y_t. SSE of S_t is:

$$SSE_S = \sum_{t=1}^{T}(S_t - \overset{\wedge}{S_t})^2. \qquad (2)$$

If Y_t is deterministic, $SSE_Y < SSE_S$. Intuitively, the more deterministic Y_t is, the lower will be the ratio of $SSE_Y \div SSE_S$. Therefore, such ratio is a measure of the proportion of entropy in Y_t before shuffling. Alternatively, if the Y_t is stochastic, using GP or any equation estimation method should not have any predictive power, and at least theoretically, the ratio of $(SSE_Y \div SSE_S) \approx 1$. This means that predictive information entropy of the stochastic Y_t was at 100 percent before shuffling.

Statistically, the probability of predictability and non-predictability are mutually exclusive and collectively exhaustive events. This means that if a variable is x% predictable, it must (100-x%) non-predictable. Given that GP is a random search mechanism, the ratio $SSE_Y \div SSE_S$ will differ from one run to the other. Therefore, although the ratio may initially seem as a good measure of the proportion of entropy in the data, it is an inconsistent one. Consistency may be introduced by taking an average of SSE over a sufficiently larger number of runs. If the mean before and after shuffling SSE remains the same, then there was no increase in information entropy due to shuffling; Y_t is stochastic. If the mean SSE increases, shuffling must have dismembered information.

Therefore, the degree of predictability relative to a series' shuffled sequence may be measured by:

$$\eta = 100 * (1 - (\overline{SSE}_Y \div \overline{SSE}_S), \tag{3}$$

where

$$\overline{SSE_Y} = k^{-1} * SSE_Y, \tag{4}$$

k is the number of GP runs performed in search for the fittest equation, and $\overline{SSE}_S$ is computed similar to $\overline{SSE}_Y$ but for the shuffled data.

The test measures the percentage hypothetical reversed entropy (or gain in information) if a shuffled or randomized series were put back to its original order or sequence. It is a percentage based of a statistical mean resulting from a random search. Percentage boundaries should, at least theoretically, be between zero and 100. If the computed η = 100%, this means that the series is totally predictable. If η = 0, this means that the series is totally unpredictable. Somewhere between zero and 100% is the proportion of the series that is predictable using GP or perhaps any other forecasting methodology. Practically, however, the lower boundary may be violated. This is possible since a stochastic Y_t and its shuffled counterpart are somewhat predictable using GP as the highest R^2 values for GS in Table 2 indicate. Therefore, it is not unusual to find that the search was more successful in finding some determinism in the shuffled data a number of times greater than that found for the original random set. Under such conditions, $\eta < 0$. Given that the shuffled series is useless, even if it is found more predictable than its originally random series, the final form for the proposed test is conditional. Formally, the final η test is:

$$\eta \;=\begin{cases} 0, & if \quad (\overline{SSE}_Y \div \overline{SSE}_S) > 1, \\ 100 * (1 - (\overline{SSE}_Y \div \overline{SSE}_S), & otherwise. \end{cases} \tag{5}$$

4. EVALUATION AND COMPARISON

To evaluate the performance of the proposed predictability test and ascertain that its results are meaningful, eight experimental series with known dynamical characteristics were artificially generated and used. Their dynamical structures contain linear, linear stochastic, nonlinear, nonlinear stochastic, and pseudo-random. No attempt was made in this study to control the signal-to-noise ratio of the data used in testing. Although desirable and was used by Kaboudan (1998), it is left for future investigation. The eight structural data generating processes selected here serve this investigation. They are different in complexity and should produce different degrees of predictability. The eight data generating processes investigated in this study are:

(1) A simple linear model – OZ:

$$Y_t = 1.8708 \quad Y_{t-1} - Y_{t-2}. \tag{6}$$

This is a noise free linear function capable of generating an infinite time series. It is known as the Ozaki function and is studied by Tong (1990, p. 76).

(2) The logistic map - LG:

$$Y_t = 4 \quad Y_{t-1}(1 - Y_{t-1}). \tag{7}$$

This is a nonlinear chaotic function in Grassberger and Procaccia (1983) and often cited in the chaos literature. It is a discrete function that generates an infinite time series.

(3) The Henon map - HN:

$$Y_t = 0.3 \quad Y_{t-2} + 1 - 1.4 \quad Y_{t-1}^2. \tag{8}$$

This is also a nonlinear chaotic function in Grassberger and Procaccia (1983) and is widely studied in chaos theory as well. It also provides discrete series.

(4) Trigonometric function - TF:

$$Y_t = 3.9 \quad \sin \quad Y_{t-1} + 0.85 \quad \cos \quad Y_{t-2}. \tag{9}$$

This is a simple nonlinear trigonometric function capable of producing an infinite time series. This transcendental function was created for this study.

(5) Exponentially weighted coefficients function - EF:

$$Y_t = (1.43 - 4.5 \quad e^{-Y_{t-1}^2})Y_{t-2}. \tag{10}$$

This transcendental function is a difference equation with complex roots. It is studied by Tong (1990, p. 71).

(6) AR2 model - AR:

$$Y_t = 0.6 \quad Y_{t-1} + 0.15 \quad Y_{t-2} + \varepsilon_t, \tag{11}$$

where $\varepsilon_t \sim N(0,1)$. This is a second order autoregressive model that generates an infinite series. It is a linear-stochastic model.

(7) A GARCH(1,1) - GR:

$$\begin{aligned} Y_t &= \varepsilon_t \sqrt{h_t}, \\ h_t &= 1 + 0.25 \quad Y_{t-1}^2 + 0.7 \quad h_{t-1}, \end{aligned} \tag{12}$$

where $\varepsilon_t \sim N(0,1)$. This is a generalized autoregressive model with conditional heteroscedasticity. It is a nonlinear-stochastic system capable of generating an infinite time series.

(8) Random - GS:

This is pseudo-random data with Gaussian characteristics, also N(0,1). It was generated using the statistical software package RATS.

Although the dynamic structure of data generated by these eight processes is known, one can only hypothesize about their predictability. It is highly possible that the linear data is most predictable, followed by nonlinear data without noise. Linear stochastic is probably more predictable than the nonlinear stochastic, and the random set is not predictable. We will now apply two tests to measure complexity and predictability of the experimental data: a complexity test proposed in Kaboudan (1998) and the predictability test proposed above. Neither test attempts to estimate the model or process that generates the data, they just measure complexity and predictability, respectively.

To measure complexity, Kaboudan (1998) proposed a two step method that discriminates between linear, linear stochastic, nonlinear, nonlinear stochastic, and random data. First a data set is filtered from linear structure using an AR(*p*) model with *p* determined following Akaike's (1974) method, where *p* is the number of lags. The filtering equation yields an R^2 that measures the series' linear DGP component. The residuals are then measured for complexity using a θ statistic. If the θ estimate is close to 1, the data is stochastic, and if it is close to zero, the data is deterministic. Chaotic data were found with $0.3 < \theta < 0.6$. Complexity increased when deterministic signals were tarnished with different levels of noise. The θ test is basically a relative measure of the change in a series correlation dimension estimate after it is randomly shuffled.

Different data sample sizes were needed for the two tests. To measure complexity, each DGP produced 1,500 observations first. The top 400 points were discarded to ensure that the data used was on the attractor thus avoiding bias from the selection of starting values. Linear filtering produced residuals that are a minimum of 1,000 observations for which θ was measured. Each of the residuals produced was shuffled 1,000 times to estimate a mean θ, thus avoiding any statistical bias due to random shuffling. To measure predictability, only 112 observations were selected per series, specifically observations 501 to 612 from the 1,500 originally generated. One hundred GP runs were then completed for each using twelve lagged values.

Table 2 presents both complexity and predictability measures for the eight functions listed above. Each DGP was abbreviated by two letters (above) and identified by them in the Table. The second and third columns contain complexity measures. The balance of the Table contains predictability statistics. η was estimated as the average of only the best fifty runs per series. Fifty is an arbitrary selection of the number of SSE results to average. It was selected for two reasons: It is a number sufficiently large to provide a meaningful average, and it suppresses the effect of those runs that generate meaningless results when they are trapped in a local minimum.

These results are fairly consistent and confirm one's intuition about predictability of less complex processes. Here are some observations about the information in the Table reached by observing both the η percentage and the highest R^2 for each process:

a. The linear process is surely predictable.

b. Nonlinear processes free from noise are also quite predictable but may vary in their predictability level. The variation is not significantly large, however.

c. Exponential weighted coefficients functions are the least predictable among the nonlinear ones studied here. This is not surprising given the structure of the DGP in equation (10) above.

d. Predictability decreased significantly when a stochastic component is part of the process. It is obvious and intuitive that linear stochastic is more predictable than nonlinear stochastic. It is surprising to find GARCH data as unpredictable as random series, however.

5. APPLICATION TO FINANCIAL TIME SERIES

In this section, we apply the tests to real world time series with unknown dynamical structures. Three Dow Jones stocks are analyzed here: IBM, GE, and AT&T. The sample is rather small but serves the purpose. A large sample is beyond the scope of this study and is left for future investigations. Time and quotes (TAQ) data available on CDs from the NYSE, Inc. covering October 1996 through March 1997 was used. Three frequencies were selected to represent each stock: Prices were collected every 30 minutes, every minute, and every price change. Returns, defined as the proportional change in price from one observation to the next, are computed for each. These returns are represented by very low numerical values that were normalized by multiplying their values by 100. This converts the returns to percentages. Traditional analysis of stocks involves analyzing time-stamped stock returns using daily data. There are two problems with analyzing such data. First, daily closing prices miss far too many price-changes in the course of a single trading day. Some of these stocks experience a price change every second of the day, and sometimes the price changes more than once per second. The second problem has to do with consistency. If one analyzes time-stamped returns, the data analyzed is inconsistent since some stocks trade heavily while others thinly. Even the trading frequency of the same stock changes within the course of a single day. It is therefore logical to make the hypothesis that price-change returns are most predictable, followed by minute-to-minute returns, while thirty-minute returns are not predictable.

Table 3 contains stock returns' complexity and predictability measured results. (PCR stands for price-change returns.) Tables 2 and 3 are the same format. Complexity statistics are first, followed by those of predictability. These results confirm our hypotheses. Here are some observations based on the results in Table 3:

a. Analysis of 30-minute data shows that only GE is slightly predictable.

b. Analysis of 1-minute data show that GE is not and the others are somewhat predictable.

c. Analysis of price-change returns shows GE as least complex followed by AT&T.

d. In general and on the average, there is a better chance predicting price-change than predicting one-minute returns, and the latter has a better chance than 30-minute returns.

e. Comparing the complexity statistics (R^2 and θ) and their relationship with η for IBM indicates that if the DGP of a series is a combination of linear-nonlinear, or linear-nonlinear-stochastic components, that series DGP is probably unpredictable. This is also evident from the one-minute and PCR statistics on AT&T. Evidenced increased complexity in such a meager sample is purely phenomenological and further investigation is clearly warranted.

6. CONCLUSION

We proposed a new test that helps forecasters in different disciplines to determine whether their data is actually predictable using GP or any equation forecasting technique. The new test involves computing percentage measure of predictability. If the computed percentage approaches 100, the data is predictable using conventional methods as well as

GP. If the computed percentage approaches zero, the data is random or unpredictable. The test was compared with Kaboudan's (1998) complexity test. They both seem to provide important information about data prior to attempting to model and forecast. Kaboudan's complexity test seems to be able to discriminate between linear, nonlinear, nonlinear chaotic, linear stochastic, nonlinear stochastic, and random data. It does not tell us whether the series is predictable. The proposed predictability test furnishes such information missed by the complexity test. There is another major difference distinguishing the two tests from each other. The complexity test is demanding when it comes to the number of observations needed to obtain statistically reliable results. The results are reliable if at least 1,000 observations are used. Its computational time is very low relative to the predictability test. One can determine complexity in a few minutes given an average PC with 230 MHz. This is not true with the predictability test. It can run with as few as 80 to 100 observations, but the running time is no less than 4 hours on the same PC.

REFERENCES

Akaike, H. (1974), "A New Look at the Statistical Model Identification," *IEEE Transactions on Automatic Control*, 19, 716-723.

Bettenhausen, K., and Marenbach, P., (1995), "Self-Organizing Modelling of Biotechnological Batch and Fed-Batch Fermentations," In F. Breitenecker and I. Husinsky, eds., *EUROSIM '95: Simulation Congress: Proceedings of EUROSIM Conference, Vienna, Austria*, Elsevier Science B.V., Amsterdam, 445-450.

Chen, S., and Yeh, C., (1996), "Genetic Programming and the Efficient Market Hypothesis," in *Genetic Programming: Proceedings of the First Annual Conference 1996*, edited by J. Koza, D. Goldberg, D. Fogel, and R. Riolo, The MIT Press, Cambridge, Massachusetts, 45-53.

Chen, S., and Yeh, C., (1997), "Using Genetic Programming to Model Volatility in Financial Time Series," in *Genetic Programming 1997: Proceedings of the Second Annual Conference*, edited by J. Koza, K. Deb, M. Dorigo, D. Fogel, M. Garzon, H. Iba, and R. Riolo, Morgan Kaufmann, San Francisco, 58-63.

Efron, B. (1982), *The Jackknife, the Bootstrap, and Other Resampling Plans*, Society for Industrial and Applied Mathematics, Philadelphia.

Fernandez, T., and Evett, M., (1997), "Training Period Size and Evolved Trading Systems," in *Genetic Programming 1997: Proceedings of the Second Annual Conference*, edited by J. Koza, K. Deb, M. Dorigo, D. Fogel, M. Garzon, H. Iba, and R. Riolo, Morgan Kaufmann, San Francisco, 95.

Fogel, D., and Fogel, L. (1996), "Preliminary Experiments on Discriminating Between Chaotic Signals and Noise Using Evolutionary Programming," in *Genetic Programming: Proceedings of the First Annual Conference 1996*, edited by J. Koza, D. Goldberg, D. Fogel, and R. Riolo, The MIT Press, Cambridge, Massachusetts, 512-520.

Gathercole, C., and Ross, P. (1997), "Small Populations Over Many Generations Can *Beat* Large Populations Over Few Generations in Genetic Programming," in *Genetic Programming 1997: Proceedings of the Second Annual Conference*, edited by J. Koza, K. Deb, M. Dorigo, D. Fogel, M. Garzon, H. Iba, and R. Riolo, Morgan Kaufmann, San Francisco, 111-118.

Grassberger, P., and Procaccia, I., (1983), "Measuring the Strangeness of Strange Attractors," Physica D, 9, 189-208.

Greeff, D., and Aldrich, C., (1997), "Evolution of Empirical Models for Metallurgical Process Systems," in *Genetic Programming 1997: Proceedings of the Second Annual Conference*, edited by J. Koza, K. Deb, M. Dorigo, D. Fogel, M. Garzon, H. Iba, and R. Riolo, Morgan Kaufmann, San Francisco, 138.

Harries, K., and Smith, P., (1997), "Exploring Alternative Operators and Search Strategies in Genetic Programming," in *Genetic Programming 1997: Proceedings of the Second Annual Conference*, edited by J. Koza, K. Deb, M. Dorigo, D. Fogel, M. Garzon, H. Iba, and R. Riolo, Morgan Kaufmann, San Francisco, 147-155.

Hickey, R. (1996), "Noise Modelling and Evaluating Learning from Examples," *Artificial Intelligence*, 82, 157-179.

Hiden, H., Willis, M., McKay, B., and Montague, G. (1997), "Non-Linear and Direction Dependent Dynamic Modelling Using Genetic Programming," in *Genetic Programming 1997: Proceedings of the Second Annual Conference*, edited by J. Koza, K. Deb, M. Dorigo, D. Fogel, M. Garzon, H. Iba, and R. Riolo, Morgan Kaufmann, San Francisco, 169-173.

Jonsson, P., and Barklund, J. (1996), "Characterizing Signal Behavior Using Genetic Programming," in T. Fogarty, ed., *Lectures in Computer Science 1143, Evolutionary Computing*, AISB Workshop, Brighton, U.K., Springer-Verlag, Berlin, 62-72.

Kaboudan, M. (1998), "Statistical Properties of Time-Series-Complexity Measure Applied to Stock Returns," forthcoming in *Computational Economics*.

Koza, J. (1992), *Genetic Programming*, The MIT Press, Cambridge, Massachusetts.

Lee, T., White, H., and Granger, C., (1993), "Testing for Neglected Nonlinearity in Time Series Models," *Journal of Econometrics, 56, 269-290.*

Lensberg, T. (1997), "A Genetic Programming Experiment on Investment Behavior under Knightian Uncertainty," in *Genetic Programming 1997: Proceedings of the Second Annual Conference*, edited by J. Koza, K. Deb, M. Dorigo, D. Fogel, M. Garzon, H. Iba, and R. Riolo, Morgan Kaufmann, San Francisco, 231-239.

McKay, B., Willis, M., Montague, G., and Barton, G. (1996), "Using Genetic Programming to Develop Inferential Estimation Algorithms," in *Genetic*

Programming: Proceedings of the First Annual Conference 1996, edited by J. Koza, D. Goldberg, D. Fogel, and R. Riolo, The MIT Press, Cambridge, Massachusetts, 157-165.

Mulloy, B., Riolo, R., and Savit, R. (1996), "Dynamics of Genetic Programming and Chaotic Time Series Prediction," in *Genetic Programming: Proceedings of the First Annual Conference 1996*, edited by J. Koza, D. Goldberg, D. Fogel, and R. Riolo, The MIT Press, Cambridge, Massachusetts, 166-174.

Oakly, H. (1996), "Genetic Programming, the Reflection of Chaos, and the Bootstrap: Toward a Useful Test for Chaos," in *Genetic Programming: Proceedings of the First Annual Conference 1996*, edited by J. Koza, D. Goldberg, D. Fogel, and R. Riolo, The MIT Press, Cambridge, Massachusetts, 175-181.

Oussaidene, M., Chopard, B., Pictet, O., and Tomassini, M., (1996), "Parallel Genetic Programming: An Application to Trading Models Evolution," in *Genetic Programming: Proceedings of the First Annual Conference 1996*, edited by J. Koza, D. Goldberg, D. Fogel, and R. Riolo, The MIT Press, Cambridge, Massachusetts, 357-362.

Schoenauer, M., Sebag, M., Jouve, F., Lamy, B., and Maitournam, H., (1996), "Evolutionary Identification of Macro-Mechanical Models," in *Advances in Genetic Programming:* vol. II, edited by P. Angeline and K. Kinnear, Jr., The MIT Press, Cambridge, Massachusetts, 467-488.

Singleton, A. (1995), Genetic Programming with C++, public domain genetic programming package, Creation Mechanics, Dublin, NH.

Tong, H. (1990), *Non-linear Time Series: A Dynamical System Approach,* Oxford University Press, Oxford.

Willis, M., Hiden, H., Hinchliffe, M., McKay, B., and Barton, G., (1997), "Systems Modelling using Genetic Programming," *Computers and Chemical Engineering*, 21, Suppl., pp. S1161-S1166.

Table 1
Specifications for GPQucik Configuration Files

Generations	20,000	Populations	2,000
Error	0.00001	Sample	100
Terminals	12	Max. expression	50
Init. Expression	6	Mutation rate	100
Cross self	1	Unrestrict. Wt.	70
Cross Wt.	100	Mut. Wt.	30
Mute node Wt.	100	Mute Const. Wt.	100
Mute shrink st	100	Copy Wt.	10
Select method	4	Tourn. size	7
Mate radius	500	Kill tourn.	2
Max. age	2000		

Table 2: Signal and Noisy Logistic Map Predictability & Complexity

	Original		Shuffled				
λ	Lowest 50% Mean SSE_Y	Standard Deviation	Lowest 50% Mean SSE_S	Standard Deviation	η	R^2	Mean θ
					□		
∞	0.83	0.64	10.81	0.23	92.35	0.00	0.39
60	2.51	0.61	11.68	0.28	78.50	0.00	0.40
50	2.13	0.36	10.16	0.25	79.03	0.00	0.41
40	2.67	0.47	12.36	2.62	78.43	0.00	0.41
30	3.04	0.47	11.63	0.29	73.84	0.00	0.42
20	4.20	1.05	11.20	0.20	62.53	0.00	0.45
15	5.56	0.70	10.28	0.27	45.91	0.00	0.49
10	6.37	0.84	14.11	0.29	54.83	0.00	0.58
5	10.21	0.33	14.07	0.37	27.42	0.00	0.76
1	22.81	0.76	20.32	0.39	0.00	0.00	1.03

Table 3: Applications of the Predictability & Complexity Measures

	Original		Shuffled				
Functions	**Lowest 50% Mean SSE_Y**	**Standard Deviation**	**Lowest 50% Mean SSE_S**	**Standard Deviation**	η	**R^2**	**Mean θ**
Linear:					□ □		
OZ	1.96	0.55	384.73	18.68	99.49	0.87	0.31
Nonlinear:							
TF	27.64	10.62	769.53	19.22	96.41	0.00	0.50
HN	4.18	1.11	41.71	1.24	89.98	0.27	0.54
EF	138.87	30.89	653.65	24.39	78.75	0.00	0.51
MG	3.71	0.13	6.50	0.40	42.92	0.49	0.75
Nonlinear-Stochastic:							
AR2	84.80	3.44	259.56	7.07	67.33	0.72	0.99
BL	226.29	24.40	469.82	36.68	51.84	0.03	0.84
GR	70.64	2.12	73.19	2.70	3.49	0.00	0.92
Random:							
GS	76.79	2.27	68.58	2.05	0.00	0.00	1.01
ER	16.36	0.29	12.19	0.70	0.00	0.00	1.01

Table 4: Predictability & Complexity of Stock Returns

Stocks	Lowest 50% Mean SSE_Y	Standard Deviation	Lowest 50% Mean SSE_S	Standard Deviation	η	R^2	Mean θ
30-minute:					□		
BA	13.86	0.70	12.66	1.99	0.00	0.07	0.92
GE	13.14	0.74	10.08	0.36	0.00	0.03	0.63
GM	11.65	0.39	8.61	0.30	0.00	0.49	0.80
IBM	14.12	0.75	10.28	0.31	0.00	0.46	0.36
S	34.24	0.69	34.99	1.35	2.15	0.01	0.93
T	9.28	0.34	7.77	0.18	0.00	0.08	0.67
WMT	46.19	1.11	53.55	2.40	13.74	0.07	0.88
XON	8.62	0.21	9.27	0.26	7.01	0.02	0.78
1-minute:							
BA	8.21	0.29	7.20	0.13	0.00	0.12	0.90
GE	0.60	0.01	0.61	0.02	1.07	0.10	1.48
GM	5.96	0.15	6.67	0.35	10.76	0.13	0.31
IBM	0.90	0.02	0.85	0.03	0.00	0.21	0.91
S	2.72	0.10	2.90	0.07	6.32	0.11	0.31
T	3.08	0.07	6.18	0.34	50.10	0.45	0.68
WMT	7.38	0.33	6.43	0.10	0.00	0.30	0.19
XON	0.90	0.02	0.89	0.03	0.00	0.11	1.28
PCRs:							
BA	1.02	0.02	1.66	0.06	38.59	0.32	0.27
GE	0.57	0.03	2.06	0.08	72.44	0.59	0.18
GM	2.50	0.05	5.99	0.31	58.32	0.54	0.32
IBM	0.85	0.05	1.23	0.04	30.75	0.28	0.44
S	1.37	0.02	4.69	0.14	70.70	0.42	0.04
T	9.76	0.30	16.68	0.83	41.48	0.59	0.35
WMT	1.95	0.51	22.79	0.40	91.46	0.83	0.80
XON	0.95	0.04	1.82	0.05	47.67	0.56	0.41