THE STATE SPACE BOUNDED DERIVATIVE NETWORK SUPERCEDING THE APPLICATION OF NEURAL NETWORKS IN CONTROL

P. Turner, J Guiver

Aspentech UK Ltd, Cambridge UK
paul.turner@aspentech.com

Keywords: Nonlinear Control; Neural Networks; Polymers

Abstract

This paper introduces a challenge to the general acceptance of neural networks being 'ideally suited' for use in nonlinear control schemes. The paper briefly outlines 10 significant reasons as to why neural networks should not be used in any control system that directly affects process plant. The State Space Bounded Derivative Network will then be presented as a universal approximating architecture that encompasses the power of approximation of neural networks but without the failings. This algorithm has now been widely applied to the industrial control of polymer plants worldwide and has been the key enabling technology for Aspen Apollo™ – the Worlds' first commercial truly universal model based controller. The unique features of the SSBDN include globally guaranteed invertibility; global constraints on the model gains; robust, elegant and intelligent extrapolation capability and the capability of modelling both positional and directionally dependent dynamic nonlinearities. A commercial application of this technology to an industrial polyethylene unit will be given.

1 Introduction

The suggestion that neural networks are unsuitable architectures for control models is a controversial message. There appears to be a plethora of academic citations applauding the successful application of neural networks in control (Martin *et al*, [3]; Neuroth *et al*, [4]). This paper examines the properties of neural networks and prescribes 10 reasons why neural networks should not be implemented on any control system that directly affects process plant. Although provocative, this is an essential debate that is necessary in order to maintain the integrity of process control systems. As a solution to the issues cited in this paper, an alternative algorithm (The Bounded Derivative Network) will be presented as a universal approximating architecture that truly is well suited for nonlinear control applications. This paper raises the question as to why neural networks are used at all in control systems when it appears at best they are inert (i.e. they are short-circuited or heavily suppressed in order to protect the plant) or at worst they degrade controller performance with costly and potentially catastrophic consequences.

2 Ten Reasons Why Neural Networks Should Never Be Used In Manufacturing Control Schemes

Turner *et al* [5],[9] have recently published a damning indictment of neural networks in control. They catalogue 10 significant reasons why neural networks should never be used in any manufacturing control scheme that directly affects process plant behavior. The 10 reasons can be summarised as follows:

1) Neural networks intrinsically contain regions of zero gain within their model architecture. Zero model gains result in infinite controller gains. Since this feature is intrinsic in the architecture, no training algorithm will ever address or remove this problem.

2) Neural network models are highly susceptible to model gain inversion. Algorithms such as that described by Hartman *et al* [2] attempt to address this problem, but the approach only checks for gain inversions for an extremely small sample of the input space and in no way globally guarantees against this effect. Model gain inversion will result in valves closing when they should be opening (and vice versa) with the associated impact of this on the plant.

3) Neural networks cannot extrapolate. The ability to extrapolate is a fundamental requirement of a model based control model since extrapolative assumptions are always made about disturbance behaviour over a prediction horizon. Real industrial data is never uniformly distributed and will contain many regions of data sparsity where the neural models are invalid.

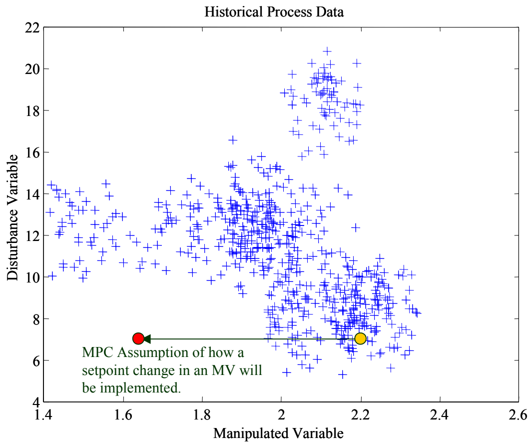

Figure 1 : Why an MPC model HAS to be able to extrapolate into sparse data regions

4) Neural networks are designed as input/output predictors or in other words – pattern recognizers. They are not designed to give a hi-fidelity inference of underlying process behaviour such as process gains. Simply taking the derivative of a neural network model will not give an accurate representation of the derivative of the process. The derivative of a neural network is typically a distribution function which represents the distribution of the training data rather than some fundamental representation of process behavior. Figure 2 displays the typical derivative of a cross validated neural model trained on real industrial data until the error on a testing set reached a minimum. No sensible engineer would suggest that this relationship is an accurate representation of process gain.

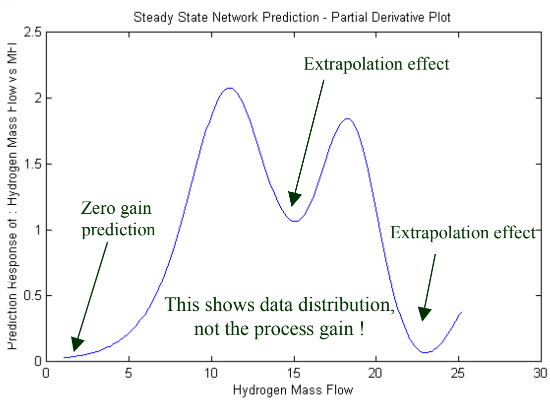

Figure 2 : The Derivative Of A Neural Network Model

5) Neural Networks are not invertible. They are not invertible for many reasons. Their lack of monotonicity, zero gain regions and their propensity to give correlated gain predictions are a few restrictions to invertibility. Correlated gain predictions occur when the model (possibly due to extrapolation) predicts that the gains between a set of MVs and an equal number of CVs are correlated. When this matrix is inverted (say in a gain scheduling scenario) the resultant move sizes calculated are huge (if not infinite). Hard coded gain limit checking will not protect the controller or the plant from this phenomenon as this is a result of gain correlation and not a result of the gain exceeding hard limits.

6) All neural networks commissioned on industrial processes have to be suppressed or short circuited. Martin *et al* [3] describe one such method where the neural model is weighted in combination with an alternative model (where the alternative model takes over in regions of extrapolation). In addition, hard limits are placed on the neural controller gain prediction. This level of suppression should raise questions. You wouldn't feel safe getting into a car if the driver was subject to such protection mechanisms (due to his/her volatility during unusual driving conditions). Why should we accept this with technology ?

7) Industry appears to have a different level of tolerance for hardware and software. If say a control valve had the same characteristics as a neural network (i.e. the valve was intrinsically designed to randomly stick (zero gain) and unpredictably suffer valve failure mode inversion when you least expect it) and in addition had to be implemented in series and parallel (to suppress and short circuit) with additional standard control valves in order that the plant could be protected - the idea would be so ridiculous it would be instantly derided. Why should we have different standards for a software equivalent ?

8) Because there are no global constraints on a neural network, copious amounts of data (and possibly expensive plant tests) are required to train them in order to attempt to extend the envelope of validity of the model. This is expensive and time consuming – and still creates no global guarantees. This process is required because of a fundamental inadequacy of the modelling paradigm. It is not an essential requirement for building a nonlinear model. This is equivalent to buying a large net containing millions of holes to use as a water receptacle and then paying large amounts of money to fill in a small number of the holes in the net. Why not buy a receptacle without any holes ?

9) Because neural networks cannot extrapolate they have to be continually re-trained in response to the inevitable process changes that occur over time. This either involves waiting for months until enough new data is available or performing more expensive plant tests. Unlike linear models (or the Bounded Derivative Network) neural models are normally completely invalid in regions of extrapolation and so loss of even rudimentary control occurs as a result of process change. This makes neural networks excessively expensive and time consuming to maintain.

10) The final point in this indictment of neural network based controllers is the fundamental reason. Why use a technology in a controller that is fundamentally not appropriate for the application? It may be possible to get these models 'on-line' by wrapping them with sophisticated bypass and suppression mechanisms but the question has to be asked as to why bother ? What is

the motivation ? If the reason is that there is no current alternative then the following description of the Bounded Derivative network should make interesting reading.

3 The State Space Bounded Derivative Network

The Bounded Derivative Network represents a groundbreaking step forward in nonlinear modelling technology. This universal approximating architecture is ideally suited to nonlinear control for the following reasons:

1) Universal approximating architecture
2) Architecturally designed to capture both input/output patterns and underlying relationships (process gains)
3) Guaranteed global invertibility
4) Globally guaranteed constraints on the model gains
5) Robust, elegant and intelligent extrapolation capability
6) Capability of modelling both positional and directional dependent dynamic nonlinearities
7) Significantly reduced training data requirements due to global envelope of extrapolation capability.

The Bounded Derivative Network (BDN) is essentially the analytical integral of a neural network. The example cited in this paper is the analytical integral of a hyperbolic tangent based neural model. Figure 3 shows the general model architecture.

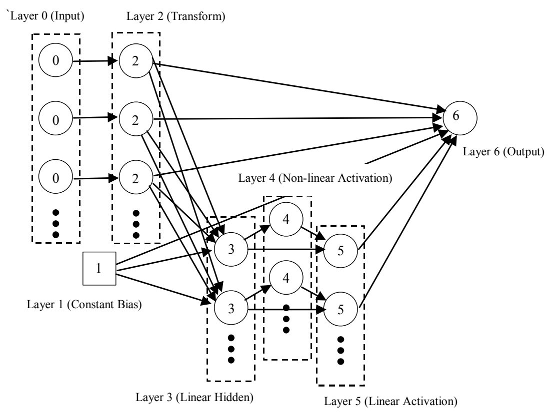

Figure 3 : Bounded Derivative Network Architecture

The activation function in the 'nonlinear activation' layer is given in equation (1).

$$f(\cdot) = \lambda_1 \log(\cosh(\beta_0 + \sum_{i=1}^{n_i} \beta_i x_i)) + \lambda_2 (\beta_0 + \sum_{i=1}^{n_i} \beta_i x_i)) \qquad (1)$$

Layer 2 in figure 1 is purely for architectural reasons and is not strictly necessary. Layers 3 and 5 contain summation processing elements only. The general equation for the BDN described in figure (3) is as follows:

$$y = w_{11}^{(6,1)} + \sum_i w_{1i}^{(6,2)} w_{ii}^{(2,0)} x_i + \sum_j w_{1j}^{(6,5)} \left(w_{jj}^{(5,4)} \left(\begin{array}{l} \log\left(\cosh\left(w_{j1}^{(3,1)} + \sum_i w_{ji}^{(3,2)} \left(w_{ii}^{(2,0)} x_i \right) \right) \right) \\ + w_{jj}^{(5,3)} \left(w_{j1}^{(3,1)} + \sum_i w_{ji}^{(3,2)} \left(w_{ii}^{(2,0)} x_i \right) \right) \end{array} \right) \right) \qquad (2)$$

Weights between nodes in each layer are notated as $w_{ij}^{(p,q)}$ which represents the connection weight from the jth node in the qth layer to the ith node in the pth layer ($q<p$). The derivative of the BDN model described in (2) with respect to one of the input variables can be calculated as:

$$\frac{\partial y}{\partial x_k} = w_{kk}^{(2,0)} \times \left(\begin{array}{l} w_{1k}^{(6,2)} + \\ \sum_j w_{1j}^{(6,5)} w_{jk}^{(3,2)} \left(\begin{array}{l} w_{jj}^{(5,3)} + \\ w_{jj}^{(5,4)} \tanh\left(\begin{array}{l} w_{j1}^{(3,1)} + \\ \sum_i w_{ji}^{(3,2)} w_{ii}^{(2,0)} x_i \end{array} \right) \end{array} \right) \end{array} \right) \qquad (3)$$

One can note that equation (3) is the general equation of a standard neural network. Since equation (3) is bounded, it is possible to calculate these bounds and constrain them during model identification. This provides a global constraint on the model derivatives and also enables robust, elegant and intelligent extrapolation. The extrapolation is robust because it is guaranteed to be linear and bounded; it is elegant because the extrapolation is analytical rather than hard coded and it is intelligent because the gradient of extrapolation is based on the derivative of the process at the point of extrapolation. Equation (4) displays a method of calculating theoretical bounds on the model derivatives. These bounds are guaranteed to encompass the actual bounds but may not be exactly equal to them. This is sufficient information to constrain the model to the correct solution.

$$\frac{\partial y}{\partial x_k}_{bound(1)} = w_{kk}^{(2,0)} \left(\begin{array}{l} \sum_j w_{1j}^{(6,5)} w_{jk}^{(3,2)} w_{jj}^{(5,3)} \\ - \sum_j \left| w_{1j}^{(6,5)} w_{jk}^{(3,2)} w_{jj}^{(5,4)} \right| \\ + w_{1k}^{(6,2)} \end{array} \right) \qquad (4)$$

$$\frac{\partial y}{\partial x_{k\ bound(2)}} = w_{kk}^{(2,0)} \left(\begin{array}{l} \sum_j w_{1j}^{(6,5)} w_{jk}^{(3,2)} w_{jj}^{(5,3)} \\ + \sum_j \left| w_{1j}^{(6,5)} w_{jk}^{(3,2)} w_{jj}^{(5,4)} \right| \\ + w_{1k}^{(6,2)} \end{array} \right) \quad (5)$$

If $w_{kk}^{(2,0)}$ is positive (as in this case), then $\frac{\partial y}{\partial x_{k\ bound(1)}}$ is a minimum theoretical gain and $\frac{\partial y}{\partial x_{k\ bound(2)}}$ is a maximum theoretical gain. If $w_{kk}^{(2,0)}$ is negative then the situation is reversed.

These are globally guaranteed limits on the input/output gain. This results in a smooth and elegant transition to a linear interpolation (constant gain) in regions of extrapolation. Figure 4 displays the bounded derivative activation function and demonstrates the 'elegant' linear extrapolation.

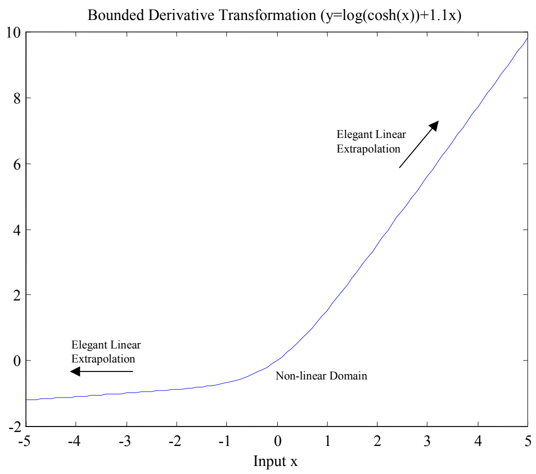

Figure 4 : Bounded Derivative Activation Function (Elegant Linear Extrapolation)

The Bounded Derivative network can be easily extended to a state space dynamic model form by presenting state vectors at the input to the model and including the steady state gains of each state vector within the gain constraint formulation. This patent pending technology forms the kernel of the Aspen Apollo controller.

Applying the steady state Bounded Derivative Network model to real industrial polypropylene data clearly demonstrates the substantial advantages of this approach compared to neural networks. Figures 5 and 6 show the model fit and the corresponding gain curve. The gain curves can be seen to be smooth and credible – in stark contrast to the 'peak and trough' nature of a neural net derivative.

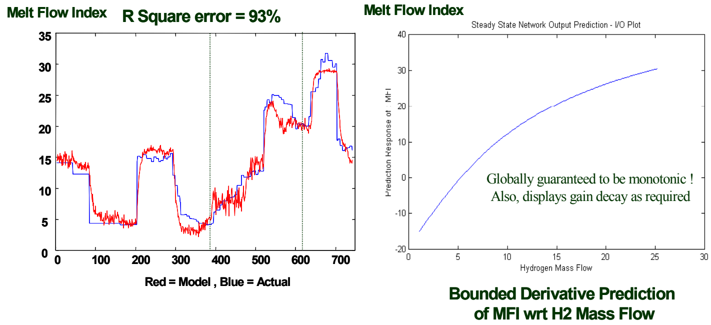

Figure 5 : Bounded Derivative Network on Polypropylene Data

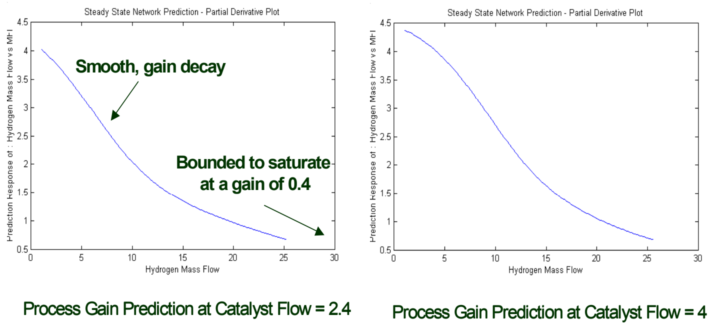

Figure 6 : Bounded Derivative Network on Polypropylene Data – Gain Analysis

4 Aspen Apollo™

Aspen Apollo is the World's only truly universal, nonlinear model based controller. The nonlinear model is fully utilized at every control execution. Aspen Apollo™ utilizes the State Space Bounded Derivative Network technology and has now been commissioned on a world class polyethylene production facility in Gelsenkirchen, Germany. Figure 7 displays simulated transitions on this unit for a product quality grade change where the controller can be clearly seen to exploit the directionally dependent nonlinearities. With a single set of tuning parameters, the controller minimizes transition speeds in both the upwards and downwards transition. The downwards transition is some 50% faster than the upwards transition (this is because it is faster to empty the reactor of reactants than to build them up). This capability is unique to Aspen Apollo™. Normally, model based controllers have to be tuned to the worst case scenario.

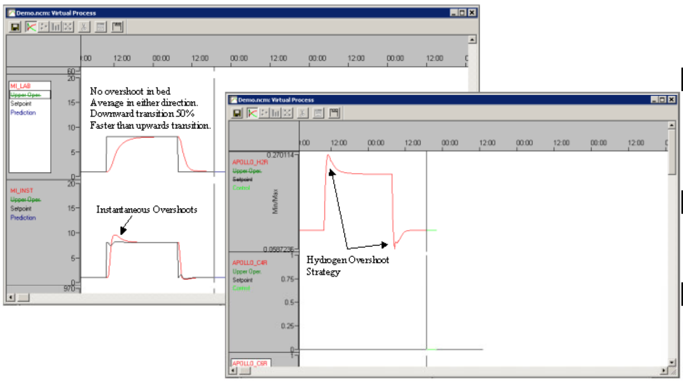

Figure 7 : Aspen Apollo Directional Dependent Dynamic Handling Capability

The response from SABIC Europe who run this particular plant is as follows:

"*Aspen Apollo has now been implemented on our Polyethylene production facility at Gelsenkirschen and has successfully been controlling product quality round the clock. Aspentech's nonlinear production control solution has delivered substantial measurable benefits to SABIC with faster transitions, increased throughput and reduced off specification material*"

Jan Versteeg, IT Manager, SABIC Europe

5 Conclusions

This paper has outlined the significant deficiencies of neural networks in control. It has also proposed an alternative algorithm (The Bounded Derivative Network) that is well suited to supercede the application of neural networks in control. This algorithm has been widely applied in over 30 nonlinear polymer production control applications worldwide. It has in addition now been incorporated into Aspen Apollo™ and successfully commissioned at a world class polyethylene production facility in Gelsenkirchen, Germany.

References

[1] Fache, A., Dubois, O., Billat, A., "On the invertibility of the RBF model in a predictive control strategy", *Proceedings of the European Symposium on Artificial Neural Networks (ESANN)*, Belgium, 21-23 April 1999, pp. 381-386, (1999).

[2] Hartman, E., " Training Feedforward Neural Networks with Gain Constraints", *Neural Computation,* Vol 12, pp. 811-829, (2000).

[3] Martin *et al*, "Method and Apparatus for Dynamic and Steady State Modeling Over a Desired Path Between Two End Points", United States Patent Number 5,933,345, (1999)

[4] Neuroth, M., MacConnell, P., Stronach, F., Vamplew, P., "Improved modeling and control of oil and gas transport facility operations using artificial intelligence.", *Knowledge Based Systems*, Vol 13, no. 2, pp. 81-92, (April 2000)

[5] Turner, P, Guiver, J, "Neural Networks – A comprehensively unsuitable technology for control", *Proceedings of Aspenworld 2002*, Washington DC, (Oct 2002)

[6] Turner, P., Guiver, J., "Neural Network APC: Fact or fantasy?", *Control Solutions Magazine*, "Optimizing Your Process" supplement, pp. 16-20, (June 2001)

[7] Turner, P., Guiver, J., Lines, B., Keenan, M., "On the commercial application of next generation nonlinear model based predictive control", Session Applications of Model Predictive Control, paper 144f, AIChE Spring Meeting, New Orleans, LA, Mar 30th – Apr 3rd 2003.

[8] Turner, P., Guiver, J., Lines, B., "Introducing The State Space Bounded Derivative Network for Commercial Transition Control", Session Transition Control, paper FP18-2, 2003, Americal Automatic Control Conference, Denver, CO, June 4-6, 2003.

[9] Turner, P., "Automated Product Grade Transitions: Exposing the inherent and latent dangers of neural networks in manufacturing Process Control", EUnite 2003, Safety Critical Task Force Meeting, Session AC3, Oulu, Finland, 10th-11th July, 2003