LPV H₂-Controller Synthesis using Nonlinear Programming

Daniel Petersson and Johan Löfberg

Linköping University Post Print

N.B.: When citing this work, cite the original article.

Original Publication:

Daniel Petersson and Johan Löfberg, LPV H₂-Controller Synthesis using Nonlinear Programming, 2011, Proceedings of the 18th IFAC World Congress, 6692-6696.

<http://dx.doi.org/10.3182/20110828-6-IT-1002.02028>

Copyright: IFAC

Postprint available at: Linköping University Electronic Press

<http://urn.kb.se/resolve?urn=urn:nbn:se:liu:diva-88935>

LPV $\mathcal{H}_2$ -Controller Synthesis Using Nonlinear Programming

Daniel Petersson * Johan Löfberg *

* *Division of Automatic Control, Department of Electrical Engineering,
Linköpings universitet, SE-581 83 Sweden; (e-mail:
{petersson,johanl}@isy.liu.se).*

Abstract: Controller synthesis for linear parameter varying (LPV) systems has received a lot of attention from the control community. This is mainly motivated by the wide range of non-linear dynamical systems that can be approximated by LPV-systems. In this paper a novel method is presented that, by only using local state space models as data, tries to solve the problem of finding a linear parameter varying output-feedback controller. The method uses non-linear programming and a quasi-Newton framework to solve the problem. The great advantages with the proposed method is that it is possible to impose structure in the controller and that you do not need an LPV-model, only state space models for different values of the scheduling parameters. Finally an example is presented to show the potential of the method.

Keywords: Linear Parameter Varying Systems; Controller Synthesis; Nonlinear Programming.

1. INTRODUCTION

Controller synthesis for linear parameter varying (LPV) systems has received a lot of attention from the control community. This is mainly motivated by the wide range of non-linear dynamical systems that can be approximated by LPV-systems and there exist many methods for doing so, see *e.g.*, Tóth [2008] or Petersson and Löfberg [2009].

The behavior of an LPV-system can be described by

$$\dot{x}(t) = \mathbf{A}(p(t))x(t) + \mathbf{B}(p(t))u(t), \quad (1a)$$

$$y(t) = \mathbf{C}(p(t))x(t) + \mathbf{D}(p(t))u(t) \quad (1b)$$

where $x \in \mathbb{R}^{n_x}$ are the states, $u \in \mathbb{R}^{n_u}$ and $y \in \mathbb{R}^{n_y}$ are the input and output signals and p is the vector of scheduling parameters. In flight control applications, the components of p are often model parameters, for instance mass, position of center of gravity, various aerodynamic coefficients, but can also include state dependent parameters such as altitude and velocity specifying current flight conditions. The methods for generating an LPV-model can be divided into two families; local methods and global methods, see Tóth [2008] for more information about this. The local methods require a number of local state space models, G_i , corresponding to models when applying a constant scheduling trajectory, $\dot{p} = 0$, to the LPV-model,

$$G_i : \begin{cases} \dot{x}(t) = \mathbf{A}_i x(t) + \mathbf{B}_i u(t) \\ y(t) = \mathbf{C}_i x(t) + \mathbf{D}_i u(t) \end{cases}$$

which will be denoted as

$$G_i : \left[\begin{array}{c|c} \mathbf{A}_i & \mathbf{B}_i \\ \hline \mathbf{C}_i & \mathbf{D}_i \end{array} \right].$$

From these local models an LPV-model is generated, that later can be used for LPV-controller synthesis. For different gain-scheduling methods, see Leith and Leithead [2000] or Rugh and Shamma [2000]. Important to note, is that a drawback with this class of methods is that it does not take time variations of the scheduling parameters into

account, thus limiting local methods to systems where the scheduling parameters vary slowly in time, which is a commonly used assumption when gain scheduling is used, Shamma and Athans [1992].

The method presented in this paper skips the intermediate step of generating an LPV-model and tries to generate an LPV-controller, $G_K(p)$, directly from local state space models.

A linear parameter varying output-feedback controller, $G_K(p)$, of order n_K can be described as an LPV-system

$$\dot{x}_K(t) = \mathbf{K}_A(p)x_K(t) + \mathbf{K}_B(p)y(t) \quad (2a)$$

$$u(t) = \mathbf{K}_C(p)x_K(t) + \mathbf{K}_D(p)y(t) \quad (2b)$$

where $x_K \in \mathbb{R}^{n_K}$ is the state vector of the controller, $y \in \mathbb{R}^{n_y}$ the measurement signal from the plant and $u \in \mathbb{R}^{n_u}$ the control signal to the plant.

In this paper we first formulate an optimization problem to find an LPV-controller that uses local state space models as data and then present an approach to solve this optimization problem.

2. OUTPUT-FEEDBACK CONTROLLER

Before looking further into the case of LPV-controller synthesis, we first look at a single linear plant and a single controller as these equations will be used later on.

A commonly used model for analyzing the performance of a system is

$$\begin{pmatrix} \dot{x} \\ z \\ y \end{pmatrix} = \begin{pmatrix} \mathbf{A} & \mathbf{B}_1 & \mathbf{B}_2 \\ \mathbf{C}_1 & \mathbf{D}_{11} & \mathbf{D}_{12} \\ \mathbf{C}_2 & \mathbf{D}_{21} & \mathbf{D}_{22} \end{pmatrix} \begin{pmatrix} x \\ w \\ u \end{pmatrix} \quad (3)$$

where $x \in \mathbb{R}^{n_x}$ is the state vector, $w \in \mathbb{R}^{n_w}$ the disturbance signal, $u \in \mathbb{R}^{n_u}$ the control signal, $z \in \mathbb{R}^{n_z}$ the performance measure and $y \in \mathbb{R}^{n_y}$ the measurement

signal. Here, the matrix $\mathbf{D}_{22}$ can be assumed, without loss of generality, to be zero, see Zhou et al. [1996]. The controller can be described as

$$\dot{x}_K(t) = \mathbf{K}_A x_K(t) + \mathbf{K}_B u(t) \quad (4a)$$

$$u(t) = \mathbf{K}_C x_K(t) + \mathbf{K}_D y(t). \quad (4b)$$

Combine equations (3) and (4) to obtain a state space representation of the closed loop system from w to z ,

$$T_{w,z} : \left[\begin{array}{c|c} \left(\begin{array}{cc} \mathbf{A} + \mathbf{B}_2 \mathbf{K}_D \mathbf{C}_2, & \mathbf{B}_2 \mathbf{K}_C \\ \mathbf{K}_B \mathbf{C}_2, & \mathbf{K}_A \end{array} \right) & \left(\begin{array}{c} \mathbf{B}_1 + \mathbf{B}_2 \mathbf{K}_D \mathbf{D}_{21} \\ \mathbf{K}_B \mathbf{D}_{21} \end{array} \right) \\ \hline (\mathbf{C}_1 + \mathbf{D}_{12} \mathbf{K}_D \mathbf{C}_2, & \mathbf{D}_{12} \mathbf{K}_C) & (\mathbf{D}_{11} + \mathbf{D}_{12} \mathbf{K}_D \mathbf{D}_{21}) \end{array} \right] \quad (5)$$

The problem of synthesizing a controller can be divided into three cases. The simple case, both in the case of $\mathcal{H}_\infty$ -controllers and $\mathcal{H}_2$ -controllers, is to find a full order controller, $n_K = n_x$, see *e.g.*, Doyle et al. [1989] or Skogestad and Postlethwaite [2007]. The two more difficult cases are to find a *reduced order output-feedback controller*, $0 < n_K < n_x$, or a *static output-feedback controller*, $n_K = 0$. However, the problem of computing a reduced order output-feedback controller can be reformulated as a static output-feedback controller problem, this is shown in El Ghaoui et al. [1997] and restated here for clarification.

To see that the problem of finding a reduced order output-feedback controller can be reformulated as a problem of finding a static output-feedback controller we first create the augmented system, G_{aug} .

$$G_{aug} : \left[\begin{array}{c|c} \mathbf{A}_{aug} & \left(\begin{array}{cc} \mathbf{B}_{1,aug} & \mathbf{B}_{2,aug} \end{array} \right) \\ \hline \left(\begin{array}{c} \mathbf{C}_{1,aug} \\ \mathbf{C}_{2,aug} \end{array} \right) & \left(\begin{array}{cc} \mathbf{D}_{11,aug} & \mathbf{D}_{12,aug} \\ \mathbf{D}_{21,aug} & \mathbf{D}_{22,aug} \end{array} \right) \end{array} \right], \text{ where}$$

$$\mathbf{A}_{aug} = \begin{pmatrix} \mathbf{A} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{pmatrix}, \mathbf{B}_{1,aug} = \begin{pmatrix} \mathbf{B}_1 \\ \mathbf{0} \end{pmatrix}, \mathbf{B}_{2,aug} = \begin{pmatrix} \mathbf{0} & \mathbf{B}_2 \\ \mathbf{I} & \mathbf{0} \end{pmatrix},$$

$$\mathbf{C}_{1,aug} = (\mathbf{C}_1 \ \mathbf{0}), \mathbf{D}_{11,aug} = \mathbf{D}_{11}, \mathbf{D}_{12,aug} = (\mathbf{0} \ \mathbf{D}_{12}),$$

$$\mathbf{C}_{2,aug} = \begin{pmatrix} \mathbf{0} & \mathbf{I} \\ \mathbf{C}_2 & \mathbf{0} \end{pmatrix}, \mathbf{D}_{21,aug} = \begin{pmatrix} \mathbf{0} \\ \mathbf{D}_{21} \end{pmatrix}, \mathbf{D}_{22,aug} = \mathbf{0},$$

with the new state space vector augmented with $x_K \in \mathbb{R}^{n_K}$, $x_{aug} = \begin{pmatrix} x \\ x_K \end{pmatrix}$, the new control signal augmented

with $u_K \in \mathbb{R}^{n_K}$, $u_{aug} = \begin{pmatrix} u_K \\ u \end{pmatrix}$ and the new measurement

signal augmented with $y_K \in \mathbb{R}^{n_K}$, $y_{aug} = \begin{pmatrix} y_K \\ y \end{pmatrix}$. The $\mathbf{0}$'s are matrices of compatible sizes with all elements zero and $\mathbf{I}$ are identity matrices of compatible sizes.

Now use the static controller, $\mathbf{K}_{aug}$, *i.e.*, $u_{aug} = \mathbf{K}_{aug} y_{aug}$, which has the structure

$$\mathbf{K}_{aug} = \begin{pmatrix} \mathbf{K}_A & \mathbf{K}_B \\ \mathbf{K}_C & \mathbf{K}_D \end{pmatrix},$$

where $\mathbf{K}_A, \mathbf{K}_B, \mathbf{K}_C$ and $\mathbf{K}_D$ are the matrices from the controller in (4). Computing the closed loop equations for this feedback system we will arrive at the same equations as we had in (5). This shows that if we have a method for calculating a static output-feedback controller, then we also have a method for calculating a reduced order controller.

Looking at the reformulation of the problem of finding a reduced order output-feedback controller to the problem of finding a static output-feedback controller, one can

easily realize that this also generalizes to LPV-controllers and LPV-systems, of the form (1) and (2). The method developed in this paper will only consider static output-feedback LPV-controllers, but we stress that this also includes reduced order LPV-controllers.

3. LPV-CONTROLLER SYNTHESIS USING THE $\mathcal{H}_2$ -NORM

The goal with the optimization problem that we want to state in this section is to synthesize a static output-feedback linear parameter varying $\mathcal{H}_2$ -controller. As said before we assume that we are given a number of linear models, G_i , for evaluating the performance for models corresponding to applying a constant scheduling trajectory, $\dot{p} = 0$, to the LPV-model,

$$G_i : \left[\begin{array}{c|c} \mathbf{A}_i & \left(\begin{array}{cc} \mathbf{B}_{1,i} & \mathbf{B}_{2,i} \end{array} \right) \\ \hline \left(\begin{array}{c} \mathbf{C}_{1,i} \\ \mathbf{C}_{2,i} \end{array} \right) & \left(\begin{array}{cc} \mathbf{D}_{11,i} & \mathbf{D}_{12,i} \\ \mathbf{D}_{21,i} & \mathbf{0} \end{array} \right) \end{array} \right].$$

We define the static output-feedback LPV-controller as

$$u = \mathbf{K}(p)y, \quad (6)$$

where $\mathbf{K}(p) \in \mathbb{R}^{n_u \times n_y}$ is a matrix that is a linear combination of some basis functions $w_k(p)$, *i.e.*,

$$\mathbf{K}(p) = \sum_k w_k(p) \mathbf{K}^{(k)}. \quad (7)$$

Our optimization variables will be the coefficient matrices $\mathbf{K}^{(k)}$.

By looking at equation (5) we see that the closed loop systems that we will obtain, using the static output-feedback LPV-controller, (6), on any of the given systems, will be

$$\begin{aligned} T_{w,z,i} : \left[\begin{array}{c|c} \mathbf{A}_{T,i} & \mathbf{B}_{T,i} \\ \hline \mathbf{C}_{T,i} & \mathbf{D}_{T,i} \end{array} \right] &= \\ &= \left[\begin{array}{c|c} \mathbf{A}_i + \mathbf{B}_{2,i} \mathbf{K}(p_i) \mathbf{C}_{2,i} & \mathbf{B}_{1,i} + \mathbf{B}_{2,i} \mathbf{K}(p_i) \mathbf{D}_{21,i} \\ \hline \mathbf{C}_{1,i} + \mathbf{D}_{12,i} \mathbf{K}(p_i) \mathbf{C}_{2,i} & \mathbf{D}_{11,i} + \mathbf{D}_{12,i} \mathbf{K}(p_i) \mathbf{D}_{21,i} \end{array} \right]. \end{aligned} \quad (8)$$

We are now ready to state the optimization problem for finding a static output-feedback LPV-controller, $\mathbf{K}(p)$,

$$\min_{\mathbf{K}^{(k)}} \sum_i \|T_{w,z,i}\|_{\mathcal{H}_2}^2. \quad (9)$$

An important thing to note with this formulation is that for the $\mathcal{H}_2$ -norm to be defined, the systems $T_{w,z,i}$ have to be stable and strictly proper, *i.e.*, $\mathbf{A}_i + \mathbf{B}_{2,i} \mathbf{K}(p_i) \mathbf{C}_{2,i}$ need to be Hurwitz and $\mathbf{D}_{11,i} + \mathbf{D}_{12,i} \mathbf{K}(p_i) \mathbf{D}_{21,i} = \mathbf{0}$. Note that already the problem of finding a $\mathbf{K}(p_i)$ that stabilizes the system is most likely an NP-hard problem, see Blondel and Tsitsiklis [1997]. Because of this, for the rest of the paper, if nothing else is mentioned, it will be assumed that we start with a $\mathbf{K}(p)$ that stabilizes the given linear state space models and that the term $\mathbf{D}_{11,i} + \mathbf{D}_{12,i} \mathbf{K}(p_i) \mathbf{D}_{21,i} = \mathbf{0}$.

In line with Petersson and Lofberg [2009] we want to solve the optimization problem (9) using a non-linear programming (NLP) approach and a quasi-Newton framework. To do this, we first of all need a way to compute the cost function efficiently. We also need an expression for the gradient of the cost function, that also can be computed efficiently.

Remark 1. Note that with the formulation of the optimization problem as in (9), no assumption that all local models have to be expressed in the same state space basis is used. This is because the cost function considers the input-output relations of the systems, which is independent of the state basis used in the models. Note also that for this method using only a finite number of models, we can not guarantee stability for all local models that can be extracted from the LPV-model. What can be guaranteed is that, because the $\mathcal{H}_2$ -norm is only defined for stable systems and given that we start with a feasible point, *i.e.* a stabilizing LPV-controller, the resulting LPV-controller will be stable in all points, p_i , from where the local models used in the optimization are extracted.

3.1 Cost Function

Computing the squared $\mathcal{H}_2$ -norm of a system, as we want to do in (9), can be rewritten as, see *e.g.*, Zhou et al. [1996],

$$\|T_{w,z,i}\|_{\mathcal{H}_2}^2 = \text{tr } \mathbf{B}_{T,i}^T \mathbf{Q}_i \mathbf{B}_{T,i} = \text{tr } \mathbf{C}_{T,i} \mathbf{P}_i \mathbf{C}_{T,i}^T,$$

where $\mathbf{Q}_i$ and $\mathbf{P}_i$ are the controllability and observability Gramians of the system and satisfy the Lyapunov equations

$$\mathbf{A}_{T,i} \mathbf{P}_i + \mathbf{P}_i \mathbf{A}_{T,i}^T + \mathbf{B}_{T,i} \mathbf{B}_{T,i}^T = \mathbf{0}, \quad (10a)$$

$$\mathbf{A}_{T,i}^T \mathbf{Q}_i + \mathbf{Q}_i \mathbf{A}_{T,i} + \mathbf{C}_{T,i}^T \mathbf{C}_{T,i} = \mathbf{0}. \quad (10b)$$

This yields the cost function

$$\begin{aligned} \sum_i \|T_{w,z,i}\|_{\mathcal{H}_2}^2 &= \sum_i \text{tr } \mathbf{B}_{1,i}^T \mathbf{Q}_i \mathbf{B}_{1,i} + \\ &+ \text{tr } \mathbf{D}_{21,i}^T \mathbf{K}(p_i)^T \mathbf{B}_{2,i}^T \mathbf{Q}_i \mathbf{B}_{2,i} \mathbf{K}(p_i) \mathbf{D}_{21,i} + \\ &+ 2 \text{tr } \mathbf{B}_{1,i}^T \mathbf{Q}_i \mathbf{B}_{2,i} \mathbf{K}(p_i) \mathbf{D}_{21,i}, \end{aligned} \quad (11a)$$

$$\begin{aligned} \sum_i \|T_{w,z,i}\|_{\mathcal{H}_2}^2 &= \sum_i \text{tr } \mathbf{C}_{1,i} \mathbf{P}_i \mathbf{C}_{1,i}^T + \\ &+ \text{tr } \mathbf{D}_{12,i} \mathbf{K}(p_i) \mathbf{C}_{2,i} \mathbf{P}_i \mathbf{C}_{2,i}^T \mathbf{K}(p_i)^T \mathbf{D}_{12,i}^T + \\ &+ 2 \text{tr } \mathbf{C}_{1,i} \mathbf{P}_i \mathbf{C}_{2,i}^T \mathbf{K}(p_i)^T \mathbf{D}_{12,i}^T. \end{aligned} \quad (11b)$$

The two equations in (11) are equivalent and can both be used to calculate the cost function. Depending on which is chosen either $\mathbf{P}_i$ or $\mathbf{Q}_i$ needs to be computed for every i .

3.2 Gradient

To use a quasi-Newton framework to solve the optimization problem and to have a good indication of a local optimum, a gradient of the cost function is needed.

One of the appealing features of using the $\mathcal{H}_2$ -norm, instead of the $\mathcal{H}_\infty$ -norm, is that the cost function is differentiable with respect to $\mathbf{K}(p)$ and thus also $\mathbf{K}^{(k)}$, since $\mathbf{K}^{(k)}$ is linear in $\mathbf{K}(p)$. To show this we will start by deriving the expression for the gradient when only having one linear state space model, G , given and differentiating with respect to $\mathbf{K}$ and then generalize this to the case where we are given an arbitrary number of linear state space models and differentiate with respect to the coefficient matrices $\mathbf{K}^{(k)}$. For the moment we will drop the index i to make the derivation easier to follow. The given system will be denoted

$$G : \left[\begin{array}{c|c} \mathbf{A} & (\mathbf{B}_1 \ \mathbf{B}_2) \\ \hline \begin{pmatrix} \mathbf{C}_1 \\ \mathbf{C}_2 \end{pmatrix} & \begin{pmatrix} \mathbf{D}_{11} & \mathbf{D}_{12} \\ \mathbf{D}_{21} & \mathbf{0} \end{pmatrix} \end{array} \right]$$

and the controller

$$u = \mathbf{K}y.$$

We will denote the elements in $\mathbf{K}$ by k_{ij} and $\frac{\partial \|T_{w,z}\|_{\mathcal{H}_2}^2}{\partial k_{ij}} = \left[\frac{\partial \|T_{w,z}\|_{\mathcal{H}_2}^2}{\partial \mathbf{K}} \right]_{ij}$. To derive the gradient of the cost function (11) with respect to $\mathbf{K}$, start by differentiating equation (11b). Straightforward elementwise differentiation of (11b) yields

$$\left[\frac{\partial \|T_{w,z}\|_{\mathcal{H}_2}^2}{\partial \mathbf{K}} \right]_{ij} = \text{tr } \frac{\partial \mathbf{P}}{\partial k_{ij}} \mathbf{C}^* + 2 \text{tr } \frac{\partial \mathbf{K}^T}{\partial k_{ij}} \mathbf{D}_{12}^T \mathbf{C}_T \mathbf{P} \mathbf{C}_2^T, \quad (12)$$

where

$$\mathbf{C}^* = \mathbf{C}_1^T \mathbf{C}_1 + \mathbf{C}_2^T \mathbf{K}^T \mathbf{D}_{12}^T \mathbf{D}_{12} \mathbf{K} \mathbf{C}_2 + 2 \mathbf{C}_2^T \mathbf{K}^T \mathbf{D}_{12}^T \mathbf{C}_1.$$

To get rid of the factor $\frac{\partial \mathbf{P}}{\partial k_{ij}}$ we will need the following lemma (see Yan and Lam [1999]).

Lemma 1. If $\mathbf{M}$ and $\mathbf{N}$ satisfy the following Sylvester equations

$$\mathbf{A} \mathbf{M} + \mathbf{M} \mathbf{B} + \mathbf{C} = \mathbf{0}, \quad \mathbf{N} \mathbf{A} + \mathbf{B} \mathbf{N} + \mathbf{D} = \mathbf{0},$$

then $\text{tr } \mathbf{C} \mathbf{N} = \text{tr } \mathbf{D} \mathbf{M}$.

We will also need equation (10a) differentiated with respect to k_{ij} ,

$$\mathbf{A}_T \frac{\partial \mathbf{P}}{\partial k_{ij}} + \frac{\partial \mathbf{P}}{\partial k_{ij}} \mathbf{A}_T^T + \mathbf{B}^* = \mathbf{0}, \quad (13)$$

where

$$\begin{aligned} \mathbf{B}^* &= \mathbf{B}_2 \frac{\partial \mathbf{K}}{\partial k_{ij}} \mathbf{C}_2 \mathbf{P} + \mathbf{P} \mathbf{C}_2^T \frac{\partial \mathbf{K}^T}{\partial k_{ij}} \mathbf{B}_2^T + \mathbf{B}_1 \mathbf{D}_{21} \frac{\partial \mathbf{K}^T}{\partial k_{ij}} \mathbf{B}_2^T + \\ &+ \mathbf{B}_2 \frac{\partial \mathbf{K}}{\partial k_{ij}} \mathbf{D}_{21} \mathbf{B}_1^T + \mathbf{B}_2 \frac{\partial \mathbf{K}}{\partial k_{ij}} \mathbf{D}_{21} \mathbf{D}_{21}^T \mathbf{K}^T \mathbf{B}_2^T + \\ &+ \mathbf{B}_2 \mathbf{K} \mathbf{D}_{21} \mathbf{D}_{21}^T \frac{\partial \mathbf{K}^T}{\partial k_{ij}} \mathbf{B}_2^T. \end{aligned}$$

Now using equations (10b) and (13) in Lemma 1 we get, after some simplifications,

$$\text{tr } \frac{\partial \mathbf{P}}{\partial k_{ij}} \mathbf{C}^* = 2 \text{tr } \frac{\partial \mathbf{K}^T}{\partial k_{ij}} \mathbf{B}_2^T \mathbf{Q} [\mathbf{P} \mathbf{C}_2^T + \mathbf{B}_T] \mathbf{D}_{21}^T.$$

This equation together with (12) entails that

$$\begin{aligned} \left[\frac{\partial \|T_{w,z}\|_{\mathcal{H}_2}^2}{\partial \mathbf{K}} \right]_{ij} &= 2 \text{tr } \frac{\partial \mathbf{K}^T}{\partial k_{ij}} [\mathbf{B}_2^T \mathbf{Q} \mathbf{B}_T \mathbf{D}_{21}^T + \\ &+ \mathbf{D}_{12}^T \mathbf{C}_T \mathbf{P} \mathbf{C}_2^T + \mathbf{B}_2^T \mathbf{Q} \mathbf{P} \mathbf{C}_2^T], \end{aligned}$$

which can be written as, reintroducing the index i again,

$$\begin{aligned} \frac{\partial \|T_{w,z,i}\|_{\mathcal{H}_2}^2}{\partial \mathbf{K}} &= 2 (\mathbf{B}_{2,i}^T \mathbf{Q}_i \mathbf{B}_{T,i} \mathbf{D}_{21,i}^T + \\ &+ \mathbf{D}_{12,i}^T \mathbf{C}_{T,i} \mathbf{P}_i \mathbf{C}_{2,i}^T + \mathbf{B}_{2,i}^T \mathbf{Q}_i \mathbf{P}_i \mathbf{C}_{2,i}^T). \end{aligned} \quad (14)$$

Now looking at the more general case, when given multiple linear state space models and assuming $\mathbf{K}(p)$ depends on p as in (7), we want to have an expression for the gradient of (11) with respect to the coefficient matrices $\mathbf{K}^{(k)}$. Since $\mathbf{K}^{(k)}$ is linear in $\mathbf{K}(p)$ this is easy to derive.

$$\frac{\partial \sum_i \|T_{w,z,i}\|_{\mathcal{H}_2}^2}{\partial \mathbf{K}^{(k)}} = 2 \sum_i w_k(p_i) (\mathbf{B}_{2,i}^T \mathbf{Q}_i \mathbf{B}_{T,i} \mathbf{D}_{21,i}^T + \mathbf{D}_{12,i}^T \mathbf{C}_{T,i} \mathbf{P}_i \mathbf{C}_{2,i}^T + \mathbf{B}_{2,i}^T \mathbf{Q}_i \mathbf{P}_i \mathbf{C}_{2,i}^T). \quad (15)$$

The closed form expression obtained when differentiating the cost function (11) with respect to the coefficient matrices $\mathbf{K}^{(k)}$ in the feedback matrix $\mathbf{K}(p)$, is expressed in the given system matrices, the feedback matrix and the matrices $\mathbf{P}_i$ and $\mathbf{Q}_i$. Equation (15) can now be used as a measure for first order optimality and in a quasi-Newton framework to create a descent direction.

Remark 2. As can be seen in the derivation of the gradient, it is never assumed that all of the parameters in the coefficient matrices, $\mathbf{K}^{(k)}$, should be free. The elementwise differentiation derivation shows that one can choose any element to be either free or constant. With this fact it is possible to impose sparsity structure in the elements of the controller.

4. COMPUTATIONAL ASPECTS

As we mentioned before, we want to solve the optimization problem (9) using an NLP approach and a quasi-Newton framework. To do this we need to be able to compute both the cost function and its gradient efficiently.

We saw in Section 3.1 that, for every i , we need to compute either the controllability- or observability Gramian, *i.e.*, we need to solve a Lyapunov equation of size $n_x + n_K$. In Section 3.2 we derived a closed form expression for the gradient, which is used in the quasi-Newton framework to compute a descent direction and as a measure of first order optimality. To compute this gradient, (15), we need to compute both the observability Gramian (10b) and the controllability Gramian (10a). But one of these we have already computed to obtain the cost function, so we only need the other one to compute the gradient. However, looking at the structure of the equations in (10), we see that they have the same factors $\mathbf{A}_T$ and $\mathbf{A}_T^T$. This can be utilized to solve these two equations very efficiently simultaneously, see Benner et al. [1998], and to speed up the computations of the cost function and its gradient. More discussion can be found in Petersson [2010].

5. EXAMPLES

In this section we apply the proposed method on an illustrative example to show that it works.

When solving the example, the function `fminunc` in MATLAB was used as the quasi-Newton solver framework. The problem of generating an initial point is an extremely important problem in need of significant amounts of research. Here, as an initial point for the solver, a controller with all zeros is used. This is possible when the systems are open loop stable.

Example 1. Here a small academic example is presented to show the potential of the new method.

The system in this example is

$$G = G_1 G_2, \quad \text{where} \quad (16a)$$

$$G_1 = \frac{1}{s^2 + 2\zeta_1 s + 1}, \quad G_2 = \frac{9}{s^2 + 6\zeta_2 s + 9}, \quad (16b)$$

$$\zeta_1 = 0.1 + 0.9p, \quad \zeta_2 = 0.1 + 0.9(1 - p), \quad p \in [0, 1]. \quad (16c)$$

From these equations we obtain $\mathbf{A}(p)$, $\mathbf{B}_2(p)$, $\mathbf{C}_2(p)$ and $\mathbf{D}_{22}(p)$, using the notation in (3), that represents the dynamical system. Then we create the matrices

$$\mathbf{B}_1(p) = \mathbf{I}_{4 \times 4}, \quad \mathbf{C}_1(p) = \mathbf{I}_{4 \times 4}$$

$$\mathbf{D}_{11}(p) = \mathbf{0}_{4 \times 4}, \quad \mathbf{D}_{12}(p) = \begin{pmatrix} \mathbf{0}_{3 \times 1} \\ 1 \end{pmatrix}, \quad \mathbf{D}_{21}(p) = \mathbf{0}_{1 \times 4}$$

to have a fully defined performance measure of the system. From this system we extract five systems representing five equidistant points in $p \in [0, 1]$, *i.e.*, we are given five linear models, extracted from the LPV-system (16), with four states each.

The data is given in a state basis where the elements in some of the system matrices happen to depend nonlinearly on the parameter p (it is the state basis from a balanced realization of the system), see Figure 1. Hence, judging from the given data, one could easily suspect that a complex LPV-controller would be required. However, in this example using the proposed method, we will try to find a static output-feedback LPV-controller that is only linear in the parameter p , *i.e.*,

$$u(t) = \mathbf{K}(p)y(t), \quad \mathbf{K}(p) = \mathbf{K}^{(0)} + \mathbf{K}^{(1)}p. \quad (17)$$

Fig. 1. The elements in the $\mathbf{A}$ -matrix as function of p .

After performing the optimization the resulting controller has the values $\mathbf{K}^{(0)} = 0.259$ and $\mathbf{K}^{(1)} = -0.305$ and took 0.16 s to compute.

To validate the method 100 validation points were generated from (16), for $p \in [0, 1]$. For each of these 100 models, an optimal static output-feedback controller was created (the associated optimization problem is a scalar problem, hence trivially solved using, *e.g.*, gridding). In Figure 2 the ratio between the $\mathcal{H}_2$ -performance with the LPV-controller, (17), and the $\mathcal{H}_2$ -performance with the optimal static output-feedback controller in the different validation points is shown. In Figure 2 we see that the LPV-controller is close to the optimal reference controller in the validation points. In Figure 3 the reference controller and

Fig. 2. The ratio between the $\mathcal{H}_2$ -performance with the LPV-controller and the $\mathcal{H}_2$ -performance with the optimal static output-feedback controller in the different validation points.

the resulting LPV-controller (linear in p) are plotted. Additionally, in Figure 3 two more LPV-controllers, computed using the same method, are plotted, these with quadratic and cubic dependence in p .

Fig. 3. The reference controller (solid line) and the resulting LPV-controllers (linear in p , dashed line, quadratic in p , dash-dotted line and cubic in p , dotted line) plotted as functions of the parameter, p .

6. CONCLUSIONS

In this paper we have formulated an optimization problem that tries to synthesize an LPV-controller given a number of state space models for different values of the scheduling parameters, corresponding to models when applying a constant scheduling trajectory. To solve this problem an NLP approach was suggested and expressions for the cost function and its gradient, that can be computed efficiently,

was derived and presented. As the method presented in this paper is based on local models, without *a priori* information about the rate of change of the parameters, it does not take time variations of the scheduling parameters into account, thus limiting it to systems where the scheduling parameters vary slowly in time. The great advantages with the method presented in this paper are that it is possible to impose sparsity structure on the controller and that you only need local state space models as data to create the controller, in fact the local state space models does not even have to be expressed in the same basis.

REFERENCES

- Peter Benner, Jose M. Claver, and Enrique S. Quintana-Orti. Efficient solution of coupled Lyapunov equations via matrix sign function iteration. In *Proceedings of the 3rd Portuguese Conference on Automatic Control*, pages 205–210, 1998.
- Vincent Blondel and John N. Tsitsiklis. NP-hardness of some linear control design problems. *SIAM Journal on Control and Optimization*, 35(6):2118 – 2127, 1997.
- John C. Doyle, Keith Glover, Pramod P. Khargonekar, and Bruce A. Francis. State-Space Solutions to Standard $\mathcal{H}_2$ and $\mathcal{H}_\infty$ Control Problems. *IEEE Transactions on Automatic Control*, 34(8):831 – 847, 1989.
- Laurent El Ghaoui, Francois Oustry, and Mustapha AitRami. A cone complementarity linearization algorithm for static output-feedback and related problems. *IEEE Transactions on Automatic Control*, 42(8):1171 – 1176, 1997.
- Douglas J. Leith and William E. Leithead. Survey of gain-scheduling analysis and design. *International Journal of Control*, 73(11):1001 – 1025, 2000.
- Daniel Petersson. *Nonlinear optimization approaches to $\mathcal{H}_2$ -norm based LPV modelling and control*. Licentiate thesis no. 1453, Department of Electrical Engineering, Linköping University, 2010.
- Daniel Petersson and Johan Löfberg. Optimization based LPV-approximation of multi-model systems. In *Proceedings of the European Control Conference*, pages 3172–3177, 2009.
- Wilson J. Rugh and Jeff S. Shamma. Research on gain scheduling. *Automatica*, 36(10):1401 – 1425, 2000.
- Jeff S. Shamma and Michael Athans. Gain scheduling: potential hazards and possible remedies. *Control Systems Magazine, IEEE*, 12(3):101–107, jun 1992.
- Sigurd Skogestad and Ian Postlethwaite. *Multivariable Feedback Control: Analysis and Design*. Wiley, second edition, 2007. ISBN 0-470-01167-6.
- Roland Tóth. *Modeling and Identification of Linear Parameter-Varying Systems, an Orthonormal Basis Function Approach*. PhD thesis, Delft University of Technology, 2008.
- Wei-Yong Yan and James Lam. An approximate approach to $\mathcal{H}_2$ optimal model reduction. *IEEE Transactions on Automatic Control*, 44(7):1341–1358, Jul 1999.
- Kemin Zhou, John C. Doyle, and Keith Glover. *Robust and optimal control*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1996. ISBN 0-13-456567-3.