IDENTIFICATON OF ALBEBRAIC AND STATE SPACE MODELS USING GENETIC PROGRAMMING

Kyaw Tun and S. Lakshminarayanan*

Department of Chemical and Environmental Engineering
National University of Singapore, Singapore 117576
**Corresponding Author. Telephone: +65-68748484 Email*: chels@nus.edu.sg

Abstract: We describe the development of an automated modelling system that uses the genetic programming paradigm. The Genetic Modelling System (GeMS) is capable of generating appropriate algebraic (linear and nonlinear) and ODE systems from experimental data. The utility of GeMS is illustrated through several examples involving laboratory systems and simulated kinetic data.

Keywords: System identification, Nonlinear models, State space models

1. INTRODUCTION

The present day chemical industry is characterized as "data rich and information poor" – there is a need for tools and methods that are capable of extracting useful process information from the terabytes of data that are archived in plant databases. A plethora of tools such as multivariate statistics, neural networks, Bayesian networks, self-organizing maps, CART (Classification and Regression Trees) etc. are available at present to convert data into useful process knowledge (models, rules etc.). A relatively recent addition to this family is the genetic programming (GP) that has its origins in genetic algorithms (GA) (Holland, 1975). GP is a biologically inspired, domain-independent method that genetically breeds populations (of computer programs, mathematical formulae etc.) to solve problems. The first runs of genetic programming were made in 1987 but the first detailed description was available in a textbook published several years later (Koza, 1992). When accompanied by robust parameter estimation methods, GP has the ability to determine the "optimal" structure of the model that "best" describes the given data set. By its very nature, GP also provides a family of comparable solutions from which the informed user can select a model that is commensurate with the physics/chemistry of the system as well as the intended end use. The GP method is computationally intensive but can come quite handy in situations where little or nothing is known about the structure of the process model.

GP has been applied in many areas ranging from engineering and finance to medicine. We primarily focus on applications in the field of process systems engineering here. McKay et al. (1997) apply GP to the steady state modelling of chemical processes and consider a tutorial example followed by applications to two typical processes - a vacuum distillation column and a chemical reactor system. Willis et al., (1997) extend the GP methodology to the development of dynamic input-output models and demonstrate its workability by the development of inferential estimation models for a vacuum distillation column and a twin screw cooking extruder. Marenbach (1998) use GP for the construction and refinement of models for a simulated bioprocess. This work employs block diagrams (such as those available in SIMULINK® under MATLAB®) for representing the models – this is in contrast to other works that are equation oriented. Greeff and Aldrich (1998) illustrate GP-based empirical modeling by means of three examples, two of which are based on data pertaining to leaching experiments. Lakshminarayanan et al. (2000) use a composite GP-PCA (genetic programming combined with a multivariate statistical technique called principal components analysis (PCA)) approach to generate nonlinear models for industrial product design applications. Gao and Loney (2001) combine GP with neural net to evolve a polymorphic neural network and apply it to predicting pH in a simulated CSTR. Grosman and Lewin (2002) employ GP for nonlinear model predictive control (NMPC). In their work, the nonlinear model predictive control uses predictions provided by the GP generated model and this is shown to improve the control performance on two multivariable simulated processes: a mixing tank and a Karr liquid-liquid extraction column.

In this paper, we wish to describe the features of a Genetic Modelling System (GeMS) that has been developed in our laboratory. GeMS uses the genetic

programming paradigm to identify models ranging form algebraic to state space models from experimental data. The paper is organized as follows. In section 2, we provide a quick overview of GP based modelling and detail some of the innovative concepts implemented in GeMS. Section 3 provides some interesting simulation and experimental case studies of modelling using GeMS. One of the key features of GeMS is the automated assembly of a nonlinear ordinary differential equation (ODE) system to represent the behaviour of process systems using experimentally observed data. This will be demonstrated in Section 4. Conclusions and planned future work will be spelt out in the final section.

2. GENETIC PROGRAMMING

Since a lot of information is available in the literature about genetic operators etc. we will only point out the unique features of GeMS here.

2.1 Data structure for gene and chromosome

The most important thing is how the population data is stored. We need to store all the chromosomes from initial to final population; it is easy to see that the storage requirements can be very high. Whenever a new chromosome is created, it is compared with all the members that were ever "created" to ensure that unnecessary calculations are avoided. It is easy to understand that this comparison must be performed very fast.

GeMS uses a novel approach to store the genes and chromosomes – this provides fast access and makes it easily extensible. GeMS employs indexing to the gene library. It is similar to using pointers in C language. In contrast to using string to represent chromosomes, the use of indexing requires less memory to store and facilitates faster comparison and manipulation. Since MATLAB is optimized for matrix manipulations, the entire population is stored in a single three-dimensional matrix. The last (third) dimension denotes the chromosome number. Let X denote the whole population. The second chromosome or chromosome number 2 is referred as X(:,:,2) - this is a two dimensional matrix. The first row contains the index of the gene library and the remaining rows represent properties or states of gene of respective column. For a typical run having population size of 200, 20 generations and with 30 as the maximum chromosome size, the storage requirement is roughly 5 MB. The size is reduced by a factor of 3 to 5 when saved to disk using sparse matrix format.

The main reason behind this flexibility is the implementation of the representation scheme using a pointer or index to the gene library. Addition of a new gene to the gene library can be accomplished without hard coding of the program simply by pointing the gene functions file (m-file) to gene's evaluation function handle. Single information (a pointer) is not enough to fully represent a gene in most cases. Additional information such as number of parameters, number of input arguments, state properties (for ODE systems) and so on must also be stored along with the gene. To address these issues, a novel representation is used. A two-dimensional matrix represents the chromosome. Each column represents a gene. Usual genetic operators manipulate along the column of this matrix. Some special genetic operators such as the adaptation operator manipulate the third row or state of the gene.

The advantages of using a matrix to represent a chromosome are: (i) Flexibility, generality and ease of implementing novel genes (ii) it requires less memory to store the information. This is particularly important since GeMS stores all chromosomes that are generated. (iii) it permits fast searching and manipulation of chromosome and (iv) it allows easy implementation of genetic operators. The unique encoding of chromosomes allows us to handle different domains without altering the main GP algorithm.

GeMS also exploits any available *a priori* knowledge in order to limit the search space or to transform the model assembled by GP into more effective models. GeMS assembles models randomly, so sometimes it may assemble absurd model structures such as $\exp(\exp(\exp(u_1)))$. In such cases, we may limit how many operands one function may take. GeMS also determines the optimal parameter position in the model. For example, the model $k_1 u_1 * k_2 u_2$ is reduced to $k_1 u_1 u_2$. In order to accommodate varied *a priori* knowledge, a new concept called evolution policy is implemented. The user can define the evolution policy using keywords for action to be taken (inclusion, exclusion, parameter reduction, etc) for conditions such as repetition of a specific gene, existence of a specific gene, etc.

2.2 Parameter Estimation

GP is used only for structure generation according to the well-known and some newly developed (superposition crossover and adaptation) genetic principles. Proven global and local optimization methods combined with a user-defined rule-based parameter pruning technique are used for parameter estimation in GeMS.

A run of GP spends most of the time in parameter estimation than in the genetic operations itself. The model structure may be nonlinear and has several local minima. Choosing right optimizer is critical. Torni *et al.* (1999) mention that choosing the right optimizer is based on features of problem, which is postulated depending on 1) region of attraction of global minimum 2) cost of function evaluations 3) embedded or isolated global minimum and 4) number of local minima. GeMS uses stochastic "global" optimization methods such as simulated annealing, genetic algorithms and differential evolution in conjunction with local optimization techniques like the Gauss

Newton method to determine the optimum values of the parameters.

2.3 Fitness measure

Fitness is a numeric value assigned to each member of the population to provide a measure of the appropriateness of the individual as a solution to the problem in question. The definition of fitness measure has a direct and significant bearing on the resulting solution. Fitness measure is defined such that the most suited model will have the highest (or lowest) score whereas the poorest model will have the lowest (or highest) score and all other models lie in the continuum between these extremes.

The goal of having a fitness evaluation is to give continuous feedback to the evolutionary algorithm regarding which individuals should have a higher probability of being allowed to multiply and reproduce and which individuals should have a higher probability of being removed form the population. The fitness function is calculated on the validation data set or the combination of both training and validation data sets.

GeMS uses well-known model selection criteria employed for system identification such as Minimum Description Length (MDL) or Akaike's Information Criterion (AIC). Generally, such criteria include two terms: (a) a term that accounts for mismatch between the experimental data and model predictions and (b) penalty for number of parameters. The first term generally decreases with increasing model complexity and the second term increases with increasing model complexity.

GeMS also incorporates a modified fitness function that basically adds a heuristically defined complexity index to the MDL or AIC criterion. This strategy is used to generate and nurture less complex models and hence the resulting models turn out to be much simpler.

GeMS uses MATLAB ODE solvers such as RK45 and ODE15S to determine the model responses for each model generated by the GP and parameter estimation sub-modules.

3. CASE STUDIES

Several researchers have employed GP for the steady state and dynamic modeling of chemical process (e.g. McKay et al., 1997). The dynamic modeling is achieved by including the lagged values of the inputs and/or the output variables thereby converting the problem into an algebraic modeling exercise. Interestingly, even some simple algebraic systems cannot be identified if suitable transformations are not enforced on the dependent variables (y). Usually, the programs attempt to construct models of the form $y = f(\underline{x},\underline{k}) + \varepsilon$ where $f(\underline{x},\underline{k})$ is a functional form (to be identified) that involves the input variables $\underline{x}$ and the parameter vector $\underline{k}$. ε denotes the noise component. However, this form is not always adequate. We have therefore made provisions in GeMS to also search for implicit models of the form $g(y,x) = f(\underline{x},\underline{k}) + \varepsilon$ where $g(y,x)$ is a functional form that also needs to be identified along with $f(x,k)$. For such cases, a chromosome consists of two trees: the first tree for function *f* and last tree for function *g*. Standard genetic operations are then used to identify the model.

Example 1: Nonlinear Algebraic Equation

The data for this example was generated from the equation $x^2 + y^2 = 1$. The output 'y' was corrupted with IID noise. The [x, y] data was presented to both the explicit and implicit algebraic equation identification modules. All runs of the explicit algebraic identification resulted in the model y = -0.02, which is nothing but the mean value of y in the data set. Eight of the ten runs with the implicit model identification procedure identified the correct model as (y*y)=(1+(-0.992*(x*x))). The data, true model and the identified model are shown in Figure 1.

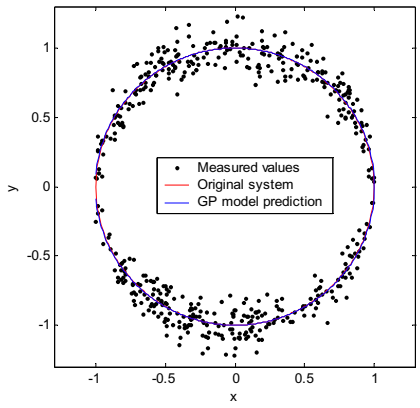

Fig 1. Data vs. model prediction for example 1

Example 2: Identification of a dynamic model for a laboratory process

An experimental process system (see schematic in Figure 2) was designed and installed recently in our research laboratory. This equipment facilitates experimental studies related to the dynamics and control of linear as well as nonlinear multivariable processes. The system is fitted with industrial grade sensors and valves; the data acquisition and control operations are performed using MATLAB/SIMULINK running on a personal computer. Industrial standard OPC communication is used. Besides such "remote control", the process can also be controlled using local devices. The sensors (thermocouples) are located such that the process dynamics and control studies can be conducted over a wide range of time delays.

The focus of our current application is the exit temperature of water in Tank 1. The inlet water that

comes in at about 27°C is heated using an electric heater that can supply up to 9 kW power. The level of water in Tank 1 is kept constant by manipulating the inlet water flow into Tank 1. The water exiting from tank 1 flows to Tank 2 through a long-winded tube. The temperature of water in Tank 2 is the controlled variable (TT5; y); the input variables are the heating power applied to Tank 1 (HT1; u1) and the temperature of the water entering Tank 1 (TT8; u2). These measurements are available at 40 seconds interval. The objective, here, is to build a model that makes a one step ahead prediction of TT5 given past values of all the variables. Two sets of data are collected – one for model construction and the other for pure validation.

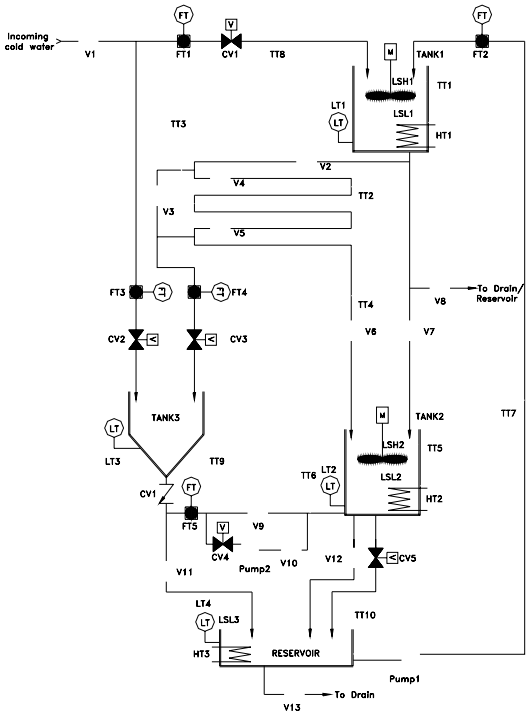

Fig 2. Experimental equipment setup (Example 2)

Ten runs were conducted. The best model obtained from GeMS is given below:

$$y_k = 26.39 + 0.711\, y_{k-1} - 2.11\, y_{k-2} + 0.1\, y_{k-1}\, y_{k-2} - 0.027\, y_{k-1}^2 - 0.00075\, y_{k-2}^2\, y_{k-1} + 0.0021\, u1_{k-3} + 0.0042\, u1_{k-4}$$

This model has a RMSE of 0.056 in the modelling data set and a RMSE of 0.066 during its "pure" validation on the test data set. The model validation is shown in Figure 3. The low RMSE value for the prediction set as well as the results shown in Figure 3 points to the fact that the process characteristic is captured well by the model reported above. Note that the variable u2 (temperature of the water entering Tank 1) does not feature in the model. This is due to the lack of excitation in u2 and is not due to the physical characteristics of the system. The GP procedure has demonstrated that it is able to distinguish between the important and unimportant variables.

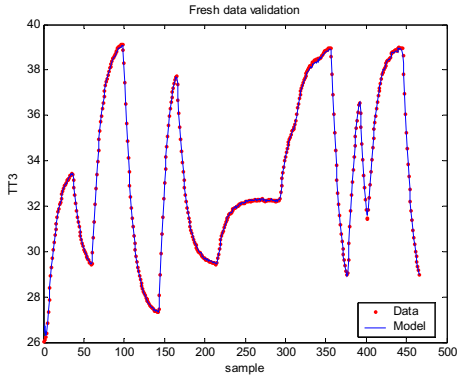

Fig 3. Model validation for Example 2

Example 3: Identification of a dynamic model for an experimental heat exchanger

For this case study, we consider the modelling of data obtained from an experimental heat exchanger system. The data set was provided by Dr. Eskinat and is also considered in the work by Eskinat *et al.* (1991) in the context of identification of a block oriented nonlinear model (hammerstein model: the model is characterized by a nonlinear static element followed by a linear dynamic element). The details of the experimental system can be found in the above reference. The nonlinearity in the system is caused by the presence of two distinct operating regions corresponding to the high and low process water flow rates. A dataset containing 334 input-output samples was available. Sixty-five percent of this data set was used to obtain the model using DACS-GP. The remaining data was used for model validation.

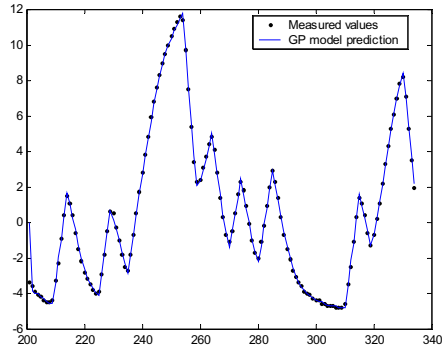

Fig 4. Model validation for Example 3

The best model obtained from GeMS is:

$$y_k = 0.84\, y_{k-1} - 6.17\, u_{k-1} + 7.55\, u_{k-1}^2 + u_{k-4}$$

The validation of the model on the test portion of the data is shown in Figure 4. The model provides an excellent approximation to the true system behavior. It is interesting to note that the identified model is very close to the hammerstein model (except for the u_{k-4}

term) structure. In fact, the above model provided a slightly better validation RMSE as compared to that obtained by methods (Eskinat *et al.* 1991; Lakshminarayanan *et al.* 1995) that assumed the hammerstein model structure and estimated its parameters from the given data.

4. IDENTIFICATION OF STATE SPACE MODELS

The genetic programming technique has been applied to a wide range of domain including finance, medicine and engineering. However, most of the research articles (at least in the engineering literature) focus on the identification of algebraic models – even dynamic models are converted to algebraic models through the incorporation of lagged variables (e.g. Rodriguez and Fleming, 1998; Hinchliffe and Willis, 2003). To the best of our knowledge, we have seen only one published work on the identification of an ordinary differential equation (ODE) system using the GP paradigm (Cao *et al.* 1999). Their models are shown to have good prediction capability. However, the ODE systems they generate are often found to be non-autonomous. The identification of ODE systems pose some interesting issues: firstly, the GP generated models might turn out to be very stiff and integrating them could pose run time problems (stalling). The choice of a suitable integrator becomes crucial; secondly, the computational cost can be exorbitant during the parameter estimation and fitness function evaluation steps. The parameter estimation for systems of known structure is itself a challenging task (Esposito and Floudas, 2000). In our case, the GP could come out with complex structures involving many parameters thereby making the parameter estimation even more difficult.

GeMS uses a multi-tree scheme to represent several simultaneous ODE's. It used one tree for each state equation. Apart from this modification in representation, no other changes in genetic operations are required. In this module, crossover is allowed across different states or trees.

The efficiency of GeMS in modeling batch data from three chemical reaction systems is demonstrated below. These examples have been taken from the parameter estimation literature in which the structure of the ODE system is assumed to be fixed and known.

Example 4: Cracking of gas oil

This model represents the catalytic cracking of gas oil (A) to gasoline (Q) and other side products (S). Only the concentrations of A and Q were measured; therefore, the concentration of S does not appear in the model for estimation. Tjoa and Biegler (1991) estimated the parameters of a proposed differential equation system model using experimental data from this system. Owing to space constraints, we do not report the data here – the reader is referred to Table 23 of Esposito and Floudas (2000) for the data set. Here, we assume no knowledge of the chemistry or the reaction mechanism. We only provide GeMS with the measurements of the two states A and Q. We specify that the terminal gene should include state variable z_1, z_2 and constant. The arithmetic operators that were included are '+', '–', '*'and '/'.

The best models from the ten independent runs conducted are summarized in the Table 1.

Table 1: Best models of GP runs for example 4

Fitness	RMSE	Model
-93.1	0.0083	(-14.81*(z1*z1)), ((12.32*(z1*z1))+(-7.859*z2))
-93.1	0.0083	(-14.81*(z1*z1)), ((12.32*(z1*z1))+(-7.859*z2))
-91.6	0.0083	(-14.81*(z1*(z1-(z1-z1)))), ((-7.859*z2)+ (12.32*(z1*z1)))
-93.1	0.0083	(-14.81*(z1*z1)), ((12.32*(z1*z1))+(-7.859*z2))
-93.1	0.0083	(-14.81*(z1*z1)), ((12.32*(z1*z1))+(-7.859*z2))

For comparison, the "true" model that was used to generate this data set is:

$$\frac{dz_1}{dt} = -14\ z_1^2\ ;\ \frac{dz_2}{dt} = 12\ z_1^2 - 8\ z_2$$

The parameters in the model generated by GeMS are different from the true parameters owing to the noise added to the measurements.

Example 5: Lotka-Volterra system

This problem has been studied by Luus (1998). This model is a representation of the predator-prey model used in ecology. The system is described by two differential equations:

$$\frac{dz_1}{dt} = k_1 z_1 (1 - z_2)$$

$$\frac{dz_2}{dt} = k_2 z_2 (z_1 - 1)$$

with the initial conditions z_0 = [1.2 1.1]. The experimental data is available for t = [1,10].

In this model, z_1 represents the population of the prey and z_2 the population of the predator. The solutions to these equations point to the periodic nature of the predator and prey populations. The data used in the study were generated using values for the parameters as $k_1 = 3$ and $k_2 = 1$ with a small amount of normally distributed random error with $\sigma = 0.02$ and zero mean added to the observations. The data set is available in Table 26 of Esposito and Floudas (2000). The best models obtained from six runs are shown in Table 2. The fourth model has the best fitness value. It is clear that the best identified model matches with the true model very closely. With this model, a plot of the measured data vs. model prediction appears as shown in Figure 5.

Table 2: Best models of GP runs for example 5

Fitness	RMSE	Model
-31.92	0.024	(-11.099*(z1-(z2*z2))), (-10.859*(z1-z2))
-31.76	0.025	(4.6951*((z2*z2)-z1)), (4.4575*(z2-z1))
-31.68	0.025	(9.8078*(z1-(z1*z2))), (0.34589*(z1-z2))
-51.09	0.0031	(2.9812*((z1+(-0.0047458*z1))-(z2*z1))), ((z2*z1)-z2)
-33.17	0.019	((8.1585*(z2-(z2*z2)))+(0.26568*z1)), (0.3633*(z1-z2))
-43.46	0.0075	(-7.4976*(z1-(z2*z1))), (2.9281*(z1-(z1*z1)))

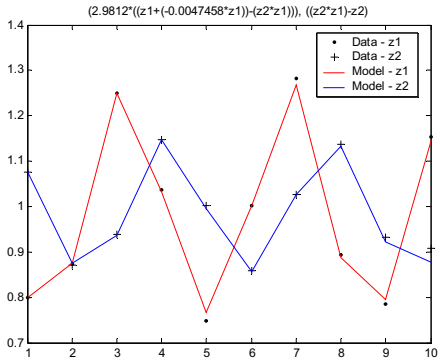

Fig 5. Model fit obtained for Example 5

5. CONCLUSIONS

We have described and demonstrated the features of a GP-based modelling system called GeMS. The capabilities include the identification of nonlinear models for static and dynamical systems. GeMS is able to identify explicit and implicit form of equations. A unique feature of GeMS is the implementation of a robust module that can identify nonlinear ODE's directly from measured data. We have also integrated some nonparametric and exploratory data analysis techniques into GeMS (not explained here) so that its search space can be quite focused.

ACKNOWLEDGEMENTS

Financial support from the Academic Research Fund (ARF), National University of Singapore is gratefully acknowledged.

REFERENCES

Cao, H., Yu, J., Kang, L., Chen, Y., and Chen, Y. (1999). The kinetic evolutionary modeling of complex systems of chemical reactions. *Computers & Chemistry,* **23**(2), 143-152.

Eskinat, E., Johnson, S. H. and Luyben, W. L. (1991). Use of Hammerstein models in identification of nonlinear system. *AICh.E J.*, **37**(2), 255-268.

Esposito, W. R. and Floudas, C. A. (2000). Global Optimization for the Parameter Estimation of Differential-Algebraic Systems. *Ind. Eng. Chem. Res*., **39**, 1291-1310.

Gao, L. and Loney, N.W., (2001). Evolutionary Polymorphic Neural Network in Chemical Process Modelling, *Comp. & Chem. Engg.*, **25**, 1403-1410.

Greeff, D. J. and Aldrich, C. (1998). Empirical Modelling of Chemical Process Systems with Evolutionary Programming. *Comp. & Chem. Engg.*, **22**(7-8), 995-1005.

Grosman, B. and Lewin, D.R., (2002). Automated nonlinear model predictive control using genetic programming. *Comp. & Chem. Engg.*, **26**, 631–640.

Hinchliffe, M.P and Willis, M.J. (2003) Dynamic systems modeling using genetic programming. *Comp. & Chem. Engg.*, **27,** 1841-1854.

Holland, J. H., (1975). *Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence***.** MIT Press, Cambridge, MA.

Koza, J. R. (1992). *Genetic programming: on the programming of computers by means of natural selection*. Cambridge, Mass., MIT Press.

Lakshminarayanan, S., Shah. S. L. and Nandakumar K. (1995). Identification of Hammerstein models using multivariate statistical tools. *Chem. Engg. Sci.,* **50** (22), 3599-3613.

Lakshminarayanan, S., Fujii, H., Grosman, B., Dassau, E., and Lewin, D.R., (2000). New product design via analysis of historical databases. *Comp. & Chem. Engg.,* **24,** 671-676.

Luus, R., (1998). Parameter Estimation of Lotka-Volterra Problem by Direct Search Optimization. *Hung. J. Ind. Chem.,* **26**, 287.

McKay, B., Willis, M. and Barton, G., (1997). Steady-state modeling of chemical processing system using genetic programming. *Computer Chem. Engg.*. **21** (9), 981-996.

Marenbach P. (1998). Using prior knowledge and obtaining process insight in data based modeling of bioprocesses. *SAMS,* **31**. 39-59.

Rodriguez-Vazquez, K. and Fleming, P.J. (1998). Multiple-objective genetic programming for nonlinear system identification, Electronics Letters, 34(9), 930-931.

Tjoa, T. B., Biegler, L. T. (1991). Simultaneous Solution and Optimization Strategies for Parameter Estimation of Differential-Algebraic Equation Systems. *Ind. Eng. Chem. Res.*, **30**, 376.

Törni, A., Ali, M.M., and Viitanen, S. (1999). Stochastic Global Optimization: Problem Classes and Solution Techniques. *Journal of Global Optimization,* **14,** 437–447.

Willis, M., Hiden, H., Hinchliffe, M., McKay, B. and Barton, G. W. (1997) Systems modeling using genetic programming. *Comp. & Chem. Engg.*, **21,** S1161-S1166.