Applied Financial Economics, 2008, **18**, 357–378

Forecasting economic time series with the DyFor genetic program model

Neal Wagner[a], Moutaz Khouja[b,*], Zbigniew Michalewicz[c,d,e] and Rob Roy McGregor[f]

[a]*Department of Mathematics and Computer Science, Augusta State University, Augusta, GA 30904, USA*
[b]*Department of Business Information Systems, University of North Carolina, Charlotte, NC 28223, USA*
[c]*School of Computer Science, University of Adelaide, Adelaide, SA 5005, Australia*
[d]*Institute of Computer Science, Polish Academy of Sciences, ul. Ordona 21, 01-237, Warsaw, Poland*
[e]*Polish-Japanese Institute of Information Technology, ul. Koszykowa 86, 02-008 Warsaw, Poland*
[f]*Department of Economics, University of North Carolina, Charlotte, NC 28223, USA*

Genetic programming (GP) uses the Darwinian principle of survival of the fittest and sexual recombination to evolve computer programs that solve problems. Several studies have applied GP to forecasting with favourable results. However, these studies, like others, have assumed a static environment, making them unsuitable for many real-world time series which are generated by varying processes. This study investigates the development of a new 'dynamic' GP model that is specifically tailored for forecasting in nonstatic environments. This dynamic forecasting genetic program (DyFor GP) model incorporates methods to adapt to changing environments automatically as well as retain knowledge learned from previously encountered environments. The DyFor GP model is tested on real-world economic time series, namely the US Gross Domestic Product and Consumer Price Index Inflation. Results show that the DyFor GP model outperforms benchmark models from leading studies for both experiments. These findings affirm the DyFor GP's potential as an adaptive, nonlinear forecasting model.

I. Introduction

Forecasting is an integral part of everyday life. Businesses, governments and people alike make, use and depend on forecasts for a wide variety of concerns. Current methods of time series forecasting require some element of human judgment and are subject to error. When the variable to be forecasted is well-understood, the error may be within acceptable levels. However, oftentimes the forecasting concern is not well-understood and, thus, methods that require little or no human judgment are desired.

*Corresponding author. E-mail: mjkhouja@email.uncc.edu

Applied Financial Economics ISSN 0960–3107 print/ISSN 1466–4305 online © 2008 Taylor & Francis
http://www.tandf.co.uk/journals
DOI: 10.1080/09603100600949200

Additionally, many forecasting situations are set in environments with continuously shifting conditions. These situations call for methods that can adjust and adapt.

The aim of this study is to develop a new adaptive model that is specifically tailored for forecasting time series produced by nonstatic environments. The proposed model is based on genetic programming (GP) with additional features that seek to capture such dynamically changing time series. This dynamic forecasting genetic program (DyFor GP) model incorporates methods to adapt to changing environments automatically as well as retain knowledge learned from previously encountered environments. Such past-learned knowledge may prove useful when current environmental conditions resemble those of *a prior* setting. Specifically, this knowledge allows for faster convergence to current conditions by giving the model searching process a 'head-start'.

The rest of this article is organized as follows: Section II is a brief review of existing time series forecasting methods, Section III describes the DyFor GP model, Sections IV and V detail experiments involving the DyFor GP model and Section VI concludes.

II. Review of Existing Time Series Forecasting Methods

Time series forecasting methods generally fall into two groups: classical methods which are based on statistical/mathematical concepts and modern heuristic methods which are based on algorithms from the field of artificial intelligence.

Classical methods

Classical time series forecasting methods can be subdivided into the following categories:

(1) exponential smoothing methods,
(2) regression methods,
(3) autoregressive integrated moving average (ARIMA) methods,
(4) threshold methods and
(5) generalized autoregressive conditionally heteroskedastic (GARCH) methods.

The first three categories can be considered linear methods, that is methods that employ a linear functional form for time series modelling and the last two are nonlinear methods.[1]

In exponential smoothing a forecast is given as a weighted moving average of recent time series observations. The weights assigned decrease exponentially as the observations get older. In regression a forecast is given as a linear function of one or more explanatory variables. ARIMA methods give a forecast as a linear function of past observations (or the differences of past observations) and error values of the time series itself and past observations of zero or more explanatory variables. See Makridakis *et al.* (1998) for a discussion of smoothing, regression and ARIMA methods.

All of the linear forecasting methods above assume a functional form which may not be appropriate for many real-world time series. Linear models cannot capture some features that commonly occur in actual data such as asymmetric cycles and occasional outlying observations (Makridakis *et al.*, 1998, pp. 433–434). Regression methods often deal with nonlinear time series by logarithmic or power transformation of the data, however this technique does not account for asymmetric cycles and outliers.

Threshold methods assume that extant asymmetric cycles are caused by distinct underlying phases of the time series and that there is a transition period (either smooth or abrupt) between these phases. Commonly the individual phases are given a linear functional form and the transition period (if smooth) is modelled as an exponential or logistic function. GARCH methods are used to deal with time series that display nonconstant variance of residuals (error values). In these methods the variance of error values is modelled as a quadratic function of past variance values and past error values. In Makridakis *et al.* (1998), McMillan (2001) and Sarantis (2001) various threshold methods are detailed while Akgiray (1989), Bollerslev (1986) and Engle (1982) describe GARCH methods.

The nonlinear methods, although capable of characterizing features found in data such as asymmetric cycles and nonconstant variance of residuals, assume that the underlying data generating process of the time series is constant. (The linear methods described earlier also make this assumption). This assumption is often invalid as shifting environmental conditions may cause the underlying data generating process to change. For all of the classical forecasting methods listed, human judgment is required to first select an appropriate method and then find appropriate parameter values for the model (or to select an appropriate parameter optimization scheme). In the event that the underlying data generating process changes, the time series data must be reevaluated and

[1] Regression and ARIMA methods can be given a nonlinear functional form, however, this is not common.

a (possibly new) method must be selected with appropriate parameter values. Because the task of repeated data monitoring and model selection is complex and time consuming, automatic nonlinear forecasting models that can handle nonstatic environments are desired.

Modern heauristic methods

Most modern heuristic methods which were applied to time series forecasting fall into two major categories:

(1) methods based on neural networks (NN) and
(2) methods based on evolutionary computation.

The latter category is further divided into methods based on genetic algorithms (GA), evolutionary programming (EP) and GP.

NN, EP and GP techniques were used to build nonlinear forecasting models, whereas genetic algorithms were primarily used to tune the parameters of some (possibly statistical, linear or nonlinear) forecasting model. All of the methods listed above are motivated by the study of biological processes.

Neurel network attempt to solve problems by imitating the mechanism used by the human brain. A NN is a graph-like structure that contains an input layer, zero or more hidden layers and an output layer. Each layer contains several 'neurons' which have weighted connections to neurons of the following layer. A neuron from the input layer holds an input variable. For forecasting models, this input is a previous time series observation or an explanatory variable. A neuron from the hidden or output layer consists of an 'activation' function.

Neurel networks are a widely adopted heuristic technique for time series forecasting and have received considerable attention for business applications (Swanson and White, 1995; Smith and Gupta, 2002). A multilayered feedfoward NN is the most popular type of NN for forecasting applications (Smith and Gupta, 2002). Figure 1 shows the feedforward NN architecture when a single hidden layer is used and a single output is to be produced.

A forecast (output) is generated in the following way. First each neuron in the hidden layer sums up its (weighted) input connections (from the input layer) and then applies its activation function to the sum. The result is then passed (via weighted connection) to the next layer. Each neuron in the next layer then sums up its inputs and applies its activation function. This procedure is continued up to the output layer. The forecast is the result of the output layer neuron's activation function. NNs employ a training algorithm to optimize the weight values of the network.

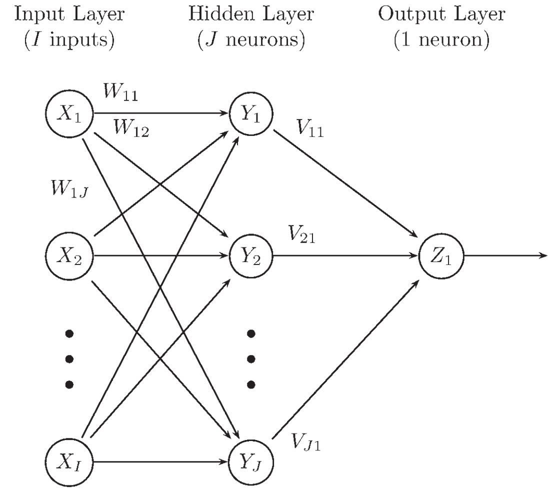

Fig. 1. Feedforward NN architecture: W_{ii} and V_{ii} represent weighted connections
Note: Not all weights are shown.

Several applications of NNs to forecasting are proffered in Gately (1996), Smith and Gupta (2002) and Trippi and Turban (1996). General descriptions of NNs can be found in Gurney (1997) and White (1992).

For methods based on evolutionary computation, the process of biological evolution is mimicked in order to solve a problem. After an initial population of potential solutions is created, solutions are ranked based on their 'fitness' (i.e. their quality relative to the optimal). New populations are produced by selecting higher-ranking solutions and performing genetic operations of 'mating' (crossover) or 'mutation' to produce offspring solutions. This process is repeated over many generations until some termination condition is reached.

When GA is applied to forecasting, first an appropriate model is selected and an initial population of candidate solutions is created. A candidate solution is produced by randomly choosing a set of parameter values for the selected forecasting model. Each solution is then ranked based on its prediction error over a set of training data. A new population of solutions is generated by selecting fitter solutions and applying a crossover and mutation operation. Crossover is performed by swapping a subset of parameter values from two parent solutions. Mutation causes one (random) parameter from a solution to change. New populations are created until the fittest solution has a sufficiently small prediction error or repeated generations produce no reduction of error. GA has been used successfully for a wide variety of difficult optimization problems including forecasting. Bäck (1996), Michalewicz (1992) and Mitchell (1996) give detailed descriptions of GA while

Chambers (1995), Chiraphadhanakul *et al.* (1997), Deboeck (1994), Goto *et al.* (1999), Ju *et al.* (1997), Kim and Kim (1997) and Venkatesan and Kumar (2002) provide additional examples of GA applied to forecasting.

For EP each candidate solution is represented as a finite state machine (FSM) rather than a numeric vector. FSM inputs/outputs correspond to appropriate inputs/outputs of the forecasting task. An initial population of FSMs is created and each is ranked according to its prediction error. New populations are generated by selecting fitter FSMs and randomly mutating them to produce offspring FSMs. EP was devised by Fogel *et al.* (1966) and has applications in many areas including forecasting (Fogel *et al.*, 1966, 1995; Fogel and Chellapilla, 1998; Sathyanarayan *et al.*, 1999).

In GP solutions are represented as tree structures instead of numeric vectors or finite state machines. Internal nodes of solution trees represent appropriate operators and leaf nodes represent input variables or constants. For forecasting applications, the operators are mathematical functions and the inputs are lagged time series values and/or explanatory variables. Figure 2 gives an example solution tree for time series forecasting. Variables x_{t1} and x_{t2} represent time series values one and two periods in the past, respectively.

Crossover is performed by exchanging subtrees from two parent solutions. Figure 3 illustrates the crossover operation. $p1$ and $p2$ represent parent solutions ($x_{t1} + \sin(5.31x_{t2})$ and $\cos(-1.29x_{t1}) - x_{t4}$, respectively) and $o1$ and $o2$ are the offspring solutions created ($x_{t1} + \sin(x_{t4})$ and $\cos(-1.29x_{t1}) - 5.31x_{t2}$, respectively). Mutation is performed by selecting a subtree of a single solution and replacing it with a randomly constructed subtree.

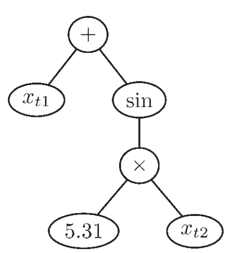

Fig. 2. Genetic programming representation of forecasting solution $x_{t1} + \sin(5.31x_{t2})$

Genetic programming was developed by Koza (1992) as a problem-solving tool with applications in many areas. Numerous studies have applied GP to time series forecasting with favourable results (Andrew and Prager, 1994; Chen and Yeh, 1996; Jonsson and Barklund, 1996; Mulloy *et al.*, 1996; Lee *et al.*, 1997; Chen *et al.*, 1998; Hiden *et al.*, 1998; Kaboudan, 1998, 1999, 2000, 2001, 2003; Iba and Sasaki, 1999; Iba and Nikolaev, 2000; Wagner and Michalewicz, 2001; Neely and Weller, 2002). GP has also been used to find successful trading rules from time series data in Fyfe *et al.* (1999), Neely *et al.* (1997), Neely (2003) and Wang (2000).

Several forecasting studies involving hybrid heuristic methods have been undertaken. A common variation is a method that combines NN and GA. In these applications, a GA is used to optimize several aspects of a NN architecture. The optimized NN is then used to produce the desired forecasts (Maniezzo, 1994; Back *et al.*, 1996; Sexton, 1998; White, 1998; Phua *et al.*, 2001; Andreou *et al.*, 2002; Leigh *et al.*, 2002; Nag and Mitra, 2002).

Because the heuristic methods described above are nonlinear, they are able to capture many aspects displayed by actual data. NN, EP and GP have the added advantage that the forecasting model need not be prescribed, allowing for automatic discovery of a befitting functional form. However, like the classical methods, these methods assume a static environment. If the underlying data generating process shifts, the methods must be reevaluated in order to accomodate the new process. Additionally, these methods require that the number of historical time series data used for analysis be designated *a priori*. This presents a problem in nonstatic environments because different segments of the time

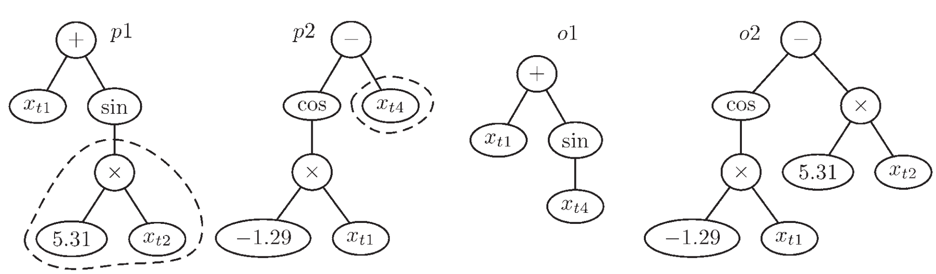

Fig. 3. Genetic programming crossover: $o1$ and $o2$ are offspring solutions of $p1$ and $p2$. Dashed lines enclose subtrees to be exchanged

series may have different underlying data generating processes. For example, a time series representing the daily stock value of a major US airline is likely to have a different underlying process before 11 September 2001 than it does afterwards. If analysed time series data span more than one underlying process, forecasts based on that analysis may have unduly large errors.

Consider the subset of time series data shown in Fig. 4. Suppose this represents the most recent historical data and has been chosen for analysis. Suppose further that the subset consists of two segments each with a different underlying process. The second segment's process represents the current environment and is valid for forecasting future data. The first segment's process represents an older environment that no longer exists. Because both segments are analysed, the forecasting model is distorted unless human judgment is brought to bear.

Some degree of human judgment is necessary to assign the number of historical data to be used for analysis. If the time series is not well-understood, then the assignment may contain segments with disparate underlying processes. This situation highlights the need for forecasting methods that can automatically determine the correct analysis 'window' (i.e. the correct number of historical data to be analysed). This investigation attempts to develop a dynamic forecasting model based on GP that can do just that. Furthermore, this study explores methods that can retain knowledge learned from previously encountered environments. Such past-learned knowledge may prove useful when current environmental conditions resemble those of *a prior* setting. Specifically, this knowledge allows for faster convergence to current conditions by giving the model searching process a 'head-start' (i.e. by narrowing the model search space).

III. The DyFor GP Model

As discussed in the previous section, an adaptive forecasting model that can handle nonstatic environments is sought. The desired model would automatically determine the appropriate analysis window (i.e. the number of recent historical data whose underlying data generating process corresponds to current environment). Also, the model should be able to adapt to changing conditions 'on-the-fly' (i.e. without the need for halting and restarting the analysis). An additional boon would be the ability to retain useful knowledge from previously encountered environments so that the current setting can be more accurately captured. In this section a discussion of the design of such a model is proffered.

$$\underbrace{22,33,30,27,24,20,21,20,20}_{\text{segment1}},\underbrace{23,26,29,30,28,29,32,30,31}_{\text{segment2}}|\underbrace{\ldots}_{\text{future}}$$

Fig. 4. Time series containing segments with differing underlying processes

Natural adaptation: a sliding window of time

In biological evolution organisms evolve to suit the occurrent conditions of their environment. When conditions shift, successful organisms adapt to the new surroundings. Over many generations and several environmental shifts, enduring organisms represent highly adaptive solutions that can survive and thrive in a variety of settings. A time series arising from real-world circumstances can be viewed in a similar light. Different segments of the time series may be produced by different underlying data generating processes. Each segment can be thought of as one set of environmental conditions. A successful forecasting model might be seen as an adaptive organism that has evolved through all of the pre-existing environments and gained valuable adaptations along the way.

To model this natural adaptation through many environmental settings, a sliding window of time is proposed. For the DyFor GP model, analysis starts at the beginning of the available historical data. Some initial windowsize (number of data observations to analyse) is set and several generations of DyFor GP are run to evolve a population of solutions. Then the data window slides to include the next time series observation. Several generations are run with the new data window and then the window slides again. This process is repeated until all available data have been analysed up to and including the most recent historical data. Figure 5 illustrates this process. In the figure, | marks the end of available historical data. The set of several generations run on a single

$$\underbrace{22,33,30,27,24,20,21,20,20}_{\text{window}-1},23,26,29,30,28,29,32,30,31|\underbrace{\ldots}_{\text{future}}$$

$$22,\underbrace{33,30,27,24,20,21,20,20,23}_{\text{window}-2},26,29,30,28,29,32,30,31|\underbrace{\ldots}_{\text{future}}$$

$$\vdots$$

$$22,33,30,27,24,20,21,20,20,\underbrace{23,26,29,30,28,29,32,30,31}_{\text{window}-i}|\underbrace{\ldots}_{\text{future}}$$

Fig. 5. A sliding data analysis window

analysis window is referred to as a 'dynamic generation'. Thus, a single run of the DyFor GP includes several dynamic generations (one for each window slide) on several different consecutive analysis windows.

This sliding window feature allows the DyFor GP to analyse all existing data and take advantage of previously observed patterns. As the window slides through past data, solutions glean useful knowledge making it easier for them to adapt to and predict the current environment.

Adapting the analysis window

Designating the correct size for the analysis window is important to the success of any forecasting model. Automatic discovery of this windowsize is critical when the forecasting concern is not well-understood. With each slide of the window, the DyFor GP adjusts its windowsize dynamically. This is accomplished in the following way.

(1) Select two initial windowsizes, one of size n and one of size $n+i$ where n and i are positive integers, $i < n$.
(2) Run dynamic generations at the beginning of the historical data with windowsizes n and $n+i$, use the best solution for each of these two independent runs to predict a number of future data points and measure their predictive accuracy.
(3) Select another two windowsizes based on which windowsize had better accuracy. For example if the smaller of the 2 windowsizes (size n) predicted more accurately, then choose 2 new windowsizes, one of size n and one of size $n-i$. If the larger of the 2 windowsizes (size $n+i$) predicted more accurately, then choose windowsizes $n+i$ and $n+2i$.
(4) Slide the analysis window to include the next time series observation. Use the two selected windowsizes to run another two dynamic generations, predict future data and measure their prediction accuracy.
(5) Repeat the previous two steps until the analysis window reaches the end of historical data.

Thus, at each slide of the analysis window, predictive accuracy is used to determine the direction in which to adjust the windowsize.

Consider the following example. Suppose the time series given in Fig. 6 is to be analysed and forecasted. As depicted in the figure, this time series consists of two segments each with a different underlying data generating process. The second segment's process represents the current environment. The first segment's process represents an older environment that no longer exists but may contain patterns that can be learned and exploited when forecasting the current environment. If there is no knowledge available concerning these segments, automatic techniques are required to discover the correct windowsize needed to forecast the current setting. The DyFor GP starts by selecting two initial windowsizes, one larger than the other. Then, two separate dynamic generations are run at the beginning of the historical data, each with its own windowsize. After each dynamic generation, the best solution is used to predict some number of future data and the accuracy of this prediction is measured. Figure 7 illustrates these steps. In the figure, **win1** and **win2** represent data analysis windows of size 3 and 4, respectively and **pred** represents the future data predicted.

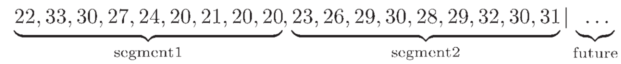

Fig. 6. Time series containing segments with differing underlying processes

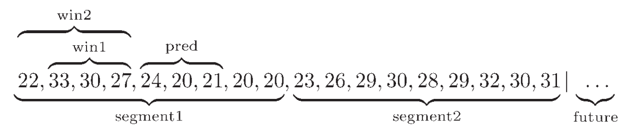

Fig. 7. Initial steps of window adaptation

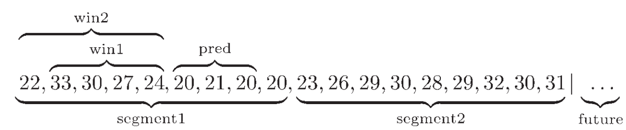

Fig. 8. Window adaptation after the first window slide
Note: **Win1** and **win2** have size 4 and 5, respectively.

The data predicted in these initial steps lies inside the first segment's process and, because the dynamic generation involving analysis window **win2** makes use of a greater number of appropriate data than that of **win1**, it is likely that **win2**'s prediction accuracy is better. If this is true, two new windowsizes for **win1** and **win2** are selected with sizes of 4 and 5, respectively. The analysis window then slides to include the next time series value, two new dynamic generations are run and the best solutions for each are used to predict future data. Figure 8 depicts these steps. In the figure data analysis windows **win1** and **win2** now include the next time series value, 24 and **pred** has shifted one value to the right.

This process of selecting two new windowsizes, sliding the analysis window, running two new

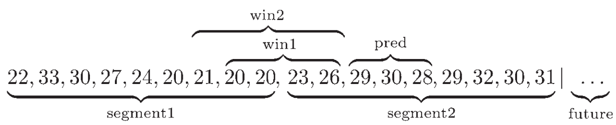

Fig. 9. Window adaptation when analysis spans both segments
Note: The smaller analysis window, **win1**, is likely to have better prediction accuracy because it includes less inappropriate data.

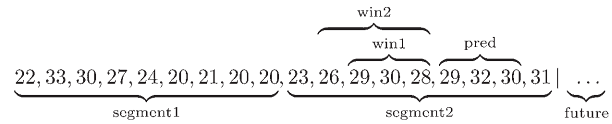

Fig. 11. Window adaptation when analysis lies entirely inside the second segment
Note: The larger analysis window, **win2**, is likely to have better prediction accuracy because it includes a greater number of appropriate data.

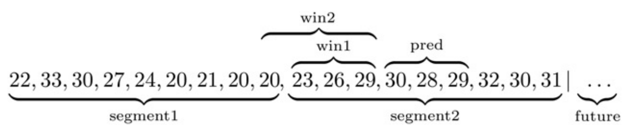

Fig. 10. Window adaptation when analysis spans both segments
Note: **Win1** and **win2** have contracted to include less inappropriate data.

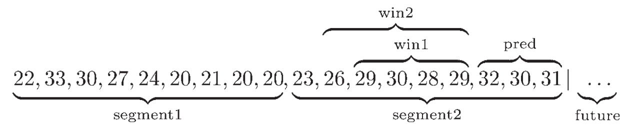

Fig. 12. Window adaptation when analysis lies entirely inside the second segment
Note: **Win1** and **win2** have expanded to include a greater number of appropriate data.

dynamic generations and predicting future data is repeated until the analysis window reaches the end of historical data. It may be noted that while the prediction data, **pred**, lies entirely inside the first segment, the data analysis windows, **win1** and **win2**, are likely to expand to encompass a greater number of appropriate data. However, after several window slides, when the data analysis window spans data from both the first and second segments, it is likely that the window adjustment reverses direction. Figures 9 and 10 show this phenomenon. In Fig. 9 **win1** and **win2** have sizes of 4 and 5, respectively. As the prediction data, **pred**, lies inside the second segment, it is likely that the dynamic generation involving analysis window **win1** has better prediction accuracy than that involving **win2** because **win1** includes less data produced by a process that is no longer in effect. If this is so, the two new windowsizes selected for **win1** and **win2** are sizes 3 and 4, respectively. Thus, as the analysis window slides to incorporate the next time series value, it also contracts to include a smaller number of inappropriate data. In Fig. 10 this contraction is shown.

After the data analysis window slides past the end of the first segment, it is likely to expand again to encompass a greater number of appropriate data. Figures 11 and 12 depict this expansion.

As illustrated in the above example, the DyFor GP uses predictive accuracy to adapt the size of its analysis window automatically. When the underlying process is stable (i.e. the analysis window is contained inside a single segment), the windowsize is likely to expand. When the underlying process shifts (i.e. the analysis window spans more than one segment), the windowsize is likely to contract. In order to test these window-adjusting dynamics of the DyFor GP model, a preliminary experiment was undertaken and is discussed in Section IV. The following section discusses how the DyFor GP model can retain and exploit knowledge of previously-encountered environments.

Retaining and exploiting knowledge from past environments

Existing forecasting methods rely, to some degree, on human judgment to designate an appropriate analysis window, that is the window of historical data whose underlying process corresponds to the current environment. If a time series is produced in a nonstatic environment, frequently only the recent historical data that correspond to the current environment are analysed and historical data that come from previous environments are ignored.

What if the current environmental conditions resemble those of *a prior* environment? In such a case, knowledge of this prior environment might be used to capture the current environment with greater speed and/or accuracy than a search that ignores this knowledge. Existing forecasting methods, assuming that the analysis window has been correctly set, do not benefit from knowledge of past environments and, thus, must search for a model of the current environment 'from scratch'. The sliding window feature allows the DyFor GP to analyse all historical data and take advantage of knowledge gleaned from previously encountered environments, giving the model search a 'head-start'. This knowledge comes in the form of adaptations (i.e. solution subtrees) gained by evolution through these previous environments. Past-evolved subtrees are used by the DyFor GP as promising exploration points from which to search for a model that is appropriate for the current environment. These subtrees are retained and exploited in two ways:

(1) implicitly by the evolutionary process when it is coupled with the sliding window feature of the DyFor GP and
(2) explicitly through the use of 'dormant' solutions.

The following two sections discuss these ways in which past-evolved subtrees are maintained and utilized. For the remainder of this article we will refer to such subtrees as 'adaptations'.

Implicit adaptation: the role of introns

In biology unexpressed genotypic regions are commonly called introns. For GP, this term has been adopted to refer to inactive regions in the solution representation, that is subtrees of a solution which do not affect its fitness (Angeline, 1994; Brameier and Banzhaf, 2001). Consider the solution tree depicted in Fig. 13. This solution tree represents the expression

$$2.53(x_{t-2})^2 + \left(\frac{2.53x_{t-3}}{x_{t-2}}\right)(x_{t-1} - x_{t-1})$$

which after simplification becomes

$$2.53(x_{t-2})^2$$

The % sign in the figure represents a protected division operator that does not allow division by zero. In the figure the intron subtree (enclosed by dashed lines) does not affect the fitness of the solution as its output simplifies to zero regardless of the values given for variables x_{t-1}, x_{t-2} and x_{t-3}.

A well-known characteristic of the GP process is the tendency for evolved solution trees to have introns make up a significant percentage of the tree structure. This was first recognized by Koza (1992, p. 7). Several studies have suggested that introns are a beneficial component in the evolutionary search for optimal solutions (Iba *et al.*, 1994; Nordin and Banzhaf, 1995; Levenick, 1999). Introns are seen as particularly valuable when the environment is nonstatic (Levenick, 1999). To understand how introns are utilized to retain and exploit previously learned adaptations, consider the following example.

Suppose the time series given in Fig. 14 is analysed. This time series consists of three segments each with a different underlying data generating process. The third segment's underlying process represents the current environment and is valid for forecasting future data. The first and second segments' processes represent older environments but may contain information that can be used to more accurately capture the current environment. Suppose further that similar environmental conditions produce segments 1 and 3 while differing conditions produce segment 2. The aim is to retain adaptations learned from segment 1 and utilize these adaptations to find an appropriate model for segment 3. The DyFor GP

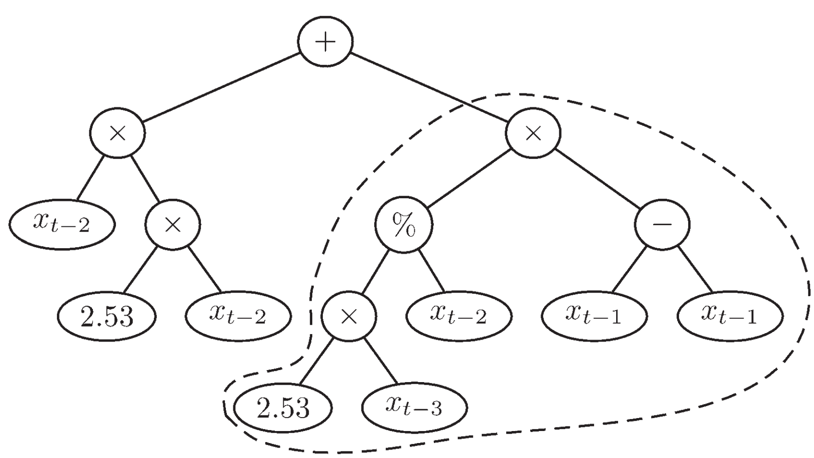

Fig. 13. A GP solution tree containing an intron. Dashed lines enclose the intron subtree

$$\underbrace{22, 33, 30, 27, 24, 20, 21, 20, 20}_{\text{segment1}}, \underbrace{23, 26, 29, 30, 28, 29, 32, 30, 31}_{\text{segment2}}, \underbrace{31, 32, 30, 30, 31, 28, 29, 28, 29}_{\text{segment3}} | \underbrace{\ldots}_{\text{future}}$$

Fig. 14. Time series containing segments with differing underlying processes

sets its analysis window at the beginning of segment 1's data and starts the evolutionary process in search of a model. Perhaps, after several dynamic generations inside segment 1, the solution tree of Fig. 15 is evolved as a befitting model. This solution tree represents the expression

$$12.33 + \left(\frac{2.53x_{t-3}}{x_{t-2}}\right)(\cos(x_{t-1} - x_{x-1}))$$

which simplifies to

$$12.33 + \left(\frac{2.53x_{t-3}}{x_{t-2}}\right) \qquad (1)$$

In the figure, suppose the subtree rooted by the protected division operator (%) is an adaptation that fits the environmental conditions of segment 1. This subtree is equivalent to the second term of the Equation 1.

When the DyFor GP's analysis window moves into segment 2, this adaptation is no longer suitable as the environmental conditions have changed. Nevertheless, through crossover this adaptation can be retained by becoming a part of an intron subtree in a fit solution for segment 2. Figures 16 and 17 illustrate this phenomenon. In Fig. 16, the adaptation equivalent to the second term of Equation 1 is part of tree $p1$ and is to be exchanged with a subtree of tree $p2$. Figure 17 gives the offspring solution trees produced. In the figure, the adaptation is now a part of offspring solution tree $o2$. Furthermore, the adaptation is contained in an intron subtree of this offspring (the same intron subtree as depicted in Fig. 13).

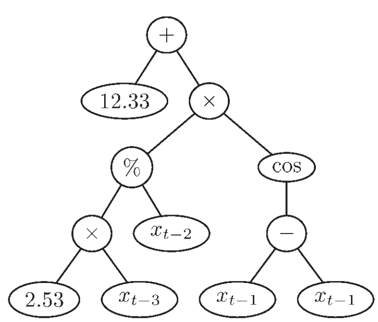

Fig. 15. An evolved solution tree for segment 1

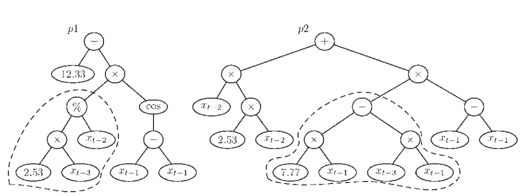

Fig. 16. Retention of a no longer suitable adaptation via crossover. $p1$ and $p2$ are parent solution trees to undergo crossover. Dashed lines enclose subtrees to be exchanged

Thus, the adaptation evolved during analysis of segment 1 can be retained as the DyFor GP analyses segment 2 even though this adaptation is not relevant for segment 2's environment. While this retention of a previously learned adaptation may be possible, one may ask if it is likely. Given that the adaptation in question suits the environment of segment 1, the evolutionary process is likely to produce many solutions containing the adaptation when the DyFor GP analyses segment 1's data. When the analysis window switches to segment 2's data to start analysis of this new environment, natural selection will tend to favour these fitter solutions from segment 1 and, thus, solutions with this adaptation will be chosen for crossover many times. Therefore, it is likely that a number of those crossovers will result in the adaptation being moved to an intron subtree as described in the above example especially given the fact that a large percentage of GP solution trees are made up of introns. Hence retention of past-evolved adaptations into intron subtrees is a probable scenario.

When the analysis window slides to segment 3's data, it is likely that some solution trees in the population contain the adaptation evolved from segment 1 as a part of an intron subtree. Since segment 3's environment resembles that of segment 1, solutions that contain the adaptation in an active subtree will survive and thrive. Just as crossover can move an adaptation from an active subtree to an intron subtree, it can also move an adaptation from an intron subtree back to an active one.

The aforementioned example illustrates how evolved adaptations from previously encountered environments can be retained in nonapplicable environments by becoming part of intron subtrees and can then be reactivated in applicable environments by moving back to active (nonintron) subtrees. A preliminary experiment which tests these dynamics was undertaken and is discussed in Section IV. The following section discusses an explicit method of maintaining and exploiting past-evolved adaptations through the use of 'dormant' solutions.

Explicit adaptation: dormant solutions

The DyFor GP also contains a feature that explicitly saves evolved adaptations from past environments and then injects them back into the evolutionary process when conditions are suitable. This feature involves the use of 'dormant' solutions, that is solutions that remain inactive during environments with inapplicable conditions becoming active only

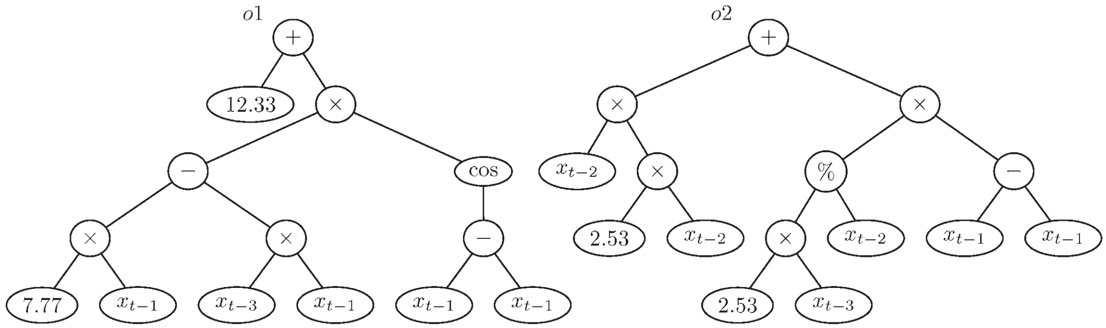

Fig. 17. Retention of a no longer suitable adaptation via crossover. $o1$ and $o2$ are offspring solutions produced after crossover is performed on $p1$ and $p2$ from Fig. 16

when applicable conditions arise. Section 'Adapting the analysis window' explains how the DyFor GP adapts the size of its analysis window dynamically. It is noted that when the analysis window lies entirely inside a segment of historical data generated by a single underlying process, the window is likely to expand to encompass a greater number of appropriate data. Conversely, it is shown that when the analysis window spans historical data generated by more than one underlying process, the window is likely to contract to include a smaller number of inappropriate data. A fortunate side effect of this windowsize adjustment is that the boundaries of each underlying process can be deduced. Consecutive expansions of the analysis window describe a segment of historical data with a stable underlying process. Consecutive contractions of the analysis window signal that a shift in environmental conditions has occurred and that a new underlying process is currently coming into effect.

The idea is to save fit solutions evolved during segments when the underlying process is stable to be used later for quicker capture of new environmental conditions when the underlying process shifts. This is accomplished by the following steps.

(1) As the analysis window of the DyFor GP slides, note the direction of window adjustment.
   (a) N consecutive window expansions are likely to signal the beginning of a stable process. Here N is a pre-specified control parameter of the DyFor GP.
   (b) N consecutive window contractions are likely to signal the beginning of a process shift.
(2) If a stable process is signaled, save a few fit solutions as potential dormant solutions.
(3) For each further windowslide in which expansion is observed, replace the potential dormant solutions previously saved with new ones (i.e. fit solutions for the current dynamic generation).
(4) When a process shift is signaled, the most recently saved potential dormant solutions become actual dormant solutions and are saved permanently.
(5) Now, because a process shift is in effect, inject all actual dormant solutions saved from previous environments (with the exception of those saved from the most recent previous environment) into the GP population to compete with current solutions. Injected dormant solutions that contain adaptations applicable to the current environment will survive and thrive.
(6) Keep injecting these actual dormant solutions at each windowslide until a stable process is again signaled. Once a stable process has been signaled, go back to step #2 and continue.

In order to test the efficacy of dormant solutions, a preliminary experiment was undertaken. This experiment is discussed in the Section IV. The following section discusses two concerns that have an important impact on the performance of the DyFor GP model.

Forecast combination and fitness measures

In many forecasting situations, the 'best' forecasting model is not known and, thus, several 'good' forecasting models are developed. A forecaster is then faced with the problem of choosing a single forecast from a set of several candidate forecasts produced by each of the forecasting models employed. Many times it is better to combine multiple forecasts into one. This issue is called forecast combination and its relationship to the DyFor GP model is discussed subsequently.

Evolution-based techniques such as the DyFor GP use Darwin's principle of 'survival of the fittest' and sexual recombination (crossover) as a way of designing computer programs to solve complex problems. For these kinds of methods, some fitness measure or fitness function is used to measure the quality of candidate solutions. However, it may not be clear how to select such a measure for a particular problem. It may be that a single measure performs well under certain conditions but badly in others.

The GP algorithm is a fitness-driven random search. When GP is applied to forecasting, the search-space is the set of all possible mathematical equations that can be constructed using specified operands (explanatory variables) and mathematical operators. This space is quite large and, in general, intractable for most conventional deterministic algorithms. The size of the search-space coupled with the stochastic nature of the evolutionary process cause the results of a GP-based forecasting experiment to vary from run to run. Thus, a common practice is to execute a set of GP runs (usually 20 to 100) and designate the forecasts of the best run as the result (see, for example, Kaboudan, 2001, 2003). In the real world this practice is not useful since one cannot know which run produces the best forecast for a given time period without first knowing the corresponding actual value of that time period.

The DyFor GP model is based on the GP algorithm and, thus, it is necessary to execute a set of DyFor GP runs. Therefore, at any given time period, there is a set of multiple forecasts to choose from. Here, it becomes necessary to apply some forecast combination method to produce a single useful forecast. To be useful in a real-world setting, the forecast must be generated using an out-of-sample methodology where no data beyond the point of forecast is utilized for analysis, model construction, or forecast combination.

Many forecast combination methods have been developed and, in general, they can be divided into to two groups, variance-covariance methods and regression methods (Diebold, 1998). In variance-covariance methods, the combination model is

$$F = \alpha_1 f_1 + \alpha_2 f_2 + ,\ldots, + \alpha_n f_n \qquad (2)$$

where F is the combined forecast, $f_1, f_2, \ldots, f_n$ are the single forecasts to be combined and $\alpha_1, \alpha_2, \ldots, \alpha_n$ are corresponding weights that sum up to one. Optimal weights for this equation are estimated by minimizing the variance of past forecast errors.

For regression methods, the following combination model is used:

$$F = \beta_0 + \beta_1 f_1 + \beta_2 f_2 + ,\ldots, + \beta_n f_n + e \qquad (3)$$

where β_0 is a constant, $\beta_1, \beta_2, \ldots, \beta_n$ are regression coefficients and e is an error term. Here, the coefficients are estimated by regressing actual time series values on past forecasts for those values.

Variations on these combining methods are numerous and a discussion of these variations can be found in Diebold (1998, pp. 347–365). Some examples of recent studies which focus on forecast combination include Billio *et al.* (2000), Dekker *et al.* (2004), Fischer and Harvey (1999), Hendry and Clements (2004) and Yang (2004).

Since forecast combination is not the focus of this study, we restrict our attention to simple, well-known combining techniques. Thus, the following procedure is selected for combining multiple DyFor GP forecasts produced by a set of multiple DyFor GP runs into a single, out-of-sample forecast.

(1) For the first forecast, designate the median forecast of the set as the single forecast to be used.
(2) For all remaining forecasts, repeat the following:
    (a) Compare the previous forecast of all runs to the actual data for that (already past) time period and rank each run based on its accuracy.
    (b) Select the current forecast of the top 3 runs from this ranking, compute the average of these 3 forecasts and designate this average as the single forecast to be used.

This procedure is a form of the variance-covariance combination method in which only the most recent past forecast of each run is considered when estimating the weights, the weight assigned to forecasts of the 3 top-ranked runs $= 1/3$ and the weight assigned to the remaining forecasts $= 0$.

Choosing the fitness measure to be employed by the DyFor GP model is of great importance. Most GP forecasting applications use a mean squared error (MSE). To date, there have been no significant studies investigating alternative fitness measures for GP forecasting applications. One alternative fitness measure is the mean absolute deviation (MAD). Comparing the MSE and MAD measures, it can be seen that the error value of MSE grows quicker than that of MAD in the presence of noisy data. Thus, outliers tend to influence analyses based on MSE more than they do analyses based on MAD. The question of which fitness measure to employ would depend upon the characteristics of the time series.

Another interesting possibility is to develop a new 'combined' fitness (CF) measure that incorporates aspects of both the MSE and MAD. The idea is to use MSE when data encountered is not considered an outlier and MAD when data encountered is

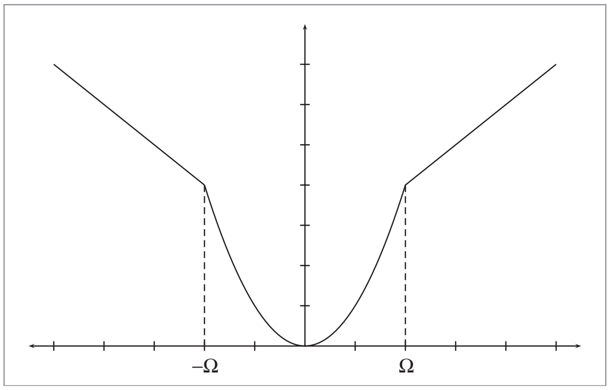

Fig. 18. The CF measure as a function of the relative error

considered an outlier. This CF measure requires a user-specified parameter, Ω, to determine which data are outliers and which are not. Figure 18 gives a graphical depiction of the CF measure as a function of the relative error. From the figure, when the relative error is within the threshold given by Ω, CF measure values follow those of the squared error. However, when the relative error falls outside of the Ω threshold, CF measure values follow those of the absolute deviation.

The DyFor GP model could potentially use any one of the above fitness measures for a given forecasting experiment. Therefore, in order to be applicable over a diverse range of forecasting tasks, the DyFor GP model includes a parameter specifying which of these three fitness measures should be employed during a run.

IV. Testing the DyFor GP: A Preliminary Experiment

In the previous section the DyFor GP model was presented and its features discussed. Some DyFor GP features are based on intuitions that, although logical, might benefit from some empirical evidence. With this in mind, a preliminary experiment was undertaken. The goal of the preliminary experiment is to test the following DyFor GP dynamics:

(1) Windowsize adjustment—windowsize is likely to expand when a time series' underlying data generating process is stable and windowsize is likely to contract when the underlying process shifts.
(2) An adaptation (evolved subtree) from a previously encountered underlying process (segment) can be retained during analysis of a new, nonsimilar process by becoming part of an intron subtree.
(3) If the current underlying process is similar to a past process, explicit retention and re-injection of dormant solutions can be used to capture this current process with greater speed and/or accuracy than could be accomplished if dormant solutions are not used.

To allow for these tests, an artificial time series was constructed consisting of three segments, each segment being a small time series generated by a known process. Furthermore, the first and third segments were generated by similar (but not equivalent) processes while the second segment was generated by a dissimilar process. Equation 4 gives the underlying process used to generate the entire time series.

$$f(x) = \begin{cases} \sin(x) + \sqrt{x} & \text{for } 1 \le x \le 2 \text{ (segment 1)} \\ e^x + 2 & \text{for } 21 \le x \le 40 \text{ (segment 2)} \\ \sin(x) - \sqrt{x} + 22 & \text{for } 41 \le x \le 60 \text{ (segment 3)} \end{cases} \tag{4}$$

The time series has 60 observations corresponding to $x = 1 \ldots 60$, 20 for each segment.

Values of the explanatory variable x are utilized by the DyFor GP as input and outputs generated are one-step-ahead forecasts for $f(x)$. Two DyFor GP runs are executed, one run with dormant solutions and one run without. The first 14 time series values are used as initial training data and 46 one-step-ahead forecasts are generated that correspond to actual time series values beginning at value #15 and ending at value #60. Because these are preliminary experiments on an artificial time series, forecast combination is not used.

The DyFor GP model requires that a number of parameters be specified before a run. Some of these are general GP parameters commonly found in any GP application while others are special parameters only used by the DyFor GP model. Table 1 gives the general GP parameters and their assigned values while Table 2 lists parameters and assigned values that are specific to the DyFor GP model.

All parameter values listed in Table 1 were selected to match those used by Koza (1992) for his experiments in symbolic regression[2] with the following exceptions.

(1) The 'max. no. of generations' parameter has a slightly different meaning when

[2] 'Symbolic regression' is the term Koza uses to describe the search for a mathematical expression that closely fits a given finite sample of data.

Table 1. General GP parameter settings for the preliminary experiment

Parameter	Value
Crossover rate	0.9
Reproduction rate	0.0
Mutation rate	0.1
Max. no. of generations	41
Termination	Max. gens. reached
Elitism used?	yes
Fitness measure	MSE
Population size	25 000 total nodes

Table 2. Specific DyFor GP parameter settings for the preliminary experiment

Parameter	Value
Windowslide increment	1
Max windowsize	14
Min windowsize	2
Start windowsize	4
Window difference	6
Window adj. stepsize	1

applied to the DyFor GP model. For DyFor GP it means the maximum number of generations used for one dynamic generation, that is a set of generations run on a single analysis window. Since the DyFor GP model executes many dynamic generations over the course of a single run, this parameter is reduced from 51 to 41 generations to decrease computation time.

(2) Population size is specified as a limit for total nodes allowed in a population rather than as a limit for the total number of solutions allowed in a population. Since solutions are represented as tree structures of varying size/depth, it is possible that many solutions in a population have large tree structures. If this is the case, available memory may be exhausted and cause a run to abort. Thus, specifying a limit on the total number of nodes in a population ensures that this cannot occur.

(3) Elitism (reproduction of the best solution of the population) is used.

(4) Parameter values for 'reproduction rate' and 'mutation rate' were exchanged. This was done for two reasons: (1) increasing the mutation rate allows for greater search-space exploration (Michalewicz, 1992) and (2) decreasing the reproduction rate to zero was not thought to harm the effectiveness of the evolutionary process since elitism is specified.

In Table 2 parameter 'window slide increment' is the number of newer (more recent) historical data to incorporate at each slide of the analysis window. The 'max windowsize' and 'min win-dowsize' parameters specify the maximum and minimum analysis window-sizes, respectively. The adjustable windowsize feature of the DyFor GP model calls for using two analysis windows of differing sizes. Parameter 'start window-size' refers to the initial windowsize setting of the smaller of the two windows and parameter 'window difference' refers to the size difference between the larger and the smaller window. Parameter 'window adj. stepsize' gives the adjustment amount to use when adjusting the size of the windows.

Experimental results address each of the three DyFor GP dynamics listed at the beginning of this section and are discussed subsequently.

Result #1: The windowsize expanded when the underlying process was stable and contracted when the underlying process shifted. This was seen in both DyFor GP runs. For example, in one run several expansions occurred when the analysis window was focused on segment 1 (time series values 1–20) and when analysis reached time series value #22 the first of several contractions took place. These contractions continued until analysis reached time series value #30 and expansions began again. When the analysis window reached time series value #42, contractions started once again and continued until analysis reached time series value #47, after which only expansions were seen. These expansions/contractions correspond to the three different segments of the time series. Contractions start at time series value #22, only two values after a new underlying process has come into effect. When the analysis window reaches time series value #30, it is entirely contained inside this new segment (segment 2) and expansions begin again. These dynamics are repeated when the analysis window enters and then becomes entirely contained in segment 3.

Result #2: Adaptations (evolved subtrees) from a previously encountered underlying process are retained during analysis of a new, nonsimilar process by becoming part of intron subtrees. This was seen in both DyFor GP runs. For example, two adaptations relevant for segment 1, $\sin(x)$ and $\sqrt{x}$ were saved into two intron subtrees when analysis reached segment 2. When the analysis window reached time series value #29 (well inside segment 2), the best evolved forecasting model included two intron subtrees equivalent to the following two mathematical

Table 3. DyFor GP forecasting results with and without dormants

Forecasting model	RMSE (all segment 3)	RMSE (first 4 points)
DyFor GP (with dormants)	0.41	0.83
DyFor GP (w/o dormants)	0.45	0.90

expressions, $0/87.5\sin(x)$ and $0/\sqrt{x}+0.05x$. Here $\sin(x)$ of the first expression and $\sqrt{x}$ of the second are included as part of the denominator of a fraction whose numerator is 0. Both expressions evaluate to 0 regardless of their denominators and, thus, these adaptations which are relevant for segment 1 but not relevant for segment 2 do not affect the fitness of the solution.

Result #3: When the current underlying process is similar to a past process, the use of dormant solutions provides for more efficient capture of this process than is attained if dormants are not used. This is seen by comparing the forecasting performances of the two DyFor GP runs. Table 3 summarizes the forecasting results of both runs. In the table RMSE is the root mean squared error of forecasts.

Table 3 shows two RMSE values for each DyFor GP run. The first column is the RMSE of all forecasts that correspond to segment 3 and the second column is the RMSE of forecasts that correspond to the beginning of segment 3 only (the first 4 values) rather than the entire segment. Recall that segment 3's underlying process is similar to that of segment 1 and dissimilar to segment 2's process. Thus, if dormant solutions are effective, then the DyFor GP run that uses them should have better segment 3 forecasts than the DyFor GP run that does not use them. This is revealed in the table. The first column shows that the use of dormants provides for more accurate forecasts over the entire segment 3. The second column shows that when DyFor GP analysis enters segment 3 (i.e. the underlying process shifts from segment 2's process to that of segment 3), the use of dormants provides for quicker capture of this new process.

These results support the dynamics discussed in Section III and indicate the viability of the DyFor GP model. The following section details DyFor GP experiments of a larger scale.

V. Testing the DyFor GP: Full Experiments

Two time series were selected for larger experiments, the US gross domestic product (GDP) and the US consumer price index (CPI) Inflation rate. Figures 19 and 20 give graphical depictions of the quarterly GDP (growth) time series and the monthly CPI Inflation rate time series, respectively. In Fig. 19 real GDP growth is calculated as a quarter-over-quarter annualized percent change, while in Fig. 20 CPI inflation is calculated as a year-over-year percent change. These forecasting experiments were chosen because both the US GDP and CPI Inflation series are widely-studied, nonlinear time series with well-known sets of explanatory variables. Such characteristics are conducive to preparing a DyFor GP experiment and comparing DyFor GP results to those of leading studies.

Forecasting the US gross domestic product and CPI inflation

According to the US Department of Commerce (2004), the GDP is defined as 'the market value of goods and services produced by labor and property in the United States'. A leading model designed by Kitchen and Monaco (2003) forecasts the GDP, a time series with quarterly frequency, using multiple economic indicators that are measured monthly. The idea is to produce a single, one-step-ahead, quarterly GDP forecast by incorporating and analysing the latest monthly indicator values. For example, if the national unemployment rate and unemployment insurance claims are selected as (monthly) economic indicators and their latest announced values are for the month of January, then the forecasting model incorporates these latest values and aggregates them to produce a forecast for the current quarter (in this case quarter 1 or Q1). When indicator values for February are announced, the model incorporates these, re-analyses and re-aggregates them to produce an updated forecast for Q1. Thus, a 'real-time' GDP forecast for the current quarter can be constructed and updated as soon as new data become available.

The real-time forecasting system (RTFS) of Kitchen and Monaco (2003) makes use of 30 monthly economic indicators as explanatory variables. These economic indicators can be subdivided into the following categories:

(1) employment indicators (6),
(2) financial indicators (4),
(3) survey indicators (6),
(4) production and sales indicators (12) and
(5) other indicators (2).

A linear regression model is used to relate an indicator to GDP growth:

$$y_t = \alpha + \beta(L)x_t + e_t \tag{5}$$

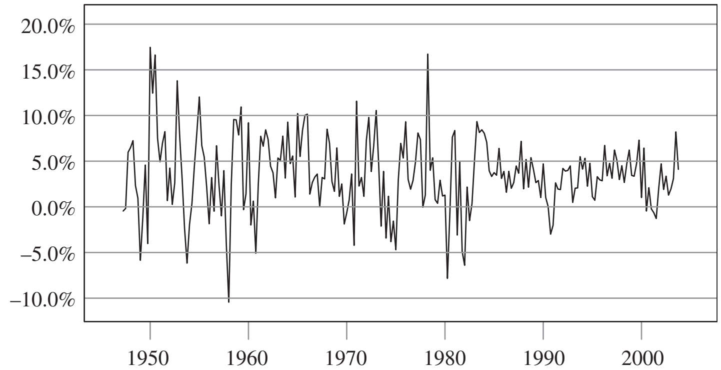

Fig. 19. Gross domestic product (growth): 1947–2003

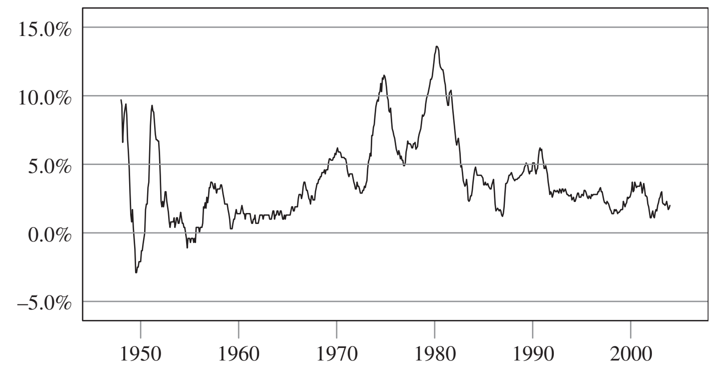

Fig. 20. Consumer price index Inflation: 1948–2003

where y_t is the real GDP growth for quarter t at an annualized rate, x_t is an indicator, $\beta(L)$ is a set of coefficients for current and lagged values of the indicator and e_t is an error term. While Equation 5 theoretically may include numerous indicator lags, Kitchen and Monaco choose zero or four lags and use the Schwarz criterion[3] to determine which results the RTFS should utilize. Each indicator has three separate regression models relating it to GDP growth, one for each (monthly) period of a quarter. When a new month's data for an indicator becomes available, the appropriate regression model is used to produce a forecast for GDP growth that is based only on that indicator. This is repeated for all indicators. Then, all of these single-indicator GDP forecasts are aggregated into one to yield a combined GDP forecast. RTFS generates 1-step-ahead forecasts in a 'real-time' fashion. All RTFS forecasts are made using an out-of-sample methodology where no data beyond the point of forecast is used for model fitting.

The RTFS is used to generate quarterly GDP forecasts when one month, two months and three months of indicator data are available, respectively. These results are compared to those produced by a linear autoregressive (AR) forecasting model with four lags. Historical data dating back to 1982Q1 is used and one-step-ahead GDP forecasts are generated for an 8-year range starting with 1995Q1 and ending with 2003Q1. The results of the Kitchen and Monaco study show that the RTFS model outperforms the AR model by a large margin.

The US CPI Inflation rate is a highly-scrutinized concern. The inflation time series has monthly frequency and available historical data exists from 1947. The Phillips Curve is a bivariate linear forecasting model that is widely considered as a consistent and accurate predictor of US inflation. Stock and Watson (1999) re-investigated the efficacy of this model, both in its conventional form and in alternate forms that include various macroeconomic variables. The conventional Phillips Curve specification used in their study is meant to forecast inflation over a 12-month period and is given by the following regression model:

$$\pi^h_{t+h} - \pi_t = \phi + \beta(L)u_t + \gamma(L)\Delta\pi_t + e_{t+h} \qquad (6)$$

[3] The Schwarz criterion is defined in [Diebold (1998), p. 26].

where $\pi_t^h = (1200/h) * \ln(P_t/P_{t-h})$ is the h-period inflation rate ($h = 12$), $\pi_t = (1200) * \ln(P_t/P_{t-1})$ is the monthly inflation rate, u_t is the unemployment rate and β(L) and γ(L) are lag operators specifying 0 to 11 lags. Alternate Phillips Curve specifications are constructed by substituting the unemployment rate, u_t, of Equation 6 with other macroeconomic variables or indices.

Historical CPI Inflation data dating back to January 1959 are used for analysis and 12-month horizon forecasts are generated for the period of January 1970 to September 1996. Forecasting results are presented for two sub-periods, 1970 to 1983 and 1984 to 1996. As in Kitchen and Monaco's GDP forecasting experiments, Stock and Watson use an out-of-sample methodology.

The Stock and Watson study shows that the Phillips Curve in its conventional form outperforms univariate autoregressive models as well as most alternative Phillips Curve specifications in which the unemployment rate is replaced by a different economic variable. The alternate specifications that do surpass the conventional one are those that replace unemployment with a measure of aggregate economic activity such as real manufacturing and trade sales or capacity utilization. Stock and Watson also develop a new composite index of 168 economic activity measures using principal component analysis and construct another alternative Phillips Curve specification with this index. This composite-index specification proves to be the best CPI Inflation forecasting model overall.

Test setup

The DyFor GP model was applied to the GDP and CPI Inflation forecasting experiments detailed earlier.

For the GDP experiment, economic indicators from all categories listed in the Kitchen and Monaco (2003) study are utilized as inputs to the DyFor GP model. In all, 29 of the 30 listed indicators are utilized.[4] Outputs are one-step-ahead, quarterly forecasts for the current quarter when only one month of historical data for that quarter is available. Historical GDP data dating back to 1951Q3 is used for analysis and forecasts for 1995Q1 through 2003Q1 are produced.

For the CPI Inflation forecasting experiment, the goal is to compare the performance of the conventional phillips curve (CPC) specification with that of the DyFor GP model. Therefore, inputs to the DyFor GP model are the same inputs employed by this conventional specification, namely the unemployment rate and past values of the monthly inflation rate. Historical CPI Inflation data dating back to 1950:01 is used for analysis and forecasts for 1970:01 through 1983:12 are produced.

Table 4. General GP parameter settings for the GDP and CPI Inflation experiments

Parameter	Value
Crossover rate	0.9
Reproduction rate	0.0
Mutation rate	0.1
Max. no. of generations	41
Termination	Max. gens. reached
Elitism used?	Yes
Fitness measure	MSE/MAD/CF
Population size	38 000 total nodes

Table 5. Specific DyFor GP parameter settings for the GDP and CPI inflation experiments

Parameter	Value (GDP experiment)	Value (Inflation experiment)
No. training dyn. gens.	121	121
Windowslide increment	1	1
Max windowsize	80	240
Min windowsize	40	12
Start windowsize	54	120
Window difference	12	24
Window adj. stepsize	1	1

In both experiments single run DyFor GP forecasts are generated in a 'real-time' fashion, that is, after the DyFor GP model produces the first forecast, the analysis window is slid to incorporate the actual data for that time period, analysis continues and then the DyFor GP produces the second forecast. This procedure is continued for each forecast until all required forecasts have been generated. The forecast combining method used for these experiments is the one described in section 3 and the number of DyFor GP runs comprising a set =0 20.

Tables 4 and 5 give general GP parameter values and specific DyFor GP parameter values, respectively used for both experiments.

All parameter values listed in Table 4 are the same as those used for the preliminary experiment (Table 1) with the following exceptions.

(1) The limit for total nodes in a population is increased to account for the greater complexity of the forecasting tasks.

[4] One of the indicators, 'Business Week Production Index', could not be obtained at the time of the experiments.

(2) Besides MSE, two additional fitness measures have been added, MAD and the CF measure described in section Forecast combination and fitness measures.

Each experiment includes three separate DyFor GP runsets, one using each fitness measure. The fitness measures are only used for evolution and are not used to measure the quality of forecasts produced by the DyFor GP model. The CF measure requires a user-specified parameter, Ω, to determine which data are outliers and which are not. For these experiments Ω is set to 7.5% of the median level of the time series to be forecast. An optimal value for Ω is not known and, thus, intuition was used to specify this parameter.

In Table 5 parameter 'no. training dyn. gens.' is the number of dynamic generations executed before producing the first forecast. Although the value for both of these parameters is the same for both the GDP and CPI Inflation experiments, the implications are different for each experiment because the frequency of the GDP time series is quarterly and the frequency of the CPI Inflation time series is monthly. For the GDP experiment, 121 training dynamic generations means the analysis window slides through 30 years of historical data (4 slides per year, 1 dynamic generation per slide +1 initial dynamic generation). For the CPI Inflation experiment, the same value means the analysis window slides through 10 years of historical data (12 slides per year, 1 dynamic generation per slide +1 initial dynamic generation).

Also in the table, the GDP experiment gives values of 80 and 40 for parameters 'max windowsize' and 'min windowsize,' respectively which correspond to max/min analysis windowsizes of 20 and 10 years, respectively and the CPI Inflation experiment gives values of 240 and 12 for these parameters which correspond to max/min windowsizes of 20 and 1 years, respectively.

Results

Tables 6 and 7 compare DyFor GP results to those of the benchmark models for the GDP and CPI Inflation experiments, respectively. In the tables, results of 3 DyFor GP models are shown, one for each of the 3 fitness measures. Figures 21 and 22 plot GDP growth and CPI Inflation forecasts produced by the best DyFor GP model with the corresponding actual values of GDP growth and CPI Inflation, respectively.

Tables 6 and 7 show that the DyFor GP model outperforms the benchmark models. In the CPI Inflation experiment the margin is small, but for the GDP experiment the margin proves large. Also, the fitness measure employed by the DyFor GP model appears to have an important influence on its performance.

Table 6. Gross domestic product forecasting results

Forecasting model	RMSE
RTFS	1.85
AR	2.46
DyFor GP (MSE fitness measure)	1.87
DyFor GP (CF fitness measure)	1.80
DyFor GP (MAD fitness measure)	1.57

Table 7. Consumer price index inflation forecasting results

Forecasting model	RMSE
CPC	2.4
DyFor GP (MSE fitness measure)	2.3
DyFor GP (CF fitness measure)	2.6
DyFor GP (MAD fitness measure)	2.6

The DyFor GP model's superior performance in both experiments may be due to its ability to capture nonlinearities present in the GDP and CPI Inflation time series that are not captured by the competing linear models. In the GDP experiment, historical data starting in 1951Q3 is analysed and forecasts for the 1995Q1 to 2003Q1 period are produced. In the CPI inflation experiment, historical data starting in 1950 is analysed and forecasts for the 1970:01 to 1983:12 period are produced. The behaviour of real GDP growth over its forecast horizon is reasonably stable compared to its preceding behaviour (mean and SD of 3.26 and 4.06 over the 1951Q3 to 1994Q4 period versus mean and SD of 2.95 and 2.13 over the 1995Q1 to 2003Q1 period). Thus, DyFor GP is able to capture and successfully extrapolate real GDP growth. By contrast, the behaviour of CPI inflation over its forecast horizon was drastically different from its preceding behaviour (mean and SD of 2.16 and 1.95 over the 1950:01 to 1969:12 period versus mean and SD of 7.13 and 2.97 over the 1970:01 to 1983:12 period). Thus, DyFor GP is less able to capture and successfully extrapolate CPI inflation. This could be the reason why the DyFor GP's margin of advantage over competitors is smaller for the inflation experiment as opposed to the GDP experiment.

Considering the three fitness measures utilized by the DyFor GP model, the performance ranking order of measures is in reverse order for the two experiments. The GDP experiment

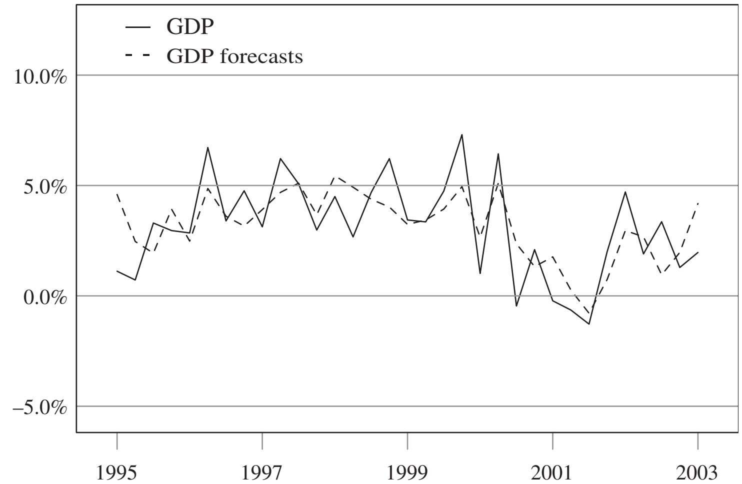

Fig. 21. Gross domestic product growth and forecasts produced by the DyFor GP model

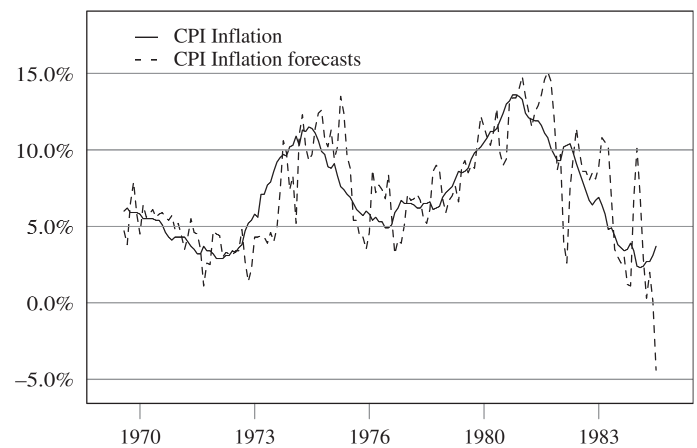

Fig. 22. Annual CPI Inflation and forecasts produced by the DyFor GP model 12 months earlier

gives a ranking order, first to last, of MAD, CF, MSE. The inflation experiment gives a ranking order of MSE, CF, MAD.[5] This may also be explained by the difference in forecast horizon stability between the GDP and inflation series. Analyses based on MSE are more heavily influenced by the existence of outliers than analyses based on MAD. An outlier datum can represent one of two possibilities: noise (which should be ignored or have reduced impact on model construction) or new information representing a shift in the underlying process. The GDP series is less volatile in the forecast horizon than the inflation series which may mean that outliers represent noise and should not dramatically affect model construction. Thus, for the GDP case, the MAD measure is the better measure. The inflation series is more volatile in the forecast horizon and outliers may frequently represent a shift in the underlying process. Therefore, the MSE measure is the most useful because it can more easily track the rapidly occuring shifts of the inflation series. Following this line of reasoning, the CF measure should have median utility for both experiments as it is a combination of MSE and MAD measures.

Other experimental results concerning retention of past adaptations and window behaviour proved interesting as well. Forecasting models evolved by the DyFor GP contained adaptations learned from

[5] The results reported by Stock and Watson carry a precision of one decimal place and, thus, DyFor GP results in Table 7 are reported with the same precision. This precision obscures the difference between results produced by DyFor GP models with the CF and MAD fitness measures, respectively.

the past. The following describes two examples of this retention.

(1) In the GDP experiment the adaptation, e^{prod} ($prod$ = industrial production index), was evolved as early as 1995 and is retained over the next 8 years, showing up in evolved models for 1997, 1998, 1999, 2001 and 2003.
(2) In the Inflation experiment the adaptation, $\sin(u_{t-11} - u_t)$ (u_t = current unemployment rate, u_{t-11} = unemployment rate 11 months ago), was evolved as early as 1976 and is retained over the next 7 years, showing up in evolved models for several years up to 1983.

Window adjustment also appeared to have an important effect. In the GDP experiment, the window-size was initially set at 16.5 years and the best performing runs generally adjusted their windowsize to approximately 14 years. In the Inflation experiment, the windowsize was initially set at 10 years and the best performing runs generally adjusted to approximately 12.5 years.

The DyFor GP model generates forecasts in a real-time fashion, that is, after the first forecast is produced, the analysis window is slid to incorporate the actual data for that time period, analysis continues and then the second forecast is produced. Thus, the forecasting model changes (evolves) over the course of a forecasting experiment. Usually this means that a new and unique forecasting model is constructed for each forecast. The forecasting models evolved by the DyFor GP often consist of several hundred operators and operands. These evolved models are too large to be displayed in this article.

VI. Conclusions and Future Work

In this study the DyFor GP model is developed and tested for forecasting efficacy on two important economic time series, the US Gross Domestic Product and Consumer Price Index Inflation. Results show that the DyFor GP model outperforms benchmark models from leading studies for both experiments. These findings affirm the DyFor GP's potential as an adaptive technique for real-world forecasting applications. The DyFor GP model presents an attractive forecasting alternative for the following reasons.

(1) It is not necessary to specify the functional form of the forecasting model in advance and, thus, a befitting nonlinear model, albeit complex, can be automatically discovered.
(2) The DyFor GP is automatically self-adjusting. Thus, in the presence of a changing environment, it may be able to adapt and predict accurately without human intervention.
(3) It can take advantage of a large amount of data. Conventional forecasting models require that the number of historical data to be analysed be set *a priori*. In many cases this means that a large number of historical data is considered to be too old to represent the current data generating process and is, thus, disregarded. This older data, however, may contain information that can be used to better capture the current process. The DyFor GP model is designed to analyse all historical data, save knowledge of past processes and exploit this learned knowledge to capture the current process.
(4) With greater computing power comes potentially better forecasting performance. The DyFor GP model is essentially a heuristic, fitness-driven random search. As with any random search, when a larger percentage of the search-space is covered, better results can be expected. Greater computational power allows for greater search-space coverage and DyFor GP forecasting performance can be improved by simply increasing such power.

Continued development and testing of the DyFor GP model is planned. One way of possibly improving the forecasting results of the GDP and CPI Inflation experiments would be to increase the computing power employed. The above results were achieved in a modest cluster computing environment (see North Carolina State University, 2004 for details). Concerning the CPI Inflation experiment, Stock and Watson (1999) report that using an alternative Phillips Curve specification that replaces the unemployment rate explanatory variable with a new composite index of 168 economic variables that they developed yields better forecasting performance than the conventional Phillips Curve specification. Thus, this composite index could be utilized by the DyFor GP model to potentially produce further performance advances.

The choice of fitness measure plays an important part in the forecasting performance of the DyFor GP model. In some cases a DyFor GP model with MSE measure proved the most effective while for other cases a DyFor GP model with MAD was best. A novel fitness measure, CF, which combines aspects of MSE and MAD was developed and tested. The CF measure relies on a parameter, Ω, to determine which data are considered outliers and which are not. Since an optimal specification for Ω was not known, intuition was used to set this parameter. Further studies might investigate different settings for this parameter and/or develop an algorithm to automatically adjust this parameter toward its optimal setting.

Perhaps the most intriguing direction for DyFor GP development is in the area of forecast combination. The method utilized in this study is a simple one that ranks each DyFor GP run based on the accuracy of its most recent past forecast, selects the top 3 runs, averages their current forecasts and designates this average forecast as the single, out-of-sample forecast to be used. A more sophisticated forecast combining method would likely result in performance improvements. Suppose the combination model of Equation 2 is considered. Using all past forecasts produced by a set of n DyFor GP runs as training data, a GA could be employed to evolve optimal weights for this model.

References

Akgiray, V. (1989) Conditional heteroskedasticity in time series and stock returns: evidence and forecasts, *Journal of Business*, **62**, 55–80.

Andreou, A., Georgopoulos, E. and Likothanassis, S. (2002) Exchange rates forecasting: a hybrid algorithm based on genetically optimized adaptive neural networks, *Computational Economics*, **20**, 191–202.

Andrew, M. and Prager, R. (1994) Genetic programming for the acquistion of double auction market strategies, *Advances in Genetic Programming*, **1**, 355–68.

Angeline, P. (1994) Genetic programming and emergent intelligence, *Advances in Genetic Programming*, **1**, 75–98.

Back, T. (1996) *Evolutionary algorithms in theory and practice: Evolution Strategies, Evolutionary Programming and Genetic Algorithms*, Oxford University Press, New York.

Back, B., Laitinen, T. and Sere, K. (1996) Neural networks and genetic algorithms for bankruptcy predictions, *Expert Systems with Applications*, **11**, 407–13.

Billio, M., Sartore, D. and Toffano, C. (2000) Combining forecasts: some results on exchange and interest rates, *European Journal of Finance*, **6**, 126–45.

Bollerslev, T. (1986) Generalized autoregressive conditional heteroskedasticity, *Journal of Econometrics*, **31**, 307–27.

Brameier, M. and Banzhaf, W. (2001) A comparison of linear genetic programming and neural networks in medical data mining, *IEEE Transactions on Evolutionary Computation*, **5**, 17–26.

Chambers, L. (Ed.) (1995) *Practical Handbook of Genetic Algorithms: Applications*. CRC Press, Oca Raton, FL.

Chen, S. and Yeh, C. (1996) Toward a computable approach to the efficient market hypothesis: an application of genetic programming, *Journal of Economics Dynamics and Control*, **21**, 1043–63.

Chen, S., Yeh, C. and Lee, W. (1998) Option pricing with genetic programming, in *Genetic Programming 1998: Proceedings of the Third Annual Conference*, Vol. 1, Madison, WI, pp. 32–7.

Chiraphadhanakul, S., Dangprasert, P. and Avatchanakorn, V. (1997) Genetic algorithms in forecasting commercial banks deposit, in *Proceedings of the IEEE International Conference on Intelligent Processing Systems*, Vol. 1, Beijing, pp. 557–65.

Deboeck, G. (Ed.) (1994). *Trading on the Edge: Neural, Genetic, and Fuzzy Systems for Chaotic and Financial Markets*. John Wiley and Sons, Inc, New York.

Dekker, M., van Donselaar, K. and Ouwehand, P. (2004) How to use aggregation and combined forecasting to improve seasonal demand forecasts, *International Journal of Production Economics*, **90**, 151–67.

Diebold, F. (1998) *Elements of Forecasting*, International Thomson Publishing, Cincinnati, OH.

Engle, R. (1982) Autoregressive conditional heteroskedasticity with estimates of the variance of UK inflation, *Econometrica*, **50**, 987–1008.

Fischer, I. and Harvey, N. (1999) What information do judges need to outperform the simple average?, *International Journal of Forecasting*, **15**, 227–46.

Fogel, L., Angeline, P. and Fogel, D. (1995) An evolutionary programming approach to self-adaptation on finite state machines, in *Proceedings of the 4th Annual Conference on Evolutionary Programming*, San Diego, CA, pp. 355–65.

Fogel, L., Owens, A. and Walsh, M. (1966) *Artificial Intelligence through Simulated Evolution*, Wiley, New York.

Fogel, D. and Chellapilla, K. (1998) Revisiting evolutionary programming, in *SPIE Aerosense 98, Applications and Science of Computational Intelligence*, Vol. 1, Orlando, FL, 2–11.

Fyfe, C., Marney, J. and Tarbert, H. (1999) Technical analysis versus market efficiency—a genetic programming approach, *Applied Financial Economics*, **9**, 183–91.

Gately, E. (1996) *Neural Networks for Financial Forecasting*, John Wiley and Sons, Inc., New York.

Goto, Y., Yukita, K., Mizuno, K. and Ichiyanagi, K. (1999) Daily peak load forecasting by structured representation on genetic algorithms for function fitting, *Transactions of the Institute of Electrical Engineers of Japan*, **119**, 735–6.

Gurney, K. (1997) *An Introduction to Neural Networks*, UCL Press, London.

Hendry, D. and Clements, M. (2004) Pooling of forecasts, *Econometrics Journal*, **7**, 1–31.

Hiden, H., McKay, B., Willis, M. and Tham, M. (1998) Nonlinear partial least squares using gentic programming, in *Genetic Programming* 1998: *Proceedings of the Third Annual Conference*, Vol. 1, Madison, WI, pp. 128–33.

Iba, H. and Sasaki, T. (1999) Using genetic programming to predict financial data, in *Proceedings of the Congress of Evolutionary Computation*, Vol. 1, Piscataway, NJ, 244–51.

Iba, H. and Nikolaev, N. (2000) Genetic programming polynomial models of financial data series, in *Proceedings of the 2000 Congress of Evolutionary Computation*, La, Jolla, CA, Vol. 1, pp. 1459–66.

Iba, H., de Garis, H. and Sato, T. (1994) Genetic programming using a minimum description length principle, *Advances in Genetic Programming*, **1**, 265–84.

Jonsson, P. and Barklund, J. (1996) Characterizing signal behaviour using genetic programming, *Evolutionary Computing: Lecture Notes in Computer Science*, **1143**, 62–72.

Ju, Y., Kim, C. and Shim, J. (1997) Genetic based fuzzy models: interest rate forecasting problem, *Computers and Industrial Engineering*, **33**, 561–4.

Kaboudan, M. (2003) Forecasting with computer-evolved model specifications: a genetic programming application, *Computer and Operations Research*, **30**, 1661–81.

Kaboudan, M. (2001) Genetically evolved models and normality of their residuals, *Journal of Economics Dynamics and Control*, **25**, 1719–49.

Kaboudan, M. (1998) Forecasting stock returns using genetic programming in C++, in *Proceedings of 11th Annual Florida Artificial Intelligence International Research Symposium*, Vol. 1, Venlo Park, CA, pp. 502–11.

Kaboudan, M. (2000) Genetic programming prediction of stock prices, *Computational Economics*, **6**, 207–36.

Kaboudan, M. (1999) Genetic evolution of regression models for business and economic forecasting, in *Proceedings of the Congress of Evolutionary Computation*, Vol. 2, Washington, DC, 1260–68.

Kim, D. and Kim, C. (1997) Forecasting time series with genetic fuzzy predictor ensemble, *IEEE Transactions on Fuzzy Systems*, **5**, 523–35.

Kitchen, J. and Monaco, R. (2003) Real-time forecasting in practice, *Business Economics: the Journal of the National Association of Business Economists*, **38**, 10–19.

Koza, J. (1992) *Genetic Programming: On the Programming of Computers by Means of Natural Selection*, MIT Press, Cambridge, MA.

Lee, D., Lee, B. and Chang, S. (1997) Genetic programming model for long-term forecasting of electric power demand, *Electric Power Systems Research*, **40**, 17–22.

Leigh, W., Purvis, R. and Ragusa, J. (2002) Forecasting the NYSE composite index with technical analysis, pattern recognizer, neural network and genetic algorithm: a case study in romantic decision support, *Decision Support Systems*, **32**, 361–77.

Levenick, J. (1999) Swappers: introns promote flexibility, diversity and invention, in *Proceedings of the Genetic and Evolutionary Computation Conference GECCO '99*, Vol. 1, Orlando, FL, 361–8.

Makridakis, S., Wheelwright, S. and Hyndman, R. (1998) *Forecasting: Methods and Applications*, John Wiley and Sons, Inc., New York.

Maniezzo, V. (1994) Genetic evolution of the topology and weight distribution of neural networks, *IEEE Transactions on Neural Networks*, **5**, 39–53.

McMillan, D. G. (2001) Nonlinear predictability of stock market returns: evidence from nonparametric and threshold models, *International Review of Economics and Finance*, **10**, 353–68.

Michalewicz, Z. (1992) *Genetic Algorithms + Data Structures = Evolution Programs*, Springer-Verlag, New York.

Mitchell, M. (1996) *An Introduction to Genetic Algorithms*, MIT Press, Cambridge, MA.

Mulloy B., Riolo R. and Savit, R. (1996) Dynamics of genetic programming and chaotic time series prediction, in *Genetic Programming* 1996: *Proceedings of the First Annual Conference*, Vol. 1, Standford, CA, pp. 166–74.

Nag, A. and Mitra, A. (2002) Forecasting daily foreign exhange rates using genetically optimized neural networks, *Journal of Forecasting*, **21**, 501–11.

North Carolina State University (2004) High Performance and Grid Computing. Available at http://www.ncsu.edu/itd/hpc/main.php

Neely, C. (2003) Risk-adjusted, ex ante, optimal technical trading rules in equity markets, *International Review of Economics and Finance*, **12**, 69–87.

Neely, C. and Weller, P. (2002) Predicting exchange rate volatility: genetic programming versus GARCH and RiskMetrics^{TM}, *The Federal Reserve Bank of St. Louis*, Missouri.

Neely, C., Weller, P. and Dittmar, R. (1997) Is technical analysis in the foreign exchange market profitable? A genetic programming approach, *Journal of Financial and Quantitative Analysis*, **32**, 405–26.

Nordin, P. and Banzhaf, W. (1995) Complexity compression and evolution, in *Proceedings of the Sixth International Conference on Genetic Algorithms*, Vol. 1, Pittsburg, PA, 310–7.

Phua, P., Ming, D. and Lin, W. (2001) Neural network with genetically evolved algorithms for stocks prediction, *Asia-Pacific Journal of Operational Research*, **18**, 103–7.

Sarantis, N. (2001) Nonlinearities, cyclical behaviour and predictability in stock markets: international evidence, *International Journal of Forecasting*, **17**, 459–82.

Sathyanarayan, R., Birru, S. and Chellapilla, K. (1999) Evolving nonlinear time series models using evolutionary programming, in *CECCO 99: Proceedings of the 1999 Congress on Evolutionary Computation*, Vol. 1, Orlando, FL, pp. 243–53.

Sexton, R. (1998) Identifying irrelevant input variables in chaotic time series problems: using genetic algorithms for training neural networks, *Journal of Computational Intelligence in Finance*, **6**, 34–41.

Smith, K. and Gupta, J. (2002) *Neural Networks in Business: Techniques and Applications*, Idea Group Pub., Hershey, PA.

Stock, J. and Watson, M. (1999) Forecasting inflation, *Journal of Monetary Economics*, **44**, 293–335.

Swanson, N. and White, H. (1995) A model selection approach to assessing the information in the term structure using linear models and artificial neural networks, *Journal of Business and Economics Statistics*, **13**, 265–75.

Trippi, R. and Turban, E. (Eds.) (1996) *Neural Networks in Finance and Investing: Using Artificial Intelligence to Improve Real-World Performance*, Irwin Professional Pub, New York.

US Department of Commerce Bureau of Economic Analysis (2004) Available at http://www.bea.doc.gov/bea/glossary/glossaryg.htm

Venkatesan, R. and Kumar, V. (2002) A genetic algorithms approach to growth phase forecasting of wireless subscribers, *International Journal of Forecasting*, **18**, 625–46.

Wagner, N. and Michalewicz, Z. (2001) Genetic programming with efficient population control for financial times series prediction, in *2001 Genetic and Evolutionary Computation Conference Late Breaking Papers*, Vol. 1, Sanfrancisco, CA, pp. 458–62.

Wang, J. (2000) Trading and hedging in S&P 500 spot and futures markets using genetic programming, *The Journal of Futures Markets*, **20**, 911–42.

White, H. (1992) *Artificial Neural Networks: Approximation and Learning Theory*, Blackwell, Cambridge, MA.

White, J. (1998) A genetic adaptive neural network approach to pricing option: a simulation analysis, *Journal of Computational Intelligence in Finance*, **6**, 13–23.

Yang, Y. (2004) Combining forecasting procedures: some theoretical results, *Econometric Theory*, **20**, 176–222.