Non-Linear Continuum Regression Using Genetic Programming

Ben McKay Ben.McKay@ncl.ac.uk
Mark Willis Mark.Willis@ncl.ac.uk
Dominic Searson D.P.Searson@ncl.ac.uk
Gary Montague Gary.Montague@ncl.ac.uk

Dept. Chemical & Process Engng.
University of Newcastle
NE1 7RU, UK
http:\\lorien.ncl.ac.uk\control\

Abstract

In this contribution, genetic programming is combined with continuum regression to produce two novel non-linear continuum regression algorithms. The first is a 'sequential' algorithm while the second, adopts a 'team-based' strategy. Having discussed continuum regression, the modifications required to extend the algorithm for non-linear modelling are outlined. The results of applying the derived non-linear algorithms to the development of an inferential model of a food extrusion process are then presented. The superior performance of the 'sequential' algorithm, as compared to a similar non-linear partial least squares algorithm, is demonstrated. In addition, these results clearly demonstrate that the 'team-based' strategy significantly outperforms the 'sequential' approach.

1 INTRODUCTION

In this paper Genetic Programming (GP) is used to develop two non-linear continuum regression algorithms for building input-output models of chemical process systems. A 'team-based' strategy is investigated, and compared to a more traditional 'sequential' approach.

Why are chemical engineers interested in data-based modelling ?

Recent years have seen an increase in the emphasis on product 'quality', economic process performance, environmental and safety issues in the process industries, and these factors have placed significant demands on existing operational procedures. Process monitoring, optimisation and advanced control have the potential to satisfy many of these demands. However, in order to realise the benefits of these techniques, it is generally necessary to have an accurate model of the process. While it may be possible to develop such a model from first-principles (using a detailed knowledge of the physics and chemistry of a system), there are a number of drawbacks to this approach. As many industrial process systems are extremely complex and relatively poorly understood, the development of a realistic model can take a considerable amount of time and effort. In addition, the modelling process inevitably involves a number of simplifying assumptions that have to be made in order to provide a tractable solution. Therefore, a mechanistic model will often be costly to develop and may be subject to inaccuracies. Consequently, data-based modelling (using plant data to build an input-output model that describes the response of process outputs to changes in inputs, without attempting to represent the underlying process mechanisms) presents a popular alternative.

Why use non-linear models ?

Many chemical process systems are non-linear in nature. Therefore, in order to achieve sufficient model accuracy, a non-linear model structure is typically required.

Why not use a standard GP algorithm ?

The performance of a standard GP algorithm when used to develop non-linear models of chemical process systems (using either simulated or industrial input-output data) can be disappointing when compared to other non-linear regression techniques such as feedforward neural networks (e.g. see McKay, 1997, Hiden, 1998). Generally, a GP algorithm is much slower than competitive techniques (especially for increasing numbers of input variables) and the prediction errors on unseen (test) data are typically higher, indicating a less accurate model.

Why the interest in Continuum Regression ?

Our recent work has shown that strategies that reduce the algorithm's search space can help the overall performance of a GP algorithm (when used for model development). A number of multivariate statistical modelling techniques provide systematic procedures for the decomposition of an input-output relationship. By extracting and structuring information in an appropriate manner, these techniques can be used to reduce the effective search space. In Hiden *et al.* (1998) it was demonstrated that the combination of GP with one such technique, known as Partial Least Squares (PLS), could produce results of comparable accuracy to alternative non-linear regression techniques such as neural networks. However, PLS is merely one of a family of linear multivariate statistical modelling

techniques. It is Continuum Regression (CR) that provides a unified framework encompassing all of these techniques. Wise and Ricker (1997) demonstrated the effectiveness of CR, developing models of improved accuracy when compared to alternative approaches (without sacrificing model robustness and generalisation). However, standard CR algorithms produce linear models. In this paper, we combine GP and CR to produce non-linear CR algorithms. The objective being the development of an algorithm that can develop accurate and robust non-linear models.

How is this paper organised ?

The next section provides some background information on linear regression. CR is then introduced and the range of techniques it encompasses are explained. Next, the modifications required to extend the algorithm to a non-linear strategy are outlined. Two approaches are proposed, a 'sequential' and a 'team-based' algorithm. Having detailed the algorithms, their performance on a benchmark example are studied: The development of an inferential model of a food extrusion process. After discussion of the results, conclusions are drawn and recommendations for further work are made.

2 LINEAR REGRESSION

Given a set of input and output measurements for a process, a typical modelling objective is to obtain a relationship, that can be used to explain the variation in an (n x 1) output vector, **y,** in response to changes in an (n x m) input matrix, **X**. This can be expressed mathematically as follows:

$$\mathbf{y} = f(\mathbf{X}) + \mathbf{e} \quad (1)$$

Where the function f(.) is chosen so as to minimise the vector of prediction errors, **e**. If f(.) is linear in the parameters, then the input-output model reduces to:

$$\mathbf{y} = \mathbf{X}\mathbf{r} + \mathbf{e} \quad (2)$$

where **r** is a (m x 1) vector of regression coefficients.

A family of linear multivariate statistical modelling algorithms have been developed for the purpose of estimating **r**. Three of the more common and effective of these are Multiple Linear Regression, Principal Component Regression and Partial Least Squares.

What is Multiple Linear Regression ?

Multiple Linear Regression (MLR) uses one of the many variants of the Least Squares technique (such as Batch Least Squares) to parameterise Equation 2. A draw-back of this method is that it fails to produce accurate and robust models if the input data is correlated (the columns of input data are linearly related to each other). This is the norm when data is collected from chemical process systems.

What is Principal Component Regression ?

PCR avoids the problems associated with the modelling of correlated input data (*ie*. singular solutions or imprecise parameter estimations) by transforming the inputs into a new set of uncorrelated data (typically of reduced dimensionality), and then performing MLR between this transformed data set and the output data. The procedure is based on a technique known as Principal Component Analysis (PCA), one of the oldest and best documented multivariate statistical techniques (Pearson, 1901). PCA is performed on the input data, **X**, and generates a set of transformed input variables (known as principal components) that are uncorrelated and ranked in term of significance. This allows the least significant principal components (ones that only describe process noise) to be discarded prior to modelling.

More rigorously, PCA rotates **X** to produce an (n x m) scores matrix, **T** and an (m x m) loadings matrix, **V** (describing the required rotation) such that, $\mathbf{T}=\mathbf{X}\mathbf{V}$. The loading matrix **V** contains the eigenvectors of the input data correlation matrix, $\mathbf{X}^T\mathbf{X}$. The scores matrix, **T**, contains orthogonal projections of the input variables, sorted (from left to right) in terms of decreasing contribution to the variance in the input data. Retaining the first 'p' principal components leads to a reduced (n x p) scores matrix $\mathbf{T}_p$ and an (m x p) loadings matrix $\mathbf{V}_p$. To perform PCR, batch least squares is used to regress $\mathbf{T}_p$ against **y**, giving the (p x 1) regression vector **b**, such that $\mathbf{y}=\mathbf{T}_p\mathbf{b} + \mathbf{e}$. The input-output mapping is then given as,

$$\mathbf{y} = \mathbf{X}\mathbf{V}_p\mathbf{b} + \mathbf{e} \quad (3)$$

Unfortunately, as PCA only considers the variation in the input data, there is no guarantee that the variation represented by the major principal components will correspond to the variation that best describes the relationship with the output variable.

What is Partial Least Squares ?

Partial Least Squares (PLS) was first described in Wold (1966). As with PCR, the PLS algorithm overcomes problems associated with measurement noise and correlation between input variables. However, an additional advantage of PLS (as compared to PCR), is that by considering both the input variance and the input-output covariance, the algorithm provides the opportunity for more accurate model development. A good introduction to the technique can be found in Geladi and Kowalski (1986).

PLS is commonly implemented using the NIPALS (Non-linear Iterative Partial Least Squares) algorithm. NIPALS sequentially extracts pairs of 'latent vectors' (analogous to the columns of the scores matrix in PCA) from the input and output data. A univariate (single input, single output) regression is performed at each stage, to model the relationship between these latent vector pairs. The final model is then constructed by summing the contribution from each step.

More rigorously, PLS (starting with i=1, $\mathbf{X}_1=\mathbf{X}$ and, $\mathbf{y}_1=\mathbf{y}$) sequentially extracts latent vector pairs, $\mathbf{t}_i$ and $\mathbf{u}_i$, from $\mathbf{X}_i$ an $\mathbf{y}_i$, in order of decreasing predictive power. For the single output case, the vector $\mathbf{u}_i$ is equal to $\mathbf{y}_i$, while the vector $\mathbf{t}_i$ corresponds to the projection of $\mathbf{X}_i$, in the

direction most correlated to $\mathbf{u}_i$ (i.e. $\mathbf{t}_i=\mathbf{X}_i\mathbf{w}_i$. where $\mathbf{w}_i$ are the optimal input projection weights). The vectors $\mathbf{t}_i$ and $\mathbf{u}_i$, are then regressed to obtain a univariate linear model with a regression parameter b_i (such that $\mathbf{u}_i = b_i\mathbf{t}_i + \mathbf{e}_i$) . As each latent vector is calculated the 'information' used is deducted from the input-output data and these residuals ($\mathbf{X}_{i+1}$ and $\mathbf{y}_{i+1}$) are used to calculate the next set of latent vectors. This procedure is repeated until additional latent vector pairs fail to improve the model performance. The final model is then constructed by summing the contributions from each of the N_{LV} significant latent vector pairs,

$$\hat{\mathbf{y}} = \sum_{i=1}^{N_{LV}} b_i \, \mathbf{X}_i \mathbf{w}_i \tag{4}$$

where $\hat{\mathbf{y}}$ is the estimated output $\mathbf{y}$. Substituting the expression for calculating the input residuals ($\mathbf{X}_{i+1} = \mathbf{X}_i - \mathbf{X}_i\mathbf{w}_i\mathbf{w}_i^{\mathrm{T}}$) into Equation 4 and recalling that $\mathbf{X}_1=\mathbf{X}$ gives**,**

$$\begin{aligned}\hat{\mathbf{y}} &= \mathbf{X} b_1 \mathbf{w}_1 \\ &+ \mathbf{X}(\mathbf{I} - \mathbf{w}_1\mathbf{w}_1^{\mathrm{T}}) b_2 \mathbf{w}_2 + \ldots \\ &+ \mathbf{X}(\mathbf{I} - \mathbf{w}_1\mathbf{w}_1^{\mathrm{T}} - \ldots \\ &\quad - \mathbf{w}_{(N_{LV}-1)}\mathbf{w}_{(N_{LV}-1)}^{\mathrm{T}}) b_{N_{LV}} \mathbf{w}_{N_{LV}}\end{aligned} \tag{5}$$

Using the fact that the $\mathbf{w_i}$ are orthogonal (and thus $\mathbf{w}_i^{\mathrm{T}}\mathbf{w}_j=0$, for $i\neq j$), this reduces to a form that is equivalent to Equation 2,

$$\mathbf{y} = \mathbf{X}\sum_{i=1}^{N_{LV}} b_i \mathbf{w}_i + \mathbf{e} \tag{6}$$

Using PLS, difficulties associated with modelling correlated input variables are avoided by calculating latent variables sequentially and only performing a univariate regression at each stage. Problems associated with measurement noise are also overcome by projecting the inputs and outputs onto a lower dimensional space. In addition, considering both the input variance and the input-output covariance leads to improved model accuracy.

How are the methods related ?

For a linear input-output mapping, if all principal components or latent variables are retained when performing PCR or PLS, the two methods are equivalent to MLR. It is primarily the difference in emphasis placed on variation in the input data compared to covariance between the inputs and the output that distinguishes the various techniques. A mathematical framework for unifying these techniques will be introduced in the next section.

3 CONTINUUM REGRESSION

Stone and Brooks (1990) and Wise and Ricker (1993) have demonstrated that MLR, PCR and PLS can be unified under a single framework known as Continuum Regression. At one extreme of the continuum is PCR, at the other is MLR, while PLS is in the middle. A qualitative interpretation of this concept can be obtained by considering this continuum in terms of the emphasis that is placed on the variance in the input data, when compared to the covariance between the inputs and the outputs. With MLR, only the input-output covariance is considered when formulating the model (in fact the inputs are regarded as independent).With PCR, a model is developed between the output and the significant principal components. These components represent the most important variations in the input data, but their covariance with the output is not necessarily maximised. While with PLS, a compromise between MLR and PCR is obtained by consideration of both the correlation in the input data and input-output covariance. CR allows a smooth transition between these techniques.

What is the CR algorithm ?

The CR algorithm proposed by Wise and Ricker (1993), first creates a transformed data set by applying a sophisticated scaling technique to the input data. This effectively 'warps' the input space to place either more or less emphasis on existing correlations (depending on a factor to be referred to as the 'continuum coefficient'). PLS (using NIPLAS, as outlined in Section 2) is then performed on the transformed data set.

The first step in transforming the input data is to decompose $\mathbf{X}$ using the singular value decomposition (SVD) as,

$$\mathbf{X} = \mathbf{U}\Sigma\mathbf{V}^{\mathrm{T}} \tag{7}$$

where $\mathbf{U}$ is an (n x n) matrix of the eigenvectors of $\mathbf{X}\mathbf{X}^{\mathrm{T}}$ and $\mathbf{V}$ is an (m x m) matrix of the eigenvectors of the data correlation matrix, $\mathbf{X}^{\mathrm{T}}\mathbf{X}$. The diagonal elements of the (n x m) matrix Σ are the positive square roots of the eigenvalues of $\mathbf{X}^{\mathrm{T}}\mathbf{X}$ and are called the singular values (all other elements of Σ are zero). A modified $\mathbf{X}$ matrix, ($\mathbf{X}^{\mu}$**)** is then formed as,

$$\mathbf{X}^{\mu} = \mathbf{U}\Sigma^{\mu}\mathbf{V}^{\mathrm{T}} \tag{8}$$

where the matrix Σ^{μ} is the singular values raised to the power μ ($0 \le \mu \le \infty$). Where μ will be referred to as the continuum coefficient. The standard NIPALS algorithm is then applied to $\mathbf{X}^{\mu}$ and **y,** giving a final model of the following form,

$$\hat{\mathbf{y}} = \sum_{i=1}^{N_{LV}} b_i \, \mathbf{X}_i^{\mu} \mathbf{w}_i \tag{9}$$

In a similar manner to Equation 4, this can be converted into a form that is equivalent to Equation 2 as follows,

$$\mathbf{y} = \mathbf{X}\mathbf{V}\Sigma^{(\mu-1)}\mathbf{V}^{\mathrm{T}}\sum_{i=1}^{N_{LV}} b_i \mathbf{w}_i + \mathbf{e} \tag{10}$$

What is the qualitative interpretation of the continuum coefficient ?

$\mu \to 0$	CR tends towards MLR.
$\mu = 1$	CR is equivalent to PLS.
$\mu \to \infty$	CR tends towards PCR.

4 NON-LINEAR CR USING GP

Two approaches for combining GP and CR to create a non-linear CR algorithm are proposed. The first evolves inner models sequentially, while the second evolves a team of inner models simultaneously.

What is the 'sequential algorithm' ?

The 'sequential' non-linear CR algorithm performs a GP run each time a non-linear inner model is required (*ie*. at each iteration of the latent variable loop). The mathematical details of the algorithm can be found in Appendix 1. At Step 8, a standard GP algorithm minimises the cost function '*J*', by judicius choice of non-linear function, $f_i(.)$. The optimal continuum coeficient (μ) is obtained by an outer loop that performs a unidimensional search using Brent's method (Press *et al*., 1992).

What is the'team-based' algorithm ?

Rather than perform a GP run each time a non-linear inner model is required, it is possible to evolve a team of inner models simultaneously using a multi-population GP algorithm. Each team is composed of N_p individuals (each an inner model), drawn from N_p separate populations $\{P_1, \ldots, P_{Np}\}$. Team members are assigned specific tasks, based on the population from which they are drawn (*eg*. Members of population P_1 are assigned the task of developing an inner model between the first latent vector pair, while population P_j contains candidates for the *j*'th inner model). In this manner, a team of non-linear functions $\{f_1(.), \ldots, f_{Np}(.)\}$.are evolved to minimise the overall model prediction error (**e** in Equation 1).

Fitness is calculated on a team basis. The fitness of a team is determined by implementing the CR algorithm given in Appendix 1 (using the inner models defined by the given team). For the *i*'th latent variable pair, at Step 8, the team member corresponding to the *i*'th inner model is evaluated . Each individual in a team is then assigned the teams fitness.

Each population is homogenous (*ie*. only individuals from the same populations can be crossed over). The crossover strategy employed does not maintain the integrity of teams (*ie*. it is non-cohesive). The various populations are treated independently when selecting individuals for crossover (rather than crossing over complete teams).

Prior to fitness evaluation, a teams continuum coefficient (μ) was determined in one of two ways. Either, μ was optimised using Brent's method (with a probability of 0.1), or a random Gaussian perturbation was made to the value of μ associated with the fittest team of the previous generation. This perturbed value of μ was then used to evaluate the teams fitness.

	Sequential	**Team-Based**
Function Set	+, -, /, *, protected ^, protected √, exp(), protected ln(), tanh()	
Terminal Set	$\mathbf{t}_i$,constants	$\{\mathbf{t}_1, \ldots \mathbf{t}_m,\}$,constants
Output:	A single inner model for the *i*'th latent variable pair.	A team of N_P inner models.
Population Size:	40	500
Fitness function	Linear ranking	
Crossover Probability	0.7	
Mutation Probability:	0.2	
Direct Reproduction Probability	0.1	
Proportion of elite individuals retained:	0.1	
Termination criterion	40 Gens.	60 Gens.
Parameter optimisation	Yes	No
Total # of runs	20	
Average # individuals processed per run:	5.0×10^5	2.5×10^5

Table 1: Configuration of the GP Algorithms

5 CONFIGURATION OF THE GP ALGORITHMS

Table 1 shows the values of the most important GP algorithm control parameters, as well as a summary of the terminal and function sets used. It may be noted that the 'team-based' algorithm used a significantly larger population than the 'sequential'. However, as the sequential algorithm was employing a parameter optimiser (Levenberg-Marquardt non-linear least squares), the sequential algorithm actually processed twice as many individuals per run.

6 EXPERIMENTAL METHOD

For all of the results presented, the Root Mean Square (RMS) error between the actual and predicted output was calculated based on an 'unseen' (or test) data set. This test set is used to verify the generalisation of the model. Due to the stochastic nature of the GP algorithm, using it on anything but the simplest of systems produces different results for every run. Therefore, in order to investigate the performance of a GP-based algorithm for model building multiple runs were performed for each technique.

7 MODELLING A FOOD EXTRUSION PROCESS

As described in Elsey et al (1997), the parameters of this simulated process have been fitted to plant data obtained from a pilot-scale APV Baker MPF 40 cooking extruder, processing a mixture of corn flour and water. Given the screw speed, feed rate, feed moisture content, feed temperature and barrel temperature profile, the model calculates the degree of gelatinisation of the product. This was selected as the process output of interest in these studies. Therefore, the terminal set for this system consists of four variables, screw speed, feed rate, feed moisture content and feed temperature. Four hundred steady-state readings were available for modelling. This data was split evenly with the first 200 points being used

to train the models and the remaining 200 points reserved for testing.

This data set has become our benchmark example for comparing non-linear modelling techniques. A selection of pertinent results are presented in Table 2. These results demonstrate that standard GP performs poorly when compared to a feedforward neural network (FANN). This led to the development of two GP-based non-linear PLS algorithms.

The first, GP_PLS1 (the so called 'quick and dirty' approach), is equivalent to the 'sequential' algorithm presented in this paper, when the continuum coefficient is set to one. While the second, GP_PLS2, demonstrated that optimisation of the input projection weights ($\mathbf{w}_i$) can be beneficial in reducing the model prediction errors. This is because, in linear PLS the $\mathbf{w}_i$ is calculated to achieve maximum 'linear' covariance between $\mathbf{t}_i$ and $\mathbf{u}_i$ even though the inner models are in fact non-linear. Therefore, for a given non-linear function, $\mathbf{w}_i$ may not correspond to the direction that produces the best approximation of $\mathbf{u}_i$. Hiden *et al.* (1998) demonstrated that GP_PLS2 could produce results of comparable accuracy to a FANN. However, it was concluded that optimising $\mathbf{w}_i$ was an unacceptable computational burden (especially with increasing numbers of input variables), limiting the application of the algorithm. As such, in this work no attempt was made to optimise the input projection weights.

Figure 1 shows a comparison of the RMS error distributions obtained using the 'sequential' and 'team-based' algorithms (while Table 3 summarises the average and best RMS errors). It may be noted that the 'sequential' algorithm outperforms GP_PLS1. This is because the optimisation of μ has a similar effect to optimising the input weights $\mathbf{w}_i$ (in that it changes the projection of the input data in an attempt to maximise the model fit). However, as there is only one continuum coefficient (as compared to $\mathbf{w}_i$ which is an (m x 1) vector, where m is the number of input variables) the optimisation problem becomes more tractable.

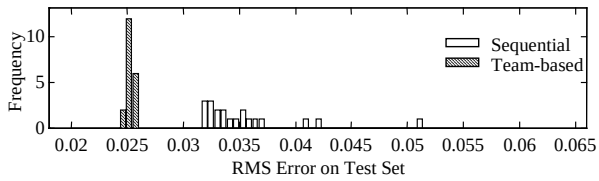

Figure 1: Model error distributions on the cooking extruder data.

Method	Mean Test RMS	Min. Test RMS	Source
FANN	0.031	0.024	McKay (1997)
Standard GP	0.280	0.188	Hiden(1998)
PLS - Linear	0.925	0.925	Hiden *et al.* (1998)
GP_PLS1	0.049	0.044	
GP_PLS2	0.022	0.017	

Table 2: Summary of prior modelling work on the cooking extruder data.

Method	Mean Test RMS	Min. Test RMS	μ Best Model	N_{LV} Best Model
'Sequential'	0.0354	0.0316	1.09	3
'Team-Based'	0.0252	0.0243	0.61	3

Table 3: Results for the 'sequential' and 'team-based' algorithms on the cooking extruder data.

It is also apparent that the 'team-based' approach produces significantly lower RMS errors on the test set, compared to the 'sequential' algorithm. This is despite the fact that the sequential algorithm processed more individuals (as shown in Table 1).

The 'sequential' algorithm finds the best fit for each latent variable pair in turn. This procedure effectively constrains the type of solutions the algorithm will find. While resulting in good local solutions at each step, it is conjectured that it may be prone to being trapped in local minima. In comparison, the 'team-based' algorithm 'optimises' all of the inner models simultaneously. As Fitness is calculated on a team-basis, this allows a trade-off between team member contributions to the overall fitness measure. This provides the potential for escape from local minima and the possibility of obtaining a more global optimum. The best set of inner models obtained using the 'team-based' algorithm are given by Equations 11, 12, and 13,

$$u_1 = 2.61 \exp(\tanh(\tanh(\exp(4t_1)))) - 4.58 \quad (11)$$

$$u_2 = 0.0498 (|\tanh(t_2)|^{|t2|} + (2\, t_2 - 0.0474)/\exp(t_2) + u1 - 0.0503\, t_2^2) - 0.0196 \quad (12)$$

$$u_3 = 0.180\, (t_3 - 0.0381) \tanh(2t_3) - 0.0352 \quad (13)$$

8 CONCLUSIONS

The food extrusion benchmark example shows that non-linear CR using GP gives superior performance to a similar implementation of non-linear PLS. More specifically, the 'sequential' non-linear CR algorithm outperforms GP_PLS1 of Hiden *et al.*(1998). It is conjectured that this is because the optimisation of μ has a similar effect to optimising the input projection weights $\mathbf{w}_i$. Both change the projection of the input data in an attempt to maximise the model fit. However, optimising the continuum coefficient is less computationally intensive than optimising $\mathbf{w}_i$ (which is an (m x 1) vector, where m is the number of input variables).

The case study also demonstrates the effectiveness of the 'team-based' approach to non-linear CR, as compared to the 'sequential' algorithm. By finding the best fit for each latent variable pair in turn, the 'sequential' technique effectively constrains the type of solutions the algorithm will find. While concentrating on achieving good inner models at each step, it may be prone to being trapped in local minima. By optimising all of the inner models simultaneously, the 'team-based' algorithm allows trade-offs between individuals contributions to the overall team

fitness. This may allow the evolution of a more 'globally' optimal solution.

As explained in Section 4, the 'team-based' algorithm employs a non-cohesive reproduction strategy. This means that teams are broken up during reproduction (*ie* the integrity of the team is not maintained). If only one type of team strategy is going to work (*ie*. if there is only one 'good' way of performing each task), then disrupting the integrity of the teams will probably not have a detrimental effect. While we believe that the non-linear CR modelling task is an example of such a system (where the form of each inner model, and hence the task of each team member, is largely dictated by the data set), future work will examine the effect of implementing a cohesive crossover strategy.

As noted in Section 7, the 'team-based' non-linear CR algorithm does not outperform GP_PLS2, the weight optimised non-linear PLS algorithm of Hiden *et al*., (1998). However, it was considered that optimising the projection weights was currently an unacceptable computational burden. As such, finding an improved method of implementing the input weight optimisation will be a focus of future work.

9 REFERENCES

Elsey, J. Riepenhausen, J. McKay, B., Barton, G.W. and Willis, M.J., (1997), 'Modelling and Control of a Food Extrusion Process', *Computers and Chemical Engineering*, Vol. S21, pp S361-S366.

Geladi, P. and Kowalski, B.R. (1986), 'Partial least squares regression: A tutorial', *Analytica Chimica Acta*, Vol.185, pp.1-17.

Hiden, H.G. (1998), *Data-Based Modelling using Genetic Programming*, PhD Thesis, Dept. Chemical and Process Engineering, University of Newcastle, UK.

Hiden, H.G., McKay, B., Willis, M.J. and Montague, G.A. (1998) 'Non-linear partial least squares using genetic programming', *Proc. 3rd Annual Conf. On Genetic Programming*, University of Wisconsin, Maddison, USA. July 22-25th., pp.128-133.

McKay, B., (1997), *Studies in Data-Based Modelling*, PhD Thesis, Dept.Chemical Engineering, The University of Sydney, Australia.

McKay, B., Willis, M.J. and Barton, G.W., (1997), 'Steady-state modelling of chemical process systems using genetic programming', *Computers and Chemical Engineering*, Vol.21, No.9, pp.981-996.

Pearson, K., (1901), 'On lines and planes of closest fit to systems of points in space', *Phil. Mag.*, Ser.6, 2(11), pp.559-572.

Press, W.H., Flannery, B.P., Teukolsky, S.A and Vetterling, W.T., (1992), *Numerical Recipes in C: The Art of Scientific Computing*, 2nd Ed, Cambridge University Press, USA.

Stone and Brooks (1990), 'Continuum regression: Cross-validated sequentially constructed prediction embracing ordinary least squares, partial least squares, and principal components regression', *J.R.Statist. Soc.*, B,52, pp.337-369.

Wise, B.M. and Ricker, N.L. (1993) 'Identification of finite impulse response models with continuum regression', *Journal of Chemometrics*, Vol. 7, 1-14.

Wold, S., (1966), 'Non-linear estimation by iterative least squares procedures', *Research Papers in Statistics*, Ed. David, F., Wiley, New York, USA.

APPENDIX 1:
Non-Linear Continuum Regression Algorithm

Sequential Algorithm: The algorithm in Table A1 represents a single function evaluation for a given value of μ. This is called repeatedly by Brent's method to optimise μ. Note that a full GP run is performed at Step 8 to optimise the fit of each inner model.

Team-Based Algorithm: The algorithm in Table A1is called by a multi-population GP algorithm as a single fitness evaluation, for a given team of inner models, and a given μ.

(1)	Mean centre and normalise the input and output data (**X** and **y**)		
(2)	Perform SVD on scaled input data (**X**)	$\mathbf{U}\Sigma\mathbf{V}^{\mathrm{T}} = \mathbf{X}$	(n x n)(n x m)(m x m)
(3)	Generate modified input data ($\mathbf{X}^{\mu}$)	$\mathbf{X}^{\mu} = \mathbf{U}\Sigma^{\mu}\mathbf{V}^{\mathrm{T}}$	(n x m) matrix
(4)	Set the output latent vector ($\mathbf{u}_i$) equal to the output.	$\mathbf{u}_i = \mathbf{y}$	(n x 1) column vector
(5)	Calculate weights ($\mathbf{w}_i$) that correspond to the direction in $\mathbf{X}_i^{\mu}$ with the greatest covariance with $\mathbf{u}_i$.	$\mathbf{w}_i = (\mathbf{X}_i^{\mu})^{\mathrm{T}}\mathbf{u}_i$	(m x 1) column vector
(6)	Normalise $\mathbf{w}_i$ to unit length (Note ‖.‖ refers to the Euclidean Norm)	$\mathbf{w}_i = \mathbf{w}_i / \lVert\mathbf{w}_i\rVert$	(m x 1) column vector
(7)	Calculate the latent vector $\mathbf{t}_i$ which is the projection of $\mathbf{X}_i^{\mu}$ onto the direction defined by $\mathbf{w}_i$.	$\mathbf{t}_i = \mathbf{X}_i^{\mu}\mathbf{w}_i$	(n x 1) column vector
(8)	Sequential: $\min_{\mathrm{f}_i(.)} J \quad J = (\mathbf{u}_i - \mathrm{f}_i(.\mathbf{t}_i))(\mathbf{u}_i - \mathrm{f}_i(.\mathbf{t}_i))^{\mathrm{T}}$	Team-Based: Use supplied $\mathrm{f}_i(.)$	
(9)	Calculate residual matrices	$\mathbf{X}_{i+1}^{\mu} = \mathbf{X}_i^{\mu} - \mathbf{t}_i\mathbf{w}_i^{\mathrm{T}}$ $\mathbf{y}_{i+1} = \mathbf{y}_i - \mathrm{f}_i(\mathbf{t}_i)$	(n x m) matrix (n x 1) column vector
(10)	If $i < N_{LV}$, {Return to Step 5; $i = i+1$}		
(11)	Construct the final model	$\hat{\mathbf{y}} = \sum_{i=1}^{N_{LV}} \mathrm{f}_i(\mathbf{X}_i^{\mu}\mathbf{w}_i)$	(m x 1) column vector

Table A1: The non-linear continuum regression algorithm.