GAIN SCHEDULING CONTROL OF NONLINEAR SYSTEMS BASED ON NEURAL STATE SPACE MODELS

Jan Dimon Bendtsen * **Jakob Stoustrup** *

* *Department of Control Engineering, Aalborg University*
Fredrik Bajersvej 7C, 9220 Aalborg East, Denmark.
Email: {dimon,jakob}@control.auc.dk

Abstract: This paper presents a novel method for gain scheduling control of nonlinear systems based on extraction of local linear state space models from neural networks with direct application to robust control. A neural state space model of the system is first trained based on in- and output training samples from the system, after which linearized state space models are extracted from the neural network in a number of operating points according to a simple and computationally cheap scheme. Robust observer-based controllers can then be designed in each of these operating points, and gain scheduling control can be achieved by interpolating between each controller. In this paper, we propose to use the Youla-Jabr-Bongiorno-Kucera parameterization to achieve a smooth scheduling between the operating points with certain stability guarantees.

Keywords: Robust Gain Scheduling Control, Neural Networks, Youla Parameterization

1. INTRODUCTION

Many dynamical systems found in real-life applications behave roughly like linear systems in the vicinity of an operating point, but exhibit saturations and other forms of nonlinear behavior when the state of the system diverges from the operating point. Nonlinear control schemes such as feedback linearization and backstepping, both of which rely on explicit cancelation of nonlinearities, are often used to deal with the nonlinear control problem, but may run into difficulties in the presence of unmodeled dynamics, parameter uncertainties, etc. Another approach to control of such "reasonably-behaved" nonlinear systems that is commonly used in practice, involves linearization of the system model in an appropriate set of operating points, whereupon one or more linear controllers for the system are designed in these points. It is for instance possible to consider the linearizations to be nominal (state space) models and then carry out $\mathcal{H}_2$ or $\mathcal{H}_\infty$ robust control synthesis, assuming some kind of uncertainty model can be estimated for the control design (see e.g., Zhou et al. (1995)). The robust controller should be able to handle the discrepancies between the nominal linear model and the uncertain nonlinear system in the vicinity of the operating point. If the closed-loop system is expected to operate in several operating points, it is possible to apply *gain scheduling control*, i.e., interpolating between the controllers in different operating points as the system state moves from one operating point to another (see e.g. Shamma and Athans (1990)). In any of the aforementioned approaches, however, the success of the control scheme largely depends on the quality of the model on which the control design is based, which can sometimes be difficult to obtain in practice.

Artificial neural networks (ANN) such as multilayer perceptrons or neuro-fuzzy networks have been shown to be able to model the kind of nonlin-

ear systems described above accurately. The idea of training an ANN as a nonlinear state space model of the plant and then extracting linearized system models on which to base the controller design in given operating points thus seems tempting, but does not appear to have received much attention so far. Some work along these lines has been presented in e.g. J. A. K. Suykens (1995). The results presented tend to be conservative, however, since the plant nonlinearities modeled by the neural network are simply considered sector-bounded disturbances to a single nominal linear design, i.e., very little knowledge of the actual nonlinearities is exploited in practice. Bendtsen and Trangbaek (2002) also extracts a single nominal linear plant from a trained neural state space model, but considers the plant nonlinearities explicitly in a quasi-LPV framework, thereby removing some of the conservativeness. However, as will be pointed out in the following, it is possible to extract linearized state space models from a trained neural network in a computationally cheap and efficient manner in any number of operating points. This means that it is feasible to apply gain scheduling control based on the neural state space model in a systematic manner.

In this paper, we propose a scheme that allows us to design a number of controllers in different operating points based on several linearized models extracted from an ANN, and to interpolate between them using the Youla-Jabr-Bongiorno-Kucera (YJBK) parameterization. The latter point is important, since even if two controllers K_1 and K_2 are designed for the same plant in the same operating point, there is no automatic guarantee that a simple linear combination of the two controllers $K = \alpha K_1 + (1-\alpha)K_2$, where $\alpha \in [0;1]$ is a scheduling variable, stabilizes the plant for $0 < \alpha < 1$. By using the YJBK parameterization of all stabilizing controllers for the interpolation, however, it is possible to switch between individual (robustly) stabilizing controllers in a stable manner.

The outline of the rest of the paper is as follows. In Section 2 we describe the ANN framework and how to extract local linearizations from a trained network. After that, Section 3 gives a brief outline of how to use the YJBK parameterization to describe all stabilizing controllers, and how to use it to interpolate between individual controllers. Also, the dual YJBK parameterization of all systems stabilized by a given controller is presented. Section 4 then discusses the actual gain scheduling control method and illustrates the method on a simulation example, and finally Section 5 sums up the conclusions of the work.

2. NEURAL STATE SPACE MODELS

We consider systems of the form

$$\dot{x} = f(x,u), \quad y = Cx \tag{1}$$

where $x \in \mathbb{R}^n$ is the state vector, $u \in \mathbb{R}^m$ is a control signal and $y \in \mathbb{R}^p$ is the output vector of the system. $f(\cdot,\cdot) : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$ is an unknown smooth function of the states and control inputs describing the system dynamics.

From neural network theory (see e.g. Lu and Basar (1998)), it is known that we can approximate this function to a desired accuracy with a single hidden layer multi-layer perceptron (MLP) with q neurons (assuming q is chosen large enough):

$$\hat{f}(x,u) = W_o \sigma\left(W_x x + W_u u + W_b\right) \tag{2}$$

where $W_o \in \mathbb{R}^{n\times q}$ contains the output layer weights and $W_x \in \mathbb{R}^{q\times n}, W_u \in \mathbb{R}^{q\times m}$ contain the hidden layer weights, respectively. $\sigma(\cdot) : \mathbb{R}^q \to \mathbb{R}^q$ is a continuous, diagonal, static nonlinearity. $W_b \in \mathbb{R}^q$ contains a set of biases which will allow us to model non-odd functions with odd neuron functions such as the hyperbolic tangent.

Alternatively, generalized linear models such as neuro-fuzzy or radial basis function ANNs of the form $\hat{f}(x,u) = \theta^T \phi\left([x^T\ u^T]^T\right)$, where $\phi(\cdot) : \mathbb{R}^{n+m} \to \mathbb{R}^r$ maps the network inputs into a vector of basis functions and $\theta \in \mathbb{R}^{r\times n}$ is a weight matrix, can also be used. With obvious choices of weight/bias matrices and neuron functions, these ANNs can be written on the form (2). Refer to e.g. Brown and Harris (1994) for further information on this type of ANN model.

Assume now that an ANN has been trained (refer to e.g. Levin and Narendra (1996) or Brown and Harris (1994) for descriptions of training methods etc. for ANNs) to satisfy $\|f(x,u) - \hat{f}(x,u)\|_\infty \leq \varepsilon_{max}$, i.e., we can write

$$\dot{x} = \hat{f}(x,u) + \varepsilon_x \tag{3}$$

with $\varepsilon_x \in \mathbb{R}^n$, $\|\varepsilon_x\|_\infty \leq \varepsilon_{max}$, representing modeling errors, noise, and other uncertainties. An estimate of ε_{max} can for instance be obtained by evaluating the ANN simulation performance on a test set, i.e., comparing output samples simulated by the ANN with actual samples measured from the plant, which have not been included in the training. We can then calculate the linearization of $\hat{f}$ around any operating point $(\bar{x},\bar{u})$ as

$$\begin{aligned}\dot{\tilde{x}} &\approx \left.\frac{\partial \hat{f}}{\partial x^T}\right|_{x=\bar{x}} \tilde{x} + \left.\frac{\partial \hat{f}}{\partial u^T}\right|_{u=\bar{u}} \tilde{u} \\ &= W_o \frac{d\sigma(\xi)}{d\xi^T} W_x \tilde{x} + W_o \frac{d\sigma(\xi)}{d\xi^T} W_u \tilde{u} \\ &= \hat{A}(\bar{x},\bar{u})\tilde{x} + \hat{B}(\bar{x},\bar{u})\tilde{u}\end{aligned} \tag{4}$$

where $\tilde{x} = \bar{x} - x$ and $\tilde{u} = \bar{u} - u$ are the locally linearized state and input in the vicinity of $(\bar{x}, \bar{u})$, respectively. $d\sigma(\xi)/d\xi^T$ is the derivative of the neuron function mapping wrt. its input $\xi = W_x\bar{x} + W_u\bar{u} + W_b$ and depends on the choice of ANN. For instance, in case of an MLP with hyperbolic tangent neuron functions, we have $\sigma(\xi) = \mathrm{diag}(\tanh(\xi_i))$ and $d\sigma(\xi)/d\xi^T = I - \mathrm{diag}(\tanh(\xi_i)^2), i = 1, \ldots, q$. The linearization in (4) can be evaluated for other types of ANN as well; e.g., in case of the neuro-fuzzy model mentioned above, we would have $[\hat{A}\ \hat{B}] = \theta^T d\phi(\xi)/d\xi^T$, where $d\phi(\xi)/d\xi^T$ depends on the choice of basis functions.

Remark 1 As can be seen, the evaluation of the linearized models in the different operating points is quite cheap, computationally speaking. The stabilizability and detectability of the system modeled by the ANN can thus be evaluated in any appropriate part of the state space using eqn. (4). In the following we will assume that the system is stabilizable and detectable throughout the entire region of interest. ◁

3. CONTROLLER PARAMETERIZATION

In this Section, we will provide a brief review of the framework established in Niemann and Stoustrup (1999), on which we base the controller synthesis.

Consider a system G with the state space realization

$$\begin{bmatrix} \dot{x} \\ \hline z \\ y \end{bmatrix} = \left[\begin{array}{c|cc} A & B_w & B_u \\ \hline C_z & D_{zw} & D_{zu} \\ C_y & D_{yw} & D_{yu} \end{array}\right] \begin{bmatrix} x \\ \hline w \\ u \end{bmatrix} \tag{5}$$

where $x \in \mathbb{R}^n$ is the state vector, $y \in \mathbb{R}^{p_y}$ is the measurement vector, $u \in \mathbb{R}^{m_u}$ is the control vector, $z \in \mathbb{R}^{p_z}$ is the signal to be controlled (which may coincide with y) and $w \in \mathbb{R}^{m_w}$ is a disturbance vector containing noise and command signals. If the subsystem G_{yu} given by the matrices (A, B_u, C_y, D_{yu}) is stabilizable and detectable, G can be stabilized by an observer-based feedback controller K with the state space realization (see e.g. Zhou et al. (1995))

$$\begin{bmatrix} \dot{x}_c \\ \hline u \end{bmatrix} = \left[\begin{array}{c|c} A + B_uF + LC_y + LD_{yu}F & -L \\ \hline F & 0 \end{array}\right] \begin{bmatrix} x_c \\ \hline y \end{bmatrix} \tag{6}$$

where $x_c \in \mathbb{R}^n$ is the controller state and $A + B_uF$ and $A + LC_y$ are stable matrices. This setup is illustrated in the left part of Figure 1.

Let $G_{yu}(s) = C_y(sI - A)^{-1}B_u + D_{yu}$ be written using coprime factorization as

$$G_{yu}(s) = NM^{-1} = \tilde{M}^{-1}\tilde{N} \tag{7}$$

with $N, M, \tilde{M}, \tilde{N} \in \mathcal{R}H_\infty$. Further, let a number of controllers for G_{yu} be given by

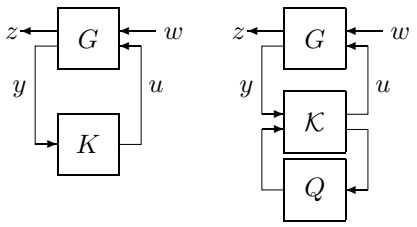

Fig. 1. *The interconnection of the system G and the controller K (left). The controller is parameterized by means of a star product between an augmented system $\mathcal{K}$ and a stable parameter system Q (right).*

$$K_i(s) = U_iV_i^{-1} = \tilde{U}_i^{-1}\tilde{V}_i,\ i = 0, \ldots, \nu \tag{8}$$

where $U_i, V_i, \tilde{U}_i, \tilde{V}_i \in \mathcal{R}H_\infty$. These coprime factorizations can be chosen to satisfy the double Bezout identity

$$\begin{aligned} \begin{bmatrix} I & 0 \\ 0 & I \end{bmatrix} &= \begin{bmatrix} \tilde{V}_i & -\tilde{U}_i \\ -\tilde{N} & \tilde{M} \end{bmatrix} \begin{bmatrix} M & U_i \\ N & V_i \end{bmatrix} \\ &= \begin{bmatrix} M & U_i \\ N & V_i \end{bmatrix} \begin{bmatrix} \tilde{V}_i & -\tilde{U}_i \\ -\tilde{N} & \tilde{M} \end{bmatrix} \end{aligned}$$

for $i = 0, \ldots, \nu$. All stabilizing controllers for G_{yu} based on K_0 can now be written according to the setup depicted in the right part of Figure 1, where the controller K is parameterized by means of a star product (cf. Zhou et al. (1995)) between an augmented system $\mathcal{K}$ and a stable parameter system Q:

$$\begin{aligned} K(Q) &= \mathcal{K} \star Q \\ &= K_0 + \tilde{V}_0^{-1}Q(I + V_0^{-1}NQ)^{-1}V_0^{-1} \end{aligned} \tag{9}$$

$\mathcal{K}$ is chosen such that for $Q = 0$ we obtain the controller K_0. We now have the following result (Moore et al. (1990)).

Theorem 1. Let a number of stabilizing controllers (8) be given for a system (7). Then $K_i, i = 0, \ldots, \nu$ can be implemented as $K(Q_i) = \mathcal{K} \star Q_i$, with $Q_i \in \mathcal{R}H_\infty$ given by

$$Q_i = \tilde{U}_iV_0 - \tilde{V}_iU_0 = \tilde{V}_i(K_i - K_0)V_0.$$

Proof: Follows by inserting $Q_i = \tilde{V}_i(K_i - K_0)V_0$ in (9), rewriting the expression as

$$K_0(Q_i) = K_0 + \tilde{V}_0^{-1}\tilde{V}_i(I + (K_i - K_0)N\tilde{V}_i)^{-1}(K_i - K_0)$$

and using the Bezout identity to show that $I + (K_i - K_0)N\tilde{V}_i = \tilde{V}_0^{-1}\tilde{V}_i$. ◁

Theorem 1 states that it is possible to implement a controller as a function of a stable parameter system Q based on another stabilizing controller. This implies that it is possible to change between two controllers online, say, from K_0 to K_i, in a smooth fashion by scaling the Q_i parameter by the factor $\alpha \in [0; 1]$. Furthermore, as pointed out

in Niemann and Stoustrup (1999) it is not only possible to change from K_0 to K_i, but indeed from any K_i to any $K_j, i, j = 0, \ldots, \nu$. In this case, we compute the parameter Q as the following linear combination of the Q_i's given in Theorem 1:

$$Q = \sum_{i=1}^{\nu} \alpha_i Q_i, \quad \sum_{i=1}^{\nu} \alpha_i = 1, \ \alpha_i \in [0; 1].$$

Then the resulting controller is given as

$$K(Q) = \left(\sum_{i=1}^{\nu} \alpha_i \tilde{V}_i \right)^{-1} \sum_{i=1}^{\nu} \alpha_i \tilde{U}_i. \qquad (10)$$

This controller will stabilize the system depicted in Figure 1, giving rise to a closed loop transfer function T_{zw} from w to z given by

$$T_{zw} = G_{zw} + G_{zu} M \left(\sum_{i=1}^{\nu} \alpha_i \tilde{U}_i \right) G_{yw} \qquad (11)$$

where G_{zw}, G_{zu} and G_{yw} represent the transfer functions of the subsystems of (5) from w to z, u to z, and w to y, respectively.

As an analogy to the parameterization presented above, the so-called dual YJBK parameterization expresses all the systems parameterized by a given controller as a function of a stable parameter. Consider the system $G_{yu,0}$ factored as in (7), and assume that the controller K_0 factored as in (8) stabilizes $G_{yu,0}$. Then all systems $G_{yu}(S)$ stabilized by K_0 can be written as

$$G_{yu,i}(S) = G_{yu,0} + \tilde{M}_0^{-1} S (I + M_0^{-1} U_0 S)^{-1} M_0^{-1} \qquad (12)$$

where $S \in \mathcal{R}H_\infty$ is the so-called dual YJBK parameter. The following result follows completely analogously to Theorem 1, see Moore et al. (1990).

Theorem 2. Let a stabilizing controller K_0 factored as in (8) be given for a number of systems $G_{yu,i}$ factored as in (7). Then $G_{yu,i}, i = 0, \ldots, \nu$ can be implemented as $G_{yu,0}(S_i)$, with $S_i \in \mathcal{R}H_\infty$ given by

$$S_i = \tilde{N}_i M_0 - \tilde{M}_i N_0 = \tilde{M}_i (G_{yu,i} - G_{yu,0}) M_0.$$

Proof: The proof is omitted. ◁

4. GAIN SCHEDULING CONTROL

Based on the results presented in the previous two Sections, we will now describe the gain scheduling method proposed in this work in greater detail. The method can be summarized as follows.

(1) Train an ANN state space model of the plant to be controlled, and validate it using an appropriate test set.
(2) Choose an appropriate number of operating points (typically equilibrium points) and determine the states and control signals in these points, e.g., by using the ANN as a simulation model.
(3) Extract linearized models in each operating point using the weight matrices and neuron function derivatives as presented in (4).
(4) Design controllers for each local linearized model.
(5) Choose an appropriate scheduling variable and implement the gain scheduled control law as specified in Theorem 1 and (10).

The design should then be evaluated by simulation and/or real-life tests. If the design does not seem feasible, it may be possible to make it feasible by choosing a finer grid of operating points, thus obtaining a more smooth scheduling. In this case it may be a help to use the dual YJBK parameterization to examine the 'range' of systems that can be stabilized by a given controller, in order to get an indication of how fine a grid of operating points that may be required.

The method is now illustrated on a simulation example. Inspired by Cabrera and Narendra (1999) we choose the following nonlinear continuous-time process

$$\dot{\eta}_1 = -\eta_1 + \eta_2 \qquad (13)$$
$$\dot{\eta}_2 = -\eta_2 + \eta_3 \qquad (14)$$
$$\dot{\eta}_3 = \frac{\eta_1^2 + \eta_2^2 + \eta_3^2 + \tanh(u)}{\eta_1^2 + \eta_2^2 + \eta_3^2 + 1} - \eta_3 \qquad (15)$$
$$y = \eta_1 \qquad (16)$$

as our example plant, and simulate it with MatLab's `ode45` differential equation solver. The plant itself is assumed to be unknown, and only input-output information is assumed to be available. A training set consisting of $N = 5000$ corresponding samples of input and output is obtained from the plant [1] by applying a low-pass filtered random input signal to the plant, and sampling corresponding input and output values at a sample frequency of 2 Hz. Uniform random measurement noise in the interval $[-0.05; 0.05]$ is added, and an MLP state space model with 5 neurons in the hidden layer is trained to identify the model. The ANN is trained according to the methods presented in Levin and Narendra (1996), where the state vector x_k at sample time $k, n \leq k \leq N$, is constructed from delayed samples of the measured output y_k, i.e., the discrete-time neural state space model is formulated as

[1] Training continuous-time neural network models implemented on digital computers is not very practical. However, the results presented in Section 3 apply equally well to discrete-time systems, so the method can be employed for sampled-data systems as long as the intersample behavior of the system is sufficiently smooth.

$$
\begin{aligned}
x_k &= \begin{bmatrix} y_k & y_{k-1} & y_{k-2} \end{bmatrix}^T \\
\mu_k &= \begin{bmatrix} u_k & u_{k-1} & u_{k-2} \end{bmatrix}^T \\
\hat{x}_{k+1} &= W_o \sigma \left(W_x x_k + W_u \mu_k + W_b \right) \\
\hat{y}_k &= \hat{x}_{1,k}
\end{aligned}
$$

with $\hat{\cdot}$ denoting estimates. The ANN is trained 50 epochs with the Levenberg-Marquardt algorithm and validated using a test set. Figure 2 shows a simulation of a test set, where the state estimate from the ANN is fed back and used as input to the network:

$$
\begin{aligned}
\hat{x}_{k+1} &= W_o \sigma \left(W_x \hat{x}_k + W_u \mu_k + W_b \right) \\
\epsilon_k &= y_k - \hat{x}_{1,k}.
\end{aligned}
$$

For comparison purposes, a linear model is identified using the same training set. The same test set is simulated and the output of the linear model is plotted in Figure 2. As can be seen, the ANN is not a perfect simulation model, but at least superior to the linear model.

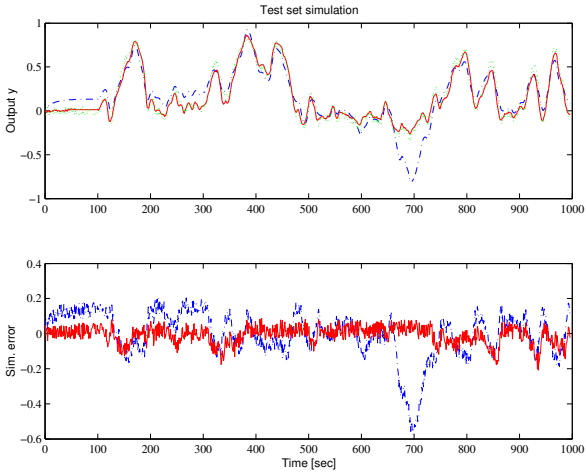

Fig. 2. *Top: open-loop simulation of the test set (green dotted line) by the linear (blue dash-dot line) and the MLP model (red full line). Bottom: open-loop simulation errors for the linear (blue dash-dot line) and the MLP model (red full line).*

The next step is then to linearize the model in a number of relevant operating points $(\bar{x}_i, \bar{u}_i)$ using the weight matrices and derivative of the neuron mapping as shown in eqn. (4). For simplicity, we only linearize around two equilibrium points, given by $\bar{y}_0 = 0.25$ and $\bar{y}_1 = 0.75$. By supplying constant control signals to the ANN model, we find $(\bar{x}_0, \bar{u}_0) = ([0.25, 0.25, 0.25]^T, 0.14)$ and $(\bar{x}_1, \bar{u}_1) = ([0.75, 0.75, 0.75]^T, 0.31)$.

Two observer-based controllers, K_0 and K_1, of the form (6) are then designed in these two operating points. Since the only information about noise and modeling error assumed to be available to the control design is the simulation error shown in Figure 2, we choose to design the controllers using LQG optimal control design methods. That is, the feedback and observer gains F and L are chosen such that the $\mathcal{H}_2$ norm of the closed loop system gain from w to z given in (11) is minimized. The controllers are augmented with integral states in order to remove steady-state errors.

Finally, we choose the scheduling variable α. Since we are interested in controlling the output y_k, the simplest choice of α at sample k is

$$
\alpha_k = \frac{\|y_k - \bar{y}_0\|_2}{\|\bar{y}_0 - \bar{y}_1\|_2}.
$$

The controller is thus parameterized at each sample instant according to

$$
\begin{aligned}
K_0 &= \tilde{V}_0^{-1} \tilde{U}_0 \\
K_1 &= \tilde{V}_1^{-1} \tilde{U}_1 \\
K(Q_k) &= \left(\alpha_k \tilde{V}_0 + (1 - \alpha_k) \tilde{V}_1 \right)^{-1} \times \\
&\qquad \left(\alpha_k \tilde{U}_0 + (1 - \alpha_k) \tilde{U}_1 \right).
\end{aligned}
$$

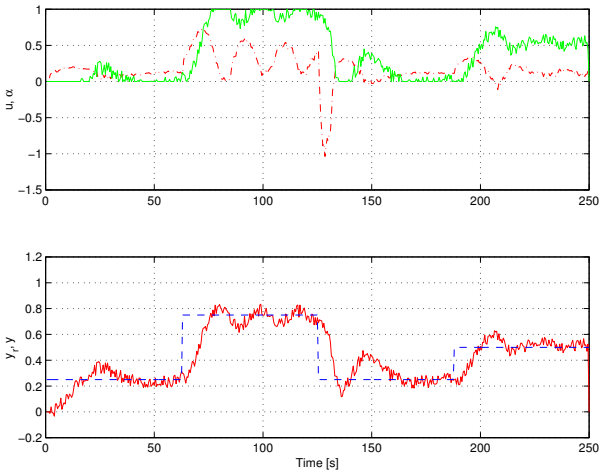

Fig. 3. *Simulation of the controlled system. Top: scheduling variable α (green full line) and control signal u (red dash-dot line). Bottom: reference signal y_r (blue dash-dot line) and controlled output y (red full line).*

Figure 3 shows a simulation of the plant controlled by the gain scheduled controller. At each sample instant the plant output (including measurement noise) is sampled, α_k is calculated and the interpolated controller is found according to the scheme given above; then the plant response to the zero-order-hold control signal u_k is simulated in continuous time for one sample period. The initial states were set to 0. The reference was chosen as a series of steps, such that the output should reach the first operating point first, then the second operating point, then back to the first, and finally settle in between the two operating points. The scheduling variable is plotted together

with the control signal. As can be seen, the control loop stabilizes the plant without difficulties, even in between the operating points. The controllers K_0 and K_1 could probably be tuned more aggressively, but the simulation is sufficient to demonstrate that the gain scheduling scheme works.

5. DISCUSSION

This paper presented a novel use of neural state space models for gain scheduling control. It is assumed that a neural state space model of a nonlinear system can be trained based on in- and output training samples from the system, after which local linearized state space models are extracted from the neural network in a number of operating points according to a simple and computationally cheap scheme. Controllers can then be calculated based on e.g. $\mathcal{H}_2$ theory in each of these operating points, such that they stabilize the system in that operating point in the presence of noise etc.

It was then discussed how the YJBK-parameterization can be exploited to achieve a scaling of the different controllers in each operating point that can be guaranteed to be stable. It was also suggested to use the dual YJBK-parameterization to evaluate whether a given controller will stabilize the systems between a set of operating points. This provides a tool for checking whether the overall design is sufficient, or further refinement is required. Finally, the feasibility of the method was demonstrated on a simulation example involving a continuous-time plant modeled by a discrete-time multi-layer perceptron and a corresponding discrete-time YJBK-parameterized gain scheduling controller.

It should be pointed out that the work presented in this paper is merely a preliminary presentation of the basic gain scheduling scheme based on neural state space models. Further work will be needed to provide strict stability proofs in the presence of (bounded) modeling errors, for instance by considering more stringent uncertainty models that can be used for robust $\mathcal{H}_\infty$ synthesis. Other future work will involve gain scheduling in multi-dimensional systems and research into how to switch between different controllers in order to keep the overall controller order low.

Furthermore, it is implicitly assumed that the transitions between the individual operating points must happen "sufficiently slowly" for the gain scheduling scheme to succeed. The YJBK scheme does not guarantee stability when scheduling in between operating points, only when switching in an operating point. Further research should therefore address this issue, possibly by considering the scheme in the gain scheduling framework presented in e.g., Shamma and Athans (1990) or the LPV framework of Apkarian and Gahinet (1995).

References

P. Apkarian and P. Gahinet. A convex characterization of gain-scheduled h_∞ controllers. *IEEE Transactions on Automatic Control*, 35: 898–907, 1995.

J. D. Bendtsen and K. Trangbaek. Robust quasi-lpv control based on neural state space models. *IEEE Transactions on Neural Networks*, 13: 355–368, 2002.

M. Brown and C. Harris. *Neurofuzzy Adaptive Modelling and Control*. Prentice-Hall International, 1994. ISBN 0-13-134453-6.

J. B. D. Cabrera and K. S. Narendra. Issues in the application of neural networks for tracking based on inverse control. *IEEE Transactions on Automatic Control*, 44:2007–2027, 1999.

B. De Moor J. A. K. Suykens, J. Vandewalle. Nonlinear system identification using neural state space models, applicable in robust control design. *International Journal of Control*, 1:129–152, 1995.

A. U. Levin and K. S. Narendra. Control of nonlinear dynamical systems using neural networks—part ii: Observability, identification and control. *IEEE Transactions on Neural Networks*, 7:30–42, 1996.

S. Lu and T. Basar. Robust nonlinear system identification using neural network models. *IEEE Transactions on Neural Networks*, 9:407–429, 1998.

J. B. Moore, K. Glover, and A. Telford. All stabilizing controllers as frequency shaped state estimate feedback. *IEEE Transactions on Automatic Control*, 35:203–208, 1990.

H. Niemann and J. Stoustrup. An architecture for implementation of multivariable controllers. In *Proc. of the American Control Conference*, 1999.

J. S. Shamma and M. Athans. Analysis of gain scheduled control for nonlinear plants. *IEEE Transactions on Automatic Control*, 35:898–907, 1990.

K. Zhou, J. Doyle, and K. Glover. *Robust And Optimal Control*. Prentice-Hall International, 1995. ISBN 0-13-456567-3.