Aalborg Universitet

**AALBORG
UNIVERSITY**

Robust Quasi-LPV Control Based on Neural State Space Models

Bendtsen, Jan Dimon; Trangbæk, Klaus

Publication date:
2000

Document Version
Også kaldet Forlagets PDF

Link to publication from Aalborg University

Robust Quasi-LPV Control Based on Neural State Space Models

Jan Dimon Bendtsen, Klaus Trangbæk

Abstract— **In this paper we derive a synthesis result for robust LPV output feedback controllers for nonlinear systems modelled by neural state space models. This result is achieved by writing the neural state space model on a linear fractional transformation form in a non-conservative way, separating the system description into a linear part and a nonlinear part. Linear parameter-varying control synthesis methods are then applied to design a nonlinear control law for this system. Since the model is assumed to have been identified from input-output measurement data only, it must be expected that there is some uncertainty on the identified nonlinearities. The control law is therefore made robust to noise perturbations. After formulating the controller synthesis as a set of LMIs with added constraints, some implementation issues are addressed and a simulation example is presented.**

Index Terms—Multi-Layer Perceptrons, Neural Networks, Linear Fractional Transformation, Quasi-LPV Control, Linear Matrix Inequalities

I. INTRODUCTION

Many nonlinear systems found in real-life situations are almost linear in a limited region of the relevant state space, but exhibit saturation and other nonlinear phenomena more strongly when the state of the system gets outside this region. The classical approach to control such systems has been to linearise the system model in some set of operating points and design one or more linear controllers for the system in said points. Modern control paradigms such as robust $\mathcal{H}_\infty$ control synthesis methods typically deal with this by requiring a linear nominal (state space) model plus some kind of residual model for the control design. Recent work on linear parameter varying (LPV) control has taken these ideas further, compensating for known parameter variations directly in the control design [9], [16], [17], [18], [19]. Linear parameter-varying systems are linear systems whose system matrices depend on some time-varying parameter vector that is either fully known or at least known to be contained in some known set. In LPV control design this knowledge is employed to provide systematic gain scheduling in order to guarantee stability and performance of the closed loop. One problem with these types of approaches, however, is that it can be difficult to obtain a suitable model to build the control design on.

With the right choice of neuron functions, artificial neural networks such as Multi-Layer Perceptrons (MLPs) have been shown to be able to model the kind of nonlinear systems described above to an arbitrary degree of accuracy, under mild assumptions on continuity and boundedness.

Neural networks have therefore found many applications in control theory, for instance for feedback linearisation [5], [8], [11] and sliding mode control laws [14]. They have also been proven useful as observers [10], in direct adaptive control [7], [20], and in other roles. Not much work has been done on achieving gain scheduling control based on artificial neural networks so far, however. In [12] a previously tuned gain scheduling controller was approximated by a neural network which then replaced the gain scheduling controller in the loop. Other approaches (e.g. [4]) uses a neural network to schedule between a finite set of previously designed classical controllers, and have been somewhat *ad hoc*.

With the emergence of Linear Matrix Inequalities (LMIs) as a powerful tool for robust control and the subsequent development of a theory for LPV control design, on the other hand, a door has been opened for a more analytical approach to gain scheduling control based on neural state space models. Such an extension of controller synthesis ideas from linear theory to the nonlinear framework of neural networks is a fundamentally sound idea, of course, but requires a method for separating the neural state space model into a linear and a nonlinear part in a manner suitable for the synthesis.

Hence, one of the things we wish to do in this paper is to establish a link from the MLP description to a Linear Fractional Transformation (LFT) description. Some work along these lines has already been presented in [1], [21] and [22], among others. Another previous approach to the control problem has been to design a fixed controller for the identified system and making it robust to the nonlinearities isolated in the uncertainty block of the LFT description [1]. However, the fact that the nonlinearities are actually known at the design stage means that the control law can be designed to take advantage of this information as well, achieving a nonlinear and less conservative controller. The key to this approach is the observation that the LFT formulation described above allows for a (quasi-)LPV description of a nonlinear system. In this case, the extracted nonlinearities define the parameter variation, and it is hence possible to exploit this information in the control design. However, since we are dealing with identified models, it can be expected that there are noise, modelling errors etc. that may cause adverse effects on the control. Therefore, as the main result of this paper we show how to make the LPV control law robust to small disturbances. This step involves the formulation of extra constraints on the synthesis solutions.

The outline of the rest of the paper is as follows. In the following two sections, some preliminaries necessary for deriving the main result will be presented. A method for

The authors are with the Department of Control Engineering, Aalborg University, Fredrik Bajersvej 7C, 9220 Aalborg East, Denmark. Email: {dimon,ktr}@control.auc.dk

transforming a nonlinear state space model parameterised via an MLP into an LFT description in a non-conservative manner is given in Section II, followed by a brief review in Section III of some key concepts in LPV control: controller synthesis matrix inequalities and multipliers. This section also presents a method to achieve a controller for a non-strictly proper system.

Section IV presents the main result of the paper. It is a formulation of a set of extra constraints for the controller synthesis that, if possible to fulfill, will provide a robust quasi-LPV control law for a nonlinear system parameterised by an MLP. Section V discusses some implementation issues, after which Section VI presents a simulation example of modelling and control of a nonlinear system with the proposed method. Finally, Section VII sums up the conclusions of the work.

II. From Neural State Space Models to an LFT Framework

We consider a system of the form

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

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

From neural network theory—see e.g. [13]—it is known that we can approximate this function to a desired accuracy with a single hidden layer MLP with l neurons (assuming l is chosen large enough):

$$f(\tilde{x}, \tilde{u}) = W_o \sigma\left(W_x \tilde{x} + W_u \tilde{u} + \tilde{W}_b\right) + \varepsilon_x$$

where $W_o \in \mathbb{R}^{n \times l}$ and $W_x \in \mathbb{R}^{l \times n}, W_u \in \mathbb{R}^{l \times m}$ contain the output and hidden layer weights, respectively. $\sigma(\cdot) : \mathbb{R}^l \to \mathbb{R}^l$ is a continuous, diagonal, static nonlinearity. $\tilde{W}_b \in \mathbb{R}^l$ contains a set of biases which will allow us to model non-odd functions with odd neuron functions $\sigma(\cdot)$ such as the hyperbolic tangent. We assume it is possible to achieve a smaller modelling error than the measurement noise by choosing the MLP large enough and train it long enough on a sufficiently rich training set.

In other words, we will assume that the neural network can be trained to estimate the states in the system (1). In practice this can for instance be achieved by employing back-propagation of errors (see for instance [15]), and we will for simplicity only consider off-line training here; i.e. we will not consider time-varying systems.

Consider a system for which a neural state space model has been trained according to the guidelines given above, until ε_x is small enough to be ignored:

$$\dot{\tilde{x}} = W_o \sigma\left(W_x \tilde{x} + W_u \tilde{u} + \tilde{W}_b\right), \quad \tilde{y} = C\tilde{x}. \tag{2}$$

We wish to rewrite the neural model (2) as the linear frac-

tional transformation

$$
\begin{aligned}
\dot{x} &= Ax + Bu + B_1\Omega(\xi) \\
\xi &= W_x x + W_u u \\
y &= Cx
\end{aligned}
\tag{3}
$$

where the residual function $\Omega(\cdot) : \mathbb{R}^l \to \mathbb{R}^l$ is a static diagonal nonlinearity, which is bounded with l_2-gain less than 1, and where the coordinates (x, u) only differ from $(\tilde{x}, \tilde{u})$ by the possible subtraction of an equilibrium point. The presented method was first discussed in [1], but for the sake of completeness we will reiterate it in the following.

We assume that there exists an equilibrium, $(\tilde{x}, \tilde{u}) = (\tilde{x}^\circ, \tilde{u}^\circ)$, i.e.

$$0 = W_o \sigma(W_x \tilde{x}^\circ + W_u \tilde{u}^\circ + \tilde{W}_b).$$

We can then change the network coordinates in such a way that instead of the arbitrary equilibrium point $(\tilde{x}^\circ, \tilde{u}^\circ)$ we have $0 = W_o \sigma'(0)$ (σ' is a new neuron function mapping which will be defined shortly). Let the new coordinates be given as $x = \tilde{x} - \tilde{x}^\circ, u = \tilde{u} - \tilde{u}^\circ$. Then (2) can be written as

$$\dot{x} = W_o \sigma\left(W_x(x + \tilde{x}^\circ) + W_u(u + \tilde{u}^\circ) + \tilde{W}_b\right).$$

Here we will define a new bias vector $W_b = W_x \tilde{x}^\circ + W_u \tilde{u}^\circ + \tilde{W}_b$ and the new neuron function $\sigma'(\xi)$, where ξ is defined as in (3):

$$
\begin{aligned}
\sigma'(\xi) &= \sigma\left(\xi + W_x \tilde{x}^\circ + W_u \tilde{u}^\circ + \tilde{W}_b\right) - \sigma(W_b) \\
&= \sigma\left(W_x(x + \tilde{x}^\circ) + W_u(u + \tilde{u}^\circ) + \tilde{W}_b\right) - \sigma(W_b).
\end{aligned}
$$

Adding and subtracting $W_o\sigma(W_b)$ in (2) then gives

$$
\begin{aligned}
\dot{x} &= W_o \sigma\left(W_x \tilde{x} + W_u \tilde{u} + \tilde{W}_b\right) + W_o\sigma(W_b) - W_o\sigma(W_b) \\
&= W_o\left(\sigma\left(W_x \tilde{x} + W_u \tilde{u} + \tilde{W}_b\right) - \sigma(W_b)\right) + W_o\sigma(W_b) \\
&= W_o\sigma'(W_x x + W_u u) = W_o\sigma'(\xi).
\end{aligned}
$$

$W_o\sigma(W_b) = 0$, because this is in fact the equilibrium point.

Remark 1 Note that, apart from providing a way to shift the operating point to the origin, the main purpose of the steps given above is to remove the bias from σ instead of having to consider it as a constant disturbance input, as suggested in [22]. ◁

Remark 2 It should furthermore be noted that the method given above applies equally well to sampled-data systems $\tilde{x}_{k+1} = f(\tilde{x}_k, \tilde{u}_k)$. In this case the MLP equilibrium point is of the form $\tilde{x}_{k+1}^\circ = f(\tilde{x}_k^\circ, \tilde{u}_k^\circ)$, $\forall k$, but the definition of $\sigma'(\cdot)$ turns out to be the same. ◁

Now we can find the effective range of the input arguments to the neuron functions. This is simply done by calculating

$$\xi_{j,max} = \sup_{0 \geq t \geq T} \left\{ |W_x^j x(t) + W_u^j u(t)| \right\}$$

for $1 \leq j \leq l$ where $t \in [0; T]$ is the time interval in which the training data have been acquired and W_x^j, W_u^j denote the j'th rows in the hidden layer weight matrices. Then we have the following bounds on the active input range[1] of the j'th neuron:

$$\xi_j = W_x^j x + W_u^j u \in [-\xi_{j,max}; \xi_{j,max}].$$

Hence the neuron function response to the active input range must belong to the sector $\sigma'_j \in [k_{j,min}, \ k_{j,max}]$ where

$$k_{j,min} = \inf_{\xi_j \in [-\xi_{j,max}; \xi_{j,max}]\backslash\{0\}} \left\{ \frac{\sigma'_j(\xi_j)}{\xi_j} \right\} \qquad (4)$$

and

$$k_{j,max} = \sup_{\xi_j \in [-\xi_{j,max}; \xi_{j,max}]\backslash\{0\}} \left\{ \frac{\sigma'_j(\xi_j)}{\xi_j} \right\}. \qquad (5)$$

In other words, the sector bounds are determined such that $k_{j,min}\xi_j^2 \leq \sigma'(\xi_j)\xi_j \leq k_{j,max}\xi_j^2$. The actual expressions for these sector bounds must be found for each neuron function individually and will in general depend on the bias, but the bounds obviously exist and are the least conservative easily achievable bounds. A procedure for finding them for $\tanh(\cdot)$ neuron functions is given below.

Once the sector bounds are found, we go back to vector notation and define the nonlinear function $\omega(\cdot): \mathbb{R}^{n+m} \to \mathbb{R}^n$ as

$$\omega(\xi) = \sigma'(\xi) - \frac{1}{2}(K_{min} + K_{max})\xi \qquad (6)$$

where $K_{min} = \text{diag}\{k_{j,min} - \epsilon\}$ and $K_{max} = \text{diag}\{k_{j,max} + \epsilon\}, 1 \leq j \leq l$. ϵ is a small positive quantity included to make the sector bounds strict. It is observed that $\omega(\cdot)$ belongs to the sector $(-\frac{1}{2}(K_{max} - K_{min}), \ \frac{1}{2}(K_{max} - K_{min}))$. Now we can write the equation for $\dot{x}$ as

$$\begin{aligned} \dot{x} &= W_o \sigma'(W_x x + W_u u) \\ &= W_o \left(\omega(\xi) + \frac{1}{2}(K_{min} + K_{max})\xi \right) \\ &= Ax + Bu + B_1 \Omega(\xi) \end{aligned}$$

in which A, B, B_1 and Ω are given by

$$A = \frac{1}{2}W_o(K_{min} + K_{max})W_x \qquad (7)$$

$$B = \frac{1}{2}W_o(K_{min} + K_{max})W_u \qquad (8)$$

$$B_1 = \frac{1}{2}W_o(K_{max} - K_{min}) \qquad (9)$$

$$\Omega(\xi) = 2(K_{max} - K_{min})^{-1}\omega(\xi). \qquad (10)$$

Note that the diagonal scaling by $\frac{1}{2}(K_{max} - K_{min})$ is included in order to make the diagonal static nonlinearity Ω belong to the sector $(-1, 1)$.

[1] The input ranges are in general not symmetric around 0, so the bounds given here may be slightly conservative.

Remark 3 When designing LPV or quasi-LPV controllers, we are interested in the tightest possible bounds $K_{max} - K_{min}$ in order to avoid conservatism. Although the LPV synthesis method described in Section III is essentially non-conservative, it is usually necessary to use simplified multipliers, for instance by disregarding knowledge on the rate of change of the gains of the residual function, to make the synthesis implementable and to avoid controller switching. A quasi-LPV representation potentially introduces further conservatism due to non-uniqueness of the nonlinear function representation. For the sake of the controller synthesis we are therefore interested in keeping these gains from varying too much. ◁
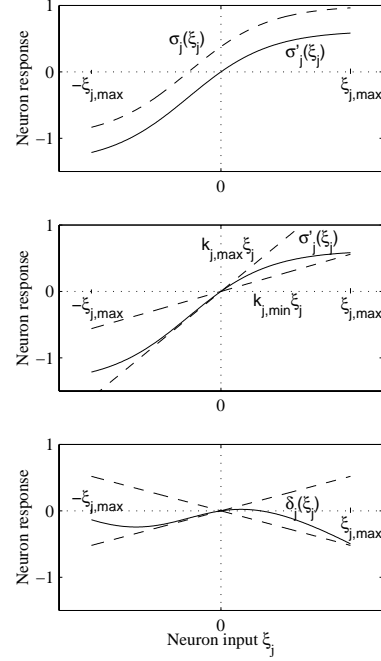

Fig. 1. *Extraction of linear content from a hyperbolic tangent neuron.*

In order to illustrate the procedure above we will provide an expression for the sector bounds (4) and (5) for the $\tanh(\cdot)$ neuron function, which is probably the most popular neuron function employed in MLPs. Consider the neuron $\sigma_j(\xi_j) = \tanh(\xi_j + w_b)$ where w_b is the scalar bias on the input ξ_j. Refer to Figure 1, where the top plot shows the parallel translation of the original neuron function with bias w_b to the origin. We will without loss of generality assume that $w_b > 0$. Only the section of the neuron function, which corresponds to the input interval $[-\xi_{j,max}; \xi_{j,max}]$, is considered.

On the middle plot the straight lines $k_{j,min}\xi_j$ and $k_{j,max}\xi_j$ have been added. Since $d^2\tanh(s)/ds^2 < 0$ for $s > 0$ it is immediately concluded that $k_{j,min}$ is given by $k_{j,min} = \sigma'_j(\xi_{j,max})/\xi_{j,max}$. $k_{j,max}$, on the other hand, can either be given by $\sigma'_j(-\xi_{j,max})/-\xi_{j,max}$ if the endpoint of the input range is sufficiently close to 0, or by the slope of the tangent to the neuron function which intersects 0. The relationship between the bias and the argument ξ_b for which said tangent coincides with the neuron function has

been found numerically[2] as

$$\xi_b = -0.00379 w_b^3 + 0.07274 w_b^2 - 1.5146 w_b.$$

Hence, if $\xi_b > -\xi_{j,max}$ we have $k_{j,max} = \sigma_j'(\xi_b)/\xi_b$, otherwise $k_{j,max} = \sigma_j'(-\xi_{j,max})/-\xi_{j,max}$.

Note that there is no loss of generality in the assumption $w_b > 0$ since the fact that the (original) neuron function is odd ensures that the expressions given above hold for negative biases as well, with a simple sign change of ξ_b and $\xi_{j,max}$.

Once the sector bounds for the nonlinearity have been determined we also have an explicit, smooth expression for the new set of neuron functions (given by eqns. (6) and (10)). If there is any uncertainty in the knowledge of x and u, then this will of course result in an uncertainty on the knowledge of $\Omega(\cdot)$. However, if we assume that some bound on this uncertainty is known, then the above expression can be exploited to provide a bound on the gain of the nonlinearity itself:

$$\frac{\hat{\Omega}_j(\xi_j)}{\xi_j} = \frac{\Omega_j(\xi_j)}{\xi_j} + \varepsilon_{\Omega j}, \quad |\varepsilon_{\Omega j}| < \bar{\varepsilon}_{\Omega j}$$

where Ω_j is the j'th diagonal element of Ω. Such a bound can for instance be found by conducting a numerical search over the range of all permissible values of ξ. The bound on the measurement noise can be used together with W_x and W_u to estimate the uncertainty on ξ; then this uncertainty can be used to calculate an upper bound on ε_Ω.

III. LPV Control Preliminaries

This section reviews some key concepts in the synthesis of linear parameter-varying controllers that we will need for the control design in the following section. Consider the system

$$\left[\begin{array}{c} \dot{x} \\ \hline z_u \\ z_p \\ y \end{array}\right] = \left[\begin{array}{c|ccc} A & B_u & B_p & B \\ \hline C_u & D_{uu} & D_{up} & E_u \\ C_p & D_{pu} & D_{pp} & E_p \\ C & F_u & F_p & 0 \end{array}\right] \left[\begin{array}{c} x \\ \hline w_u \\ w_p \\ u \end{array}\right] \quad (11)$$

with $x \in \mathbb{R}^n, u \in \mathbb{R}^m$ and $y \in \mathbb{R}^p$ representing states, inputs and outputs. $w_p, z_p \in \mathbb{R}^{n_p}$ are used to specify performance and $w_u, z_u \in \mathbb{R}^{n_u}$ are the channels connecting the residual gains collected in Δ with the nominal linear system described by (A, B, C). All the matrices are assumed to be real and of appropriate dimensions. Assume for now that $w_u = 0$. If we interconnect the LTI controller described by the system matrices (A_c, B_c, C_c, D_c) with input y, output u and controller state x_c with the nominal system we arrive at a system of the form

$$\left[\begin{array}{c} \dot{x} \\ \dot{x}_c \\ \hline z_u \\ z_p \end{array}\right] = \left[\begin{array}{c|c} \mathcal{A} & \mathcal{B} \\ \hline \mathcal{C} & \mathcal{D} \end{array}\right] \left[\begin{array}{c} x \\ x_c \\ \hline w_u \\ w_p \end{array}\right]$$

[2]A closed form most likely does not exist. The polynomial given here provides values of $k_{j,max}$ with errors of the order of magnitude 10^{-5}.

where

$$\mathcal{A} = \begin{bmatrix} A + BD_cC & BC_c \\ B_cC & A_c \end{bmatrix}$$

$$\mathcal{B} = \left[\begin{array}{c|c} \mathcal{B}_u & \mathcal{B}_p \end{array}\right]$$

$$= \left[\begin{array}{c|c} B_u + BD_cF_u & B_p + BD_cF_p \\ B_cF_u & B_cF_p \end{array}\right]$$

$$\mathcal{C} = \left[\begin{array}{c} \mathcal{C}_u \\ \hline \mathcal{C}_p \end{array}\right]$$

$$= \left[\begin{array}{cc} C_u + E_uD_cC & E_uC_c \\ \hline C_p + E_pD_cC & E_pC_c \end{array}\right]$$

$$\mathcal{D} = \left[\begin{array}{c|c} \mathcal{D}_{uu} & \mathcal{D}_{up} \\ \hline \mathcal{D}_{pu} & \mathcal{D}_{pp} \end{array}\right]$$

$$= \left[\begin{array}{c|c} D_{uu} + E_uD_cF_u & D_{up} + E_uD_cF_p \\ \hline D_{pu} + E_pD_cF_u & D_{pp} + E_pD_cF_p \end{array}\right].$$

Define

$$P_p = \begin{bmatrix} Q_p & S_p \\ S_p^T & R_p \end{bmatrix}, \quad R_p \geq 0.$$

Then the controlled system is exponentially stable and fulfills the performance specification

$$\exists \varepsilon > 0 : \int_0^\infty \begin{bmatrix} w_p \\ z_p \end{bmatrix}^T P_p \begin{bmatrix} w_p \\ z_p \end{bmatrix} dt \leq -\varepsilon \int_0^\infty w_p^T w_p dt$$
$$\text{for } \begin{bmatrix} x(0) \\ x_c(0) \end{bmatrix} = 0 \quad (12)$$

if and only if there exists a Lyapunov matrix $\mathcal{X} > 0$ such that the following matrix inequality is fulfilled:

$$\begin{bmatrix} \mathcal{A}^T\mathcal{X} + \mathcal{X}\mathcal{A} & \mathcal{X}\mathcal{B}_p \\ \mathcal{B}_p^T\mathcal{X} & 0 \end{bmatrix} + \begin{bmatrix} 0 & I \\ \mathcal{C}_p & \mathcal{D}_{pp} \end{bmatrix}^T P_p \begin{bmatrix} 0 & I \\ \mathcal{C}_p & \mathcal{D}_{pp} \end{bmatrix} \leq 0. \quad (13)$$

A matrix inequality such as (13) is satisfied if the eigenvalues of the matrix expression on the left-hand side (LHS) are less than or equal to 0. If the LHS is linear in the unknown matrices, the matrix inequality is called an LMI, and if it has a solution in the unknown(s), it is said to be feasible. LMIs can be solved efficiently using standard software tools; refer to e.g. [2] for more information on LMIs in general. When used for analysis, the only unknown in (13) is $\mathcal{X}$, and hence it is an LMI. In connection with synthesis it becomes a non-convex problem due to the presence of the matrix product $\mathcal{X}\mathcal{A}$. However, it is possible to reformulate (and solve) the problem in a synthesis LMI framework. Partition the inverse of the performance matrix P_p by

$$P_p^{-1} = \begin{bmatrix} \tilde{Q}_p & \tilde{S}_p \\ \tilde{S}_p^T & \tilde{R}_p \end{bmatrix} \quad (14)$$

and assume that the numbers of positive and negative eigenvalues of P_p are equal to the dimensions of R_p and Q_p, respectively. Then it is possible to fulfill (13) if we can compute basis matrices

$$\Phi = \begin{bmatrix} \Phi_1 \\ \Phi_2 \end{bmatrix} \quad \text{and} \quad \Psi = \begin{bmatrix} \Psi_1 \\ \Psi_2 \end{bmatrix}$$

of ker $\begin{bmatrix} B^T & E_p^T \end{bmatrix}$ and ker $\begin{bmatrix} C & F_p \end{bmatrix}$ and then find symmetric X and Y that satisfy the following coupled linear matrix inequalities in X and Y (see [19]):

$$\begin{bmatrix} X & I \\ I & Y \end{bmatrix} \geq 0$$

$$\Psi^T \begin{bmatrix} * \\ * \\ * \\ * \end{bmatrix}^T \left[\begin{array}{cc|cc} 0 & X & 0 & 0 \\ X & 0 & 0 & 0 \\ \hline 0 & 0 & Q_p & S_p \\ 0 & 0 & S_p^T & R_p \end{array}\right] \begin{bmatrix} I & 0 \\ A & B_p \\ 0 & I \\ C_p & D_p \end{bmatrix} \Psi < 0$$

$$\Phi^T \begin{bmatrix} * \\ * \\ * \\ * \end{bmatrix}^T \left[\begin{array}{cc|cc} 0 & Y & 0 & 0 \\ Y & 0 & 0 & 0 \\ \hline 0 & 0 & \tilde{Q}_p & \tilde{S}_p \\ 0 & 0 & \tilde{S}_p^T & \tilde{R}_p \end{array}\right] \begin{bmatrix} -A^T & -C_p^T \\ I & 0 \\ -B_p^T & -D_p^T \\ 0 & I \end{bmatrix} \Phi > 0.$$

Once X and Y have been found, it is possible to construct the controller matrices (A_c, B_c, C_c, D_c) and the Lyapunov matrix $\mathcal{X}$.

Turning to LPV systems, we consider the LFT setup depicted in Figure 2, where $w_p, z_p \in \mathbb{R}^m$ are used to specify performance and $w_u, z_u \in \mathbb{R}^l$ are the channels connecting the residual gains collected in Δ with the nominal linear system M. The objective is, if possible, to find a gain-scheduled control law $K(\Delta_c)$ and a scheduling function $\Delta_c(\Delta(t))$ such that the closed loop system fulfills a given performance specification.
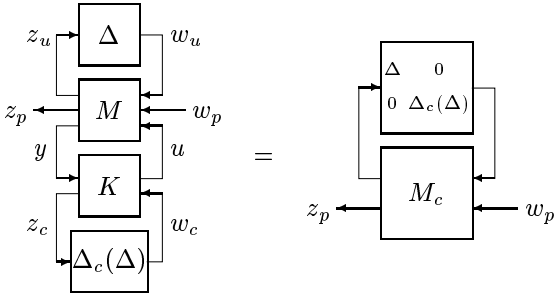

Fig. 2. *The interconnection of the nominal system M, the residual gains Δ, and the controller K.*

We assume that the LPV system is described as an LTI system of the form (11) and the parameter variation is captured via the residual gain channel as

$$w_u(t) = \Delta(t) z_u(t), \quad \Delta \in \mathbf{\Delta}. \tag{15}$$

Explicit online knowledge of $\Delta(t)$ allows (and is, indeed, necessary for) scheduling the controller $K(\Delta_c)$. The controller is chosen to be of the form

$$\dot{x}_c = A_c x_c + B_c \begin{bmatrix} y \\ w_c \end{bmatrix}, \quad \begin{bmatrix} u \\ z_c \end{bmatrix} = C_c x_c + D_c \begin{bmatrix} y \\ w_c \end{bmatrix} \tag{16}$$

with $w_c(t) = \Delta_c(\Delta(t)) z_c(t)$, in itself a nonlinear function of Δ. If we interconnect the controller and the nominal system we get the linear time invariant system

$$\left[\begin{array}{c} \dot{\chi} \\ \hline z_u \\ z_c \\ z_p \end{array}\right] = \left[\begin{array}{c|ccc} \mathcal{A} & \mathcal{B}_u & \mathcal{B}_c & \mathcal{B}_p \\ \hline \mathcal{C}_u & \mathcal{D}_{uu} & \mathcal{D}_{uc} & \mathcal{D}_{up} \\ \mathcal{C}_c & \mathcal{D}_{cu} & \mathcal{D}_{cc} & \mathcal{D}_{cp} \\ \mathcal{C}_p & \mathcal{D}_{pu} & \mathcal{D}_{pc} & \mathcal{D}_{pp} \end{array}\right] \left[\begin{array}{c} \chi \\ \hline w_u \\ w_c \\ w_p \end{array}\right] \tag{17}$$

and a parameter dependency defined by (as depicted in the right part of Figure 2)

$$\begin{bmatrix} w_u \\ w_c \end{bmatrix} = \begin{bmatrix} \Delta(t) & 0 \\ 0 & \Delta_c(\Delta(t)) \end{bmatrix} \begin{bmatrix} z_u \\ z_c \end{bmatrix}. \tag{18}$$

Robust quadratic performance for the controlled system is defined as follows.
• The interconnection of system and controller is well-posed, i.e. $I - \Delta \begin{bmatrix} \mathcal{D}_{uu} & \mathcal{D}_{uc} \\ \mathcal{D}_{cu} & \mathcal{D}_{cc} \end{bmatrix}$ is nonsingular for all $\Delta \in \mathbf{\Delta}$.
• Positive constants K and α exist such that $\|\chi(t)\| \leq \|\chi(0)\| K e^{-\alpha t}$ for $t > 0$ and all $\Delta \in \mathbf{\Delta}$ if $w_p = 0$.
• The performance specification (14) holds.
It can now be shown that a sufficient condition for the closed-loop system to achieve robust quadratic performance can be formulated as follows.

Theorem 1 (Scherer) Robust quadratic performance is achieved for (17) if there exist a symmetric $\mathcal{X} > 0$ and a symmetric multiplier

$$P_e = \left[\begin{array}{cc|cc} Q & S & Q_{12} & S_{12} \\ S^T & R & S_{21}^T & R_{12} \\ \hline Q_{12}^T & S_{21} & Q_2 & S_2 \\ S_{12}^T & R_{12}^T & S_2^T & R_2 \end{array}\right] \tag{19}$$

which fulfills

$$\left[\begin{array}{cc} \Delta & 0 \\ 0 & \Delta_c(\Delta) \\ \hline I & 0 \\ 0 & I \end{array}\right]^T P_e \left[\begin{array}{cc} \Delta & 0 \\ 0 & \Delta_c(\Delta) \\ \hline I & 0 \\ 0 & I \end{array}\right] > 0 \quad \forall \Delta \in \mathbf{\Delta} \tag{20}$$

such that

$$\tau^T \left[\begin{array}{cc|cc|cc|cc} 0 & \mathcal{X} & 0 & 0 & 0 & 0 & 0 & 0 \\ \mathcal{X} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \hline 0 & 0 & Q & S & Q_{12} & S_{12} & 0 & 0 \\ 0 & 0 & S^T & R & S_{21}^T & R_{12} & 0 & 0 \\ \hline 0 & 0 & Q_{12}^T & S_{21} & Q_2 & S_2 & 0 & 0 \\ 0 & 0 & S_{12}^T & R_{12}^T & S_2^T & R_2 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 & 0 & 0 & Q_p & S_p \\ 0 & 0 & 0 & 0 & 0 & 0 & S_p^T & R_p \end{array}\right] \tau < 0$$

$$\tag{21}$$

where

$$\tau = \left[\begin{array}{cccc} I & 0 & 0 & 0 \\ \mathcal{A} & \mathcal{B}_u & \mathcal{B}_c & \mathcal{B}_p \\ \hline 0 & I & 0 & 0 \\ \mathcal{C}_u & \mathcal{D}_{uu} & \mathcal{D}_{uc} & \mathcal{D}_{up} \\ \hline 0 & 0 & I & 0 \\ \mathcal{C}_c & \mathcal{D}_{cu} & \mathcal{D}_{cc} & \mathcal{D}_{cp} \\ \hline 0 & 0 & 0 & I \\ \mathcal{C}_p & \mathcal{D}_{pu} & \mathcal{D}_{pc} & \mathcal{D}_{pp} \end{array}\right]. \tag{22}$$

Proof: See [18]. ∎

The extended multiplier P_e in (19) is constructed from multipliers P and $\tilde{P}$ of lower dimension such that:

$$P_e = \begin{bmatrix} P & * \\ * & * \end{bmatrix}, \quad P_e^{-1} = \begin{bmatrix} \tilde{P} & * \\ * & * \end{bmatrix}. \tag{23}$$

P and $\tilde{P}$ must fulfill the following requirements:

$$P = \begin{bmatrix} Q & S \\ S^T & R \end{bmatrix}, \tag{24}$$

$$\begin{bmatrix} \Delta \\ I \end{bmatrix}^T P \begin{bmatrix} \Delta \\ I \end{bmatrix} > 0 \quad \forall\, \Delta \in \boldsymbol{\Delta} \tag{25}$$

and

$$\tilde{P} = \begin{bmatrix} \tilde{Q} & \tilde{S} \\ \tilde{S}^T & \tilde{R} \end{bmatrix}, \tag{26}$$

$$\begin{bmatrix} I \\ -\Delta^T \end{bmatrix}^T \tilde{P} \begin{bmatrix} I \\ -\Delta^T \end{bmatrix} < 0 \quad \forall\, \Delta \in \boldsymbol{\Delta}. \tag{27}$$

We furthermore calculate the basis matrices

$$\Phi^T = \begin{bmatrix} \Phi_1^T & \Phi_2^T & \Phi_3^T \end{bmatrix} \quad \text{and} \quad \Psi^T = \begin{bmatrix} \Psi_1^T & \Psi_2^T & \Psi_3^T \end{bmatrix}$$

of $\ker \begin{bmatrix} B^T & E_u^T & E_p^T \end{bmatrix}$ and $\ker \begin{bmatrix} C & F_u & F_p \end{bmatrix}$, respectively. LPV control synthesis can then be formulated as in the following theorem.

Theorem 2 (Scherer) There exists a controller (16) such that the system (17) admits an $\mathcal{X} > 0$ and a symmetric multiplier P_e that satisfy the LMI (21) if and only if there exist symmetric X and Y and multipliers $P, \tilde{P}$ fulfilling (25) and (27) respectively, that satisfy the following inequalities:

$$\begin{bmatrix} X & I \\ I & Y \end{bmatrix} \geq 0 \tag{28}$$

$$\psi^T \left[\begin{array}{cc|cc|cc} 0 & X & 0 & 0 & 0 & 0 \\ X & 0 & 0 & 0 & 0 & 0 \\ \hline 0 & 0 & Q & S & 0 & 0 \\ 0 & 0 & S^T & R & 0 & 0 \\ \hline 0 & 0 & 0 & 0 & Q_p & S_p \\ 0 & 0 & 0 & 0 & S_p^T & R_p \end{array} \right] \psi < 0 \tag{29}$$

$$\phi^T \left[\begin{array}{cc|cc|cc} 0 & Y & 0 & 0 & 0 & 0 \\ Y & 0 & 0 & 0 & 0 & 0 \\ \hline 0 & 0 & \tilde{Q} & \tilde{S} & 0 & 0 \\ 0 & 0 & \tilde{S}^T & \tilde{R} & 0 & 0 \\ \hline 0 & 0 & 0 & 0 & \tilde{Q}_p & \tilde{S}_p \\ 0 & 0 & 0 & 0 & \tilde{S}_p^T & \tilde{R}_p \end{array} \right] \phi > 0 \tag{30}$$

where

$$\psi = \left[\begin{array}{ccc} I & 0 & 0 \\ A & B_u & B_p \\ \hline 0 & I & 0 \\ C_u & D_{uu} & D_{up} \\ \hline 0 & 0 & I \\ C_p & D_{pu} & D_{pp} \end{array} \right] \Psi$$

and

$$\phi = \left[\begin{array}{ccc} -A^T & -C_u^T & -C_p^T \\ I & 0 & 0 \\ \hline -B_u^T & -D_{uu}^T & -D_{pu}^T \\ 0 & I & 0 \\ \hline -B_p^T & -D_{up}^T & -D_{pp}^T \\ 0 & 0 & I \end{array} \right] \Phi.$$

Proof: See [18]. $\blacksquare$

An LPV controller designed to fulfill these requirements will stabilise the system and achieve the required robust performance defined by (12). Furthermore, one choice of scheduling function that fulfills the 'if' part of Theorem 2 is of the form given by the following Lemma.

Lemma 1 (Scherer) Let X, Y, P and $\tilde{P}$ be calculated as specified in Theorem 2 and suppose that $\tilde{P}$ is nonsingular. Choose the matrix U such that its columns form an orthogonal basis of the image of $P - \tilde{P}^{-1}$, and such that

$$U^T (P - \tilde{P}^{-1})U = N = \begin{bmatrix} N_- & 0 \\ 0 & N_+ \end{bmatrix} \tag{31}$$

with $N_- < 0$ and $N_+ > 0$. Define

$$\begin{bmatrix} V_-(\Delta) & V_+(\Delta) \end{bmatrix} = \begin{bmatrix} \Delta^T & I \end{bmatrix} U \tag{32}$$

with V_- and V_+ having $\dim\{N_-\}$ and $\dim\{N_+\}$ columns, respectively. Construct the extended multiplier

$$P_e = \begin{bmatrix} P & U \\ U^T & N^{-1} \end{bmatrix}.$$

Then P_e fulfills (23), and a controller scheduling function that guarantees (20) is given by

$$\Delta_c(\Delta) = N_- V_-(\Delta)^T \times$$

$$\left(\begin{bmatrix} \Delta \\ I \end{bmatrix}^T P \begin{bmatrix} \Delta \\ I \end{bmatrix} - V_-(\Delta) N_- V_-(\Delta)^T \right)^{-1} V_+(\Delta). \tag{33}$$

Proof: See [18] for a constructive proof. $\blacksquare$

In practice it may happen that the system we design a controller for, is not strictly proper, i.e. $F_3 \neq 0$ in the system

$$\left[\begin{array}{c} \dot{x} \\ z_u \\ z_p \\ y \end{array} \right] = \left[\begin{array}{c|ccc} A & B_u & B_p & B \\ \hline C_u & D_{uu} & D_{up} & E_u \\ C_p & D_{pu} & D_{pp} & E_p \\ C & F_u & F_p & F_3 \end{array} \right] \left[\begin{array}{c} x \\ w_u \\ w_p \\ u \end{array} \right] \tag{34}$$

Theorem 2 does indeed require that $F_3 = 0$. This problem can be overcome by finding a controller $\tilde{K}(s, \Delta_c)$ for the corresponding system with $F_3 = 0$ and then transforming the controller into another controller $K(s, \Delta_c)$ yielding the same closed loop system for the actual system. Denote the system matrix in (34) by M_s. Define

$$\Gamma = \begin{bmatrix} 0 & I \\ I & -F_3 \end{bmatrix}, \quad \Gamma^{-\star} = \begin{bmatrix} 0 & I \\ I & F_3 \end{bmatrix}$$

and observe that $\Gamma \star \Gamma^{-\star} = \Gamma^{-\star} \star \Gamma = \begin{bmatrix} 0 & I \\ I & 0 \end{bmatrix}$ is the Redheffer star product identity, and that

$$M_p = M_s \star \Gamma = \left[\begin{array}{c|ccc} A & B_u & B_p & B \\ \hline C_u & D_{uu} & D_{up} & E_u \\ C_p & D_{pu} & D_{pp} & E_p \\ C & F_u & F_p & 0 \end{array} \right].$$

Now assume that a controller, $\tilde{K}$, has been obtained for the system defined by M_p, and write this controller as

$$\begin{bmatrix} u \\ \dot{x}_c \\ z_c \end{bmatrix} = \tilde{K}_c \begin{bmatrix} y \\ x_c \\ w_c \end{bmatrix}, \quad \tilde{K}_c = \begin{bmatrix} D_{c1} & C_{c1} & D_{c12} \\ B_{c1} & A_c & B_{c2} \\ D_{c21} & C_{c2} & D_{c2} \end{bmatrix}.$$

Then, assuming that $I + F_3 \tilde{K}_c$ is non-singular, the closed loop is given by

$$\begin{bmatrix} \dot{x} \\ z_u \\ z_p \\ \dot{x}_c \\ z_c \end{bmatrix} = M_c \begin{bmatrix} x \\ w_u \\ w_p \\ x_c \\ w_c \end{bmatrix}$$

in which

$$\begin{aligned} M_c &= M_p \star \tilde{K}_c = M_p \star (\Gamma^{-\star} \star \Gamma) \star \tilde{K}_c \\ &= (M_p \star \Gamma^{-\star}) \star (\Gamma \star \tilde{K}_c) = M_s \star (\Gamma \star \tilde{K}_c). \end{aligned}$$

A controller for the system defined by M_s is thus given by

$$\begin{bmatrix} u \\ \dot{x}_c \\ z_c \end{bmatrix} = K_c \begin{bmatrix} y \\ x_c \\ w_c \end{bmatrix}, \quad K_c = (\Gamma \star \tilde{K}_c). \tag{35}$$

To sum up, the synthesis progresses as follows. Assuming a solution X, Y, P, and $\tilde{P}$ to (28)–(30) has been found, it is possible to construct the extended multiplier P_e and the scheduling function Δ_c as given above. The Lyapunov matrix $\mathcal{X}$ is for instance calculated as

$$\mathcal{X} = \begin{bmatrix} X & I \\ I & (X - Y^{-1})^{-1} \end{bmatrix}.$$

Now τ in (22) is a linear function of the controller matrices (A_c, B_c, C_c, D_c), which means that (21) becomes a Quadratic Matrix Inequality (QMI) in (A_c, B_c, C_c, D_c). A solution method for the QMI problem can for instance be found in [18].

IV. Main Result

In Section II a method for rewriting a neural state space model as a linear fractional transformation was presented. Then, in Section III a series of results on how to design an LPV controller achieving a given performance for a linear parameter varying system was outlined. Comparing the LFT formulation of the neural state space model with the LPV system that forms the basis of the controller design, we realize that the controller design methodology is applicable to this type of model if we formulate it as a quasi-LPV system. The term quasi-LPV system refers to a system for which the parameter variation is dependent on the current states and inputs, i.e. the nonlinear part of the system is of the form $\alpha(\zeta) = \beta(\zeta)\zeta$. The state-dependent matrix function $\beta(\cdot)$ is then simply considered as a time-varying matrix, as in (15). In other words—using the terminology defined in Section II— we consider the gains of $\Omega(\xi(t))$ to be the time-varying residual gains in the controller synthesis, i.e.

$$\Delta(t)\xi(t) = \Omega(\xi(t))$$

and disregard the explicit state dependency in the controller design. It is noted that the nonlinear mapping $\Omega(\cdot)$ is diagonal.

Theorem 2 shows that if we can find a set of controller matrices and multipliers that fulfill (28)–(30) for the LFT model, then Theorem 1 ensures that the closed loop is stable and achieves a specified performance. Logically, the next step will therefore be to construct a quasi-LPV controller based on a sufficiently trained MLP model of a given nonlinear plant.

However, Theorem 2 assumes (near-)perfect knowledge of the states and the nonlinear mapping, a knowledge that is rarely available in realistic situations. In the face of noisy in- and output measurements and a possibly imperfect model, it becomes necessary to consider the robustness of the control loop.

If $\Delta(t)$ depends only on measured quantities, as in the case of the MLP considered in Section II, the sensitivity to measurement noise translates into an uncertainty on the knowledge of $\Delta(t)$, but does not affect the system description (11). Hence, instead of ensuring (20), we need to fulfill

$$\begin{bmatrix} \Delta & 0 \\ 0 & \Delta_c(\hat{\Delta}) \\ \hline I & 0 \\ 0 & I \end{bmatrix}^T P_e \begin{bmatrix} \Delta & 0 \\ 0 & \Delta_c(\hat{\Delta}) \\ \hline I & 0 \\ 0 & I \end{bmatrix} > 0 \tag{36}$$

where $\hat{\Delta}$ is an estimate of Δ based on measurements. If this can be fulfilled for all $(\Delta, \hat{\Delta})$, then the quasi-LPV controller will stabilise the system and achieve the required robust quadratic performance.

The problem that will be addressed in this section, and the main contribution of this paper, is thus to find additional constraints on the multipliers P and $\tilde{P}$ such that we can satisfy (36) even in the presence of uncertainty on $\Delta(t)$. As mentioned earlier, Ω is diagonal, which enables us to choose $\Delta(t)$ as a diagonal, nonlinear matrix function. To simplify the derivations, we will only consider constant, diagonal multipliers.

The main result of this work can be summarised as in Theorem 3. In Section V we will discuss how to solve the synthesis problem in practice.

Theorem 3: Consider the system (11) with

$$w_u = \Delta(t)z_u, \ \Delta \in \mathbf{\Delta},$$

where $\mathbf{\Delta} = \{\Delta : \Delta = \mathrm{diag}_{1 \leq i \leq l}\{\delta_i\}, |\delta_i| < 1\}$. Assume that the estimate $\hat{\Delta}(t)$ of Δ is accurate except for some bounded uncertainty, i.e. $\hat{\Delta}(t) = \Delta(t) + E(t), E = \mathrm{diag}_{1 \leq i \leq l}\{e_i\}, |e_i| < \bar{e}_i$. Consider P and $\tilde{P}$ partitioned as in (24)–(26). If there exist X, Y, P and $\tilde{P}$ fulfilling (28)–(30) with

$$Q = \mathrm{diag}_{1 \leq i \leq l}\{q_i\} < 0, R = \mathrm{diag}_{1 \leq i \leq l}\{r_i\} > 0, S = 0, \tag{37}$$

$$\tilde{Q} = \mathrm{diag}_{1 \leq i \leq l}\{\tilde{q}_i\} < 0, \tilde{R} = \mathrm{diag}_{1 \leq i \leq l}\{\tilde{r}_i\} > 0, \tilde{S} = 0 \tag{38}$$

$$q_i(1 + \bar{e}_i)^2 + r_i > 0, \tilde{q}_i^{-1} + \tilde{r}_i^{-1}(1 + \bar{e}_i)^2 > 0 \tag{39}$$

and if for any $1 \leq i \leq l$ for which $(q_i - \tilde{q}_i^{-1})(r_i - \tilde{r}_i^{-1}) < 0$

it additionally holds that

$$((1 + \bar{e}_i)^2 q_i + \tilde{r}_i^{-1})^2 (\tilde{q}_i^{-1} + \tilde{r}_i^{-1})(q_i + r_i) >$$
$$((1 + \bar{e}_i)q_i - \tilde{r}_i^{-1})^2 \bar{e}_i^2 (\tilde{r}_i^{-1} - r_i)(q_i - \tilde{q}_i^{-1}) \quad (40)$$

if $q_i > \tilde{q}_i^{-1}$ and $r_i < \tilde{r}_i^{-1}$ or

$$((1 + \bar{e}_i)^2 \tilde{q}_i^{-1} + r_i)^2 (q_i + r_i)(\tilde{q}_i^{-1} + \tilde{r}_i^{-1}) >$$
$$((1 + \bar{e}_i)\tilde{q}_i^{-1} - r_i)^2 \bar{e}_i^2 (r_i - \tilde{r}_i^{-1})(\tilde{q}_i^{-1} - q_i) \quad (41)$$

if $q_i < \tilde{q}_i^{-1}$ and $r_i > \tilde{r}_i^{-1}$, then there exists a controller $K(\hat{\Delta})$ on the form (16) with $w_c = \Delta_c(\hat{\Delta}(t))z_c(t)$ that yields the robust performance specified in (12).

Proof: It is first of all noted that (25) and (27) are implied by (37)–(39), and that the estimated residual gains $\hat{\Delta}$ only appear in the inequalities involving the extended multiplier P_e. In order to prove the result, we hence need to show that with the extra requirements given above, (36) can be fulfilled. If that can be shown, then Theorem 2 ensures the existence of the desired controller $K(\hat{\Delta})$. We construct N, P_e and $\Delta_c(\Delta)$ as in Lemma 1. However, U must be constructed in a particular way, which will be addressed below. Then (20) is satisfied, which is equivalent to

$$\begin{bmatrix} [\begin{smallmatrix}\Delta\\I\end{smallmatrix}]^T P [\begin{smallmatrix}\Delta\\I\end{smallmatrix}] & V_-\Delta_c + V_+ \\ \Delta_c^T V_-^T + V_+^T & \Delta_c^T N_-^{-1} \Delta_c + N_+^{-1} \end{bmatrix} > 0.$$

By a Schur argument, this is equivalent to

$$\begin{bmatrix} [\begin{smallmatrix}\Delta\\I\end{smallmatrix}]^T P [\begin{smallmatrix}\Delta\\I\end{smallmatrix}] & V_-\Delta_c + V_+ & 0 \\ \Delta_c^T V_-^T + V_+^T & N_+^{-1} & \Delta_c^T \\ 0 & \Delta_c & -N_- \end{bmatrix} > 0.$$

Via a congruence transformation, this expression can be rewritten as

$$\begin{bmatrix} [\begin{smallmatrix}\Delta\\I\end{smallmatrix}]^T P [\begin{smallmatrix}\Delta\\I\end{smallmatrix}] - V_-N_-V_-^T & V_+ & V_-N_- \\ V_+^T & N_+^{-1} & \Delta_c^T \\ N_-V_-^T & \Delta_c & -N_- \end{bmatrix} > 0. \quad (42)$$

Furthermore, since P fulfills (25) or equivalently

$$[\begin{smallmatrix}\Delta\\I\end{smallmatrix}]^T P [\begin{smallmatrix}\Delta\\I\end{smallmatrix}] = [\begin{smallmatrix}\Delta\\I\end{smallmatrix}]^T P [\begin{smallmatrix}\Delta\\I\end{smallmatrix}] - V_-N_-V_-^T + V_-N_-V_-^T > 0$$

we have by Schur complement that

$$\begin{bmatrix} [\begin{smallmatrix}\Delta\\I\end{smallmatrix}]^T P [\begin{smallmatrix}\Delta\\I\end{smallmatrix}] - V_-N_-V_-^T & V_-N_- \\ N_-V_-^T & -N_- \end{bmatrix} > 0.$$

This implies that we can apply the Schur complement to (42) and obtain the equivalent inequality

$$\begin{bmatrix} N_+^{-1} & \Delta_c^T \\ \Delta_c & -N_- \end{bmatrix} - \Pi \left([\begin{smallmatrix}\Delta\\I\end{smallmatrix}]^T P [\begin{smallmatrix}\Delta\\I\end{smallmatrix}] - V_-N_-V_-^T \right)^{-1} \Pi^T > 0 \quad (43)$$

where $\Pi = \begin{bmatrix} V_+ & V_-N_- \end{bmatrix}^T$. With the diagonal structure of the multiplier, we can define the matrix

$$D = \Delta Q \Delta + R - V_-N_-V_-^T$$

and write (43) as

$$\begin{bmatrix} N_+^{-1} & \Delta_c^T \\ \Delta_c & -N_- \end{bmatrix} - \Pi D^{-1} \Pi^T > 0$$

and (33) as $\Delta_c(\Delta) = N_-V_-^T D^{-1} V_+$, respectively. We will now choose U in the following way. Let $U = \begin{bmatrix} T_1 & T_2 \end{bmatrix}$ such that $V_- = \begin{bmatrix} \Delta & I \end{bmatrix} T_1$ and $V_+ = \begin{bmatrix} \Delta & I \end{bmatrix} T_2$. If necessary we can perturb $\tilde{P}$ such that it is nonsingular. Since the columns of U form an orthogonal basis of the image of $P - \tilde{P}^{-1}$ (if this matrix happens to be singular, we can again perturb $\tilde{P}$ such that it is nonsingular as well), it is possible to partition it such that

$$U = \begin{bmatrix} T_1 & | & T_2 \end{bmatrix} = \begin{bmatrix} T_{1u} & 0 & | & T_{2u} & 0 \\ 0 & T_{1l} & | & 0 & T_{2l} \end{bmatrix}$$

in which the number of rows of U is $2l$ and the upper and lower parts each have l rows, and where each column contains exactly one 1 and $2l - 1$ zeros. If L_1 is some $2l \times 2l$ diagonal matrix then the product $T_1^T L_1 T_1$ is a diagonal matrix with the elements of L corresponding to the negative entries in $P - \tilde{P}^{-1}$ in its main diagonal. Similarly, if L_2 is some diagonal matrix of appropriate dimensions then the product $T_1 L_2 T_1^T$ is a diagonal $2l \times 2l$ matrix with zero entries everywhere except for the entries corresponding to the negative entries in $P - \tilde{P}^{-1}$. T_2 has the corresponding effect for the positive entries in $P - \tilde{P}^{-1}$.

Tedious calculations based on equations (31) and (32) show that D is of the form

$$D = \operatorname*{diag}_{1 \le i \le l} \left\{ \delta_i^2 \max\{q_i, \tilde{q}_i^{-1}\} + \max\{r_i, \tilde{r}_i^{-1}\} \right\} \quad (44)$$

since U rearranges the negative and positive diagonal elements of $P - \tilde{P}^{-1}$ into N_- and N_+, respectively, which means that N_- contains exactly those elements where $q_i - \tilde{q}_i^{-1} < 0, r_i - \tilde{r}_i^{-1} < 0$. With perfect knowledge about Δ it is then easy to choose $\Delta_c(\Delta)$ such that (43) is fulfilled, for instance as in (33) where the off-diagonal blocks are made to vanish, leaving a positive definite block diagonal matrix on the LHS.

However, as stated above we are not scheduling the controller based on the exact Δ, but rather on the estimate $\hat{\Delta}$. This prompts us to define the diagonal matrices $\hat{D}, \hat{V}_-$ and $\hat{V}_+$ analogously with (44) and (32) (replacing δ_i with $\hat{\delta}_i$) and rewrite (43) with $\Delta_c(\hat{\Delta}) = N_-\hat{V}_-(\hat{\Delta})^T \hat{D}(\hat{\Delta})^{-1} \hat{V}_+(\hat{\Delta})$ instead of $N_-V_-(\Delta)^T D(\Delta)^{-1} V_+(\Delta)$:

$$\begin{bmatrix} N_+^{-1} & \hat{V}_+^T \hat{D}^{-1} \hat{V}_- N_- \\ N_- \hat{V}_-^T \hat{D}^{-1} \hat{V}_+ & -N_- \end{bmatrix} -$$
$$\begin{bmatrix} V_+^T D^{-1} V_+ & V_+^T D^{-1} V_- N_- \\ N_- V_-^T D^{-1} V_+ & N_- V_-^T D^{-1} V_- N_- \end{bmatrix} > 0. \quad (45)$$

Let $\tilde{D}$ denote the matrix

$$\tilde{D} = \begin{bmatrix} \hat{\Delta} \\ I \end{bmatrix} \hat{D}^{-1} \begin{bmatrix} \hat{\Delta} \\ I \end{bmatrix}^T - \begin{bmatrix} \Delta \\ I \end{bmatrix} D^{-1} \begin{bmatrix} \Delta \\ I \end{bmatrix}^T.$$

This allows us to rewrite (45) as

$$\begin{bmatrix} T_2^T(N^{-1} - [\frac{\Delta}{I}]D^{-1}[\frac{\Delta}{I}]^T)T_2 & T_2^T\tilde{D}T_1 T_1^T N T_1 \\ T_1^T N T_1 T_1^T \tilde{D}T_2 & -N_-(N_-^{-1} + V_-^T D^{-1} V_-)N_- \end{bmatrix} > 0.$$

Some straightforward computations reveal that $\tilde{D}$ consists of diagonal submatrices

$$\tilde{D} = \begin{bmatrix} \tilde{D}_{11} & \tilde{D}_{12} \\ \tilde{D}_{12} & \tilde{D}_{22} \end{bmatrix}$$

given by

$$\tilde{D}_{11} = \operatorname*{diag}_{1 \le i \le l} \left\{ \frac{(\hat{\delta}_i^2 - \delta_i^2)r_{mi}}{(\hat{\delta}_i^2 q_{mi} + r_{mi})(\delta_i^2 q_{mi} + r_{mi})} \right\}$$

$$\tilde{D}_{12} = \operatorname*{diag}_{1 \le i \le l} \left\{ \frac{(\hat{\delta}_i \delta_i^2 - \hat{\delta}_i^2 \delta_i)q_{mi} + (\hat{\delta}_i - \delta_i)r_{mi}}{(\hat{\delta}_i^2 q_{mi} + r_{mi})(\delta_i^2 q_{mi} + r_{mi})} \right\}$$

$$\tilde{D}_{22} = \operatorname*{diag}_{1 \le i \le l} \left\{ \frac{(\delta_i^2 - \hat{\delta}_i^2)q_{mi}}{(\hat{\delta}_i^2 q_{mi} + r_{mi})(\delta_i^2 q_{mi} + r_{mi})} \right\}$$

where $q_{mi} = \max\{q_i, \tilde{q}_i^{-1}\}$ and $r_{mi} = \max\{r_i, \tilde{r}_i^{-1}\}, 1 \le i \le l$.

Applying the Schur complement to the inequality above and simplifying gives the following equivalent matrix inequality:

$$T_2^T \left(N^{-1} - [\tfrac{\Delta}{I}]D^{-1}[\tfrac{\Delta}{I}]^T \right) T_2$$
$$+ T_2^T \tilde{D}T_1 \left(N_-^{-1} + V_-^T D^{-1} V_- \right)^{-1} T_1^T \tilde{D}T_2 > 0. \quad (46)$$

Let $G_- = N^{-1} - [\frac{\Delta}{I}]D^{-1}[\frac{\Delta}{I}]^T$ and $G_+ = N^{-1} + [\frac{\Delta}{I}]D^{-1}[\frac{\Delta}{I}]^T$, such that (46) can be written as

$$T_2^T G_- T_2 + T_2^T \tilde{D}T_1(T_1^T G_+ T_1)^{-1}T_1^T \tilde{D}T_2 > 0 \quad (47)$$

in which, using (44), it is seen that G_- and G_+ must be of the form

$$G_- = \begin{bmatrix} G_{-11} & G_{-12} \\ G_{-12} & G_{-22} \end{bmatrix} \text{ and } G_+ = \begin{bmatrix} G_{+11} & G_{+12} \\ G_{+12} & G_{+22} \end{bmatrix}$$

where

$$G_{\pm 11} = \operatorname*{diag}_{1 \le i \le l} \left\{ \frac{1}{q_i - \tilde{q}_i^{-1}} \pm \frac{\delta_i^2}{\delta_i^2 q_{mi} + r_{mi}} \right\} \quad (48)$$

$$G_{\pm 12} = \operatorname*{diag}_{1 \le i \le l} \left\{ \pm \frac{\delta_i}{\delta_i^2 q_{mi} + r_{mi}} \right\} \quad (49)$$

$$G_{\pm 22} = \operatorname*{diag}_{1 \le i \le l} \left\{ \frac{1}{r_i - \tilde{r}_i^{-1}} \pm \frac{1}{\delta_i^2 q_{mi} + r_{mi}} \right\}. \quad (50)$$

Now Lemma 2 (in the Appendix) implies that

$$T_2^T \tilde{D}T_1(T_1^T G_+ T_1)^{-1}T_1^T \tilde{D}T_2 = T_2^T \Lambda T_2$$

where

$$\Lambda = \begin{bmatrix} \Lambda_1 & 0 \\ 0 & \Lambda_2 \end{bmatrix}$$

which means that (47) is equivalent to

$$T_2^T(G_- + \Lambda)T_2 > 0. \quad (51)$$

Λ is diagonal and $\lambda_{1,i} = \lambda_{2,i} = 0$ for the i's for which $q_i > \tilde{q}_i^{-1}$ and $r_i > \tilde{r}_i^{-1}$. We also know from Lemma 2 that $\Lambda_{1,i} = \tilde{d}_{12,i}^2/g_{+22,i}$ for the i's for which $q_i > \tilde{q}_i^{-1}$ and $r_i < \tilde{r}_i^{-1}$ and $\Lambda_{i+l,i+l} = \tilde{d}_{12,i}^2/g_{+11,i}$ for the i's for which $q_i > \tilde{q}_i^{-1}$ and $r_i < \tilde{r}_i^{-1}$ (lower-case letters with subscript i refer to the i'th diagonal element of the matrix denoted by the corresponding upper-case letter). Furthermore, the pre- and postmultiplication by T_2^T and T_2, respectively, eliminates the elements for which $q_i < \tilde{q}_i^{-1}$ and $r_i < \tilde{r}_i^{-1}$.

By a permutation (51) can then be seen to be equivalent to the fulfilment of a number of 1×1 or 2×2 matrix inequalities of the form

$$q_i > \tilde{q}_i^{-1}, r_i > \tilde{r}_i^{-1} \quad : \quad \begin{bmatrix} g_{-11,i} & g_{-12,i} \\ g_{-12,i} & g_{-22,i} \end{bmatrix} +$$
$$\begin{bmatrix} \lambda_{1,i} & 0 \\ 0 & \lambda_{2,i} \end{bmatrix} > 0 \quad (52)$$

$$q_i > \tilde{q}_i^{-1}, r_i < \tilde{r}_i^{-1} \quad : \quad g_{-11,i} + \lambda_{1,i} > 0 \quad (53)$$

$$q_i < \tilde{q}_i^{-1}, r_i > \tilde{r}_i^{-1} \quad : \quad g_{-22,i} + \lambda_{2,i} > 0. \quad (54)$$

As mentioned above we have $\lambda_{1,i} = \lambda_{2,i} = 0$ for the i's for which $q_i > \tilde{q}_i^{-1}, r_i > \tilde{r}_i^{-1}$. Furthermore, the submatrix of G_- can be seen to be positive definite by combining equations (48)–(50) with the basic assumptions (39), which imply that $\delta^2 q_i + r_i > 0, \delta^2 \tilde{q}_i^{-1} + \tilde{r}_i^{-1} < 0$. Hence, (52) is automatically satisfied.

This leaves us with (53) and (54), which represent a set of simple scalar inequalities. By combining (48) and (50) with the definition of $\tilde{D}_{12}$ we can rewrite these inequalities as

$$\frac{1}{q_i - \tilde{q}_i^{-1}} - \frac{\delta_i^2}{\delta_i^2 q_i + \tilde{r}_i^{-1}} + \mu_1 \left(\frac{1}{r_i - \tilde{r}_i^{-1}} + \frac{1}{\delta_i^2 q_i + \tilde{r}_i^{-1}} \right)^{-1} > 0$$

for $q_i > \tilde{q}_i^{-1}, r_i < \tilde{r}_i^{-1}$, and

$$\frac{1}{r_i - \tilde{r}_i^{-1}} - \frac{\delta_i^2}{\delta_i^2 \tilde{q}_i^{-1} + r_i} + \mu_2 \left(\frac{1}{q_i - \tilde{q}_i^{-1}} + \frac{1}{\delta_i^2 \tilde{q}_i^{-1} + r_i} \right)^{-1} > 0$$

for $q_i < \tilde{q}_i^{-1}, r_i > \tilde{r}_i^{-1}$, in which

$$\mu_1 = \frac{\delta_i \hat{\delta}_i(\delta_i - \hat{\delta}_i)q_i + (\hat{\delta}_i - \delta_i)\tilde{r}_i^{-1}}{(\hat{\delta}_i^2 q_i + \tilde{r}_i^{-1})(\delta_i^2 q_i + \tilde{r}_i^{-1})}$$

$$\mu_2 = \frac{\delta_i \hat{\delta}_i(\delta_i - \hat{\delta}_i)\tilde{q}_i^{-1} + (\hat{\delta}_i - \delta_i)r_i}{(\hat{\delta}_i^2 \tilde{q}_i^{-1} + r_i)(\delta_i^2 \tilde{q}_i^{-1} + r_i)}.$$

Finally, applying Lemma 3 to each of these inequalities shows that each of them are satisfied if

$$((1 + \bar{e}_i)^2 q_i + \tilde{r}_i^{-1})^2 (\tilde{q}_i^{-1} + \tilde{r}_i^{-1})(q_i + r_i) -$$
$$((1 + \bar{e}_i)q_i - \tilde{r}_i^{-1})^2 \bar{e}_i^2 (\tilde{r}_i^{-1} - r_i)(q_i - \tilde{q}_i^{-1}) > 0$$

if $q_i > \tilde{q}_i^{-1}$ and $r_i < \tilde{r}_i^{-1}$ or

$$((1 + \bar{e}_i)^2 \tilde{q}_i^{-1} + r_i)^2 (q_i + r_i)(\tilde{q}_i^{-1} + \tilde{r}_i^{-1}) >$$
$$((1 + \bar{e}_i)\tilde{q}_i^{-1} - r_i)^2 \bar{e}_i^2 (r_i - \tilde{r}_i^{-1})(\tilde{q}_i^{-1} - q_i)$$

if $q_i < \tilde{q}_i^{-1}$ and $r_i > \tilde{r}_i^{-1}$. Hence, (51) will be fulfilled if (40) respectively (41) are satisfied, which was what we wanted to show. ∎

Remark 4 Normally, if $P - \tilde{P}^{-1}$ loses rank, it would be more efficient to construct an extended multiplier of lower dimension. However, to keep the proof simple, it was chosen to ignore this possibility, since there is no loss of generality in assuming that $P - \tilde{P}^{-1}$ is indeed invertible. If necessary, it is always possible (due to the strictness of the matrix inequalities) to perturb $P - \tilde{P}^{-1}$ in the right direction, such that there is no need to schedule according to the particular diagonal elements which are the cause of loss of rank, i.e. Δ_c will be independent of these elements. ◁

V. Implementing the Controller

After having formulated the test for existence of a robust quasi-LPV controller in the previous section, we will now give an outline of an implementation method. One important point is that in order to implement the continuous time controller designed by the methods in Sections III and IV in a computerised control loop, a discrete time version of the controller is needed.

1. *Modelling:* Obtain samples of in- and output signals from the plant and train an MLP model of the plant. This will typically take place in discrete time.

2. *Model transformation:* Transform the MLP model into a (discrete-time) LFT model and calculate the bounds on the uncertainties on Δ as suggested in Section II. Then use some appropriate method to transform the discrete-time LFT model into its continuous-time equivalent, yielding the system (11).

3. *Performance specification:* Define an appropriate performance specification, for instance

$$P_p = \begin{bmatrix} -\gamma I & 0 \\ 0 & \frac{1}{\gamma} I \end{bmatrix} \tag{55}$$

providing a bound γ on the induced 2-norm

$$\sup_{w \neq 0} \frac{\|z\|_2}{\|w\|_2} \leq \gamma.$$

4. *Synthesis:* Test if the LMIs (28)–(30) can be fulfilled with the additional constraints (40) and (41). If they are not immediately fulfilled, it can be attempted to impose positive definite diagonal margins ϵ and $\tilde{\epsilon}$ on the multiplier matrices

$$-(I + \epsilon)Q > R$$
$$-(I + \tilde{\epsilon})\tilde{R} > \tilde{Q}$$

and increase the individual elements of ϵ and $\tilde{\epsilon}$ until the corresponding constraints are fulfilled or the LMI becomes infeasible. There is no guarantee that either of these events

will happen, but the method appears to work well in practice. Alternatively, it can be shown that by restricting the product $r_i \tilde{r}_i^{-1}$ to some interval $r_l < r_i \tilde{r}_i^{-1} < r_u$, it can be guaranteed that one of them will happen as the elements of ϵ are increased. However, this will typically yield slightly more conservative results.

5. *Controller implementation:* Once the controller matrices have been found from the solutions of the synthesis LMIs and the controller scheduling function has been determined, it is then possible to transform the controller back to discrete time and implement it. Discretising the controller by for instance Tustin's method (see e.g. [6]) will yield a discrete time controller on the form

$$\begin{bmatrix} x_{c,k+1} \\ u_k \\ z_{c,k} \end{bmatrix} = \begin{bmatrix} A_{cd} & B_{c1d} & B_{c2d} \\ C_{c1d} & D_{c1d} & D_{c12d} \\ C_{c2d} & D_{c21d} & D_{c2d} \end{bmatrix} \begin{bmatrix} x_{c,k} \\ y_k \\ w_{c,k} \end{bmatrix}$$

with

$$w_{c,k} = \Delta_{c,k} z_{c,k}.$$

Implementation-wise this poses two algebraic loop problems. Firstly, $w_{c,k}$ depends on $z_{c,k}$ and vice versa, and secondly, $\Delta_{c,k}$ depends on $\hat{\Delta}_k$ which—in general—depends on u_k, which, if D_{c12d} is non-zero, in turn depends on $\Delta_{c,k}$. The first problem can be easily resolved by a method similar to the one discussed at the end of Section III. The same behaviour will be obtained by using the transformed scheduling function $\bar{\Delta}_{c,k} = (I - \Delta_{c,k} D_{c2d})^{-1} \Delta_{c,k}$ and a controller with the same parameters except for a zero matrix in the place of D_{c2d}.

The second problem is harder to overcome. If the sampling rate is sufficiently high, and the controller has a reasonably low high-frequency gain, then the control signal can be expected to change only slightly from sample to sample, and u_{k-1} can be used as an estimate of u_k in computing $\hat{\Delta}_k$. Alternatively, an iterative scheme could be used to alternately compute u_k and $\hat{\Delta}_k$ in an algebraic loop until the results (hopefully) converge. This will rarely be a good alternative to just increasing the sample rate, however.

Remark 5 The transformation back and forth between continuous and discrete time introduces a potential source of errors. Should a discrete-time version of Theorem 2 be developed, it would most likely be a good idea to employ this. However, Theorem 3 would probably still be applicable in its present form, since Δ is a memoryless mapping and the multipliers would be of the same form. Another interesting—although not entirely realistic—option within the permissible scope of the method would be to train the MLP in continuous time, since the transformation of the state space model from MLP to LFT form does not distinguish between continuous-time and discrete-time models. ◁

VI. Simulation Examples

In the following we will illustrate the LPV control design procedure on a nonlinear and noisy simulation process. Inspired by [3] we chose the third order continuous-time pro-

cess

$$\dot{x}_1 = x_2 - x_1 \tag{56}$$

$$\dot{x}_2 = x_3 - x_2 \tag{57}$$

$$\dot{x}_3 = \frac{x_1^2 + x_2^2 + x_3^2 + \tanh(u)}{x_1^2 + x_2^2 + x_3^2 + 1} - x_3 \tag{58}$$

$$y = x_1 \tag{59}$$

as our example plant and simulated it with a higher-order differential equation solver. The input was chosen as a discrete-time sequence of samples with a sample time of $1/2$ second. The input samples were generated by a simple controller that was implemented to drive the system around in the operating range. As the output was sampled, uniform noise in the interval $[-\sigma_n; \sigma_n], \sigma_n = 0.05$, was added to the output. The plant was sampled at a sampling frequency of $2\,Hz$, and an MLP with $l = 5\,\tanh(\cdot)$ neurons in the hidden layer and three linear neurons in the output layer was trained to recognise the model

$$\hat{x}_{k+1} = W_o \sigma(\xi_k + W_b)$$
$$\hat{y}_k = \hat{x}_{1,k}$$
$$\xi_k = W_i \zeta_k$$

with $\zeta_k = [y_k,\ y_{k-1},\ y_{k-2}, u_k,\ u_{k-1},\ u_{k-2}]^T$ and $\hat{x}_k = [\hat{y}_k,\ \hat{y}_{k-1},\ \hat{y}_{k-2}]^T$ ($\hat{\ }$ denotes estimates). The MLP was trained using the Levenberg-Marquardt algorithm. For comparison, a linear model was identified using system identification based on the same in- and output samples. Figure 3 shows a simulation of a separately generated test set. As can be seen from the figure, the MLP learned the plant behaviour satisfyingly well, and as expected, simulated the nonlinear plant far better than the linear model.
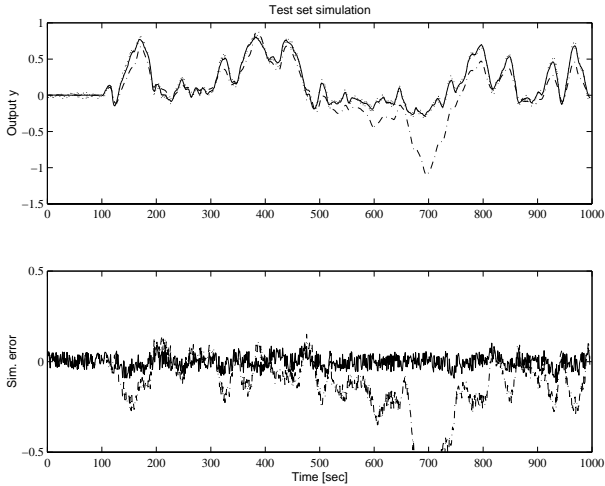

Fig. 3. *Test set simulation. The plant output is shown with a dotted line $(\cdots)$, the MLP output with a full line (—) and the linear model output with a dash-dotted line $(-\cdot-)$. Linear system identification is insufficient to identify the behaviour, as the plant is highly nonlinear.*

Next, the transformation from MLP form to LFT form described in Section II was performed. This gave the model

$$\hat{x}_{k+1} = A\hat{x}_k + Bu_k + B_1\Omega(\xi_k), \quad \hat{y}_k = C\hat{x}_k$$

where ξ_k denotes the input to the nonlinearity at sample k. The sector bounds on the nonlinearities were found to $K_{max} - K_{min} = \mathrm{diag}\{0.003, 0.005, 0.06, 0.23, 0.45\}$ and the bounds on the uncertainty on the gains of Ω were found to $\bar{e}_i = \{0.24, 0.13, 0.30, 0.17, 0.12\}, 1 \le i \le l$.

We let the outputs from the nonlinearity be denoted by α, the performance channel by z and the noise and reference channel by w, and constructed the LTI system

$$\begin{bmatrix} \hat{x}_{k+1} \\ \hline \xi_k \\ z_k \\ \hat{y}_k \end{bmatrix} = \begin{bmatrix} A & B_1 & 0 & B \\ \hline W_x & 0 & 0 & W_u \\ C & 0 & D_{22} & 0 \\ C & 0 & D_{32} & 0 \end{bmatrix} \begin{bmatrix} \hat{x}_k \\ \hline \alpha_k \\ w_k \\ u_k \end{bmatrix}.$$

$D_{22} = [0 \quad -I]$ and $D_{32} = [\sigma_n I \quad -I]$ defined the noise and reference contributions. The performance output was then augmented by a first-order filter that allowed frequency tuning of the controller; in this example we placed more emphasis on the low-frequency behaviour by placing the pole in $z = 0.99$. This system was then transformed into its continuous-time equivalent using Tustin's method in order to carry out the controller computation.

The next step was to solve the LMIs (28)–(30) in order to compute a nominal controller. The performance specification (55) was chosen and a bisectional search for the smallest γ for which (28)–(30) were feasible could then be performed. Once a non-robust controller was found (referred to as K_1) the controller calculations were repeated, but this time with the added constraints (40) and (41). The method proposed in Section V, where the diagonal elements of ϵ and $\tilde{\epsilon}$ are increased until the constraints are fulfilled, was employed. This robust controller is referred to as K_2. In this case it was necessary to increase γ slightly. Finally, a non-robust controller with the same γ as achieved by K_2 was calculated for comparison purposes. This detuned non-robust controller will be referred to as K_3. In the example the following performance values were achieved: $\gamma_1 = 0.062$ and $\gamma_2 = \gamma_3 = 0.088$. Each of the controllers were then discretised with a sample time of $1/2$ second.

Figure 4 shows a series of closed loop simulations for each of the control loop configurations with identical reference signals and random noise added to the output y. For the sake of clarity, however, the output is shown without the noise. The top and middle plots in the figure show simulations with the non-robust (K_1) and the robust (K_2) LPV controllers. The simulations are identical, except for the random measurement noise; the robust controller shows the same behaviour in both simulations, while the non-robust controller shows much greater sensitivity to the noise. This is particularly evident from the control signal, but it can also be seen in the output, which shows a tendency toward becoming unstable in the top plot. In the middle plot, on the other hand, the closed loop does not exhibit this tendency. The bottom plots show the robust LPV controller (K_2) versus the non-robust detuned controller (K_3). It is seen that the steady-state performance, at least, is better for K_2.
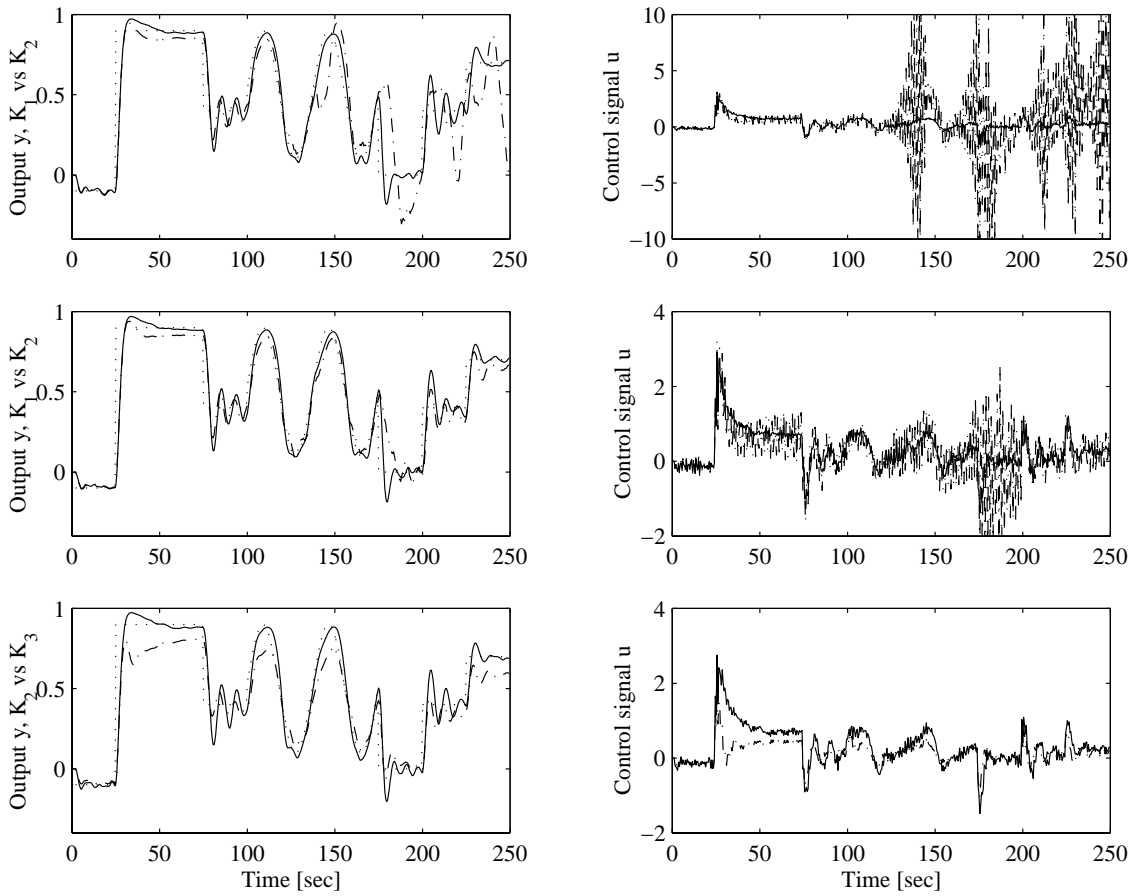
Fig. 4. *Closed loop simulation. The reference is shown with a dotted line ($\cdots$), the signals generated by the robust controller with a full line (——) and the signals generated by the non-robust controllers with a dash-dotted line ($-\cdot-$). The top and middle figures show simulations with the nominal and the robust LPV controllers. The simulations are identical, except for the random measurement noise; K_2 shows the same behaviour in both simulations, while K_1 shows much greater sensitivity to the noise. The bottom row shows the robust LPV controller versus the non-robust detuned controller. It is seen that the steady-state performance is better for K_2.*

VII. CONCLUDING REMARKS

In this paper a novel method that combines the use of feedforward neural networks with quasi-LPV control synthesis for control of nonlinear, noisy systems has been proposed. This method involves a separation of the nonlinear model into a linear and diagonal nonlinear part for which a pointwise bound is known. Hereafter, a controller is constructed such that one of its inputs is generated by a scheduling function derived from the nonlinear part of the model. The synthesis of the controller and scheduling function is achieved by solving a set of linear matrix inequalities constructed from the model parameters. The main contribution of this paper was to derive additional constraints on the scheduling function synthesis matrix inequalities, such that the control loop is robust to bounded measurement noise.

It was also discussed how to implement the control design on a digital computer, and the proposed quasi-LPV controller's robustness to noise-induced uncertainty was confirmed on a nonlinear, noisy simulation example.

Since the class of nonlinear systems that can be modelled well by state space models of the above-mentioned type is large, the proposed method is considered to be widely ap-

plicable. However, it does not incorporate any information on how fast the gains of the nonlinearity change over time, which means that less conservative controllers could potentially be constructed for a given plant. Future research in this area should therefore focus on how to solve this problem, as well as trying out the method on real-life applications.

APPENDIX

Lemma 2: Consider the index sets $\mathcal{I}_-, \mathcal{I}_+, \mathcal{J}_-$ and $\mathcal{J}_+$ with cardinalities $n_I, l - n_I, n_J$ and $l - n_J$, respectively, defined such that $\mathcal{I}_- \cup \mathcal{I}_+ = \mathcal{J}_- \cup \mathcal{J}_+ = \{1, \ldots, l\} \subset \mathbb{N}$. Let $e_i, 1 \le i \le l$, denote the i'th unit coordinate vector of $\mathbb{R}^l$. Let $T_1 = \begin{bmatrix} T_{1u} & 0 \\ 0 & T_{1l} \end{bmatrix} \in \mathbb{R}^{2l \times n_I + n_J}$ and $T_2 = \begin{bmatrix} T_{2u} & 0 \\ 0 & T_{2l} \end{bmatrix} \in \mathbb{R}^{2l \times 2l - (n_I + n_J)}$, where the columns of T_1 and T_2 are unit coordinate vectors of $\mathbb{R}^{2l}$, be defined by

$$T_{1u}^T e_i = 0 \Leftrightarrow i \in \mathcal{I}_+, \quad T_{1l}^T e_i = 0 \Leftrightarrow i \in \mathcal{J}_+,$$
$$T_{2u}^T e_i = 0 \Leftrightarrow i \in \mathcal{I}_-, \quad T_{2l}^T e_i = 0 \Leftrightarrow i \in \mathcal{J}_-.$$

Furthermore, let D and G be any two matrices such that $D = \begin{bmatrix} D_{11} & D_{12} \\ D_{12} & D_{22} \end{bmatrix} \in \mathbb{R}^{2l \times 2l}$ and $G = \begin{bmatrix} G_{11} & G_{12} \\ G_{12} & G_{22} \end{bmatrix} \in \mathbb{R}^{2l \times 2l}$ where each of the submatrices $D_{11}, D_{12}, \ldots, G_{22} \in \mathbb{R}^{l \times l}$ are diagonal.

Then, assuming that $T_1^T G T_1$ is invertible, we have

$$T_2^T D T_1 (T_1^T G T_1)^{-1} T_1^T D T_2 = T_2^T \Lambda T_2$$

where

$$\Lambda = \begin{bmatrix} \mathrm{diag}_{1 \le i \le l}\{\lambda_{1i}\} & 0 \\ 0 & \mathrm{diag}_{1 \le i \le l}\{\lambda_{2i}\} \end{bmatrix}.$$

Furthermore, the elements of Λ that do not vanish by the pre- and postmultiplication by T_2 are given by

$$\begin{aligned}
i \in \mathcal{I}_+ \cap \mathcal{J}_+ &\Rightarrow \lambda_{1i} = \lambda_{2i} = 0 \\
i \in \mathcal{I}_+ \cap \mathcal{J}_- &\Rightarrow \lambda_{1i} = d_i^2 / g_{2i} \qquad (60) \\
i \in \mathcal{I}_- \cap \mathcal{J}_+ &\Rightarrow \lambda_{2i} = d_i^2 / g_{1i}
\end{aligned}$$

in which d_i, g_{1i} and g_{2i} are the i'th diagonal elements of D_{12}, G_{11} and G_{22}, respectively.

Proof: First of all it is noticed that $T_2^T D T_1 = T_2^T \begin{bmatrix} 0 & D_{12} \\ D_{12} & 0 \end{bmatrix} T_1$ since

$$\begin{aligned}
\begin{bmatrix} T_{2u}^T & 0 \\ 0 & T_{2l}^T \end{bmatrix} &\begin{bmatrix} D_{11} & D_{12} \\ D_{12} & D_{22} \end{bmatrix} \begin{bmatrix} T_{1u} & 0 \\ 0 & T_{1l} \end{bmatrix} \\
&= \begin{bmatrix} T_{2u}^T D_{11} T_{1u} & T_{2u}^T D_{12} T_{1l} \\ T_{2l}^T D_{12} T_{1u} & T_{2l}^T D_{22} T_{1l} \end{bmatrix} \\
&= \begin{bmatrix} 0 & T_{2u}^T D_{12} T_{1l} \\ T_{2l}^T D_{12} T_{1u} & 0 \end{bmatrix}
\end{aligned}$$
$$(61)$$

because T_1 and T_2 do not have non-zero entries on the same rows and hence these particular products must vanish.

Let us define

$$\Gamma = T_1 (T_1^T G T_1)^{-1} T_1^T = \begin{bmatrix} \Gamma_{11} & \Gamma_{12} \\ \Gamma_{12} & \Gamma_{22} \end{bmatrix}$$

where $\Gamma_{11}, \Gamma_{12}, \Gamma_{22} \in \mathbb{R}^{l \times l}$. It is deduced that each of the submatrices Γ_{11}, Γ_{12} and Γ_{22} are diagonal with

$$\begin{aligned}
\Gamma_{11,i} &= g_{11,i}^{-1} & \text{for} \quad & i \in \mathcal{I}_- \cap \mathcal{J}_+ \\
\Gamma_{11,i} &= 0 & \text{for} \quad & i \in \mathcal{I}_+ \\
\Gamma_{22,i} &= g_{22,i}^{-1} & \text{for} \quad & i \in \mathcal{I}_+ \cap \mathcal{J}_- \qquad (62) \\
\Gamma_{22,i} &= 0 & \text{for} \quad & i \in \mathcal{J}_+ \\
\Gamma_{12,i} &= 0 & \text{for} \quad & i \in \mathcal{I}_+ \cup \mathcal{J}_+.
\end{aligned}$$

This is seen by noticing that $T_1^T G T_1$ is equivalent, via a permutation, to a block diagonal matrix where each subblock is either of dimension 2×2 arising from elements corresponding to $i \in \mathcal{I}_- \cap \mathcal{J}_-$, or 1×1 arising from elements corresponding to $i \in (\mathcal{I}_+ \cap \mathcal{J}_-) \cup (\mathcal{I}_- \cap \mathcal{J}_+)$. Matrix inversion preserves this equivalence, and pre- and postmultiplying by T_1 and T_1^T then produces a matrix where the newly formed elements are rearranged back to the corresponding positions in G.

In light of (61) it can then be seen that

$$\begin{aligned}
T_2^T D T_1 &(T_1^T G T_1)^{-1} T_1^T D T_2 = \\
&\begin{bmatrix} T_{2u}^T D_{12} \Gamma_{22} D_{12} T_{2u} & T_{2u}^T D_{12} \Gamma_{12} D_{12} T_{2l} \\ T_{2l}^T D_{12} \Gamma_{12} D_{12} T_{2u} & T_{2l}^T D_{12} \Gamma_{11} D_{12} T_{2l} \end{bmatrix}.
\end{aligned}$$

The off-diagonal blocks in this matrix can furthermore be seen to be zero, since $\Gamma_{12,i} = 0$ for $i \in \mathcal{I}_+ \cup \mathcal{J}_+$ and pre- and postmultiplying by T_{2l}^T and T_{2u} eliminates the elements corresponding to $i \in \mathcal{I}_- \cup \mathcal{J}_-$. That leaves us with

$$\begin{aligned}
T_2^T D T_1 &(T_1^T G T_1)^{-1} T_1^T D T_2 \\
&= \begin{bmatrix} T_{2u}^T D_{12} \Gamma_{22} D_{12} T_{2u} & 0 \\ 0 & T_{2l}^T D_{12} \Gamma_{11} D_{12} T_{2l} \end{bmatrix} \\
&= T_2^T \begin{bmatrix} D_{12} \Gamma_{22} D_{12} & 0 \\ 0 & D_{12} \Gamma_{11} D_{12} \end{bmatrix} T_2 \\
&= T_2^T \Lambda T_2.
\end{aligned}$$

Looking at the diagonal elements, we see that pre- and postmultiplying by T_{2u}^T and T_{2u} eliminates the elements corresponding to $i \in \mathcal{I}_-$, while pre- and postmultiplying by T_{2l}^T and T_{2l} eliminates the elements corresponding to $\mathcal{J}_-$. This implies that only those diagonal elements in $D_{12} \Gamma_{11} D_{12}$ and $D_{12} \Gamma_{22} D_{12}$ corresponding to $i \in \mathcal{J}_+$ and $i \in \mathcal{I}_+$, respectively, will not vanish by this operation. Since Γ_{11} and Γ_{22} have the structures given in (62) we deduce that the non-vanishing elements in $T_2^T \Lambda T_2$ must be of the form (60), which completes the proof. ∎

Lemma 3: Consider the inequality

$$\frac{1}{q - \tilde{q}^{-1}} - \frac{\delta^2}{\delta^2 q + \tilde{r}^{-1}} + \mu \left(\frac{1}{r - \tilde{r}^{-1}} + \frac{1}{\delta^2 q + \tilde{r}^{-1}} \right)^{-1} > 0$$
$$(63)$$

where $\mu = \frac{\delta \hat{\delta}(\delta - \hat{\delta})q + (\hat{\delta} - \delta)\tilde{r}^{-1}}{(\hat{\delta}^2 q + \tilde{r}^{-1})(\delta^2 q + \tilde{r}^{-1})}$. Assuming that $|\delta| < 1, |\delta - \hat{\delta}| < e, 0 > q > \tilde{q}^{-1}, (1 + e^2)q + r > 0$ and $\tilde{r}^{-1} > r > 0$, (63) is satisfied if

$$\begin{aligned}
((1 + e)^2 q + \tilde{r}^{-1})^2 (\tilde{q}^{-1} + \tilde{r}^{-1})(q + r) \\
> ((1 + e)q - r)^2 e^2 (\tilde{r}^{-1} - r)(q - \tilde{q}^{-1}).
\end{aligned}$$

Proof: Rewriting (63) as a single fraction gives

$$\frac{t_1 + t_2 + t_3}{(q - \tilde{q}^{-1})(\hat{\delta}^2 q + \tilde{r}^{-1})^2 (\delta^2 q + \tilde{r}^{-1})(\delta^2 q + r)} > 0$$

where

$$\begin{aligned}
t_1 &= (\hat{\delta}^2 q + \tilde{r}^{-1})^2 (\delta^2 q + \tilde{r}^{-1})(\delta^2 q + r) \\
t_2 &= -(q - \tilde{q}^{-1})(\hat{\delta}^2 q + \tilde{r}^{-1})^2 (\delta^2 q + r)\delta^2 \\
t_3 &= (\delta \hat{\delta}(\delta - \hat{\delta})q + (\hat{\delta} - \delta)r)^2 (r - \tilde{r}^{-1})(q - \tilde{q}^{-1}).
\end{aligned}$$

It is seen that the denominator is positive since $q > \tilde{q}^{-1}$ and $q + \tilde{r}^{-1} > q + r > 0$ by assumption. Furthermore it is obvious that the inequality is hardest to satisfy for $\delta \to 1$, so we will let $\delta = 1$. Similarly, the worst case for $\hat{\delta}$ is for $\hat{\delta} \to \delta + e$, so we will let $\hat{\delta} = 1 + e$ and examine the numerator inequality $t_1 + t_2 > -t_3$ or:

$$\begin{aligned}
((1 + e)^2 q + \tilde{r}^{-1})^2 ((q + \tilde{r}^{-1})(q + r) - (q - \tilde{q}^{-1})(q + r)) \\
> ((1 + e)q - r)^2 e^2 (\tilde{r}^{-1} - r)(q - \tilde{q}^{-1}). \quad (64)
\end{aligned}$$

In other words, if (64) is satisfied, then (63) will be satisfied as well. It is now easy to see that (64) can be simplified to

$$((1+e)^2 q + \tilde{r}^{-1})^2 (\tilde{q}^{-1} + \tilde{r}^{-1})(q + r)$$
$$> ((1+e)q - r)^2 e^2 (\tilde{r}^{-1} - r)(q - \tilde{q}^{-1})$$

which was what we wanted to show. ∎

REFERENCES

[1] J. Bendtsen, K. Trangbæk, "Transformation of Neural State Space Models into LFT Models for Robust Control Design," in *Proc. of the Sixth International Conference on Control, Automation, Robotics and Vision, Dec. 2000*

[2] S. Boyd, L. El Ghaoui, E. Feron, V. Balakrishnan, "Linear Matrix Inequalities in System and Control Theory," Philadelphia, PN: SIAM, 1994

[3] J. B. D. Cabrera, K. S. Narendra, "Issues in the Application of Neural Networks for Tracking Based on Inverse Control," *IEEE Transactions on Automatic Control* Vol 44, 11:2007–2027, 1999

[4] J. S. Chai, S. Tan; C. C. Hang, "Gain Scheduling Control of Nonlinear Plant Using a RBF Neural Network," in *Proc. of the IEEE International Symposium on Intelligent Control*, Sept. 1996

[5] F. C. Chen, H. K. Khalil, "Adaptive Control of a Class of Nonlinear Discrete-time Systems Using Neural Networks," *IEEE Transactions on Automatic Control* Vol 40, 9:791–801, 1995

[6] G. F. Franklin, J. D. Powell, M. Workman, "Digital Control of Dynamic Systems," 3rd Ed., Menlo Park, CA: Addison-Wesley, 1998

[7] M. French, E. Rogers, "Input/Output Stability Theory for Direct Neuro-Fuzzy Controllers," *IEEE Transactions on Fuzzy Systems* Vol. 6, 3:331–345, 1998

[8] S. He, K. Reif, R. Unbehauen, "A Neural Approach for Control of Nonlinear Systems With Feedback Linearization," *IEEE Transactions on Neural Networks* Vol. 9, 6:1409–1421, 1998

[9] A. Helmersson, "Methods for Robust Gain Scheduling," Ph.D. thesis, Linköping University, 1995

[10] Y. H. Kim, F. L. Lewis, C. T. Abdallah, "A Dynamic Recurrent Neural-network-based Adaptive Observer for a Class of Nonlinear Systems," *Automatica* Vol. 33, 8:1539–1543, 1997

[11] A. U. Levin, K. S. Narendra, "Control of Nonlinear Dynamical Systems Using Neural Network: Contollability and Stabilization," *IEEE Transactions on Neural Networks* Vol. 4, 2:192–205, 1993

[12] T. T. Lee, J. T. Jeng, C. L. Shih, "Using Neural Networks to Improve Gain Scheduling Techniques for Linear Parameter-varying Systems," in *Proc. of the International Workshop on Advanced Motion Control*, March 1996

[13] S. Lu, T. Basar, "Robust Nonlinear System Identification Using Neural Network Models," *IEEE Transactions on Neural Networks* 9:407–429, 1998

[14] M. J. Mears, M. M. Polycarpou, "Stable Local Neural Control of Uncertain Systems," in *Proc. of the IEEE International Conference on Decision and Control*, Dec. 1999

[15] W. T. Miller, R. S. Sutton, P. J. Werbos (eds.), "Neural Networks for Control," Boston, MA: MIT press, 1990

[16] A. Packard, "Gain Scheduling via Linear Fractional Transformations," *Systems and Control Letters* Vol. 22, 79–92, 1994

[17] W. Rugh, J. S. Shamma, "Research on Gain Scheduling," *Automatica* Vol. 36, 1401–1425, 2000

[18] C. W. Scherer, "LPV Control and Full Block Multipliers," to appear in: *Automatica*

[19] C. W. Scherer, "Robust Mixed Control and LPV Control and Full Block Scalings," to appear in: L. El Ghaoui, S. Niculescu, *Recent Advances in LMI Methods in Control*, SIAM, 1999

[20] Y. Su, A. M. Annaswamy, "Stable Neural Controllers for Nonlinear Dynamic Systems," in *Automatica* Vol. 34, 5:641–650, 1998

[21] J. A. K. Suykens, J. Vandewalle, B. De Moor, "Nonlinear System Identification using Neural State Space Models, Applicable in Robust Control Design," *International Journal of Control* 1:129–152, 1995

[22] J. A. K. Suykens, J. Vandewalle, B. De Moor, "Global Asymptotic Stability Criteria for Multilayer Recurrent Neural Networks with Applications to Modelling and Control," in *Proc. of the IEEE International Conference on Neural Networks*, Nov.-Dec. 1995

[23] J. A. K. Suykens, J. Vandewalle, B. De Moor, "Lur'e Systems with Multilayer Perceptron and Recurrent Neural Networks: Absolute Stability and Dissipativity," *IEEE Transactions on Automatic Control* 4:770–774, 1999