INVITED

Extruder Modelling Using Genetic Programming

JUSTIN ELSEY[+], BEN McKAY[+], MARK J. WILLIS[*] and GEOFFREY W. BARTON[+]

[+] Dept. of Chemical Engineering
University of Sydney
NSW 2006, Australia.

[*] Dept. of Chemical & Process Engineering
University of Newcastle
Newcastle upon Tyne, UK.

Abstract Complex processes are often modelled using input-output data from experimental tests. Regression and neural network modelling techniques address this problem to some extent and are proving useful in optimisation or model-based control applications. Unfortunately, the latter methods provide no physical insight into the underlying structural relationships inherent within the data. Genetic Programming (GP) is an alternative data-based modelling technique that appears to potentially overcome these limitations. The nature of GP-based modelling is that solutions are 'evolved' from a set of potential solutions by a process which mimics Darwinian 'survival of the fittest'. The advantage of GP modelling over neural networks and statistical techniques is that this method can also discriminate between relevant and irrelevant process inputs, evolving to yield parsimonious model structures that accurately represent process characteristics. This eliminates the need for restrictive a priori modelling assumptions. Moreover, as the methodology determines complex input-output interactions as well as inherent non-linearities, non-intuitive process features are also revealed with comparative ease. In this contribution, the application of GP techniques to steady-state and dynamic extruder modelling is presented. Extruders are currently used in a broad range of process industries. Cooking extruders are used widely in the food industry, as they provide an efficient means for the continuous processing of a wide range of foodstuffs. Plasticating extruders are used to extrude polymers for a range of applications, such as the coating of electronic wire or cables. Available extruder models range from the purely empirical to those based on fundamental transport equations. In most cases these models have limited validity in terms of operating conditions and system configuration, due to the complex flow patterns, pressure and temperature gradients and thermal effects that exist within an extruder. In this paper, a steady-state model of a cooking extruder is developed using input-output data generated from a rigorous mechanistic model. GP is then applied to dynamic data from an industrial plasticating extruder to generate a dynamic model using a time history of input values. It is concluded that GP techniques may have further applications in the modelling of complex processes from experimental input-output data.

INTRODUCTION

With the advent of cheap computers and mass data storage devices, information has become a major commodity. Unfortunately the capacity to store and retrieve data is not always a blessing: large amounts of data can be difficult to handle and interpret. In engineering, a crucial step is the transformation of raw data into a format useful for solving a particular problem. To do this, most current approaches tend to be based upon the development of a 'model' of the process in question. While an attempt can be made to develop a model from basic scientific principles, there are a number of drawbacks to using this approach. Chemical processes are often extremely complex. Thus, it may take a considerable period of time and effort to develop a realistic mechanistic model of the system. Moreover, in many instances simplifying assumptions have to be made in order to provide a tractable solution. Therefore, a first-principles model will often be costly to construct and may be subject to built in limitations on its accuracy. Consequently, data-based modelling techniques are often preferred because of their cost effectiveness.

Extrusion processes are generally difficult to model. Sensitivity to varying screw and die geometries, fluid properties, and complex flow patterns combine to ensure that mechanistic models are difficult to develop. Data-based modelling in this instance can provide a practical alternative approach.

Currently, the majority of the popular data-based modelling methodologies can be categorised under two headings. The first utilise artificial neural networks while the second are based upon statistical and regression analyses. This contribution describes a novel approach which offers a useful alternative to these data-based modelling methodologies. The technique uses Genetic Programming (GP) (eg. Koza, [1992]; Kinnear, [1994]) to develop non-linear models of chemical process systems from plant input-output data. Unlike conventional modelling procedures that seek the best set of parameters for a pre-specified model, the GP methodology performs symbolic regression to determine the structure and complexity of the required model simultaneously with model parameter estimation.

GENETIC PROGRAMMING

Genetic Programming began as an attempt to discover how computers could learn to solve problems without being

explicitly programmed to do so. The GP technique bears a strong resemblance to Genetic Algorithms (eg. see Goldberg, [1989]) but uses a more flexible coding system which facilitates direct execution of 'genetic material' (Kinnear, [1994]).

To date, GP has been used in the development of image classification systems, the design of filters, fitting of chaotic non-linear equations, feature detection and the evolution of neural networks (see Kinnear, [1994]). Here, we use GP to evolve mathematical models of extrusion processes.

A flow diagram of a typical GP algorithm is shown in Figure 1. Initially, a population of 'N' random mathematical expressions must be generated. Each expression is coded as a tree structure. For a thorough discussion about trees and their properties, see Tenenbaum and Augenstein [1981]. However, it is sufficient for the purposes of this paper to introduce relevant terminology by means of a simple example. Consider the problem of predicting the numeric value of an output variable, y, from two input variables a and b. An example of a symbolic representation for y in terms of a and b is:

$$y = (a - b) / 3 \qquad (1)$$

Figure (2) demonstrates how this expression may be represented using a tree structure. How tree structures 'evolve' via the application of genetic operators will be discussed later.

Assignment of Fitness Values

Once the population has been filled with N distinct, valid trees, it is necessary to evaluate each individual to determine its fitness. Fitness is a numeric value assigned to each member of the population to provide a measure of the appropriateness of the solution. In the literature, the majority of fitness functions utilised are based on the error between the actual and predicted solutions. However, South [1994] found that the correlation coefficient is a more useful fitness function for numeric prediction problems, therefore this performance index was used throughout this work. The absolute value of the correlation coefficient is given by:

$$r(i) = \left| \frac{\left(\sum \frac{y_P(i,j)\, y_T(j)}{R} \right) - \left(\sum \frac{y_P(i,j)}{R} \right) \left(\sum \frac{y_T(j)}{R} \right)}{\sigma_P(i)\, \sigma_T} \right| \qquad (2)$$

where $y_P(i,j)$ is the predicted value of record 'j' by tree 'i', $y_T(j)$ is the target value, R is the number of data records, $\sigma_P(i)$ is the standard deviation of the predicted values for tree 'i' and σ_T is the standard deviation of the target values.

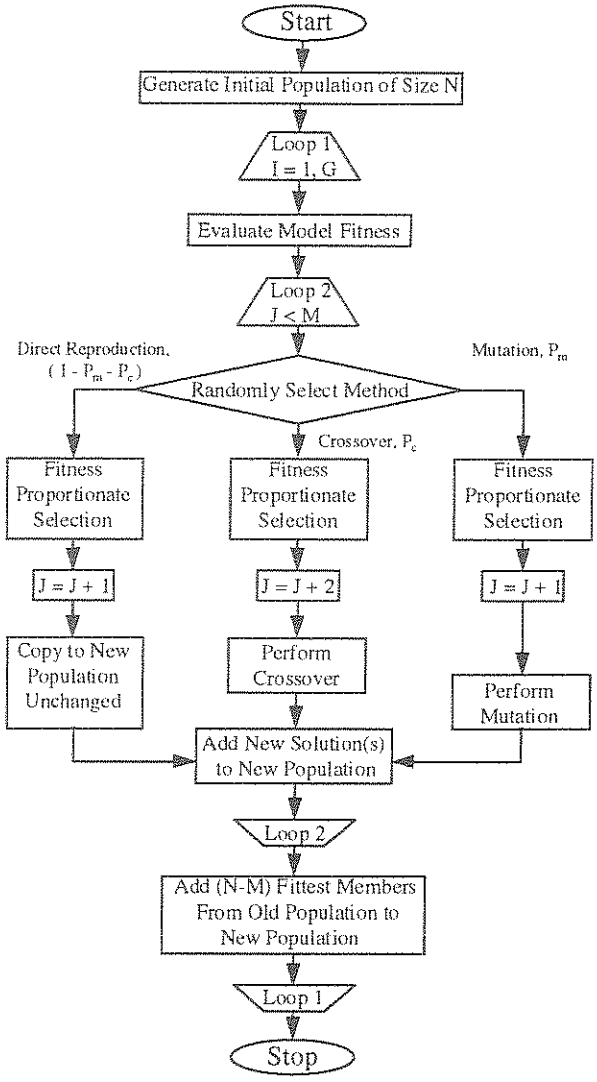

Figure 1: Typical genetic programming algorithm flowsheet.

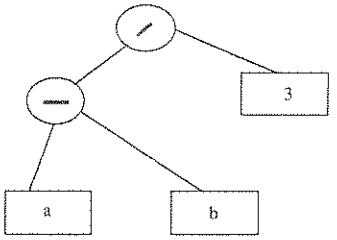

Figure 2: Representation of a numerical expression using a tree structure.

The correlation coefficient lies in the range [0 1] and measures the way in which the predicted and target values vary together. Therefore, credit is given to mathematical functions of the correct form.

Once the fitness of each individual has been determined, this information is used as the basis for selecting members of the population for 'reproduction'.

Selection Strategies

A number of selection methods have been suggested in the literature. These include the elitist strategy, tournament selection and fitness proportionate selection. With the elitist scheme, the population is sorted into descending order according to individual fitness values. The fittest M ($M \leq N$) individuals then undergo reproduction. Tournament selection involves random sampling (with replacement) of a fixed number of individuals from the parent population to form a subset. The fittest member of this small subset is then chosen to reproduce, and the process is repeated as required. With fitness proportionate selection, an individual is sampled from the parent population (with replacement), with a probability proportional to its fitness. Thus, if the i^{th} tree structure in the parent population has a fitness f_i , the probability of this member being chosen is f_i/ F , where F is the sum of the fitness of each member of the population, ie.

$$F = \sum f_i, \quad i=1,.....N. \tag{3}$$

In this paper, fitness proportionate selection was used as it appears to be the currently favoured technique within the GP literature.

Application of Genetic Operators

Once an individual has been selected from the current population, three basic genetic operators (direct reproduction, crossover and mutation) may be applied. The direct reproduction operator directly copies a member from the parent population to the next generation. The genetic operation of crossover takes two members and combines them to create new offspring. The operation of mutation allows new individuals to be created by making random changes to individuals about to enter the new generation, thus keeping the population varied. The choice of each operator is probabilistic, with crossover, mutation or direct reproduction being applied with probabilities P_c, P_m and $(1 - P_c - P_m)$, respectively.

Direct reproduction requires no further explanation, however a more detailed discussion of the other genetic operators (crossover and mutation) is presented below.

Crossover: The role of the crossover operator is to exploit good material by recombining it in different ways. This allows the construction of new individuals from existing ones, enabling new parts of the solution space to be searched.

During crossover, two mathematical expressions (ie. members of the existing population) are chosen using fitness proportionate selection. A randomly selected subtree from parent 1 is then interchanged with a subtree randomly chosen from parent 2. The newly created trees are then inserted into the mating pool that will be used to form the next generation.

The operation of crossover must preserve the syntax of the mathematical expressions. In other words, application of the genetic operator must produce an equation that can be evaluated. A valid crossover operation is shown below. Two parent expressions are given by:

$$\text{Parent 1:} \quad y = (a - b) / 3 \tag{4}$$
$$\text{Parent 2:} \quad y = (c - b) * (a + c) \tag{5}$$

Parent 1 has input variables 'a' and 'b' and a constant '3' while parent 2 has three input variables 'a', 'b' and 'c'. Both expressions attempt to predict the process output, 'y'. The crossover operation can produce two new trees which are given by:

$$y = (a - b) / (a + c) \tag{6}$$
$$y = (c - b)* 3 \tag{7}$$

It will be noted that crossover produces two children that may be quite different from the two parents. Nevertheless, the two offspring are created entirely from genetic material possessed by the parents. Intuitively, if a parent represents a reasonable solution to a numeric prediction problem, then it would be expected that it contains subtrees with information relevant to the solution. By recombining effective subtrees, new expressions may be evolved that provide an even fitter solution.

Mutation: The role of the mutation operator is to facilitate exploration of different parts of the solution space. Mutation creates an altered individual which is copied to the next generation of the population. Therefore, mutation is used to maintain population diversity, hence reducing the risk of premature (poor) convergence. Mutation consists of randomly changing a mathematical operator, an input or a constant in one of the members of the existing population.

Thus, during evolution the direct reproduction, crossover and mutation operators are applied to members of the population. Fit individuals have a higher probability of passing elements of their structure on to the next generation. This allows for the progressive evolution towards an acceptable solution. In practice, evolution can be surprisingly rapid.

Creating the Next Generation

Once a population of new individuals has been generated, a decision must be made as to which members of the old population should "die" to make room for the next generation. Several methods may be adopted. The simplest, given $M = N$ new individuals, is to replace the entire old population so that there is a complete turnover for every cycle of the algorithm. However, it may be expedient to

retain the fittest members of the old population, in order to ensure the survival of structures that perform well. Thus, a proportion P_{old} of the original generation is preserved, leaving $P_{gap} = (1 - P_{old})$ as a 'generation gap' (the proportion to be replaced with new individuals).

Implementation Aspects

The Terminal Set: The first step when implementing GP is the definition of the terminal set. This is the set of input variables that are thought to affect the output. In addition, the algorithm should have the ability to generate constants. It will generally be a combination of input values and numeric constants that produce the required regression model.

The Functional Set: When initialising the algorithm it is also necessary to identify the functional set that will be used to generate a model. When developing a mathematical model, typical functionals that may be supplied include, +, -, /, *, sqrt, exp, log. However, it is important to note that the functional set when combined with the set of terminals should have the property of closure. In other words, each function must be able to accept and return a numeric value when presented with an input. It may be noted, for example, that the function 'sqrt' does not have this property if presented with negative numbers. In order to 'protect' the sqrt function, the absolute value of the input should be taken before presentation to the function.

Optimisation of Individual Expressions: Whenever a new expression is generated, a non-linear least-squares optimisation is performed to obtain the 'best' value of the constant(s) in the expression. This allows the optimal values to be obtained for each suggested solution, thus reducing the possibility of eliminating a potentially good structure due to poor parameter values.

The benefits of constant optimisation are demonstrated by the following example. Suppose the objective is to fit a model to the data presented in Table 1, where 'a' and 'b' are inputs and 'y' is the output.

a	b	y
0.91	1.71	5.00
0.76	1.16	1.67
0.26	0.26	1.00
0.05	-0.35	0.71
0.74	-0.06	0.56

Table 1: Input-Output Data

Two candidate models may be,

$$y = 5.6*(a-b)^2 \quad (8)$$

$$y = 1/(a - 0.16) \quad (9)$$

which have correlation fitness values of 0.5 and 0.1, respectively. If the crossover operator was applied to these two expressions, one possible offspring would be,

$$y = 1/((a-b)-0.16) \quad (10)$$

which has a correlation fitness value of only 0.2. However, if the value of the constant in this new expression is optimised, the following model is obtained,

$$y = 1/((a-b)-1.0) \quad (11)$$

which yields a fitness value of 1.0. Thus, without performing an optimisation of the model constants, it is very difficult to successfully transfer 'good' building blocks from one mathematical expression to another. This means that models with an inherently good structure could be discarded due to sub-optimal constants.

Penalise Large Solutions: It is well recognised that as the complexity of a model increases its ability to generalise (that is, to fit unseen data as well as the training data) is compromised. Consequently, large trees can cause overfitting of the data. In order to prevent the formation of large trees, the correlation fitness function was penalised according to the size of a tree.

The default settings used in this work for each run are summarised in Table 2.

Functional set:	+, -, /, *, ^, sqrt, square, log, exp
Relative abundance of operators:	equal
Population size:	N = 25
Number of generations:	G = 60
Mutation probability:	$P_m = 0.2$
Crossover probability:	$P_c = 0.8$
Fitness function:	Correlation
Generation gap:	$P_{gap} = 90\%$

Table 2: Default Parameters used by the Genetic Program.

The settings were chosen in a heuristic fashion. There is, no doubt, for a given problem an 'optimum' combination that maximises the performance of the algorithm. However, intensive trials would have to be conducted before such an optimum parameter set could be defined.

It will be noted that each case study involved multiple runs. This allowed a statistical analysis of the results to be performed. In particular it enabled:

- The frequency of runs that produced successful results to be determined. This enables an assessment to be made as to the adequacy of the program settings. If the probability of obtaining a successful result (P_{suc}) is low, then it may be necessary to adjust some of the parameters. In particular, it may be necessary to increase either the population size or the number of generations or both.
- An analysis of the frequency of occurrence of each member of the terminal set. This gives the probability that a successful model contains a particular input. In other words, it gives the user a 'feel' for the relevance of

a particular input to the modelling task. This is particularly important in the context of chemical process systems, as the determination of the inputs that affect an output can be as time consuming as actual model development.

- A test as to whether the standard deviation of the residual error of each successful model is significantly different from the best model. This reduces the number of successful models to a set that demonstrates comparable performance on the verification data.

In addition to being able to perform a statistical analysis of the results, the use of multiple runs also allows the engineer to make an educated decision as to the most appropriate model for the task. Typically, the choice would be the simplest model structure or the structure that conforms as closely as possible to current process understanding.

STEADY-STATE MODELLING OF A COOKING EXTRUDER

Cooking extrusion has attracted considerable attention from the food industry in recent years, as it provides an efficient means for the continuous processing of a wide range of foodstuffs. Cooking extruders are already well established in the production of snack foods, baby foods, breakfast cereals and pastas. Extruders allow short residence time and high temperature cooking conditions which result in a high nutritional value of the product and low processing costs (Smith and Ben-Gera, [1979]).

A typical extruder consists of a barrel inside which one or more helical screws rotate to convey the feed material towards a die at one end, as illustrated in Figure (3). The section nearest the feed point is referred to as the solids conveying zone (SCZ), where the screw channels are only partially filled and there is no pressure build-up. At some point along the extruder, the channels become completely filled and the temperature and pressure increase considerably as a result of viscous heat dissipation and material transport. This section is referred to as the melt zone (MZ). Additional temperature control may be provided by heating or cooling sections along the barrel.

Available extruder models range from the purely empirical to those based on fundamental transport equations. In most cases, these models have limited validity in terms of operating conditions and system configuration due to the complex flow patterns, pressure and temperature gradients, changing material properties, and thermal effects that exist within an extruder. For complicated screw configurations, exact calculation of internal conditions may be intractable.

In this section, GP is used to develop a model for the steady-state temperature and pressure at the die, T_d and P_d, the distance of the MZ/SCZ boundary from the feed point, x_{MZ}, and the steady-state temperature distribution, T, of a twin-screw cooking extruder. A rigorous extruder model proposed by Kulshreshtha et. al., [1991] was used to generate 200 random, steady-state data points. The input variables presented to the algorithm were the feed flowrate, Q_F, feed moisture content, m, the screw speed, ω, and the extruder barrel temperature, T_b (assumed uniform for the length of the barrel). The model settings used to generate training data for the GP are summarised in Table 3 below.

Extruder type:	twin-barrel
Barrel length:	25cm
Screw diameter:	5cm
Screw pitch length:	1.0cm
Extruded material:	wheat flour/water mixture
Feed inlet temperature:	20°C
Feed rate, Q_F:	5-50 kg/h
Moisture content, m:	0.1-0.8 v/v
Screw speed, ω:	200-900 rpm
Barrel temperature, T_b:	10-200°C

Table 3: Extruder parameters used to generate training data for the Genetic Program.

Steady-State Modelling Results

When the GP algorithm was applied to the extruder data to generate models for T_d, 45 out of the 50 runs were successful on the basis of a RMS error on the verification data set of

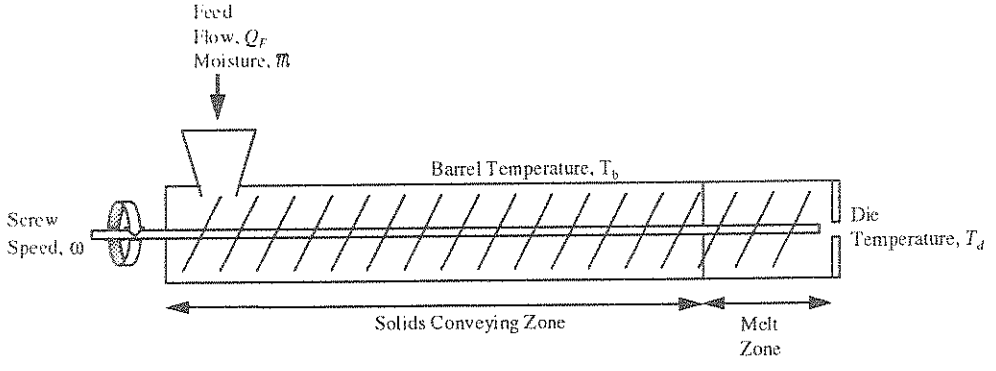

Figure 3: Diagram of a typical cooking extruder.

Parameter	RMS tolerance	Successful runs at 95% confidence	Variables Present in Successful Models	Probability of Occurrence in Successful Models
T_d	10	78-96%	Q_F	90-100%
			ω	90-100%
			$\mathcal{M}$	13-38%
			T_b	2-21%
P_d	4.0×10^8	26-55%	Q_F	82-100%
			ω	82-100%
			$\mathcal{M}$	0-18%
			T_b	0-26%
x_{MZ}	0.002	48-77%	Q_F	86-100%
			ω	86-100%
			$\mathcal{M}$	0-13%
			T_b	0-13%
T	10	26-55%	x	82-100%
			Q_F	82-100%
			ω	82-100%
			$\mathcal{M}$	4-40%
			T_b	2-33%

Table 4: Summary of results obtained using GP to generate extruder input-output models.

less than 10. Using the binomial distribution at the 95% confidence level, 78-96% of all runs should meet this error tolerance. This success rate was deemed satisfactory and no further runs were made. The frequency of occurrence of each member of the terminal set was then determined. This indicated (at the 95% confidence level) that 90-100% of all successful models would contain input variables Q_F and ω, 13-38% would contain $\mathcal{M}$, while only 2-21% would include T_b. This clearly indicated that, as expected, Q_F and ω are the dominant variables influencing the temperature at the die.

After determining that the model residual errors were of zero mean and normally distributed, an F-test was used to reduce the set of candidate models to 8. The simplest model from this set was then chosen as being the most appropriate, yielding:

$$T_d = 2\omega + \frac{\omega}{Q_F}\left(1.579\cdot10^{-3}\omega + 8.140\cdot10^{-3}\right) - 1072\,Q_F + 40.27 \qquad (12)$$

This model structure had an RMS error on the training data set of 5.21 and an RMS error on the (unseen) verification set of 4.69. Thus, an excellent fit can been obtained without the inclusion of T_b and $\mathcal{M}$.

The die temperature is quite non-linear with respect to Q_F and ω, particularly in the high temperature region. It is interesting to note that the inverse relationship between feed flow and die temperature is shown explicitly in the model by the inclusion of the term '$1/Q_F$'. It would prove quite difficult to extract this kind of process information from other 'black box' modelling techniques, such as neural networks.

Similar runs were performed to generate models for the die pressure, P_d, location of the melt zone, x_{MZ}, and the distributed temperature profile within the extruder, T. The results of the runs are summarised in Table 4.

Following the procedure detailed above for T_d, the final model structures were:

$$P_d = \left(28.2 + 6544Q_F - 1.458\omega + 20900\frac{Q_F}{\omega}\right)\times10^8 \qquad (13)$$

$$x_{MZ} = 0.2502 - \left(\frac{393.7{Q_F}^{1.760}}{\omega^2} + \frac{16.22{Q_F}^{0.7579}}{\omega}\right) \qquad (14)$$

$$T = 8.828\times10^{-3}\frac{\omega^2 x}{Q_F} + \omega x^2 + \frac{630.4}{32.15 - 307.5x^2} \qquad (15)$$

Equation (13) is the model obtained for the pressure at the die. It can be seen that the GP algorithm has excluded both input variables $\mathcal{M}$ and T_b from the model, which is a function of Q_F and ω only. As the pressure generated within an extruder is largely due to material transport and viscous heat dissipation, it seems reasonable that Q_F and ω, parameters that have a large influence on these factors, should be present in successful models for P_d.

The location of the MZ/SCZ boundary is predicted by (14). It may be noted that the constant term in the expression (0.2502) is a close approximation to the actual extruder barrel length (0.25m) used in this problem. From this is subtracted a term which represents the length of the MZ. The form of this latter term indicates that the predicted length of the MZ would increase with increasing Q_F, and decrease with increasing ω. This is precisely the relationship expected from our mechanistic knowledge of the process.

The temperature at any point along the extruder's length is predicted by (15). When x, the distance from the feed point, is set to zero in this equation, the expression reduces to

T=19.6, which is approximately equal to the actual inlet temperature of 20°C. The first term in this equation also exhibits an ω^2 dependence, a term that is representative of the viscous power generated by the screw, divided by Q_F, a term which can be associated with the heat capacity rate of the material being extruded. Hence, this term, with a suitable constant, might be loosely interpreted as an expression for the temperature change per unit length due to viscous heat dissipation within the extruder.

For each variable modelled, the algorithm has determined simple expressions that accurately describe the output behaviour. Although all inputs that were thought to affect an output were made available to the algorithm, irrelevant inputs have been eliminated from the final expressions. GP has also shown the ability to determine the relevant constants, and produce models of a structure that provide insight into the underlying process characteristics.

DYNAMIC MODELLING OF A PLASTICATING EXTRUDER

In this section, the GP algorithm is applied to dynamic data from an industrial plasticating extruder used to coat wire. The process utilises a plasticating extruder fitted with a wire coating die through which the wire is continuously drawn.

There are a number of parameters that affect the quality of the wire product, such as the thickness and physical characteristics of the polymer coating. It has been determined experimentally that there is a good correlation between product quality and the viscosity of the polymer as it leaves the extruder, η_d. This can be measured using an on-line rheometer, however the great cost of such equipment has encouraged the development of a model to infer η_d from other measurements. The aim here was to develop a model for η_d, using past values of feed rate, Q_F, temperature at the die, T_d, and screw speed, ω only.

There has been little success in the development of mechanistic models of this process. However, Wagner and Montague [1994] have previously developed neural network and NARMAX (Non-Linear Auto-Regressive Moving Average eXogeneous input) models with some success.

NARMAX models (Billings and Leonartis, [1985]) are a recursive time series relationship between inputs and outputs. The single input, single output case has the general form:

$$y(t) = f\big(u(t), u(t-1), \ldots, u(t-n), y(t-1), \ldots, y(t-m)\big) \qquad (16)$$

where u is the system input and y is the system output. The non-linearities in (16) are usually limited to polynomial functions of the time-delayed input and output variables. Nevertheless, selection of the optimal set of non-linear terms can prove somewhat problematic, particularly if the set of inputs to the system is large.

The approach presented here is based on NARMAX modelling principles, however the GP algorithm was used to develop the model and determine the relevant non-linear groups.

In order to develop a dynamic model, a time history of inputs Q_F, T_d, ω and the predicted output, $\hat{\eta}_d$, were presented to the GP algorithm. For each of the input variables, the time history consisted of the previous three sample values. As preliminary data analysis had revealed a dead time of at least four sample time units between Q_F and η_d, the algorithm was provided with a time history of six sampled values of Q_F beginning with $Q_F(t\text{-}3)$.

Due to the computationally intensive nature of this problem, it was only possible to complete 30 runs. Of these, 5 were deemed successful on the basis of a RMS error on the verification data set of less than 0.16. Clearly, this is insufficient to enable a meaningful analysis of the frequency of occurrence of members of the terminal set in successful models.

An F-test was used to reduce the set of successful models to 4. The model having the smoothest dynamic response from this set was then chosen as being the most appropriate, yielding:

$$\begin{aligned}\eta_d(t) &= 0.712\hat{\eta}_d(t-1) - 0.453Q_F(t-5) \\ &+\sqrt{4.27\times10^{-4} - 0.0210Q_F(t-4) + 0.0227Q_F(t-4)^2} \\ &\quad -0.356T_d(t-3)Q_F(t-4)^2 - 0.590\end{aligned} \qquad (17)$$

This model structure had a RMS error on the training data set of 0.135 and a RMS error on the verification set of 0.128. This result is significantly better than that obtained by a linear (ARMAX) model structure (with a RMS on verification data of 0.221).

A plot of the actual output compared with the estimated output calculated using (17) is shown in Figure 4. Plot (a) contains the training data set, while (b) contains the verification data. The data has been scaled. As the plots show, an excellent fit has been obtained on both the training and verification data.

From (17) it may be observed that $Q_F(t\text{-}3)$ has been effectively eliminated from the model, consistent with the presence of a time lag between Q_F and η_d. Also, no past values of screw speed have been included in the successful model. It is possible that there was insufficient excitation of screw speed in the data used.

It should be noted that the model was developed using a relatively small set of plant data, approximately 350 data points. A comparable result obtained using for example neural networks would require a far larger set of training data.

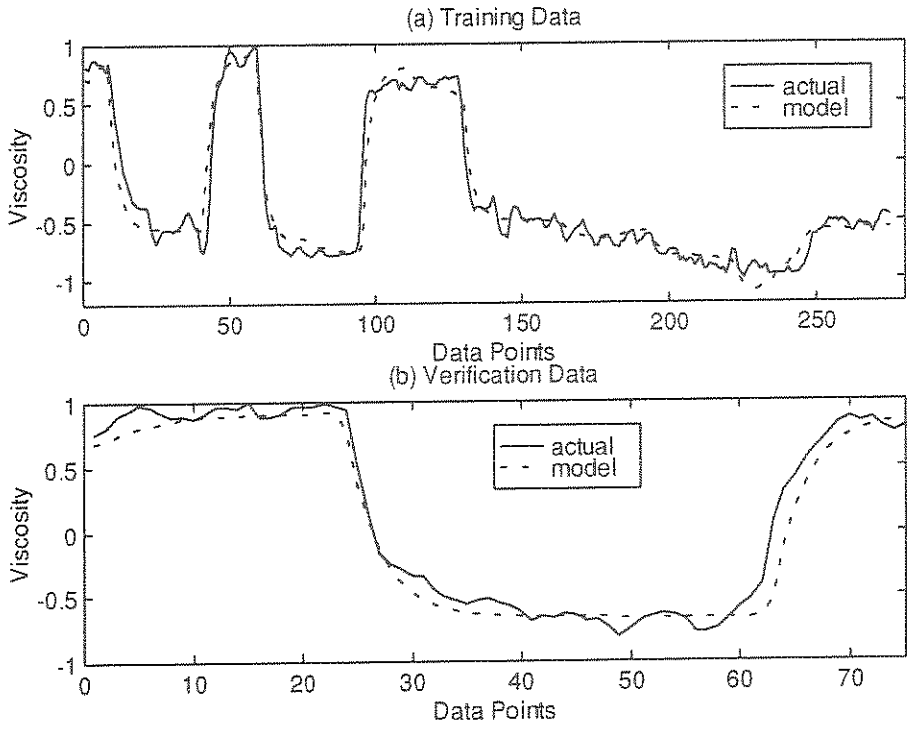

Figure 4: (a) Training data and model fit, (b) verification data and model fit.

CONCLUSIONS

The application of Genetic Programming to the development of steady-state and dynamic input-output models of extrusion processes has been considered. Two examples were used to highlight the utility of the algorithm. The results revealed that the algorithm could generate an appropriate model based solely on observed data. A distinct advantage of this method is that no a priori assumptions have to be made about the actual model form - the structure and complexity of the model evolve as part of the problem solution.

ACKNOWLEDGMENTS

The authors would like to thank Gary Montague for supplying the wire coating extrusion data used in this paper.

REFERENCES

Goldberg, D.E., *Genetic Algorithms in Search, Optimisation and Machine Learning*, Addison-Wesley, Reading, Massachusetts, 1989.

Kinnear, K.E., *Advances in Genetic Programming*, MIT Press, 1994.

Koza, J.R., *Genetic Programming: On the programming of computers by means of natural selection*, MIT Press, 1992.

Kulshreshtha, M.K., Zaror, C.A., Jukes, D.J. and Pyle, D.L. A generalised steady-state model for twin screw extruders, *Trans. IChemE, Vol 69(C)*, 189-199, 1991.

Leonartis, I.J., Billings, S.A., Input-output parametric models for non-linear systems, part 1: Deterministic non-linear systems, *Int. J. Control, 41(2)*, 303-328, 1985.

Wagner, M.G., and Montague, G., Modelling and control of a wire coating process, *IEE Conf. Pub., No. 389, V1*, Michael Faraday House, UK., 374-378, 1994.

Smith, D.B., and Ben-Gera, I., The Application of High Temperature Short Time Extrusion Cooking in the Food Industry, in *Food Process Engineering Vol 1*, 726-744, Linko, P. *et al* (Eds), Applied Science Publishers, UK., 1979.

South, M.C., The Application of Genetic Algorithms to Rule Finding in Data Analysis, Ph.D. thesis, Dept. of Chemical and Process Engineering, University of Newcastle upon Tyne, England, 1994.

Tenenbaum, A.M. and Augenstein, M.J., *Data Structures Using Pascal*, Prentice Hall Inc, Englewood Cliffs, NJ 07632, 1981.