Data-driven Linear Parameter Varying Controller Synthesis Using Iterative Feedback Tuning

An I/O Approach

N. Willemstein

Master of Science Thesis

TU Delft Delft University of Technology

Delft Center for Systems and Control

Data-driven Linear Parameter Varying Controller Synthesis Using Iterative Feedback Tuning

An I/O Approach

Master of Science Thesis

For the degree of Master of Science in Systems and Control at Delft University of Technology

N. Willemstein

January 6, 2016

Faculty of Mechanical, Maritime and Materials Engineering (3mE) · Delft University of Technology

Abstract

Controller tuning is commonly performed to improve the closed loop response. The tuning of a controller can be done through either manual or numerical methods. Unfortunately manual tuning of controllers can be both cumbersome and time consuming. Furthermore one can't give any guarantees with respect to the optimality of a, manually, tuned controller. A commonly used numerical method for optimizing controller in closed loop is Iterative Feedback Tuning (IFT). It accomplishes controller optimization by estimating the gradient of a cost function by using two dedicated (closed loop) experiments. The original IFT framework, as proposed by Hjalmarsson, was conceived to deal with LTI systems & control.

This thesis investigated the extension of the IFT framework to include LPV systems/control. This would extent the applicability of the framework from merely LTI control to LPV control (of which LTI control can be considered a subset). In the literature it has already been shown that to integrate LPV control into the IFT framework through an LPV state space model will lead to the curse of dimensionality. Therefore an I/O approach was investigated in this thesis. Two modelling approaches were used in this thesis namely; an LPV-ARX model structure and a lifted representation of the LPV-ARX model.

When an LPV-ARX model structure is used the integration of LPV control into the IFT framework is very intuitive. This extension is made possible through the use of an extended regressor. It will be shown that the algorithm needs the scheduling sequence to be the same in both experiments. Fortunately by augmenting the gradient experiment, through the introduction of extra signals, the algorithm is capable of compensating changes in the scheduling sequence. Unfortunately this capability comes at the cost of needing a model of the plant. The advantage of this algorithm is that a single gradient experiment is needed for any number of controller parameters for an arbitrarily scheduled LPV system. This result will be shown to apply to LPV-MIMO systems for both degrees of controller freedom.

The second model structure which was investigated is the lifted representation of the LPV-ARX model. For the lifted representation of the LPV-ARX model structure the key challenge

is the dimensionality of the factorization matrix. The factorization matrix is needed to compensate for a change in the scheduling sequence. This approach has the advantage that one doesn't need a model of the plant, but it does suffer from the curse of dimensionality. To mitigate the curse the LPV system can be approximated as a LPV-FIR filter which will reduce the curse of dimensionality from exponential to linear. Furthermore it will be shown that when a lifted representation is used the gradient experiment needs to be performed per basis function and multiple times to ensure a unique solution. Therefore even though the lifted representation preserves the model free nature of linear IFT, it does suffer from a significant increase in the amount of experiments.

To numerically verify the algorithms a case study, based on a numerical model of a smart aerofoil, was performed. Performance results for this LPV system were obtained for a set of LTI controllers and LPV controllers optimized by linear IFT and the novel algorithms, respectively. The linear IFT algorithm was used to tune a set of LTI controllers for a frozen scheduling variable. The set of optimized LTI controllers were gain scheduled afterwards. A comparison of the LPV and gain scheduled LTI controllers' performance showed that they were only comparable for a small portion of the wind speeds. Although this was to be expected when observing the optimal LTI controllers' parameters evolution for increasing wind speeds. It was observed that the novel algorithms are capable of optimizing LPV controllers for realistic LPV systems as convergence of the performance was observed. However as the structure of the LPV controller is suboptimal the data has been inconclusive whether the algorithms would be capable of reaching comparable performance for the entire scheduling space if the structural bias would be removed.

Thus both algorithms are capable of optimizing LPV controller for realistic LPV systems and extend the applicability of the IFT framework to LPV systems. The best prospects are given to the extended regressor-based algorithm as acquiring a LPV model of the plant is preferable to the curse of dimensionality. However without the ability to optimize the underlying structure of the LPV controller it might still be outperformed by gain scheduled LTI controllers in terms of optimal performance.

Table of Contents

Chapter 1

Introduction

Nowadays controllers are mostly synthesized using model-based controller synthesis methods. The model-based approach uses, as the name implies, a model to synthesize the control law. Acquiring this model can be done using either system identification techniques or first principles modelling. Subsequently the controller can be synthesized through use of the identified model. The large body of literature for Linear Time Invariant (LTI) systems, or systems that can be approximated as such, show that model-based controller synthesis can lead to adequate performance in a significant number of applications.

The downside is that one must identify an accurate model through the use of a finite set of I/O data. Afterwards the identified model can be used to synthesize the control law to achieve adequate closed loop performance. The identification & controller synthesis problem must therefore be solved successively. However another class of synthesis methods exists which solves the controller synthesis problem directly. These *direct* data-driven controller synthesis approaches use the I/O data directly to synthesize the controller. This is in contrast to the model-based approach which takes the detour of first identifying a model. A natural question would then be; are there classes of systems in which either acquiring an accurate model and/or model-based controller synthesis is challenging? These kind of systems can be interesting to examine from a *direct* data-driven perspective as it condenses the identification and controller synthesis sub-problems into a direct synthesis problem. Solving the direct synthesis problem might be simpler than solving the two sub-problems separately. The class of Linear Parameter Varying systems is a class of systems which might benefit from this approach.

1-1 Linear Parameter Varying Systems

Linear Parameter Varying (LPV) systems are a special class of nonlinear systems. This class of systems can be defined as a set of gain scheduled linear models (two common representations of LPV systems are discussed in Appendix A). The temporal variation of the system dynamics are quantified through the scheduling variables which can be either endogenous or exogenous. Examples of LPV systems include pick & place manipulators and wafer stages which both

show position dependent dynamics [1]. The position dependency of their dynamics implies that they can be modelled as LPV systems with the position as a scheduling variable. Other application areas of LPV systems are flight control, magnetic bearings and many more e.g. [2][3].
An interesting feature of LPV systems is that when the scheduling variable is frozen, i.e. the scheduling variable stays constant in time, value the LPV system reduces to a LTI system. Although LPV systems are still a simplification of the true nonlinear problem it allows for a significant increase in modelling accuracy when compared to LTI systems. This increase in accuracy makes it possible to compensate nonlinear properties of the system dynamics which opens up new possibilities for control engineers. Thereby paving the way to higher performance than approximating the system as an LTI system and/or LTI control can achieve.

Modelling LPV systems by hand is, unfortunately, still a challenging problem [4]. Luckily the field of LPV system identification has developed a number of successful techniques. For an overview of the different LPV system identification techniques the interested reader is referred to [5][6][7][8]. In this section the class of LPV systems was discussed and references were given to LPV system identification literature. At the onset of this chapter two problems were introduced namely; the identification and controller synthesis problems. The first problem (identification) has been discussed briefly, but the LPV controller synthesis problem is still left untouched.

1-2 Linear Parameter Varying Controller Synthesis

Directly synthesizing a control law from an identified LPV plant is more difficult than for LTI plants. Mainly due to the significant increase of the complexity of the LPV plant which is introduced due to its gain scheduled nature. An instance of unwanted, but significant side-effect, is the notion that two individually stable LTI systems can become unstable when they are gain scheduled [9]. Even though these challenges for the LPV system class exist two main branches for synthesizing LPV controllers are present in the literature e.g. [2][3][4];

- Conventional gain scheduling: This method is a general approach for nonlinear control problems. It consists of synthesizing LTI controllers for a set of operating points. The controllers at these operating points are then interpolated in such a way that (hopefully) stability and performance are preserved over the entire state space.
- Model-based Optimization: This approach formulates the LPV controller synthesis problem as an optimization problem based on an identified LPV model of the plant. In the literature it is often (re-)formulated as a Linear Matrix Inequality (LMI) [3].

In the literature there are examples of LPV controllers increasing the performance of LPV systems when compared to LTI control. For instance in [4] the application of an LPV controller to a wafer stage was shown to improve the closed loop performance. Even though this success was reported the same paper also discussed the difficulties present in current LPV controller synthesis methods, namely:

- LPV modelling is still difficult.

- Model-Based Optimization: LMIs show numerical problems for higher order systems.

The main challenges present for LPV controller synthesis are therefore both related to the model-based approach (i.e. the LMIs and modelling). As discussed before another type of controller synthesis methods uses a more direct approach. The question which remains is therefore; what kind of *direct* data-driven approach exist that can be used?

1-3 The Promise of Iterative Feedback Tuning and Linear Parameter Varying Systems

Iterative Feedback Tuning (IFT) is a direct data-driven algorithm which tunes the controller parameters iteratively [10]. IFT uses a model-free framework in which I/O data, acquired in dedicated experiments, is used to estimate the gradient of a desired performance measure. The controller parameters are updated through the use of the aforementioned gradient estimate. In Chapter 2 the IFT framework for LTI systems will be discussed in-depth. Currently there are two interesting results for IFT w.r.t. nonlinear systems namely; the ability of linear IFT to handle (slowly varying) nonlinear systems and the, successful, integration of LPV state space control into the IFT framework [11] (referred to as IFT-LPV). Unfortunately IFT-LPV suffers from the curse of dimensionality therefore the problem of integrating LPV control into the IFT framework is still an open issue.

The two primary bottlenecks, as discussed in the previous section, in model-based LPV controller synthesis can be solved through the use of the IFT framework. As IFT is a model free approach LPV modelling is not necessary. Due to it being an optimization approach based on the Newton method and I/O data the need for solving an LMI through numerical techniques is also circumvented. This combined with the favourable prospects in the literature, i.e. IFT-LPV [11], makes IFT a viable candidate for use in *direct* data-driven LPV controller synthesis.

1-4 Goal Of This Thesis

The goal of this thesis is to extend the applicability of the IFT framework from merely LTI control to LPV control. In the literature a LPV state space approach was already investigated [11]. Unfortunately it was shown to suffer from the curse of dimensionality. Therefore in this thesis an I/O approach will be investigated in hopes of circumventing the curse of dimensionality. The problem this thesis aims to tackle is therefore the development of a novel data-driven controller tuning algorithm based on the IFT framework to be able to iteratively optimize an LPV controller. Usage of the IFT framework seems to be a viable choice for LPV controller synthesis mainly due to two aspects namely;

- Current bottlenecks present in model-based approaches to LPV controller synthesis are circumvented (as discussed in the previous section).
- The IFT framework has already shown potential for LPV controller synthesis [11].

Algorithm	**LTI**	**Regulated μ**	**Arbitrary μ**
Linear IFT [10]	Yes	Yes*	Yes*
IFT-LPV [11]	Yes	Yes^C	Yes^C
ILPVFT (Chapter 3)	Yes	Yes	Yes^m
A2S-ILPVFT (Chapter 4)	Yes	Yes	Yes^C

Table 1-1: Predicted applicability of the algorithms for different types of systems. Meaning of symbols; μ: scheduling variables, *=only for slowly varying μ, ^{C}= works but suffers from the curse of dimensionality, ^{m} = needs a model (Section 3-5).

The aforementioned reasons make the IFT framework a viable candidate for usage as a data-driven LPV controller synthesis method. The two methods present in the literature, i.e. IFT and IFT-LPV, both suffer from their own drawbacks, namely; only slowly varying scheduling variables (IFT) and the curse of dimensionality (IFT-LPV). Thus it is still an open problem in the literature to integrate LPV control into the IFT framework. Finding a solution to this problem and, numerically, verifying that it works as desired will be the main subject of this thesis. It can formally be defined as:

Extending the applicability of the IFT framework to LPV control whilst being able to handle arbitrary scheduling sequences and without suffering from the curse of dimensionality.

In Table 1-1 the algorithms and their predicted applicability for different types of systems are shown. The three classes of systems which are considered in this thesis are LTI systems, LPV systems with regulated and arbitrary scheduling sequences.

The problem will be decomposed in two parts, namely; a part consisting of mathematical proofs and afterwards the numerical verification of the developed algorithms.

1-5 Structure Of The Thesis

This thesis is concerned with developing the mathematical foundations as well as providing numerical verification for two novel LPV controller tuning algorithms. The first three chapters, i.e. Chapters 2, 3 and 4, are concerned with developing the mathematical foundations. In Chapter 5 a numerical case study concerning the optimization of a LPV controller for a realistic LPV system will be discussed and evaluated.

Chapter two discusses the IFT framework for LTI systems, as conceived by Hjalmarsson. It will re-formulate the controller synthesis problem as an optimization problem. Subsequently the optimization problem will be solved through the use of an estimate of the gradient. The estimation of the gradient, using only I/O data, will be derived using a lifted representation of the LTI system. The estimated gradient can then be used to update the controller parameters through use of the Newton method. Other properties which are discussed include; the drawbacks of the IFT framework and its performance for general nonlinear systems.

Chapter three starts out from the assumption that the scheduling sequence can be regulated. This is a simplification of the physical reality of most LPV systems. Through the use of

the "regulated scheduling"-assumption LPV control can be integrated intuitively into the IFT framework. A key component for this integration will be the extended regressor (which is also used in LPV system identification [8]). A numerical verification of the algorithm will be provided through the use of a simple case study. Afterwards the algorithm will be augmented to be able to not need the "regulated scheduling"-assumption. This augmentation will come at the cost of needing a model of the LPV plant.

Chapter four integrates LPV control into the IFT framework by modelling the LPV plant as a lifted LPV-ARX model. It will be shown that a factorization matrix is necessary to compensate changes in the scheduling sequence. The factorization matrix must be able to divide a Toeplitz matrix into a scheduling variable matrix and a matrix filled with scalars. It will be shown that its dimensionality is the main problem when using the LPV-ARX model structure. The chapter is finalized with a case study which verifies the algorithm through numerical simulation.

Chapters three and four provided the mathematical foundations and verified them numerically through a simple LPV system. However this system was simplistic and not very challenging from a control perspective. The fifth chapter will discuss a much more challenging case namely the control of a smart airfoil [12]. This LPV system has the challenging feature of becoming unstable when the scheduling variable, in this case the wind speed, exceeds a certain threshold. A second challenge is to evaluate whether the algorithms are capable of optimizing a LPV controller in such a way that comparable, to a set of gain-scheduled LTI controllers, performance is achieved.

Chapter six will provide conclusions with regards to both the mathematical foundations and the numerical results and will also discuss what points might be interesting for future work.

Chapter 2

Linear Iterative Feedback Tuning

Data-driven control methods have the advantage to be able to adapt their behaviour and controller parameters online. This can, for instance, be beneficial when compensating controller gains for a change in the system dynamics due to wear of the components over time. The optimization capabilities, inherent to most data-driven control methods, make it possible to start from an initial, non-optimal, controller and tune it to become an optimal controller. Thereby reducing time spent on, manual, tuning as it can be done in an automated fashion on the plant itself. Multiple methods for the online optimization of the controller parameters exist in the literature [13]. One of these methods is Iterative Feedback Tuning (IFT) first proposed by Hjalmarsson [10]. The IFT framework is able to optimize a controller using solely I/O data. It achieves this feat by using two dedicated experiments to estimate the gradient. IFT for the optimization of LTI controllers is the primary subject of this chapter to provide the reader with an overview of the original (LTI) framework.

Section 2-1 re-formulates the controller synthesis problem as an optimization problem. Optimization problems can be solved efficiently through gradient information. IFT estimates the gradient by performing a dedicated experiment. In Section 2-2 the gradient will be derived from a lifted representation of the LTI system. The estimated gradient can be used to obtain iterative improvements of the cost function through the use of the Newton method (Section 2-3). In Section 2-4 the drawbacks, which can have significant effects on the closed loop performance, of the IFT are reviewed. As the goal of this thesis is extending the IFT framework to LPV systems it will be interesting to examine how well linear IFT performs for general nonlinear systems (Section 2-5).

2-1 Formulating the Optimal LTI Control Problem

For optimization algorithms, like IFT, to correctly function the control problem must be reformulated as an optimization problem. The optimization problem the IFT framework must solve can mathematically be defined as (with θ the controller parameters and $J(\theta)$ a cost function)

$$\arg\min_{\theta} J(\theta) \tag{2-1a}$$

An example of a, commonly used, cost function for IFT is the tracking cost function defined as (for a SISO system):

$$\min_\theta \ J(\theta) = \min_\theta \ \frac{1}{N}\sum_{i=1}^{N}(y_i(\theta) - r_i)^2 \tag{2-1b}$$

In Equation 2-1b N stands for the amount of measurements. This can be generalized to the cost function below with T_d the desired closed loop response

$$\min_\theta \ J(\theta) = \min_\theta \ \frac{1}{N}\sum_{i=1}^{N}(y_i(\theta) - T_d r_i)^2 \tag{2-1c}$$

The controller synthesis problem is now properly formulated as an optimization problem. But how can one solve the optimization problem? A natural, but very inefficient, approach would be to check all possible sets of controller parameters. IFT solves the optimal control problem very efficiently, and elegantly, by using an iterative optimization procedure. This feat is accomplished through the use of an estimate of the gradient of the cost function of Equation 2-1c. How to estimate this gradient using solely I/O data will be discussed next.

2-2 Gradient Estimation: A Lifted Approach

To efficiently solve the controller optimization problem one can use the gradient for updating the controller parameters. This section will derive an estimate of the gradient by using a lifted representation of the LTI system. In Hjalmarsson's original work, see for example [10], transfer functions were used. The modelling paradigm which will be used is an LTI ARX model as defined below

$$y_k = \sum_{i=1}^{n_a} a_i y_{k-i} + \sum_{j=0}^{n_b} b_j u_{k-j} \tag{2-2a}$$

The output of the ARX model can be put into the following lifted representation

$$Y = \mathbf{A}Y + \mathbf{B}U \tag{2-2b}$$

With the matrix $\mathbf{B}$ defined as follows (derived from the ARX model in Equation 2-2a)

$$\mathbf{B} = \begin{bmatrix} b_0 & 0 & \dots & 0 & 0 & 0 & 0 \\ b_1 & b_0 & \dots & 0 & 0 & 0 & 0 \\ \vdots & \ddots & \ddots & \vdots & \vdots & \vdots & \vdots \\ b_{nb} & b_{nb-1} & \ddots & b_0 & 0 & \dots & 0 \\ 0 & b_{nb} & \ddots & b_1 & b_0 & \dots & 0 \\ \vdots & \vdots & \ddots & \ddots & \ddots & \ddots & \vdots \\ 0 & 0 & \dots & b_{nb} & \dots & b_1 & b_0 \end{bmatrix} \tag{2-2c}$$

The matrix $\mathbf{A}$ is defined similarly but with $a_0 = 0$. The loop will be closed by using the following control law

$$U = \mathbf{H}(R - Y) \tag{2-2d}$$

The matrix $\mathbf{H}$ can be defined similar to that of matrix $\mathbf{B}$ in Equation 2-2c. This leads to the following lifted closed loop representation

$$Y = (\mathbf{A} - \mathbf{BH})Y + \mathbf{BH}R \tag{2-2e}$$

As the gradient with respect to the i'th controller parameter (θ_i) must be uncovered it is important to know which matrices depend on the controller parameters. Thus let's explicitly add this dependency (with Θ a vector containing all the controller parameters)

$$Y(\Theta) = (\mathbf{A} - \mathbf{BH}(\Theta))\, Y(\Theta) + \mathbf{BH}(\Theta)R \tag{2-2f}$$

This experiment, referred to as the reference experiment, is visualized in Figure 2-1. The terms, in Figure 2-1, $\sum_{i=1}^{n_a} a_i q^{-i}$, $\sum_{j=0}^{n_b} b_j q^{-i}$ are condensed to $B(q)$ and $A(q)$ (Equation 2-2a for one time step) and $\sum_{i=0}^{n_h} h_i q^{-i}$ to $H(q)$. After performing the experiment of Figure 2-1 the I/O data will contain R and Y which will be needed in the second experiment.

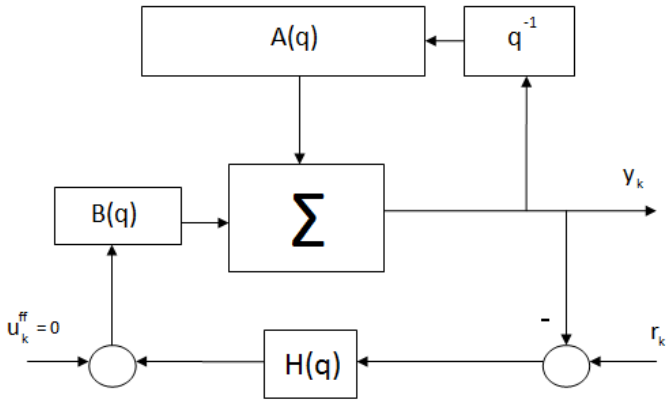

Figure 2-1: The Reference Experiment using an ARX model representation.

The derivative of Equation 2-2f for an arbitrary controller parameter θ_i is equal to

$$\frac{\partial Y(\Theta)}{\partial \theta_i} = (\mathbf{A} - \mathbf{BH}(\Theta))\, \frac{\partial Y(\Theta)}{\partial \theta_i} + \mathbf{B}\frac{\partial \mathbf{H}(\Theta)}{\partial \theta_i}(R - Y(\Theta)) \tag{2-2g}$$

From Equation 2-2g it can be observed that one must use $\frac{\partial \mathbf{H}(\Theta)}{\partial \theta_i}(R - Y)$ as a feedforward input for the second experiment. The second experiment, which will be referred to as the gradient experiment, is visualized in the block diagram shown in Figure 2-2 with $y_k^g = \frac{\partial y}{\partial \theta_i}$ and $u_k^{ff} = \frac{\partial H(\Theta)}{\partial \theta_i}(r_k - y_k)$.

Note that the block diagram (Figure 2-2) is merely an experiment with a feedforward input $\frac{\partial H(\Theta)}{\partial \theta_i}(r_k - y_k)$, which is known from the reference experiment, with the reference r_k^g set to zero. By performing the experiment of Figure 2-2 one acquires the term $\frac{\partial Y(\Theta)}{\partial \theta_i}$ which is related to the gradient of the cost function (i.e. Equation 2-1c). As the gradient, in lifted form, of Equation 2-1c is equal to (with Y^d the desired output)

$$\frac{\partial J(\theta)}{\partial \theta_i} = \frac{1}{N}\left(Y(\theta) - Y^d\right)^T \frac{\partial Y(\Theta)}{\partial \theta_i} \tag{2-2h}$$

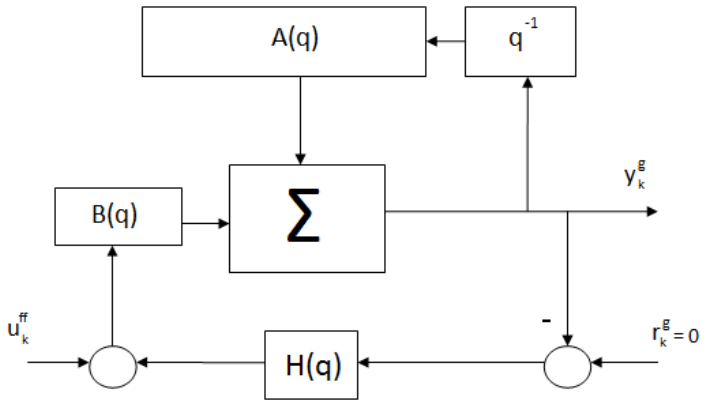

Figure 2-2: Gradient Experiment using the ARX model representation.

All the necessary I/O data seen in Equation 2-2h can be obtained by performing the reference and gradient experiments (which give $Y(\theta)$ and $\frac{\partial Y(\Theta)}{\partial \theta_i}$, respectively). Thereby making it possible to estimate the gradient of the cost function of Equation 2-1c through the usage of the I/O data of both experiments.

Equation 2-2g will now be used to show that only one, gradient, experiment needs to be performed to find the gradients with respect to *every* controller parameter. First rewrite Equation 2-2g to the format below (with $\mathbf{I}$ an identity matrix of appropriate dimension)

$$\frac{\partial Y(\Theta)}{\partial \theta_i} = (\mathbf{I} - \mathbf{A} + \mathbf{BH}(\Theta))^{-1} \mathbf{B} \frac{\partial \mathbf{H}(\Theta)}{\partial \theta_i} (R - Y(\Theta)) \tag{2-2i}$$

The equation above can be simplified by noting that the controller derivative matrix $\frac{\partial \mathbf{H}(\Theta)}{\partial \theta_i}$ is a (scalar) transfer function (F_i) times an identity matrix ($\mathbf{I}$)

$$\frac{\partial Y(\Theta)}{\partial \theta_i} = (\mathbf{I} - \mathbf{A} + \mathbf{BH}(\Theta))^{-1} F_i \mathbf{IB} (R - Y(\Theta)) \tag{2-2j}$$

$$\frac{\partial Y(\Theta)}{\partial \theta_i} = F_i \, (\mathbf{I} - \mathbf{A} + \mathbf{BH}(\Theta))^{-1} \mathbf{B} (R - Y(\Theta)) \tag{2-2k}$$

Equation 2-2k shows that filtering the data of a single gradient experiment through F_i is enough to estimate the gradient of θ_i. Therefore only one gradient experiment has to be performed to uncover the gradient for all controller parameters.

The results of this section have provided a means to estimate the gradient based solely on I/O data of two dedicated experiments. So the question becomes; how can the estimated gradient be used to improve the closed loop performance?

2-3 Controller Parameter Update

The controller parameters should be adjusted in such a way that the cost function improves. If every set of control parameters would have to be evaluated this would take a considerable

time. As not all sets of controller parameters will lead to an improvement of the cost function. Luckily an experimental method for estimating the gradient has been derived in the previous section. The gradient points in the direction of the largest increase of the cost function, therefore "moving" in the opposite direction will yield the largest decrease. This is the basis for the Newton update method defined as:

$$\theta_{i+1} = \theta_i - \gamma_i \frac{\partial J(\theta_i)}{\partial \theta} \tag{2-3a}$$

With θ_i & θ_{i+1} vectors containing (respectively) the current and next iteration's controller parameters, γ the step length and $\frac{\partial J(\theta_i)}{\partial \theta}$ the gradient estimate. This parameter update scheme allows for an iterative improvement of the cost function. The Newton update method can be improved by introducing a matrix H

$$\theta_{i+1} = \theta_i - \gamma_i H^{-1} \frac{\partial J(\theta_i)}{\partial \theta} \tag{2-3b}$$

The matrix H is often chosen to be an approximation of the cost function's Hessian. Examples of methods used in the Hessian approximation are; trust region, BFGS or the Gauss-Newton methods [10]. To improve convergence the Hessian can be regularized [14]. Although this iterative improvement of the performance is advantageous there are some notable drawbacks inherent to the IFT framework.

2-4 Drawbacks of The IFT Framework

Although IFT is capable of iteratively improving the closed loop performance based on data collected online it is not without its drawbacks. Some of these drawbacks can make physical implementations considerably more difficult. Three known drawbacks of the linear IFT scheme are (the need for an initial stabilizing controller is not something the author constitutes as a drawback)

- The time specification T_d must be chosen carefully to ensure that the algorithm improves w.r.t. the desired performance [10].
- The gradient-based Newton method can lead to an unstable closed loop [10]. Results in the literature indicate that this risk can be minimized by introducing a stability constraint [15].
- The Newton update method doesn't guarantee that the cost function will improve in every iteration [16].

The first one can make the formulation of the cost function challenging as it might be difficult to convert a performance measure to time domain specifications. The other two points are related as both imply a worsening of the closed loop performance. But whereas the stability problem can be solved using constraints, guarantees that the cost function decreases in every iteration can't be given.

These drawbacks can have a significant effect on the closed loop performance and should therefore be taken into account. The last property which will be discussed is the performance of linear IFT for nonlinear systems.

2-5 Nonlinear Systems

The goal of this thesis is extending the applicability of the IFT framework to LPV control. Therefore it is interesting to observe how well IFT performs in the literature for general nonlinear systems. In [17] it was shown that the IFT framework can also handle nonlinear systems. This is a very notable result as it implies that the IFT scheme can actually compensate some nonlinearities in the dynamics of the system. Even though it wasn't developed to incorporate this "robustness" to nonlinear dynamics [10].

Unfortunately there is a catch namely; IFT can only handle nonlinear plants when the dynamics don't vary too quickly [17][18]. This makes linear IFT unviable for LPV systems as the scheduling variable can vary rapidly. Take for example the large accelerations present in wafer stages which change the scheduling variable, i.e. the position, very rapidly. This makes the capability for handling arbitrarily fast scheduling sequences a must.

The capability to compensate for arbitrarily fast scheduling sequences will be an essential part of the novel algorithms. In the literature the results in [11] indicate that this capability is possible to integrate into the IFT framework. Unfortunately the extension of the IFT framework in the aforementioned paper suffers from the curse of dimensionality. Thus integrating LPV control into the IFT framework is still an open issue.

This concludes the discussion on the properties of the original IFT framework. The next chapter will integrate LPV control whilst modelling the LPV plant as an LPV-ARX system.

Chapter 3

LPV-ARX Model Structure

This chapter revolves around the integration of LPV control into the IFT framework by using a LPV-ARX model structure. The LPV-ARX model structure is a gain scheduled extension of the LTI-ARX model. This thesis will show that through usage of the extended regressor representation, which is also used in LPV system identification [8], of the LPV-ARX model structure an elegant extension of the IFT framework is possible. The extended regressor approach has been shown to be able to handle arbitrarily fast scheduling sequences and does not suffer from the curse of dimensionality when used for LPV system identification [8]. In Sections 3-1 to 3-4 it will be assumed that the scheduling sequence is regulated. Under this assumption the integration of LPV control into the framework will be intuitive. The regulated scheduling sequence assumption will then be shown to be unnecessary through an augmentation of the gradient experiment. The aim of this chapter is to provide the mathematical foundations and numerical verification of a novel data-driven controller tuning algorithm based on the extended regressor and the IFT framework.

The first section (3-1) will formulate the optimal LPV control problem which the novel algorithm should solve. Section 3-2 will introduce the extended regressor and its modelling structure. The extended regressor will be used in the construction of the mathematical foundations of the algorithm in Section 3-3. To numerically verify the algorithm the optimization of a LPV PI controller is performed for a simple LPV system (Section 3-4). The final section (3-5) will generalize the preceding results to arbitrary scheduling sequences.

3-1 Formulating the Optimal LPV Control Problem

For the optimal control law one would want to optimize the LPV controller's structure as well as the controller parameters. A general LPV controller would be of the form (with θ_i the controller parameters, N_θ controller parameters, N_f basis functions and e_k the error signal at time k):

$$\mathcal{H}(\mu_k) = \sum_{j=0}^{N_\theta} \sum_{i=1}^{N_f} \theta_j^i(\mu_k) e_{k-j} \tag{3-1a}$$

Such a LPV control law would be able to properly compensate the time varying dynamics of the LPV system. The optimization problem one wants to solve is then an LPV extension of Equation 2-1c

$$\arg\min_{\Theta} J(\mu_k, \Theta) = \min_{\theta} \frac{1}{2N} \sum_{i=1}^{N} \left(y_i(\theta, \mu_k) - T_d(\mu_k) r_i\right)^2 \tag{3-1b}$$

Note that the cost function can depend on the scheduling variables. An important question left unanswered is; how should the structure of the LPV controller's basis functions look like? This question will not be answered in this thesis and it is just assumed that one can select a proper set of basis functions. The next section will discuss the extended regressor which has also seen success in LPV I/O system identification [8].

3-2 The Extended Regressor

In the LPV system identification literature, specifically [12], the same curse of dimensionality was found as in IFT-LPV [11]. Both used a LPV state space model representation as their basis. Therefore the author conjectures that using a modelling strategy from the field of LPV I/O system identification which doesn't suffer from the curse of dimensionality can be a viable option. The modelling approach which will be investigated is the extended regressor. This modelling strategy doesn't suffer from the curse and can handle arbitrarily fast scheduling sequences [8]. The extended regressor will now be introduced to familiarize the reader with its notation and properties.

Definition: Extended Regressor - Control Law

The extended regressor representation is an extension of the LTI regressor representation. One of the definitions which is important for the extended regressor is the inner product of a matrix

$$< A, B >= \text{trace}(A^T B) \tag{3-2a}$$

Assume a control law of the following format

$$u_k = \mathcal{H}(\mu_k) e_k \tag{3-2b}$$

The controller $\mathcal{H}(\mu_k)$ is defined in Equation 3-2c. Note that the shift operator q^{-n} is modelled to only affect the error signal e_k although this might seem ambiguous in the expression below (as also discussed in Appendix A).

$$u_k(\mu_k) = \left(\theta_0(\mu_k) + \theta_1(\mu_k) q^{-1} + \theta_2(\mu_k) q^{-2} + \ldots + \theta_{n_H}(\mu_k) q^{-n_H}\right) e_k(\mu_k) \tag{3-2c}$$

A single term θ_i is a linear combination of the multiplication of a controller parameter θ_i^j and basis function $\psi_j(\mu_k)$ (for instance $\theta_i = \theta_i^1 \psi_1(\mu_k) + \theta_i^2 \psi_2(\mu_k)$). Define the matrix containing all controller parameters Θ as

$$\Theta = \begin{bmatrix} \theta_1^1 & \ldots & \theta_1^{N_f} \\ \theta_2^1 & \ldots & \theta_2^{N_f} \\ \vdots & \ldots & \vdots \\ \theta_n^1 & \ldots & \theta_n^{N_f} \end{bmatrix} \tag{3-2d}$$

The second matrix Φ_k is defined in Equation 3-3e. The shift operators can be replaced with any causal filter (based on shift operators) the only constraint is that one must be able to write the input u_k as $u_k =< \Theta, \Phi_k > e_k$.

$$\Phi_k = \begin{bmatrix} 1 \\ q^{-1} \\ \vdots \\ q^{-n} \end{bmatrix} \begin{bmatrix} \psi_1(\mu_k) & \psi_2(\mu_k) & \dots & \psi_{N_f}(\mu_k) \end{bmatrix} \tag{3-2e}$$

The subscript of the basis function $\psi_n(\mu_k)$ signifies its column in the row vector. Even though all the matrices Φ_k are functions of the scheduling parameter this will not be indicated for clarity.

The incorporation of multiple scheduling variables can be achieved by adding basis functions in the row vector dependent on another scheduling variable. The extended regressor will now be explained using an example to understand how the recasting can be done.

Example: LPV-PI Controller

The extended regressor will now be used to re-formulate a LTI-PI to a LPV controller to help visualize the approach. First the transformation of a LTI-PI controller into a regressor format will be discussed. Subsequently it will be shown how to introduce LPV elements. Define the LTI PI control law as (with e_k the error signal $(r_k - y_k)$)

$$u_k = \left(K_p + K_i \frac{T_s q^{-1}}{1 - q^{-1}} \right) e_k \tag{3-3a}$$

This can be written in a regressor representation as

$$\Phi_k = \begin{bmatrix} 1 \\ \frac{T_s q^{-1}}{1-q^{-1}} \end{bmatrix} \begin{bmatrix} 1 \end{bmatrix} \tag{3-3b}$$

$$\Theta = \begin{bmatrix} K_p \\ K_i \end{bmatrix} \tag{3-3c}$$

By extending the Θ-matrix of Equation 3-3c and Φ_k's row vector LPV elements can be introduced

$$\Phi_k = \begin{bmatrix} 1 \\ \frac{T_s q^{-1}}{1-q^{-1}} \end{bmatrix} \begin{bmatrix} 1 & \psi_1(\mu_k) & \psi_2(\mu_k) & \dots & \psi_{N_f}(\mu_k) \end{bmatrix} \tag{3-3d}$$

$$\Theta = \begin{bmatrix} K_p & \theta_1^2 & \dots & \theta_1^{N_f} \\ K_i & \theta_2^2 & \dots & \theta_2^{N_f} \end{bmatrix} \tag{3-3e}$$

Thus by using the regressor representation the original PI controller can easily be extended to encompass LPV behaviour.

An important note is that if a fixed structure controller structure is used wherein not all basis functions are used by all rows of the Φ_k matrix (for example $K_p + \frac{T_s q^{-1}}{1-q^{-1}}(K_i + \theta_2^2 \mu_k))e_k$) then the extend regressor can be defined as a superposition of multiple Φ_k matrices and the matrix Θ;

$$\Phi_k = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \begin{bmatrix} 1 & 0 \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{T_s q^{-1}}{1-q^{-1}} \end{bmatrix} \begin{bmatrix} 1 & \mu_k \end{bmatrix} \tag{3-3f}$$

$$\Theta = \begin{bmatrix} K_p & 0 \\ K_i & \theta_2^2 \end{bmatrix} \tag{3-3g}$$

The extended regressor representation therefore allows for an intuitive transformation from a LTI to a LPV controller.

End of Definition

For LPV system identification the extended regressor approach is able to fulfil the goals which were formulated in Section 1-4. This leads to the following conjecture this chapter hopes to validate:

By reformulating the optimization problem as an extended regressor problem the ensuing matrix representation can be used within the IFT framework. The resulting Newton update law can then be verified to work without suffering from the curse of dimensionality.

Mathematically this means solving the question whether we can transform the triplet of an LPV controller, gradient experiment and a regulated LPV system into a Newton update law of the form

$$\Theta_{j+1} = \Theta_j - \gamma_i f(\Phi_k, ?) \tag{3-4}$$

Observe that the function f is an, as of yet unknown, but conjectured to be a function of the regressor matrix Φ_k and possibly other variables. The algorithm developed in this chapter will be dubbed *ILPVFT*. In the next section the mathematical foundations will be developed for the novel algorithm.

3-3 The Mathematical Foundations

This section details the proofs related to the extended regressor approach to IFT for a SISO LPV system. The derivation starts from an open loop representation of the LPV plant. Afterwards the controller and its parameterization will be defined. This controller will be used to close the loop. Subsequently the closed loop system will be used to fully define the cost function (whose gradient will be derived afterwards). Extensions of this result to MIMO systems and feedforward control are discussed in Appendix B-2 and B-3. In Section 3-5 the results will be generalized to arbitrary scheduling sequences.

Open Loop Representation

The LPV system will in open loop be defined as

$$y_k(\mu_k) = \frac{B(q, \mu_k)}{A(q, \mu_k)} u_k = P(q, \mu_k) u_k \tag{3-5a}$$

With q the shift operator and μ_k the scheduling variable at time k. Define A and B as follows

$$A(q, \mu_k) = 1 + a_1(\mu_k) q^{-1} + a_2(\mu_k) q^{-2} + \ldots + a_{n_a}(\mu_k) q^{-n_a} \tag{3-5b}$$

$$B(q, \mu_k) = b_0(\mu_k) + b_1(\mu_k) q^{-1} + b_2(\mu_k) q^{-2} + \ldots + b_{n_b}(\mu_k) q^{-n_b} \tag{3-5c}$$

The factors n_b and n_a are equal to the amount of shift operators present in A and B, respectively. Before closing the loop the controller must be defined.

Controller Definition

The controller parameterization will play a significant role in deriving the gradient. The controller parameter matrix Θ^{fb} has the superscript fb as it pertains to the feedback controller. The input will be defined as follows with $\mathcal{H}$ the LPV controller

$$u_k = \mathcal{H}(\mu_k, \Theta^{fb})(r_k - y_k) = \mathcal{H}(\mu_k, \Theta^{fb})e_k \tag{3-6a}$$

The controller $\mathcal{H}(\mu_k, \Theta^{fb})$ will be defined as

$$\mathcal{H}(\mu_k, \Theta^{fb}) = (\theta_0(\mu_k) + \theta_1(\mu_k)q^{-1} + \theta_2(\mu_k)q^{-2} + \ldots + \theta_{n_H}(\mu_k)q^{-n_H})e_k \tag{3-6b}$$

Or in regressor representation

$$u_k = \mathcal{H}(\mu_k, \Theta^{fb})e_k =< \Theta^{fb}, \Phi_k > e_k \tag{3-6c}$$

In the final subsection of this section a generalization will be shown from this LPV-FIR controller to general LPV controllers. With the controller defined the loop can now be closed.

Closing The Loop

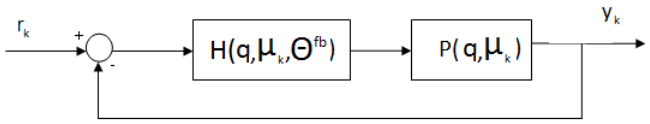

Figure 3-1: Block diagram for the closed loop SISO LPV system.

A visualisation of the closed loop LPV system is shown in Figure 3-1. The closed loop system representation can be computed as follows

$$y_k(\mu_k) = \frac{B(q,\mu_k)}{A(q,\mu_k)}u_k \tag{3-7a}$$

$$u_k = \mathcal{H}(\mu_k, \Theta^{fb})(r_k - y_k) \tag{3-7b}$$

$$y_k(\mu_k) = \frac{B(q,\mu_k)}{A(q,\mu_k)}\mathcal{H}(\mu_k, \Theta^{fb})(r_k - y_k) \tag{3-7c}$$

$$\left(1 + \frac{B(q,\mu_k)\mathcal{H}(\mu_k, \Theta^{fb})}{A(q,\mu_k)}\right) y_k = \frac{B(q,\mu_k)\mathcal{H}(\mu_k, \Theta^{fb})}{A(q,\mu_k)}r_k \tag{3-7d}$$

$$\frac{A(q,\mu_k) + B(q,\mu_k)\mathcal{H}(\mu_k, \Theta^{fb})}{A(q,\mu_k)}y_k = \frac{B(q,\mu_k)\mathcal{H}(\mu_k, \Theta^{fb})}{A(q,\mu_k)}r_k \tag{3-7e}$$

This leads to the closed loop system

$$y_k(\mu_k) = \frac{B(q,\mu_k)\mathcal{H}(\mu_k, \Theta^{fb})}{A(q,\mu_k) + B(q,\mu_k)\mathcal{H}(\mu_k, \Theta^{fb})}r_k \tag{3-7f}$$

With the closed loop representation now known the cost function can be fully defined.

IFT Framework: Cost Function Analysis

In the previous subsection the closed loop system (Equation 3-7f) was found. This closed loop system shows dependency of the controller parameters on the previous output(s), the reference and (more problematically) on the scheduling variables. As shown in Equation 3-1b the general cost function is equal to

$$J(\Theta^{fb}) = \frac{1}{2N}\sum_{i=1}^{N}\Big(y_i(\Theta^{fb},\mu_k) - T_d(\mu_k)r_i\Big)^2 \tag{3-8a}$$

The gradient of this cost function is equal to the following expression

$$\frac{\partial J(\Theta^{fb})}{\partial \Theta^{fb}} = \frac{1}{N}\sum_{i=1}^{N}(y_i(\Theta^{fb},\mu_i) - T_d(\mu_k)r_i)\frac{\partial y_i(\Theta^{fb},\mu_i)}{\partial \Theta^{fb}} \tag{3-8b}$$

This gradient will now be evaluated using the closed loop system of Equation 3-7f.

IFT Framework: Gradient Estimation

The most important information the ILPVFT algorithm must find using the I/O data is an estimate of the gradient of the cost function (Equation 3-8b). The gradient consists of data known from the reference experiment (y_k,r_k) and the desired closed loop response ($T_d(\mu_k)$). Thus the only term left to compute is $\frac{\partial y_k(\Theta^{fb},\mu_k)}{\partial \Theta^{fb}}$. By using Equation 3-7f it can be shown to be equal to

$$\frac{\partial y_k(\Theta^{fb},\mu_k)}{\partial \Theta^{fb}} = \frac{\partial}{\partial \Theta^{fb}}\left(\frac{B(q,\mu_k)\mathcal{H}(\mu_k,\Theta^{fb})}{A(q,\mu_k)+B(q,\mu_k)\mathcal{H}(\mu_k,\Theta^{fb})}\right)r_k \tag{3-9a}$$

The result of this differentiation is equal to (for the proof see Appendix B-1)

$$\frac{\partial y_k(\Theta^{fb},\mu_k)}{\partial \Theta^{fb}} = \frac{\partial \mathcal{H}(\mu_k,\Theta^{fb})}{\partial \Theta^{fb}}\frac{B(q,\mu_k)}{A(q,\mu_k)+B(q,\mu_k)\mathcal{H}(\mu_k,\Theta^{fb})}(r_k - y_k(\mu_k)) \tag{3-9b}$$

This is the same result as in the LTI case for IFT. However there is one property not present in the LTI case. Namely the dependence of the output on the scheduling variables. This leads to the following important observation;
For ILPVFT the scheduling variable should be the same in the reference and gradient experiment for every time index.

The final part for estimating the gradient is finding an expression for the partial derivatives of the controller itself. To compute the derivative of the controller the extended regressor will be used. First rewrite the previously obtained equation by setting the output of the gradient equal to (the superscript g indicates the gradient experiment)

$$y_k^g(\Theta^{fb},\mu_k) = \frac{B(q,\mu_k)}{A(q,\mu_k)+B(q,\mu_k)\mathcal{H}(\mu_k,\Theta^{fb})}(r_k - y_k(\mu_k)) \tag{3-9c}$$

This will lead to the derivative, Equation 3-9b, being rewritten in the more compact way shown below

$$\frac{\partial y_k(\Theta^{fb},\mu_k)}{\partial \Theta^{fb}} = \frac{\partial \mathcal{H}(\mu_k,\Theta^{fb})}{\partial \Theta^{fb}}y_k^g(\Theta^{fb},\mu_k) \tag{3-9d}$$

As observed previously this can be rewritten to an extended regressor representation. With Φ_k a matrix consisting of shift operators and basis functions (Section 3-2)

$$\frac{\partial y_k(\Theta^{fb},\mu_k)}{\partial \Theta^{fb}} = \frac{\partial \text{trace}(\Theta^{fb},\Phi_k)}{\partial \Theta^{fb}} y_k^g(\Theta^{fb},\mu_k) \tag{3-9e}$$

By using the property $\frac{\partial\ \text{trace}(X^T B)}{\partial X} = B$

$$\frac{\partial y_k(\Theta^{fb},\mu_k)}{\partial \Theta^{fb}} = \Phi_k y_k^g(\Theta^{fb},\mu_k) \tag{3-9f}$$

To round up this subsection it can now be seen that the gradient of the cost function is equal to (for an arbitrary time step k)

$$\frac{\partial J(\Theta_k^{fb})}{\partial \Theta^{fb}} = \Big(y_k(\Theta^{fb},\mu_k) - T_k^d(\mu_k) r_k\Big)\Big(\Phi_k y_k^g(\Theta^{fb},\mu_k)\Big) \tag{3-9g}$$

This result is the same as for the LTI case with two key differences namely the notion that the expression is now a matrix instead of a vector and there is a dependence on the scheduling sequence.

This result shows that by using an extended regressor modelling approach no curse of dimensionality is introduced for a LPV-FIR controller. This controller type is far from the most versatile control methodology in the arsenal of the control engineer. Thus let's generalize the result to a general LPV controller.

Generalization: From LPV-FIR Filters to General LPV Controllers

The penultimate step in this proof is to generalize this section's proof for a FIR filter to a general LPV controller. Define the general LPV controller as follows

$$u_k(\mu_k) = \mathcal{H}_2(\mu_k,\Theta^{fb}) e_k(\mu_k) = \frac{G(q,\mu_k)}{1+F(q,\mu_k)} e_k(\mu_k) \tag{3-10a}$$

Define $G(q,\mu_k)$ and $F(q,\mu_k)$ as

$$G(q,\mu_k) = \theta_0^g(\mu_k) + \theta_1^g(\mu_k)q^{-1} + \theta_2^g(\mu_k)q^{-2} + \ldots + \theta_{n_g}^g(\mu_k)q^{-n_g} \tag{3-10b}$$

$$F(q,\mu_k) = \theta_1^f(\mu_k)q^{-1} + \theta_2^f(\mu_k)q^{-2} + \ldots + \theta_{n_f}^f(\mu_k)q^{-n_f} \tag{3-10c}$$

Using the same parameterization as before this can be rewritten as an extended regressor

$$u_k(\mu_k) = <\Theta_g, \Phi_k^g> e_k(\mu_k) + <\Theta_f, \Phi_k^f> u_{k-1}(\mu_{k-1}) \tag{3-10d}$$

Under the assumption that both Φ-matrices have the same basis function it can be rewritten as a stacked matrix. This means that a stacking can be performed (e.g. e_k and u_{k-1} data) as done below

$$\Phi_k^{f,g} = \begin{bmatrix} u_{k-1} \\ \vdots \\ u_{k-n_f} \\ e_k \\ e_{k-1} \\ \vdots \\ e_{k-n_g} \end{bmatrix} \begin{bmatrix} \psi_1(\mu_k) & \psi_2(\mu_k) & \ldots & \psi_{N_f}(\mu_k) \end{bmatrix} \tag{3-10e}$$

The stacking of the controller coefficients can be performed as

$$\Theta_{\mathcal{H}_2} = \begin{bmatrix} \theta_{f,1}^1 & \dots & \theta_{f,1}^{N_f} \\ \theta_{f,2}^1 & \dots & \theta_{f,2}^{N_f} \\ \vdots & \vdots & \vdots \\ \theta_{f,n_f}^1 & \dots & \theta_{f,n_f}^{N_f} \\ \theta_{g,1}^1 & \dots & \theta_{g,1}^{N_f} \\ \theta_{g,2}^1 & \dots & \theta_{g,2}^{N_f} \\ \vdots & \vdots & \vdots \\ \theta_{g,n_g}^1 & \dots & \theta_{g,n_g}^{N_f} \end{bmatrix} \tag{3-10f}$$

Thus the general LPV control problem can be rewritten in the same structure as Equation 3-9f. Although the input is now also needed, but this data is readily available from the I/O data. There is however one remaining problem with the formulation of our control objective (Equation 3-8a) namely that it doesn't penalize the control input. This drawback will now be compensated for by re-formulating the control objective.

Penalizing The Control Input

High controller inputs can be a problem due to for instance actuator saturation. This can be compensated for by penalizing high controller inputs in the control objective itself. The new cost function will be defined as (with a scalar $\lambda \geq 0$)

$$J_2(\Theta^{fb}) = \frac{1}{2N}\sum_{i=1}^{N}(y_i(\Theta^{fb},\mu_i) - y_i^d)^2 + \lambda u_i^2(\Theta^{fb},\mu_i) \tag{3-11a}$$

The gradient of this cost function is equal to

$$J_2(\Theta^{fb}) = \frac{1}{N}\sum_{i=1}^{N}(y_i(\Theta^{fb},\mu_i) - y_i^d)\frac{\partial y_i(\Theta^{fb},\mu_i)}{\partial \Theta^{fb}} + \lambda \frac{\partial u_i(\Theta^{fb},\mu_i)}{\partial \Theta^{fb}} u_k(\Theta^{fb},\mu_i) \tag{3-11b}$$

The gradient of the first term has already been discussed extensively in this chapter. Therefore let's concentrate our attention to the term $\frac{\partial u_i}{\partial \Theta^{fb}}$. The input's (Equation 3-6b) partial derivative is equal to

$$\frac{\partial u_k(\Theta^{fb},\mu_k}{\partial \Theta^{fb}} = \frac{\partial \mathcal{H}(\mu_k,\Theta^{fb})}{\partial \Theta^{fb}}(r_k - y_k(\Theta^{fb})) - \mathcal{H}(\mu_k,\Theta^{fb})\frac{\partial y_k(\Theta^{fb})}{\partial \Theta^{fb}} \tag{3-11c}$$

It can be seen that this partial derivative consists of data which is known from the two dedicated experiments. So one can penalize higher control inputs within the estimated gradient through usage of the I/O data from both experiments.

The final important property of linear IFT is the unbiased nature of the estimate of the gradient [10]. This result also holds, for output noise, in the ILPVFT algorithm as by vectorization of the Θ-matrix the expression of linear IFT returns and so the same results as in [10] can be shown to hold. The mathematical foundations regarding the ILPVFT algorithm have now been discussed. To verify the theoretical claims of Section 3-3 a numerical case study will now be performed.

3-4 Case Study

In this chapter the integration of LPV control into the IFT framework has been discussed. In the next section the mathematical foundations will be extended to general scheduling sequences. However before this generalization is discussed a numerical verification under the assumption of regulated scheduling sequences will be performed to verify that it works as intended.

Case Description

The numerical example will be a SISO LPV system, which was introduced in [19]. This SISO LPV system is defined as

$$x^g_{k+1} = \mu_k x^g_k + u_k \tag{3-12a}$$

$$y_k = x^g_k \tag{3-12b}$$

The control law will be a LPV-PI controller defined in I/O format as

$$u_k = u_{k-1} + \theta_0(\mu_k)(r_k - y_k) + \theta_1(\mu_k)(r_{k-1} - y_{k-1}) \tag{3-12c}$$

With θ_i for $i = 1, 0$ defined as

$$\theta_0(\mu_k) = \theta^0_0 + \theta^1_0 \mu_k \tag{3-12d}$$

$$\theta_1(\mu_k) = \theta^0_1 + \theta^1_1 \mu_{k-1} \tag{3-12e}$$

With the desired closed loop LPV behaviour, in state space, defined as

$$A(\mu_k) = \begin{bmatrix} -1 & 1 \\ -1 - (\mu_k - \mu_{k-1}) & 1 \end{bmatrix} \tag{3-12f}$$

$$B(\mu_k) = \begin{bmatrix} 1 + \mu_k \\ 1 + (\mu_k - \mu_{k-1}) \end{bmatrix} \tag{3-12g}$$

$$C = \begin{bmatrix} 1 & 0 \end{bmatrix} \tag{3-12h}$$

$$D = 0 \tag{3-12i}$$

Which in state space form is equal to (with the state defined as $x_k = [x^g_k,\ x^u_k]^T$, with the second state related to the input)

$$x_{k+1} = A(\mu_k)x_k + B(\mu_k)r_k \tag{3-12j}$$

$$y_k = Cx_k \tag{3-12k}$$

In [19] the following set of controller parameters was shown to attain the desired closed loop behaviour precisely

$$\theta_0(\mu_k) = \theta^0_0 + \theta^1_0 \mu_k = 1 + \mu_k \tag{3-12l}$$

$$\theta_1(\mu_k) = \theta^0_1 + \theta^1_1 \mu_{k-1} = -\mu_{k-1} \tag{3-12m}$$

Therefore verifying the algorithm is equal to stating that they should converge to these optimal values.

Deriving the Controller Parameter Gradient

For the IFT framework the partial derivatives of the controller w.r.t. the controller parameters are needed. As discussed in Section 3-2 the input (Equation 3-12c) needs to be rewritten to a form wherein $F(q)u_k =< \Theta^{fb}, \Phi_k > e_k$ as to fit the developed extension of the IFT framework. Therefore let's rewrite it using shift operators and $r_k - y_k = e_k$

$$u_k = u_{k-1} + \theta_0(\mu_k)(r_k - y_k) + \theta_1(\mu_k)(r_{k-1} - y_{k-1}) \tag{3-13a}$$

As discussed earlier the extended regressor assumes that the shift operator doesn't operate on the controller parameters (i.e. $\theta_i^j(\mu_k)y_k(\mu_k)q^{-1} = \theta_i^j(\mu_k)y_{k-1}(\mu_{k-1})$)

$$u_k = u_k q^{-1} + \theta_0(\mu_k)e_k + \theta_1(\mu_k)e_k q^{-1} \tag{3-13b}$$

$$(1 - q^{-1})u_k = (\theta_0(\mu_k) + \theta_1(\mu_k))\, e_k q^{-1} \tag{3-13c}$$

$$u_k = \frac{\theta_0(\mu_k) + \theta_1(\mu_k)q^{-1}}{1 - q^{-1}} e_k = \mathcal{H}(\mu_k)e_k \tag{3-13d}$$

The partial controller derivatives are therefore equal to

$$\frac{\partial \mathcal{H}(\mu_k)}{\partial \theta_0^0} = \frac{1}{1 - q^{-1}} \tag{3-13e}$$

$$\frac{\partial \mathcal{H}(\mu_k)}{\partial \theta_0^1} = \frac{\mu_k}{1 - q^{-1}} \tag{3-13f}$$

$$\frac{\partial \mathcal{H}(\mu_k)}{\partial \theta_1^0} = \frac{q^{-1}}{1 - q^{-1}} \tag{3-13g}$$

$$\frac{\partial \mathcal{H}(\mu_k)}{\partial \theta_1^1} = \frac{\mu_{k-1}q^{-1}}{1 - q^{-1}} \tag{3-13h}$$

The extended regressor can be written as

$$\Phi_k = \frac{\partial \mathcal{H}(\mu_k)}{\partial \Theta^{fb}} = \frac{1}{1 - q^{-1}} \begin{bmatrix} 1 \\ 0 \end{bmatrix} \begin{bmatrix} 1 & \mu_k & 0 \end{bmatrix} + \frac{q^{-1}}{1 - q^{-1}} \begin{bmatrix} 0 \\ 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & \mu_{k-1} \end{bmatrix} \tag{3-13i}$$

This concludes the necessary preliminary derivations, so it is time to examine the results acquired from numerical simulations.

Results

The algorithm used the optimization parameters shown in Table 3-1. Two termination criteria were implemented namely; a minimum gradient norm and a maximum number of iterations. The reference signal shown in Figure 3-2 was used during the simulation.

The algorithm was tested for two initial conditions for both a deterministic and stochastic version of the LPV system (output noise only). The first initial condition is equal to

$$\Theta_{0,1} = \begin{bmatrix} 0.2 & 0 & 0 \\ 0.5 & 0 & 0 \end{bmatrix} \tag{3-14}$$

Parameter	Numerical Value
Amount of measurements N	200
Learning rate γ	0.5
Output (white) noise variance	$(0.1)^2$ (stochastic only)
Scheduling noise	Not Implemented
Scheduling sequence (Dedicated)	$0.4\sin(\frac{2\pi}{N}k)$
Maximum number of iterations	3500
Minimum norm gradient	10^{-9}
Hessian	Identity Matrix

Table 3-1: Optimization Parameters for the ILPVFT algorithm.

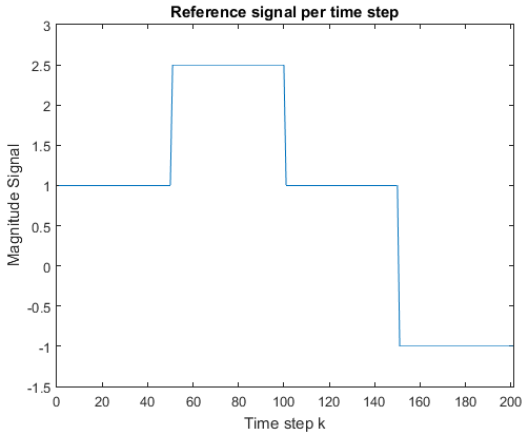

Figure 3-2: Reference signal used in the reference experiment.

Initial Condition I

The outputs of interest are the performance and the controller parameters (Figures 3-3 to 3-5). For the deterministic system, Figures 3-3 and 3-4, it can be observed that the controller parameters converge to the globally optimal parameters and the performance converges to zero. When output noise is added the performance stabilizes around 0.02 (due to noise in the reference experiment) but as seen in the controller parameters plot, Figure 3-5, it converges to the globally optimal parameters (showing oscillatory behaviour around the optimal parameters).

The second initial condition will now be discussed which is equal to

$$\Theta_{0,2} = \begin{bmatrix} 1.5 & 0.1 & 0 \\ 0.2 & 0 & -0.1 \end{bmatrix} \tag{3-15}$$

Initial Condition II

The same outputs as before are of interest and are shown in Figures 3-7 to 3-10 (γ was reduced to 0.25). It can be seen that the same properties in terms of convergence and unbiasedness of

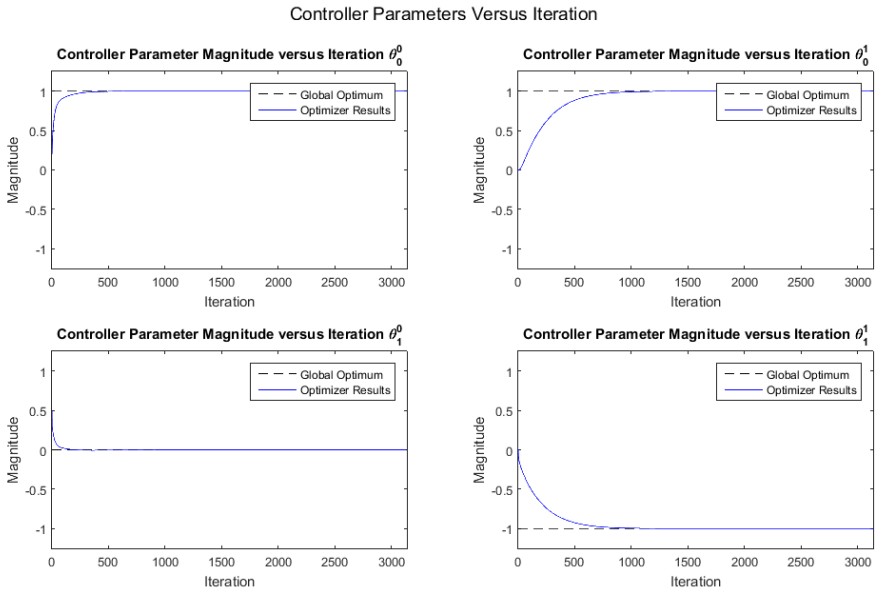

Figure 3-3: Evolution of the controller parameters per iteration when using a deterministic system.

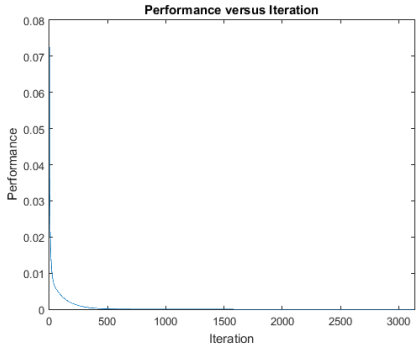

Figure 3-4: Evolution of the performance per iteration when using a deterministic system.

the gradient estimate are present. There is however one important property of the underlying Newton method which was not visible in the previous example namely; no guarantee of a decrease in the performance function. This can be seen in Figure 3-8 which shows that in the first iterations the performance jumps from approximately 0.14 to 0.2. But as seen in Figure 3-7 the controller parameters still eventually converge to the globally optimal parameters.

The numerical results verify that the algorithm works as designed. However there is one drawback present in the algorithm which once solved extends its application range tremendously namely; the need for regulated scheduling sequences. This will be done by augmenting the gradient experiment and using a model of the plant.

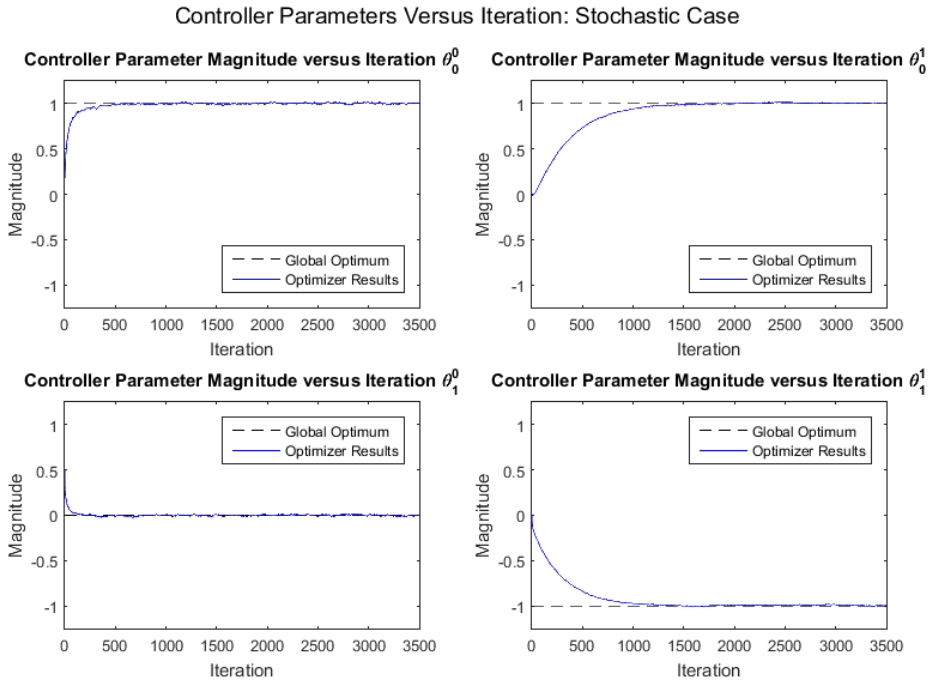

Figure 3-5: Evolution of the controller parameters per iteration when using a stochastic system.

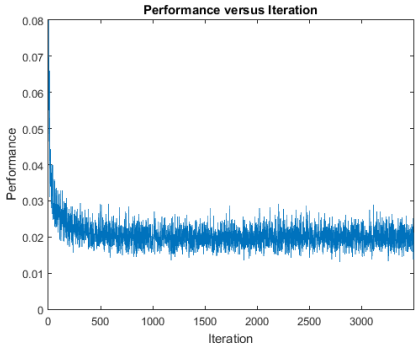

Figure 3-6: Evolution of the performance per iteration when using a stochastic system.

3-5 Model Based Scheduling Compensation

The previous section of this chapter assumed that the scheduling sequences were regulated as a simplification of the problem. In this section the results will be extended to arbitrary scheduling sequences by augmenting the gradient experiment which will come at the cost of needing a model of the plant.

The desired gradient experiment is equal to (with $g_k = r_k - y_k$ as derived earlier in this chapter)

$$y_k^r(\mu_k^r) = \sum_{i=1}^{N_p} p_i(\mu_k^r) y_{k-i}^r + \sum_{j=0}^{N_b} b_j(\mu_k^r) g_{k-j} \tag{3-16a}$$

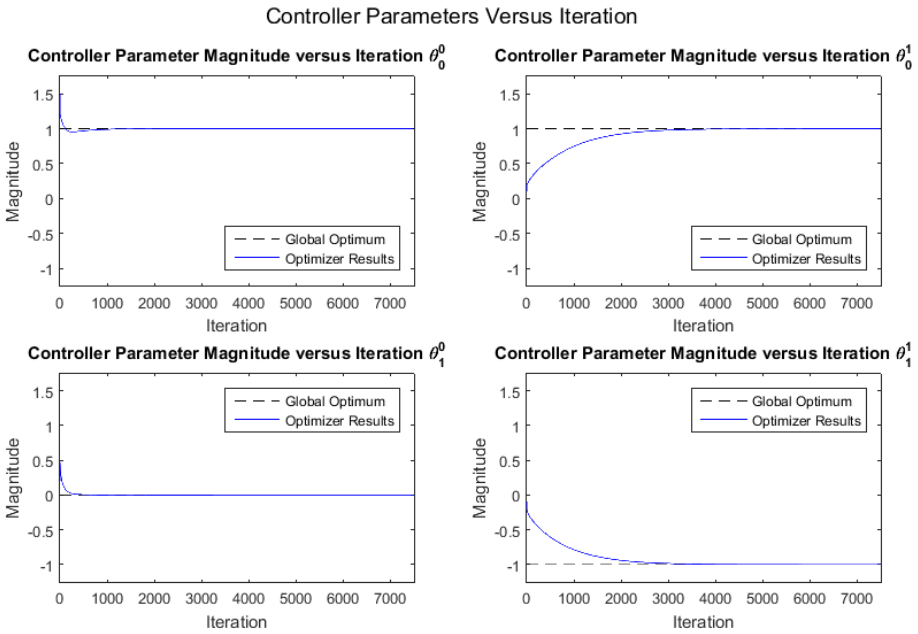

Figure 3-7: Evolution of the controller parameters per iteration when using a deterministic system.

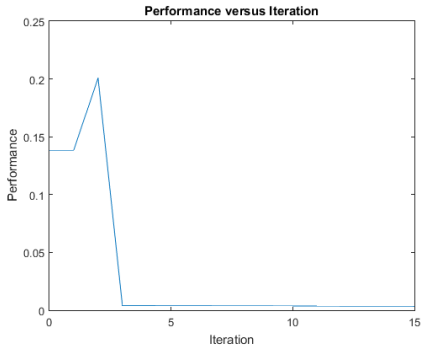

Figure 3-8: Evolution of the performance per iteration when using a deterministic system (only showing the first ten iterations).

In general the gradient experiment will be equal to

$$y_k^g(\mu_k^g) = \sum_{i=1}^{N_p} p_i(\mu_k^g) y_{k-i}^g + \sum_{j=0}^{N_b} b_j(\mu_k^g) g_{k-j} \tag{3-16b}$$

Equality between both experiments (Equations 3-16a and 3-16b) is only true if $\mu_k^g = \mu_k^r$ for all k. Let's now augment the gradient experiment by introducing two compensating signals u_k^c and f_k^c. The aforementioned signals should guarantee equality between the desired and the performed gradient experiment even when the scheduling sequences are not the same.

$$\tilde{y}_k^g(\mu_k^g) = \sum_{i=1}^{N_p} p_i(\mu_k^g) \tilde{y}_{k-i}^g + \sum_{j=0}^{N_b} b_j(\mu_k^g) \left(g_{k-j} + u_{k-j}^c + f_{k-j}^c \right) \tag{3-16c}$$

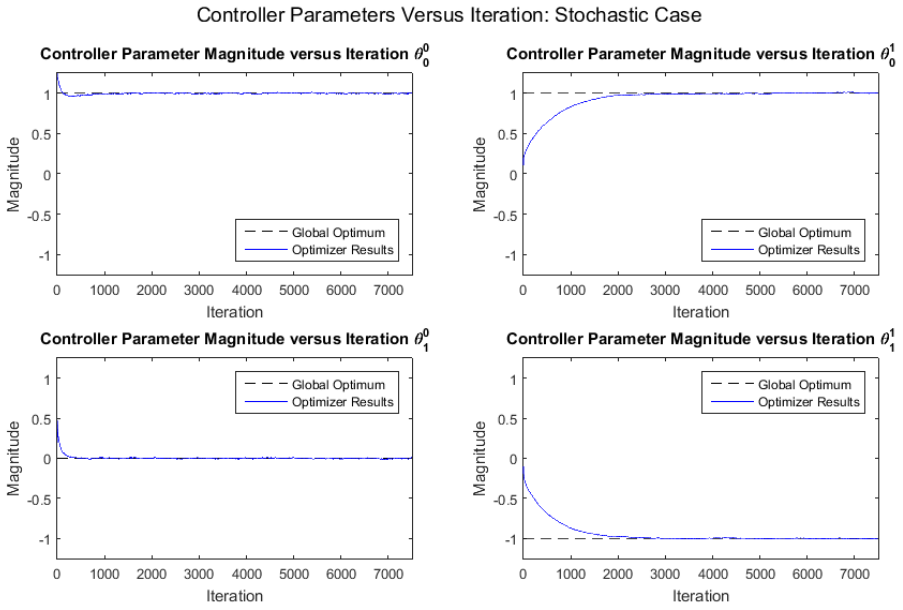

Figure 3-9: Evolution of the controller parameters per iteration when using a stochastic system.

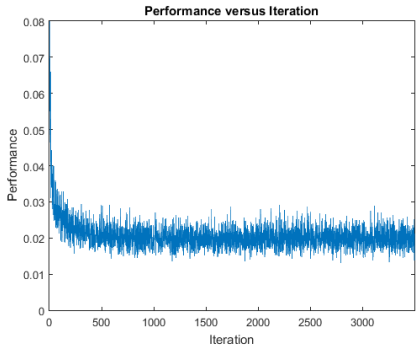

Figure 3-10: Evolution of the performance per iteration when using a stochastic system.

The input signal g_k will be corrected through u_k^c to acquire the same effect, on the dynamics, as g_k has in Equation 3-16a. For proper compensation the following equality must then hold

$$\sum_{j=0}^{N_b} b_j(\mu_k^r) g_{k-j} = \sum_{j=0}^{N_b} b_j(\mu_k^g) \left(g_{k-j} + u_{k-j}^c \right) \tag{3-16d}$$

$$\sum_{j=0}^{N_b} \left(b_j(\mu_k^r) - b_j(\mu_k^g) \right) g_{k-j} = \sum_{j=0}^{N_b} b_j(\mu_k^g) u_{k-j}^c \tag{3-16e}$$

At time k the signal u_{k-i}^c for all $i > 0$ have already been computed and are thus part of the I/O data. Therefore only computation of u_k^c is needed which leads to

$$b_0(\mu_k^g) u_k^c = \sum_{j=0}^{N_b} \left(b_j(\mu_k^r) - b_j(\mu_k^g) \right) g_{k-j} - \sum_{n=1}^{N_b} b_j(\mu_k^g) u_{k-n}^c \tag{3-16f}$$

$$u_k^c = \frac{1}{b_0(\mu_k^g)} \left(\sum_{j=0}^{N_b} \left(b_j(\mu_k^r) - b_j(\mu_k^g) \right) g_{k-j} - \sum_{n=1}^{N_b} b_j(\mu_k^g) u_{k-n}^c \right) \tag{3-16g}$$

By defining u_k^c in this fashion the gradient experiment, Equation 3-16c, can be rewritten as

$$\tilde{y}_k^g(\mu_k^g) = \sum_{i=1}^{N_p} p_i(\mu_k^g) \tilde{y}_{k-i}^g + \sum_{j=0}^{N_b} \left(b_j(\mu_k^r) g_{k-j} + b_j(\mu_k^g) f_{k-j}^c \right) \tag{3-16h}$$

The second signal f_k^c will be used to compensate the LPV system dynamics. Let's derive the expression for f_k^c by computing the output for the time steps $k = 0, 1, 2$ and generalize afterwards. For $k = 0$ it is assumed that the initial conditions are equal

$$\tilde{y}_0^g(\mu_0^g) = y_0^r(\mu_0^r) = 0 \tag{3-16i}$$

Evaluation of time step $k = 1$ gives

$$\tilde{y}_1^g(\mu_1^g) = p_1(\mu_1^g)\tilde{y}_0^g + b_0(\mu_1^r)g_1 + b_0(\mu_1^g)f_1^c = b_0(\mu_1^r)g_1 + b_0(\mu_1^g)f_1^c \tag{3-16j}$$

$$y_1^r(\mu_1^r) = p_1(\mu_1^r)y_0^r + b_0(\mu_1^r)g_1 = b_0(\mu_1^r)g_1 \tag{3-16k}$$

It can be observed that equality is attained when $f_1^c = 0$. Time step $k = 2$ gives

$$\tilde{y}_2^g(\mu_2^g) = p_1(\mu_2^g)\tilde{y}_1^g + p_2(\mu_2^g)\tilde{y}_0^g + b_0(\mu_2^r)g_2 + b_1(\mu_2^r)g_1 + b_0(\mu_2^g)f_2^c + b_1(\mu_2^g)f_1^c \tag{3-16l}$$

This can be simplified by noting that $\tilde{y}_1^g = y_1^r$, $\tilde{y}_0^g = 0$ and $f_1^c = 0$ (as derived earlier)

$$\tilde{y}_2^g(\mu_2^g) = p_1(\mu_2^g)y_1^r + b_0(\mu_2^r)g_2 + b_1(\mu_2^r)g_1 + b_0(\mu_2^g)f_2^c \tag{3-16m}$$

The output of the "correct" gradient experiment is equal to

$$y_2^r(\mu_2^r) = p_1(\mu_2^r)y_1^r + b_0(\mu_2^r)g_2 + b_1(\mu_2^r)g_1 \tag{3-16n}$$

To acquire the equality $y_2^r = \tilde{y}_2^g$ the signal f_2^c must be equal to

$$p_1(\mu_2^g)y_1^r + b_0(\mu_2^r)g_2 + b_1(\mu_2^r)g_1 + b_0(\mu_2^g)f_2^c = p_1(\mu_2^r)y_1^r + b_0(\mu_2^r)g_2 + b_1(\mu_2^r)g_1 \tag{3-16o}$$

$$p_1(\mu_2^g)y_1^r + b_0(\mu_2^g)f_2^c = p_1(\mu_2^r)y_1^r \tag{3-16p}$$

$$b_0(\mu_2^g)f_2^c = \left(p_1(\mu_2^r) - p_1(\mu_2^g) \right) y_1^r \tag{3-16q}$$

$$f_2^c = \frac{1}{b_0(\mu_2^g)} \left(p_1(\mu_2^r) - p_1(\mu_2^g) \right) y_1^r \tag{3-16r}$$

This can be generalized to

$$f_k^c = \frac{1}{b_0(\mu_k^g)} \left(\sum_{i=1}^{N_p} \left(p_i(\mu_k^r) - p_i(\mu_k^g) \right) y_{k-i}^r - \sum_{n=1}^{N_b} b_j(\mu_k^g) f_{k-n}^c \right) \tag{3-16s}$$

Evaluating this function for Equations 3-16r and 3-16j leads to the correct results. Based on these theoretical results the following augmented gradient experiment can be defined.

Augmented Gradient Experiment

The arbitrarily scheduled gradient experiment can be defined as

$$y_k^g(\mu_k^g) = \sum_{i=1}^{N_p} p_i(\mu_k^g) y_{k-i}^g + \sum_{j=0}^{N_b} b_j(\mu_k^g) u_{k-j}^{ff} \tag{3-17a}$$

With the feedforward signal u_k^{ff} set equal to

$$u_k^{ff} = g_k + u_k^c + f_k^c \tag{3-17b}$$

The three individual signals are defined as follows (with μ_k^r the scheduling variable at time k of the reference experiment)

$$g_k = r_k - y_k \tag{3-17c}$$

$$u_k^c = \frac{1}{b_0(\mu_k^g)} \left(\sum_{j=0}^{N_b} \left(b_j(\mu_k^r) - b_j(\mu_k^g) \right) g_{k-j} - \sum_{n=1}^{N_b} b_j(\mu_k^g) u_{k-n}^c \right) \tag{3-17d}$$

$$f_k^c = \frac{1}{b_0(\mu_k^g)} \left(\sum_{i=1}^{N_p} \left(p_i(\mu_k^r) - p_i(\mu_k^g) \right) y_{k-i}^r - \sum_{n=1}^{N_b} b_j(\mu_k^g) f_{k-n}^c \right) \tag{3-17e}$$

This augmented gradient experiment can be used for any LPV system. The author conjectures that qLPV systems can also be tuned in this fashion. A better analysis might be needed to evaluate whether unwanted vibrations or other side-effects might occur in practical applications. Because the signal u_k^{ff} can be large, for example when the scheduling variable changes from 1 to 10^5, which can potentially lead to actuator saturation.

Numerical Verification

In the preceding section the claim was made that compensation can be achieved by augmenting the gradient experiment. This will now be numerically verified by comparing the controller parameter evolution when comparing dedicated and non-dedicated scheduling. The initial controller parameter matrix is set equal to

$$\Theta_0 = \begin{bmatrix} 0.2 & 0.1 & 0 \\ 0.1 & 0 & -0.1 \end{bmatrix} \tag{3-18}$$

The scheduling sequence was changed to $\mu_k^g = -0.4\cos\left(\frac{2\pi}{N}k\right)$ for the augmented gradient experiment. All other experiments were performed with $\mu_k^r = 0.4\sin\left(\frac{2\pi}{N}k\right)$. The numerical results are shown in Figures 3-11 and 3-12 for the augmented and dedicated gradient experiments, respectively. Comparing both resulting controller parameters over all iterations,through the use of a norm, gave a discrepancy in the order of 10^{-15}. Thereby verifying the claim that compensation is possible through the proposed method. In Appendix B-7 proofs are provided for the augmented gradient experiment w.r.t. the stochastic properties and LPV MIMO systems.

The compensation method can therefore be considered a success and extends the applicability of the ILPVFT algorithm to a much wider range of LPV systems. The question which now remains is; can the same feat be accomplished without a model? Succeeding in this feat will be the topic of the next chapter through the use of a lifted form of the LPV-ARX model.

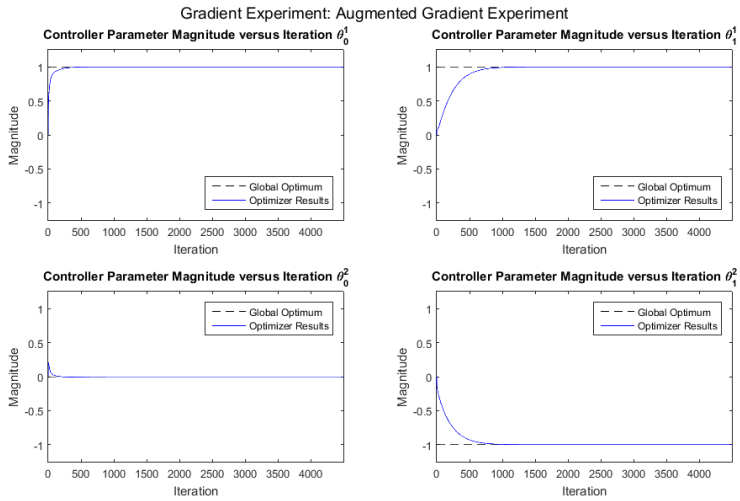

Figure 3-11: Evolution of the controller parameters per iteration for the augmented gradient experiment.

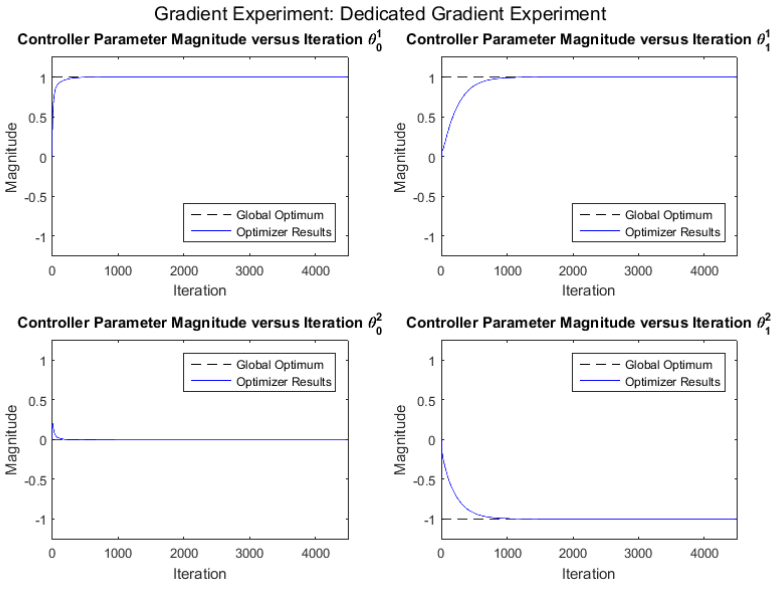

Figure 3-12: Evolution of the controller parameters per iteration for the gradient experiment with dedicated scheduling.

Chapter 4

Lifted LPV-ARX Model

In the previous chapter an algorithm was developed based on a LPV-ARX representation of the system. It was shown that through the use of an extended regressor that LPV control can be embedded in the IFT framework. Unfortunately to be able to compensate for a change in the scheduling sequence an augmentation of the gradient experiment was necessary. This augmentation meant that a LPV model of the plant is needed to properly compensate for the change in scheduling sequence.

In this chapter a lifted representation of the LPV-ARX model will be used to integrate LPV control into the IFT framework (dubbed *A2S-ILPVFT*). Through the use of a lifted representation it will be shown that a model-free integration of LPV systems is possible. The algorithm will be developed using a factorization strategy similar to [11]. As the matrices involved are banded (Appendix C-2) it was hoped that, when compared to [11], a less significant curse of dimensionality would be incurred. Unfortunately the same curse will be shown to appear for the LPV-ARX model.

The first section (4-1) will discuss the modelling of the LPV system dynamics and the mathematical tools needed for integrating LPV control with a lifted LPV-ARX model structure. Through the use of the mathematical tools from the first section a new set of dedicated experiments will be developed (4-2). This new set of experiments will be shown to need a factorisation matrix to compensate for a change in scheduling sequence. The factorisation matrix will be derived and analyzed in Section 4-3. Special interest will be in the dimensionality of the factorisation matrix. As the dimensionality of the factorization matrix was the source of the curse of dimensionality in [11]. The final section (4-4) of this chapter will numerically verify the developed algorithm.

4-1 The System Dynamics & Mathematical Tools

The modelling paradigm which will be used is a lifted representation of the LPV ARX model. This derivation will have similarities with its lifted LTI counterpart of Chapter 2. The, not

lifted, model of the system which will be used is equal to

$$y_k(\mu_k) = \sum_{i=1}^{n_a} a_i(\mu_k) y_{k-i} + \sum_{j=0}^{n_b} b_j(\mu_k) u_{k-j} \tag{4-1a}$$

The Toeplitz structure shown in Equation 4-1b will be used (based on the LPV-ARX model). The superscript n indicates which set of scheduling variables is used. For example if $n = r$ it indicates usage of reference experiment's scheduling variables.

$$Y^n(\mu^n) = \mathcal{A}(\mu^n) Y^n(\mu^n) + \mathcal{B}(\mu^n) U^n(\mu^n) \tag{4-1b}$$

With the matrix $\mathcal{B}$ as defined in Equation 4-1c (derived from the LPV-ARX model). The matrix $\mathcal{A}$ is defined in a similar fashion (with $a_0(\mu_k) = 0$ for all k).

$$\mathcal{B}(\mu^n) = \begin{bmatrix} b_0(\mu_1^n) & 0 & \dots & 0 & 0 & \dots & 0 \\ b_1(\mu_2^n) & b_0(\mu_2^n) & \dots & 0 & 0 & \dots & 0 \\ \vdots & \ddots & \ddots & \vdots & \vdots & \vdots & \vdots \\ b_{nb}(\mu_{nb}^n) & b_{nb-1}(\mu_{nb}^n) & \ddots & b_0(\mu_{nb}^n) & 0 & \dots & 0 \\ 0 & b_{nb}(\mu_{nb+1}^n) & \ddots & b_1(\mu_{nb+1}^n) & b_0(\mu_{nb+1}^n) & \dots & 0 \\ \vdots & \vdots & \ddots & \ddots & \ddots & \ddots & \vdots \\ 0 & 0 & \dots & \dots & \dots & \dots & b_0(\mu_N^n) \end{bmatrix} \tag{4-1c}$$

The control law will be defined as ($\mathcal{H}(\mu^n)$ will have a structure similar to $\mathcal{B}(\mu^n)$)

$$U^n = \mathcal{H}(\mu^n)(R^n - Y^n(\mu^n)) \tag{4-1d}$$

Therefore the closed loop Toeplitz representation will be equal to

$$Y^n(\mu^n) = (\mathcal{A}(\mu^n) - \mathcal{B}\mathcal{H}(\mu^n)) Y^n(\mu^n) + \mathcal{B}\mathcal{H}(\mu^n) R^n \tag{4-1e}$$

To compute the gradient it must be known which matrices depend on the controller parameters (Θ) thus let's explicitly add this dependency

$$Y^n(\Theta, \mu^n) = (\mathcal{A}(\mu^n) - \mathcal{B}\mathcal{H}(\mu^n, \Theta)) Y^n(\Theta, \mu^n) + \mathcal{B}\mathcal{H}(\mu^n, \Theta) R^n \tag{4-1f}$$

The derivative with respect to the i'th controller parameters is equal to

$$\frac{\partial Y^n(\Theta, \mu^n)}{\partial \theta_i} = (\mathcal{A}(\mu^n) - \mathcal{B}\mathcal{H}(\mu^n, \Theta)) \frac{\partial Y^n(\Theta, \mu^n)}{\partial \theta_i} + \mathcal{B} \frac{\partial \mathcal{H}(\mu^n, \Theta)}{\partial \theta_i} (R^n - Y^n(\Theta, \mu^n)) \tag{4-1g}$$

This can be rewritten to the variant below (with $\mathbf{I}$ an appropriately sized identity matrix)

$$\frac{\partial Y^n(\Theta, \mu^n)}{\partial \theta_i} = (\mathbf{I} - \mathcal{A}(\mu^n) + \mathcal{B}\mathcal{H}(\mu^n, \Theta))^{-1} \mathcal{B} \frac{\partial \mathcal{H}(\mu^n, \Theta)}{\partial \theta_i} (R^n - Y^n(\Theta, \mu^n)) \tag{4-1h}$$

The last important mathematical tool is the ability to factorize LPV Toeplitz matrices. This factorisation must "divide" the LPV Toeplitz matrix in two matrices namely; one dependent solely on the scheduling variables and the second filled with only scalars. It can be shown that LPV Toeplitz matrices can, in general, be factorized as

$$\mathcal{T}(\mu) = M(\mu) \hat{T} \tag{4-1i}$$

Applying this factorization to the gradient, i.e. Equation 4-1h, gives

$$\frac{\partial Y^n(\Theta,\mu^n)}{\partial \theta_i} = M^n(\mu^n)\left((\mathbf{I}-\hat{A}+\hat{B}\hat{H}(\Theta))^{-1}\hat{B}\frac{\partial \mathcal{H}(\mu,\Theta)}{\partial \theta_i}\right)(R^n - Y^n(\Theta,\mu)) \tag{4-1j}$$

Through use of the mathematical knowledge gained in this section LPV control will be integrated into the IFT framework for a lifted representation of the LPV-ARX system.

4-2 Integration into the Iterative Feedback Tuning Framework

In the previous section the mathematical tools needed for integrating LPV control into the IFT framework have been derived. The goal of A2S-ILPVFT is estimating the gradient of the cost function in an efficient manner. The, lifted, cost function is equal to

$$J(\Theta) = \frac{1}{2N}\left(Y^r(\mu^r,\Theta) - Y^d(\mu^r)\right)^T\left(Y^r(\mu^r,\Theta) - Y^d(\mu^r)\right) \tag{4-2a}$$

The gradient which must be estimated, for an arbitrary controller parameter θ_i, is therefore equal to

$$\frac{\partial J(\Theta)}{\partial \theta_i} = \frac{1}{N}\left(Y^r(\mu^r,\Theta) - Y^d(\mu^r)\right)^T\frac{\partial Y^r(\mu^r,\Theta)}{\partial \theta_i} \tag{4-2b}$$

The first experiment which must be performed is the reference experiment which is defined as

$$Y^r = (\mathbf{I} - \mathcal{A}(\mu^r) + \mathcal{B}\mathcal{H}(\mu^r))^{-1}\mathcal{B}\mathcal{H}(\mu^r)R^r \tag{4-2c}$$

A visualization of the reference experiment is shown in the block diagram of Figure 4-1. After performing the reference experiment the known I/O data will be; R^r, μ^r and $Y^r(\mu^r)$.

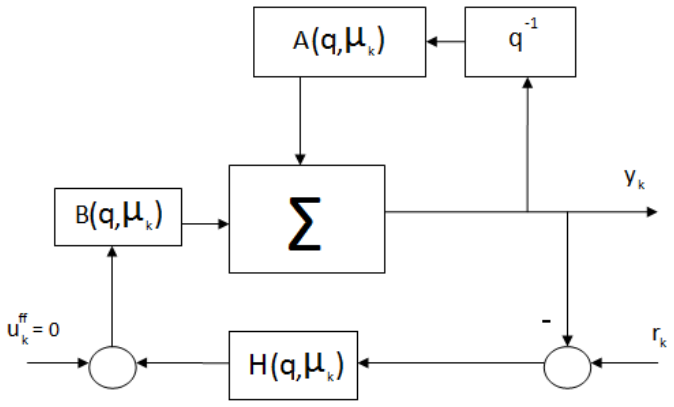

Figure 4-1: Reference Experiment using a LPV-ARX model structure.

The *correct* gradient, as derived in Equation 4-1g, which must be estimated will then be (note the explicit dependency on the reference experiment's scheduling sequence μ^r!)

$$\frac{\partial Y^r(\mu^r,\Theta)}{\partial \theta_i} = (\mathcal{A}(\mu^r) - \mathcal{B}\mathcal{H}(\mu^r,\Theta))\frac{\partial Y^r(\mu^r,\Theta)}{\partial \theta_i} + \mathcal{B}\frac{\partial \mathcal{H}(\mu^r,\Theta)}{\partial \theta_i}(R^r - Y^r(\mu^r,\Theta)) \tag{4-2d}$$

The feedforward input is equal to $\frac{\partial \mathcal{H}(\mu^r,\Theta)}{\partial \theta_i}(R^r - Y^r)$ which consists of known data; the partial derivative w.r.t. one of the controller parameters is known from the controller definition and the I/O data (R^r, μ^r and Y^r) from the reference experiment. This experiment can, as shown in Equation 4-1j, be factorized as

$$\frac{\partial Y^r(\mu^r,\Theta)}{\partial \theta_i} = M(\mu^r)\left((\mathbf{I} - \hat{A} + \hat{B}\hat{H}(\Theta))^{-1}\hat{B}\frac{\partial \mathcal{H}(\mu^r,\Theta)}{\partial \theta_i}(R^r - Y^r(\mu^r))\right) \tag{4-2e}$$

However performing the gradient experiment with an arbitrary scheduling sequence leads to (note the explicit dependency of the output on the different scheduling sequence μ^g!)

$$\frac{\partial Y^r(\mu^g,\Theta)}{\partial \theta_i} = (\mathcal{A} - \mathcal{B}\mathcal{H}(\mu^g,\Theta))\frac{\partial Y^r(\mu^g,\Theta)}{\partial \theta_i} + \mathcal{B}\frac{\partial \mathcal{H}(\mu^r,\Theta)}{\partial \theta_i}(R^r - Y^r(\mu^r)) \tag{4-2f}$$

The gradient experiments of Equation 4-2d and 4-2f are only equal if $\mu^g = \mu^r$. Therefore one must compensate the output of Equation 4-2f. In Figure 4-2 the gradient experiment is visualised. The signals, in Figure 4-2, can be equated to the following signals in Equation 4-2f $y_k^g = \frac{\partial y_k}{\partial \theta_i}$ and $u_k^{ff} = \frac{\partial \mathcal{H}(\mu_k^r,q)}{\partial \theta_i}(r_k - y_k(\mu_k^r))$.

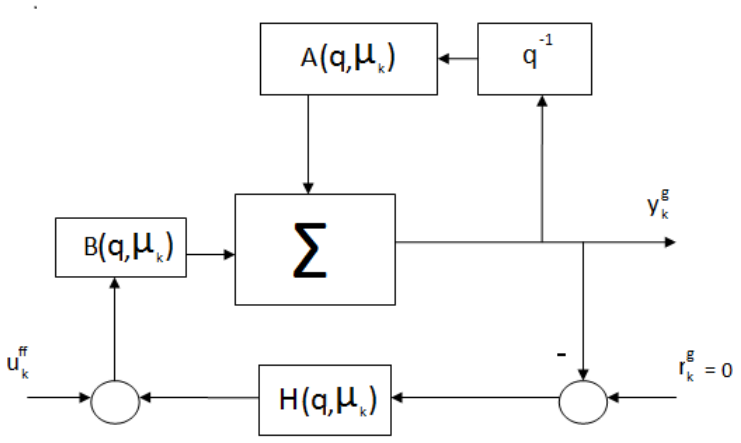

Figure 4-2: The LPV Gradient Experiment in LPV-ARX model representation.

The gradient experiment with the scheduling sequence μ^g can be factorized as (the superscript j indicates the experiment number the reason why will be apparent later on)

$$\frac{\partial Y^{r,j}(\mu^g,\Theta)}{\partial \theta_i} = M^j(\mu^{g,j})\left((\mathbf{I} - \hat{A} + \hat{B}\hat{H}(\Theta))^{-1}\hat{B}\frac{\partial \mathcal{H}(\mu^r,\Theta)}{\partial \theta_i}(R^r - Y^r(\mu^r))\right) \tag{4-2g}$$

Thus by comparing the *factorized* gradient experiment output of Equation 4-2g with the desired *factorized* gradient, i.e. Equation 4-2e, it can be seen that the left (pseudo-)inverse of the matrix $M^j(\mu^{g,j})$ is needed (with X^+ the pseudo-inverse of matrix X)

$$\frac{\partial Y^{r,j}(\mu^r,\Theta)}{\partial \theta_i} = M(\mu^r)\left(M(\mu^{g,j})^+\frac{\partial Y^{r,j}(\mu^g,\Theta)}{\partial \theta_i}\right) \tag{4-2h}$$

The problem with solving this equation is that the solution is, in general, not unique when only a single gradient experiment is performed. Because to have a unique solution to the

equation $A^+b = x$ the matrix A must have *full* column rank. To solve this rank problem define the matrix $M(\mu^g)$ which stacks a set of $\mathcal{L}$ factorization matrices $M^j(\mu^{g,j})$

$$M(\mu^g) = \begin{bmatrix} M^1(\mu^{g,1}) \\ M^2(\mu^{g,2}) \\ \vdots \\ M^{\mathcal{L}}(\mu^{g,\mathcal{L}}) \end{bmatrix} \tag{4-2i}$$

Equation 4-2h can then be rewritten to the one below with $\frac{\partial Y^r(\mu^g,\Theta)}{\partial \theta_i}$ a stacked vector

$$\frac{\partial Y^r(\mu^r,\Theta)}{\partial \theta_i} = M(\mu^r)\left(M(\mu^g)^+ \frac{\partial Y^r(\mu^g,\Theta)}{\partial \theta_i}\right) \tag{4-2j}$$

To guarantee that there is only one unique solution to Equation 4-2j one needs to ensure that the matrix $M(\mu^g)$ has full column rank. This rank condition can be fulfilled by performing the gradient experiment multiple times. Under the assumption that every row is independent this is equal to stating that the amount of rows must be at least equal to the amount of columns.

The *minimum* amount of "repeated" experiments required will then be equal to (with n and m the amount of columns and rows of the, as of yet undefined, factorization matrix $M^j(\mu^{g,j})$)

$$\text{ceil}\left(\frac{n}{m}\right) = \mathcal{L} \tag{4-2k}$$

Therefore by performing an amount of $\mathcal{L}$ repeated gradient experiments will ensure that Equation 4-2j has a unique solution (under the assumption of full column rank). Thereby compensating the outputs for the change in scheduling sequence.

An interesting observation is that this condition disappears if and only if $M^r(\mu^r) = M^r(\mu^{r,j})$ because then (by using a property of the pseudo-inverse)

$$M^r(\mu^r)M^j(\mu^{g,j})^+M(\mu^{g,j}) = M^r(\mu^r)M^j(\mu^{r,j})^+M^j(\mu^{r,j}) = M^r(\mu^r) \tag{4-2l}$$

This makes factorization redundant when the scheduling sequence is the same in both experiments.

One of the key advantages of linear IFT and ILPVFT (Chapter 3) is that only two experiments need to be performed to acquire the entire gradient. In Section 2-2 it was shown that the partial derivative w.r.t. a controller parameter, for an LTI controller, is a LTI filter. This result was used to prove that only one gradient experiment needs to be performed.

The same approach can be used for an LPV controller when its, partial, derivative is equal to (with ψ_k^i the *scalar* basis function belonging to θ_i)

$$\frac{\partial \mathcal{H}(\mu_k,\Theta)}{\partial \theta_i} = \mathcal{F}_i = F_i\psi_k^i \tag{4-2m}$$

The partial derivative, shown in Equation 4-2m, is a multiplication of an LTI-ARX filter F_i and a scalar basis function ψ_k^i. An example of a controller which fulfils this criterion is the

LPV PI controller from Section 3-2. The Toeplitz matrix of $\frac{\partial \mathcal{H}(\mu_k,\Theta)}{\partial \theta_i}$ representation can then be written as

$$\frac{\partial \mathcal{H}(\mu,\Theta)}{\partial \theta_i} = F_i \begin{bmatrix} \psi(\mu_1) & 0 & \dots & 0 \\ 0 & \psi(\mu_2) & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & \psi_(\mu_N) \end{bmatrix} = F_i \Psi^r(\mu) \tag{4-2n}$$

The result of Equation 4-2n can be incorporated into Equation 4-2e

$$\frac{\partial Y^r(\mu^r,\Theta)}{\partial \theta_i} = F_i \left(\mathbf{I} - \mathcal{A}(\mu^r) + \mathcal{B}\mathcal{H}(\mu^r,\Theta)\right)^{-1} \mathcal{B}\Psi^g(\mu)(R^r - Y^r(\mu^r,\Theta)) \tag{4-2o}$$

From Equation 4-2o it can be observed that only one gradient experiment per basis function needs to be performed. This is a considerable improvement as the number of basis functions will *always* be lower than or equal to the amount of controller parameters. The partial derivative of Equation 4-2m is a general type of nonlinear controller namely an affine LPV controller which has a form similar to

$$u_k = \left(H_1(q)\theta_1 + H_2(q)\theta_2\right)\psi_1(\mu_k) + H_3(q)\theta_3\psi_2(\mu_k) \tag{4-2p}$$

For a controller of this format the amount of experiments can therefore be reduced. The factorized gradient experiment will then look like

$$\frac{\partial Y^r(\Theta,\mu^g)}{\partial \theta_i} = F_i M(\mu^g)\left((\mathbf{I} - \hat{A} + \hat{B}\hat{H}(\Theta))^{-1}\hat{B}\Psi^r(R^r - Y^r(\mu^r))\right) \tag{4-2q}$$

When the controller derivative can't be written in the form of Equation 4-2m then one gradient experiment per controller parameter must be performed. The derivation in this section rely heavily on the factorization matrix $M^n(\mu^n)$ however it has not yet been formally defined. Special interest is in the dimensionality of the factorization matrix as it was the key problem in IFT-LPV [11].

4-3 The Factorisation Matrix $M(\mu^n)$

The previous section showed the integration of LPV control into the IFT framework from a lifted I/O representation. Although one important matrices which was only slightly touched upon is the factorisation matrix $M(\mu^n)$. It was observed that this matrix must have full column rank to compute a unique solution of the gradient. The amount of columns of $M(\mu^n)$ therefore directly affects the magnitude of the amount of repeated gradient experiments (Equation 4-2k). This section will show that by modelling the system as a LPV-FIR filter that the dimensionality will scale linearly.

Factorization of Equation 4-2g is needed. First re-write Equation 4-2g (for brevity) to

$$\frac{\partial Y^r(\Theta,\mu^g)}{\partial \theta_i} = (\mathbf{I} - \mathcal{P}(\mu^g))^{-1}\mathcal{B}(\mu^g)G \tag{4-3a}$$

With $\mathcal{P}(\mu^g) = \mathcal{A}(\mu^g) - \mathcal{B}\mathcal{H}(\mu^g,\Theta)$ and $G = \frac{\partial \mathcal{H}(\mu^r,\Theta)}{\partial \theta_i}(R^r - Y^r(\mu^r))$. Unfortunately Equation 4-3a's factorization matrix shows the same exponential increase in dimensionality as in [11] when $\mathcal{P}(\mu^g) \neq 0$ (for the proof see Appendix C-2).

For a more favourable, in terms of dimensionality, factorization a LPV-FIR filter (i.e. $\mathcal{P}(\mu^g) = 0$) is used. By modelling the system in this fashion the banded structure can be preserved (Appendix C-2). The LPV-FIR filter will be defined as follows

$$y_k = \sum_{i=0}^{N_c} c_i(\mu_k) g_{k-i} \tag{4-3b}$$

The amplitudes $c_i(\mu_k)$ can then be defined as (with N_f the amount of basis functions, d_i^j some scalar and $\psi_j(\mu_k)$ the basis function)

$$c_i = \sum_{j=0}^{N_f} d_i^j \psi_j(\mu_k) \tag{4-3c}$$

The LPV-FIR filter can now be rewritten into the form

$$y_k = \sum_{i=0}^{N_c} \sum_{j=0}^{N_f} d_i^j \psi_j(\mu_k) g_{k-i} \tag{4-3d}$$

Note that the series of Equation 4-3d is finite for every time step k. Factorizing a single time step for the input g_{k-i} leads to

$$\sum_{j=0}^{N_f} d_i^j \psi_j(\mu_k) g_{k-i} = \begin{bmatrix} 1 & \psi_1(\mu_k) & \psi_2(\mu_k) & \dots & \psi_{N_f}(\mu_k) \end{bmatrix} \begin{bmatrix} d_i^0 \\ d_i^1 \\ \vdots \\ d_i^{N_f} \end{bmatrix} g_{k-i} \tag{4-3e}$$

An extension to incorporate a second input g_{k-i+1} leads to

$$\sum_{n=i}^{i+1} \sum_{j=0}^{N_f} d_i^j \psi_j(\mu_k) g_{k-n} = \begin{bmatrix} 1 & \psi_1(\mu_k) & \psi_2(\mu_k) & \dots & \psi_{N_f}(\mu_k) \end{bmatrix} \begin{bmatrix} d_i^0 & d_{i+1}^0 \\ d_i^1 & d_{i+1}^1 \\ \vdots & \\ d_i^{N_f} & d_{i+1}^{N_f} \end{bmatrix} \begin{bmatrix} g_{k-i} \\ g_{k-i+1} \end{bmatrix} \tag{4-3f}$$

This can be written compactly as

$$\sum_{n=i}^{i+1} \sum_{j=0}^{N_f} d_i^j \psi_j(\mu_k) g_{k-n} = \mathcal{M}_k \begin{bmatrix} \hat{\mathbf{d}}_i & \hat{\mathbf{d}}_{i+1} \end{bmatrix} \begin{bmatrix} g_{k-i} \\ g_{k-i+1} \end{bmatrix} \tag{4-3g}$$

The matrix $\mathcal{M}$'s subscript indicates the time step which it factorizes. The variable $\hat{\mathbf{d}}_i$ constitutes a vector containing all the amplitudes belonging to the set of amplitudes $d_i^0, d_i^1, ..., d_i^{N_f}$. This is shown in the lifted representation in Equation 4-3h (the (desired) structure $Y = M(\mu)\hat{T}G$ is clearly visible). Note that Y is a column vector containing the outputs (i.e. $[y_1, y_2, ..., y_N]^T$)

$$Y = \begin{bmatrix} \mathcal{M}_1 & 0 & \dots & 0 & 0 & \dots & 0 \\ 0 & \mathcal{M}_2 & \dots & 0 & 0 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \dots & \vdots \\ 0 & 0 & \dots & \mathcal{M}_{N_c} & 0 & \dots & 0 \\ 0 & 0 & \dots & 0 & \mathcal{M}_{N_c+1} & \dots & 0 \\ \vdots & \vdots & \dots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 0 & 0 & \dots & \mathcal{M}_N \end{bmatrix} \begin{bmatrix} \hat{\mathbf{d}}_0 & 0 & \dots & 0 & 0 & \dots & 0 \\ \hat{\mathbf{d}}_1 & \hat{\mathbf{d}}_0 & \dots & 0 & 0 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \dots & \vdots \\ \hat{\mathbf{d}}_{N_c} & \hat{\mathbf{d}}_{N_c-1} & \dots & \hat{\mathbf{d}}_0 & 0 & \dots & 0 \\ 0 & \hat{\mathbf{d}}_{N_c} & \dots & \hat{\mathbf{d}}_1 & \hat{\mathbf{d}}_0 & \dots & 0 \\ \vdots & \vdots & \dots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 0 & 0 & \dots & \hat{\mathbf{d}}_0 \end{bmatrix} \begin{bmatrix} g_1 \\ g_2 \\ \vdots \\ g_{N_c} \\ g_{N_c+1} \\ \vdots \\ g_N \end{bmatrix} \tag{4-3h}$$

In Equation 4-3h it can be observed that the dimensionality scales linearly using this factorization. As a maximum of $(N_f + 1)(N_c + 1)$ extra columns are added per measurement. The linear scaling of dimensionality is a significant improvement over the exponential scaling seen in the LPV-ARX model (Appendix C-2) or IFT-LPV.

This concludes the mathematical foundations of the A2S-ILPVFT algorithm. A proof regarding the unbiasedness of the gradient estimate is provided in Appendix C-1. The final section of this chapter will provide a numerical verification of the A2S-ILPVFT algorithm.

4-4 Case Study

In the previous sections it was shown that based on a lifted representation of the LPV-ARX model that one can embed LPV control in the IFT framework without needing a model. Based on the mathematical foundations the algorithm was implemented in Matlab. By using the same LPV plant as discussed in Section 3-4 a numerical verification will be performed. To verify whether the algorithm compensates the scheduling sequence change the following adaption was made (d_k a random number $\in [-0.075, 0.05]$):

$$\mu_k^r = \sin\left(\frac{2\pi}{N}k\right) \tag{4-4a}$$

$$\mu_k^g = \sin\left(\frac{2\pi}{N}k\right) + d_k \tag{4-4b}$$

The same initial conditions (Equations 3-14 and 3-15), reference signal (Figure 3-2) and optimization parameters (Table 3-1) were used as in the previous chapter. The general type of correct factorization is equal to (derived from the full factorization of a LPV-ARX model in Appendix C-2)

$$\sum_{n=i}^{i+1}\left(\sum_{j=0}^{3} d_i^j \psi_j(\mu_k)\right) g_{k-n} = \begin{bmatrix} 1 & \mu_k & \mu_{k-1} & \mu_k\mu_{k-1} & \mu_{k-2} & \mu_k\mu_{k-2} & \mu_{k-1}\mu_{k-2} \end{bmatrix} \begin{bmatrix} d_i^0 & d_{i+1}^0 \\ d_i^1 & d_{i+1}^1 \\ d_i^2 & d_{i+1}^2 \\ d_i^3 & d_{i+1}^3 \\ d_i^4 & d_{i+1}^4 \\ d_i^5 & d_{i+1}^5 \\ d_i^6 & d_{i+1}^6 \end{bmatrix} \begin{bmatrix} g_{k-i} \\ g_{k-i+1} \end{bmatrix} \tag{4-4c}$$

The factorization matrix was implemented as in Equation 4-6 using μ_{k-4} through μ_k. The controller as defined in Equation 3-12c is in control affine form therefore the reduced number of experiments can be used. The filters needed were already derived in the previous chapter in Section 3-4. The numerical results will now be discussed.

Numerical Results

The results for the initial condition of Equation 3-14 are shown in Figures 4-3 and 4-4 for a deterministic system. It can be observed that the algorithm performs as desired as it converges to the global optimum. In Figures 4-5 and 4-6 the stochastic system's results are shown. It can be noted that three of the four controller parameters converge to the, deterministically, optimal controller parameters. However the lower right controller parameter implies a bias as it oscillates *above* the, optimal, -1 value. The author conjectures that a bias might be introduced due to the combination of the LPV-FIR filter approximation and noise. The converged performance for the stochastic system is equal to that found in Chapter 3 implying that the algorithms perform equally well.

In Figures 4-7 to 4-10 the same conclusions can be drawn for the numerical results for the second initial condition (Equation 3-15). Thereby providing a numerical verification of the algorithm working as conceived.

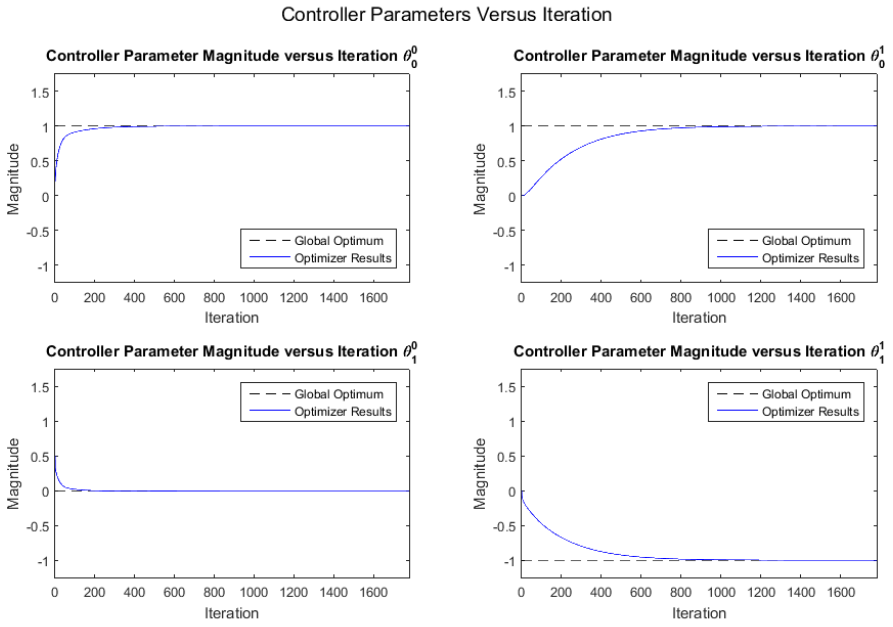

Figure 4-3: Evolution of the controller parameters per iteration when using a deterministic system.

The numerical and mathematical results of this chapter and the preceding one indicate that both algorithms work as conceived. The lifted representation has succeeded in providing a model free integration of LPV control into the IFT framework. Unfortunately the increased dimensionality of the problem and considerable increase in the amount of experiments makes

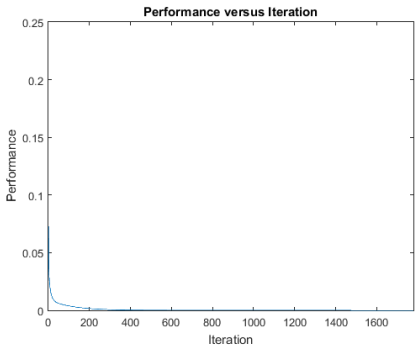

Figure 4-4: Evolution of the performance per iteration when using a deterministic system.

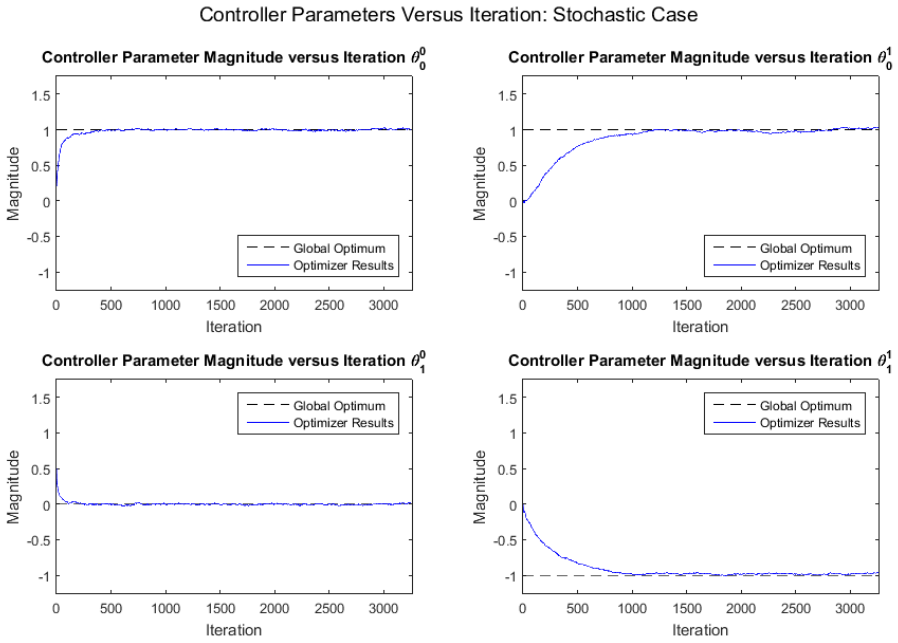

Figure 4-5: Evolution of the controller parameters per iteration when using a stochastic system.

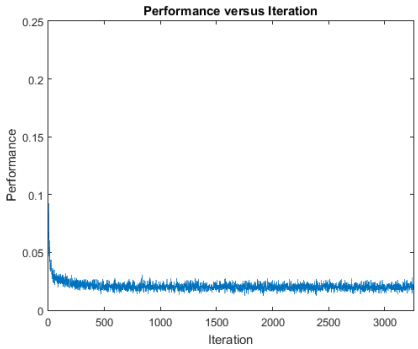

Figure 4-6: Evolution of the performance per iteration when using a stochastic system.

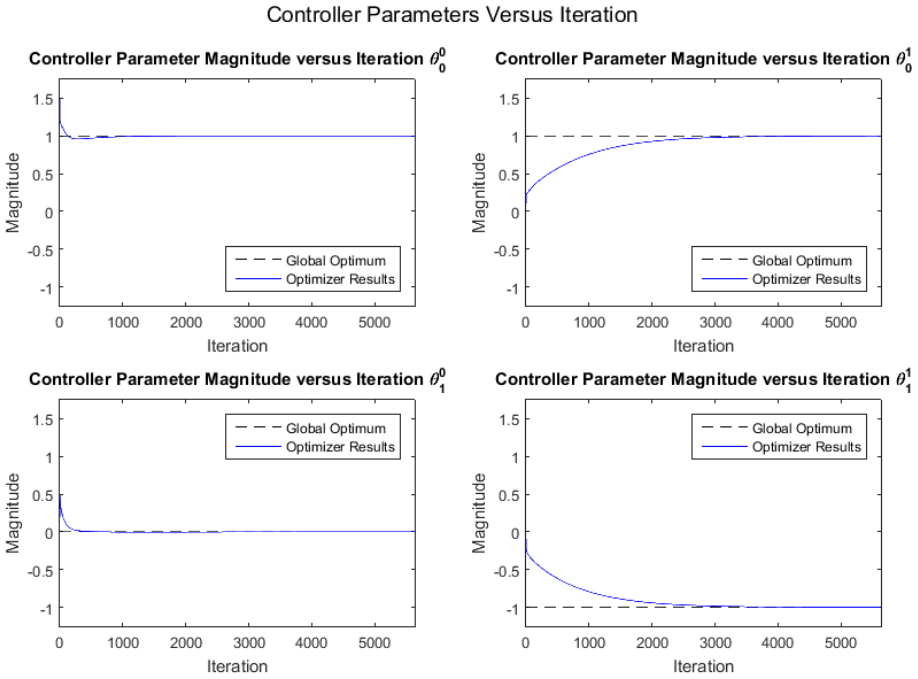

Figure 4-7: Evolution of the controller parameters per iteration when using a deterministic system.

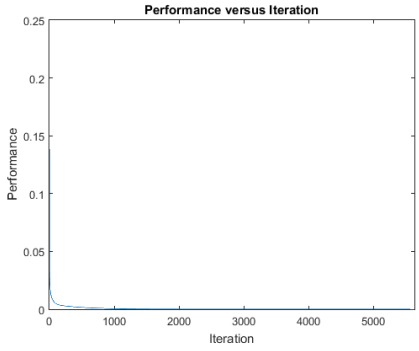

Figure 4-8: Evolution of the performance per iteration when using a deterministic system.

it much less efficient than the ILPVFT algorithm (which requires two experiments and a model of the plant).

The numerical example was not a realistic LPV system and not very challenging from a control perspective. Therefore to conclude this thesis the final chapter will be dedicated to a numerical case study with a realistic LPV model of a smart airfoil.

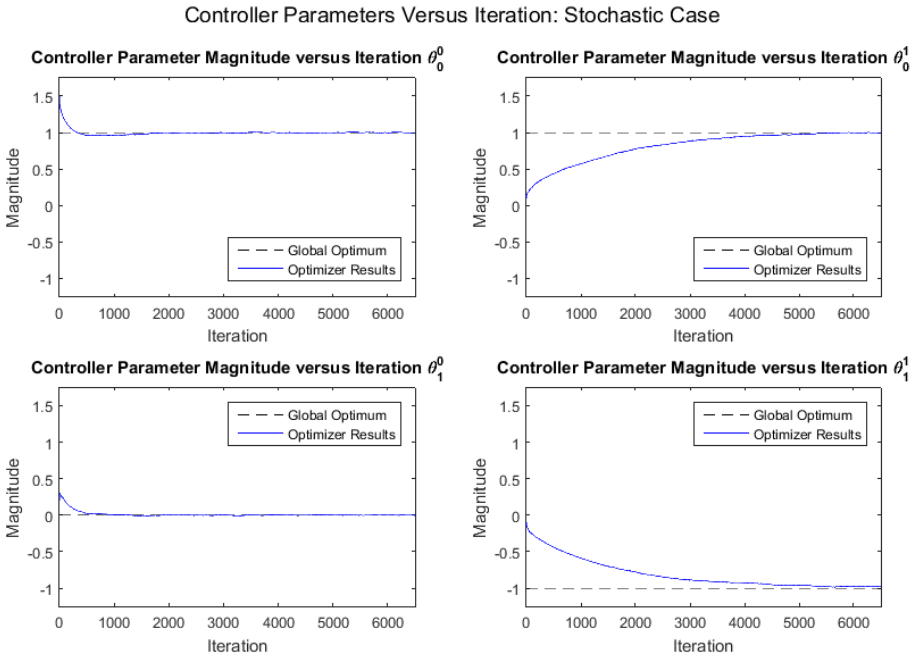

Figure 4-9: Evolution of the controller parameters per iteration when using a stochastic system.

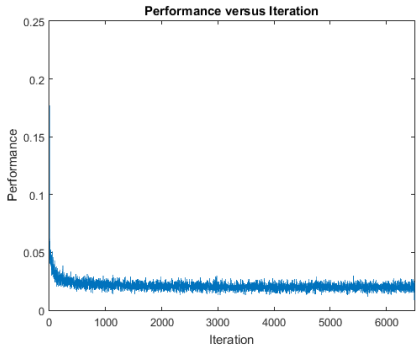

Figure 4-10: Evolution of the performance per iteration when using a stochastic system.

Chapter 5

Case Study: Data-Driven Flutter Control

Flutter is a phenomenon in the field of aero-elastic dynamics which occurs whenever the wind speed increases beyond a threshold value. The basic explanation of flutter is that the aerodynamic forces, which increase in conjunction with the wind speed, cancel out, or are higher than, the internal damping of the mechanical structure. This will lead to the structure exhibiting vibrations which can be sustained, by the aerodynamic forces, and even worsen as long as the wind speed is beyond the threshold value. A well known example of its destructive potential is the Tacoma Narrows Bridge which exhibited flutter and eventually collapsed due to structural failure due to the, flutter-induced, vibrations. This makes flutter an interesting type of LPV system as it exhibits the property of becoming unstable beyond a certain threshold value of the scheduling variable (i.e. the wind speed). The ability to control this phenomenon is needed for instance to accommodate for faster aircraft, in which the relative wind speed will increase, and wind turbines (to work even in higher wind speeds which could otherwise lead to structural failure/damage).

This case study will use a continuous time LPV model of a two-dimensional airfoil, from [12], which exhibits flutter after a certain wind speed is exceeded. A discretized version of the continuous time model will be used as the plant. The aforementioned modelling aspects will be discussed in Section 5-1. In Section 5-2 the linear IFT algorithm will be used to optimize a set of LTI controllers. This set of LTI controller will be gain scheduled and its performance will be compared to the LPV controllers (Section 5-3).

5-1 State Space Description

The state space model for this case study and its parameters are based on a model of a smart 2-D airfoil from [12]. The continuous time LPV state space model is equal to (in Table 5-1

the parameters are explained and their numerical values presented)

$$\begin{bmatrix} \dot{h} \\ \dot{\alpha} \\ \ddot{h} \\ \ddot{\alpha} \end{bmatrix} = (A_1 + A_2 V + A_3 V^2) \begin{bmatrix} h \\ \alpha \\ \dot{h} \\ \dot{\alpha} \end{bmatrix} + B_3 V^2 \beta \tag{5-1a}$$

$$\begin{bmatrix} h \end{bmatrix} = C_1 \begin{bmatrix} h \\ \alpha \\ \dot{h} \\ \dot{\alpha} \end{bmatrix} \tag{5-1b}$$

The states are plunge h and the pitch α (and their derivatives w.r.t. time) and the scheduling variable V is the wind speed (in m/s). The individual state space matrices are equal to

$$A_1 = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ \frac{I_\alpha}{q_1} k_h & \frac{-m_w x_a b}{q_1} k_\alpha & \frac{I_\alpha}{q_1} c_h & \frac{-m_w x_\alpha b}{q_1} c_\alpha \\ \frac{-m_w x_\alpha b}{q_1} k_h & \frac{m_t}{q_1} k_\alpha & \frac{-m_w x_\alpha}{q_1} c_h & \frac{m_t}{q_1} c_\alpha \end{bmatrix} \tag{5-1c}$$

$$A_2 = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{I_\alpha}{q_1} q_2 + \frac{m_w x_\alpha b}{q_1} q_4 & \frac{I_\alpha}{q_1} q_2 q_6 + \frac{m_w x_\alpha b}{q_1} q_4 q_6 \\ 0 & 0 & -\frac{m_t}{q_1} q_4 - \frac{m_w x_\alpha b}{q_1} q_2 & \frac{-m_t}{q_1} q_4 q_6 + \frac{m_w x_\alpha b}{q_1} q_2 q_6 \end{bmatrix} \tag{5-1d}$$

$$A_3 = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & \frac{I_\alpha}{q_1} q_2 + \frac{m_w x_\alpha b}{q_1} q_4 & 0 & 0 \\ 0 & -\frac{m_t}{q_1} q_4 - \frac{m_w x_\alpha b}{q_1} q_2 & 0 & 0 \end{bmatrix} \tag{5-1e}$$

$$B_3 = \begin{bmatrix} 0 \\ 0 \\ \frac{I_\alpha}{q_1} q_3 + \frac{m_w x_\alpha b}{q_1} q_5 \\ -\frac{m_t}{q_1} q_5 - \frac{m_w x_\alpha b}{q_1} q_3 \end{bmatrix} \tag{5-1f}$$

$$C_1 = \begin{bmatrix} 1 & 0 & 0 & 0 \end{bmatrix} \tag{5-1g}$$

The only variables left to define are q_i for $i = 1, 2, 3, 4, 5, 6$ which can be defined as

$$q_1 = -m_t I_\alpha + m_w^2 x_\alpha^2 b^2 \tag{5-2a}$$

$$q_2 = \rho b s_p c l_\alpha \tag{5-2b}$$

$$q_3 = \rho b c_{l_\beta} s_p \tag{5-2c}$$

$$q_4 = \rho b^2 s_p c_{m_\alpha} \tag{5-2d}$$

$$q_5 = \rho b^2 c_{m_\beta} s_p \tag{5-2e}$$

$$q_6 = (0.5 - \alpha)\, b \tag{5-2f}$$

Variable	Meaning	Value
β	Control Surface Deflection	n/a
b	Semi-chord of the wind	0.135 m
k_h,k_α	Thickness coefficients of plunge and pitch	2844.4 & 3.84 N/m
c_h,c_α	Damping coefficients of plunge and pitch	27.43 & 0.036 Ns/m
$c_{m\alpha}$,$c_{l\alpha}$	Lift and moment coefficient per angle of attack	-1.1599 & 6.28
$c_{m\beta}$,$c_{l\beta}$	Lift and moment coefficient per control surface deflection	-0.635 & 3.358
m_w,m_t	Wing mass & Total mass	2.049 & 12.387 kg
I_α	Moment of Inertia	0.0558 kgm^2
ρ	The air density	1.225 kg/m^3
a	Non-dimensionalized distance from the midchord to the elastic axis	-0.6847
x_α	Non-dimensionalized distance of the center of mass	0.3314
s_p	Span	1 m
T_s	Sampling time	0.08 s

Table 5-1: Numerical values of parameters and their physical meaning.

This fully defines the continuous time LPV state space system for the 2D smart airfoil which exhibits flutter. A discretization of the LPV model can be computed using [12]

$$A_d(V, V^2, \frac{1}{V}, \ldots) = \left(\mathbf{I} + \frac{T_s}{2}A(V, V^2)\right)\left(\mathbf{I} - \frac{T_s}{2}A(V, V^2)\right)^{-1} \tag{5-3a}$$

$$B_d(V, V^2, \frac{1}{V}, \ldots) = \sqrt{T_s}\left(\mathbf{I} - \frac{T_s}{2}A(V, V^2)\right)^{-1} B(V^2) \tag{5-3b}$$

$$C_d(V, V^2, \frac{1}{V}, \ldots) = \sqrt{T_s}C\left(\mathbf{I} - \frac{T_s}{2}A(V, V^2)\right)^{-1} \tag{5-3c}$$

$$D_d(V, V^2, \frac{1}{V}, \ldots) = \frac{T_s}{2}C\left(\mathbf{I} - \frac{T_s}{2}A(V, V^2)\right)^{-1} B(V^2) \tag{5-3d}$$

This case study will use the discrete time LPV model as the LPV plant which must be controlled and for which the LTI/LPV controller should be optimized. The next section will derive the optimal controllers for multiple LTI plants, i.e. the scheduling variable is kept constant. This will lead to a set of LTI controller which will be gain scheduled as a benchmark for the LPV controllers of Section 5-3.

5-2 Linear Time Invariant Control

The original linear IFT algorithm, as conceived by Hjalmarsson, allows for the optimization of LTI controllers. If the scheduling sequence encompasses all the linear plants, i.e. all possible scheduling variables, during the experiment then a robust LTI controller will be acquired for all the individually linear plants [20]. This property of linear IFT will not be made use of in this chapter instead the optimization will revolve around optimizing LTI controllers for a set of wind speeds. By gain scheduling these LTI controllers a benchmark for the LPV controllers

Variable	Value
Initial Controller	$[\theta_P, \theta_I, \theta_D] = [-4, -3, 1]$
Learning rate γ	0.5
Hessian	Identity matrix
Amount of measurements	200
Noise	None
Minimum performance decrease	10^{-6}

Table 5-2: Numerical values used during the optimization of the LTI controllers.

of Section 5-3 will be obtained. In the ideal case the LPV controllers will reach performance levels comparable to the optimal, gain scheduled, LTI controllers.

The set of wind speeds for which the LTI controllers will be optimized are $V = 5, 7, 9$ and 11 m/s. By defining the set in this manner one can compare the wind speeds from 4 to 12 m/s as one will have a new optimized LTI controller per 2 m/s. The desired closed loop performance, i.e. the cost function, will be equal to

$$y_k^d = \frac{1}{q^2} \frac{0.0305q + 0.027}{q^2 - 1.6346q + 0.6921} r_k \tag{5-4}$$

The input will be defined as

$$u_k = \left(\theta_P + \frac{1}{1 - q^{-1}} \theta_I + (1 - q^{-1}) \theta_D \right) e_k \tag{5-5}$$

The important numerical parameters used during optimization are defined in Table 5-2. The optimization results of the performance versus iterations for the LTI plants are shown in Figure 5-1. It can be seen that all four controller have attained considerably better performance than their respective initial controllers had. Convergence can be observed for all the individual LTI controllers.

The second point of interest is how well the LTI controllers perform over the entire spectrum of 4 to 12 m/s (Figure 5-2). The optimal LPV controllers should, in the ideal case, reach comparable performance. Figure 5-2 shows that comparable performance over the entire spectrum of wind speeds is attained. Although a slight decreasing trend can be observed for increasing wind speeds.

The final point of interest is how the optimal LTI controller parameters evolve for increasing wind speeds. This knowledge gives insight in the optimal structure which can achieve comparable performance over the entire spectrum of wind speeds (Figure 5-3). As to be able to predict how well the fixed structure, i.e. $\theta_0 + \theta_1 \mu_k$, of the next section will perform. It can be seen that the linear structure is not optimal but comparable performance should be possible for a subset of wind speeds. However due to the nonlinear relation present in all controller parameters the LPV controller will not attain comparable performance throughout the spectrum of 4 to 12 m/s.

This section has shown that gain scheduled LTI controller can attain decent performance throughout the entire spectrum of wind speeds. But the question that we want to answer is; are the novel algorithms are capable of attaining approximately the same performance?

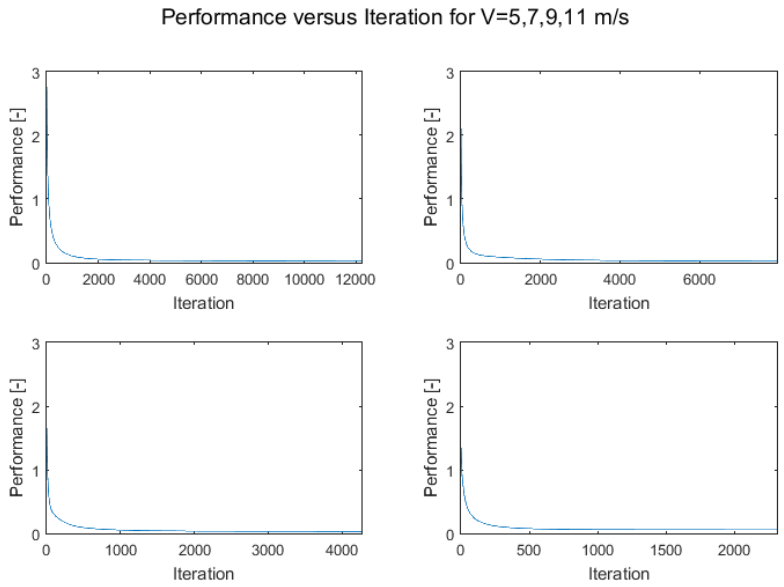

Figure 5-1: Performance versus iteration for the LTI controller for the set of wind speeds.

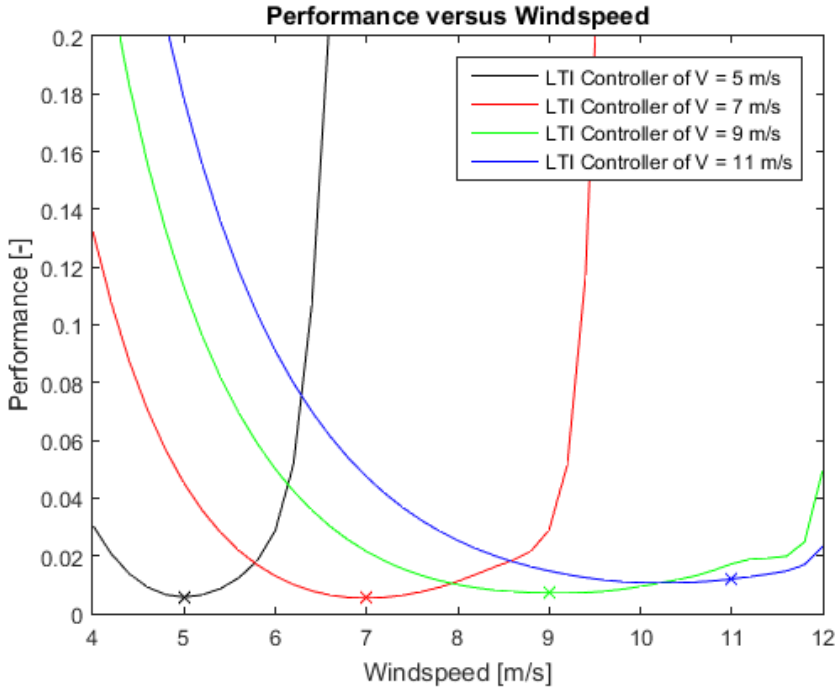

Figure 5-2: Performance of the optimized LTI controllers throughout the entire spectrum of wind speeds.

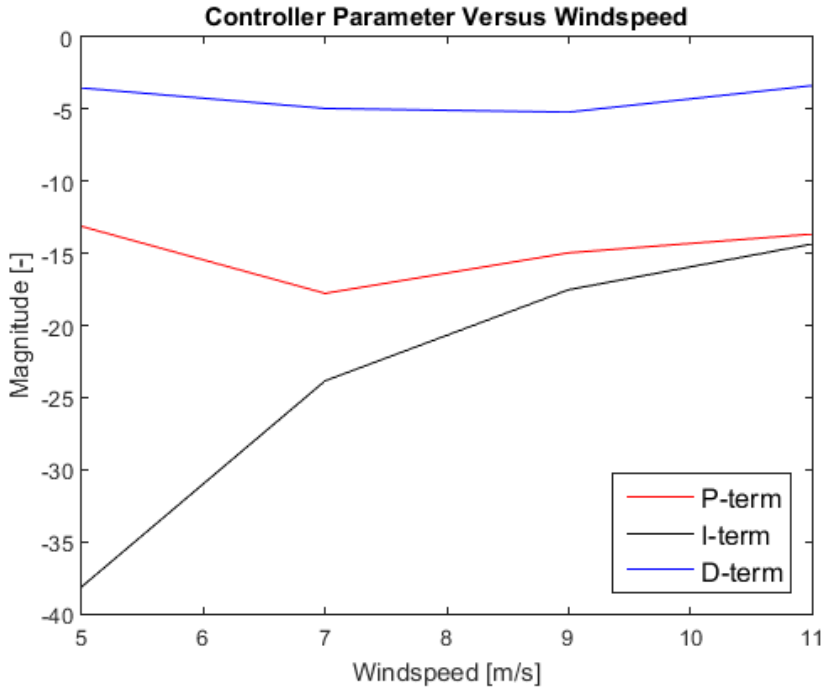

Figure 5-3: Controller parameters versus wind speed based on the optimal LTI controllers.

5-3 Linear Parameter Varying Control

This thesis has primarily been concerned with extending the applicability of the IFT framework from merely LTI control to LPV control. Due to the inherent gain scheduled nature of LPV controllers the performance, in terms of amount of experiments, of the IFT framework is improved in the following fashion:

The time spent on tuning, through use of the IFT framework, is significantly reduced as instead of optimizing an LTI controller for a set of wind speeds one only needs to optimize a single LPV controller. If the underlying structure of the LPV controller is also optimal than the performance of the gain scheduled LTI controllers can be comparable for the entire scheduling space.

Unfortunately as observed in the previous section the structure of the LPV controller (i.e. $\theta_0 + \theta_1$) will not be able to fit the optimal LTI controller parameters throughout the entire spectrum of wind speeds. So even though a gain scheduled controller is acquired after one run of the IFT framework, comparable performance throughout the entire scheduling space will not be possible.

The terms of the LPV controller will be defined as (based on an LPV variant of Equation 5-5)

$$\theta_p(\mu_k) = \theta_P^0 + \theta_P^1 K \mu_k \tag{5-6}$$

With K a scalar, equal to 1/8, which allows for a better scaling of the scheduling dependent parameters' gradients (see Appendix D-2). The other terms, i.e. θ_i and θ_d, are defined similarly. The scheduling sequence used for the ILPVFT algorithm and the A2S-ILPVFT's

reference experiment will be

$$\mu_k = 8 + 3\sin\left(\frac{2\pi}{N}k\right) \tag{5-7}$$

In the A2S-ILPVFT's gradient experiment it is set to (with ϕ_k and d_k two random numbers in the intervals [-0.1,0.5] and [-0.5,0], respectively)

$$\mu_k = 8 + 3\sin\left(\frac{2\pi}{N}k + \phi_k\right) + d_k \tag{5-8}$$

Other parameters which are relevant for the IFT framework are defined in Table 5-3. The factorization was kept the same as in Chapter 4. Four initial controllers were used as defined below (sorted by row as $\theta_P, \theta_I, \theta_D$)

$$\Theta_0^{fb,1} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \tag{5-9}$$

$$\Theta_0^{fb,2} = \begin{bmatrix} -3 & 0 \\ -4 & 0 \\ 1 & 0 \end{bmatrix} \tag{5-10}$$

$$\Theta_0^{fb,3} = \begin{bmatrix} 1.5 & 0.1 \\ 0.2 & -0.1 \\ 0 & 0 \end{bmatrix} \tag{5-11}$$

$$\Theta_0^{fb,4} = \begin{bmatrix} 1.5 & 0 \\ 0 & -1 \\ -1 & 1 \end{bmatrix} \tag{5-12}$$

The performance evolution per iteration is shown in Figure 5-4 for two initial controllers (for both algorithms). Figure 5-4 shows that the algorithms are capable of optimizing the LPV controllers in a fashion such that overall performance increase can be observed.
The non-smooth behaviour in the A2S-ILPVFT algorithms' performance evolution, Figure 5-4, was introduced due to the low order of the LPV-FIR filter. The low order approximation created problems which caused the gradient estimate to, in some iterations, become too large which subsequently could destabilize the algorithm. To filter out this problem a Gauss-Newton approximation of the Hessian was used per controller parameter as follows

$$(\theta_i^j)_{new} = (\theta_i^j)_{old} - \gamma \left(\left(\frac{\partial J}{\partial \theta_i^j} \right)^2 + 1 \right)^{-1} \frac{\partial J}{\partial \theta_i^j} \tag{5-13}$$

This update law prevented large steps whereas small steps are nearly unaffected (as the Hessian is then approximately 1).

Three of the four initial controller parameters showed a similar nature when comparing their optimal controller parameters (for the ILPVFT algorithm). The two least similar, per the Euclidean norm, optimal controller parameters are shown in Equation 5-14 and 5-15. Because they are not alike it is implied that the cost function exhibits a non-convex behaviour. The other two initial controller parameters gave results very similar to the optimal parameters

shown in Equation 5-14. The optimal controller parameters obtained by the A2S-ILPVFT algorithm show more dissimilarity between each other as seen in Equations 5-16 and 5-17. But interestingly Figure 5-4 indicates that both algorithms perform comparably well in terms of performance.

$$\Theta^{fb,4}_{ILPVFT} = \begin{bmatrix} -5.11109735263185 & -7.95010262164501 \\ -11.8006151598837 & -7.74308950227149 \\ -4.69744545968179 & -3.85602810050835 \end{bmatrix} \tag{5-14}$$

$$\Theta^{fb,2}_{ILPVFT} = \begin{bmatrix} -7.98371858658402 & -6.04379075838753 \\ -13.9406274629084 & -5.22520566037331 \\ -2.27334166957201 & -4.31601239432479 \end{bmatrix} \tag{5-15}$$

$$\Theta^{fb,4}_{A2S} = \begin{bmatrix} -9.63242421157763 & -2.15890532442709 \\ -13.662715507876 & -4.52094326744164 \\ -8.51924305931301 & -0.243746506361507 \end{bmatrix} \tag{5-16}$$

$$\Theta^{fb,2}_{A2S} = \begin{bmatrix} -11.6294361786087 & -1.43912142979108 \\ -16.4632911618522 & -1.37865655083784 \\ -5.16249047024916 & -1.12323217325337 \end{bmatrix} \tag{5-17}$$

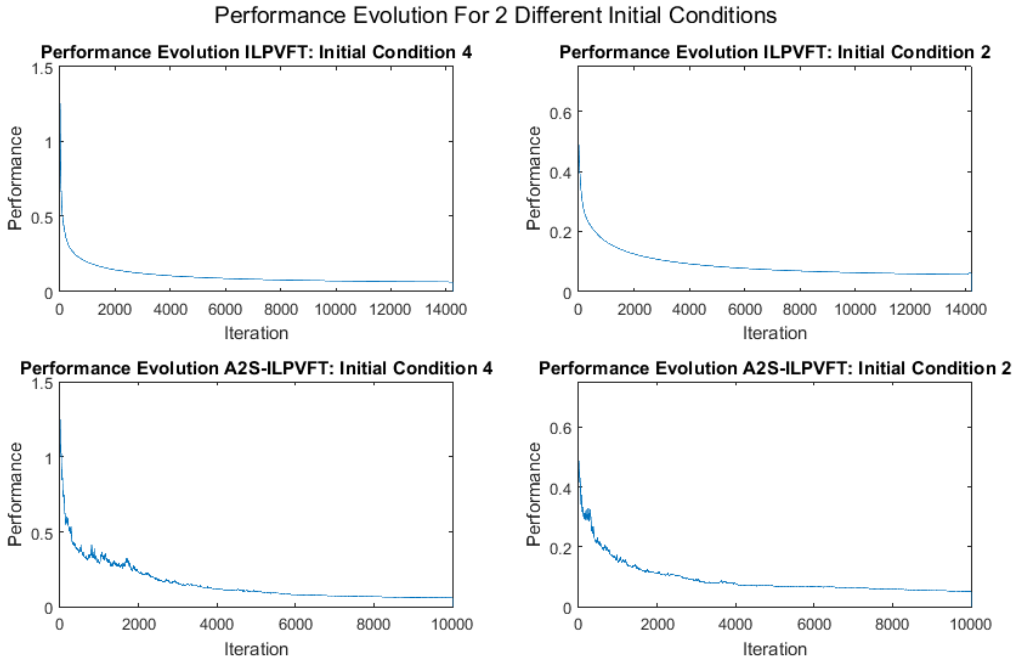

Figure 5-4: Performance Versus Iteration for two initial controllers, upper: ILPVFT and lower: A2S-ILPVFT.

The performance over the entire spectrum of wind speed for both the LPV and the LTI controllers are shown in Figures 5-5 and 5-6 for ILPVFT and A2S-ILPVFT, respectively. The author conjectures that altering the internal structure of the LPV controller could improve the optimal performance of the LPV controllers possibly making them viable. Unfortunately the

Variable	A2S – ILPVFT	ILPVFT
Amount of Measurements	200	200
Learning rate γ	1	0.2
Maximum number of iterations	10000	n/a
Minimum cost function decrease	n/a[1]	10^{-6}
Hessian Approximation	Gauss-Newton	Identity Matrix
Regularization	Identity matrix	n/a

Table 5-3: Optimization Parameters for the A2S-ILPVFT and ILPVFT algorithms of Section 5-3.

chosen structure of the LPV controllers is only capable of near LTI performance for $\approx 7-9.5$ m/s as elsewhere the LPV controller is not comparable to the gain scheduled LTI controllers. However an improvement is notable as instead of a quadratic degradation of the performance a slower decline is visible. Figures 5-5 and 5-6 also allow us to conclude that none of the LPV controllers are capable of stabilizing the unstable region (observe the rapid performance decrease when approaching 12 m/s).

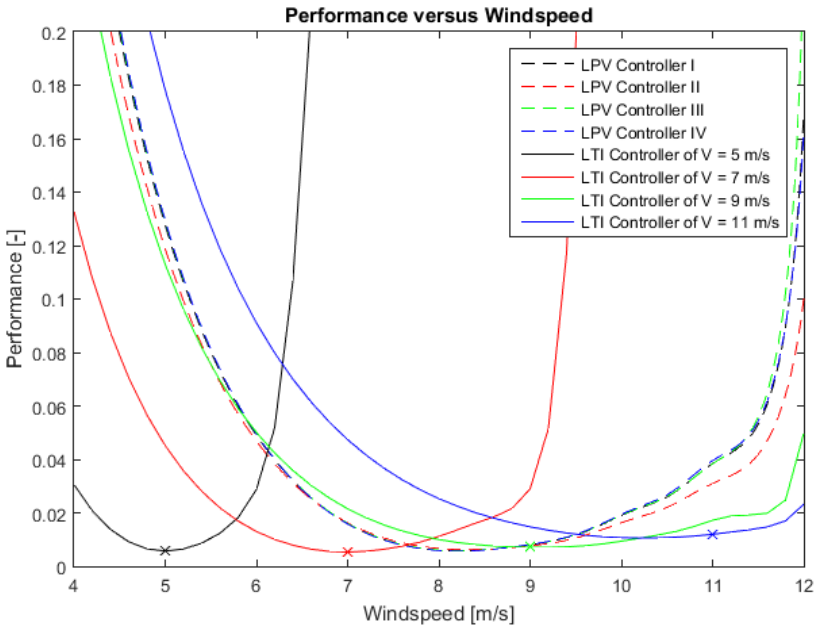

Figure 5-5: Performance versus Wind speed (5 to 11 m/s) for linear IFT & ILPVFT.

The numerical results indicate that the algorithms are both capable of tuning a LPV controller even for realistic LPV systems. The main problem which has arisen is the effect of the internal structure of the LPV controller. If this hurdle could be solved only then it might become possible for the LPV controller to acquire comparable performance to its gain scheduled LTI counterpart throughout the entire spectrum of wind speeds. Additionally it might make it possible to stabilize the unstable region which did not occur for this structure.

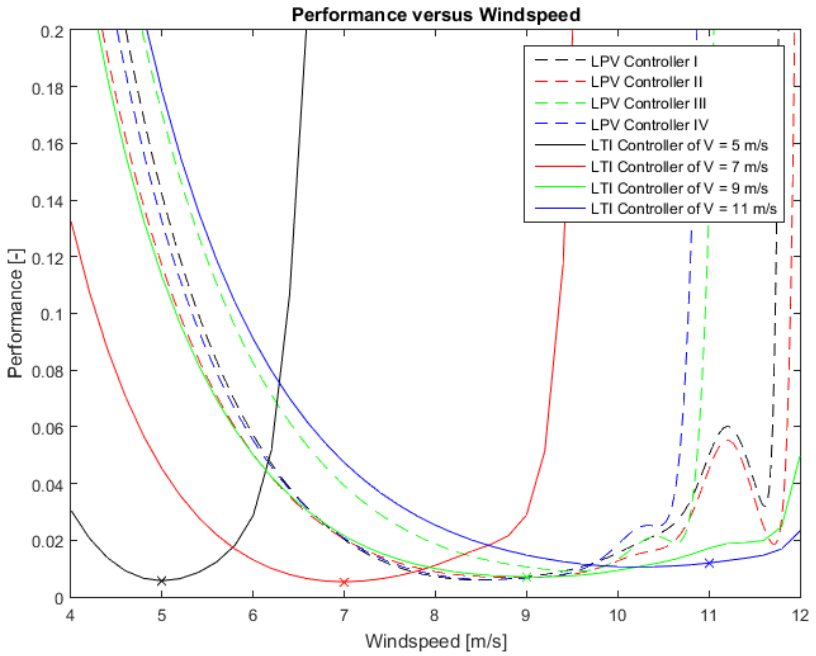

Figure 5-6: Performance versus Wind speed (5 to 12 m/s) for linear IFT & A2S-ILPVFT.

The combination of the numerical results (this chapter) and the mathematical derivations (Chapters 3 & 4) will now be used to see whether the goal set in Section 1-4 has been fulfilled.

Chapter 6

Conclusions & Recommendations

This thesis has presented the mathematical foundations for extending the applicability of the IFT framework to LPV control. Two algorithms were developed based on I/O LPV model structures. The first was based on a LPV-ARX model and the second on a lifted representation of the LPV-ARX system.

Both were shown to be capable of handling any scheduling sequence for general LPV systems (including MIMO LPV system and both degrees of controller freedom). However both suffered from distinct disadvantages. The ILPVFT algorithm (Chapter 3) needs a model of the plant to compensate for a change in scheduling sequence, whereas the A2S-ILPVFT algorithm (Chapter 4) needs a lot of experiments due to the curse of dimensionality.

The goal of the thesis was to extend the applicability of the IFT framework to LPV control. In addition the goal was to develop an algorithm which can handle arbitrarily fast scheduling without suffering from the curse of dimensionality. It can be concluded from the above paragraph that the ILPVFT algorithm succeeds in accomplishing all the aforementioned goals. The drawback is that the identification of a LPV model is required to compensate for a change in scheduling sequence.

Through a case study regarding the control of a smart airfoil a comparison of the optimal performance of LPV (novel algorithms) and gain scheduled LTI (linear IFT) controllers was made. The fixed LPV controller structure which was assumed was unable to fit the nonlinear relation underlying the controller parameters' relation with the scheduling variable. Because of this structural bias the optimal LPV controllers were only able to have performance comparable to the gain scheduled LTI controllers for a small subset of wind speeds. It can be concluded that the algorithms are capable of optimizing LPV controllers for realistic LPV systems. However the case study was inconclusive to whether an optimally structured LPV controller would have been able to reach comparable performance for the entire spectrum of wind speeds.

It can be concluded that the algorithms work as designed but without the optimization of the LPV structure the acquired optimal LPV controllers will, in general, not perform as well as gain scheduled LTI controllers. Especially the ILPVFT algorithm shows potential of being

a viable choice for extending the IFT framework to LPV control as it is able to handle any scheduling sequence without suffering from the curse of dimensionality. The author's personal recommendations will now be stated which could benefit the algorithms.

Recommendations

As seen in the previous chapters LPV control can be embedded into the IFT framework. Unfortunately either an accurate model (Chapter 3) or a large amount of experiments (Chapter 4) is needed. Therefore a more efficient integration of LPV control might be possible which suffers from neither problem. The following points of interest are subjects which, in the author's opinion, *could* make the algorithms (more) viable:

1. Performance Function: An improvement would be a performance function formulated as a convex optimization problem (for instance an LMI). This will guarantee global optimality and makes multi-start unnecessary.
2. Structural bias: The structure of the LPV controllers, in Chapter 5, was not based on any specific property of the system nor control objective. Incorporating methods to optimize the structure could increase performance. The author conjectures that the solution might be found through the combination of the ILPVFT algorithm and radial basis functions (which are capable of approximating any function [21]).
3. Constraints: Often in industrial applications certain safety requirements must be met, for example an overshoot of less than 5%. Currently the original framework has been shown to be able to incorporate constraints [22]. Extending this capability to LPV control can for instance guarantee stability throughout the optimization process. This was already shown to be possible for LTI systems [15]. The author conjectures that through the use of LMI's and the use of penalty functions (Appendix D-3) an efficient implementation can be achieved.
4. Scheduling Noise: The question remains whether scheduling noise would introduce a bias into the gradient estimate.
5. Efficient Factorization: A more efficient factorization method might make the algorithm of Chapter 4 viable.
6. Effects of model mismatch: Section 3-5 showed that by augmenting the gradient experiment an extension to arbitrary scheduling sequences is possible. However an unanswered question is how model mismatches affect the gradient estimate.

Appendix A

LPV Modelling Paradigms

The goal of this appendix is to introduce two LPV modelling paradigms which are used in this thesis.

LPV systems can be considered a set of gain scheduled linear systems. When the scheduling variable doesn't vary in time, i.e. $\mu_k = \mu$ for all k, then a LPV systems reduces to a LTI system. Therefore LTI systems can be considered a special case of LPV systems. Due to the inherent gain scheduled nature the class of LPV systems can be considered a class of nonlinear systems.

The first modelling paradigm which will be discussed is the LPV state space representation.

LPV State Space Representation

LPV state space matrices can be defined as

$$\mathcal{A}(\mu_k) = A_0 + \sum_{i=1}^{N_f} A_i \psi_i(\mu_k) \tag{A-1a}$$

In Equation A-1a the function $\psi_i(\mu_k)$ is a basis function. This basis function produces a scalar value for example; μ_k or $(\mu_k)^2$. These basis functions can be considered a set of functions which together form the gain scheduling underlying the LPV system. The total set is assumed to consist of N_f basis functions.

$$\mathcal{B}(\mu_k) = B_0 + \sum_{i=1}^{N_f} B_i \psi_i(\mu_k) \tag{A-1b}$$

The $\mathcal{C}(\mu_k)$ and $\mathcal{D}(\mu_k)$ matrices are defined in a similar fashion. These matrices can be used to construct a LPV state space system defined as

$$\mathbf{x}_{k+1} = \mathcal{A}(\mu_k)\mathbf{x}_k + \mathcal{B}(\mu_k)\mathbf{u}_k \tag{A-1c}$$

$$\mathbf{y}_k = \mathcal{C}(\mu_k)\mathbf{x}_k + \mathcal{D}(\mu_k)\mathbf{u}_k \tag{A-1d}$$

From Equation A-1d it can be observed that when $\mu_k = \mu$ the LPV system will reduce to a LTI system. This makes LPV state space systems an intuitive extension of LTI state space systems.

The second major type of LPV modelling paradigms is the LPV-ARX model structure which will be defined next.

The LPV-ARX Model

Another common way of modelling LPV systems is using an LPV-ARX model structure which can be defined as

$$y_k(\mu_k) = \sum_{i=1}^{n_a} -a_i(\mu_k)y_{k-i} + \sum_{j=0}^{n_b} b_j(\mu_k)u_{k-j} \tag{A-2a}$$

It can be observed that the LPV-ARX model reduces to a LTI-ARX model only when $\mu_k = \mu$ for all k. The variables b_j and a_i are defined as linear combinations

$$a_i(\mu_k) = a_i^0 + a_i^1\psi_1(\mu_k) + a_i^2\psi_2(\mu_k) + \ldots + a_i^{N_f}\psi_{N_f}(\mu_k) \tag{A-2b}$$

$$b_i(\mu_k) = b_i^0 + b_i^1\psi_1(\mu_k) + b_i^2\psi_2(\mu_k) + \ldots + b_i^{N_f}\psi_{N_f}(\mu_k) \tag{A-2c}$$

Equation A-2a can be condensed into the following shorthand notation which replaces the summation signs through the use of shift operators (i.e. q)

$$y_k(\mu_k) = \frac{B(q,\mu_k)}{A(q,\mu_k)}u_k \tag{A-2d}$$

Define $A(q,\mu_k)$ and $B(q,\mu_k)$ as

$$A(q,\mu_k) = 1 + a_1(\mu_k)q^{-1} + a_2(\mu_k)q^{-2} + \ldots + a_{n_a}(\mu_k)q^{-n_a} \tag{A-2e}$$

$$B(q,\mu_k) = b_0(\mu_k) + b_1(\mu_k)q^{-1} + b_2(\mu_k)q^{-2} + \ldots + b_{n_b}(\mu_k)q^{-n_b} \tag{A-2f}$$

Although this notation is used in for example [8] it should be pointed out that the shift operator becomes ambiguous as $a(\mu_k)y_k(\mu_k)q^{-1}$ could be solved as either; $a(\mu_k)y_{k-1}(\mu_{k-1})$ or $a(\mu_{k-1})y_k(\mu_k)$.

Throughout this thesis the shift operator will be modelled as only operating on the output $y_k(\mu_k)$:

$$a(\mu_k)y_k(\mu_k)q^{-1} = a(\mu_k)y_{k-1}(\mu_{k-1}) \tag{A-2g}$$

Another modelling approach encountered in the literature are quasi-LPV (qLPV) systems which approximate general nonlinear systems as LPV systems. This is done by setting the scheduling variable equal to a part of the state. An example of a qLPV system is a robotic manipulator in which the scheduling variable is set equal to the position [1]. For a more in-depth treatment of qLPV systems the reader is referred to [3].

Appendix B

Proofs: ILPVFT

This appendix contains proofs related to the ILPVFT algorithm of Chapter 3. These proof were omitted in the chapter itself to preserve clarity. In this appendix proofs are provided which show the gradient estimation, omitted in Section 3-3, and the extension of the algorithm to; MIMO systems, feedforward control, gradient experiments with a non-zero reference signal and related to the augmented gradient experiment (Section B-1 to B-5).

B-1 Gradient Estimation: Feedback Control

To estimate the gradient for the feedback controller the following derivative must be evaluated (Section 3-3)

$$\frac{\partial y_k(\Theta^{fb}, \mu_k)}{\partial \Theta^{fb}} = \frac{\partial}{\partial \Theta^{fb}} \left(\frac{B(q,\mu_k)\mathcal{H}(\mu_k, \Theta^{fb})}{A(q,\mu_k) + B(q,\mu_k)\mathcal{H}(\mu_k, \Theta^{fb})} \right) r_k \tag{B-1a}$$

This differentiation will be evaluated by using the quotient rule for derivatives. With Q_1 and Q_2 the to be calculated derivatives as per the quotient rule.

$$\frac{\partial}{\partial \Theta^{fb}} \frac{B(q,\mu_k)\mathcal{H}(\mu_k, \Theta^{fb})}{A(q,\mu_k) + B(q,\mu_k)\mathcal{H}(\mu_k, \Theta^{fb})} r_k = \frac{Q_1 - Q_2}{(A(q,\mu_k) + B(q,\mu_k)\mathcal{H}(\mu_k, \Theta^{fb}))^2} r_k \tag{B-1b}$$

The first term Q_1 is computed in Equation B-1d. In this case $B(q,\mu_k)$ acts as a scalar due to it not being dependent on Θ^{fb} which allows for the simplification

$$\frac{\partial \mathcal{H}(\mu_k, \Theta^{fb}) B(q,\mu_k)}{\partial \Theta^{fb}} = \frac{\partial \mathcal{H}(\mu_k, \Theta^{fb})}{\partial \Theta^{fb}} B(q,\mu_k) + \frac{\partial B(q,\mu_k)}{\partial \Theta^{fb}} \mathcal{H}(\mu_k, \Theta^{fb}) = \frac{\partial \mathcal{H}(\mu_k, \Theta^{fb})}{\partial \Theta^{fb}} B(q,\mu_k) \tag{B-1c}$$

Thus Q_1 can be written as

$$Q_1 = \frac{\partial B(q,\mu_k)\mathcal{H}(\mu_k, \Theta^{fb})}{\partial \Theta^{fb}} \left(A(q,\mu_k) + B(q,\mu_k)\mathcal{H}(\mu_k, \Theta^{fb}) \right) \tag{B-1d}$$

$$Q_1 = \frac{\partial \mathcal{H}(\mu_k, \Theta^{fb})}{\partial \Theta^{fb}} B(q, \mu_k) \left(A(q, \mu_k) + B(q, \mu_k)\mathcal{H}(\mu_k, \Theta^{fb}) \right) \tag{B-1e}$$

The second term Q_2 is equal to

$$Q_2 = \frac{\partial (A(q, \mu_k) + B(q, \mu_k)\mathcal{H}(\mu_k, \Theta^{fb}))}{\partial \Theta^{fb}} B(q, \mu_k)\mathcal{H}(\mu_k, \Theta^{fb}) \tag{B-1f}$$

By observing that $\frac{\partial A(q,\mu_k)}{\partial \Theta^{fb}} = 0$

$$Q_2 = \frac{\partial \mathcal{H}(\mu_k, \Theta^{fb})}{\partial \Theta^{fb}} B(q, \mu_k)\mathcal{H}(\mu_k, \Theta^{fb}) B(q, \mu_k) \tag{B-1g}$$

The terms of Equation B-1e and B-1g seem unrelated to any experiment. However one can acquire a dedicated experiment by filling them into Equation B-1b. The term Q_1 gives

$$\frac{Q_1}{(A(q, \mu_k) + B(q, \mu_k)\mathcal{H}(\mu_k, \Theta^{fb}))^2} r_k = \frac{\partial \mathcal{H}(\mu_k, \Theta^{fb})}{\partial \Theta^{fb}} \frac{B(q, \mu_k)(A(q, \mu_k) + B(q, \mu_k)\mathcal{H}(\mu_k, \Theta^{fb}))}{(A(q, \mu_k) + B(q, \mu_k)\mathcal{H}(\mu_k, \Theta^{fb}))^2} r_k \tag{B-1h}$$

$$\frac{Q_1}{(A(q, \mu_k) + B(q, \mu_k)\mathcal{H}(\mu_k, \Theta^{fb}))^2} r_k = \frac{\partial \mathcal{H}(\mu_k, \Theta^{fb})}{\partial \Theta^{fb}} \frac{B(q, \mu_k)}{A(q, \mu_k) + B(q, \mu_k)\mathcal{H}(\mu_k, \Theta^{fb})} r_k \tag{B-1i}$$

The second part involving Q_2 can be evaluated as

$$\frac{-Q_2}{(A(q, \mu_k) + B(q, \mu_k)\mathcal{H}(\mu_k, \Theta^{fb}))^2} r_k = -\frac{\partial \mathcal{H}(\mu_k, \Theta^{fb})}{\partial \Theta^{fb}} \frac{B(q, \mu_k)\mathcal{H}(\mu_k, \Theta^{fb}) B(q, \mu_k)}{(A(q, \mu_k) + B(q, \mu_k)\mathcal{H}(\mu_k, \Theta^{fb}))^2} r_k \tag{B-1j}$$

This equation can be rewritten, as per the original LTI derivation [10], through use of

$$y_k(\mu_k) = \frac{B(q, \mu_k)\mathcal{H}(\mu_k, \Theta^{fb})}{A(q, \mu_k) + B(q, \mu_k)\mathcal{H}(\mu_k, \Theta^{fb})} r_k \tag{B-1k}$$

This definition of $y_k(\mu_k)$ can be used to rewrite Equation B-1j to

$$\frac{-Q_2}{(A(q, \mu_k) + B(q, \mu_k)\mathcal{H}(\mu_k, \Theta^{fb}))^2} r_k = -\frac{\partial \mathcal{H}(\mu_k, \Theta^{fb})}{\partial \Theta^{fb}} \frac{B(q, \mu_k)}{A(q, \mu_k) + B(q, \mu_k)\mathcal{H}(\mu_k, \Theta^{fb})} y_k(\mu_k) \tag{B-1l}$$

Thus to estimate the gradient the following experiment must be performed

$$\frac{\partial y_k(\Theta^{fb}, \mu_k)}{\partial \Theta^{fb}} = \frac{\partial \mathcal{H}(\mu_k, \Theta^{fb})}{\partial \Theta^{fb}} \frac{B(q, \mu_k)}{A(q, \mu_k) + B(q, \mu_k)\mathcal{H}(\mu_k, \Theta^{fb})} (r_k - y_k(\mu_k)) \tag{B-1m}$$

B-2 Multi-Input Multi-Output Systems

Chapter 3 has shown that the ILPVFT algorithm doesn't suffer from the curse of dimensionality for SISO LPV systems. This section will generalize the result of Chapter 3 to LPV MIMO systems with any amount of controllers and sensors. This proof will show that a strategy present in the literature for LTI MIMO systems, see for example [23], also works for LPV MIMO systems. The general cost function for MIMO system is equal to

$$J(\Theta^{fb}) = \frac{1}{2N} \sum_{k=1}^{N} (Y_k - T_d Y_k^d)^T \mathbf{W}_k (Y_k - T_d Y_k^d) \tag{B-2a}$$

In Equation B-2a the column vectors Y_k are Y_k^d are stackings of all the acquired and desired outputs, respectively. The matrix $\mathbf{W}$ is a *positive* definite matrix which allows for the weighting of the different outputs at time step k. The general parameterization for the MIMO controller is shown below (for a system with two in- and outputs)

$$\mathcal{H}(\Theta^{fb}, \mu_k) = \begin{bmatrix} \mathcal{H}_{11}(\Theta_{11}^{fb}, \mu_k) & \mathcal{H}_{21}(\Theta_{21}^{fb}, \mu_k) \\ \mathcal{H}_{12}(\Theta_{12}^{fb}, \mu_k) & \mathcal{H}_{22}(\Theta_{22}^{fb}, \mu_k) \end{bmatrix} \tag{B-2b}$$

These controllers are, as seen in equation B-2b, all parameterized by their own controller coefficient parameter matrix Θ_{ij}^{fb}. The controller $\mathcal{H}_{ij}$ is the controller from output y_i to input u_j. The controller $\mathcal{H}_{ij}$ will be referred to as a controller block. This parameterization of the controller can be visualized using the block diagram of Figure B-1 (for simplicity the reference is not shown in this block diagram).

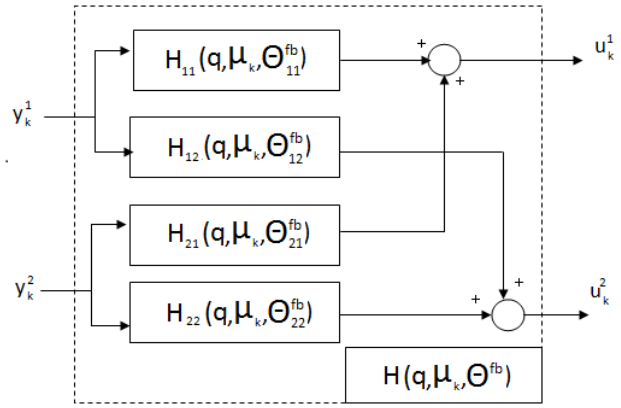

Figure B-1: Block diagram of the MIMO controller parameterization.

In [23] it was shown that, for LTI systems, *one* reference experiment is sufficient for both MIMO and SISO systems. The main difference is due to the amount of gradient experiment. For SISO systems one gradient experiment must be performed to acquire the gradient w.r.t. every controller parameter whereas MIMO systems need one gradient experiment per controller block [10][23]. This result for LTI MIMO systems will be extended to LPV systems. The MIMO LPV system will be modelled as (for a plant with two inputs)

$$Y_k = \begin{bmatrix} G_{11} & G_{12} & G_1^{ff} \\ G_{21} & G_{22} & G_2^{ff} \end{bmatrix} \begin{bmatrix} u_k^1(\mu_k) \\ u_k^2(\mu_k) \\ u_k^{ff} \end{bmatrix} = \mathcal{G}(\mu_k, q) \begin{bmatrix} U_k(\mu_k) \\ u_k^{ff} \end{bmatrix} \tag{B-2c}$$

$$U_k(\mu_k) = \mathcal{H}(\mu_k, \Theta^{fb})(R_k - Y_k) \tag{B-2d}$$

The parameterization of the MIMO controller into independently parameterized controller blocks (Figure B-1) allows us to transform the LPV system into a "new" plant for an arbitrary controller block $\mathcal{H}_{ij}$

$$y_k^i = \hat{\mathcal{G}}(\mu_k, q) \begin{bmatrix} u_k^j(\mu_k) \\ u_k^{ff,ij} \end{bmatrix} \tag{B-2e}$$

$$u_k^j(\mu_k) = \mathcal{H}_{ij}(\mu_k, \Theta_{ij}^{fb})(r_k^{ij} - y_k^i) \tag{B-2f}$$

The "new" plant $\hat{\mathcal{G}}(\mu_k, q)$ is equal to the interconnection of the plant $\mathcal{G}(\mu_k, q)$ and all controller blocks *except* the controller block $\mathcal{H}_{ij}$. This transforms the LPV MIMO gradient estimation problem to an *equivalent* LPV SISO gradient estimation problem for controller block $\mathcal{H}_{ij}$.

Therefore to obtain an estimate of the gradient for controller block $\mathcal{H}_{ij}$ one must perform the gradient experiment with all references set to zero and $u_k^{ff,ij} = r_k^i - y_k^i$. Afterwards to obtain the gradient of every controller parameter in the controller block matrix Θ_{ij}^{fb} one merely needs to perform the appropriate filtering. Therefore to acquire the gradient for every controller block one must perform the gradient experiment (of Equation B-2e) per controller block.

So the amount of experiments needed increases to $1 + N_{cb}$ with N_{cb} the amount of controller blocks. The ILPVFT algorithm therefore shows a *linear* nature in terms of experiments for MIMO systems.

The A2S-ILPVFT algorithm of Chapter 4 shows a slightly different nature. If the condition of Equation 4-2n is fulfilled then one has N_f^{ij} gradient experiments extra for controller block ij (with N_f^{ij} the amount of different basis functions). When this condition is not fulfilled the amount of extra experiments for controller block ij is equal to N_c^{ij} (with N_c^{ij} the amount of controller parameters).

B-3 LPV Feedforward Control

The IFT framework can also be used to tune feedforward controllers as shown in [24]. This section will show that LPV feedforward control is also possible albeit with a slight twist. The block diagram of the feedforward control problem is shown in Figure B-2.

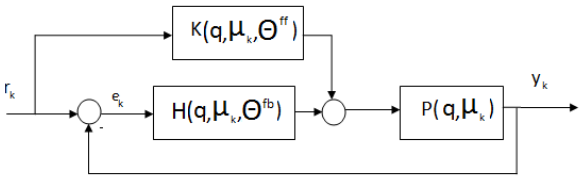

Figure B-2: Block diagram for the two degree of freedom LPV control plant.

This section will be focused solely on optimizing the LPV feedforward controller

$$u_k^{ff} = \mathcal{K}(\mu_k, \Theta^{ff}) r_k \tag{B-3a}$$

The feedforward control law $\mathcal{K}(\mu, \Theta^{ff})$ will be defined as

$$\mathcal{K}(q, \mu_k) = \left(\theta_0^{ff}(\mu_k) + \theta_1^{ff}(\mu_k) q^{-1} + \theta_2^{ff}(\mu_k) q^{-2} + \ldots + \theta_{n_K}^{ff}(\mu_k) q^{-n_K}\right) r_k \tag{B-3b}$$

In regressor representation it can be defined as

$$\mathcal{K}(q, \mu_k) =< \Theta^{ff}, \Phi > r_k \tag{B-3c}$$

The purpose of the feedforward controller is improving tracking performance therefore its cost function can be defined as

$$J(\Theta^{ff}) = \frac{1}{2N}\sum_{i=1}^{N}\Big(r_i - y_i(\Theta^{ff},\mu_i)\Big)^2 = \frac{1}{2N}\sum_{i=1}^{N}\Big(e_i(\Theta^{ff},\mu_i)\Big)^2 \tag{B-3d}$$

An important equation is the relation between the tracking error e_k and the reference r_k

$$e_k(\Theta^{ff},\mu_k) = \frac{1-P(q,\mu_k)\mathcal{K}(\mu_k,\Theta^{ff})}{1-P(q,\mu_k)\mathcal{H}(q,\mu_k)}r_k \tag{B-3e}$$

The gradient of Equation B-3d is equal to

$$\frac{\partial J(\Theta^{ff})}{\partial \Theta^{ff}} = \frac{1}{N}\sum_{i=1}^{N} e_i(\Theta^{ff},\mu_k)\frac{\partial e_i(\Theta^{ff},\mu_k)}{\partial \Theta^{ff}} \tag{B-3f}$$

Thus we need a way to estimate $\frac{\partial e(\Theta^{ff},\mu_k)}{\partial \Theta^{ff}}$. For this estimation let's use Equation B-3e and use the quotient rule

$$\frac{\partial e_k(\Theta^{ff},\mu_k)}{\partial \Theta^{ff}} = \frac{Q_1 - Q_2}{(1-P(q,\mu_k)\mathcal{H}(q,\mu_k))^2}r_k \tag{B-3g}$$

The term Q_2 drops out, but the term Q_1 will not be equal to zero

$$Q_2 = \frac{\partial\left(1-P(q,\mu_k)\mathcal{H}(q,\mu_k)\right)}{\partial\Theta^{ff}}\Big(1-P(q,\mu_k)\mathcal{K}(\mu_k,\Theta^{ff})\Big) = 0 \tag{B-3h}$$

$$Q_1 = \frac{\partial\Big(1-P(q,\mu_k)\mathcal{K}(\mu_k,\Theta^{ff})\Big)}{\partial\Theta^{ff}}\left(1-P(q,\mu_k)\mathcal{H}(q,\mu_k)\right) \tag{B-3i}$$

$$Q_1 = -\frac{\partial\mathcal{K}(\mu_k,\Theta^{ff})}{\partial\Theta^{ff}}P(q,\mu_k)(1-P(q,\mu_k)\mathcal{H}(q,\mu_k)) \tag{B-3j}$$

Filling this into Equation B-3g and simplifying gives

$$\frac{\partial e_k(\Theta^{ff},\mu_k)}{\partial\Theta^{ff}} = -\frac{\partial\mathcal{K}(\mu_k,\Theta^{ff})}{\partial\Theta^{ff}}\frac{P(q,\mu_k)}{1-P(q,\mu_k)\mathcal{H}(q,\mu_k)}r_k \tag{B-3k}$$

This result implies that a second experiment is necessary. However by multiplying with the, known, term $\frac{\mathcal{H}(q,\mu_k)+\mathcal{K}(\mu_k,\Theta^{ff})}{\mathcal{H}(q,\mu_k)+\mathcal{K}(\mu_k,\Theta^{ff})} = 1$ as proposed in [24] leads to

$$\frac{\partial e_k(\Theta^{ff},\mu)}{\partial\Theta^{ff}} = -\frac{\partial\mathcal{K}(\mu_k,\Theta^{ff})}{\partial\Theta^{ff}}\frac{\mathcal{H}(q,\mu_k)+\mathcal{K}(\mu_k,\Theta^{ff})}{\mathcal{H}(q,\mu_k)+\mathcal{K}(\mu_k,\Theta^{ff})}\frac{P(q,\mu_k)}{1-P(q,\mu_k)\mathcal{H}(q,\mu_k)}r_k \tag{B-3l}$$

$$\frac{\partial e_k(\Theta^{ff},\mu_k)}{\partial\Theta^{ff}} = -\frac{\partial\mathcal{K}(\mu_k,\Theta^{ff})}{\partial\Theta^{ff}}\frac{1}{\mathcal{H}(q,\mu_k)+\mathcal{K}(\mu_k,\Theta^{ff})}\frac{P(q,\mu_k)(\mathcal{H}(q,\mu_k)+\mathcal{K}(\mu_k,\Theta^{ff}))}{1-P(q,\mu_k)\mathcal{H}(q,\mu_k)}r_k \tag{B-3m}$$

Through use of the closed loop system, Equation B-3n, it can be observed that Equation B-3m can be rewritten as a mixture of I/O data and a scaling as shown in Equation B-3o

$$y_k(\Theta^{ff},\mu_k) = \frac{P(q,\mu_k)(\mathcal{H}(q,\mu_k)+\mathcal{K}(q,\mu_k,\Theta^{ff}))}{1-P(q,\mu_k)\mathcal{H}(q,\mu_k)}r_k \tag{B-3n}$$

$$\frac{\partial e_k(\Theta^{ff},\mu_k)}{\partial \Theta^{ff}} = -\frac{\partial \mathcal{K}(\mu_k,\Theta^{ff})}{\partial \Theta^{ff}} \frac{1}{\mathcal{H}(q,\mu_k)+\mathcal{K}(\mu_k,\Theta^{ff})} y_k(\Theta^{ff},\mu_k) \tag{B-3o}$$

This can be rewritten using the extended regressor representation and $\frac{\partial \text{trace}(X^T B)}{\partial X} = B$

$$-\frac{\partial e_k(\Theta^{ff},\mu_k)}{\partial \Theta^{ff}} = \frac{<\Theta^{ff},\Phi_k>}{\partial \Theta^{ff}} \frac{1}{\mathcal{H}(q,\mu_k)+\mathcal{K}(\mu_k,\Theta^{ff})} y_k(\Theta^{ff},\mu_k) = \Phi\left(\frac{1}{\mathcal{H}(q,\mu_k)+\mathcal{K}(\mu_k,\Theta^{ff})} y_k(\Theta^{ff},\mu_k)\right) \tag{B-3p}$$

Therefore only the reference experiment must be performed to estimate the gradient

$$\frac{\partial J(\Theta^{ff})}{\partial \Theta^{ff}} = e_k(\Theta^{ff},\mu_k)\Phi\left(\frac{1}{\mathcal{H}(q,\mu_k)+\mathcal{K}(\mu_k,\Theta^{ff})} y_k(\Theta^{ff},\mu_k)\right) \tag{B-3q}$$

B-4 Gradient Experiment with $r_k^g \neq 0$

The results of Chapter 3 show that two experiments must be performed with the gradient experiment equal to

$$y_k^g(\Theta^{fb},\mu_k) = \frac{B(q,\mu_k)}{A(q,\mu_k)+B(q,\mu_k)\mathcal{H}(\mu_k,\Theta^{fb})}(r_k - y_k(\mu_k)) \tag{B-4a}$$

It might not be possible to perform the gradient experiment with the reference set to zero. In that case the gradient experiment which is performed is equal to

$$\tilde{y}_k^g(\Theta^{fb},\mu_k) = \frac{B(q,\mu_k)}{A(q,\mu_k)+B(q,\mu_k)\mathcal{H}(\mu_k,\Theta^{fb})}(r_k - y_k(\mu_k)) + \frac{B(q,\mu_k)\mathcal{H}(\mu_k,\Theta^{fb})}{A(q,\mu_k)+B(q,\mu_k)\mathcal{H}(\mu_k,\Theta^{fb})} r_k^g \tag{B-4b}$$

This experiment can be considered a "corrupted" gradient experiment and can be rewritten to

$$\tilde{y}_k^g(\Theta^{fb},\mu_k) = y_k^g(\Theta^{fb},\mu_k) + \frac{B(q,\mu_k)\mathcal{H}(\mu_k,\Theta^{fb})}{A(q,\mu_k)+B(q,\mu_k)\mathcal{H}(\mu_k,\Theta^{fb})} r_k^g \tag{B-4c}$$

If the equality $r_k^g = r_k$ holds then the "corruption" is equal to the reference experiment (see Chapter 3). One might then be tempted to use the equality

$$y_k^g(\Theta^{fb},\mu_k) = \tilde{y}_k^g(\Theta^{fb},\mu_k) - y_k(\Theta^{fb},\mu_k) \tag{B-4d}$$

In a deterministic system this would be an appropriate compensation. Unfortunately in a stochastic system one would acquire (with v_k^g and v_k^r the *output* noise in the gradient and reference experiment respectively)

$$\hat{y}_k^g(\Theta^{fb},\mu_k) = \tilde{y}_k^g(\Theta^{fb},\mu_k) + v_k^g - y_k(\Theta^{fb},\mu_k) - v_k^r \tag{B-4e}$$

This will introduce a bias into the gradient estimate as shown below (the bias test is based on [10])

$$\text{E}\left[\frac{\partial \hat{J}(\Theta^{fb})}{\partial \Theta^{fb}}\right] = \frac{1}{N}\left(\text{E}\left[\sum_{i=1}^{N}(y_i(\Theta^{fb},\mu_i) - y_i^d)\frac{\partial y_i(\Theta^{fb},\mu_k)}{\partial \Theta^{fb}}\right] + \text{E}\left[\sum_{i=1}^{N}(y_i(\Theta^{fb},\mu_i) - y_i^d)\delta_i(\Theta^{fb})\right]\right) \tag{B-4f}$$

Which when considering that v_k^r is now also present in the gradient experiment's noise, per Equation B-4f, we acquire (with $q_v \neq 0$)

$$\mathrm{E}\left[\frac{\partial \hat{J}(\Theta^{fb})}{\partial \Theta^{fb}}\right] = \frac{\partial J(\Theta^{fb})}{\partial \Theta^{fb}} + q_v \tag{B-4g}$$

Therefore the "simple" solution, of Equation B-4d, makes the gradient estimate biased. The proper solution (i.e. which preserves unbiasedness) to this problem is performing a third experiment, namely;

$$y_k^t(\Theta^{fb}, \mu_k) = \frac{B(q,\mu_k)\mathcal{H}(\mu_k, \Theta^{fb})}{A(q,\mu_k) + B(q,\mu_k)\mathcal{H}(\mu_k, \Theta^{fb})} r_k^g \tag{B-4h}$$

Then by filling in Equation B-4e as before (but now with v_k^t the third experiments' noise)

$$y_k^g(\Theta^{fb}, \mu_k) = \tilde{y}_k^g(\Theta^{fb}, \mu_k) + v_k^g - y_k^t(\Theta^{fb}, \mu_k) - v_k^t \tag{B-4i}$$

Now if one assumes that the noise is not correlated between the three experiments then the unbiasedness property returns. This can be seen by again using equation B-4f as the starting point but now with no correlation between the gradient estimate and the reference experiment

$$\mathrm{E}\left[\frac{\partial \hat{J}(\Theta^{fb})}{\partial \Theta^{fb}}\right] = \frac{\partial J(\Theta^{fb})}{\partial \Theta^{fb}} + \frac{1}{N}\mathrm{E}\left[\sum_{i=1}^{N}(y_i(\Theta^{fb}, \mu_i) - y_i^d)\right]\mathrm{E}\left[\delta_i(\Theta^{fb})\right] \tag{B-4j}$$

Which if one assumes zero mean white noise gives

$$\mathrm{E}\left[\frac{\partial \hat{J}(\Theta^{fb})}{\partial \Theta^{fb}}\right] = \frac{\partial J(\Theta^{fb})}{\partial \Theta^{fb}} + 0 \tag{B-4k}$$

Therefore the conclusion which can be drawn is;
If the gradient experiment needs to be performed with a reference unequal to zero. Then to preserve unbiasedness a third experiment needs to performed.

B-5 Model-Based Compensation: Stochastic Properties & MIMO Systems

Noisy Gradient Estimates

A key advantage of the IFT framework (and the ILPVFT extension) is the unbiased nature of the gradient estimate. This section will provide a proof which will show that the unbiased nature is preserved for the augmented gradient experiment. The augmented gradient experiment with output noise can be defined as

$$\hat{y}_k^g(\mu_k^g) = \sum_{i=1}^{N_p} p_i(\mu_k^g) y_{k-i}^g - \sum_{n=0}^{N_c} c_j(\mu_k^g) v_{k-j} + \sum_{j=0}^{N_b} b_j(\mu_k^g) u_{k-j}^{ff} \tag{B-5a}$$

Note the addition of the, zero mean, white noise term v_k. The term $c_j(\mu^g)$ represents the controller gains with $c_0(\mu^g) = -1$ (which models $\hat{y}_k = y_k + v_k$). The feedforward signal u_k^{ff} is equal to

$$u_k^{ff} = g_k + u_k^c + f_k^c \tag{B-5b}$$

The three aforementioned individual signals are defined as in Equation B-5c. The noise of the reference experiment v_k^r is assumed to be independent from the noise in the gradient experiment v_k.

$$g_k = r_k - y_k - v_k^r \tag{B-5c}$$

$$u_k^c = \frac{1}{b_0(\mu_k^g)} \left(\sum_{j=0}^{N_b} \left(b_j(\mu_k^r) - b_j(\mu_k^g) \right) g_{k-j} - \sum_{n=1}^{N_b} b_j(\mu_k^g) u_{k-n}^c \right) \tag{B-5d}$$

$$f_k^c = \frac{1}{b_0(\mu_k^g)} \left(\sum_{i=1}^{N_p} \left(p_i(\mu_k^r) - p_i(\mu_k^g) \right) \hat{y}_{k-i}^r - \sum_{n=1}^{N_b} b_j(\mu_k^g) f_{k-n}^c \right) \tag{B-5e}$$

In the preceding equations (B-5a to B-5e) it is implied that one can ignore g_k and u_k^c to draw conclusions w.r.t. the bias as those signals are not dependent on v_k. This conjecture will now be verified in a more formal setting.

The initial conditions are set equal to

$$y_0^g = y_0^r = 0 \tag{B-6a}$$

Evaluation for the first time step $k = 1$ gives

$$\hat{y}_1^g(\mu_1^g) = p_1(\mu_1^g) y_0^g - c_0(\mu_1^g) v_1 + b_0(\mu_1^r) g_1 + b_0(\mu_1^g) f_1^c \tag{B-6b}$$

This can be rewritten by using Equation B-5a and the notion that $c_0(\mu_k) = -1$ and $f_1^c = 0$ (see Section 3-5)

$$\hat{y}_1^g(\mu_1^g) = v_1 + b_0(\mu_1^r) g_1 \tag{B-6c}$$

The correct *stochastic* output is equal to

$$\hat{y}_1^r(\mu_1^r) = v_1 + b_0(\mu_1^r) g_1 \tag{B-6d}$$

One can therefore conclude that the outputs are equal for $k = 1$. However it will now be shown that this doesn't hold for $k \geq 2$

$$\hat{y}_2^g(\mu_2^g) = p_1(\mu_2^g) y_1^g - c_0^g(\mu_2^g) v_2 - c_1^g(\mu_2^g) v_1 + b_0(\mu_2^r) g_1 + b_0(\mu_2^g) f_2^c \tag{B-6e}$$

$$\hat{y}_2^g(\mu_2^g) = p_1(\mu_2^r) y_1^r + v_2 - c_1^g(\mu_2^g) v_1 + b_0(\mu_2^r) g_1 + \left(p_1(\mu_2^r) - p_1(\mu_2^g) \right) v_1 \tag{B-6f}$$

$$\hat{y}_2^r(\mu_2^r) = p_1(\mu_2^r) y_1^r - c_1^g(\mu_1^r) v_1 + v_2 + b_0(\mu_2^r) g_2 + b_1(\mu_2^r) g_1 \tag{B-6g}$$

It is easy to see that $\hat{y}_2^r(\mu_2^r) \neq \hat{y}_2^g(\mu_2^g)$. Although one can make the observation that $\hat{y}_2^g(\mu_2^g)$ is a noisy estimate of $\hat{y}_2^r(\mu_2^r)$. The unbiased nature will be proven by showing that the expected values of the compensated and real gradient experiment are equal

$$\mathrm{E}\left[\hat{y}_2^r(\mu_2^r)\right] = \mathrm{E}\left[p_1(\mu_2^r) y_1^r - c_1^g(\mu_1^r) v_1 + v_2 + b_0(\mu_2^r) g_2 + b_1(\mu_2^r) g_1\right] \tag{B-6h}$$

Through use of the additive property of the expected value: $\mathrm{E}[x+y] = \mathrm{E}[x] + \mathrm{E}[y]$

$$\mathrm{E}\left[\hat{y}_2^r(\mu_2^r)\right] = \mathrm{E}\left[p_1(\mu_2^r)y_1^r + b_0(\mu_2^r)g_2 + b_1(\mu_2^r)g_1\right] + \mathrm{E}\left[-c_1^g(\mu_1^r)v_1 + v_2\right] \tag{B-6i}$$

Assuming that the scheduling variable and the output noise are not correlated (i.e. $\mathrm{E}[xy] = \mathrm{E}[x]\mathrm{E}[y]$) and usage of the definition of y_2^r leads to

$$\mathrm{E}\left[\hat{y}_2^r(\mu_2^r)\right] = y_2^r(\mu_2^r) + \mathrm{E}\left[-c_1^g(\mu_1^r)\right]\mathrm{E}[v_1] + \mathrm{E}[v_2] \tag{B-6j}$$

By the assumption that v_k is zero mean white noise

$$\mathrm{E}\left[\hat{y}_2^r(\mu_2^r)\right] = y_2^r(\mu_2^r) + 0 \tag{B-6k}$$

For the compensated gradient experiment it can be written as

$$\mathrm{E}\left[\hat{y}_2^g(\mu_2^g)\right] = \mathrm{E}\left[p_1(\mu_2^r)y_1^r + v_2 - c_1^g(\mu_2^g)v_1 + b_0(\mu_2^r)g_1 + \left(p_1(\mu_2^r) - p_1(\mu_2^g)\right)v_1\right] \tag{B-6l}$$

This can be rewritten by again using the additive property of the expected value function

$$\mathrm{E}\left[\hat{y}_2^g(\mu_2^g)\right] = \mathrm{E}\left[p_1(\mu_2^r)y_1^r + b_0(\mu_2^r)g_2 + b_1(\mu_2^r)g_1\right] + \mathrm{E}\left[-c_1^g(\mu_2^r)v_1 + v_2 + \left(p_1(\mu_2^r) - p_1(\mu_2^g)\right)v_1\right] \tag{B-6m}$$

The first important observation is that the first term is the definition of y_2^r (see Section 3-5). Then under the assumption that the scheduling variable and output noise are uncorrelated

$$\mathrm{E}\left[\hat{y}_2^g(\mu_2^g)\right] = y_2^r(\mu_2^r) + \mathrm{E}\left[-c_1^g(\mu_1^r) - p_1(\mu_2^g)\right]\mathrm{E}[v_1] + \mathrm{E}[v_2] \tag{B-6n}$$

By the assumption that v_k is zero mean white noise gives

$$\mathrm{E}\left[\hat{y}_2^g(\mu_2^g)\right] = y_2^r(\mu_2^r) + 0 \tag{B-6o}$$

Therefore the expected values of both signals are equal. So the unbiasedness property is also present for the augmented gradient experiment. This can be generalized to an arbitrary time step by noting that by the definition of f_k^c one never multiplies two noise terms but only scale them with terms associated with the dynamics.

For practical applications it should be noted that the noise does increase in magnitude (compare Equation B-6f and B-6g). This can be taken into account by either increasing the SNR and/or the amount of measurements. A numerical verification is provided in Figure B-3 which shows that, for the same LPV system and optimization parameters as in Section 3-5, convergence to the optimal values is again visible (for a zero mean white noise with a variance of $(0.2)^2$).

Multi-input Multi-Output Systems

For the augmented gradient experiment one needs to compensate the LPV system dynamics. In a SISO system this means adapting through the only input the system has, but in a MIMO system compensation of every output must be achieved through a set of inputs. Deriving an expression for the compensation signal for LPV MIMO systems is the subject of this section.

The LPV-ARX model structure will be used for deriving an expression of the two compensating signal *vectors* (i.e. U_k^c and F_k^c). The MIMO LPV-ARX model for the *desired* gradient

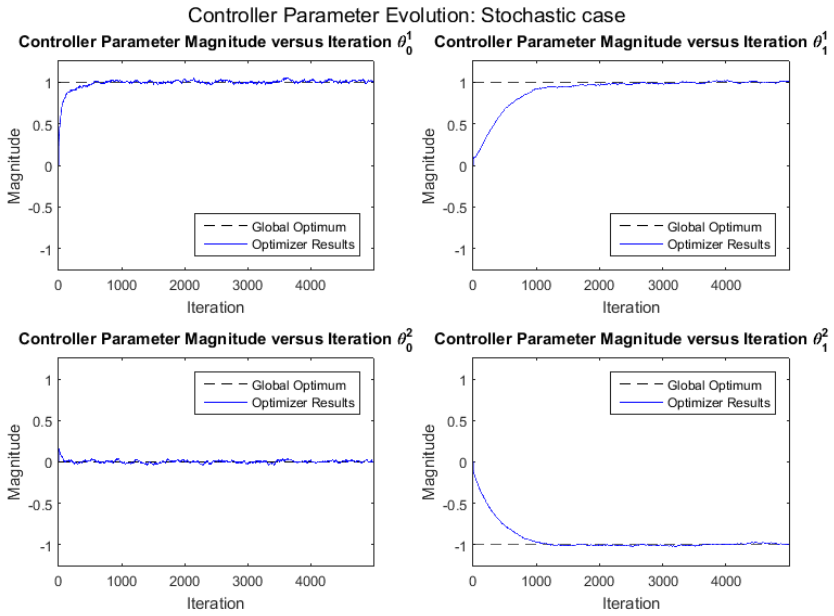

Figure B-3: Evolution of the controller parameters per iteration for the compensated *stochastic* gradient experiment.

experiment is shown below in Equation B-7a with n_y outputs and n_u inputs. In Equation B-7a the notation is as follows; bold capital letters are matrices, non-bold capital letters are vectors. The sizes of the vectors and matrices are; $Y_k^{r,n} \in \mathbb{R}^{n_y}$ and $G_k^n \in \mathbb{R}^{n_u}$, $\mathbf{B}_i(\mu_k^r)$ and $\mathbf{P}_j(\mu_k^r)$ are respectively n_y-by-n_u and n_y-by-n_y matrices.

$$Y_k^{r,n}(\mu_k^r) = \sum_{j=1}^{N_p} \mathbf{P}_j(\mu_k^r) Y_{k-j}^{r,n} + \sum_{i=0}^{N_b} \mathbf{B}_i(\mu_k^r) G_{k-i}^n \tag{B-7a}$$

The superscript n indicates the controller block n whose gradient experiment is performed (see Appendix B-2). It will be assumed that the system is fully actuated and $\mathbf{B}_0(\mu_k^r)$ (i.e. $n_y = n_u$) is invertible for all μ_k. The signal G_k^n is defined as a zero vector except for the n'th row which is defined as

$$\{G_k^n\}_n = r_k^n - y_k^n \tag{B-7b}$$

The superscripts in the above definition refers to the n'th controller block's reference and output signal of the reference experiment. The compensated gradient experiment will be defined as

$$Y_k^{g,n}(\mu_k^g) = \sum_{j=1}^{N_p} \mathbf{P}_j(\mu_k^g) Y_{k-j}^{g,n} + \sum_{i=0}^{N_b} \mathbf{B}_i(\mu_k^g) U_{k-i}^{ff,n} \tag{B-7c}$$

The compensation signal $U_k^{ff,n}$ is a superposition of three signals (of which G_k^n was already defined in Equation B-7b)

$$U_k^{ff,n} = G_k^n + U_k^{c,n} + F_k^{c,n} \tag{B-7d}$$

A method similar to the derivation of the compensation signal in Section 3-5 will be used,

therefore let's start with $U_k^{c,n}$

$$\sum_{i=0}^{N_b} \mathbf{B}_i(\mu_k^r) G_{k-i}^n = \sum_{i=0}^{N_b} \mathbf{B}_i(\mu_k^g) \left(G_{k-i}^n + U_{k-i}^{c,n} \right) \tag{B-7e}$$

Solving for $U_k^{c,n}$ gives

$$\mathbf{B}_0(\mu_k^g) U_k^{c,n} = \sum_{i=0}^{N_b} \left(\mathbf{B}_i(\mu_k^r) - \mathbf{B}_i(\mu_k^g) \right) G_{k-i}^n - \sum_{i=1}^{N_b} \mathbf{B}_i(\mu_k^g) U_{k-i}^{c,n} \tag{B-7f}$$

$$U_k^{c,n} = \mathbf{B}_0^{-1}(\mu_k^g) \left(\sum_{i=0}^{N_b} \left(\mathbf{B}_i(\mu_k^r) - \mathbf{B}_i(\mu_k^g) \right) G_{k-i}^n - \sum_{i=1}^{N_b} \mathbf{B}_i(\mu_k^g) U_{k-i}^{c,n} \right) \tag{B-7g}$$

Equation B-7g shows why $\mathbf{B}_0(\mu_k^g)$ was assumed to be invertible. By introducing the definition of $U_k^{c,n}$ in the compensated gradient experiment one acquires

$$Y_k^{g,n}(\mu_k^g) = \sum_{j=1}^{N_p} \mathbf{P}_j(\mu_k^g) Y_{k-j}^{g,n} + \sum_{i=0}^{N_b} \left(\mathbf{B}_i(\mu_k^r) G_{k-i}^n + \mathbf{B}_i(\mu_k^g) F_{k-i}^{c,n} \right) \tag{B-7h}$$

The following equality should hold to ensure proper compensation

$$Y_k^{r,n}(\mu_k^r) = Y_k^{g,n}(\mu_k^g) \tag{B-7i}$$

$$\sum_{j=1}^{N_p} \mathbf{P}_j(\mu_k^r) Y_{k-j}^{r,n} + \sum_{i=0}^{N_b} \mathbf{B}_i(\mu_k^r) G_{k-i}^n = \sum_{j=1}^{N_p} \mathbf{P}_j(\mu_k^g) Y_{k-j}^{g,n} + \sum_{i=0}^{N_b} \left(\mathbf{B}_i(\mu_k^r) G_{k-i}^n + \mathbf{B}_i(\mu_k^g) F_{k-i}^{c,n} \right) \tag{B-7j}$$

$$\sum_{j=1}^{N_p} \mathbf{P}_j(\mu_k^r) Y_{k-j}^{r,n} = \sum_{j=1}^{N_p} \mathbf{P}_j(\mu_k^g) Y_{k-j}^{g,n} + \sum_{i=0}^{N_b} \mathbf{B}_i(\mu_k^g) F_{k-i}^{c,n} \tag{B-7k}$$

The results of Section 3-5 for LPV SISO systems imply that the following equality will hold; $Y_{k-j}^{g,n} = Y_{k-j}^{r,n}$ for $j \geq 1$. Therefore the problem can be written as follows (while preserving generality)

$$\sum_{j=1}^{N_p} \mathbf{P}_j(\mu_k^r) Y_{k-j}^{r,n} = \sum_{j=1}^{N_p} \mathbf{P}_j(\mu_k^g) Y_{k-j}^{r,n} + \sum_{i=0}^{N_b} \mathbf{B}_i(\mu_k^g) F_{k-i}^{c,n} \tag{B-7l}$$

$$\mathbf{B}_0(\mu_k^g) F_k^{c,n} = \sum_{j=1}^{N_p} \left(\mathbf{P}_j(\mu_k^r) - \mathbf{P}_j(\mu_k^g) \right) Y_{k-j}^{r,n} - \sum_{i=1}^{N_b} \mathbf{B}_i(\mu_k^g) F_{k-i}^{c,n} \tag{B-7m}$$

The signal F_k^c is therefore equal to

$$F_k^{c,n} = \mathbf{B}_0^{-1}(\mu_k^g) \left(\sum_{j=1}^{N_p} \left(\mathbf{P}_j(\mu_k^r) - \mathbf{P}_j(\mu_k^g) \right) Y_{k-j}^{r,n} - \sum_{i=1}^{N_b} \mathbf{B}_i(\mu_k^g) F_{k-i}^{c,n} \right) \tag{B-7n}$$

To sum it up the augmented gradient experiment for general LPV systems will be defined.

Augmented Generalized Gradient Experiment

For an arbitrarily scheduled LPV system (with an arbitrary amount of inputs and outputs) the augmented gradient experiment is equal to

$$Y_k^{g,n}(\mu_k^g) = \sum_{j=1}^{N_p} \mathbf{P}_j(\mu_k^g) Y_{k-j}^{g,n} + \sum_{i=0}^{N_b} \mathbf{B}_i(\mu_k^g) U_{k-i}^{ff,n} \tag{B-8a}$$

The compensation signal $U_k^{ff,n}$ is built up of three individual signals

$$U_k^{ff,n} = G_k^n + U_k^{c,n} + F_k^{c,n} \tag{B-8b}$$

The signal G_k^n is defined as a zero vector except for the n'th row which is equal to

$$\{G_k^n\}_n = r_k^n - y_k^n \tag{B-8c}$$

The other two signals are defined as

$$U_k^{c,n} = \mathbf{B}_0^{-1}(\mu_k^g) \left(\sum_{i=0}^{N_b} \left(\mathbf{B}_i(\mu_k^r) - \mathbf{B}_i(\mu_k^g) \right) G_{k-i}^n - \sum_{i=1}^{N_b} \mathbf{B}_i(\mu_k^g) U_{k-i}^{c,n} \right) \tag{B-8d}$$

$$F_k^{c,n} = \mathbf{B}_0^{-1}(\mu_k^g) \left(\sum_{j=1}^{N_p} \left(\mathbf{P}_j(\mu_k^r) - \mathbf{P}_j(\mu_k^g) \right) Y_{k-j}^{r,n} - \sum_{i=1}^{N_b} \mathbf{B}_i(\mu_k^g) F_{k-i}^{c,n} \right) \tag{B-8e}$$

This will now be generalized for the three different degrees of actuation.

Generalization: Different Degrees of Actuation

There are different levels of actuation in MIMO systems namely; under-actuated ($n_y > n_u$), fully actuated ($n_y = n_u$) and over-actuated ($n_y < n_u$). Due to the need for matrix inversion in the computation of the compensation signals this is of interest as the matrices in both the over- and under-actuated matrices are not square (as it is defined as a n_y-by-n_u matrix). The inversion problem can be handled as follows:

- Under-actuated
  In the under-actuated case the matrix $\mathbf{B}_0$ will have more rows than columns. In the case wherein the matrix has full column rank the *unique* minimum norm solution can be found (pseudo-inverse). Else a subset of input channels must be chosen which have full column rank.
- Fully actuated
  The matrix $\mathbf{B}_0$ will be square. If the matrix is non-singular the inverse can be used else a subset much be chosen which has full column rank (pseudo inverse).
- Over-actuated
  The process of ignoring of some input channels must be performed as the matrix can never have column rank (as $n_u > n_y$). After acquiring full column rank a (pseudo-)inverse can be performed to compute the compensation signals.

Appendix C

Proofs: A2S-ILPVFT

This appendix contains proofs related to Chapter 4 which were omitted to retain clarity. The proofs contained in this appendix are; the bias of the gradient estimate (C-1) and the factorization of the LPV-ARX model (C-2).

C-1 The Bias of the Gradient Estimate

The IFT framework, for LTI control, had the property of providing an unbiased estimate of the gradient for zero mean white output noise. This meant that some nice properties from stochastic gradient descent theory are inherited by the linear IFT framework [10]. If the unbiasedness also exists in the A2S-ILPVFT algorithm it will also inherit the same, beneficial, properties.

The first, and only, type of noise which will be examined is output noise. Assume that the output of the LPV system is equal to $Y + V$ with V as a vector consisting of a stacking of a zero mean Gaussian white noise signal. Then the following Toeplitz representation can be used

$$\tilde{Y}^r = \left(\mathcal{A}(\mu^r) - \mathcal{B}\mathcal{H}(\mu^r)\right)\tilde{Y}^r + \mathcal{B}\mathcal{H}(\mu^r)(R^r - V^r) \tag{C-1a}$$

The partial derivative with respect to one of the controller parameters then becomes

$$\frac{\partial \tilde{Y}^r(\mu^r,\Theta)}{\partial \theta_i} = \left(\mathcal{A}(\mu^r) - \mathcal{B}\mathcal{H}(\mu^r,\Theta)\right)\frac{\partial \tilde{Y}^r(\mu^r,\Theta)}{\partial \theta_i} + \mathcal{B}\frac{\partial \mathcal{H}(\mu^r,\Theta)}{\partial \theta_i}(R^r - V^r - \tilde{Y}^r(\mu^r,\Theta)) \tag{C-1b}$$

The gradient experiment itself will be corrupted with output noise and thus the (noisy) gradient experiment will be equal to

$$\frac{\partial \hat{Y}^r(\mu^g,\Theta)}{\partial \theta_i} = \mathcal{G}(\mu^g)\frac{\partial \hat{Y}^r(\mu^g,\Theta)}{\partial \theta_i} + \mathcal{B}\frac{\partial \mathcal{H}(\mu^r,\Theta)}{\partial \theta_i}(R^r - V^r - \tilde{Y}^r(\mu^r,\Theta)) + \mathcal{B}\mathcal{H}(\mu^g,\Theta)V^g \tag{C-1c}$$

For brevity the following term was condensed: $\mathcal{A}(\mu^g) - \mathcal{B}\mathcal{H}(\mu^g,\Theta) = \mathcal{G}(\mu^g)$. For simplicity let's rewrite the equation above to

$$\frac{\partial \hat{Y}^r(\mu^g,\Theta)}{\partial \theta_i} = \mathcal{P}(\mu^g,\Theta) + \mathcal{B}\mathcal{H}(\mu^g,\Theta)V^g \tag{C-1d}$$

As derived in Chapter 4 a pseudo-inverse must be used. Therefore the gradient which will be obtained is equal to

$$\frac{\partial \tilde{Y}^r(\mu^r,\Theta)}{\partial \theta_i} = M(\mu^r)\left(M(\mu^g)^+\frac{\partial \tilde{Y}^r(\mu^g,\Theta)}{\partial \theta_i}\right) \tag{C-1e}$$

Filling in the results of Equation C-1d

$$\frac{\partial \tilde{Y}^r(\mu^r,\Theta)}{\partial \theta_i} = M(\mu^r)\left(M(\mu^g)^+(\mathcal{P}(\mu^g,\Theta)+\hat{B}\hat{H}(\mu^g,\Theta)V^g)\right) \tag{C-1f}$$

The noise will be replaced by the following scaled variant of the noise (for brevity)

$$V_2^g = M(\mu^r)\left(M(\mu^g)^+\hat{B}\hat{H}(\mu^g,\Theta)V^g\right) \tag{C-1g}$$

Let's examine whether bias is introduced by V_2^g

$$\mathrm{E}\left[\frac{\partial \hat{J}}{\partial \theta_i}\right] = \mathrm{E}\left[(\tilde{Y}-Y^d)^T\frac{\partial \tilde{Y}}{\partial \theta_i}\right] \tag{C-1h}$$

$$\mathrm{E}\left[\frac{\partial \hat{J}}{\partial \theta_i}\right] = \mathrm{E}\left[(\tilde{Y}-Y^d)^T\left(M(\mu^r)\left(M(\mu^g)^+\mathcal{P}(\mu^g,\Theta)\right)\right)\right] + \mathrm{E}\left[(\tilde{Y}-Y^d)^T V_2^g\right] \tag{C-1i}$$

Under the assumption that the output noise of the gradient experiment is uncorrelated with both the state and scheduling variables

$$\mathrm{E}\left[\frac{\partial \hat{J}}{\partial \theta_i}\right] = \frac{\partial J}{\partial \theta_i} + \mathrm{E}\left[(\tilde{Y}-Y^d)\right]^T\mathrm{E}\left[V_2^g\right] \tag{C-1j}$$

The expected value of the right hand term, i.e. $\mathrm{E}\left[V_2^g\right]$, is equal to zero as seen below

$$\mathrm{E}[M(\mu^g)^+V^g] = \mathrm{E}[M(\mu^g)^+]\mathrm{E}[V^g] = \mathrm{E}[M(\mu^g)^+]0 = 0 \tag{C-1k}$$

Therefore the gradient is unbiased as

$$\mathrm{E}\left[\frac{\partial \hat{J}}{\partial \theta_i}\right] = \frac{\partial J}{\partial \theta_i} + 0 \tag{C-1l}$$

Thus the gradient estimate is unbiased for **zero mean** white *output* noise. For a more general disturbance the author conjectures that the method described in [11] would work. So the output noise itself will not introduce a bias into the system however what about a noise source which is not present in LTI systems/control namely; scheduling noise? This will not be examined in this thesis and will remain an open question.

C-2 Factorization of The LPV-ARX Model

In Section 4-3 it was stated that Equation C-2a suffers from the same curse of dimensionality as the LPV state space approach of [11], this statement will now be proven. The results of Section 4-2 indicate that one must factorize the equation shown below

$$\frac{\partial Y^r(\Theta,\mu^g)}{\partial \theta_i} = (\mathbf{I}-\mathcal{A}(\mu^g)+\mathcal{B}\mathcal{H}(\mu^g,\Theta))^{-1}\mathcal{B}(\mu^g)\frac{\partial \mathcal{H}(\mu^r,\Theta)}{\partial \theta_i}(R^r-Y^r(\mu^r)) \tag{C-2a}$$

The factorization should be of the following format

$$\mathcal{T}(\mu) = M(\mu)\hat{T} \tag{C-2b}$$

For this proof Equation C-2a will first be rewritten to (for brevity)

$$\frac{\partial Y^r(\Theta,\mu^g)}{\partial\theta_i} = (\mathbf{I}-\mathcal{P}(\mu^g))^{-1}\mathcal{B}(\mu^g)\frac{\partial\mathcal{H}(\mu^r,\Theta)}{\partial\theta_i}(R^r - Y^r(\mu^r)) \tag{C-2c}$$

The matrices $\mathcal{B}$ and $\mathcal{P}$ will be defined as follows

$$\mathcal{B}(\mu) = \begin{bmatrix} b_0(\mu_1) & 0 & \dots & 0 & 0 & 0 & 0 \\ b_1(\mu_2) & b_0(\mu_2) & \dots & 0 & 0 & 0 & 0 \\ \vdots & \ddots & \ddots & \vdots & \vdots & \vdots & \vdots \\ b_{nb}(\mu_{nb+1}) & b_{nb-1}(\mu_{nb+1}) & \ddots & b_0(\mu_{nb+1}) & 0 & \dots & 0 \\ 0 & b_{nb}(\mu_{nb+2}) & \ddots & b_1(\mu_{nb+2}) & b_0(\mu_{nb+2}) & \dots & 0 \\ \vdots & \vdots & \ddots & \ddots & \ddots & \ddots & \vdots \\ 0 & 0 & \dots & b_{nb}(\mu_N) & \dots & b_1(\mu_N) & b_0(\mu_N) \end{bmatrix} \tag{C-2d}$$

$$\mathcal{P}(\mu) = \begin{bmatrix} 0 & 0 & \dots & 0 & 0 & 0 & 0 \\ p_1(\mu_2) & 0 & \dots & 0 & 0 & 0 & 0 \\ \vdots & \ddots & \ddots & \vdots & \vdots & \vdots & \vdots \\ p_{np}(\mu_{np+1}) & p_{np-1}(\mu_{np+1}) & \ddots & 0 & 0 & \dots & 0 \\ 0 & p_{np}(\mu_{np+2}) & \ddots & p_1(\mu_{np+2}) & 0 & \dots & 0 \\ \vdots & \vdots & \ddots & \ddots & \ddots & \ddots & \vdots \\ 0 & 0 & \dots & p_{np}(\mu_N) & \dots & p_1(\mu_N) & 0 \end{bmatrix} \tag{C-2e}$$

The inversion is the main source of problems, because as shown in Equation C-2f it changes the structure of the matrix from a banded matrix to lower triangular matrix (for simplicity the case for 4 measurements is written below with $p_i(\mu_k) = 0$ for $i \geq 3$)

$$(\mathbf{I}-\mathcal{P}(\mu^g))^{-1} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ p_1(\mu_2) & 1 & 0 & 0 \\ p_1(\mu_2)p_1(\mu_3)+p_2(\mu_3) & p_1(\mu_3) & 1 & 0 \\ p_1(\mu_2)p_1(\mu_3)p_1(\mu_4)+p_1(\mu_2)p_2(\mu_4)+p_2(\mu_3)p_1(\mu_4) & p_1(\mu_3)p_1(\mu_4)+p_2(\mu_4) & p_1(\mu_4) & 1 \end{bmatrix} \tag{C-2f}$$

An interesting feature is the fact that every column has the same structure. Therefore for the remainder of this section only the first column will be considered. Because of the similar structure between columns this will still lead to a result which can be generalized.

One could add rows, and thus measurements, indefinitely while the lower triangular structure persists. A notable exception is the case wherein $\mathcal{P}(\mu_k) = 0$, i.e. a LPV-FIR filter, for all k as then one inverts the identity matrix. To verify that when $\mathcal{P}(\mu_k) \neq 0$ that the banded structure doesn't return (after inversion). Consider the case wherein $p_i(\mu_k) = 0$ for $i \neq 1$. Then Equation C-2f becomes

$$(\mathbf{I}-\mathcal{P}(\mu^g))^{-1} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ p_1(\mu_2) & 1 & 0 & 0 \\ p_1(\mu_2)p_1(\mu_3) & p_1(\mu_3) & 1 & 0 \\ p_1(\mu_2)p_1(\mu_3)p_1(\mu_4) & p_1(\mu_3)p_1(\mu_4) & p_1(\mu_4) & 1 \end{bmatrix} \tag{C-2g}$$

One can intuitively verify that the lower triangular structure will persist and *not* revert back to a banded matrix. If one now computes the remaining matrix multiplication of Equation *C-2f* for the first column only

$$(\mathbf{I} - \mathcal{P}(\mu^g))^{-1}\mathcal{B}(\mu^g) = \begin{bmatrix} b_0(\mu_1) \\ b_1(\mu_2) + b_0(\mu_1)p_1(\mu_2) \\ b_1(\mu_2)p_1(\mu_3) + b_0(\mu_1)\left(p_2(\mu_3) + p_1(\mu_2)p_1(\mu_3)\right) \end{bmatrix} \tag{C-2h}$$

To proof that the factorization suffers from the curse of dimensionality, as in [11], let's assume that the LPV system has the affine form equal to $p_i = p_i^0 + p_i^1\mu_k$. Then for $k = 1$ one can factorize Equation C-2h as follows

$$b_0(\mu_1) = b_0^0 + b_0^1\mu_1 = \begin{bmatrix} 1 & \mu_1 \end{bmatrix} \begin{bmatrix} b_0^0 \\ b_0^1 \end{bmatrix} \tag{C-2i}$$

Time step $k = 2$ gives

$$b_1(\mu_2)+b_0(\mu_1)p_1(\mu_2) = b_1^0+b_1^1\mu_2+(b_0^0+b_0^1\mu_1)(p_1^0+p_1^1\mu_2) = \begin{bmatrix} 1 & \mu_1 & \mu_2 & \mu_1\mu_2 \end{bmatrix} \begin{bmatrix} b_0^0p_1^0 + b_1^0 \\ b_0^1p_1^0 \\ b_0^0p_1^1 + b_1^1 \\ b_0^1p_1^1 \end{bmatrix} \tag{C-2j}$$

For time step $k = 3$

$$b_1(\mu_2)p_1(\mu_3) + b_0(\mu_1)(p_2(\mu_3) + p_1(\mu_2)p_1(\mu_3)) \tag{C-2k}$$

$$(b_1^0 + b_1^1\mu_2)(p_1^0 + p_1^1\mu_3) + (b_0^0 + b_0^1\mu_1)\left((p_2^0 + p_2^1\mu_3) + (p_1^0 + p_1^1\mu_2)(p_1^0 + p_1^1\mu_3)\right) \tag{C-2l}$$

This can be factorized as

$$\begin{bmatrix} 1 & \mu_1 & \mu_2 & \mu_1\mu_2 & \mu_3 & \mu_1\mu_3 & \mu_2\mu_3 & \mu_1\mu_2\mu_3 \end{bmatrix} \begin{bmatrix} b_1^0p_1^0 + b_0^0(p_2^0 + p_1^0p_1^0) \\ b_0^1(p_1^0p_1^0 + p_2^0) \\ p_1^0b_1^1 + b_0^0p_1^0p_1^1 \\ b_0^1p_1^0p_1^1 \\ b_1^0p_1^1 + b_0^0(p_2^1 + p_1^0p_1^1) \\ b_0^1(p_2^1 + p_1^0p_1^1) \\ b_1^1p_1^1 + b_0^0p_1^1p_1^1 \\ b_0^1p_1^1p_1^1 \end{bmatrix} \tag{C-2m}$$

This is the same exponential increase in dimensionality as seen in [11]. Therefore one can conclude that the LPV-ARX and LPV state space model structures both suffer from the same curse of dimensionality. So this modelling approach can be concluded to be infeasible for circumventing the curse of dimensionality. However Equation C-2f does imply that using a LPV-FIR filter might be a better alternative (i.e. $\mathcal{P}(\mu^g) = 0$).

Appendix D

Optimization Theory & Iterative Feedback Tuning

This appendix discusses general properties/ideas from optimization theory which are relevant when using the IFT framework. Three relevant properties of interest will be discussed namely; the initial value (D-1), the importance of scaling (D-2) and constraints (D-3).

D-1 Initial Controller

The initial controller is of significant importance to whether the algorithm fails, i.e. diverges, or succeeds. The divergence, or failure, can be attributed to the combination of the IFT framework and the Newton method. Luckily only two properties carry significance namely;

1. The initial controller *must* stabilize the closed loop [10].
2. The Newton method can diverge if the initial value is not in the neighbourhood of a local minimum and the step length parameter γ_i be set appropriately [16].

The first property is, in the author's opinion, not really a drawback. Because performing experiments in closed loop when the plant is unstable is not really an option (due to potential damage to the machine). Besides the damage argument results in the literature indicate that the IFT framework will in, general, fail in such a situation [10].

The second property can be a problem but is inherent to the Newton method and should be considered a word of caution.

D-2 Gradient Scaling

A proper scaling of the gradient is required to mitigate the difference in magnitude of the different controller parameters (for a general discussion on scaling the reader is referred to

[16]). One method to directly scale the gradient is by scaling the reference signal. This can be seen by looking at Equation D-1a (the LTI gradient experiment for time step k)

$$\frac{\partial y_k(\Theta)}{\partial \theta_i} = (1 - A + BH(\Theta))^{-1} B \frac{\partial H(\Theta)}{\partial \theta_i}(r_k - y_k(\Theta)) \tag{D-1a}$$

If one would now scale the reference signal by a factor α the gradient experiment becomes (note that the discussion is about LTI systems so y_k will also scale with α)

$$\frac{\partial y_{k,2}(\Theta)}{\partial \theta_i} = (1 - A + BH(\Theta))^{-1} B \frac{\partial H(\Theta)}{\partial \theta_i}\alpha(r_k - y_k(\Theta)) \tag{D-1b}$$

It is easy to see that one has scaled the original gradient, of Equation D-1a, by the same factor α

$$\alpha\frac{\partial y_k(\Theta)}{\partial \theta_i} = (1 - A + BH(\Theta))^{-1} B \frac{\partial H(\Theta)}{\partial \theta_i}\alpha(r_k - y_k(\Theta)) \tag{D-1c}$$

One problem which was present in the flutter system was the badly scaled nature of the cost function for both algorithms. This problem can most intuitively be seen from the extended regressor representation of the gradient (Chapter 3). Take for example the extended regressor defined as

$$\Phi_k y_k^g = y_k^g \begin{bmatrix} 1 \\ q^{-1} \end{bmatrix} \begin{bmatrix} 1 & \mu_k \end{bmatrix} \tag{D-1d}$$

The gradient is then equal to

$$\frac{\partial J(\Theta)}{\partial \theta_i} = (y_k - y_k^d) \begin{bmatrix} y_k^g & y_k^g\mu_k \\ y_{k-1}^g & y_{k-1}^g\mu_k \end{bmatrix} \tag{D-1e}$$

Note that besides the influence of the difference between the desired and current output signals signal that the gradient signal is also scaled by the basis function.

This observation poses a key issue as the inherent scaling can cause problems if, for instance, $|\mu_k| >> 1$ for all k. The gradient of the controller parameters related to the basis functions of the scheduling variable will then be much larger than those of the time-invariant controller parameters. Unfortunately this might not be an accurate description of the proper gradient. With a proper gradient defined as a gradient which is unaffected by the underlying signal magnitudes and is thus *proper*ly corrected for the different scales of the controller parameters.

In my simulations of the flutter model this was compensated by scaling with a factor of $\frac{1}{8}$ to let the scheduling variable vary between $\frac{5}{8}$ (for a wind speed of 5 m/s) and $\frac{11}{8}$ (11 m/s). However the author does admit that the implementation could possibly be improved by using a smarter scaling method.

D-3 Constraints

An important aspect of optimization problems in real applications is the addition of constraints. Examples of constraints in control systems are; the physical limitations of actuators

($u_k \leq u_{max}$) or related to safety (for example an overshoot lower than 10%). The *constrained* optimization problem can be formulated as

$$\min_{\Theta^{fb}} J(\Theta^{fb}) = \min_{\Theta^{fb}} \frac{1}{N}\sum_{i=1}^{N}(y_i(\Theta^{fb}) - y_i^d)^2 \tag{D-2a}$$

Under the constraints

$$h(\Theta^{fb}) = 0 \tag{D-2b}$$

$$g(\Theta^{fb}) \leq 0 \tag{D-2c}$$

The constraints $h(\Theta^{fb})$ and $g(\Theta^{fb})$ are, respectively, the equality and inequality constraints. It has been shown that the original IFT framework, for LTI systems has, can incorporate constraints. Examples in the literature include; stability constraints as in [15] and robustness constraints, related to prevention of machine damage, as in [22].

The general approach to implementing constraints in the IFT framework is transforming the optimization problem of Equation D-2a to D-2c into an unconstrained optimization problem. As an example the transformation of a constrained optimization problem will be performed using penalty functions (based on [22]). Only inequality constraints will be considered in this example.

$$\min_{\Theta^{fb}} \tilde{J}(\Theta^{fb}) = J(\Theta^{fb}) + J^g(\Theta^{fb}) \tag{D-3a}$$

With $J(\Theta^{fb})$ as in equation D-2a and $J^g(\Theta^{fb})$ is a penalty function based on Equation D-2c (with α a positive real number)

$$J^g(\Theta^{fb}) = \frac{\alpha}{2}\phi(g)g(\Theta^{fb})^2 \tag{D-3b}$$

With $\phi(g) = 0$ if $g \leq 0$ and $\phi(g) = 1$ otherwise (i.e. only active when it is exceeded).

An important note is that one should have a model available to evaluate these constraints. Therefore adding constraints does mean that IFT becomes a model-based, or rather model dependent, method. The gradient of the transformed cost function (Equation D-3a) is equal to

$$\frac{\partial \tilde{J}(\Theta^{fb})}{\partial \Theta^{fb}} = \frac{\partial J(\Theta^{fb})}{\partial \Theta^{fb}} + \frac{\partial J^g(\Theta^{fb})}{\partial \Theta^{fb}} \tag{D-4a}$$

The first term (i.e. $\frac{\partial J(\Theta^{fb})}{\partial \Theta^{fb}}$) can be acquired using the dedicated experiments as discussed in Chapter 2. The gradient $\frac{\partial J^g(\Theta^{fb})}{\partial \Theta^{fb}}$ can be evaluated to be

$$\frac{\partial J^g(\Theta^{fb})}{\partial \Theta^{fb}} = \alpha\phi(g)g(\Theta^{fb})\left(\phi(g)\frac{\partial g(\Theta^{fb})}{\partial \Theta^{fb}} + g(\Theta^{fb})\frac{\partial \phi(g)}{\partial g(\Theta^{fb})}\frac{\partial g(\Theta^{fb})}{\partial \Theta^{fb}}\right) \tag{D-4b}$$

This can be simplified by noting that $\frac{\partial \phi(g)}{\partial g(\Theta^{fb})} = 0$ (as shown in [22])

$$\frac{\partial J^g(\Theta^{fb})}{\partial \Theta^{fb}} = \alpha\phi^2(g)g(\Theta^{fb})\frac{\partial g(\Theta^{fb})}{\partial \Theta^{fb}} \tag{D-4c}$$

Computing the gradient of D-4c can be done through, for example, finite difference approximations of the derivative using an identified model of the plant.

Although the addition of constraints is as of yet limited to LTI systems the author conjectures that one can achieve the addition of constraints to LPV control in the IFT framework by implementing them as LMIs. The addition of constraints will not be developed in this thesis but LMIs, and their wide range of applicability for LPV systems and control, make them a prime candidate [2][3].

The author conjectures that combining LMIs with the penalty transformation as described in this section could allow for the addition of constraints in the developed algorithms. Note that although LMIs are inherently matrices one can use the determinant to acquire a scalar output from the constraints as needed for the, scalar, cost function.

Bibliography

[1] R. Toth, M. van de Wal, P. S. C. Heuberger, and P. M. J. van den Hof, "Lpv identification of high performance positioning devices," *2011 American Control Conference*, pp. 151–158, 2011.

[2] J. Mohammadpour and C. Scherer, *Control of Linear Parameter Varying Systems with Applications*. Springer, 2012.

[3] C. Briat, *Linear Parameter-Varying and Time-Delay Systems: Analysis, Observation, Filtering & Control*. Springer, 2015.

[4] M. G. Wassink, M. van de Wal, C. Scherer, and O. Bosgra, "Lpv control for a wafer stage: beyond the theoretical solution," *Control Engineering Practice 13*, pp. 231–245, 2005.

[5] J.-W. van Wingerden and M. Verhaegen, "Subspace identification of bilinear and lpv systems for open- and closed-loop data," *Automatica 45*, pp. 372–381, 2002.

[6] J. de Caigny, R. Pintelon, J. F. Camino, and J. Swevers, "Interpolation-based modeling of mimo lpv systems," *IEEE Transactions on Control Systems Technology, Vol. 22*, pp. 2232–2246, 2014.

[7] D. Vizer and G. Mercere, "An $\mathcal{H}_\infty$-norm-based approach for operating point selection and lpv model identification from local experiments," *Periodica Polytechnica - Electrical Engineering and Computer Science*, pp. 121–131, 2014.

[8] B. Bamieh and L. Giarre, "Identification of linear parameter varying models," *International Journal of Robust and Nonlinear Control 12*, pp. 841–853, 2002.

[9] B. de Schutter and M. Heemels, *Lecture Notes: Modeling and Control of Hybrid Systems*. Delft Univerity of Technology, 2012.

[10] H. Hjalmarsson, "Iterative feedback tuning-an overview," *International Journal of Adaptive Control and Signal Processing*, pp. 373–395, 2002.

[11] S. Navalkar, T. Oomen, and J.-W. van Wingerden, "Ift-lpv: Data-based tuning of fixed structure controllers for lpv systems," *Accepted for Publication*, p. n/a.

[12] J.-W. van Wingerden, *Control of Wind Turbine with 'Smart' Rotors: Proof of Concept & LPV Subspace Identification.* PhD Thesis, TU Delft, 2008.

[13] Z.-H. Hou and Z. Wang, "From model-based control to data-driven control: Survey, classification and perspective," *Information Sciences 235*, pp. 3–35, 2013.

[14] J. Huusom, N. Poulsen, and S. Jorgensen, "Improving convergence of iterative feedback tuning," *Journal of Process Control 19*, pp. 570–578, 2009.

[15] F. de Bruyne and L. C. Kammer, "Iterative feedback tuning with guaranteed stability," *Proceedings of the American Control Conference*, pp. 3317–3321, 1999.

[16] P. Papalambros and D. White, *Princples of Optimal Design: Modeling and Computation, second edition.* Cambridge University Press, 2003.

[17] F. de Bruyne, B. Anderson, M. Gevers, and N. Linard, "Iterative controller optimization for nonlinear systems," *Proceedings of the 36th Conference on Decision & Control*, pp. 3749–3754, 1997.

[18] H. Hjalmarsson, M. Gevers, S. Gunnarson, and O. Lequin, "Iterative feedback tuning: Theory and applications," *IEEE Control Systems August*, pp. 26–41, 1998.

[19] S. Formentin, D. Piga, R. Toth, and S. Savaresi, "Direct data-driven control of linear parameter varying systems," *2013 IEEE 52nd Annual Conference on Decision and Control (CDC)*, pp. 4110 – 4115, December 2013.

[20] S. Veres and H. Hjalmarsson, "Tuning for robustness and performance using iterative feedback tuning," *Proceedings of the 41st IEEE Conference on Decision and Control*, pp. 4682–4687, 2002.

[21] S. Theodoridis and K. Koutroumbas, *Pattern Recognition: Fourth Edition.* Elsevier, 2009.

[22] B. J. C. H. van der Velden, T. Oomen, and M. F. Heertjes, "Constrained iterative feedback tuning for robust high-precision motion control," *Preprints of the 19th World Congress, The International Federation of Automatic Control*, pp. 4915–4920, August 2014.

[23] H. Hjalmarsson and T. Birkeland, "Iterative feedback tuning of linear time-invariant mimo systems," *Proceedings of the 37th IEEE Conference on Decision and Control*, pp. 3893–3898, 1998.

[24] S. van der Meulen, R. L. Tousain, and O. Bosgra, "Fixed structure feedforward controller design exploiting iterative trials: Application to a wafer stage and a desktop printer," *Journal of Dynamic Systems, Measurements and Control*, pp. 1–16, September 2008.