

Stability of Neural Net Based Model Predictive Control

John W. Eaton and James B. Rawlings[†]
Department of Chemical Engineering
The University of Texas at Austin
Austin, Texas 78712

Lyle H. Ungar
Department of Chemical Engineering
University of Pennsylvania
Philadelphia, Pennsylvania 19104

1. Introduction

The use of artificial neural networks in model predictive control has recently attracted a great deal of attention, primarily because neural networks appear to provide a convenient means for modelling complicated nonlinear processes at low cost. Numerous journal articles and meeting papers have appeared on the use of neural network models as the basis for model predictive controllers with finite prediction horizons, including Bhat and McAvoy [1], Cooper *et al.* [2], Hernandez and Arkun [5], Lee and Park [6], Pottman and Seborg [11], Psychogios and Ungar [12], Saint-Donat *et al.* [13], and Willis *et al.* [14]. Most of these publications concentrate on the issues related to constructing neural network models. Very little attention is given to issues of stability or closed-loop performance, although these are still open and unresolved issues.

Neural network models are often promoted as an ideal modelling tool for complex nonlinear processes because they are able to match the input-output behavior of any continuous nonlinear system. It is often suggested that these network models may be easily obtained from historical plant data, thus solving the problems associated with generating nonlinear process models. However, some care must be taken in constructing these "unstructured" models. In addition to the standard problems of gathering enough informative data to model reliably a complex nonlinear process, simple feedforward networks used as one-step-ahead predictors can give arbitrarily bad predictions when used iteratively over a future time horizon. Moreover, networks that attempt to predict simultaneously outputs at different future times are not guaranteed to be causal.

Although it is true that neural networks may provide a convenient way to construct nonlinear models from process data, it would be a mistake to believe that they can provide a generic solution for all complex modelling problems, or that they can always be constructed from whatever data happen to be on hand. For example, historical data from an industrial chemical process that has been collected in the course of plant operation is typi-

cally obtained while the system, or part of the system, is running under some sort of closed-loop feedback control. Use of such data can lead to the identification of the inverse of the controller transfer function instead of the transfer function of the process. In practice, many process variables are also highly correlated, which can result in empirical models that are unable to provide accurate predictions even though they are able to match the process data. The reader is referred to MacGregor *et al.* [7] for an illuminating discussion of these issues.

This paper illustrates the stability problems associated with the use of finite horizon model predictive controllers by using a recurrent neural network to predict accurately the input-output response of a simple linear system that exhibits nonminimum-phase behavior. The stability of the resulting closed-loop system depends on the particular tuning parameters of the controller (horizon length, penalty weights, etc.) even in the absence of process-model mismatch.

Although these problems are not unique to neural net models, or even finite-horizon model predictive controllers, most modern control theories (LQG, Q-parameterization, etc.) have addressed the nominal stability problems encountered with controllers based on other forms of input-output models. Similarly, it is desirable to establish a framework for model predictive control using neural network models that does not suffer from nominal stability problems. In this paper, we propose a state-space description of a class of externally recurrent neural network models. This state-space description allows the application of recent theoretical results for nonlinear systems to ensure nominal stability for the resulting closed-loop system.

2. The Controller

This discussion will only consider the use of neural network models in a finite horizon model predictive control framework.

The steps in the model predictive controller algorithm may be summarized as follows:

0. Select a model of the process to be controlled.
1. Using the process model to predict behavior over a future time horizon, determine the optimal control

[†]To whom all correspondence should be addressed.
E-mail: jbraw@che.utexas.edu.

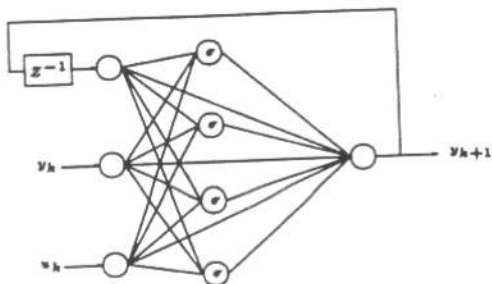


Figure 1: An externally recurrent network.

action by minimizing a performance index over the horizon.

2. Implement a portion of the optimal input trajectory and obtain a measurement.
3. Update the minimization (for example, by obtaining a new estimate of the model parameters, or by estimating a disturbance to the process, etc.).
4. Go to step 1.

Even when the process model is constructed to avoid problems with causality, and sufficiently exciting data is used for the estimation of model parameters, the resulting finite-horizon model predictive controller is not guaranteed to be nominally stabilizing. This problem arises because of the finite horizon of the model prediction.

3. The Model

Many different forms of nonlinear models are referred to generically as neural network models. The hype surrounding neural computing and the use of the generic term can easily lead the novice to believe that any neural network model will do. This is not the case, and some care must be taken to select an appropriate model structure for the specific task at hand.

For example, not all neural network models are appropriate for modelling dynamic systems. For example, a single fully connected feedforward network in which the outputs of the network are interpreted as the sequence of future outputs produced by a dynamic system may result in a noncausal model. This can happen even if the data used to "train" the network is taken from a causal system, because the weights on the network connections from future inputs to past outputs do not generally become identically zero. Although these effects may be small if the model does a reasonable job of predicting the system behavior, it is best to construct the network so as to avoid such problems altogether.

For the examples that follow, we have selected an externally recurrent network model with the structure shown in Figure 1. The equations describing the network used in this model are

$$y_{net} = W_2 \sigma(W_1 u_{net}) + W_3 u_{net} \quad (1)$$

where the vector of network inputs, u_{net} is composed of current and past system inputs and outputs

$$u_{net} = [y_k, y_{k-1}, \dots, y_{k-M+1} | u_k, u_{k-1}, \dots, u_{k-P+1}]^T.$$

For a single input single output system, the vector u_{net} includes the past M system outputs and the past P system inputs that are required to produce a predicted output at the next sampling time.

To form a prediction over a future time horizon, a set of known system inputs and outputs is used to produce a value for the output at the next sampling time. This value is then used as an input for the network at the next sampling time.

The input and output layers in this network are linear, and the function $\sigma(\cdot)$ used in the single hidden layer is the sigmoidal activation function

$$\sigma(x) = 2/(1 + \exp(-x/a)) - 1.$$

Using this structure, the model is guaranteed to be causal (there are no connections from future inputs to past outputs) and it has the property that if all inputs to the network are zero, the output will also be zero (the bias term is zero).

The direct connection of the linear input and output layers makes it possible for the network to represent linear systems exactly and is similar to the network architecture proposed by Haesloop and Holt [4]. The direct linear connection is important for the examples discussed below, because even the nonlinear system that we examine includes linear terms.

4. Examples

The following examples were selected because they are known to cause difficulties for finite horizon model predictive controllers when using other model representations even in the absence of process-model mismatch.

Gill and Murray's NPSOL [3] was used to find the best set of network parameters to fit sample data which included 2% normally distributed noise. Since our goal was to obtain network models that closely approximate the behavior of the actual systems, we selected the number of network inputs and outputs and the initial conditions for the optimization based on our knowledge of the system that generated the data.

4.1. Linear Example

Consider the system

$$G(s) = \frac{s-1}{(s+1)^2}$$

The right half plane zero requires that the finite horizon model predictive controller be tuned to achieve closed-loop stability for this system. Although this may be acceptable to practitioners, it is generally desirable to

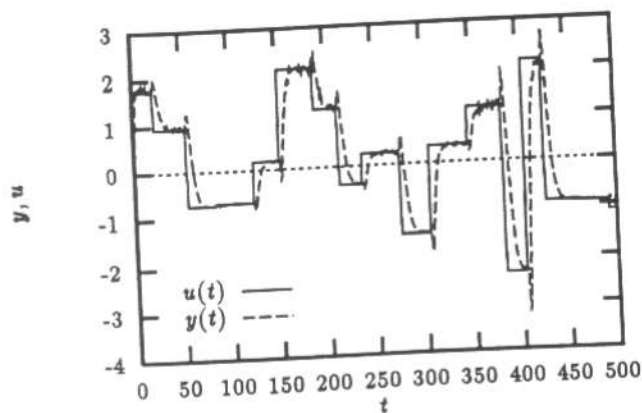


Figure 2: Sample data used for estimation of network parameters.

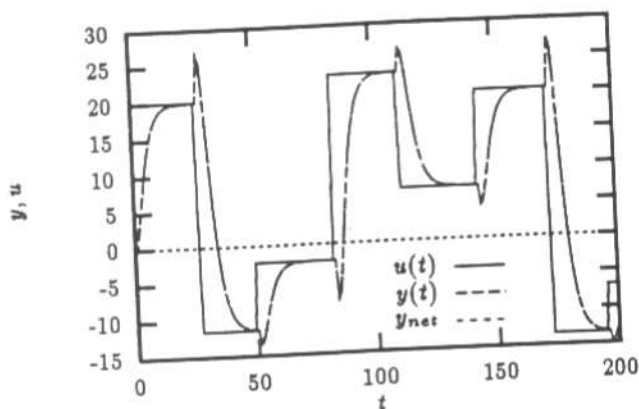


Figure 3: Network approximation of the linear system $G(s) = (s-1)/(s+1)^2$, superimposed over data for $G(s)$

guarantee nominal stability by design, and allow tuning to influence performance only.

For this system, the network inputs are the system inputs and outputs at the two previous sampling times, u_{k-1} , u_k , y_{k-1} , and y_k . The network output is the system output at the next sampling time, y_{k+1} . Three hidden nodes were used, though they are not actually needed in this case, since the system can be represented exactly with only the linear term of Equation 1 (though this would not generally be known).

Figure 2 shows a sample of the data used to determine parameters for the network model. Figure 3 shows the behavior of the network for a new set of inputs, superimposed over the exact value computed from the system equation, $G(s)$. Note that the two curves are indistinguishable, and the approximation is good over a much wider range than the range of data used for estimation of the network parameters. This is not generally true of neural network models, but is true in this particular case because the linear portion of the network model given by equation 1 is very accurate, and the contribution of the nonlinear portion is very small.

The response of the closed-loop system to a step

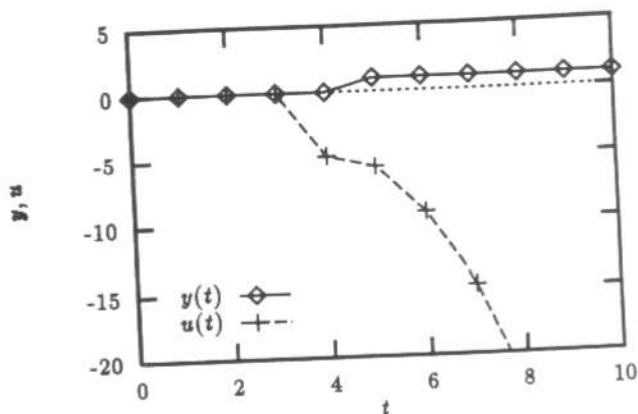


Figure 4: Closed-loop response for step change in the reference.

change in the setpoint is shown in Figure 4. The prediction and control horizons are both 10 sampling times and there is a small penalty on the input. The system is able to match the setpoint exactly (we have used the horizon to allow the controller to know the setpoint change in advance), but only by using an unstable input. For this simulation, model error is eliminated by using the network model to represent the plant. Even in the absence of modelling error, the finite horizon model predictive controller can result in unstable closed-loop systems.

One approach to avoiding this problem is to rewrite the neural network model in the state-space form

$$x_{k+1} = f(x_k, u_k)$$

$$y_k = g(x_k).$$

Then it is possible to ensure a stabilizing controller by enforcing a state constraint $x = 0$ at the end of the prediction horizon rather than tuning the controller to achieve stability. The reader is referred to the recent work of Mayne and Michalska [8, 10] and Meadows *et al.* [9] for details of the the stability issues for model predictive controllers using final time state constraints.

The only hurdle to implementing the final time state constraint when using a neural network based model predictive controller is determining an appropriate state vector for the model. One simple approach is to use

$$x_k = [y_k, y_{k-1}, \dots, y_{k-M+1} | u_{k-1}, \dots, u_{k-P+1}]^T.$$

With this choice, the state of the system is the same as the set of inputs to the network model, u_{net} except that it does not contain the current input u_k , and t

function $f(x_k, u_k)$ may be written as

$$x_{k+1} = \begin{bmatrix} \mathcal{N}(x_k, u_k) & y_k \\ & y_{k-1} \\ & \vdots \\ & y_{k-M+1} \\ & u_{k-1} \\ & u_{k-2} \\ & \vdots \\ & u_{k-P+1} \\ & u_k \end{bmatrix}$$

where $\mathcal{N}(x_k, u_k)$ is the network model given by Equation 1 and the function $g(x_k)$ simply selects the first element of the state vector.

Note that in order to zero the state of this system at the end of the prediction horizon, it is necessary to zero $P-1$ consecutive system inputs and M consecutive system outputs, placing a significant restriction on the minimum length of the prediction horizon.

4.2. Nonlinear Example

The nonlinear system

$$\begin{bmatrix} y_1 \\ y_2 \end{bmatrix}_{k+1} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \end{bmatrix}_k + \begin{bmatrix} u_k \\ u_k^3 \end{bmatrix} \quad (2)$$

has no continuous asymptotically stabilizing feedback law $u_k = \mu(y_k)$ [9]. We wish to construct a neural network model using input-output data taken from this system in order to see if a discontinuous feedback law is produced by applying the model predictive controller to the neural network model.

For this system, the network inputs are the system inputs and outputs at the previous sampling time, u_k , and y_k . The network output is the system output at the next sampling time, y_{k+1} . For this network, five hidden nodes were used.

Figure 5 shows a sample of the data used for estimation of the network parameters. Figure 6 shows the network prediction for a number of different overlapping horizons, each 10 sampling times in length. The network is able to predict the system behavior reasonably well over the region covered by the data used for estimation, but outside that region, the prediction becomes significantly worse.

Figure 7 shows the feedback law as a function of the initial condition as y_1 and y_2 are allowed to vary around the unit circle. The controller has a prediction horizon of 10, with the constraint $x = 0$ enforced at the end of the horizon. Notice that there are large jumps in the value of the manipulated variable, u at certain state values. Such discontinuities arise naturally from the solution of a sequence of optimal control problems, but may result in behavior that is counterintuitive.

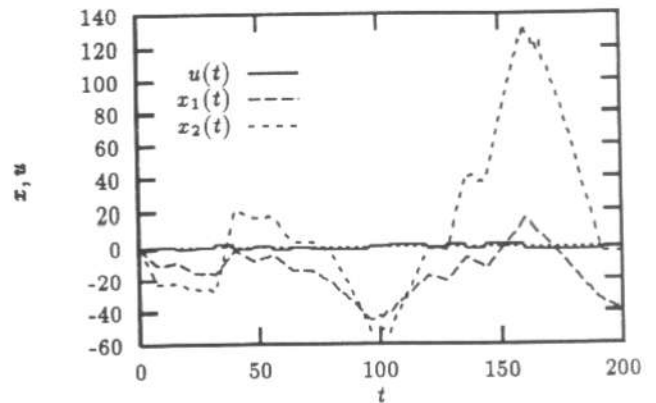


Figure 5: Sample data used for estimation of network parameters.

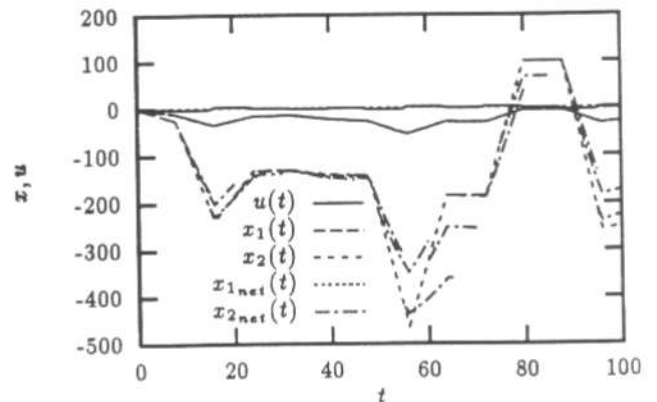


Figure 6: Network approximation of nonlinear system given by equation 2.

Figure 8 shows what appears to be the global minimum of the controller calculation as the initial conditions for y_1 and y_2 are allowed to vary around the unit circle. For this example, the objective function is continuous, though it is not required, and for some systems, the model predictive controller will produce feedback laws that are discontinuous in both the state and the objective function.

Figure 9 shows the behavior of the closed-loop system for two initial conditions on either side of the discontinuity in the feedback map near $\theta = 1.93$. Note that even though the initial conditions are very similar, the model predictive controller finds much different solutions for the two cases.

5. Conclusions

Finite horizon model predictive controllers based on neural network models may result in closed-loop unstable systems, and can generate discontinuous feedback laws. By reformulating the neural network model in a state-space framework, it is possible to ensure stability of the neural network based model predictive controller, but not without placing some restriction on the minimum

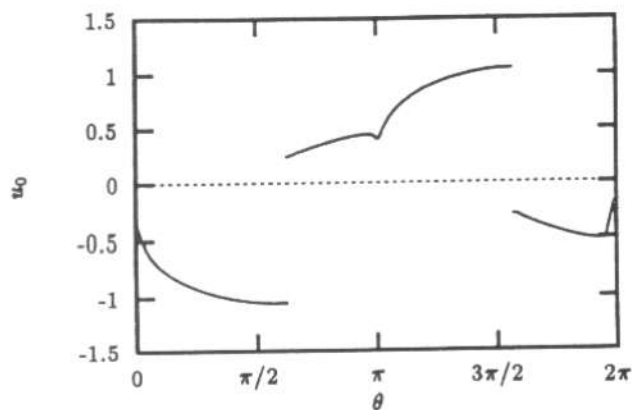


Figure 7: Optimal feedback versus state along unit circle.

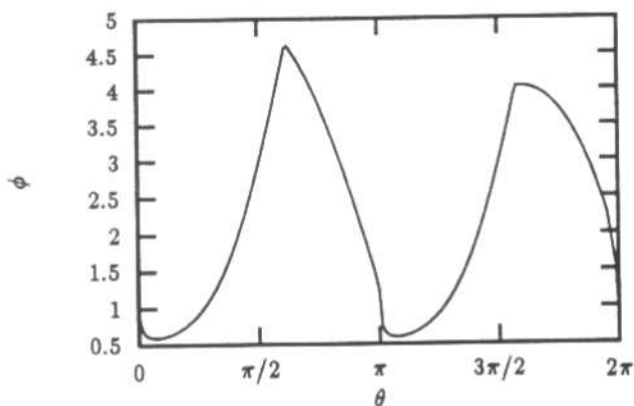


Figure 8: Objective function computed by model predictive controller for first move versus state along unit circle.

prediction horizon.

References

- [1] Bhat, N. and T. McAvoy. Use of neural nets for dynamic modeling and control of chemical processes. *Comput. Chem. Eng.*, 14:573-583, 1990.
- [2] Cooper, D. J., L. Megan, and J. Ralph F. Hinde. Comparing two neural networks for pattern based adaptive process control. *AIChE J.*, 38(1):41-55, 1992.
- [3] Gill, P. E., W. Murray, M. A. Saunders, and M. H. Wright. User's guide for SOL/NPSOL (Version 4.0): A Fortran package for nonlinear programming, technical report SOL 86-2. Technical report, Systems Optimization Laboratory, Department of Operations Research, Stanford University, 1986.
- [4] Haesloop, D. and B. R. Holt. A neural network structure for system identification. In *Proceedings of the 1990 American Control Conference*, pages 2460-2465, 1990.
- [5] Hernandez, E. and Y. Arkun. Neural network modeling and an extended dmc algorithm to control nonlinear systems. In *Proceedings of the 1990*

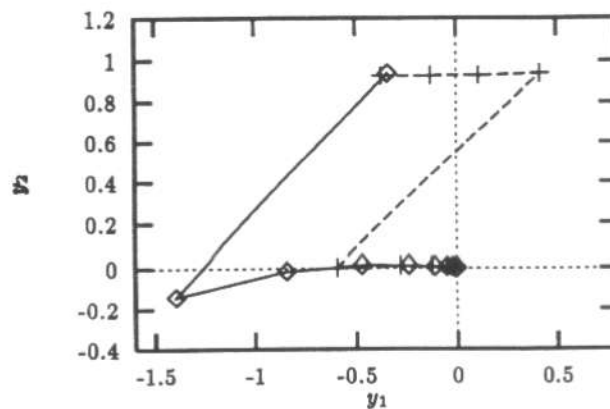


Figure 9: Closed-loop behavior from two initial conditions.

American Control Conference, pages 2454-2459, 1990.

- [6] Lee, M. and S. Park. A new scheme combining neural feedforward control with model-predictive control. *AIChE J.*, 3(2):193-200, 1992.
- [7] MacGregor, J. F., T. E. Marlin, and J. V. Kresta. Some comments on neural networks and other empirical modelling methods. In Arkun, Y. and W. H. Ray, editors, *Chemical Process Control—CPC IV, Proceedings of the Fourth International Conference on Chemical Process Control*, pages 665-672. CACHE, 1991.
- [8] Mayne, D. Q. and H. Michalska. Receding horizon control of nonlinear systems. *IEEE Trans. Auto. Cont.*, 35(7):814-824, July 1990.
- [9] Meadows, E. S., M. S. Henson, J. W. Eaton, and J. B. Rawlings. Receding horizon control and discontinuous state feedback stabilization. Accepted for publication in *Int. J. Control*, December 1993.
- [10] Michalska, H. and D. Q. Mayne. Robust receding horizon control of constrained nonlinear systems. *IEEE Trans. Auto. Cont.*, 38(11):1623-1633, November 1993.
- [11] Pottmann, M. and D. E. Seborg. A nonlinear predictive control strategy based on radial basis function networks. In *3rd IFAC Symposium on Dynamics and Control of Chemical Reactors, Distillation Columns and Batch Processes*, pages 309-314, 1992.
- [12] Psychogios, D. C. and L. H. Ungar. Direct and indirect model based control using artificial neural networks. *Ind. Eng. Chem. Res.*, 30:2564-2573, 1991.
- [13] Saint-Donat, J., N. Bhat, and T. J. McAvoy. Neural net based model predictive control. *Int. J. Control*, 54(6):1453-1468, 1991.
- [14] Willis, M. J., G. A. Montague, C. D. Massimo, M. T. Tham, and A. J. Morris. Non-linear predictive control using optimization techniques. In *Proceedings of the 1991 American Control Conference*, pages 2788-2793, 1991.