Hierarchical Causal Parameter Abduction in Integral-Hybrid Logic Nets

D. Al-Dabass, D. Evans and S. Sivayoganathan

School of Computing and Mathematics
Faculty of Computing and Technology
Nottingham Trent University
Nottingham NG1 4B
david.al-dabass@ntu.ac.uk

Abstract

Hybrid logic nets contain nodes that exhibit recurrent characteristics, in that a node output shows temporal tendencies even when the inputs are constant (Al-Dabass et al. 1999a and b). Given the behaviour trajectory of such a node it is required to abduct the values of the causal parameters. A new causal parameter estimation algorithm is given for abducting knowledge embedded within recurrent logic networks. The results show the new algorithm to be superior to previous algorithms when two of the causal parameters possess temporal behaviour.

Introduction

Numerous intelligent systems in practice exhibit complex behaviour that cannot be easily modelled using simple logic nets, such as a combination of basic arithmetic and logical nodes, or low order differential models. In this paper we re-cast this problem in terms of hybrid logic nets which consist of combinations of static nodes, either logical or arithmetic, and recurrent nodes. The behaviour of a typical recurrent node is given and modelled as a second order dynamical system. The causal parameters of such a recurrent node may themselves exhibit temporal behaviour that can be modelled in terms of further recurrent nodes. Layers of recurrent nodes are added until a complete account of the behaviour of the system has been achieved.

Deduction and Abduction in Inference Networks

To engineer a knowledge base to represent intelligent systems, a multilevel structure is needed. By it's very nature the knowledge embedded within these systems is continually changing and need dynamic paradigms to represent and acquire their parameters from observed data. In a normal inference network the cause and effect relationship is static and the effect can be easily worked out through a deduction process by considering all the causes through a step-by-step procedure which works through all the levels of the network to arrive at the final effect. More over, reasoning in the reverse direction, such as that used in diagnosis, starts with observing the effect and working back through the nodes of the network to determine the causes; this is termed abduction.

Dynamical Abduction Processes. Work in this paper extends these ideas to recurrent or dynamical systems networks where some or all the data within the knowledge base is time varying. The effect is now a time dependent behaviour pattern, which is used as an input to a differential abduction process to determine the knowledge about the system in terms of time varying causal parameters. These causal parameters will themselves embody knowledge (meta knowledge) which is obtained through a second level abduction process to yield 2nd level causal parameters. These abduction processes consist of a differential part to estimate the higher time derivative knowledge, followed by a non-linear algebraic part to compute the causal parameters.

Logic Network Models

Many physical, economical and biological phenomena exhibit a temporal behaviour even when the input 'causal' parameters are constant, Fig. 1.

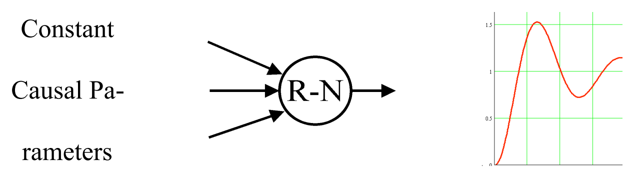

Fig. 1. A Recurrent Node (R-N) exhibits a temporal behaviour at the output despite having constant causal parameters.

The causal parameters themselves may be the output of other nodes, which may either be recurrent nodes or static nodes,- the later may be logical or arithmetic.

To model this oscillatory behaviour we propose a second order integral hybrid model shown in Fig. 2. This model is based on the well known second order dynamical system which has the following form:

$\omega^{-2} x'' + 2.\zeta.\omega^{-1}.x' + x = u$

Where x is the output of the node and ω, ζ and u are the natural frequency, damping ratio and input respectively, which represent the 3 causal parameters that form the input. To configure this differential model as a recurrent network, a twin integral elements are used to form a hybrid integral-recurrent net as shown in Fig. 2.

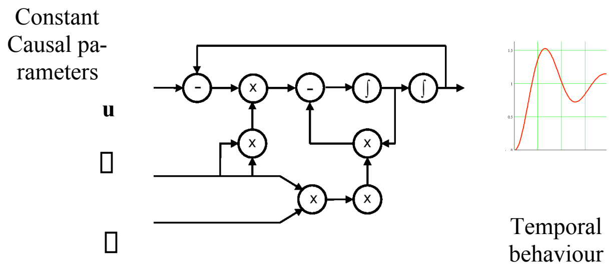

Fig. 2. Hybrid integral-logic net to model the temporal behaviour of the node in Fig. 1

Hierarchical Causal Parameters with Temporal Behaviour

The output trajectory of the system may be more complex than can be represented by a simple second order differential model. In this case each causal parameter may itself be modelled as having a dynamical behaviour, which may or may not be oscillatory. One such case is where two of the 3 causal parameters have 2nd order dynamical characteristics, as shown in Fig. 3.

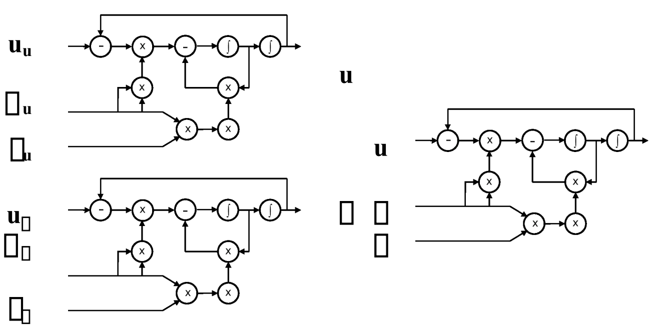

Fig. 3. Two of the causal parameters of the final node have temporal behaviour modelled as 2nd order hybrid integral recurrent nets.

Knowledge Abduction Algorithms

The knowledge embedded in such a model is represented by the parameters of the various recurrent nodes. By taking measurements of the system trajectory, tracking algorithms are employed to estimate the values of these parameters on line. Parameter tracking algorithms fall into several categories depending on the manner of accessing relevant information from the trajectory of the object and the order of parameter variation used in the derivation of the relevant models.

Multipoint Algorithms

Several algorithms are easily derived to estimate values of causal parameters using as many points from the trajectory as necessary to form a set of simultaneous algebraic equations. The parameters to be estimated form the unknown variables and the trajectory values and their time derivatives form the constant parameters of these equations (Al-Dabass 1999a and b).

Algorithm 1: Three-Points in x, x' and x''.
This algorithm yields the following expressions for estimated □, □, and u:

□□^{2} = [□''13.□'12 □ □''12.□'13] / [□12.□'13 □ □13.□'12]
□□ = [-E□$^{□2}$.□''12 □□12] / [2. □□$^{□1}$. □'12]
Eu = E□$^{□2}$. x_1'' + 2. □□. E□$^{□1}$. x_1' + x_1

Algorithm 2: Two-Points and One Extra Derivative.
This algorithm gives the following expressions for estimated □, □, and u:

□□^{2} = [(□'12). (□'''12) - (□''12)^2]/ [(□'12)^2 - □12 . □''12]
□□ = [-E□$^{□2}$.□''12/□'12 - □12 /□'12] / [2. E□$^{□1}$]
Eu = E□$^{□2}$. x_1'' + 2. □□. E□$^{□1}$. x_1' + x_1

Single Point Algorithms

These are classified according to the order of the parameter variation used in the derivation, i.e. constant, first order polynomial (constant u' but u''=0), 2nd order polynomial (constant u'' but u'''=0) etc .

Algorithm 3: Constant Parameters. Consider using the 1st to 4th time derivatives at a single point. Given the second order system:

□$^{□2}$ x'' + 2. □.□$^{□1}$.x' + x = u (1)

Differentiate with respect to t and divide by x'':

□$^{□2}$ x'''/ x'' + 2. □.□$^{□1}$ + x'/ x'' = 0 (2)

and differentiate with respect to t again to give:

□$^{□2}$.[(x''. x'''' - x'''^{2}) / x''^{2}] + 0 + [(x''^{2} – x'. x''') / x''^{2}] =0 (3)

We get expressions for estimated □, estimated □, using (2), and estimated u:

□□^{2} = [x''. x'''' - x'''^{2}] / [x'. x''' - x''^{2}]
□□ = -[E□$^{□2}$ x''' + x'] / [2. E□$^{□1}$.x'']
Eu = E□$^{□2}$. x'' + 2. □□. E□$^{□1}$. x' + x

Algorithm 4: First Order Parameters. Let the first time derivative of u to be non zero. For simplicity assume that both a and b (the coefficients of x'' and x' to make symbol manipulation easier) to be constant and hence disappear on first differentiation. The extra information needed for u' to be non zero is extracted from the 5th time derivative of the trajectory.

a.x'' + b.x' + x = u (4)

Differentiate wrt to t and assume u' is non zero to give:

a.x''' + b.x'' + x' = u' (5)

Differentiate again and set u'' = 0 gives:

a.x'''' + b.x''' + x'' = 0 (6)

Divide Equation 4 by x''' to isolate b:

a.x''''/x''' + b + x''/x''' = 0 (7)

Differentiate again to eliminate b:

a.(x'''''.x'' - x''''^{2})/x'''^{2} + (x'''^{2} - x'' . x'''')/x'''^{2} = 0 (8)

Re-arranging for a gives:

a = (x'' . x'''' - x'''^{2})/(x''''' . x''' - x''''^{2}) (9)

Solve for *b* by substituting a from equation (9) into equation (7):

b = -x''/x''' - a.x''''/x'''

Substituting for a and manipulating gives:

b = (x''.x''''' - x'''.x'''')/(x''''^{2} - x''' . x''''') (10)

We can now substitute these values for *a* and *b* into equation 4 to solve for u,

u = a.x'' + b.x' + x

Derivative Abduction Architecture

The structure of each cell of the recurrent network is shown in Figure 4. The output of each cell feeds the input to the next one to generate the next higher order time derivative, see Fig. 5. The output of the system and the cascade of 1st order recurrent network filters were simulated using the 4th order Runge-Kutta method in Mathcad. The derivatives vector is shown in Figure 6. Figure 7 shows a typical set of high order time derivatives abducted from the output trajectory of a system displaying temporal behaviour.

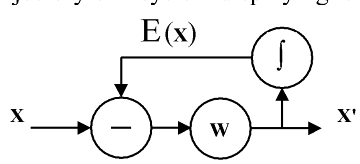

Fig. 4. A single stage recurrent sub-net using an integrator in the feedback path to abduct the derivative x' = w(x-E(x))

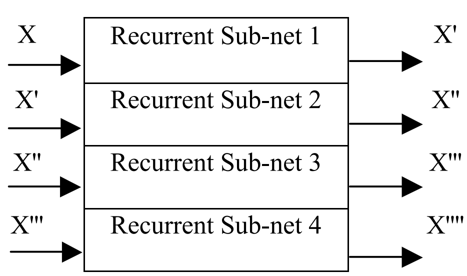

Fig. 5. A 4th order recurrent network to abduct 1st to 4th time derivatives.

$$x := \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \qquad D(t,x) := \begin{bmatrix} x_2 \\ -\square^2 \cdot x_1 - 2\cdot\square\cdot\square\cdot x_2 + \square^2\cdot u \\ G\cdot(x_1 - x_3) \\ G\cdot[G\cdot(x_1 - x_3) - x_4] \\ G\cdot[G\cdot[G\cdot(x_1 - x_3) - x_4] - x_5] \\ G\cdot[G\cdot[G\cdot[G\cdot(x_1 - x_3) - x_4] - x_5] - x_6] \\ G\cdot[G\cdot[G\cdot[G\cdot[G\cdot(x_1 - x_3) - x_4] - x_5] - x_6] - x_7] \end{bmatrix}$$

$$Z := \mathrm{Rkadapt}(x, t0, t1, N, D)$$

Fig. 6. A cascade of 5 recurrent cells plus the 2nd order trajectory model.

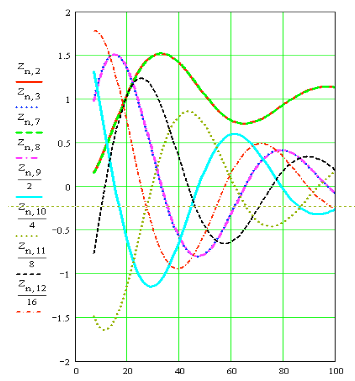

Fig. 7. A typical set of time derivatives abducted from the trajectory of an oscillatory system.

Results and Discussion

Algorithm 3 using Constant Parameters

This algorithm uses a single time point but two further time derivatives compared to Algorithm-1. The filter cascade is increased by one again to provide a continuous estimate of the 4th time derivative x''''. The separation problem disappears altogether now to provide a continuous estimate of all parameters at each point on the trajectory. Program 3 [Zreiba, 1999, Appendix A] was run, and the result of the estimation are given in Fig. 8; which shows fast and accurate convergence.

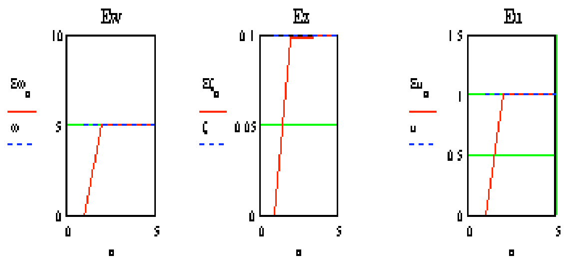

Fig. 8. Estimated constant omega, zeta and u

Discussion. Estimated values for constants parameters were close to the desired set values. The derived algorithms estimated □, □ and u for a good range of values: □ from 1 to10, □ between +/- (0.01 to 1), and u between +/- (0.5-40), and gave accurate estimates. Estimation errors decreased as □ increased, particularly for small □ (less than 0.5): where oscillation provided wide variation in the variables to decrease errors. The differences between the (simulated) system time derivatives (x, x' and x") and their estimates from the filter cascade depended on G (the cut-off frequency): high G provided more accurate estimation of derivatives but made the algorithms prone to noise and vice versa. Another disadvantage of high G from the simulation point of view is that simulation time increased considerably due to the integration routine adapting to ever smaller steps. The algorithms provided progressively faster convergence with Algorithm-3 being the fastest to converge.

Results For Algorithm 4

Mathcad routines were set up to generate the input u as second order system with its own parameters of natural frequency, damping ratio and input. The input subsystem damping ratio was set to 0.05 to generate an oscillatory behaviour for long enough to test the parameter tracking algorithm thoroughly. The frequency of the input was set to 16 radians per second, one quarter of the frequency of the object natural frequency. The derivative generation cascade was increased by one to produce the fifth time derivative. The results are shown in Fig. 9 below.

The actual input is the smooth sinusoidal trace, which gives approximately one and one quarter cycles over a period of half a second as expected, i.e. 16 radians/s = 2.546 Hertz. The widely oscillating jagged trace shows the results from the previous constant u derivation algorithm which is failing completely to track the input parameter.

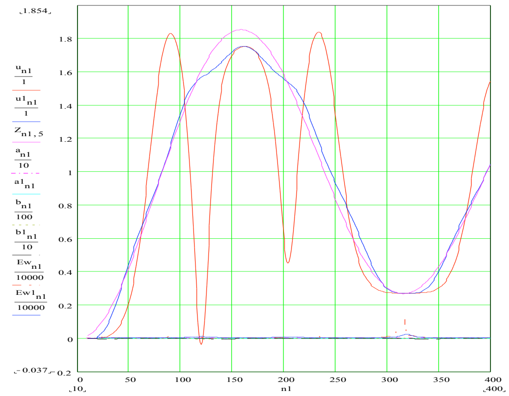

Fig. 9. Results of the high order algorithm

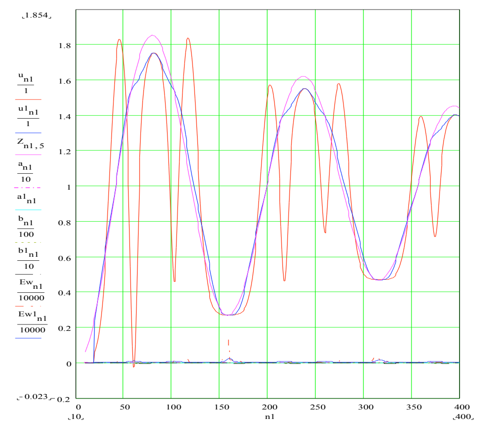

Fig. 10. Results of the high order algorithm for one second integration time.

The slightly distorted sinusoidal trace shows the result of the new algorithm, which is managing to track the input much more closely; however it start to diverge slightly near the peak of the cycle but then returns to track it well right down and round the lower trough of the input trajectory.

To check the quality of tracking as time progresses, a second set of results was obtained with integration time extended to one second to give two and a half cycles. The results are shown in Fig. 10. It is clear that tracking remain stable. It is interesting to note that the old algorithm while completely failing to track the upper half of the input trajectory it seems to track it well during the its lower half but not as well as the new algorithm.

Figures 11 and 12 show the case when two of the causal parameters are changing. It is clear that the new algorithm provide much better tracking than the previous algorithm.

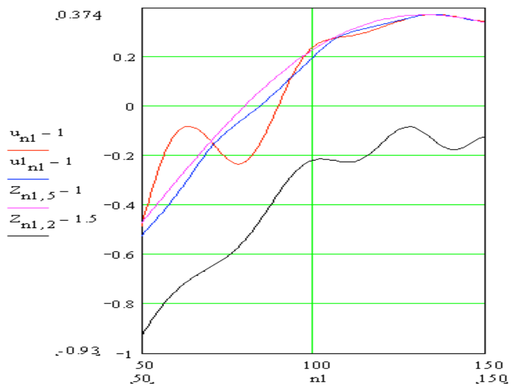

Fig. 11. The **u** causal parameter (pink trace) being tracked using algorithm-3 (red trace) and the new algorithm (blue trace). The black trace is the node output trajectory.

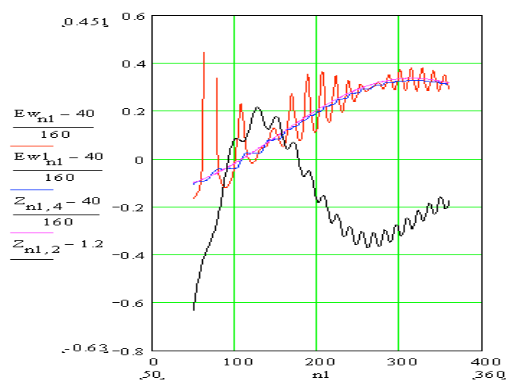

Fig. 12. The □ causal parameter (pink trace) being tracked using algorithm-3 (red trace) and the new algorithm (blue trace). The black trace is the node output trajectory.

Conclusions and Future Work

A model for the recurrent nodes of hybrid logic nets was put forward to model the complex behaviour of intelligent systems. To abduct the values of the causal parameters a number of parameter tracking algorithms were presented. Two of the algorithms used multiple points from the trajectory, 3 for the first algorithm and 2 points for the second. Two single point algorithms were presented: one that assumed constant parameters and used higher time derivatives of the trajectory (up to 4^{th}), and a second algorithm that used additional information from a 5^{th} time derivative of the trajectory to allow one of the parameters, the input parameter u, to have a non zero first time derivative.

The fourth algorithm was tested for its ability to track the input parameter for a reduced order model. The test involved the generation of a lightly damped second order recurrent net. The results showed the algorithm maintaining good tracking over an extended period of time. This algorithm proved to be far superior to the third algorithm which relied on the assumption of constant input in the derivation.

References

Al-Dabass, D.; Zreiba, A.; Evans D.; and Sivayoganathan, K. 1999a. Simulation of Three Parameter Estimation Algorithms for Pattern Recognition Architecture. *UKSIM'99, Conference Proceedings of the UK Simulation Society*, St Catharine's College, Cambridge, 7-9 April 1999, pp170-176, ISBN 0-905488-38-5, available online at URL: http://ducati.doc.ntu.ac.uk/uksim/ papers/moller/dad.doc.

Al-Dabass, D.; Zreiba, A.; Evans D.; and Sivayoganathan, K. 1999b. "Simulation of Noise Sensitivity of Parameter Estimation Algorithms, Simulation'99 Workshop, UCL, London, 29 October 1999, pp32-35.

Goodwin, C. 1997. Real Time Recursive Block Parameter Estimation of Second Order Systems. Ph.D. thesis, School of Computing & Mathematics, The Nottingham Trent University, Nottingham.

Al-Dabass, D.; Evans D.; and Sivayoganathan, K. 2002. Derivative Abduction using a Recurrent Network Architecture for Parameter Tracking Algorithms. IEEE 2002 Joint Int. Conference on Neural networks, World Congress on Computational Intelligence, pp1570-1575, May 12-17, Hawaii.

Kailath, T. 1978. Lectures on Linear Least-Squares Estimation. CISM courses and lectures No. 140, Spring-Verlag, New York.

Gersch, W. 1974. Least Squares Estimates of Structural System Parameters using Covariance Function Data. IEEE Trans. On auto. Control, 19(6).

Man, Z. 1995. Parameter-Estimation of Continuous Linear Systems using Functional Approximation. Computers and Electrical Eng. Vol. 21, No. 3, pp. 183-187.

Cawley, P. 1984. The reduction of Bias Error in Transfer Function Estimates using FFT-based Analysers. Journal of Vibration, Acoustics, Stress and Reliability in Design, pp.29-35.

Dewolf, D.; and Wiberg, D. 1993. An Ordinary Differential-Equation Technique for Continuous Time Parameter Estimation. IEEE Trans. On Auto. Control, Vol. 38, No. 4, PP. 514-528.

Kalman, R. 1960. A New Approach to Linear Filtering and Prediction Problems. Tans. Of SIAM: Journal of Basic Eng., series D, 82, PP. 35-45.

Mathcad 7 Professional Program.

Zreiba, A. 1999. MathCad Programs for Parameter Estimation. Research Report, School of Computing & Mathematics, The Nottingham Trent University, Nottingham.