Accelerating simulations of computationally intensive first principle models using accurate quasi linear parameter varying models

Robert Bos[1], Xavier Bombois[2] and Paul M.J. Van den Hof[2]

[1] Shell International Exploration and Production B.V., Rijswijk, The Netherlands

[2] Delft Center for Systems and Control, Delft University of Technology, The Netherlands

September 3, 2009

Abstract

First principles models are commonly obtained using finite element or finite difference methods. One of the advantages of these models is that the states in the model have a clear physical interpretation. This makes of them perfect candidates for the monitoring of the states of the system. Unfortunately, the CPU time associated with each evaluation of these complex models is often far too large for these models to be used for online monitoring purposes. This paper introduces a general method to approximate a computationally expensive first principles model with a quasi linear parameter varying (qLPV) model. Besides approximating the original model accurately and conserving the physical interpretation of the states, the resulting qLPV model has generally a much simpler structure than the original model. This in turn implies that the CPU time associated with each model evaluation is generally considerably reduced, allowing the use of these models for online monitoring. Unlike other qLPV identification techniques, the proposed method extensively uses the availability of the original first principles model.

1 Introduction

This paper presents a novel approach for reducing the complexity of large scale first principle models in order to accelerate the computations with these models. Accelerating such large scale models is indeed absolutely necessary to enable the use of these models for online state estimation (monitoring). The proposed approach consists of using data generated with the first principle model to identify a quasi Linear Parameter Varying (qLPV) model whose order is chosen in such a way that the simplified model is a sufficiently accurate approximation of the original first principle model.

Large scale first principles models, which can be used to describe the dynamical behaviour of complex industrial processes, are commonly obtained using finite element or finite

difference methods with a very fine spatial grid. Consequently, the obtained state-space models (1)-(2) are characterized by a very large state vector $\bar{x}(k)$ (typically $dim(\bar{x}(k)) \sim 10^3 - 10^9$) and complex nonlinear functions $\bar{f}(\cdot)$ and $\bar{h}(\cdot)$:

$$\bar{x}(k+1) = \bar{f}(\bar{x}(k), u(k)) \quad (1)$$

$$y(k) = \bar{h}(\bar{x}(k), u(k)). \quad (2)$$

In (1)-(2), the vector $u(k)$ represents the inputs of the system and the vector $y(k)$ the measured outputs. In such first principle models, the states $\bar{x}(k)$ have a clear physical interpretation. In fact, first principle modeling is the only technique[1] which allows to describe the relation between the inputs $u(k)$ and the physical properties that are not directly measured i.e. the state vector $\bar{x}(k)$. Consequently, in theory, such a model is the perfect tool for the online monitoring of these physical properties. However, we will see that, in practice, this is not the case. Indeed, the monitoring of the state vector is typically performed using nonlinear Kalman filtering. Nonlinear Kalman filtering estimates the state-vector $\bar{x}$ at time k from $u(k)$, $y(k)$ and the estimate of $\bar{x}$ at time $k-1$. For this purpose, the algorithm requires a number of evaluations of the model (1)-(2) which is proportional to (at least) $dim(\bar{x}(k))$. Knowing that, due to the complexity of the functions $\bar{f}(\cdot)$ and $\bar{h}(\cdot)$, the CPU time required by one model evaluation can be of the order of the sampling interval, these large scale first principles model can usually only be used for off-line simulation studies.

The problem of simplifying a large scale first principles model to enable online monitoring of the physical variables $\bar{x}(k)$ (and to enable control design) is thus of the highest importance. Based on the considerations above, the simplification of the model must involve both

1. the reduction of the number of states (in order to reduce the number of model evaluations required for the state estimation),

2. and the reduction of the complexity of the nonlinear functions involved in the nonlinear model (in order to reduce the time required for each of the remaining evaluations).

In this paper, we will focus on the second objective. Indeed, the problem of an overly large state dimension has already been extensively studied in the literature [8, 18, 13]. This problem can be solved by using projection based model reduction technique such as Proper Orthogonal Decomposition (POD)[8] or (Empirical) nonlinear Balancing [18][13]. POD techniques allow one to derive a matrix T with (much) more rows than columns and this matrix is used to project the original state vector $\bar{x}(k)$ into a reduced order state vector $x(k)$ (i.e. with $\dim(x(k)) << \dim(\bar{x}(k))$). This is done using the following relation $x(k) = T^{\dagger}\bar{x}(k)$ with $T^{\dagger}$ the pseudo-inverse of the mapping T. The matrix T is determined in such a way that the physical state $\bar{x}(k)$ can be accurately reconstructed from the reduced order states $x(k)$ using the mapping $Tx(k)$ i.e. $\bar{x}(k) \approx Tx(k)$. By doing this, the original first principle model (1)-(2) can be replaced by the following reduced order model:

$$x(k+1) = f(x(k), u(k)) \stackrel{\Delta}{=} T^{\dagger}\bar{f}(Tx(k), u(k)) \quad (3)$$

$$y(k) = h(x(k), u(k)) \stackrel{\Delta}{=} \bar{h}(Tx(k), u(k)) \quad (4)$$

$$\bar{x}(k) = Tx(k) \quad (5)$$

[1]Data-based modeling is only able to describe the input-output behaviour.

With (Empirical) nonlinear Balancing [18][13], the reduction of the state dimension can be obtained in a slightly different, but similar way.

The model (3)-(4) can be used in nonlinear Kalman filtering to deliver an estimate $\hat{x}(k)$ of the reduced order state $x(k)$ and (5) can then be used to construct the estimate $\hat{\bar{x}}(k)$ of the original state vector $\bar{x}(k)$ i.e. $\hat{\bar{x}}(k) = T\hat{x}(k)$. Since $\dim(x(k)) << \dim(\bar{x}(k))$, the number of evaluations of the model (3)-(4) required to compute $\hat{x}(k)$ (and thus $\hat{\bar{x}}(k)$) is strongly reduced. However, the above procedure does not necessarily reduce the computational effort[2] per model evaluation. Indeed, the model (3)-(4) still involves the complicated nonlinear functions $\bar{f}(.)$ and $\bar{h}(.)$. In fact, only when the original model (1)-(2) is linear, a state dimension reduction as presented above will result in a model that can be evaluated significantly faster than the original full order model.

As a consequence, online monitoring is generally still impossible with the model (3)-(4) and the second simplification objective has also to be performed. This second simplification objective is to reduce the complexity of the nonlinear functions $f(.)$ and $h(.)$ in (3)-(4) (or equivalently to reduce the complexity of the nonlinear functions $\bar{f}(.)$ and $\bar{h}(.)$ in (1)-(2)) and as a consequence to reduce the computation time per model evaluation.

In the literature, different approaches are available in order to reduce the complexity of the nonlinear functions $\bar{f}(.)$ (resp. $f(.)$) and $\bar{h}(.)$ (resp. $h(.)$) for this purpose.

The first approach consists of simplifying the physical relations that were used to generate the model (1)-(2). Physical relations can be simplified by ignoring higher order terms, or neglecting certain effects (see [15] and references therein). While good results can be obtained in this manner [6], such an approach is highly problem specific and requires process specialists to perform the model simplification.

The second approach is a partitioning method [5, 1]. Partitioning methods split the original state $\bar{x}(k)$ into two parts: $\bar{x}^{[1]}(k)$ and $\bar{x}^{[2]}(k)$. The model (1) is only used to compute $\bar{x}^{[1]}$, the remaining states are reconstructed using linear methods. A drawback of this approach is that it is generally difficult to find an appropriate partitioning allowing the accurate reconstruction of $\bar{x}^{[2]}(k)$.

As opposed to the previous approaches, the third approach is very simple to implement. In this approach, the initial model (1)-(2) is first linearized around a chosen working point before the state dimension is reduced [15]. Indeed, as already mentioned, for linear models, the reduction of the state dimension results in a considerable reduction of the computation time. A drawback of this method is that linearization only results in reasonably accurate models if the original system was already close to a linear system.

The fourth approach encountered in the literature simplifies the full order state equation (1) using a data-based approach [10]. More precisely, simulation data $\{\bar{x}(k),\ u(k)\}$

[2]When the model is available in implicit form, some increase in simulation speed can sometimes be obtained. This decrease in CPU time per model evaluation is mostly only minor [1, 19].

are generated with (1). The simulation data $\bar{x}(k)$ are projected into the reduced order $x(k) = T^{\dagger}\bar{x}(k)$ using the matrix $T^{\dagger}$ derived using POD techniques with the simulation data (or other data). Subsequently, a subspace estimator is used to identify a linear (and thus much faster) model of the relation between $x(k)$ and $u(k)$. Like in the third approach, the main drawback of this method is that, if the original model shows significant nonlinearities, the accuracy of the linear model will be limited.

There should always be a trade-off between the accuracy of the approximated model and the reduction of the computation time per model evaluation. That is why [17] builds on the ideas of [10] and proposes to also identify the nonlinear part using an empirical structure. In this sense, the present paper can be considered as an extension of the approach in [17]. Our procedure indeed uses simulation data to identify a simple nonlinear approximation of the complex nonlinear model. However, our approach differs from the methodology in [17] mainly in the fact that we propose a clear procedure in order to come up with the empirical structure for the nonlinear part (see below).

Since, the most CPU time is generally by far spent evaluating the state equation, we will only consider the simplification of the nonlinear function $f(.)$ (resp. $\bar{f}(.)$). Our procedure starts by identifying a linear model of the nonlinear equation (3):

$$x(k+1) = A_0\ x(k) + B_0\ u(k) + \delta_0 \tag{6}$$

We show that this can be done using a classical linear least square optimization. If this linear model is not sufficient to achieve the desired accuracy, we will use the same data to identify a quasi-Linear Parameter Varying model[3] to complete the linear model (6):

$$x(k+1) = A_0\ x(k) + B_0\ u(k) + \delta_0 + \underbrace{\sum_{m=1}^{M} \phi_m(x(k), u(k))\,[A_m x(k) + B_m u(k) + \delta_m]}_{\text{q-LPV model}} \tag{7}$$

In the q-LPV model, we see that M component linear models (A_m, B_m, δ_M) are summed after being weighted according to scalar scheduling functions $\phi_m(\cdot)$. The scheduling functions $\phi_m(\cdot)$ determine how the behaviour of the identified model should change according to the current operating point. Even using a relatively low number of component models and relatively simple scheduling functions $\phi_m(\cdot)$ a wide variety of nonlinear models can be accurately approximated using the structure (7) and this approximation will require, due to its simplicity, considerably less computation time per model evaluation.

The problem of identifying qLPV models has been extensively studied in nonlinear identification literature. The current methods can be divided into the following categories:

1. identify scheduling functions $\phi_m(\cdot)$ and models A_m, B_m, δ_m for $m = 1, \ldots, M$ simultaneously, see e.g [21, 3, 14],

[3]Quasi-Linear Parameter Varying (qLPV) models are also known under the names local linear models or fuzzy models.

2. functions $\phi_m(\cdot)$ are assumed known, identify only A_m, B_m, δ_m for $m = 1, \ldots, M$,

3. two stage methods: first determine scheduling functions $\phi_m(\cdot)$ then identify A_m, B_m, δ_m for $m = 1, \ldots, M$. [11, 2].

In this paper, we will introduce a new method to identify qLPV models which will use the specificities of the identification problem considered in this paper. The identification problem in this paper is indeed quite different from classical identification problems. The main specificity is that we will identify a simplified model[4] of the **known** nonlinear equation (3). This implies that we are able to generate simulation data $Z^N = \{u(1), x(1), \ldots, u(N), x(N)\}$ concerning the states $x(k)$ which are in practice unmeasured. This also implies that the data are not perturbed by noise. Finally, since we know the system we want to identify, we will be able to use this knowledge for the identification. Our qLPV identification approach is a two stage method:

1. Unlike most qLPV identification methods, we will first determine the linear component models. For this purpose, the complex nonlinear model (3) is linearized around each data pair $(u(k), x(k))$ in Z^N. The most significant linear behaviours among these N local linear models are subsequently determined using a similar POD technique as the one used for state reduction.

2. Then, in a second step, we parameterize the scheduling functions $\phi_m(\cdot, \theta_m)$ and estimate the optimal values for θ_m using the simulation data.

The optimal length M for the expansion in (7) is determined in order to achieve the desired accuracy.

2 qLPV models and notations

In this paper, we consider the approximation of complex non-linear state-space models using both linear and quasi-LPV model structures. In this section, we describe in details these model structures and introduce some useful notations.

A linear model structure for the state equation is given by:

$$x(k+1) = A\ x(k) + B\ u(k) + \delta \tag{8}$$

where $x(k) \in \mathbf{R}^{n\times 1}$ is the state vector at time k, $u(k)$ the input of the system, $\delta \in \mathbf{R}^{n\times 1}$ an offset vector and (A, B) the state-space matrices. The parameters that can be tuned to approximate the nonlinear system are the matrices A and B and the vector δ. These parameters can be collected in a parameter vector $\mathcal{V}$:

$$\mathcal{V} = \begin{pmatrix} vec(A)^T & vec^T(B) & vec(\delta)^T \end{pmatrix} \tag{9}$$

[4]It could seem odd to identify a model of another model (i.e. the first principle model) instead of directly identifying the system itself. However, it is important to note that collecting real-life data will only allow to identify a model of the input-output behaviour of the system while we are here also interested by the relation between inputs and states.

The column vector $\mathcal{V}$ is here defined using the operator $vec(.)$. For a matrix A, the row vector $vec(A)$ is in this paper obtained by putting aside of each other the rows of A. Note that this definition is different from the usual one. Using this notation, the linear model structure (8) can be rewritten as:

$$x(k+1) = \mathcal{Z}(x(k), u(k))\ \mathcal{V} \tag{10}$$

with $\mathcal{Z}(x(k), u(k))$ a matrix defined as:

$$\mathcal{Z}(x(k), u(k)) = \begin{pmatrix} I_n \otimes x^T(k) & I_n \otimes u^T(k) & I_n \end{pmatrix} \tag{11}$$

Let us now define what we mean by quasi-LPV model. However, let us first introduce the notion of LPV model. A LPV model is a time-varying linear combination of linear models (A_m, B_m, δ_m) $m = 1..M$:

$$x(k+1) = \sum_{m=1}^{M} \phi_m(p(k))\ \ (A_m\ x(k) + B_m\ u(k) + \delta_m)$$

We see that the coefficients ϕ_m of the linear combination are dependent on the value of a so-called scheduling parameter vector $p(k)$ which is supposed to be a measurable function of the time. These coefficients are thus scalar-valued static functions of $p(k)$ and are called scheduling functions. These scheduling functions determine how the modeled behaviour should change depending on the value of the scheduling parameter. In a LPV model, it is generally assumed that $p(k)$ is not related to the state vector $x(k)$ or to the input $u(k)$ of the system. In contrast, a quasi-LPV model or fuzzy model is a LPV model where

$$p(k) = \begin{pmatrix} x(k) \\ u(k) \end{pmatrix}$$

This yields:

$$x(k+1) = \sum_{m=1}^{M} \phi_m(x(k), u(k))\ \ (A_m\ x(k) + B_m\ u(k) + \delta_m) \tag{12}$$

or equivalently:

$$\begin{aligned} x(k+1) &= \left(\sum_{m=1}^{M} \phi_m(x(k), u(k)) \begin{pmatrix} A_m & B_m & \delta_m \end{pmatrix} \right) \begin{pmatrix} x(k) \\ u(k) \\ 1 \end{pmatrix} \\ &= \mathcal{Z}(x(k), u(k)) \left(\sum_{m=1}^{M} \phi_m(x(k), u(k))\ \mathcal{V}_m \right) \end{aligned} \tag{13}$$

where $\mathcal{V}_m$ $(m = 1...M)$ is defined similarly as $\mathcal{V}$ (see (9)) but here with A_m, B_m and δ_m. In order to be able to approximate a system using this model structure, different parameters have to be determined. First of all, the number M of components in the expansion and then the M vectors $\mathcal{V}_m$ containing the coefficients of the linear models and finally the M scheduling

functions $\phi_m(x(k), u(k))$. These scheduling functions are generally also parametrized using a parameter vector θ_m. An example of such parametrization is a simple linear form:

$$\phi_m(x(k), u(k), \theta_m) = \begin{pmatrix} x^T(k) & u^T(k) & 1 \end{pmatrix} \theta_m \tag{14}$$

Consequently, determining the scheduling functions is equivalent to determining the parameter vectors θ_m $(m = 1...M)$.

Remark. In (12)-(13), we have represented the qLPV model as a (state and input-dependent) linear combination of linear state-space models. State-space models are indeed a very compact parametrization. However, a drawback of choosing such models for the qLPV parametrization is that, even if each component model is stable, their linear combination is not necessarily stable. In order to circumvent this eventual problem, one could instead construct the qLPV model using finite length impulse response (FIR) component models. Indeed, a linear combination of FIR models is always stable. If we use M FIR component models of length L for the qLPV parametrization, the expressions (12)-(13) become:

$$\begin{aligned} x(k+1) &= \sum_{m=1}^{M} \phi_m(x(k), u(k)) \left(\sum_{i=0}^{L-1} g_m(i) u(k-i) \right) \\ &= \bar{\mathcal{Z}}(u,k) \left(\sum_{m=1}^{M} \phi_m(x(k), u(k)) \, \mathcal{G}_m \right) \end{aligned}$$

with $\bar{\mathcal{Z}}(u,k) = (u(k)\ u(k-1)...u(k-L+1)) \in \mathbf{R}^{1\times L}$ and $\mathcal{G}_m = (g_m(0)\ g_m(1)...g_m(L-1))^T \in \mathbf{R}^{L\times 1}$ $(m = 1...M)$. The procedure we present in the sequel to identify qLPV models of first-principle models can be applied for both the state-space and the FIR parametrization of qLPV models (see [4] for more details).

3 Problem statement

Suppose that a real-life system can be described by a non-linear state-space simulation model:

$$x(k+1) = f(x(k), u(k)) \tag{15}$$

$$y(k) = h(x(k), u(k)) \tag{16}$$

where $x(k) \in \mathbf{R}^{n\times 1}$ is the state vector at time k, $u(k)$ is a vector containing the inputs of the system. Finally, $f(.)$ and $h(.)$ are two static non-linear functions. We assume that this model perfectly describes the behaviour of the system and that the state vector $x(k)$ is a vector of physical quantities that are important to monitor[5]. This simulation model can be freely used for simulations; gradients can be computed (at least numerically). However, as mentioned in the introduction, this model is too complex for on-line monitoring (or for control design purposes). The main reason for that is that the static function $f(.)$ is so complex

[5]As mentioned in the introduction, this vector is generally a reduced order state vector obtained via proper orthogonal decomposition, but which can be easily back computed into the vector with physical meaning.

that the time required to update the state vector and the output vector with (15)-(16) is of the same order or even larger than the sampling interval T_s and this of course prevents the use of the model (15)-(16) for on-line control and monitoring purposes since monitoring and control algorithms generally require multiple evaluations of the model per sampling period. The complexity of the function $f(.)$ is mainly due to its large number of input and output arguments. Note that the function $h(.)$ is often less problematic since the dimension of $y(k)$ is typically much smaller that the dimension of $x(k)$. Consequently, in the sequel, we will only focus on the state equation (15).

To sum up, the available simulation model gives very useful information on the physical states of the system, but can only be used to monitor/control the system off-line (which is not very interesting). Our objective in the sequel will therefore be to keep (most) of this useful information on the physical states while allowing on-line monitoring (and control). For this purpose, we will apply to the simulation model (15) a typical input/ disturbance trajectory $u(k)$ and store the corresponding state-vector $x(k)$. It yields the following data set:

$$Z^N = \{x(k),\ u(k)\ \mid k = 1...N\}$$

Subsequently, we will use these data to identify a simplified model for (15). This simplified model

$$x(k+1) = f_{id}(x(k), u(k)) \tag{17}$$

is such that the state and the input vectors are identical to those in (15). To ensure that the simulation time of the identified model is much smaller than the one of the initial model (15), we will identify this model within model structures such as linear model structure or q-LPV model structures with a small expansion length M. The reduction of simulation time with respect to (15) is obvious if we replace the non-linear function $f(.)$ by a linear state-space model (8). For a q-LPV model (12), the reduction of computation time can be evidenced by comparing the complexity of the scheduling functions $\phi_m(.)$ and of the function $f(.)$. Both functions have the same number of input arguments i.e. $x(k)$ and $u(k)$. However, ϕ_m has only one output argument while $f(.)$ has n output arguments. Consequently, if the number M of scheduling functions $\phi_m(.)$ is much smaller than n (as it ought to be), evaluating the identified q-LPV model will require (much) less time than evaluating the initial model (15). One objective of the identification of the q-LPV model will therefore be to obtain the smallest expansion length M.

Reducing the computational burden is one aspect of the identification, the accuracy of the identified model with respect to the initial model is another one. For this purpose, we impose the following accuracy constraint:

$$\frac{\frac{1}{N}\sum_{k=1}^{N} |f(x(k),u(k)) - f_{id}(x(k),u(k))|^2}{\frac{1}{N}\sum_{k=1}^{N} |f(x(k),u(k))|^2} < \alpha_f \tag{18}$$

where the constant α_f is chosen to make a trade-off between the complexity and the accuracy of the simplified model. For the accuracy constraint (18) to be really effective, it is important that Z^N be a representative data set. We make therefore the following assumption:

Assumption 3.1 *It will be assumed that all the nonlinearities of the system that are relevant for its working area have been excited i.e., the available data Z^N is representative for the whole working area of the model (15).*

The assumption above states that the dataset Z^N enables the determination of a model (17) that is accurate in the entire working area. For nonlinear models it is not trivial to verify that a given data set has this property. In our identification procedure we will ignore this problem for now.

As already mentioned above, we consider two different model structures for the identification of (17): the linear model structure and the q-LPV model structure. Due to its simplicity, we will of course begin with the linear model structure:

Problem 1. *Given the state equation (15) and a data set Z^N fulfilling Assumption 3.1. Determine, based on Z^N, the coefficients $\mathcal{V}_0$ of a linear model (10):*

$$x(k+1) = f_{id}(x(k), u(k)) = \mathcal{Z}(x(k), u(k))\ \mathcal{V}_0$$

minimizing the prediction error $\frac{1}{N}\sum_{k=1}^{N} |f(x(k), u(k)) - f_{id}(x(k), u(k))|^2$.

If the linear model solving Problem 1 is sufficient for (18) to hold, there is no need to continue with q-LPV modeling. However, in most cases, a linear model will not be sufficient to fulfill (18) since the non-linear behaviour of $f(.)$ cannot be neglected. The nonlinear behaviour of $f(.)$ can be characterized by the difference between $f(.)$ and the linear model solving Problem 1

$$\Delta f(x(k), u(k)) \triangleq f(x(k), u(k)) - (\mathcal{Z}(x(k), u(k))\ \mathcal{V}_0) \tag{19}$$

This is the function $\Delta f(.)$ that we will try to approximate by a q-LPV model (13) in such a way that the accuracy constraint (18) holds (see Problem 2). This accuracy constraint can be rewritten in terms of Δf and of the to-be-identified q-LPV model as follows:

$$\frac{\frac{1}{N}\sum_{k=1}^{N}\left|\Delta f(x(k), u(k)) - \mathcal{Z}(x(k), u(k))\ \left(\sum_{m=1}^{M}\phi_m(x(k), u(k), \theta_m)\ \mathcal{V}_m\right)\right|^2}{\frac{1}{N}\sum_{k=1}^{N}|f(x(k), u(k))|^2} < \alpha_f \tag{20}$$

Problem 2. *Given the function (19) and a data set Z^N fulfilling Assumption 3.1. Determine, based on Z^N, the q-LPV model (13) with the smallest expansion M in such a way that (20) holds.*

To determine the optimal q-LPV model of Problem 2, we obviously need to determine the length M of the expansion, but also the coefficients $\mathcal{V}_m$ of the M linear models and the M vectors θ_m parametrizing the scheduling functions $\phi_m(.)$ (see (14)).

Remark. The constant α_f in (20) must be chosen with care. Indeed, if we choose α_f too small, we have the risk of overfitting. It is therefore advisable to validate the obtained q-LPV model with another data set.

4 Solution to the two identification problems

4.1 Problem 1: best linear approximation

Problem 1 corresponds to the following linear least-square problem:

$$\mathcal{V}_0 = arg\min_{\mathcal{V}} \frac{1}{N}\sum_{k=1}^{N} |f(x(k),u(k)) - \mathcal{Z}(x(k),u(k))\ \mathcal{V}|^2 \tag{21}$$

with $x(k)$ and $u(k)$ in the data set Z^N

4.2 Problem 2: q-LPV approximation

4.2.1 Local linearization

If the best linear approximation determined in the previous subsection is not sufficient to achieve the accuracy constraint (18), we will need to develop a q-LPV approximation of the function Δf (see (19)). Even though Δf represents the non-linear behaviour of the state equation (15), the non-linear function $\Delta f(x,u)$ can be approximated by a linear mapping at each data point in Z^N (at each point on the trajectory) by linearizing $\Delta f(x,u)$ around each of these points $(x(k),u(k))$:

$$\Delta f(x,u) \approx \Delta f(x(k),u(k)) + \mathcal{A}_{(x(k),u(k))}\ (x - x(k)) + \mathcal{B}_{(x(k),u(k))}\ (u - u(k))$$

with

$$\mathcal{A}_{(x(k),u(k))} = \left.\frac{\partial \Delta f(x,u)}{\partial x}\right|_{x=x(k),\quad u=u(k)} \qquad \mathcal{B}_{(x(k),u(k))} = \left.\frac{\partial \Delta f(x,u)}{\partial u}\right|_{x=x(k),\quad u=u(k)}$$

The above expression can be rewritten as follows:

$$\Delta f(x,u) \approx \mathcal{A}_{(x(k),u(k))}\ x + \mathcal{B}_{(x(k),u(k))}\ u + \partial_{(x(k),u(k))}$$

with

$$\partial_{(x(k),u(k))} = \Delta f(x(k),u(k)) - \mathcal{A}_{(x(k),u(k))}\ x(k) - \mathcal{B}_{(x(k),u(k))} u(k)$$

It is important to note that the non-linear mapping $\Delta f(x,u)$ at the data point $(x(k),u(k))$ is perfectly described by the linear mapping (i.e. the local linear model) developed above:

$$\begin{aligned}\Delta f(x(k),u(k)) &= \mathcal{A}_{(x(k),u(k))}\ x(k) + \mathcal{B}_{(x(k),u(k))}\ u(k) + \partial_{(x(k),u(k))} \quad (22)\\ &= \left(\begin{array}{ccc}\mathcal{A}_{(x(k),u(k))} & \mathcal{B}_{(x(k),u(k))} & \partial_{(x(k),u(k))}\end{array}\right)\left(\begin{array}{c}x(k)\\ u(k)\\ 1\end{array}\right)\\ &= \mathcal{Z}(x(k),u(k))\ \Upsilon_{(x(k),u(k))} \quad (23)\end{aligned}$$

where $\mathcal{Z}(x(k),u(k))$ is defined in (11) and $\Upsilon_{(x(k),u(k))}$ is a column vector defined similarly as $\mathcal{V}$ (see (9)).

Note that such a local linear model (23) can easily be computed (e.g. numerically) using the available knowledge of the non-linear function $f(.)$. Note also that this model can be derived for the N data points yielding a set of N vectors $\Upsilon_{(x(k),u(k))}$ $(k = 1...N)$ describing the N local linear models.

Using (23), the constraint (20) can be rewritten as follows:

$$\frac{\frac{1}{N}\sum_{k=1}^{N}\left|\mathcal{Z}(x(k),u(k))\left(\Upsilon_{(x(k),u(k))} - \sum_{m=1}^{M}\phi_m(x(k),u(k),\theta_m)\ \mathcal{V}_m\right)\right|^2}{\frac{1}{N}\sum_{k=1}^{N}|f(x(k),u(k))|^2} < \alpha_f \quad (24)$$

Solving Problem 2 (i.e. determining the smallest M (and the corresponding $\mathcal{V}_m$ and θ_m) such that (24) holds) can be tackled using the following optimal algorithm.

Algorithm 4.1 *An optimal algorithm to solve Problem 2 is as follows.*

1. *Set* $M = 1$.
2. *Based on the data* Z^N, *determine* θ_m, $\mathcal{V}_m$ $(m = 1...M)$ *as follows:*

$$arg \min_{\mathcal{V}_m,\ \theta_m\ (m=1...M)} \frac{1}{N}\sum_{k=1}^{N}\left|\mathcal{Z}(x(k),u(k))\left(\Upsilon_{(x(k),u(k))} - \sum_{m=1}^{M}\phi_m(x(k),u(k),\theta_m)\ \mathcal{V}_m\right)\right|^2 \quad (25)$$

3. *Verify whether the q-LPV model identified in step 2 satisfies (24). If it is so, STOP. Otherwise, set* $M = M + 1$ *and go to step 2.*

Unfortunately, the optimization problem (25) is a practically unsolvable problem even if a particular structure for the parametrization of the scheduling functions is fixed a-priori. In the sequel, we nevertheless propose a sub-optimal procedure which allows us to determine the vectors $\mathcal{V}_m$ and θ_m $(m = 1...M)$ of a q-LPV model achieving for $k = 1...N$:

$$\overbrace{\mathcal{Z}(x(k),u(k))\Upsilon_{(x(k),u(k))}}^{\Delta f(x(k),u(k))} \approx \mathcal{Z}(x(k),u(k)) \left(\sum_{m=1}^{M} \phi_m(x(k),u(k),\theta_m)\ \mathcal{V}_m \right)$$

This procedure is presented in Subsections 4.2.2 and 4.2.3 and can thus replace the second step of Algorithm 4.1. This leads to the following feasible algorithm for solving Problem 2.

Algorithm 4.2 *1. Set $M = 1$.*

2. *Based on the data Z^N, determine θ_m, $\mathcal{V}_m$ $(m = 1...M)$ using the procedure presented in Subsections 4.2.2 and 4.2.3.*

3. *Verify whether the q-LPV model identified in step 2 satisfies (24). If it is so, STOP. Otherwise, set $M = M + 1$ and go to step 2.*

The procedure presented in Subsections 4.2.2 and 4.2.3 first determines the M linear models of the expansion and thereafter the parameter vectors θ_m parametrizing the scheduling functions.

4.2.2 Suboptimal determination of $\mathcal{V}_m$ $(m = 1...M)$

To determine the vectors $\mathcal{V}_m$ parametrizing the linear models, we will suppose that the scheduling functions have not any structure, but are instead just scalar coefficients $\beta_{m,k}$ i.e. $\phi_m(x(k),u(k),\theta_m) = \beta_{m,k}$ $(k = 1...N)$ $(m = 1...M)$.

We then notice by inspecting (25) that if we could find an expansion $\sum_{m=1}^{M} \beta_{m,k}\ \mathcal{V}_m$ which is equal to $\Upsilon_{(x(k),u(k))}$ for $k = 1...N$, we would have found a perfect q-LPV model[6]. Based on this observation, it is justified to use the following optimization problem to determine the coefficients $\mathcal{V}_m^{opt}$ of the linear models in the q-LPV expansion

$$arg \min_{\mathcal{V}_m\ \beta_{m,k}} \frac{1}{N} \sum_{k=1}^{N} \left| \Upsilon_{(x(k),u(k))} - \sum_{m=1}^{M} \beta_{m,k}\ \mathcal{V}_m \right|^2 \tag{26}$$

As opposed to (25), (26) is easily solvable. Indeed, since $\Upsilon_{(x(k),u(k))}$ and $\mathcal{V}_m$ are vectors, solving this optimization problem is equivalent to applying a proper orthogonal decomposition of the matrix

$$\Xi = \begin{pmatrix} \Upsilon_{(x(1),u(1))} & \Upsilon_{(x(2),u(2))} & ... & \Upsilon_{(x(N),u(N))} \end{pmatrix}$$

Consequently, the optimal parameter vectors $\mathcal{V}_m^{opt}$ $(m = 1..M)$ are given by the first M column vectors of the matrix U in the singular value decomposition of $\Xi = U\Sigma V$ and the optimal coefficients $\beta_{m,k}^{opt}$ are given by:

$$\beta_{m,k}^{opt} = \Upsilon_{(x(k),u(k))}^T\ \mathcal{V}_m^{opt} \quad k = 1...N \quad m = 1...M$$

[6]Note that a perfect match (i.e. $\sum_{m=1}^{M} \beta_{m,k}\ \mathcal{V}_m = \Upsilon_{(x(k),u(k))}\ \forall k$) is possible if M is chosen equal to the dimension of the coefficient vectors Υ and $\mathcal{V}_m$. However, since M will be typically much smaller than the dimension n of the state vector, it will be also much smaller than the dimension of $\mathcal{V}_m$.

Remark. It is important to note that the optimal parameter vectors $\mathcal{V}_m^{opt}$ $(m = 1..M)$ are determined by exploiting the possibility we have to linearize the nonlinear function $\Delta_f(x, u)$ or, in other words, by exploiting our knowledge of $\Delta_f(x, u)$. The optimal parameter vectors $\mathcal{V}_m^{opt}$ $(m = 1..M)$ are consequently not "identified" using the data Z^N. The data are only used to determine at which points $\Delta_f(x, u)$ must be linearized.

4.2.3 Determination of $\phi_m(x(k), u(k), \theta_m)$ $(m = 1...M)$

In the previous subsection, we have determines an expansion such that:

$$\Delta f(x(k), u(k)) \approx \mathcal{Z}(x(k), u(k)) \left(\sum_{m=1}^{M} \beta_{m,k}^{opt} \ \mathcal{V}_m^{opt} \right)$$

for all $\{x(k),\ u(k)\}$ in Z^N. In this subsection, we will replace the unstructured coefficients $\beta_{m,k}$ by structured scheduling functions such as in (14). This step is absolutely necessary. Indeed, these coefficients $\beta_{m,k}$ only allow to compute $\Delta f(x, u)$ at the N data points in Z^N. Consequently, if these coefficients are not replaced, it would be impossible to compute $\Delta f(x, u)$ for points x and u which are not in the simulation data.

The first step here is to choose a structure for the scheduling functions $\phi_m(.)$. Since the functions $\phi_m(\cdot, \theta_m)$ are not required to have any physical interpretation, we are free to chose any model structure we would like. The choice for structure of these functions is a tradeoff between complexity and flexibility. An example of a simple structure is given in (14). The main advantage of this affine parametrization is that the structure is linear in its parameter vector θ_m and the number of parameters is relatively small. The drawback of this model structure is that such a structure may not be flexible enough to allow for an accurate identified model. If more complex scheduling functions are required, it is possible to use more complex structures using for instance radial basis functions [20] or fuzzy membership functions [2]. To select an appropriate structure for the scheduling functions, it is often helpful to look at plots of the computed coefficients $\beta_{m,k}$ as a function of the states and inputs $(x(k), u(k)) \in Z^N$.

Once a structure has been selected, all that remains is to estimate the parameter vectors θ_m $(m = 1...M)$ of the scheduling functions for the M linear models determined in the previous subsection. Given the optimal criterion (25), the most optimal way to determine these vectors θ_m $(m = 1...M)$ is as follows:

$$\theta_1^{opt}, \ldots, \theta_M^{opt} = arg \min_{\theta_m \ (m=1..M)} \frac{1}{N} \sum_{k=1}^{N} \left| \mathcal{Z}(x(k), u(k)) \left(\Upsilon_{(x(k),u(k))} - \sum_{m=1}^{M} \phi_m(x(k), u(k), \theta_m) \ \mathcal{V}_m^{opt} \right) \right|^2 \tag{27}$$

with the linear models $\mathcal{V}_m^{opt}$ as determined in the previous subsection.

Note that if the scheduling functions $\phi_m(\cdot, \theta_m)$ are chosen linear in θ_m such as in (14), the resulting minimization problem is a linear least squares problem which can be easily solved. For more complex structures that are not linear in θ_m, the parameters of the scheduling

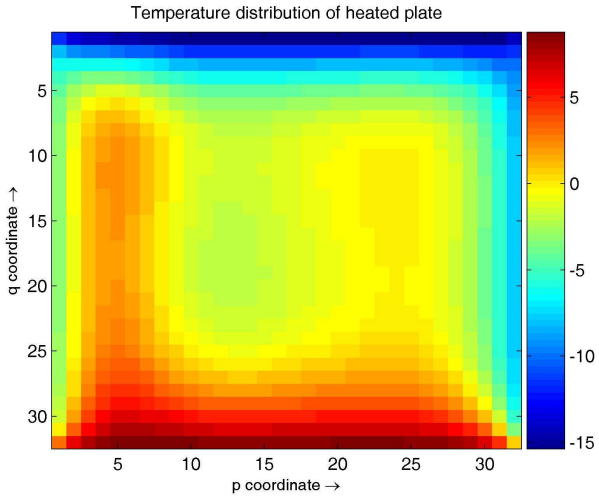

Figure 1: Simulated heated plate example. The plate is heated or cooled along the complete edges on all sides. The physical equation are solved on a 32 by 32 grid using implicit Euler integration.

functions have to be determined using nonlinear optimization techniques. For the previously mentioned structures involving radial basis functions or fuzzy membership functions, good initial conditions can be obtained by applying clustering methods on the computed coefficients $\beta_{m,k}$ [2].

Remarks. Even for simple linear parameterizations of the scheduling functions $\phi_m(\cdot, \theta_m)$ such as (14), determining parameters vectors θ_m can sometimes be difficult. The difficulties occur when the least squares problem (27) is ill-conditioned. In such cases it is advisable to approximately solve the least squares criterion (27) using a truncated SVD, to effectively reduce the degrees of freedom in the least squares problem. For more details, see [7, 16]. Note also that other methods than the one presented above exist for the determination of the scheduling functions ϕ_m; see e.g. [9]

5 Simulation example

5.1 Simulation model

In this section, in order to illustrate our results, we consider an iron solid square plate that is heated and cooled at the edges, see Figure 1. The considered plate is 0.01 meter tick and its sides are 0.5 meter long. The plate is heated via four heating surfaces that completely cover its four sides (the temperature along each of the four sides of the plate can therefore be supposed uniform and equal to the temperature of the corresponding heating surfaces). The temperatures of these heating surfaces can be freely chosen and will therefore be the four inputs of the system.

A model of the plate can be constructed from the energy balance over an infinitesimal

small surface area $\mathrm{d}p\,\mathrm{d}q$ at spatial coordinates (p,q)[7]:

$$\rho\, c\, h\, \mathrm{d}p\,\mathrm{d}q\, \frac{\partial T(p,q,t)}{\partial t} = \nabla \cdot J(p,q,t), \tag{28}$$

in which $h = 0.01\ m$ is the tickness of the plate, $\rho = 7870\ kg/m^3$ is the mass-density of the plate, $c = 4690\ J/(kg\ K)$ is the heat capacity of the plate. The ∇ operator in the above expression is defined as:

$$\nabla = \begin{pmatrix} \frac{\partial}{\partial p} \\ \frac{\partial}{\partial q} \end{pmatrix}. \tag{29}$$

The function $J(p,q,t) : \mathbf{R}^3 \to \mathbf{R}^2$ is a vector function that represents the heat transfer at location (p,q) and at time t. The heat transfer function $J(p,q,t)$ is given by:

$$J(p,q,t) = \lambda(T(p,q,t))\nabla T(p,q,t), \tag{30}$$

with $\lambda(T(p,q,t))$ a temperature-dependent heat conductivity coefficient. Here we assume that $\lambda(T(p,q,t))$ is given by:

$$\lambda(T(p,q,t)) = \frac{1}{2560}T(p,q,t)^3 + \frac{3}{8}T(p,q,t) + 80. \tag{31}$$

The heat conductivity as a function of temperature is plotted in Figure 2. It should be noted that this particular heat conductivity function is not based on physics. Instead the function was chosen for the model to have some reasonable nonlinear characteristics in order to be appropriate to test our model approximation techniques.

The model (28)-(31) can be rewritten in the nonlinear state-space form using a finite differences method. The finite differences method imposes a grid on the plate. We have chosen to use a grid of 32 by 32 elements. In each grid-cell, the temperature and all other material properties are assumed to be constant. In the remainder we shall denote the p-coordinate (horizontal position) of a column of grid-cells as $p(i_p)$ with $i_p \in \{1,\ldots,32\}$. Similarly the q-coordinate (vertical position) of a row of cells is denoted as $q(i_q)$ with $i_q \in \{1,\ldots,32\}$. Spatial derivatives with respect to spatial coordinate p and q are approximated using finite differences. For example, the spatial derivative with respect to p is approximated using the following equation:

$$\left.\frac{\partial T(p,q,t)}{\partial p}\right|_{p=p(i_p),q,t} \approx \frac{T(p(i_p+1),q,t) - T(p(i_p),q,t)}{p(i_p+1) - p(i_p)}. \tag{32}$$

The partial derivatives with respect to q are approximated in the same manner.

After approximating the spatial derivatives using finite differences, the resulting model $\bar{x}(k) = \bar{f}(\bar{x}(k), u(k))$ is a set of 1024 nonlinear ordinary differential equations. These equations are solved using an implicit Euler method, using an integration time interval of 20

[7]We use the spatial coordinates (p,q) instead of the more conventional spatial coordinates (x,y) to prevent confusion with the state vector $x(k)$ and measurement vector $y(k)$.

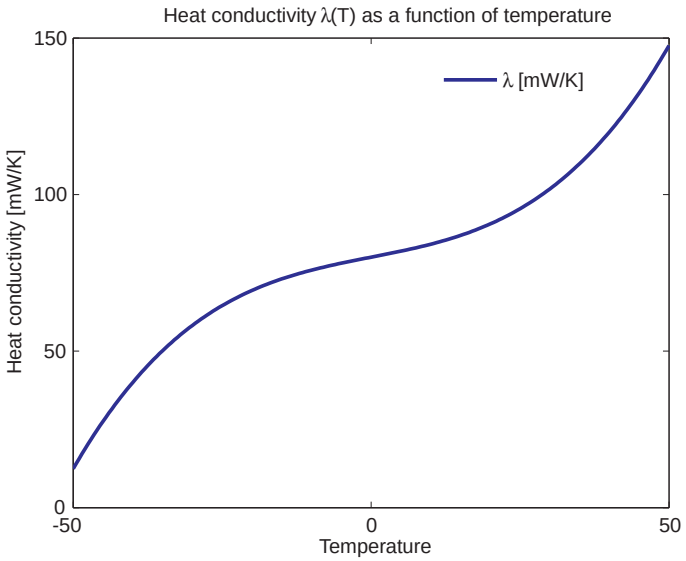

Figure 2: Plot of the chosen temperature-dependent heat conductivity function $\lambda(T)$ as given by (31).

seconds. The state vector $\bar{x}(k)$ in this model represents the temperature at each of the 1024 grid-cells while the input $u(k)$ is a vector containing the temperatures applied at each of the sides. The state-dimension of this model can nevertheless be easily reduced to 25 using classical POD techniques [1] and this without loosing the physical interpretation: the 1024 initial states can indeed be accurately estimated from the 25 states of the final model. This final model is of the form (15):

$$x(k+1) = f(x(k), u(k)) \tag{33}$$

with $x(k) \in \mathbf{R}^{25\times 1}$ and $u(k) \in \mathbf{R}^{4\times 1}$. This model is quite slow. Indeed, 131 seconds are required to perform 2500 model evaluations. In the sequel, we will see that, using our q-LPV model approximation method, we will be able to reduce this computation time by a factor 350 and keep the relative prediction error (18) below $\alpha_f = 10^{-5}$ (the desired accuracy).

5.2 qLPV identification

In order to be able to simplify the nonlinear model (33) using our q-LPV approximation technique, we will need to collect data by simulating (33). Here, we have collected $N = 2500$ simulation data $Z^N = \{x(1), u(1), \ldots, x(N), u(N)\}$. The input vector which has been chosen to generate these data is a random stair sequence i.e. a sequence which changes values only every 40 seconds. The new values taken after each 40 seconds are independent of the past and future values and are taken equal to realizations of a normal distribution with zero mean and variance $Q = diag(225, 225, 100, 100)$. In Figure 3, the first entry of the vector $u(k)$ is represented from $k = 1$ to $k = 1000$.

As proposed in Problem 1, we will first use the data to approximate (33) with a linear model (10) and verify whether we can achieve a relative prediction error smaller than $\alpha_f =$

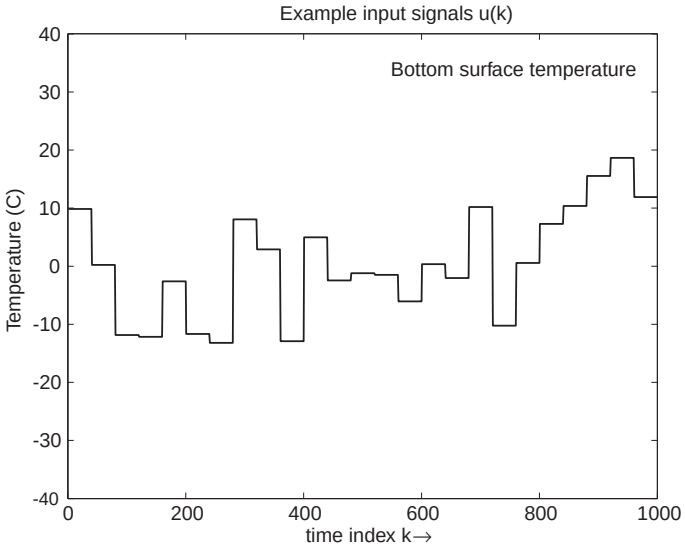

Figure 3: First entry of the vector $u(k)$ used for the data collection from $k = 1$ to $k = 1000$. This entry corresponds to the temperature applied at the bottom of the plate.

10^{-5} with such a simple model. The parameter vector $\mathcal{V}_0$ corresponding to this linear model is obtained via the least-square problem (21). This linear model is of course much faster than the nonlinear model (33): only 0.012 second is required to achieve 2500 model evaluations i.e. a reduction with a factor 10000. Moreover, the linear model achieves a relative prediction error of 1.7 10^{-4}. Note thus that this linear model is already able to predict pretty well the behaviour of the system. However, the achieved relative error is larger than the desired $\alpha_f = 10^{-5}$. Consequently, in order to obtain an approximation of (33) with a relative prediction error smaller than the desired α_f, we need to extend the best linear approximation $\mathcal{V}_0$ with a q-LPV model as proposed in Problem 2.

For this purpose, we have applied Algorithm 4.2. In this algorithm, for each value of M, the procedure presented in Subsections 4.2.2 and 4.2.3 must be followed to find the (optimal) q-LPV model with this expansion length. In this procedure, the M linear models of the expansion are first determined. For this purpose we linearize the nonlinear model (33) around all data points $(x(k), u(k)) \in Z^N$. This delivers $\Upsilon_{(x(k),u(k))}$ for $k = 1...2500$. We then use a SVD decomposition technique to solve (26) i.e. to determine the optimal expansion $\sum_{m=1}^{M} \beta_{m,k}^{opt}\ \mathcal{V}_m^{opt}$ with unstructured coefficients. The M linear models specified by $\mathcal{V}_m^{opt}$ are then used to determine the M scheduling functions via the optimization problem (27). For this purpose, we need to define a parametrization for the scheduling functions. In the sequel, we present how we have determined this parametrization for the iteration step where $M = 5$.

To find an appropriate model structure for the scheduling functions $\phi_m(x(k), u(k), \theta_m)$, we plot the relation existing between the coefficients $\beta_{m,k}^{opt}$ and the corresponding states $x(k)$ or the corresponding input $u(k)$ for $k = 1...N$. In Figure 4, we present such a plot. In this plot, the coefficients $\beta_{3,k}^{opt}$ is plotted against the fourth entry $u_4(k)$ of the input vector $u(k)$. Based on the shape of the curve in Figure 4, we can conclude that the coefficients $\beta_{3,k}^{opt}$ seem to behave as a third order polynomial function of $u(k)$. Similar plots of coefficients $\beta_{m,k}^{opt}$

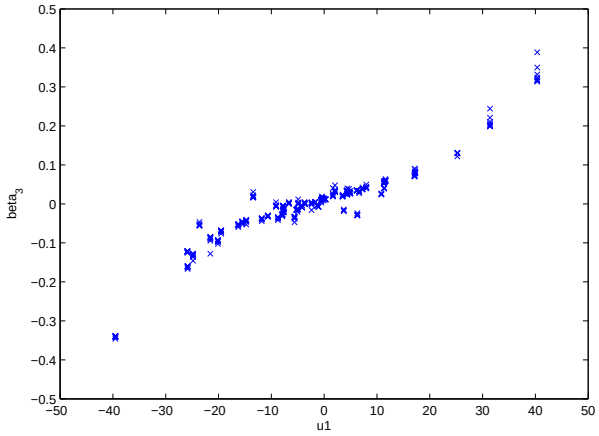

Figure 4: $\beta_{3,k}$ versus the fourth input component $u_4(k)$ (left side temperature).

against entries of $x(k)$ suggest that a similar relation exists between $\beta_{m,k}^{opt}$ and $x(k)$. This is not surprising since the nonlinearity of the original model (33) is caused by $\lambda(T)$ which is a third order polynomial function of the temperature. Finally, after some trials and errors, we have chosen the following structure for the five scheduling functions $\phi_m(x(k), u(k), \theta_m)$:

$$\phi_m(x_{red}(k), u(k), \theta_m) = [1 \ [x(k)]_{1:8}^T \ u(k)^T \ [x(k)]_{1:4}^{3\ T} \ [u(k)^3]^T]\theta_m, \tag{34}$$

with θ_m a parameter vector of dimension 21. Since we have chosen a scheduling function that is linear in the parameter vectors θ_m for $m = 1, \ldots, 5$, the optimization problem (27) delivering the five optimal parameter vectors θ_m^{opt} is here a least square problem.

The solution of Algorithm 4.2 that we have found following this procedure is a q-LPV model with five component models ($M = 5$). Indeed, $M = 5$ is the smallest value of M for which the corresponding q-LPV model satisfies (20) with $\alpha_f = 10^{-5}$. The nonlinear model can thus be approximated with a model made up of the summation of this q-LPV model ($M = 5$) and the best linear approximation. This approximation achieves the required accuracy since the relative prediction error is smaller than α_f. Moreover, only 0.37 second is required to achieve 2500 model evaluations whereas 131 seconds were required with the initial nonlinear model (33). Consequently, we have been able to reduce the computation time by a factor 350 while keeping a very high accuracy for the prediction of the behaviour of the system.

As said previously, the solution of Algorithm 4.2 is a q-LPV model with five component models ($M = 5$). To obtain this model, we had also to construct q-LPV models with an expansion length $M = 1$ till $M = 4$ (see Algorithm 4.2). For the sake of completion, we give, in Table 1, the achieved relative error obtained with the estimation data Z^N as well as the computation time to achieve 2500 model evaluations for these four intermediary q-LPV models.

As a final verification of the quality of our approximations, we have computed the relative prediction errors obtained with a set of 2500 new data points generated by the initial

M	Relative error	computation time ([s])
1	$0.95\ 10^{-4}$	0.10
2	$0.58\ 10^{-4}$	0.17
3	$0.22\ 10^{-4}$	0.24
4	$0.12\ 10^{-4}$	0.31

Table 1: Achieved relative error (left-hand side of (18)) obtained with the estimation data Z^N and the computation time to achieve 2500 model evaluations for the four intermediary q-LPV models

system (33). The obtained relative error for the q-LPV model with five components (i.e. the solution of Algorithm 4.2) is $0.58\ 10^{-4}$. As a comparison, the obtained relative error for the best linear approximation is $3.34\ 10^{-4}$.

5.3 Filtering example

In this section we will use the qLPV model as identified in the previous section in a state estimation example. In the state estimation example we will estimate the temperature at all positions of the heated plate (i.e. the full state $\bar{x}(k)$ of the system) using just five temperature measurements at each time step. In order to arrive at a proper state estimation problem, we first extend the original heated plate model using a additive process noise term:

$$\bar{x}(k+1) = \bar{f}(\bar{x}(k), u(k)) + w(k), \tag{35}$$

with $\bar{f}(x(k), u(k))$ the physical nonlinear model derived in the previous section using a finite difference method. The vector $\bar{x}(k) \in \mathbf{R}^{1024\times 1}$ is the state vector before POD reduction and represents thus the temperature $T(p(i_p), q(i_q), k)$ in each of the 1024 grid-cells ($i_p = 1...32$, $i_q = 1...32$). The additive noise $w(k)$ is chosen as a zero mean stationary white noise signal with:

$$\mathbb{E}\{w(k)\} = 0 \tag{36}$$

$$\mathbb{E}\{w(k)w(k)^T\} = 0.01\ I \tag{37}$$

Apart from a process noise, the model was extended with a observation model, i.e. a model that describes the available measurements. In this example the measurement model is given by the following linear model:

$$y(k) = \begin{bmatrix} T(p(1), q(1), k) \\ T(p(32), q(1), k) \\ T(p(16), q(16), k) \\ T(p(1), q(32), k) \\ T(p(32), q(32), k) \end{bmatrix} + v(k), \tag{38}$$

with $v(k) \in \mathbf{R}^{5\times 1}$ a zero mean stationary white noise signal with:

$$\mathbb{E}\{v(k)\} = 0 \tag{39}$$

$$\mathbb{E}\{v(k)v(k)^T\} = I \tag{40}$$

As can be seen from the above equation, the temperature is measured in the center and at the four corners of the plate. These five temperatures are of course entries of $\bar{x}(k)$.

Using the model (35)-(40), $N = 500$ states $\bar{x}(k)$ and measurements $y(k)$ were generated. The input $u(k)$ used to generate these new states and measurements were generated using the procedure described in the previous subsection.

The input $u(k)$ and the output $y(k)$ will now be used to estimate the state $\bar{x}(k)$ (i.e. the temperature of the plate in each grid-cell and for $k = 1...500$) using a filtering technique. The state estimate will be denoted by $\hat{\bar{x}}(k)$. In order to obtain this estimate, not only input-output data are required, but also a model. We will here consider the estimate obtained with three different models. The first model is the nonlinear model (33) of order 25 obtained from the original nonlinear model (35) of order 1024 by applying a classical POD technique. The second model is the qLPV model for $M = 5$ that was identified from (33) in the previous subsection and finally the third model is the linear model also identified in the previous subsection. Since the first two models are nonlinear the Unscented Kalman Filter (UKF) [12] was used to estimate $\hat{\bar{x}}(k)$. The UKF is an extension of the normal Kalman filter that is able to handle nonlinear models. For the linear model, the normal Kalman filter was used to obtain[8] $\hat{\bar{x}}(k)$.

The three models will be compared in their ability to estimate the state vector $\bar{x}(k)$. For this purpose, we introduce the following relative error $\mathcal{E}$:

$$\mathcal{E} = \frac{\frac{1}{N}\sum_{k=1}^{N} |\bar{x}(k) - \hat{\bar{x}}(k)|^2}{\frac{1}{N}\sum_{k=1}^{N} |\bar{x}(k)|^2} \tag{41}$$

with $\hat{\bar{x}}(k)$ the state estimate and $\bar{x}(k)$ the actual value obtained from simulation (here $N = 500$). Besides the estimation accuracy, the three models will also be compared with respect to the computational time required to perform the 500 estimations.

The results using each of the models are summarized in Table 2. This table confirms our previous results. Indeed, we observe that the qLPV model is a perfect trade-off between computational efficiency and state estimation accuracy.

6 Conclusions

In this paper, we have presented a method to approximate a computationally expensive first principles model by a qLPV model. Besides approximating the original model accurately and

[8]The linear and qLPV model are both approximations of the reduced-order model (33) obtained after POD technique. This means that the filtering technique for the three models delivers an estimate of the reduced-order state $x(k)$. At the top of page 3, it is explained how to get the estimate of the actual state $\bar{x}(k)$ from the one of $x(k)$.

Model	Filter	Relative error $\mathcal{E}$	Time required for 500 state estimates
Original nonlinear model (after POD reduction)	UKF	$2\ 10^{-4}$	1100 s
Identified qLPV model ($M = 5$)	UKF	$4\ 10^{-4}$	8.6 s
Identified linear model	KF	$33\ 10^{-4}$	3.3 s

Table 2: State estimation results using the original reduced-order physical model (33), the identified qLPV model (for $M = 5$) and an identified linear model. The relative estimation error $\mathcal{E}$ is calculated via (41). For the first two nonlinear models the Unscented Kalman Filter (UKF) was used, for the linear model we used the normal Kalman filter.

conserving the physical interpretation of the states, the resulting qLPV model has generally a much simpler structure than the original model. This in turn implies that the CPU time associated with each model evaluation is generally considerably reduced, allowing the use of these models for online monitoring. Moreover, we have evidenced the applicability of the method on an example.

References

[1] P. Astrid. *Reduction of Process Simulation models - a Proper Orthogonal Decomposition Approach*. PhD thesis, Eindhoven University of Technology, 2004.

[2] R. Babuška. *Fuzzy Modeling for Control*. Kluwer Academic, Boston/Dordrecht/London, 1998.

[3] B. Bamieh and L. Giarré. Identification of linear parameter varying models. *International Journal of Robust and Nonlinear Control*, 12:841–853, 2002.

[4] R. Bos. *Monitoring of Large Scale First Principles Models*. PhD thesis, Delft University of Technology, 2006.

[5] R. Bos, X. Bombois, and P.M.J. Van den Hof. Accelerating large-scale non-linear models for monitoring and control using spatial and temporal correlations. In *Proceedings of American Control Conference 2004*, pages 3705–3710, Boston, 2004.

[6] N. Ganesh and L. T. Biegler. A robust technique for process flowsheet optimization using simplified model approximations. *Computers and Chemical Eng.*, 11(6):553–565, 1987.

[7] G. H. Golub and C. F. Van Loan. *Matrix Computations*. John Hopkins Univ. Press, Baltimore, USA, second edition, 1989.

[8] P. Holmes, J. L. Lumley, and G. Berkooz. *Turbulence, Coherent Structures, Dynamical Systems and Symmetry*. Cambridge University Press, 1996.

[9] K. Hsu, T. Vincent, C. Novara, M. Milanese, and K. Poolla. Identification of nonlinear maps in interconnected systems. In *Proceedings of 44th IEEE Conference on Decision and Control and European Control Conference*, pages 6430–6435, Seville, Spain, 2005.

[10] L. Huisman and S. Weiland. Identification and model predictive control of an industrial glass-feeder. In *Proceedings 13th IFAC Symposium on System Identification*, pages 1685–1689, Rotterdam, 2003.

[11] T. A. Johansen and B. A. Foss. Identification of non-linear system structure and parameters using regime decomposition. *Automatica*, 31(2):321–326, 1995.

[12] S. Julier, J. Uhlmann, and H.F. Durrant-Whyte. A new method for the non-linear transformation of means and covariances in filters and estimators. *IEEE Trans. Autom. Control*, 45(3):477–482, 2000.

[13] S. Lall, J. E. Marsden, and S. Glavaski. A subspace approach to balanced truncation for model reduction of nonlinear control systems. *International Journal on Robust and Nonlinear Control*, 12:519–535, 2002.

[14] L. H. Lee and K. Poolla. Identification of linear parameter-varying systems using nonlinear programming. *Journal of Dynamic Systems, Measurement, and Control*, 121:71–78, 1999.

[15] W. Marquardt. Nonlinear model reduction for optimization based control of transient chemical processes. *AIChE Symp. Ser. 326*, 98:12–42, 2002.

[16] R. Murray-Smith and T. A. Johansen, editors. *Multiple Model Approaches to Modelling and Control*. The Taylor and Francis Systems and Control Book Series. Taylor and Francis, 1997.

[17] R. Romijn, L. Ozkan, S. Weiland, J. Ludlage, and W. Marquardt. A grey-box modeling approach for the reduction of nonlinear systems. *Journal of Process Control*, 18:906–914, 2008.

[18] J. M. A. Scherpen. Balancing for nonlinear systems. *Systems and Control Letters*, pages 143–153, 1993.

[19] J. van den Berg. *Model Reduction for Dynamic Real-Time Optimization of Chemical Processes*. PhD thesis, Delft University of Technology, 2005.

[20] V. Verdult. *Nonlinear System Identification - A State-Space Approach*. PhD thesis, University of Twente, 2002.

[21] V. Verdult, L. Ljung, and M. Verhaegen. Identification of composite local linear state-space models using a projected gradient search. *Int. J. Control*, 75:1385–1398, 2002.