Genetic Programming as a Means of Assessing and Reflecting Chaos

E Howard N Oakley

EHN & DIJ Oakley
Wroxall, Ventnor, Isle of Wight
PO38 3EQ, UK.

Introduction

The central theme of experimental science can be viewed as discovering the functions which best describe a set of observations. When these functions are simple, either because of the real simplicity of the system being studied, or because of ingenious experimental design, traditional tools usually suffice. However, as the systems being studied are known to be more complex (for example, with the benefit of more accurate measurement techniques), so progressively more time and effort has to be devoted to the analysis rather than the winning of the data. As a consequence, the traditional fields of statistics and data analysis have developed in new ways, reflected by a new vocabulary. Laboratories echo less to terms such as 'linear regression' and 'analysis of variance', and more to the new genres of 'data visualisation' and 'data mining'.

Among the systems which are most resistant to analysis are those which are manifestly non-linear and possibly complex and dynamic (chaotic), which can only be measured in short series or are non-stationary, and which are believed to contain significant amounts of noise. In biological sciences, these are the norm. Consequently, there are many assertions that biological systems are chaotic in nature (Glass & Kaplan 1994), but remarkably little conventional evidence (Ruelle 1990).

Koza's (1992, 1994) two tracts which define and explore the technique of genetic programming (GP) include glimpses of the value of this new approach in the fitting of functions to various datasets, including one which is known to be chaotic. In two previous studies (Oakley 1994a, 1994b), I have shown how GP can be used to predict future values for a known chaotic data series, even when it is contaminated with substantial amounts of noise. The information yielded from such predictive studies can be used to gain structural information about the underlying system, and to a limited degree, can confirm its chaotic characteristics.

The purpose of this study was to extend the application of GP to real experimental data, and to see whether it could be coupled with the examination of surrogate data, an increasingly popular technique in the study of chaotic systems (Theiler et al. 1992), and thus to test for chaos in even brief, noisy datasets.

Surrogate data series are synthetic datasets which are assembled by reordering original series, or from random values, in such a way that they share some properties with the original series, but that other properties are quite different. Typically, a surrogate series will be generated from an original possibly chaotic series by reordering the individual values so that the Fourier power spectrum and statistical properties are preserved, but the chaotic relationship between individual values is broken. Comparing the results of tests, such as fitting by GP, on the original and surrogate data series can then yield invaluable information as to whether there was underlying chaos which was destroyed during the creation of the surrogate series.

Methods

Original data for this study were derived from two sources: first from the Mackey-Glass flow (Mackey & Glass 1977), both as a flow approximation via Runge-Kutta techniques and discretised to a map (Oakley 19994a); second from physiological experiments involving the measurement of skin blood flow using a laser Doppler rheometer. Surrogates generated

Data series	noise	number	mean	variance
Mackey-Glass flow	no	516	0.8767	0.0839
– surrogate	no	512	0.8838	0.0585
Mackey-Glass map	no	various	approx 0.8	approx 0.08
Blood flow 1	yes	513	163.7	4884.6
– surrogate	yes	512	166.1	3324.5
Blood flow 2	yes	102	49.5	488.1
Blood flow 3	yes	76	222.3	2723.0

Table 1: Characteristics of the data series used in this study.

according to the first method of Theiler et al. (1992), which possess the same Fourier power spectrum and statistics, were used of each series as well. The main characteristics of the different series are given in Table 1.

For each data series, GP was performed using Koza's Simple Lisp code (Koza 1992) under Macintosh Common Lisp 2.0.1 running on an Apple Macintosh IIci with a Radius Rocket 68040 accelerator. Settings were those used in previous studies in this series (Oakley 1994b), in brief using a ramped half-and-half generation method to create initial populations of size 100 to 1000. Following this, a fitness proportionate selection method was used with fitness-proportionate reproduction fraction 0.1, crossover at any point fraction 0.2, and crossover at function points fraction 0.7. The maximum depth of new individuals was 6, maximum depth of new subtrees for mutants 4, and maximum depth of individuals after crossover 17.

The terminal set was the effectively time-embedded data series, in most cases using values at time steps 1, 2, 3, 4, 5, 6, 11, 16, 21, and 31 prior to the value being predicted, together with random real numeric constants. The aim of S-expressions in the population of each run was to predict future values of the series, up to 60 steps into the future. Whenever the length of a data series permitted, this was performed in multiple non-overlapping windows. The function set consisted of the four basic arithmetic operators (with divide protected from zero divide errors), together with error-protected sine, cosine and exponentiation to the power of 10 (a necessary component for sufficiency for series involving the Mackey-Glass map).

Raw and standardised fitness were taken as the average over the fitness cases of the squared error between predicted and actual points. However, for the purposes of comparisons, the normalised mean squared error (Gershenfeld & Weigend 1994) was derived from this, as the standardised fitness divided by the sample variance of the actual points in the prediction window.

Runs were performed over several days, including a range of different population sizes from 100 to 1000, over 51 and 101 generations without termination criteria. The fittest S-expression in each individual run was recorded, together with details of it fitness and generation.

Results

Prediction error, as gauged by the normalised mean squared error, for the Mackey-Glass flow was similar to that reported earlier (Oakley 1994a) for its discretised map, and increased with increasing duration of prediction. However, whilst errors for the flow and its surrogate were similar for brief predictions (30 steps into the future), they rose disproportionately in the case of the surrogate series, to reach three times the error at 30 when

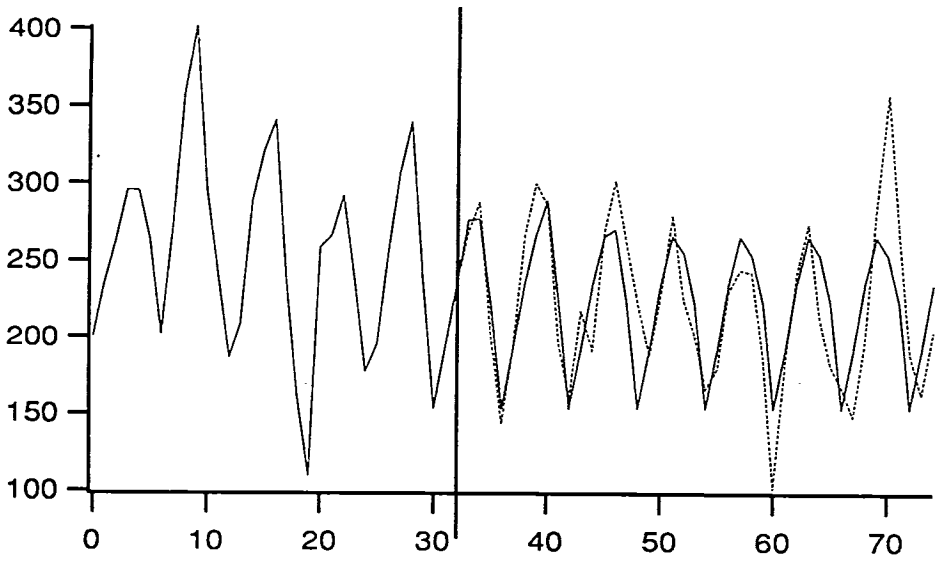

Figure 1 An experimental data series (dotted) with the forecast series (solid) superimposed. The vertical line marks the point at which predicted values begin for the latter.

prediction length had only doubled to 60. Errors from the experimental data series were of similar magnitude as those from the Mackey-Glass flow. Figure 1 shows one of the better examples of a fit S-expression predicting blood flow data.

In both the Mackey-Glass flow and experimental data series, increasing the number of individuals used in each GP run (by increasing the population size and/or the number of generations) generally led to reduced prediction errors. Typically, in the case of the Mackey-Glass flow, an increase in the number of individuals used from 5100 to 50500 (approximately tenfold) resulted in a halving of the prediction error; the magnitude of this effect in experimental data series was reduced.

Study of the complexity of the fittest S-expressions, estimated using the simple metric of counting left parentheses, and the number of individuals required to arrive at the fittest demonstrated the expected interrelationships. Fitter S-expressions tended to be more complicated and to require more individuals (i.e. higher numbers of generations as well as greater population sizes) for them to be generated. Two metrics which give a good feel for this effect are the change in the number of S-expressions evaluated before the fittest emerged in that run, and for a given group of runs (with the same population size and number of generations) the average number of S-expressions evaluated for the fittest to emerge in that group of runs. Both of these increased with increasing number of S-expressions evaluated, and with reducing prediction error.

The behaviour of surrogate data series was quite distinct. In the case of the Mackey-Glass flow surrogate, a few very simple S-expressions recurred time after time and dominated runs. They were typically found in the initial population or after relatively few generations, so that complex S-expressions rarely appeared among the fittest (indeed, they failed to evolve within populations). There was thus no reduction in prediction error with increasing number of S-expressions evaluated, and the average S-expression complexity and number of generations required to find the fittest S-expression remained fixed and low as population sizes and numbers of generations increased. Indeed, the average generation of the fittest S-expression actually reduced as the population size increased.

Discussion

The efficacy of genetic programming as a means of predicting data from short, noisy, possibly chaotic series has been demonstrated again. This has important consequences for those searching for a technique which can be used to examine a wide range of experimental data: GP is a prime candidate which has a number of significant advantages over other techniques.

First, GP as a technique does not need to make any underlying assumptions about the nature of the data with which it is presented. Particularly when used in its original Common Lisp implementation, it is remarkably easy to provide for almost any data input and operator type in GP. This contrasts with other popular alternatives such as neural networks and other evolutionary computing approaches, in which structural considerations dominate the outcome.

Second, GP requires a minimal programming effort, although this does in consequence make it more computationally intensive. In scientific computing, this spares the most expensive, human resource. Third, as has been remarked before (Oakley 1994b), GP avoids the trap of overfitting data in the face of noise. It also yields structural information, and is an effective means of system identification (Iba et al. 1993, Oakley 1994a).

Coupling GP with the use of surrogate data series opens up a rich range of possibilities beyond these. Studying the progress of multiple GP runs on actual and surrogate datasets appears to discriminate between the two. This suggests that the rich ever-improving populations of S-expressions found in some real data reflect non-linearity, even chaos, within those data. Here, the fitness landscape is varied and continuing evolutionary search yields higher peaks of fitness.

In contrast, the surrogate datasets appear dominated by the shallow and featureless landscapes typical of stochastic processes. A few prominent peaks of fitness are found quickly, and further evolutionary search fails to improve on them.

It is interesting to speculate that populations of S-expressions in genetic programming come to reflect the nature of the fitness landscape, and thus the complexity of the data which they are trying to fit. This is best seen in the simple and artificial case here of the Mackey-Glass flow. Using real chaotic data, the S-expressions in the evolving populations tended to become complicated and rich in their composition, in response to their varied fitness landscape. In contrast, those of the surrogate series generated from the Mackey-Glass flow quickly came to reflect their less varied landscape.

Conclusion

This series of studies (Oakley 1994a, Oakley 1994b) has shown how genetic programming can be used to perform a number of vital functions in the analysis of experimental data. These include system identification, prediction, and the distinction between chaotic and stochastic systems. It only remains to package the technique into a form which makes it accessible to the practitioner.

References

Gershenfeld N A Weigend A S **1994** The future of time series: learning and understanding *Time Series Prediction: Forecasting the Future and Understanding the Past* ed A S Weigend N A Gershenfeld (Reading, Mass: Addison-Wesley) 1–70

Glass L Kaplan D **1994** Complex dynamics in physiology and medicine *Time Series Prediction: Forecasting the Future and Understanding the Past* ed A S Weigend N A Gershenfeld (Reading, Mass: Addison-Wesley) 513–528

Iba H Kurita T de Garis H Sato T **1993** System identification using structured genetic algorithms *Proceedings of the Fifth International Conference on Genetic Algorithms* ed S Forrest (San Mateo, Cal: Morgan Kaufmann)

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

Koza J R **1994** *Genetic Programming II: Automatic Discovery of Reusable Programs* (Cambridge, Mass: MIT Press)

Mackey M C Glass L **1977** Oscillation and chaos in physiological control systems *Science* **197** 287–289

Oakley E H N **1994a** Two scientific applications of genetic programming: stack filter and non-linear equation fitting to chaotic data *Advances in Genetic Programming* ed K E Kinnear (Cambridge, Mass: MIT Press) 369–389

Oakley E H N **1994b** The application of genetic programming to the investigation of short, noisy, chaotic data series *Lecture Notes in Computer Science: Evolutionary Computing* ed T C Fogarty (New York, NY: Springer Verlag)

Ruelle D **1990** Deterministic chaos: the science and the fiction *Proc. Roy. Soc. Lond. A* **427** 241–248

Theiler J Galdrikian B Longtin A Eubank S Farmer J D **1992** Using surrogate data to detect nonlinearity in time series *Nonlinear Modelling and Forecasting* ed M Casdagli S Eubank (Redwood City, Cal: Addison-Wesley) 163–188