# A NON-LINEAR APPROACH TO MODELLING AND CONTROL OF ELECTRICALLY STIMULATED SKELETAL MUSCLE

A DISSERTATION

SUBMITTED TO THE DEPARTMENT OF ELECTRONICS AND ELECTRICAL ENGINEERING

OF GLASGOW UNIVERSITY

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS

FOR THE DEGREE OF

DOCTOR OF PHILOSOPHY

By

Henrik Gollee

March 1998

# Abstract

This thesis is concerned with the development and analysis of a non-linear approach to modelling and control of the contraction of electrically stimulated skeletal muscle.

For muscle which has lost nervous control, artificial electrical stimulation can be used as a technique aimed at providing muscular contraction and producing a functionally useful movement. This is generally referred to as Functional Electrical Stimulation (FES) and is used in different application areas such as the rehabilitation of paralysed patients and in cardiac assistance where skeletal muscle can be used to support a failing heart. For both these FES applications a model of the muscle is essential to develop algorithms for the controlled stimulation.

For the identification of muscle models, real data are available from experiments with rabbit muscle. Data for contraction with constant muscle length were collected from two muscles with very different characteristics. An empirical modelling approach is developed which is suitable for both muscles. The approach is based on a decomposition of the operating space into smaller sub-regions which are then described by local models of simple, possibly linear structure. The local models are blended together by a scheduler, and the resulting non-linear model is called a Local Model Network (LMN). It is shown how *a priori* knowledge about the system can be used directly when identifying Local Model Networks. Aspects of the structure selection are discussed and algorithms for the identification of the model parameters are presented. Tools for the analysis of Local Model Networks have been developed and are used to validate the models.

The problem of designing a controller based on the LMN structure is discussed. The structure of Local Controller Networks is introduced. These can be derived directly from Local Model Networks. Design techniques for input–output and for state feedback controllers, based on pole placement, are presented.

Aspects of the generation of optimal stimulation patterns (which are defined as stimulation patterns which deliver the smallest number of pulses to obtain a desired contraction) are discussed, and various techniques to generate them are presented. In particular, it is shown how a control structure can be used to generate optimal stimulation patterns. A Local Controller Network is used as the controller with a design based on a non-linear LMN model of muscle.

Experimental data from a non-linear heat transfer process have been collected and are used to demonstrate the basic modelling and control principles throughout the first part of the thesis.

# Acknowledgements

The completion of a thesis relies greatly on contributions in various forms from many different people and this thesis is no exception. I would like to thank (as best as I can) the following people for their support and encouragement during the past three years.

Nick Donaldson for initially establishing the link between engineers and people working in the area of biomedicine which was essential for this interdisciplinary project to evolve. Jonathan Jarvis for providing the data and for sharing his knowledge of biomedicine and physiology with me in many insightful discussions. Martin Kwende for carrying out the muscle experiments from which I could use the data.

Ken Hunt for getting me involved with biomedical engineering, and for his encouragement, feedback and comments on my thesis.

My supervisor David Murray–Smith for giving me the opportunity to work on a project in the area of biomedical engineering, and for providing guidance in the right direction at the right time.

Rafał Żbikowski for insightful discussions about system theory and Lyapunov stability. Robert Shorten for his suggestions and comments on model analysis, and Roderick Murray–Smith for discussions about system modelling.

Many thanks to my friends and colleagues for making it so enjoyable to work in the Centre for Systems and Control, in particular Gary Gray, Eric Ronco, KC Tan, Anna Esparcia–Alcázar and all the other "Box people". Jesús Rico Melgoza for the interesting and motivating discussions at lunch time.

On the personal side, I would like to thank my parents for giving me the best start in life I could have hoped for.

Finally, thanks to all Glaswegians for being such friendly people, and to Glasgow for being such an enjoyable place.

# Contents

# List of Tables

# List of Figures

# 1 Introduction

This thesis is concerned with modelling and control of electrically stimulated muscle. A non-linear modelling approach is presented. Techniques to design controllers based on these model structures are developed.

This introductory chapter gives an overview of the thesis and provides background information about the project of which the work presented forms a part. This project is coordinated by J. C. Jarvis at the Department of Human Anatomy and Cell Biology, University of Liverpool.

## 1.1 Cardiac Support using Skeletal Muscle Ventricles

Under certain conditions a weak heart can fail to provide the pumping power required to keep the circulatory system in a stable state. Depending on the level of weakness, different strategies can be used to assist the failing heart:

1. A slightly damaged heart can be stimulated directly by periodic electrical impulses to provide a consistent heart beat. This technique is used, for example, in pace makers.

2. A weak heart can be supported by additional hydraulic power which is provided by a pumping device. The pumping device is activated in phase with the heart beat when the power generated by the heart is not sufficient to sustain the blood circulation. The device can be placed at the artery and is primarily aimed at taking load off the damaged heart thus enabling the heart to recover.

3. If the heart fails completely, it needs to be replaced by either an artificial heart or, if available, a heart transplant.

For the project of which this work forms a part we are primarily concerned with supporting a weak heart (i.e. item 2 above). A straightforward approach is to use a mechanical pump to assist the heart. However, such a system requires an external power supply to provide the energy needed for the supporting device and is therefore only suitable for application in a clinical environment.

Another approach is based on Functional Electrical Stimulation (FES) of skeletal muscle transformed into a so-called Skeletal Muscle Ventricle (SMV). In general skeletal muscle may

be made to contract by artificial electrical stimulation of the corresponding motoneurons. This technique is used, for example, in the rehabilitation of paralysed patients where natural nervous control of muscular contraction has been lost due to injury of the spinal cord. In that case, the use of FES is aimed at restoring function to affected limbs by providing artificial electrical stimulation patterns (Kralj and Bajd 1989).

In cardiac assistance, FES can be used to stimulate a skeletal muscle of the patient which has been transformed to create a Skeletal Muscle Ventricle to support the weak heart (Chagas *et al.* 1989, Hooper and Stephenson 1991, Pochettino *et al.* 1991, Salmons and Jarvis 1992). The advantage of this setup is that the assisting device (i.e., the SMV) obtains its energy supply from the patient's body. Together with an implanted stimulator, this technique is therefore suitable for autonomous use. (Salmons and Jarvis 1992) have shown that using a muscle from the patient's back (*latissimus dorsi*) and reconfiguring it as a Skeletal Muscle Ventricle can potentially provide sufficient power to assist a failing heart. A schematic diagram of a possible setup is shown in Figure 1.1.

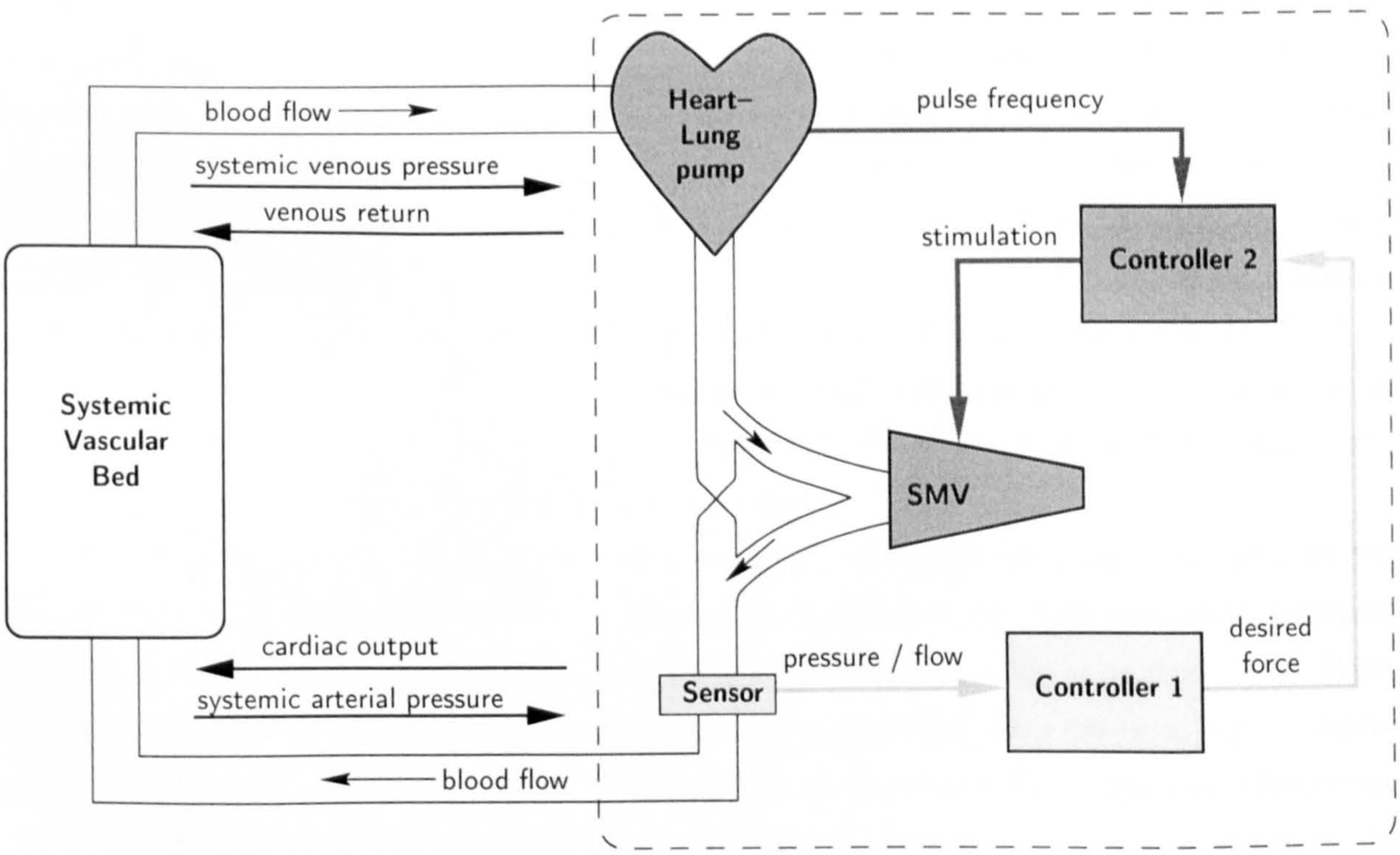



Figure 1.1: Control structure for cardiac support using a Skeletal Muscle Ventricle. The force generating elements are surrounded by a dashed line.

The circulatory system is interpreted as a closed loop hydraulic system. It can be divided into the heart-lung pump as the force generating element and the systemic vascular bed as the load (Sagawa 1973). Systemic arterial pressure and systemic venous pressure can be thought of as input variables to the heart-lung pump and the output of the systemic vascular

bed, whereas cardiac output and venous return are given the same value and are regarded as the forcing variable in analysing the systemic vascular bed and the dependent variable in analysing the heart-lung pump.

In the right half of Figure 1.1, the elements of the supporting device are shown. It consists of the SMV and two controllers which generate the electrical stimulation pulses for the SMV. The controller design is aimed to meet the following criteria:

1. The SMV should support the weak heart in such a way that only the power actually needed is generated. Thus, the required pumping activity depends on parameters such as blood pressure and flow.

2. The SMV should be stimulated in such a way that it generates the pumping power required with an optimised phase relationship to the natural heartbeat. Furthermore, long term changes in the skeletal muscle due to the artificial stimulation should not cause progressive deterioration of the SMV performance.

Controller 1 is designed to meet objective 1. It requires a model of the pressure–flow relationship in the arterial tree of the circulatory system. The oldest model, dating back to the beginning of the century, is the Windkessel model which is based on linear time-domain considerations. It describes the artery as a linear compliance which stores energy during systole and is discharged via the peripheral resistance during diastole. A number of more complex models have been developed since which include non-linear effects. An introduction to modelling of the arterial tree can be found in (Timmons 1995) and (Westerhof *et al.* 1977).

The command signal for Controller 1 is based on sensor measurements of arterial pressure and flow. As these variables change relatively slowly (in the order of seconds to minutes) the time constants of this controller will be relatively large.

Controller 1 generates a desired force which is used to generate the command signal for the second control task which corresponds to Controller 2 in Figure 1.1. This controller generates the actual stimulation pulses delivered to the muscle, in phase with the heart beat. Here, the desired activation of the SMV can reasonably be described as a series of square waves whose amplitude is defined by the desired force generated by Controller 1 and whose frequency depends on the frequency of the heart beat. As the pulse frequency of a human varies in the range of 70 beats/min at rest over 150 beats/min during medium activation to a maximum of 200 beats/min for very high activation (Gray 1995), a frequency range of $[1 \ldots 3]$ Hz can be expected. Hence, the second control task requires much smaller time constants than the first. The design of Controller 2 is based on a model of the contraction of electrically stimulated skeletal muscle.

This thesis is primarily aimed at developing such a model and to show how it can be used to design a corresponding stimulation controller. As muscle is a highly non-linear system, a non-linear modelling approach needs to be used. The primary modelling objective is to

develop stimulation controllers. Hence, the model structure needs to be controller orientated. This means that most analytical modelling approaches for muscle are not directly suitable, as they are mainly aimed at describing the physiological characteristics of muscle, either on a microscopic or on a macroscopic level, which usually leads to complex non-linear model structures which cannot be used directly for the design of controllers. We will therefore employ an empirical approach to model the contraction of muscle which is based on input–output data from standard experiments and which leads to a model structure which can readily be used to design stimulation controllers.

For the identification of muscle models, real data are available from experiments with rabbit muscle (*tibialis anterior*). Data for contraction with constant muscle length (isometric muscle contraction) were collected from two muscles with very different characteristics. We aim to develop a generic modelling approach which is suitable for both muscles.

The use of the modelling technique is not limited to the design of stimulation controllers for Skeletal Muscle Ventricles in cardiac assistance but can be applied to design stimulation controllers for different applications of FES, e.g. for unsupported standing of a paraplegic subject (Hunt *et al.* 1997, Munih *et al.* 1997).

## 1.2 Empirical Non-linear Modelling

Many of the model structures used for empirical non-linear system identification, especially some popular structures of artificial neural networks, treat the system as a black–box. This means that they are not intended to have physical significance or to be related to the structure of the real system. No *a priori* knowledge is employed in the design of a model in the hope that the modelling process can be fully automated.

For most real world problems, *a priori* information is available in some way: the approximate dynamic order may be known, some information about non-linearities and their physical background may be available, and states of the system might be known. Thus, a system identification algorithm is required which can make use of this system knowledge. In such a way, the class of potential model structures can be limited and the system identification task can be radically simplified.

Another important aspect of system identification is the validation of an identified model. For black–box modelling, the only way of validating a model is to use cross–validation techniques (Weiss and Kulikowski 1991), i.e. to compare the output of the model with the output of the real system for all operating conditions known. When dealing with real-world non-linear systems, it is usually impossible to cover all operating conditions with experimental data. With a purely empirical modelling approach, it is then never completely known how the model will perform for operating conditions other than those covered by the data. If

however the model structure can be subject to analysis of its properties, then model properties can be compared with characteristics of the real system. This is an additional way of validating the model structure. Such analytical validation is almost always essential when the model is to be used in a real application.

In this thesis, the use of Local Model Networks (LMN) for non-linear system identification is reviewed. It is shown how *a priori* knowledge about the system can be used directly when identifying Local Model Networks. Tools for analysis of Local Model Networks have been developed and can be used to validate the model.

Local Model Networks are used to model the contraction of electrically stimulated muscle. The design of Controller 2 in Figure 1.1 can be based on a non-linear model of the muscle which forms the SMV. Based on the structure of a Local Model Network, a non-linear controller with a similar structure can be designed, using standard controller design techniques. This approach is referred to as a Local Controller Network.

To investigate how the algorithms developed in this work can be applied to model an engineering systems, experimental data from a non-linear heat transfer process have been collected and are used to demonstrate the basic modelling and control principles.

## 1.3 Outline of Thesis

In Part I of this thesis, theoretical aspects of the modelling and control techniques used are discussed.

- In **Chapter 2**, Local Model Networks are introduced as the modelling approach used. An overview of modelling techniques which are based on operating regime decomposition is given. Aspects of the structure selection are discussed and algorithms for the identification of the model parameters are presented. The interpretation, analysis and validation of the resulting model are discussed.

- In **Chapter 3**, the problem of designing a controller based on the model structure introduced in Chapter 2 is discussed. The structure of Local Controller Networks which are directly based on Local Model Networks is found to be related to the gain scheduling approach to control. Design techniques for input–output controllers and state feedback controllers, based on pole placement, are introduced.

Part II forms the main part of the thesis and deals with the application of the modelling and control techniques introduced in Part I to electrically stimulated muscle.

- **Chapter 4** provides an overview of aspects of artificial electrical stimulation of skeletal muscle. After a short introduction to muscle physiology, the main differences between natural and artificial stimulation are discussed. Following that, a review of muscle

models is given, ranging from analytical to empirical models, with special emphasise on their relevance to Functional Electrical Stimulation.

- In **Chapter 5**, Local Model Networks are used to model the contraction of electrically stimulated muscle under isometric conditions. This chapter forms the main contribution of the thesis. It is shown that the model developed can describe non-linear characteristics of the muscle with great accuracy. The models obtained are thoroughly analysed and validated, and it is shown how a model can be modified to incorporated *a priori* knowledge of the system characteristics.

- In **Chapter 6**, aspects of the generation of optimal stimulation patterns (which are defined as stimulation patterns which deliver the smallest number of pulses to obtain a desired contraction) are discussed, and various techniques to generate them are presented. In particular, it is shown how a closed loop control structure based on the models obtained in Chapter 5 can be used to generate optimal stimulation patterns.

Conclusions and recommendations for future work are presented in the final **Chapter 7**.

The thesis has six appendices. Appendices A, B, C and D contain standard material which is relevant for the work presented in the main parts of the thesis.

- **Appendix A** discusses aspects of the linearisation of non-linear systems which are relevant for the modelling approach discussed in Chapter 2.

- In **Appendix B** optimisation algorithms are introduced which are used to estimate the parameters of the Local Model Networks.

- Linear controller design techniques are presented in **Appendix C**. These techniques are used for the design of Local Controller Networks in Chapter 3.

- In **Appendix D** aspects of the simulation of systems in continuous and in discrete time are discussed.

**Appendix E** introduces the heat transfer process and presents the which are used throughout Part I of the thesis to demonstrate the modelling and control concepts.

Notations and abbreviations used throughout the thesis are summarised in **Appendix F**.

## 1.4 Thesis Contributions

The contributions of the thesis with respect to the modelling and control of electrically stimulated muscle can be summarised as follows:

- A new empirical approach to model the contraction of muscle stimulated with supra-maximal impulses under isometric conditions is developed. The model is non-linear and

can therefore account for non-linear force–frequency characteristics such as the "catch-like" effect[1]. The identification of the model parameters is based on input–output data of the muscle from standard experiments. Although the model is empirical, it can be interpreted in terms of the properties of the real system. It is demonstrated how *a priori* knowledge can be incorporated. The simulation of the model is not computational expensive which makes the approach suitable for use in real time with implanted devices. The model structure is controller orientated.

- Based on the non-linear model of muscle contraction, an algorithm is developed which can be used to generate stimulation patterns which take account of non-linear force-frequency characteristics. The stimulation patterns have properties similar to optimal stimulation patterns which deliver the smallest number of pulses to obtain a desired contraction and are therefore thought to minimise muscle fatigue and to influence long term changes of the muscle due to the artificial stimulation in a positive way. The algorithm employs a closed loop control approach where the structure and the design of the controller is based on a non-linear model of the muscle. The developed controller is not computational expensive and therefore suitable for use in real time with implanted devices.

In terms of the modelling and control concepts used, the following contributions have been made:

- A review of Local Model Network based approaches to modelling is given. Different structures for the scheduler and the local models are compared. Aspects of the parameter estimation are discussed.

- Tools to analyse Local Model Networks are developed. It is shown that the model can be interpreted as a linear parameter-variant system, and how the stability of such systems can be analysed. A graphical interpretation of stability of Local Model Networks with local linear models with real eigensystems is applied. It is demonstrated how the parameters of the Local Model Networks can be interpreted with respect to known characteristics of the real system.

- An overview of controller design based on Local Model Networks is given, and its relation to gain scheduling control is outlined. Design methods based on pole-placement are introduced for Local Model Networks in input–output and in state space form.

- The modelling and control concepts are demonstrated on a heat transfer process to show their applicability to engineering systems as well as to a biological system such as muscle.

[1] The "catch-like" effect refers to a more than linear summation of the muscle response to two or three closely spaced stimulation pulses at the onset of a contraction. In our experiments, this effect is present only in one of the muscles.

# Part I

# Theory

# 2 Local Modelling

## 2.1 Introduction

In this chapter we consider the modelling of complex, non-linear, dynamic systems using a multiple–model approach. The approach is based on the concept of "divide and conquer" which is a common strategy in solving engineering problems. It can be formulated as follows:

> A complex problem is somehow partitioned into a number of simpler subproblems that can be solved independently, and whose individual solutions yield the solution of the original complex problem. (Johansen and Murray-Smith 1997)

A problem can be decomposed according to a number of different criteria. (Johansen and Murray-Smith 1997) suggest the following classification:

1. Decomposition into physical components. For example, when modelling the human circulatory system the model can be decomposed into a model of the heart–lung pump and a model of the systemic vascular bed.

2. Decomposition based on phenomena. For example when modelling the human circulatory system, heat transfer and oxygen transfer can be modelled separately.

3. Decomposition in terms of mathematical series expansions. This includes for instance Laguerre polynomials, Volterra expansions, Fourier transformation.

4. Decomposition into goals. For example, to restore walking in a paraplegic subject using Functional Electrical Stimulation of muscle ((Kralj and Bajd 1989), see also Chapter 4), one can attempt first to make the subject stand up safely before approaching the task of restoring walking capabilities.

5. Decomposition into operating regimes.

When solving an engineering problem all these decomposition approaches will be considered in some way. However, in this work we will focus on the decomposition into operating regimes.

The term *operating regime decomposition* refers to a decomposition of the full operating region of a system into smaller regions which can then be modelled locally. This

idea is employed in a number of approaches and techniques, for example Local Model Networks (Johansen and Foss 1992, Johansen and Foss 1993, Murray-Smith 1994), discrete logic (Söderman and Strömberg 1993, Billings and Voon 1987), finite state automata (Zhang *et al.* 1994, Meilă and Jordan 1997), fuzzy logic (Takagi and Sugeno 1985), probabilistic approaches (Jacobs 1995, Skeppstedt *et al.* 1992), and hierarchical modelling (Jordan and Jacobs 1994). These phrases refer to similar techniques and are partially overlapping. A comprehensive review can be found in (Johansen and Murray-Smith 1997).

In this work, we will primarily use the term *Local Model Network* when we refer to modelling techniques which are based on operating regime decompositions. One should, however, keep in mind that sometimes ideas from the related areas are employed, often using only a different terminology.

## 2.2 Structures

The basic idea of the modelling approach employed in this work is to divide a complex, non-linear modelling task into smaller, simpler sub-tasks. Each sub-task can then be handled locally by a simpler model. A scheduler (or supervisor) decides how relevant the models are for the current operating condition, and weights them accordingly. The overall model is composed as the sum of all weighted local models. Such a structure is shown in Figure 2.1.

The local models can generally be of any form, e.g. linear or non-linear, in input–output or state–space form, empirical or based on physical analysis. It is often straightforward to incorporate *a priori* knowledge at this stage.

We will first discuss aspects of local function decomposition, but then focus on issues related to the decomposition of dynamic systems.

### 2.2.1 Local Function Decomposition

We consider the local decomposition of the static relation

$$y = f(\underline{\psi}), \tag{2.1}$$

with $\underline{\psi} \in \Psi \subset \mathbb{R}^{n_\psi}$ the input vector, $y \in \mathbb{Y} \subset \mathbb{R}$ the scalar output, and $f : \Psi \to \mathbb{Y}$ a smooth non-linear continuous function.

We introduce a scheduler which consists of a set of $M$ scalar functions $\rho_i(\underline{\phi}) : \Phi \to [0, 1]$, with $\underline{\phi} \in \Phi \subset \mathbb{R}^{n_\phi}$ being the scheduling vector. These functions are smooth with a localised region of activity. The set of functions $\{\rho_i\}_{i=1}^{M}$ forms a partition of unity of the scheduling space, i.e.

$$\sum_{i=1}^{M} \rho_i(\underline{\phi}) = 1 \quad \forall \underline{\phi} \in \Phi \tag{2.2}$$

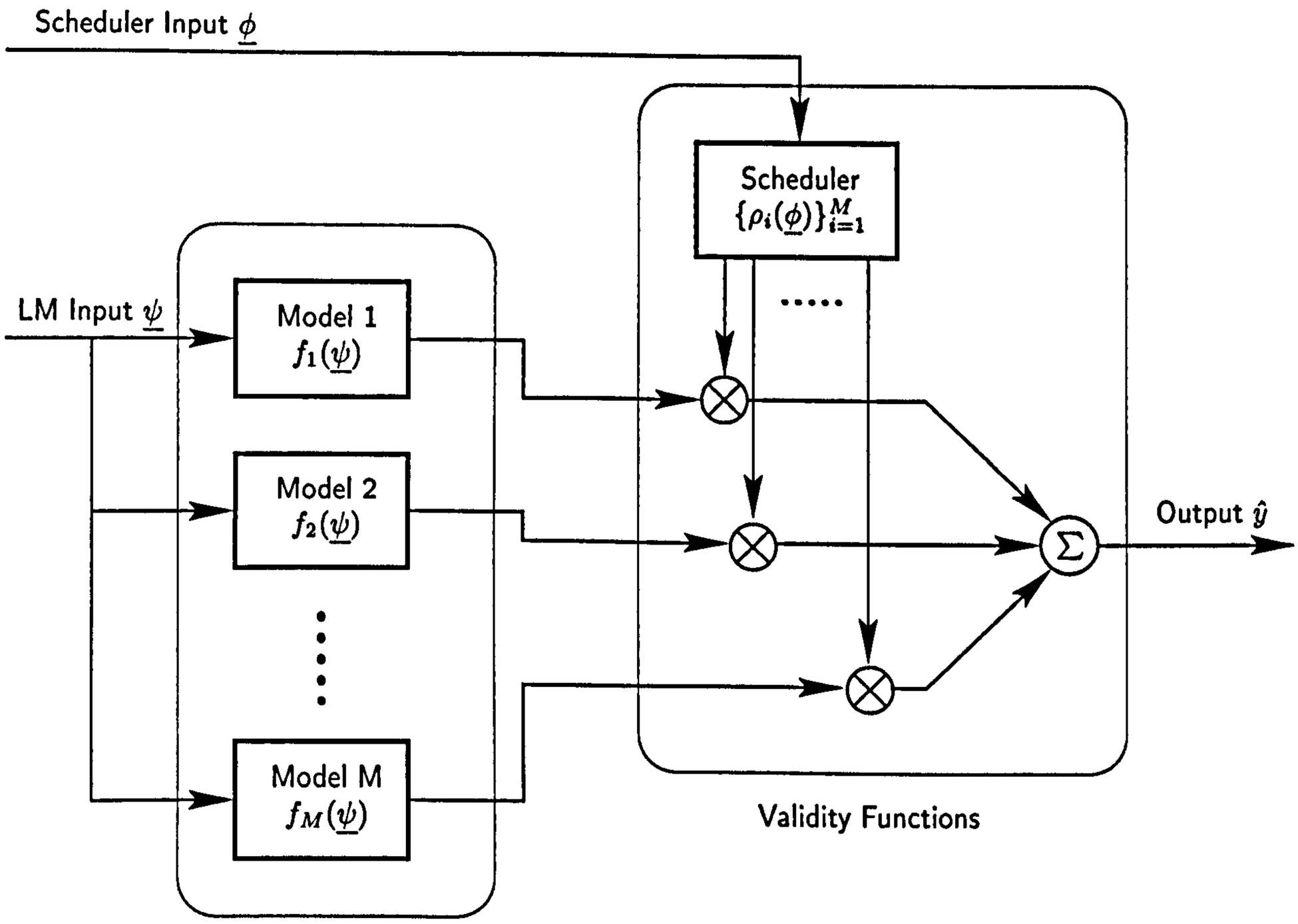



Figure 2.1: The multiple model approach (Local Model Network) (adapted from (Murray-Smith and Johansen 1997*b*)).

Employing a set of scheduling functions with these properties, we can rewrite equation (2.1) as

$$y = \sum_{i=1}^{M} \rho_i(\underline{\phi})\, f(\underline{\psi})\,. \tag{2.3}$$

If, for a given $\rho_i(\underline{\phi}) \approx 1$, a local approximation of $f$ exists,

$$f_i(\underline{\psi}) \approx f(\underline{\psi}) \quad \text{if} \quad \rho_i(\underline{\phi}) \approx 1\,, \tag{2.4}$$

then an approximation to equation (2.3) can be formulated as

$$y \approx \hat{y} = \sum_{i=1}^{M} \rho_i(\underline{\phi})\, f_i(\underline{\psi})\,. \tag{2.5}$$

Here, the functions $f_i$ which are locally valid approximations of the global function $f$, are called *local models*. The functions in the set $\{\rho_i\}_{i=1}^{M}$ define the validity of the corresponding local models, they are thus called *validity functions*. The function $\rho_i$ has the largest value for those operating conditions where the function $f_i$ is the best approximation to $f$, and is close to

zero elsewhere. The overlap between neighbouring validity functions results in interpolation between local models. Properties of this approach to function approximation are discussed in more detail in (Johansen 1995, Johansen 1994).

### 2.2.2 Operating Regime Decomposition of Dynamic Systems

In this section we discuss the concept of local decomposition of a complex function in the context of dynamic systems. As mentioned earlier, the local models $f_i$ in equation (2.5) can generally be of any form, e.g. nonlinear or linear, in state space or in input–output description, in discrete or continuous time. They can be of different character, using physical models of the system for operating conditions where they are available, and parametric models for conditions where there is no physical description available.

We will however restrict ourselves to local *linear* descriptions, employing the concept of local linearisation. First, state space descriptions of non-linear systems are considered which can be used in both continuous and discrete time. We then discuss aspects of operating regime decomposition for discrete time input–output descriptions, focusing on the well-known NARX[1] structure (Leontaritis and Billings 1985*a*).

#### State Space Model

**Non-linear state space model** We consider the following time-invariant non-linear system in state space form,

$$\dot{\underline{x}}(t) = f(\underline{x}(t), u(t - T_d)); \quad \underline{x}(0) = \underline{x}_0 \tag{2.6a}$$

$$y(t) = g(\underline{x}(t)). \tag{2.6b}$$

Here $f()$ and $g()$ are non-linear, continuous differentiable functions. For simplicity we restrict ourselves to single input – single output system, i.e. $u \in \mathbb{U} \subset \mathbb{R}$ and $y \in \mathbb{Y} \subset \mathbb{R}$ are the input and the output of the system, respectively. The dimensionality of the state vector $\underline{x} \in \mathbb{X} \subset \mathbb{R}^n$ defines the dynamic order of the system, and $\dot{\underline{x}}(t)$ denotes the derivative of the state with respect to time, $d\underline{x}/dt$. The continuous time is denoted by $t$, and the scalar $T_d \in \mathbb{R}$ is a time-delay. The initial state at $t = 0$ is $\underline{x}_0$.

When equation (2.6a) has an explicit solution, the value of the state at time $t_2$ can be calculated as a function of the state at time $t_1$ and the time history of the input $u(t_1 - T_d \ldots t_2 - T_d)$,

$$\underline{x}(t_2) = F(\underline{x}(t_1),\, u(t_1 - T_d \ldots t_2 - T_d)). \tag{2.7a}$$

Introducing the sampling period $T_s = t_2 - t_1$ and assuming that the input is constant between sampling instances, we can obtain the discrete time description of the sampled state space

[1] Non-linear AutoRegressive with eXogenous inputs.

system (Åström and Wittenmark 1990),

$$\underline{x}(t_{k+1}) = h(\underline{x}(t_k), u(t_{k-d}))\,; \quad \underline{x}(t_0) = \underline{x}_0 \tag{2.8a}$$

$$y(t) = g(\underline{x}(t_k))\,. \tag{2.8b}$$

Here, the index $k$ denotes the $k$-th sampling instance,

$$t_k = k\,T_s\,, \qquad k = 0, 1, 2, \ldots \tag{2.9}$$

The index $_d$, $d \in \mathbb{N}$, denotes the length of the time-delay, $T_d = d\,T_s$, which is assumed to be a multiple of the sampling period.

In the following discussion we will use the continuous time system (2.6). The results obtained can be easily extended to the discrete time representation (2.8).

**Linearised state space model** A detailed discussion of aspects of the linearisation of the system (2.6) can be found in Section A.1. As outlined in Section A.1.1, the linearised system can be written as

$$\dot{\underline{x}}(t) = A\,\underline{x}(t) + \underline{b}\,u(t - T_d) + \underline{d}^x \tag{2.10a}$$

$$y(t) = \underline{c}^T\,\underline{x}(t) + d^y\,, \tag{2.10b}$$

with the bias terms $\underline{d}^x$ and $d^y$ defined as

$$\underline{d}^x = f(\underline{x}^o, u^o) - A\,\underline{x}^o - \underline{b}\,u^o \qquad \in \mathbb{R}^n \tag{2.11a}$$

$$d^y = y^o - c^T\,\underline{x}^o \qquad \in \mathbb{R}. \tag{2.11b}$$

**State space LMN** In the state space description (2.6), we have two non-linear functions, $f$ and $g$, which can be approximated by means of a local function decomposition as outlined in Section 2.2.1. Applying equation (2.5), the system can be rewritten as a weighted sum of local models,

$$\dot{\underline{x}}(t) = \sum_{i=1}^{M} \rho_i(\underline{\phi}(t))\, f_i(\underline{x}(t), u(t - T_d))\,; \quad \underline{x}(0) = \underline{x}_0 \tag{2.12a}$$

$$y(t) = \sum_{i=1}^{M} \rho_i(\underline{\phi}(t))\, g_i(\underline{x}(t))\,, \tag{2.12b}$$

which is a Local Model Network representation of the system (2.6).

Employing the concept of local linearisation for different operating conditions, we choose to work with linear local state-space representations as described by equations (2.10). This results in

$$f_i(\underline{x}(t), u(t)) = A_i\underline{x}(t) + \underline{b}_i u(t - T_d) + \underline{d}_i^x\,, \tag{2.13a}$$

$$g_i(\underline{x}(t)) = \underline{c}_i^T\,\underline{x}(t) + d_i^y\,. \tag{2.13b}$$

with $i = 1, \ldots, M$. The overall system can then be approximated as

$$\dot{\underline{x}}(t) = \sum_{i=1}^{M} \rho_i(\underline{\phi}(t)) \left[ A_i \, \underline{x}(t) + \underline{b}_i \, u(t - T_d) + \underline{d}_i^x \right] ; \quad \underline{x}(0) = \underline{x}_0 \tag{2.14a}$$

$$y(t) = \sum_{i=1}^{M} \rho_i(\underline{\phi}(t)) \left[ \underline{c}_i^T \, \underline{x}(t) + d_i^y \right] . \tag{2.14b}$$

This Local Model Network structure is depicted in Figure 2.2.

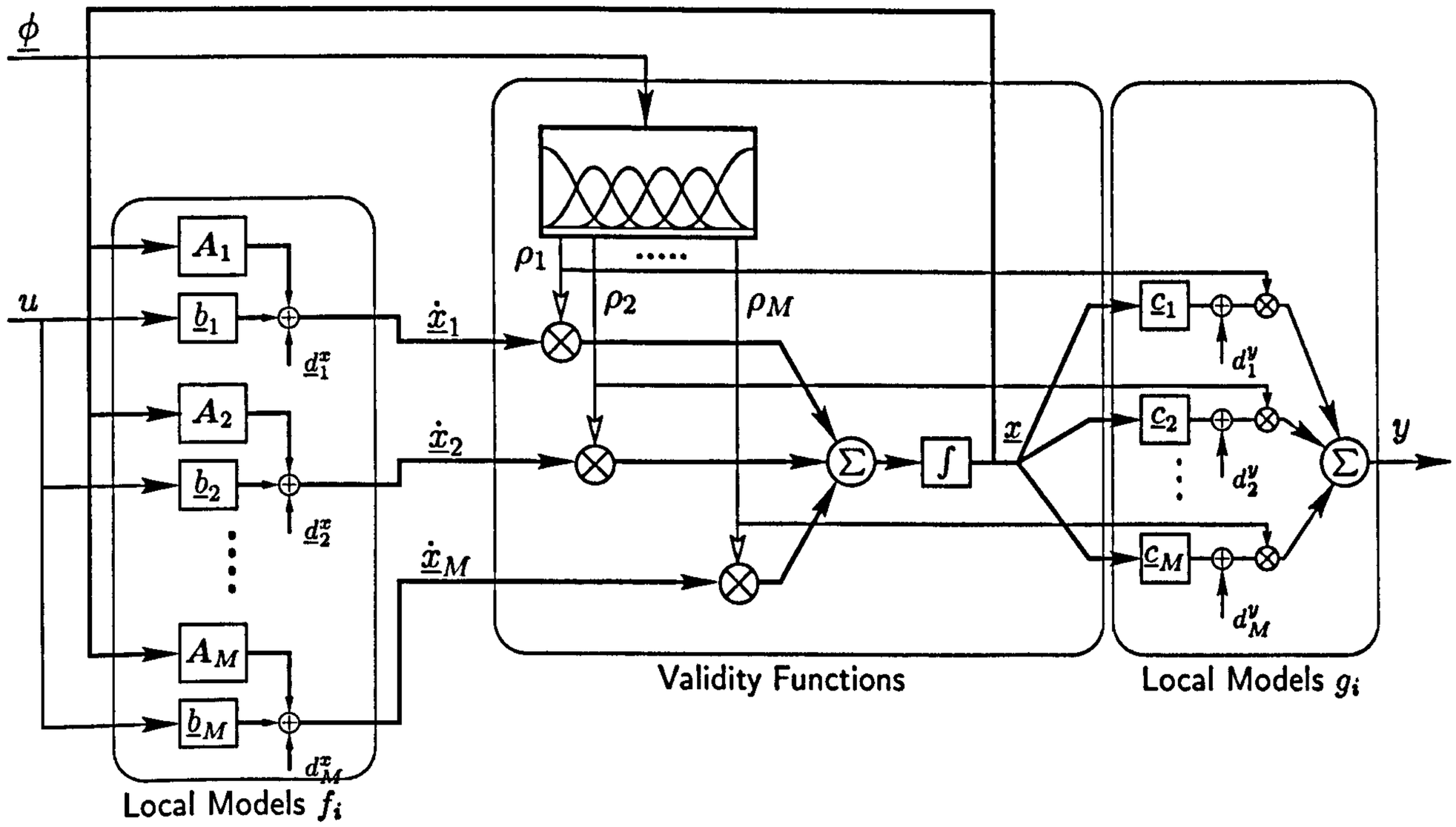



Figure 2.2: Local Model Network in state-space representation.

When using linear local models, the LMN can be described as a linear parameter-varying (LPV) system,

$$\dot{\underline{x}}(t) = A(\underline{\phi}(t)) \, \underline{x}(t) + \underline{b}(\underline{\phi}(t)) \, u(t - T_d) + \underline{d}^x(\underline{\phi}(t)) ; \quad \underline{x}(0) = \underline{x}_0 \tag{2.15a}$$

$$y(t) = \underline{c}^T(\underline{\phi}(t)) \, \underline{x}(t) + d^y(\underline{\phi}(t)) , \tag{2.15b}$$

where $A$, $\underline{b}$, $\underline{d}^x$, $\underline{c}$ and $d^y$ are the interpolated parameters of the local models,

$$\begin{aligned} A(\underline{\phi}) &= \sum_{i=1}^{M} \rho_i(\underline{\phi}) \, A_i , \quad \underline{b}(\underline{\phi}) = \sum_{i=1}^{M} \rho_i(\underline{\phi}) \, \underline{b}_i , \quad \underline{d}^x(\underline{\phi}) = \sum_{i=1}^{M} \rho_i(\underline{\phi}) \, \underline{d}_i^x , \\ \underline{c}(\underline{\phi}) &= \sum_{i=1}^{M} \rho_i(\underline{\phi}) \, \underline{c}_i , \quad d^y(\underline{\phi}) = \sum_{i=1}^{M} \rho_i(\underline{\phi}) \, d_i^y . \end{aligned} \tag{2.16}$$

The parameters of (2.15) depend only on the scheduling vector $\underline{\phi}$. Aspects of the analysis of this system will be discussed in Section 2.4.

The state space notation for LMNs introduced here uses a single global state which has the same value for all local models. Another approach is to construct LMNs where each local model has its individual state, and only the local outputs are interpolated by the validity functions, (Gawthrop 1996, Gawthrop 1995). Local state LMNs have the advantage that each local model is a real *local* approximation of the global system for its region of activity, and its dynamics are not influenced by neighbouring models. The problem in local state LMNs is, however, that the local states of models which are inactive do not influence the overall output and have therefore no physical significance.

### Input–Output Model

The state space description introduced in the previous section is the most general model description considered in this thesis. In practical applications, often only sampled input and output data of the system can be measured. It is therefore straightforward to investigate discrete time input–output model structures.

**NARX Structures** For the general discrete time state space description given in equations (2.8), we can obtain a non–linear input–output model, provided that i) the system is finite realisable (i.e. $\underline{x}$ is of finite dimensionality) and ii) that a linearisation of the system exists around its equilibrium, and that this linearisation has the full order $n$. For some restricted region of operation around the equilibrium point, the system can then be described in terms of the well-known NARX structure (Leontaritis and Billings 1985*a*, Leontaritis and Billings 1985*b*, Chen and Billings 1989),

$$y(t_k) = f(u(t_{k-d}), \dots u(t_{k-d-n_u}),\, y(t_{k-1}), \dots y(t_{k-n_y})) + e(t_k)\,. \tag{2.17}$$

Here, $f()$ is a generally non-linear, continuous function, $y(t_k)$ is the system output, $u(t_k)$ is the system input and $e(t_k)$ describes a white noise disturbance. The discrete time is expressed by $t_k$, cf. equation (2.9), and the index $_d$ denotes the length of the input delay, $T_d = dT_s$, with the sampling period $T_s$. If we restrict ourselves to single-input single-output systems, then $y(t) \in \mathbb{Y} \subset \mathbb{R}$, $u(t) \in \mathbb{U} \subset \mathbb{R}$ and $e(t) \in \mathbb{E} \subset \mathbb{R}$.

The arguments of $f()$ can be written as a single vector $\underline{\psi}$,

$$\begin{aligned} \underline{\psi}(t_k) &= [u(t_{k-d}), \dots u(t_{k-d-n_u}),\, y(t_{k-1}), \dots y(t_{k-n_y})]^T \\ &\in \mathbb{U}^{n_u+1} \times \mathbb{Y}^{n_y} \subset \mathbb{R}^{n_u+n_y+1}\,. \end{aligned} \tag{2.18}$$

Then, equation (2.17) becomes

$$y(t_k) = f(\underline{\psi}(t_k)) + e(t_k)\,. \tag{2.19}$$

When identifying a system one aims at finding a parametrised structure which approximates the unknown function $f()$ in equation (2.19).

**ARX Structure** The NARX structure, equation (2.19) with (2.18), can be linearised around an operating point $[u^o, y^o]$. The Taylor series expansion described in Section A.2 results in the linear ARX structure,

$$f(\underline{\psi}(t_k)) = \underline{\psi}^T(t_k)\,\underline{\Theta} + d^y\,, \tag{2.20}$$

where $\underline{\psi}$ is the data vector (2.18), $d^y \in \mathbb{R}$ denotes the bias term determined by the operating point (cf. equation (A.29)), and $\underline{\Theta} \in \mathbb{R}^{n_u+n_y+1}$ is the parameter vector

$$\underline{\Theta} = [\Theta_0, \ldots \Theta_{n_u}, \Theta_{n_u+1}, \ldots \Theta_{n_u+n_y}]^T\,. \tag{2.21}$$

The ARX model can also be described in transfer function form (cf. equation (A.35)), applying the forward shift operator $q$, $q^n\,y(t_k) = y(t_{k+n})$,

$$A(q^{-1})y(t_k) = q^{-d}B(q^{-1})u(t_k) + d^y + e(t_k)\,, \tag{2.22}$$

where $A$ and $B$ are polynomials in $q^{-1}$ as defined by equation (A.33) on page 124.

**Input–output LMN** A local function decomposition can be applied to the non-linear function $f$ in (2.19). Using equation (2.5), it can be approximated by a set of interpolated local models,

$$f(\underline{\psi}(t_k)) \approx \sum_{i=1}^{M} \rho_i(\underline{\phi}(t_k))\, f_i(\underline{\psi}(t_k))\,. \tag{2.23}$$

If we choose to work with local linear ARX models, equation (2.20), we obtain a Local Model Network of the form

$$y(t_k) = \sum_{i=1}^{M} \rho_i(\underline{\phi}(t_k))\,[\underline{\psi}^T(t_k)\,\underline{\Theta}_i + d_i^y] + e(t_k) \tag{2.24}$$

This Local Model Network structure is depicted in Figure 2.3.

When we work with linear local models, the LMN (2.24) can be rewritten as a linear parameter-varying (LPV) system,

$$y(t) = \underline{\psi}^T(t_k)\,\underline{\Theta}(\underline{\phi}(t_k)) + d^y(\underline{\phi}(t_k)) + e(t_k)\,, \tag{2.25}$$

where $\underline{\Theta}$ and $d^y$ are the interpolated parameters of the local models,

$$\underline{\Theta}(\underline{\phi}) = \sum_{i=1}^{M} \rho_i(\underline{\phi})\underline{\Theta}_i\,, \quad d^y(\underline{\phi}) = \sum_{i=1}^{M} \rho_i(\underline{\phi})d_i^y\,. \tag{2.26}$$

The parameters of (2.25) depend only on the scheduling vector $\underline{\phi}$. Aspects of the analysis of this system will be discussed in Section 2.4.

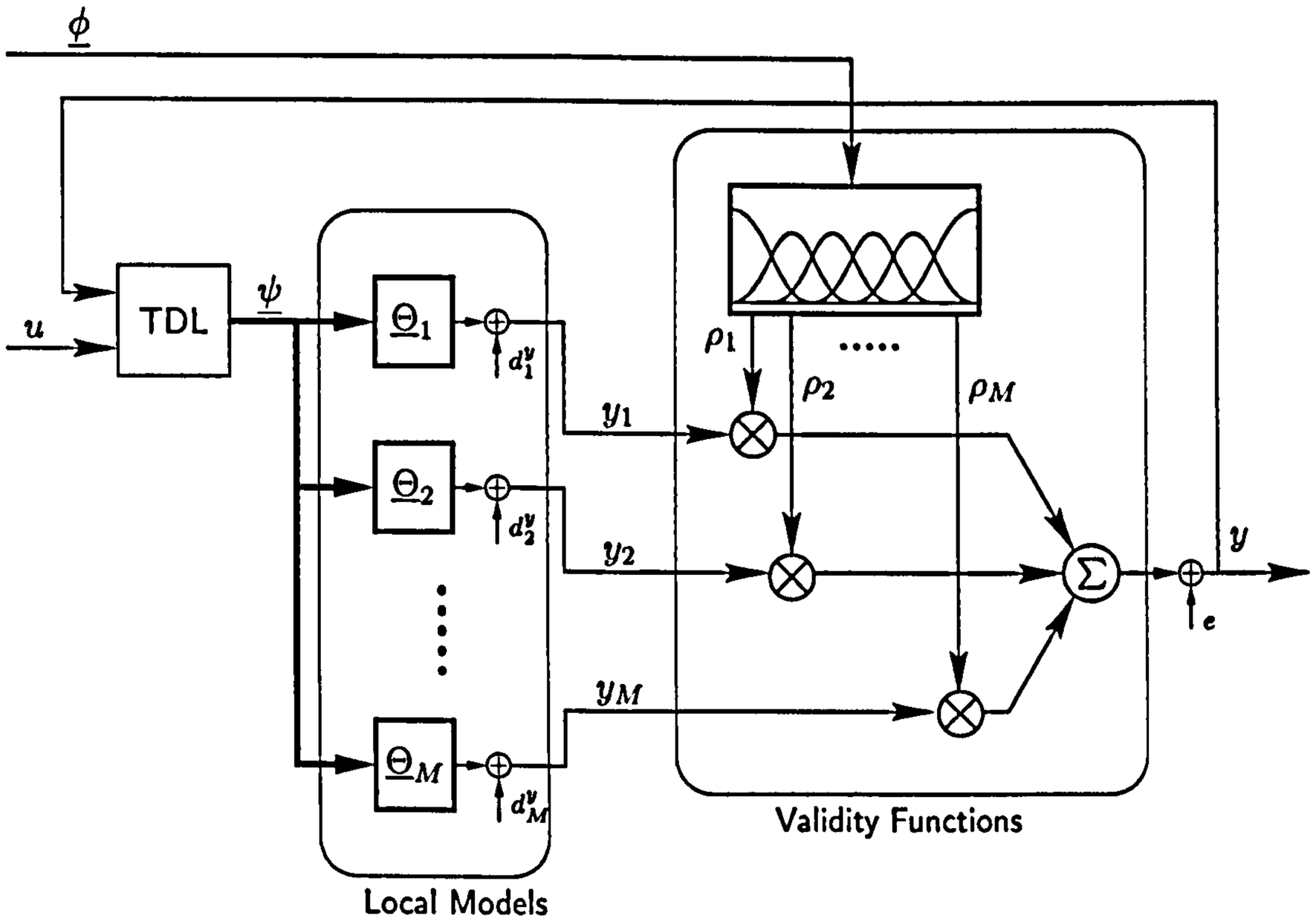



Figure 2.3: Local Model Network in Input–Output description. TDL denotes a tapped delay line which implements equation (2.18).

### 2.2.3 Validity Functions

To satisfy the approximation described in equation (2.5) we require the following properties from the set of validity functions:

a) A validity function transforms its input to a value between 0 and 1:

$$\{\rho_i : \Phi \rightarrow [0, 1]\}_{i=1}^{M} \tag{2.27}$$

b) The activation of a validity function decreases with increasing distance of the input from its maximum (its 'centre'). The activation converges to zero for inputs which are far away from the 'centre'.

c) The set of validity functions forms a partition of unity of its input space, i.e.

$$\sum_{i=1}^{M} \rho_i(\underline{\phi}) = 1 \quad \forall \underline{\phi} \in \Phi \subset \mathbb{R}^{n_\phi} \tag{2.28}$$

This ensures that every point in the input space is covered to the same degree.

d) The shape of the validity functions is smooth.

Although any function with the properties listed above could be applied as a validity function, the most popular choices include Gaussian bells (Hlaváčková and Neruda 1993, Johansen and Foss 1993, Murray-Smith 1994, Shorten and Murray-Smith 1997), B-splines (Brown and Harris 1994), e.g. the ASMOD[2] structure (Kavli 1993, Kavli and Weyer 1995, Weyer and Kavli 1997) and MARS[3] (Friedman 1991), and Kernel functions, e.g. KBFs[4] (Hlaváčková 1995).

In this work we will restrict ourselves to the use of B-splines.

### B-splines

A set of B-splines can be defined recursively as follows (de Boor 1978): The j-th first order B-splines consists of zero order polynomials,

$$B_{j,1}(x) = \begin{cases} 1 & \text{if } \tau_j \leq x < \tau_{j+1} \\ 0 & \text{otherwise} \end{cases} , \tag{2.29}$$

where $x \in \mathbb{R}$. B-splines of order $p$ with $p > 1$ are then defined as,

$$B_{j,p}(x) = \frac{x - \tau_j}{\tau_{j+p-1} - \tau_j} B_{j,p-1}(x) + \frac{\tau_{j+p} - x}{\tau_{j+p} - \tau_{j+1}} B_{j+1,p-1}(x) . \tag{2.30}$$

Thus, a B-spline of order $p$ is a composition of $p$ polynomials of order $p-1$. A set of $M$ B-splines is defined by the location of $M+p$ knots $\tau_j$. Examples for B-splines of order 1 to 4 are shown in Figure 2.4.

As we require the validity functions to be smooth we choose to work with 3rd order B-splines which are composed of quadratic polynomials. Thus, the location and the shape of the $M$ validity functions are defined by the set of knots $\{\tau_j\}_{j=1}^{M+3}$. Examples of 3rd order B-splines with uniformly and non-uniformly distributed knots are shown in Figure 2.5. Although the shape of the validity functions changes significantly when the location of the knots changes, each validity functions is exactly defined by a limited number of knots (in this case four). This rules out the possibility of reactivation and loss of locality as observed with normalised Gaussian bells (Shorten and Murray-Smith 1997).

B-splines are by definition one-dimensional functions, $B_{j,p} : \mathbb{R} \to \mathbb{R}$. They can be extended to cover a multi-dimensional input space by using the tensor product (Friedman 1991, Kavli 1993).

[2] Adaptive Spline Modeling of Observation Data
[3] Multivariate Adaptive Regression Splines
[4] Kernel Basis Functions

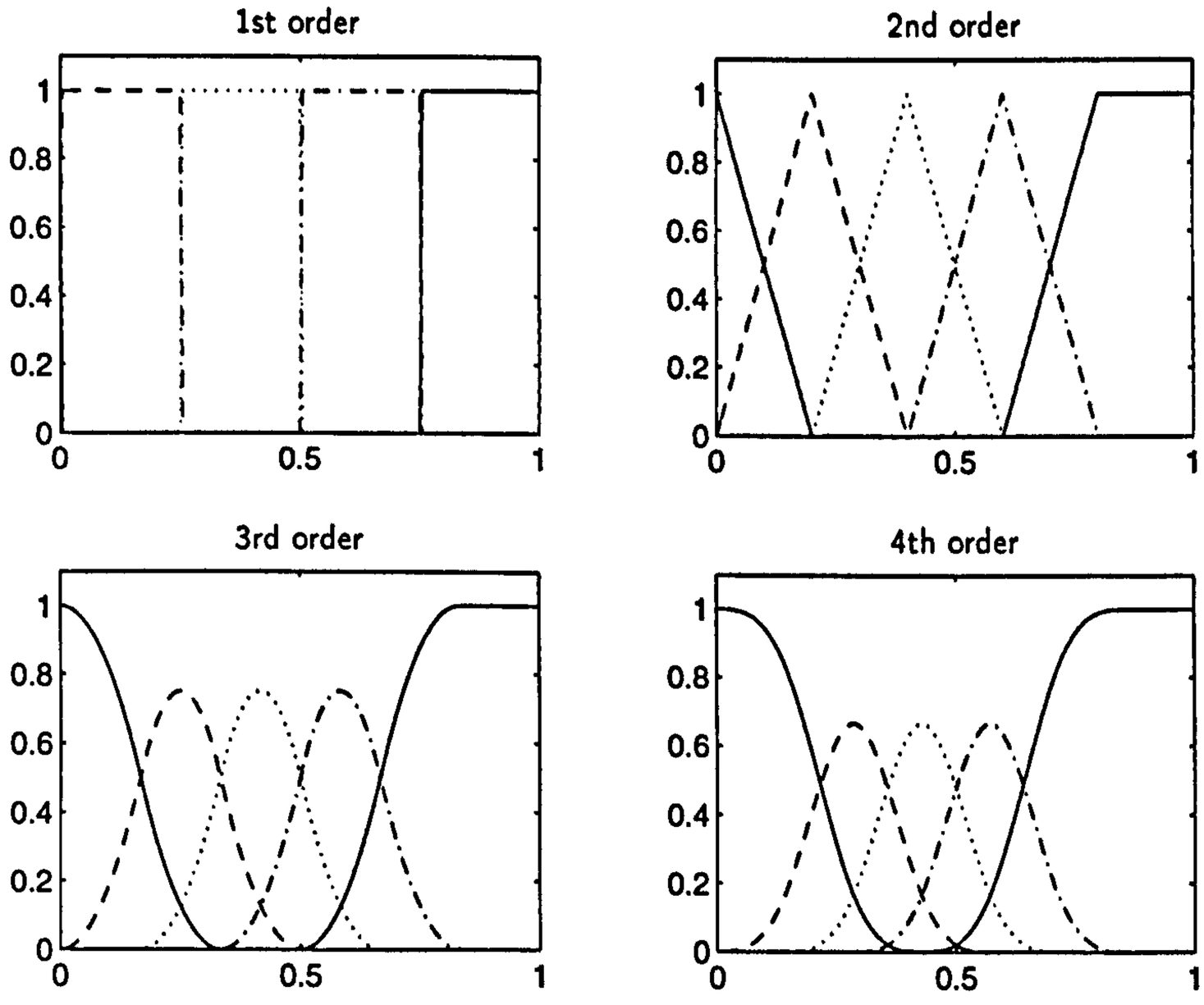
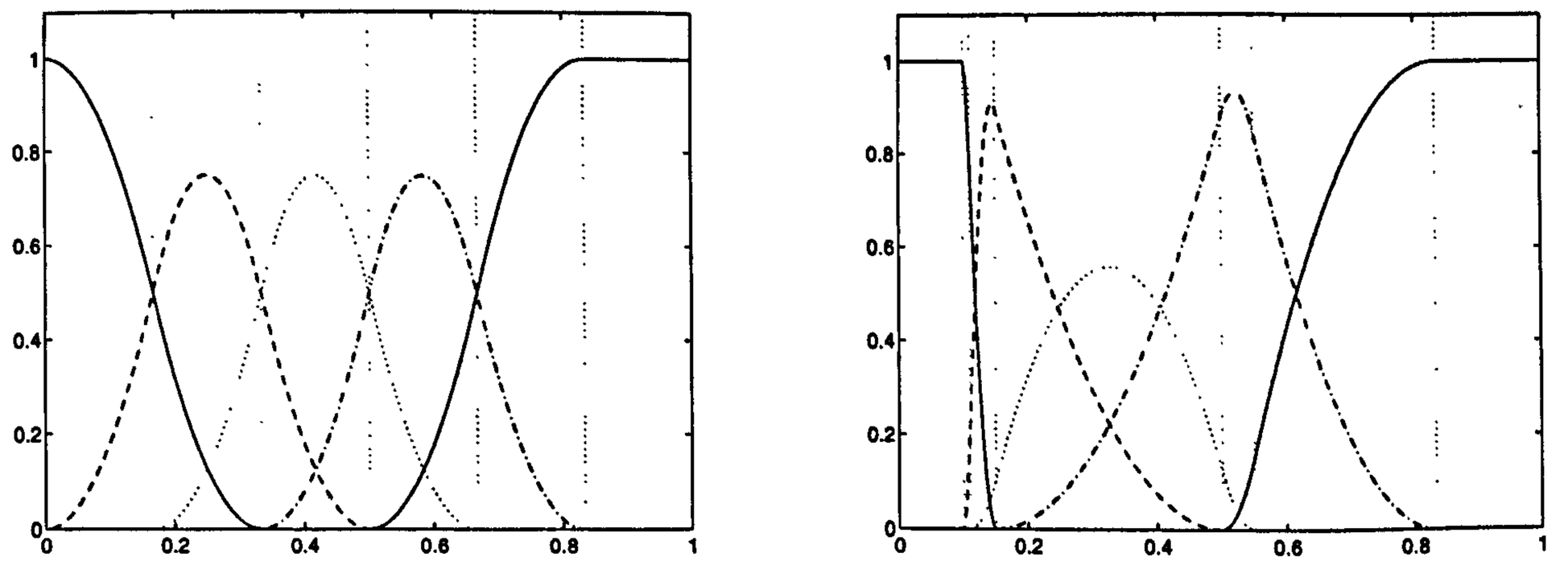



Figure 2.4: Examples for B-splines.




(a) uniform knot location (b) non-uniform knot location

Figure 2.5: Example of 3rd order B-splines. The vertical dotted lines mark the knot locations.

### 2.2.4 Choice of the Scheduling Vector

We introduced a scheduler of the form $\{\rho_i(\underline{\phi})\}_{i=1}^{M}$ and discussed some properties of it in the previous section. However, we did not specify the scheduling vector $\underline{\phi} \in \Phi \subset \mathbb{R}^{n_\phi}$. In this section, we will discuss how the elements of this scheduling vector can be selected.

From the discussion in the previous sections it is clear that the scheduling vector should in some way reflect changes of the system characteristic which have the effect that the current local model is not valid any more, and a different local model should be activated. For linear local models such a change of system characteristics means that the operating regime has moved too far away from the operating point for which the current local model has been linearised. In this case, the scheduling vector should reflect the significant non-linearities of the system.

A change of the system's characteristics is linked to a change of the operating regime of the plant. A straightforward approach is therefore to select the elements of the scheduling vector in such a way that the current operating regime is represented by it. For state space and for input-output representations we obtain the following vectors:

- **State space:** The operating regime is determined by the current input $u(t-T_d)$ and the state $\underline{x}(t)$. Thus, the scheduling vector becomes

$$\underline{\phi}(t) = \left[u(t-T_d),\, \underline{x}^T(t)\right]^T \qquad \in \Phi \subset \mathbb{R}^{n+1} \tag{2.31}$$

- **Input–Output:** The operating regime is completely defined by the data vector $\underline{\psi}$, equation (2.18). Thus, the scheduling vector can be chosen as

$$\underline{\phi}(t_k) = \underline{\psi}(t_k) = [u(t_{k-d}), \ldots u(t_{k-d-n_u}),\, y(t_{k-1}), \ldots y(t_{k-n_y})]^T \qquad \in \Phi \subset \mathbb{R}^{n_u+n_y+1}\,. \tag{2.32}$$

In both cases, defining the scheduling vector in this way will generally lead to a scheduling space whose dimensionality is higher than necessary, because not all the dimensions of the scheduling vector might have a non-linear impact on the system. As we will discuss in Section 2.3.1, the major effort in the learning process of an LMN structure is spent on the optimisation of the parameters of the scheduler. Thus, reducing the dimensionality of the scheduling space by pruning the scheduling vector can significantly simplify the learning process. The use of *a priori* knowledge about the modelled system is of great importance here.

We will therefore discuss some aspects of pruning elements of the scheduling vector.

#### Including the input in the scheduling vector

If the system input $u(t-T_d)$ (and possibly additional delayed values of it) is part of the scheduling vector, it enters the model both through the non-linear scheduler and as an input

of the local models. As the scheduler can perform a many-to-one mapping, it is possible for the overall model structure to perform such a mapping, i.e. to have the same output behaviour for different input sequences under identical initial conditions.

If the input is excluded from the scheduling vector, it enters the model only as an input of the local models. As these local models are linear they perform a one–to–one mapping. Hence, the overall system will perform a one–to–one mapping with respect to the input: different input sequences will necessarily lead to different model output patterns, assuming the same initial conditions.

Thus, including the input in the scheduling vector leads to a more general model structure.

#### Including feedback elements in the scheduling vector

If feedback elements (i.e. the state, or delayed values of the output) are included in the scheduling vector, changes of the process (for example, an overall change of the gain due to fatigue) requires readjustment of both the parameters of the local models and the parameters of the scheduler.

If feedback elements are excluded from the scheduling vector, limited changes of the plant require only readjustment of the parameters of the local models. The parameters of the scheduler can remain unchanged.

As outlined in Section 2.3.1, the adaptation of the parameters of the scheduler can be very computationally expensive. Thus, excluding feedback elements from the scheduling vector simplifies the adaptation of the model to a changing process.

#### Scheduling on additional variables

So far, we have only discussed the use of a subset of the full vectors, (2.31), (2.32), for scheduling. Depending on the system to be modelled it might be useful to take additional variables as potential elements for the scheduling vector into account:

- **Filtered inputs:** The input of the process can often change very rapidly. When inputs are directly used for scheduling, this leads to an abrupt switch to a different local model. It is usually desirable to change the model characteristics more smoothly. Employing a low-pass filtered version of the input for scheduling can improve the switching characteristics of the model significantly.

- **Output derivative:** Many physical systems change their characteristics with the rate of change of their output. In particular, it is quite often the case that the system behaves differently when the output increases than when it decreases. If only the delayed outputs are included in the scheduling vector, equation (2.32), such a behaviour will be difficult to model. Including the output derivative as an element of the scheduling vector can improve the model performance significantly.

- **Auxiliary inputs:** Sometimes the characteristics of a system change with variables which do not represent direct inputs to the process themselves. A chemical reaction, for example, can depend on the temperature. Although the temperature does not act as an input to the reaction, and thus will not be an input to the local models, it is important as an element of the scheduling vector. Thus, it might be necessary to include auxiliary variables in the scheduling vector which are not inputs to the local models.

Example E.4.1 on page 152 illustrates the selection of the elements of the scheduling vector with the heat transfer process. [5]

## 2.3 Learning

The learning process in Local Model Networks can be divided into two tasks,

i. to find the optimal number, position and shape of the validity functions, i.e. to learn the *structure* of the network.

ii. to find the optimal set of parameters for the local models.

The learning process is usually iterative: after defining the network structure, the parameters of the local models are optimised. The structure is then refined, and the local model parameters are updated, and this is repeated until the parameters have converged (or until a maximal number of iterations has been reached).

General aspects of parameter optimisation for dynamic systems are discussed in Section B.1.

### 2.3.1 Structural Learning

The aim of structural learning is to adapt the number, position and shape of the validity functions to the complexity of the system.

The following approaches for the optimisation of the structure of non-linear dynamic systems are described in (Haber and Unbehauen 1990) and can be adapted for Local Model Networks:

- **Forward regression:** The model structure grows according to the complexity of the system. For LMNs, clustering techniques can be used to place the centres of the validity functions. With unsupervised clustering, which is an approach well-known from pattern classification, the validity functions are placed according to the complexity and the availability of data. Such techniques are, however, unable to take account of the

[5] The heat transfer process introduced in Appendix E is used as a simple system to illustrate the modelling and control principles throughout Part I of this thesis.

complexity of the local models. This can be overcome by using iterative clustering techniques (Murray-Smith and Gollee 1994, Murray-Smith 1994). Here, the data are initially estimated with a model of low complexity, and new validity functions are subsequently added where they are needed (e.g. where the modelling error is large).

- **Backward regression (pruning):** A complex model is used as an initial structure for a model reduction algorithm, through which an attempt is made to extract the essence of the system by pruning less significant parameters (Reed 1994, Jutten and Fambon 1995). The number of validity functions can be reduced for example by using cluster merging techniques (Krishnapuram and Freg 1992, Kaymak and Babuška 1995), also (Babuška and Verbruggen 1997) and (Gollee and Hunt 1997). The simplest way to choose the initial structure is to place one validity function at each data point. For most real world tasks this will be too expensive and therefore not practical. Another way of defining the initial structure is to place a large number of validity functions uniformly in the scheduling space.

Backward regression algorithms are usually relatively simple and straightforward. They are, however, based on the definition of a complex initial structure. This limits their application to problems with low dimensional scheduling space, as the number of validity functions will rise exponentially with the number of dimensions of the scheduling space.

Forward regression does not rely on a complex initial structure and can therefore be applied even when the scheduling space is of high dimension. The clustering techniques used to optimise the structure are, however, usually complicated and computationally expensive.

Forward and backward regression techniques are primarily aimed at defining the size of a model structure. Techniques to optimise the structural parameters, i.e. the parameters which define shape and location of the validity functions are often used in addition.

The optimal location and shape of the validity functions can be found by optimising their parameters directly using some standard, non-linear optimisation algorithm. Such techniques are described in (Takagi and Sugeno 1985, Sugeno and Kang 1988, Sugeno and Tanaka 1991), see also (Jang 1993). The direct optimisation of the validity functions parameters is particularly straightforward to apply when working with B–splines, as their shape and location is defined by a single set of parameters, the knots vector.

### 2.3.2 Optimisation of Local Model Parameters

The most straightforward way to optimise the parameters of the local models is to use the criterion (B.2) on page 126 with the weighting factor $\gamma$ equal to one,

$$J(\underline{\theta}_g) = \sum_{i=1}^{N} \left[ y(t_i) - \hat{y}(t_i, \underline{\theta}_g) \right]^2 . \tag{2.33}$$

Here, $\hat{y}$ is the global LMN output, and $\underline{\theta}_g$ includes the parameter vectors $\{\underline{\theta}_m\}_{m=1}^M$ of all local models,

$$\underline{\theta}_g = [\underline{\theta}_1^T, \ldots, \underline{\theta}_M^T]^T \ . \tag{2.34}$$

The global optimisation criterion (2.33) ensures that the overall model performance is optimal. However, as pointed out in (Murray-Smith and Johansen 1995, Murray-Smith and Johansen 1997*b*), global learning can give rise to problems when the network is locally over-parameterised or poorly structured[6]. Large overlap between neighbouring validity functions can lead to model parameters where a negative contribution of one local model is compensated for by a positive contribution of the neighbouring local model. While such a parameter combination might minimise the overall error, interpretability of the local models is lost, and it can lead to poor generalisation.

To overcome these problems, (Murray-Smith and Johansen 1997*b*) suggest the use of a *local optimisation criterion*, where the parameters of each local model are estimated independently. The idea is to make the output of a local model match the desired system output when this model is active. A weighted least squares optimisation criterion is defined for each local model, where the weighting factor is the current activation of the corresponding validity function,

$$J(\underline{\theta}_m) = \sum_{i=1}^{N} \rho_m(\underline{\phi}(t_i)) \ [y(t_i) - \hat{y}_m(t_i, \underline{\theta}_m)]^2 \,, \qquad m = 1, \ldots, M \,. \tag{2.35}$$

Here, $\hat{y}_m$ denotes the output of the $m$–th local model, and $\underline{\theta}_m$ is the corresponding local parameter vector.

While local learning will generally not find parameter sets which give as good an overall performance as can be achieved with parameters found using global optimisation, the local models are often more valid as local approximations. Using the local optimisation criteria has a regularising effect: the degrees of freedom are effectively reduced, which gives a reduced variance at the possible cost of an increased bias in error and in the parameter estimate (Murray-Smith and Johansen 1997*b*).

The choice of the optimisation technique which is used to minimise the criteria (2.33) and (2.35) depends on the way the model output, $\hat{y}$, is obtained, cf. Section B.1. For LMNs with linear local models which are simulated with a one–step ahead prediction horizon, equations (B.4) and (B.5), the model output depends on the local model parameters in a linear way. Thus, linear regression techniques, as described in Section B.2, can be applied to optimise the parameter vectors.

If we work with an infinite prediction horizon, equations (B.6) and (B.7), the parameter optimisation will be a non-linear problem owing to the non-linear scheduler, even when the local models are linear. Non-linear optimisation techniques as introduced in Section B.3 have

[6] An example for such a structural mismatch between the model and the real system is a model structure with a high concentration of local models in an area of low complexity.

to be used. The Levenberg–Marquardt method described there was found to perform in a very robust and efficient way for the parameter optimisations carried out in this work.

Examples E.4.3 on page 153 and E.4.4 on page 154 illustrate the use of local and global learning for parameter estimation in prediction and in simulation mode. The results obtained in these examples show i) that local learning is only of advantage when working with prediction model simulation for the parameter estimation and the network structure is large, and ii) that the simulation model results are significantly better than the prediction model results.

## 2.4 Analysis and Validation of Local Model Networks

An important aspect of empirical modelling is the external validation of the model, i.e. the process of checking whether the model represents all the characteristics of the real system which are important for the intended application (Murray-Smith 1995). A common approach to deal with this is to use cross–validation (Weiss and Kulikowski 1991) which in the simplest case means optimising the model parameters using one set of training data, and verifying the resulting model using a different set of test data. This approach has been employed in the examples shown so far by comparing the mean squared modelling error (MSE) on the data set used for training the model parameters with the MSE on a test data set which was not used for training. The exclusive use of cross–validation requires that all system information is contained in the data, which is difficult to ensure in practice without a very large amount of data. This problem becomes more important when the system is highly non-linear.

Another approach to validate a model is to analyse its properties (e.g. steady state gain, location of poles) and to check whether these properties correspond to characteristics of the real system estimated in other ways.

In this section we will introduce some tools for analysis of the properties of Local Model Networks, and discuss their limitations.

### 2.4.1 Local Model Networks as Linear Parameter Varying Systems

When the concept of operating regime decomposition was introduced for dynamic systems in Section 2.2.2, it was shown that Local Model Networks with linear local models can be represented as linear parameter-varying (LPV) systems. The corresponding equations are (2.15) for state space descriptions, and equation (2.25) when we choose to work with local ARX models. The following discussion will be focussed on the state space description of an LPV system.

Although we are dealing with *linear* parameter-varying systems, tools available from linear time-invariant systems theory can generally not be applied as no general explicit solution exists for an LPV system with order greater than one (Kailath 1980). Linear system analysis techniques are therefore of very limited usefulness for such systems, and more complex analysis

approaches have to be used.

### Lyapunov Stability

The investigation of stability is an important aspect for system identification and for controller design. When identifying a model of a system which is known to be stable, we expect the model to be stable, too. For controller design, the most basic property required from the closed loop system is stability. Thus, being able to analyse stability of a system is a significant issue.

A large number of stability definitions exist in system theory. The two basic approaches to be distinguished are Lyapunov stability (La Salle and Lefschetz 1961, Hahn 1963) and input–output stability (Zames 1966*a*, Zames 1966*b*). Whereas Lyapunov stability is concerned with the internal stability of the system, i.e. whether the state of the system remains stable, input–output stability deals with the external stability of the system, i.e. whether the system output is bounded for bounded inputs. As input–output stability becomes equivalent to Lyapunov stability if the system under study is stabilisable and detectable (Shorten 1996), we will restrict our discussion to Lyapunov stability. A more detailed discussion of stability can be found in e.g. (Vidyasagar 1993, Slotine and Li 1991, Brockett 1970).

**Definition 2.4.1** *(Slotine and Li 1991) The equilibrium state* $\underline{x} = 0$ *of equation (2.36) is said to be* stable *if, for* $R > 0$*, there exists* $r > 0$*, such that if* $||\underline{x}_0|| < r$*, then* $||\underline{x}(t)|| < R$ *for all* $t \geq t_0$*. Otherwise the equilibrium point is* unstable.

This definition of stability *in the sense of Lyapunov* means that the system trajectory can be kept arbitrarily close to the origin by starting sufficiently close to it.

In many engineering applications, it is not sufficient that the state does not leave the region defined by $R$, but it is required that the state returns asymptotically to the origin. This type of stability is referred to as asymptotic stability.

**Definition 2.4.2** *(Slotine and Li 1991) An equilibrium point* $\underline{x} = 0$ *is* asymptotically stable *if it is stable, and if in addition there exists some* $r > 0$ *such that* $||\underline{x}_0|| < r$ *implies that* $\underline{x}(t) \to 0$ *as* $t \to \infty$.

The above definitions are concerned with the internal stability of the unforced dynamic system described by equation (2.36). They form the basis for stability analysis using *Lyapunov's direct method* (Hahn 1963, La Salle and Lefschetz 1961). The basic principle of this method can informally be stated as follows.

**Theorem 2.4.1** *(Kalman and Bertram 1960) A dynamic system (2.36) is stable in the sense of Lyapunov if and only if there exists a* Lyapunov function, *i.e., some scalar function* $V(\underline{x})$ *of the state with the properties:*

*a)* $V(\underline{x}) > 0$, $\dot{V}(\underline{x}) \leq 0$ *when* $\underline{x} \neq 0$, *and*

*b)* $V(\underline{x}) = \dot{V}(\underline{x}) = 0$ *when* $\underline{x} = 0$.

*The system is asymptotically stable if* $\dot{V}(\underline{x}) < 0$ *when* $\underline{x} \neq 0$.

### Stability of Linear Parameter-Varying Systems

As outlined at the beginning of this section, no general explicit solution of the set of differential equations of a LPV system exists. Thus, results obtained for linear systems are no longer valid. In particular, the stability of an LPV system is no longer completely determined by its eigenvalues[7] (Slotine and Li 1991). An intuitive explanation for this can be given as follows (Shorten 1996, Shorten and Narendra 1998): For some constant parameters, the state of the system may experience an amplification phase before tending to the equilibrium. If during the amplification phase the system parameters change, the state may again be amplified. If the net growth of the state continues over successive switching intervals then the state will become unbounded.

There exist a number of stability criteria for LPV system which are limited to special cases. Some examples include:

a) If a common Lyapunov function for the LPV system can be found, stability is guaranteed. For Local Model Networks, approaches based on piecewise quadratic Lyapunov functions have been reported (Johansson and Rantzer 1998). These techniques require the solution of linear matrix inequalities.

b) Stability can be shown for *slowly varying systems* (Hahn 1963, Shamma 1988, Vidyasagar 1993). If limits are placed on the rate of change of the parameters, it can be possible to guarantee stability by finding a Lyapunov function for the system. The difficulty is, however, to determine the limitation of the rate of change. For use with Local Model Networks, this restriction implies a slowly varying scheduling vector.

c) Another class of systems for which stability can be shown are *parameter bounded systems*. By parameter bounded systems we mean systems with a principal linear parameter-invariant part which is perturbed by a parameter-varying component. Stability can be shown if the parameter-varying component is of limited significance (Hahn 1963). With Local Model Networks this implies that the local models are similar to each other.

d) Strong statements about stability can be made for *systems with periodic coefficients* (Åström and Wittenmark 1989, Brockett 1970, Vidyasagar 1993). However, the usefulness of these results is questionable since they involve the calculation of the state transition matrix (Shorten 1996). Local Model Networks with a periodic scheduling vector belong to this class of systems.

[7] We recall that a linear continuous time-invariant system is asymptotically stable if all its eigenvalues are located in the left half of the complex plane.

e) For *LPV systems with real eigenvectors and eigenvalues*, an approach to analyse stability which is based on geometrical interpretation of time-varying (in)stability is reported in (Shorten 1996, Shorten and Narendra 1998). The restriction to real eigensystems excludes Local Model Networks with local models which have imaginary eigenvalues or eigenvectors.

For Local Model Networks, the restrictions required for most of these stability criteria are very strong. Thus, most of these methods cannot be used for the applications investigated in this work. However, a restriction to real eigensystem proved to be appropriate for the application investigated in Part II of this thesis. We will therefore introduce the basic concepts of method e) below.

**Stability of Real Eigensystems** We consider autonomous LPV system

$$\dot{\underline{x}}(t) = \boldsymbol{A}(t)\underline{x}(t)\,, \qquad \underline{x}(t_0) = \underline{x}_0 \tag{2.36}$$

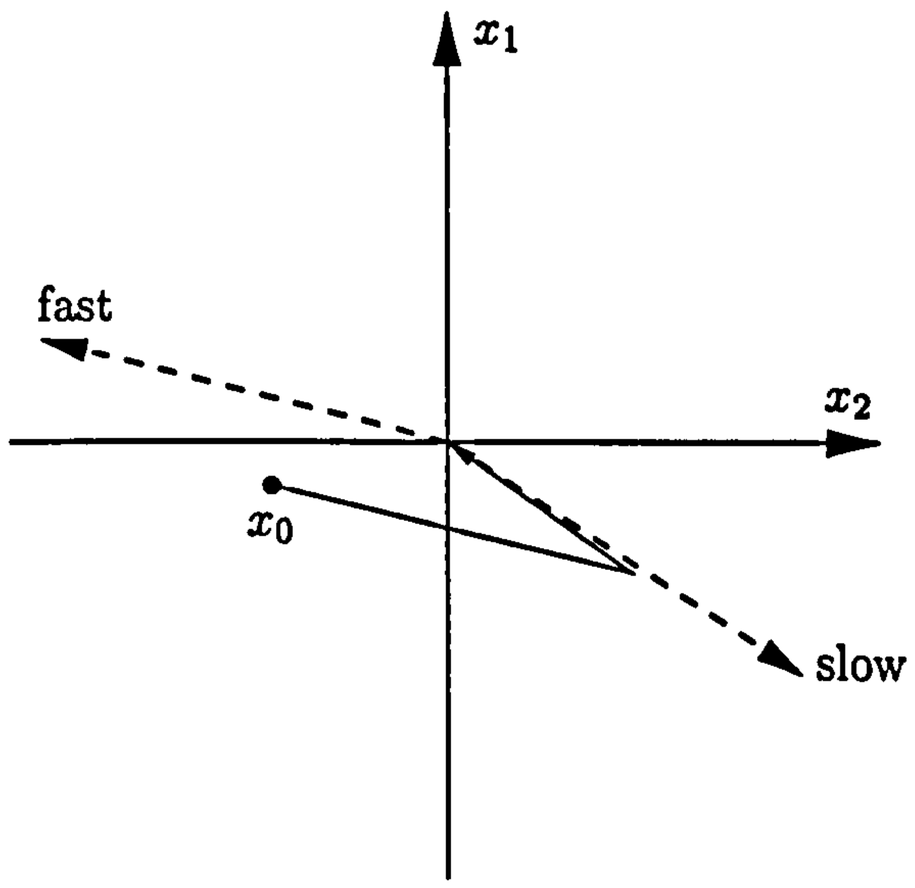
with $\boldsymbol{A}(t)$ being an asymptotically stable matrix (i.e. all its eigenvalues are located in the left half of the complex plane) which has real eigenvalues and eigenvectors for all $t$. The evolution of the state can now be described as follows:

- For a constant matrix $\boldsymbol{A}$, the movement of the state vector $\underline{x}$ is determined by the vector field created by the eigenvalues and eigenvectors of the system: each eigenvalue can be interpreted as a field component in the direction of the corresponding eigenvector. In the case where the magnitude of one eigenvalue is much greater (the "fast" eigenvalue) than the other (the "slow" one) the state will move parallel to the fast eigenvector towards the slow eigenvector, and then towards the origin along the slow eigenvector. An example of such a movement is depicted in Figure 2.6(a).

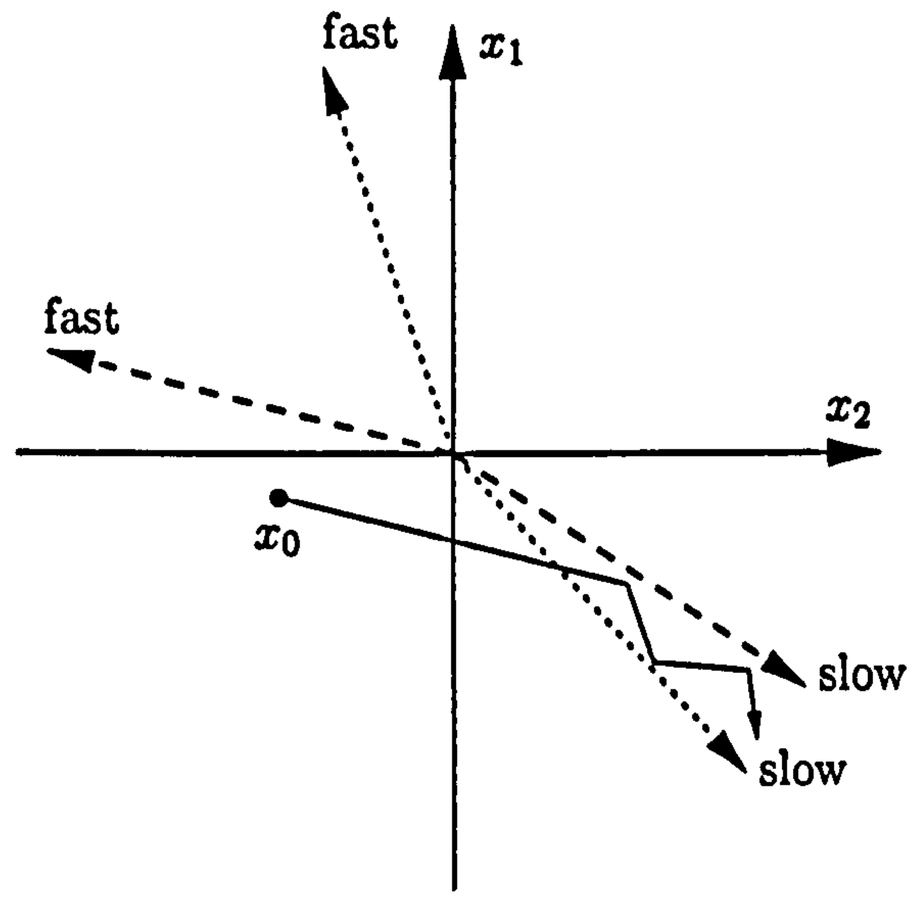
  When observed from the Cartesian coordinate system, the Euclidean distance of the state from the origin (the magnitude of the state vector) may increase before it decays to zero at the origin. In terms of the coordinate system formed by the eigenvectors the state decays monotonically to zero.

- For a parameter-varying system, the direction of the eigenvectors will change with time with respect to the Cartesian coordinate system. If such a change occurs while the magnitude of the state increases, the state could be amplified further by the new eigenvectors. Thus, repeated switching of the eigenvectors can lead to instability. An example is given in Figure 2.6(b).

In (Shorten 1996, Shorten and Narendra 1998), various conditions are outlined for which stability of the system can be proven. The most straightforward requirement is to have constant eigenvectors for all $t$, and to let only the eigenvalues of the matrix $\boldsymbol{A}$ vary. This condition ensures that the overall system will be asymptotically stable, provided that all




(a) Parameter-invariant system. The magnitude of the state increases initially, but moves asymptotically towards the origin.




(b) Parameter-variant system. The state vector first moves parallel to the first fast eigenvector towards the slow eigenvector. After switching it moves parallel to the second fast eigenvector towards the other slow eigenvector. Switching occurs when the state reaches the slow eigenvector of each system.

Figure 2.6: Graphical interpretation of state evolution for a 2nd order system. The state starts at $x_0$. The eigenvectors associated with the fast eigenvalues are labelled "fast", those related to the slow eigenvalues are marked "slow". The state trajectory is depicted only schematically.

eigenvalues of $A(t)$ are in the left half of the complex plane. Although this requirement is rather conservative, it proves to be easy to implement for system identification.

The use of constant eigenvectors is illustrated with a simple heat transfer process in Example E.4.5 on page 154.

### 2.4.2 Static Analysis of Local Model Networks

In the previous section, ways to analyse Local Model Networks as linear parameter-varying systems have been outlined. It was shown that linear system analysis techniques are generally not appropriate to interpret the Local Model Network. In this section we will analyse properties of the Local Model Network for fixed values of the scheduling vector, interpreting each system obtained this way as a linear parameter-invariant (LPI) system. Although the results of such analysis do not reflect all properties of the underlying LPV system and should therefore be interpreted with great care, we can still obtain valuable insight in terms of model

validation (Gollee and Murray-Smith 1997*a*, Gollee and Murray-Smith 1997*b*).

We recall that in Section 2.2.2 we obtained LPV descriptions for LMNs based on state space description and for LMNs based on local ARX models, equations (2.15) and (2.25). Based on these descriptions we can obtain properties of the interpolated LPI system as functions of the scheduling vector. Such properties include the gain, the location of the system poles, the bias, and the activation of the validity functions. Analysis of the change of these properties with changing scheduling vector can be useful in two ways:

- Do the changes of the model properties reflect the characteristics of the real system? When relating properties of the model to characteristics of the real system, one has to be aware of the fact that the LPI systems do not represent linearisations of the system in the classical sense, i.e. linearisations around equilibria.

- Do the model properties outside the region covered by the training data let us expect that the model's ability to generalise will be good? Related to this point is the detection of possible over-parametrisation of the model structure.

These points are illustrated with the heat transfer process in Example E.4.6 on page 155.

# 3 Local Control

## 3.1 Introduction

In the previous chapter, modelling techniques were introduced which are based on a "divide and conquer" strategy. In this chapter, we will review similar approaches for the design of controllers for non-linear systems.

### 3.1.1 The Control Problem

The problem of controlling a plant (or a process) can be formulated as follows: Depending on the *reference signal* $r$ (and subject to a possible *feedback* signal), the controller delivers some *control signal* $u$ which, when used as the input signal to the plant, makes the *plant output* $y$ follow the reference signal $r$ in some specified way. Note that the plant is, in general, subject to disturbances $d$ and $e$. A general control structure is shown in Figure 3.1.

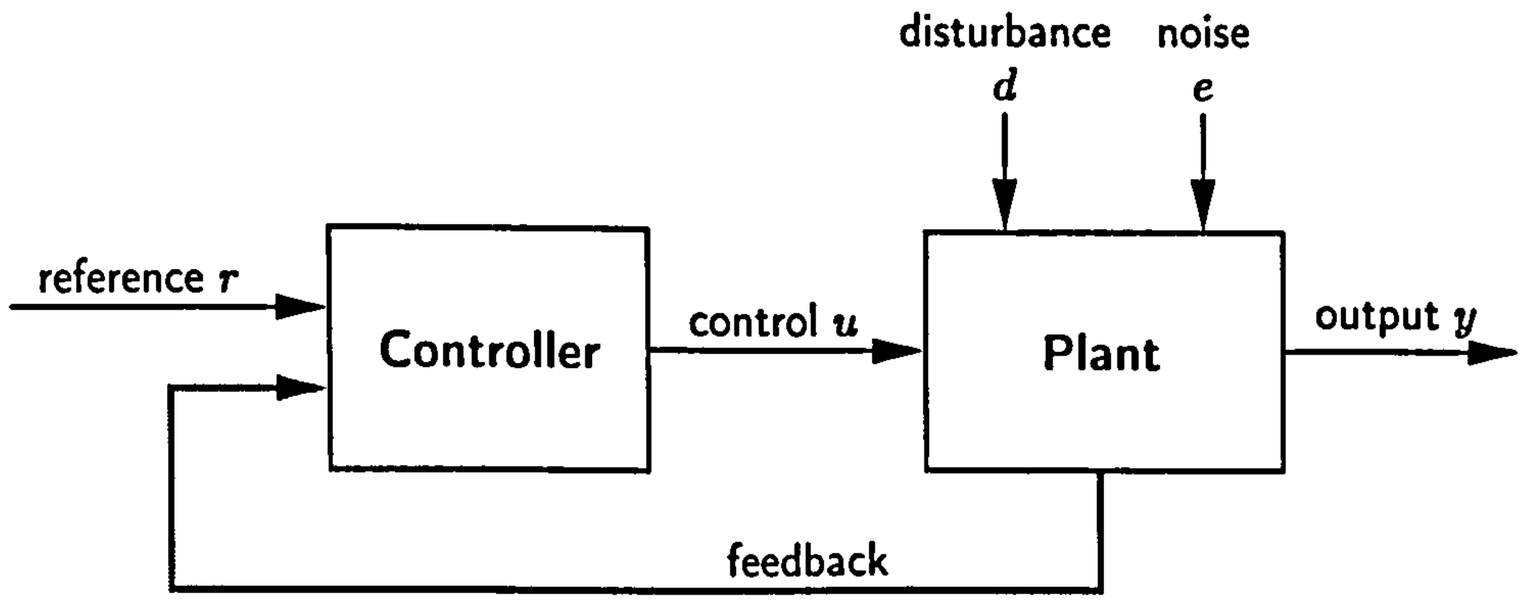



Figure 3.1: General Control Loop.

The disturbances can be divided into a low frequency component $d$ which includes modelling uncertainties and measurement offsets, and a high frequency component $e$ which includes measurement noise and high frequency measurement errors. The controller is usually designed in such a way that it counteracts the low frequency disturbance $d$ by an appropriate control signal, and rejects the high frequency noise $e$.

When the disturbance $d$ is zero, or its value is known, then the control problem can

be solved by designing a controller which simply inverts the plant's dynamics[1], resulting in a transfer function of unity from the reference to the output. The feedback loop is not necessary in this case, and no sensors are needed to measure the output of the plant. This setup is referred to as *open loop control.* It is a technique particularly popular in robotics where the actuator torques required for a specific motion are calculated using an inverse of the system. Its main limitation is that exact knowledge of the plant and the disturbances is required.

When the plant is not completely known (e.g. there are modelling uncertainties) or if the disturbance $d$ can not be quantified *a priori*, the feedback path shown in Figure 3.1 can be employed to provide information about the actual state of the plant. This configuration is referred to as *closed loop control.* The feedback signal can include the plant output, or, if the states are available, the state vector of the plant. The design of a closed loop controller requires analysis of the stability and robustness of the control loop.

### 3.1.2 Local Control Strategies

The design of a controller is usually based on a model of the process. A large number of powerful design and analysis techniques exists for linear controllers whose design is based on a linear model of the process, see for example (Franklin *et al.* 1994, Åström and Wittenmark 1990, Middleton and Goodwin 1990, Kailath 1980). These include PID controller, optimal control, pole placement, to name but a few.

If a model of the process is available in form of an LMN with linear local models as introduced in Chapter 2, a straightforward approach is to use a linear design technique to obtain a local linear controller for each local model. The local controllers can then be interpolated using the same scheduler as the Local Model Network. We will refer to such a structure, which is shown in Figure 3.2, as a Local Controller Network (LCN). The main assumption here is that, once the system has been decomposed into locally valid models, a similar decomposition can be used to design a corresponding controller.

Local Controller Networks have been used successfully in a number of applications (Gollee and Hunt 1997, Johansen *et al.* 1998). The approach is closely related to fuzzy control. It can be interpreted as a constructive way of designing gain–scheduling controllers which will be discussed in more detail in Section 3.2.

Instead of designing a network of controllers off-line, controllers can be designed on-line: the interpolated Local Model Network can be used to design a corresponding time-varying controller at each sampling instant. While this approach is closely related to adaptive control (Åström and Wittenmark 1989), it is computationally expensive as a new controller needs to

[1]This technique can only be applied for plants with minimum phase characteristics for which the plant's zeros (which will be the poles of the controller) are all stable (i.e., located in the left half of the complex plane if the system is continuous).

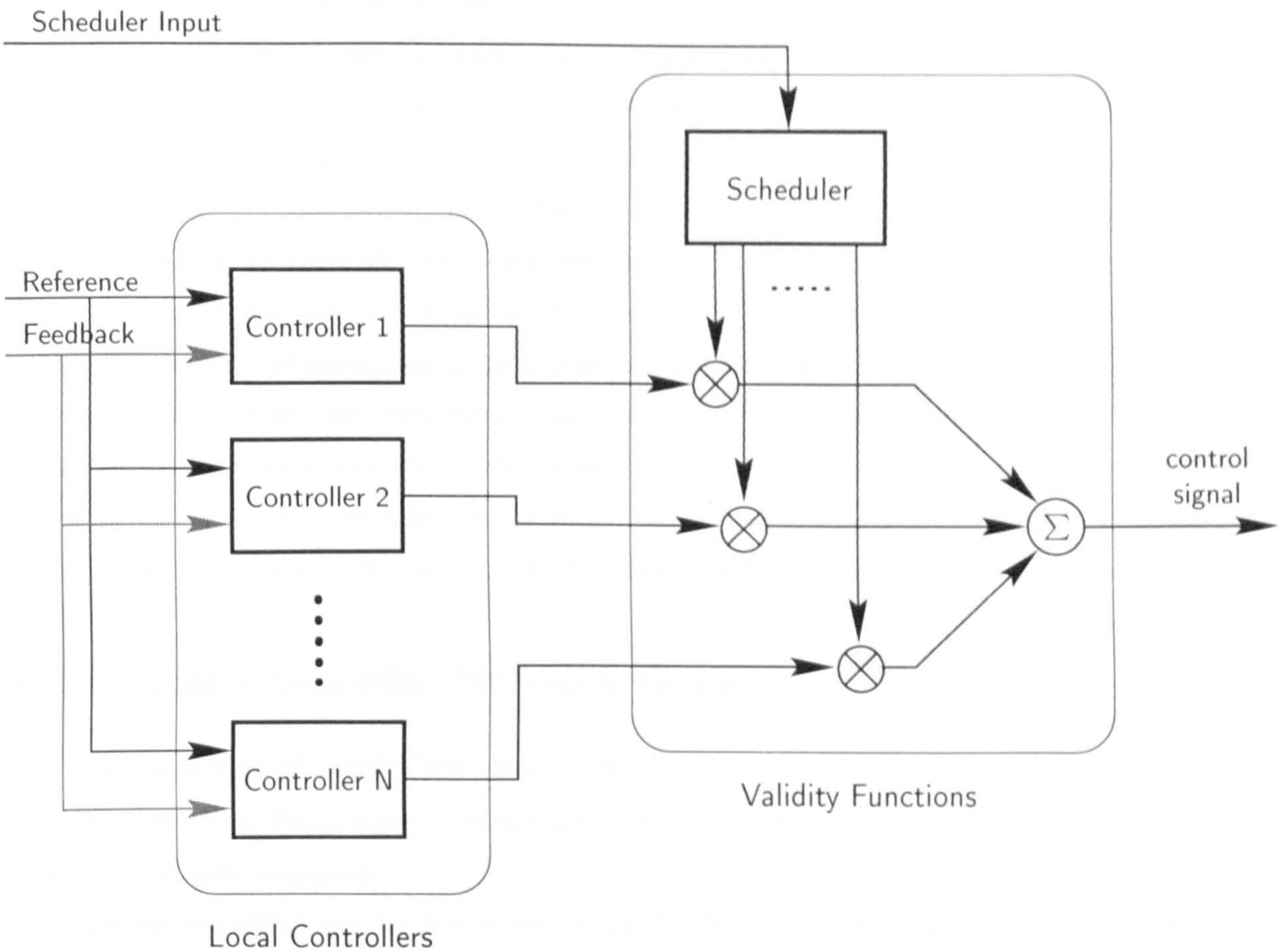



Figure 3.2: The multiple controller approach (Local Controller Network).

be designed at each sampling instant. An alternative approach is suggested in (Ronco 1997), where a new controller is only designed if the performance requirements can not be met by the current controller. Such a design approach is, however, potentially sensitive to noise and disturbances.

## 3.2 Gain Scheduling Approach to Control

The basic idea of the gain scheduling approach is to design different (linear) controllers for different operating conditions of the plant. When in operation, the parameters of these controllers are interpolated by a scheduler, resulting in a time-varying control scheme. The main advantage of this technique is that relatively simple standard design techniques (such as PID control) can be used to compose a non-linear controller.

Although gain scheduling has been proven to work successfully in many real world applications, it is difficult to obtain analytical results for stability and robustness of this control approach. The main problem is that the influence of the scheduler has to be taken into account. It can only be neglected if the scheduling variable varies slowly in which case stability

can easily be proven (Shamma 1988). For a review of the limitations and potential dangers of using gain scheduling see (Shamma and Athans 1992) and (Shorten 1996).

Traditional gain scheduling is limited to design based on local models which are linearisations of the plant's dynamics around equilibria. While this yields good results when the plant is operating close to its equilibrium manifold, performance can be unsatisfactory for transients which go through operating regions far away from the equilibrium manifold. In (Hunt and Johansen 1997) it is shown how a controller design based on local models which are not limited to linearisations around plant equilibria, but which cover off-equilibrium transient operating conditions, can significantly improve the performance of the gain scheduling controller. As the local models in LMNs are not restricted to equilibrium points of the plant this structure is a good candidate upon which to base such a controller design. An application of this approach to a vehicle speed control system has been reported in (Johansen *et al.* 1998).

## 3.3 Local Controller Network Design

The basic concepts of Local Controller Networks was introduced in Section 3.2. We want to illustrate here how linear controller design techniques which are described in Appendix C can be used with this approach.

Suppose the plant can be described by an LMN in transfer function form, as introduced in Section 2.2.2. For each local model, $\{A_i, B_i, T_{d,i}, d_i^y\}_{i=1}^M$, a corresponding local controller $\{S_i, G_i, H_i, C_i^{\text{ff}}\}_{i=1}^M$ can be designed using the pole-placement technique described in Section C.1. The offset term can be eliminated by either forward compensation or by including integral action. Owing to its robustness to low-frequency modelling uncertainties we choose to work with integral action. The time-domain closed-loop specification (rise-time $t_r$ and damping factor $\xi$) can generally be chosen differently for different local controllers.

When the plant is described by an LMN with local models in state-space notation, the state-space design method introduced in the previous section can be applied in a similar way as described above: for each local model $\{A_i, \underline{b}_i, \underline{d}_i^x; \underline{c}_i, d_i^y\}_{i=1}^M$, a corresponding local controller $\{\underline{K}_i, N_i\}_{i=1}^M$ and a local state estimator $\{\underline{L}_i\}_{i=1}^M$ can be designed. The overall controller is obtained by interpolating the local controllers using the same scheduler as for the LMN.

The superior performance of a non-linear Local Controller Network compared with a single linear controller when used to control a non-linear plant is demonstrated with the heat transfer process in Example E.4.7 on page 157. The control characteristics of the LCN control loop are consistent for varying operating conditions whereas a single linear controller meets the control specifications only for a part of the operating space.

# Part II

# Application

# 4 Functional Electrical Stimulation

## 4.1 Background

The term Functional Electrical Stimulation (FES) generally refers to the artificial electrical stimulation of muscle which has lost nervous control, with the aim of providing muscular contraction and producing a functionally useful movement (Kralj and Bajd 1989).

First experiments with artificial electrical stimulation of nerve-tissue date back to the 19th century (Hambrecht 1992). However, practical applications of this technique have only started relatively recently. Liberson's group is generally regarded as the first to apply FES in long-term clinical trials (Liberson *et al.* 1961).

The interest in FES has grown rapidly during recent years. This is partly due to progress made in hardware techniques which make small and powerful stimulators possible. New surgical techniques enable the use of chronically implanted stimulators which stimulate nerves directly within the body, e.g. (Perkins *et al.* 1996).

Functional Electrical Stimulation is widely used in the rehabilitation of paralysed patients where natural nervous control of muscular contraction has been lost due to a spinal cord injury. FES aims at restoring function to affected limbs by providing artificial electrical stimulation patterns which enable the subject, for instance, to use upper extremity functions (Peckham and Keith 1992, Allin and Inbar 1986), to stand up (Perkins *et al.* 1996, Hunt *et al.* 1998*b*), or to walk (Solomonow 1992, Graupe and Kohn 1994). A comprehensive introduction to FES for rehabilitation of standing and walking is provided in (Kralj and Bajd 1989).

A different application area of FES is to use this technique to obtain cardiac assistance from skeletal muscles (Chagas *et al.* 1989, Hooper and Stephenson 1991, Pochettino *et al.* 1991, Salmons and Jarvis 1992), cf. Chapter 1. The aim is to provide pumping support for a failing heart by a Skeletal Muscle Ventricle (SMV). A possible setup uses a muscle from the patient's back (*latissimus dorsi*) and reconfigures it as an SMV which provides additional hydraulic power to the circulation of blood. The transformed skeletal muscle is cut off from its natural nervous stimulation, and is now stimulated by FES such that i) the SMV supports the weak heart in such a way that only the power actually needed is generated, and ii) it contracts with an optimised phase relationship to the natural heart-beat.

For both areas of FES applications a model of the muscle is essential to develop algorithms

for its controlled stimulation. We are therefore interested in obtaining models of electrically activated muscle which can be used in stimulator controllers for FES.

A general overview of muscle modelling with emphasis on biomechanical applications is given in (Winters and Woo 1990). In (Stein *et al.* 1992), aspects of muscle modelling and control are discussed and examples of FES implementations are given. A collection of more recent FES related research results, including numerous applications, can be found in (Pedotti *et al.* 1996).

## 4.2 Muscle Stimulation

The activation of muscle by artificial electrical stimulation differs from the activation by the central nervous system in a number of ways. To understand these differences, it is essential to take some basic physiological properties of the neuro-muscular system into account. Thus, we introduce some physiological properties of muscle which are relevant for FES. A more detailed introduction can be found in standard physiology textbooks, e.g. in (Gray 1995). Aspect which are specific to the physiology of the neuro-muscular system are covered in (Keynes and Aidley 1991) and (McMahon 1984).

### 4.2.1 Muscle Physiology

In the neural system, transmission of information takes place in the form of impulse trains. A single stimulation pulse can create an action potential in the neuron if it exceeds a certain threshold. To avoid over-excitation, another action potential is only possible after a certain recovery period. The neural information is encoded in the inter-pulse interval (IPI), or the pulse frequency, of the stimulation pattern.

In skeletal muscle, extrafusal muscle fibres are the primary units of contraction. They are activated by axons of $\alpha$-motoneurons, which originate in the spinal cord.

One motoneuron activates 5 to 1000 muscle fibres simultaneously. All the fibres activated by the same motoneuron can be distributed over the entire muscle and form a motor unit, which represents the "unit" of muscle force in a normally innervated muscle. The contractile force of each motor unit depends on the inter-pulse interval of the action potentials. All the motoneurons going from the spinal cord to the same muscle are contained in a nerve.

Two types of motor units can be distinguished, fast and slow units. Slow motor units are more fatigue resistant and are therefore able to generate a certain force for a longer time, whereas fast motor units can produce more power but with less endurance. The ratio of fast and slow motor units in a muscle has a great influence on its characteristics. Muscle with mainly fast motor units can contract with a higher speed and produce relatively large forces, but only for short periods of time. Such muscle is therefore specialised for movement tasks, e.g. walking. Muscle which contains mainly slow motor units has a smaller contractile

speed but is more fatigue resistant than fast muscle. It is mainly used for tasks which require the production of a constant force for a longer period of time, e.g. holding and standing. The motor unit characteristics depend on the stimulation properties of the corresponding motoneuron, in particular on the pulse frequency.

The overall force developed by the muscle depends on i) the pulse frequency of the neural stimulation, and ii) the number of activated (recruited) motoneurons.

### 4.2.2 Natural Stimulation

When the neuro-muscular system is intact, the motoneurons are stimulated by the central nervous system through the spinal cord. Each motoneuron can be stimulated selectively, enabling a graduated muscle activation: those motoneurons which innervate slowly but are fatigue-resistant are recruited first, and fast motor units which fatigue quickly are only recruited if high force is necessary. The motor units are recruited in an asynchronous fashion which ensures a smooth contraction and, by allowing all muscle fibres some rest, the whole muscle fatigues more slowly.

### 4.2.3 Artificial Stimulation

Artificial muscle activation can take place by applying electrical impulses to the motoneurons, thus generating action potentials which are transmitted to the corresponding muscle fibres. The motoneurons can be stimulated by surface or chronically implanted electrodes. Surface electrodes are cheap and easy to use but, owing to the varying resistance between the skin and the electrodes and the relatively large distance from the nerve, they do not allow for a very selective stimulation. Chronically implanted electrodes are usually located directly at the nerve and therefore ensure a more exact stimulation level. Their implementation is, however, expensive.

Even with current chronically implanted electrodes, single motoneurons are not stimulated directly. The electrodes are relatively large and stimulate many neurons. Thus, with artificial electrical stimulation the muscle activation can be varied by

a) the energy of the electric pulse (i.e. the amplitude and the pulse width) which defines the number of recruited motor units, and

b) the pulse frequency, or the inter-pulse interval (IPI), which determines the contraction of the recruited muscle fibres.

If the energy delivered by the electric impulses is large enough to recruit all the motoneurons of this muscle, the stimulation is said to be *supramaximal*, and *submaximal* otherwise.

When submaximal stimulation is used, the fast motor units are recruited first as they have larger motoneurons than the slow motor units. This is contrary to the way the recruitment

takes place when the muscle is stimulated by the central nervous system. Moreover, all the recruited motor units are stimulated synchronously, as opposed to the natural stimulation which can take place asynchronously. Thus, it is more difficult to ensure smooth contraction with artificial stimulation. One way to obtain a smooth tetanus even with synchronous stimulation is to increase the stimulation frequency which leads, however, to faster muscle fatigue. Additionally, as every submaximal stimulation pulse will recruit the same motor units, these units will fatigue quickly, and the overall muscle fatigues more quickly than with natural stimulation.

### Non-linear Stimulation Characteristics

As the recruitment of the motoneurons depends on the energy of the electric stimulation pulses in a non-linear way, recruitment non-linearities can be observed. Often these non-linearities are approximated by a static curve. However, recruitment varies with the stimulation characteristics (Durfee 1992), and differences in recruitment and de-recruitment require a dynamic description of these effect (Dorgan and O'Malley 1997).

It is well-known that the muscle characteristics vary significantly with the stimulation frequency (Cooper and Eccles 1930). The dependency of the force generated by a muscle on the frequency of the stimulation pulse train can be described in a simple way by a static force-frequency curve.

Both the static recruitment curve and the static force–frequency characteristic have a similar shape which is shown in Figure 4.1. For low pulse energies or small stimulation frequencies a threshold in the activation can be observed. Saturation takes places for large energies or high stimulation frequencies as all motoneurons are recruited or they fire at their maximal rate, respectively.

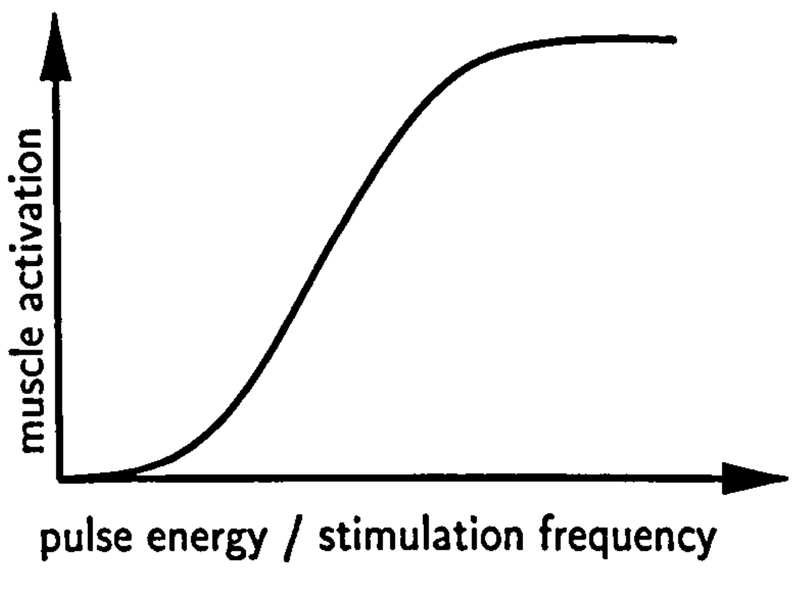



Figure 4.1: Non-linear muscle activation characteristics.

It has been observed that the force–frequency relationship is generally not static; the force developed by the muscle depends on the history of the stimulation frequency (Binder-Macleod and Barrish 1992, Duchateau and Hainaut 1986). The phrase "catch-like" effect is often used

to refer to a non-linear summation of contraction for stimulation pulses with a very short IPI at the onset of a contraction[1]. This effect is described by e.g. (Binder-Macleod and Barrish 1992), and analysed in (Stein and Parmiggiani 1981, Parmiggiani and Stein 1981). As the "catch-like" effect is normally initiated by a doublet or triplet of pulses with short IPI, it is sometimes referred to as the "doublet" or "triplet" effect.

Fatigue causes medium term changes of the muscle characteristics. As fatigue properties depend on the history of the muscle stimulation, which is normally not sufficiently known, it is difficult to describe these effects in a consistent way.

### Long Term Effects

Owing to the differences between artificial and natural activation of muscle, the characteristics of the muscle tissue can change with time when stimulated artificially. In particular, fast motor units can be transformed into slow ones and vice versa.

Muscles stimulated chronically with constant frequency pulse trains undergo significant changes in their characteristics (Kwende *et al.* 1995): Chronic stimulation of rabbit skeletal muscle at 10 Hz for several weeks renders it fatigue resistant, a property that is crucial to clinical applications that call for sustained levels of activity. However, muscles stimulated in this way also undergo a reduction in bulk and contractile speed, and this results in a substantial loss of power (Jarvis *et al.* 1991, Jarvis 1993). (Jarvis *et al.* 1996*b*) have shown that stimulation at 2.5 Hz produces muscles that are significantly faster and more powerful than those stimulated at 10 Hz, yet just as resistant to fatigue.

For this reason, and to minimise muscle fatigue, it is desirable to enable the same mechanical response to be elicited by fewer impulses. This would i) deliver a lower aggregate number of impulses over time and therefore hold out the prospect of preserving muscle power more effectively in the long term, and ii) minimise the number of repetitive stimulations of the same motor units and thus reduce fatigue (Karu *et al.* 1995). We refer to such stimulation pattern as being "optimal". The generation of optimal stimulation patterns to obtain a desired muscle response is an area of active research (Maxwell *et al.* 1996, Binder-Macleod and Barker 1991). We suggest techniques to obtain such patterns in Chapter 6, see also (Jarvis *et al.* 1996*a*).

## 4.3 Muscle Model Structures

Different types of muscle models are used for different purposes. The range extends from analytical models which are based on physical properties of the muscle, either at a microscopic or at a macroscopic level, to empirical models which are purely mathematical descriptions of the input—output characteristics of the muscle. An extensive review of various modelling

[1]Note that the term "catch-like" effect is sometimes used in the literature to describe a more general class of non-linear force–frequency characteristics. Throughout the thesis we will, however, use this term in the sense of the definition given here.

approaches can be found in (Zahalak 1992). Aspects which are particularly relevant for modelling of artificially stimulated muscle are discussed in (Durfee 1992). (Winters and Stark 1987) compare model structures based on microscopic analysis, macroscopic analysis and purely mathematical models.

### 4.3.1 Analytical Models

#### Microscopic Models

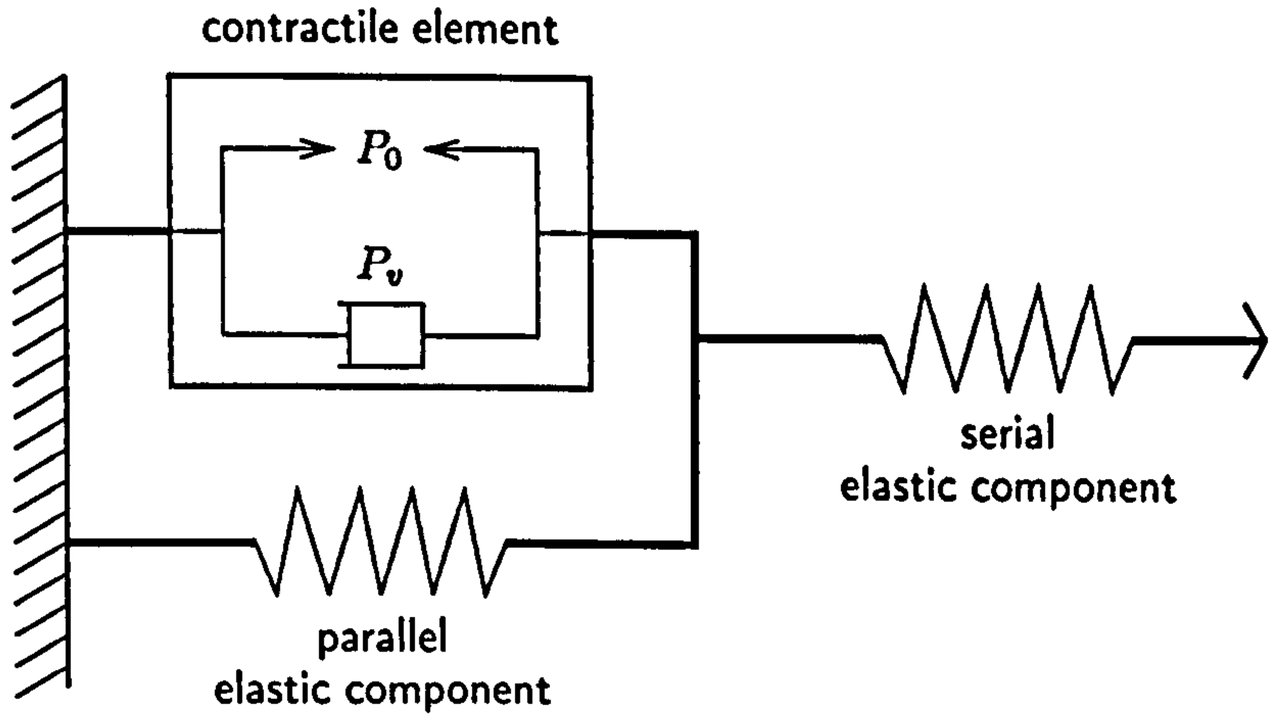
The most widely used biophysical model is the cross bridge model, the basic principles of which were developed by Huxley (Huxley 1957). It aims to describe the muscle characteristics at a microscopical level by modelling the processes within a single muscle fibre. This type of model is *in principle* useful to describe all characteristics of muscle, as all model parameters are based on physical components, which makes it very popular amongst biologists. However, the microscopic approach makes a description of entire muscle very difficult as parameters at the level of muscle fibres have to be identified. It also leads quickly to large systems of non-linear partial differential equations which are difficult to handle. Parameters of a Huxley-type model are difficult to interpret in terms of the macroscopic muscle characteristics.

A number of unconventional cross-bridge models have been suggested which make different assumptions than those of Huxley, e.g. (Hatze 1990).

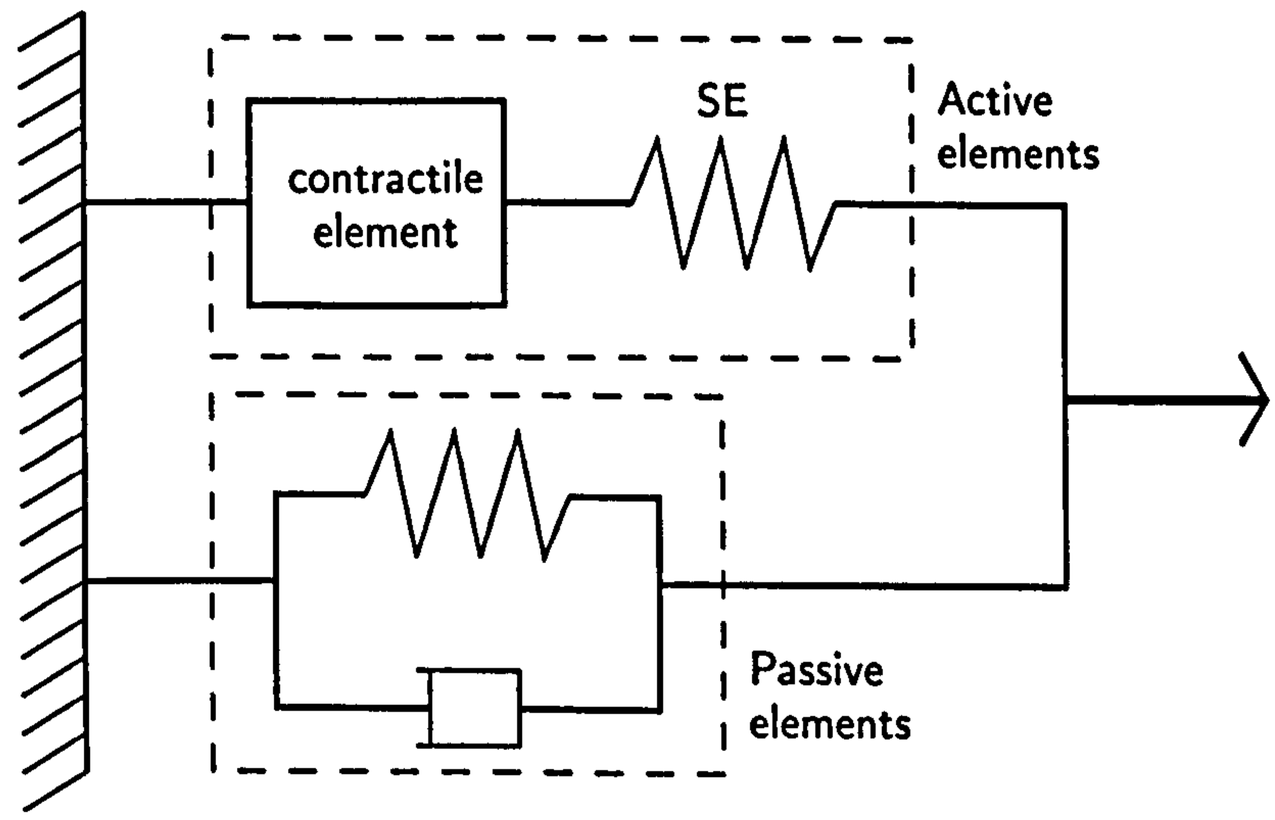
#### Macroscopic Models

In biomechanics, analogue models which are based on macroscopic muscle characteristics remain more popular, as they are more tractable and easier to interpret when used to describe an entire muscle. The most often cited model is based on the description by Hill (Hill 1938), a schematic diagram of which is shown in Figure 4.2(a) (Zahalak 1992). It comprises a contractile element which is the force generating element, in parallel with a spring representing the elasticity of the muscle (and the tendon, if included), all of which is in series with a second elastic component representing the passive tissue. The elements in this model are standard mechanical components (although they might be non-linear) and thus most engineers are familiar with them. The parameters are easy to interpret in terms of the macroscopic muscle characteristics. A large amount of experimental data and results are available for the Hill-type model from almost 60 years of experience. However, for the identification of the parameters of the model components special experiments are necessary which may not be applicable in all situations. The elements of the model are not related to underlying physiological processes; the components are artificial constructs.

Variants of the original model proposed by Hill are used in the literature, one of which is the muscle-tendon model shown in Figure 4.2(b) (Durfee and Palmer 1994, Winters and Stark 1987). The model is modified in such a way that the active and the passive components




(a) Structure according to (Zahalak 1992).




(b) Modified structure.

Figure 4.2: Hill-type muscle model structures.

of the muscle are separated. The active elements contain a contractile element in series with a series elastic element (SE). The passive elements comprise a parallel spring-damper combination.

In the original work of Hill, the contractile element did not include muscle activation dynamics; the muscle was thought to be stimulated at a constant level. Since then, much work has focused on including activation dynamics in the contractile element. Often empirical model structures as described in Section 4.3.2 are used here, e.g. (Durfee and Palmer 1994, Shue *et al.* 1995).

### Hybrid Models

Another group of model structures aims at combining the interpretability and accessibility of a model which describes the entire muscle with the physiological insight which is gained when processes inside a single muscle fibre are taken into account. Here, each muscle fibre is modelled individually based on physiological principles. These single models are then combined by a recruitment model which describes effects which can be attributed to the artificial electrical stimulation of the muscle.

The best known model structure of this type is described in (Hatze 1977, Hatze 1978). Here, each muscle fibre is described as a Hill type model whose elements are interpreted as being linked to physiological processes in the muscle. Additional elements of the model account for the effect of motor unit recruitment. Hatze claims that his model is capable of predicting correctly practically all known phenomena in terms of the muscular force-output. However, the model is complicated and requires solution of non-linear differential equations.

A number of other muscle model structures are based on the model introduced by Hatze. (Riener *et al.* 1996) propose a model where the description of each motor unit is based on Hatze's model. The motor units are divided into a group of fast and a group of slow motor units. Separate static recruitment models for each group account for effects of varying stimulation activity. This model also includes elements to describe fatigue and recovery.

A related modelling approach is described in (Dorgan and O'Malley 1996, Dorgan and O'Malley 1997). As in Riener's model, the description of each muscle fibre is based on (Hatze 1978). However, Dorgan uses a dynamic recruitment model which accounts for different recruitment characteristics when more units are recruited, the recruitment decreases, or the recruitment remains constant.

## 4.3.2 Empirical Models

For use in implantable muscle stimulator devices different model characteristics from the ones inherent to the group of analytical models become important, namely that the model

- should be easy to adapt to the muscle using data from standard experiments which do not damage the muscle tissue;
- should be controller orientated, i.e., controller design based on such a model should be possible. The designed controller must not be computationally expensive as it should be implemented in the form of an implantable device.

Whereas the first point can be partly overcome by modern analytical model structures, analytical models cannot usually be represented in the form necessary for controller design. Thus, empirical model strategies which aim to describe the input—output characteristics of muscle (often limited to conditions common in FES applications), and whose structure is suitable

for the design of stimulation controllers become useful. A review of empirical modelling approaches for muscle is given in (Durfee 1992).

In order to describe the characteristics of a muscle when working against a load, a two–input one–output system is usually considered: the muscle stimulation and the length (or the velocity) are the input variables, and the force is the output. In such a general setup, the force–length relationship depends on the characteristics of the load.

For experimental analysis, the load is often idealised in the sense that the muscle length is kept constant. The contraction is then referred to as *isometric*. As force becomes the only output variable in this setup the structure of the system is simplified to a single–input single–output system. Note that no mechanical power is generated by the muscle during isometric contraction.

As the modelling experiments described in Chapter 5 of this thesis focus on muscle contraction under isometric conditions we first review model structures which describe muscle under such conditions. At the end of this section, extensions for non-isometric contraction are briefly introduced.

### Modelling of Isometric Contraction

When the muscle contraction is isometric, the model structure simplifies to a single–input single–output system where stimulation is the input and force the system output.

Experiments have shown that under constant stimulation conditions (e.g. constant pulse energy and frequency), isometric contraction can be described by a linear second order dynamic model (Mannard and Stein 1973, Zahalak 1992) [2]. However, a purely linear description fails when the muscle stimulation varies significantly (Winters and Stark 1987).

One way to extend the linear modelling approach to stimulations with varying pulse energy is to add a model component which describes the non-linear recruitment of motor units. A widely used structure applies a static non-linear recruitment curve( cf. Figure 4.1), which leads to a Hammerstein model as shown in Figure 4.3. This approach has been used for example in the models of (Bernotas *et al.* 1986, Shue *et al.* 1995, Durfee and Palmer 1994). Techniques to identify the recruitment curve have been described in the literature (Durfee and MacLean 1989). The main advantage of this simple model structure is that the plant can be easily transformed into a linear form for controller design by shaping the control signal by the inverse static recruitment curve (Durfee 1992, Hunt *et al.* 1997).

As mentioned in Section 4.2.3, a simple static recruitment curve cannot take account of the inherently dynamic process of motor unit recruitment. It has also been observed that the assumption that the dynamics of the muscle can be described by a linear time-invariant transfer function does not hold if the level of activation varies over a wide range (Hunt *et*

[2] Note that a second order linear description was found appropriate even under non-isometric conditions (Bawa *et al.* 1976*a*, Bawa *et al.* 1976*b*).

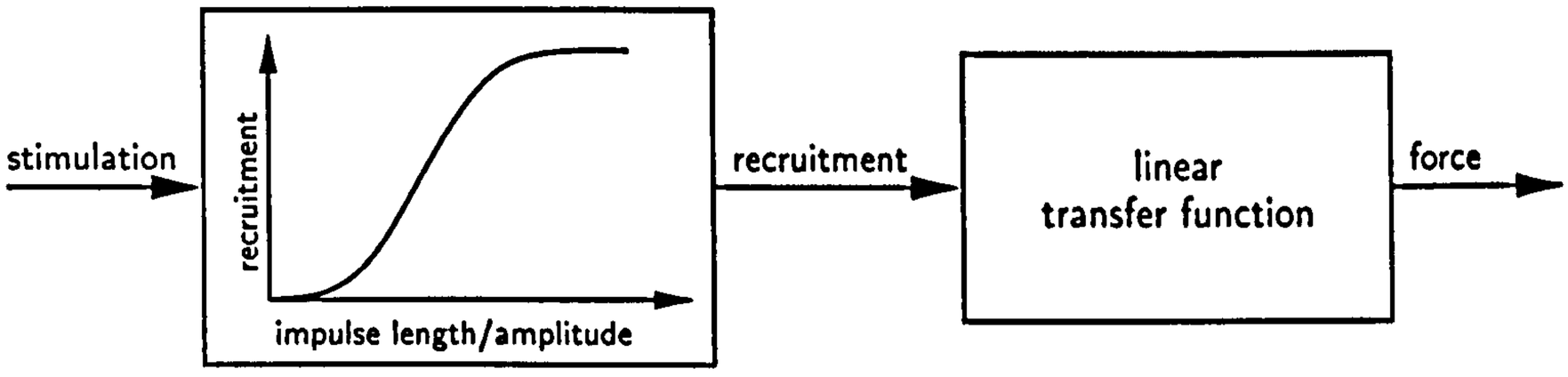



Figure 4.3: Hammerstein cascade structure.

*al.* 1998*a*).

The simple Hammerstein structure does not account for changes of the muscle characteristics with varying stimulation frequency, cf. Section 4.2.3. Controller-orientated models which describe non-linear properties such as the force–frequency curve and the "catch-like" effect will be very useful for FES, as it is important to employ such non-linear characteristics to reduce muscle fatigue (Karu *et al.* 1995).

Thus, more general non-linear model structures are needed to describe isometric muscle contraction under varying stimulation conditions. In (Donaldson *et al.* 1995), a radial basis function network is used to model isometric contraction of muscle which is stimulated with supramaximal pulse trains of varying frequency. (Schultheiss *et al.* 1997) use a neural network structure to incorporate non-linear recruitment dynamics in the model. (Bobet *et al.* 1993) show that a linear time-varying model can successfully describe muscle contraction under conditions where pulse energy and/or stimulation frequency vary. In their model structure, the muscle force is approximated by a critically damped, linear second order system which is time-invariant between stimulation pulses. The model parameters are adapted separately for each inter-pulse interval (IPI). This results in an overall model with as many linear models as IPIs. The fact that this approach can approximate isometric muscle contraction under various stimulation conditions shows that a 2nd order time-variant linear model is an appropriate model structure. However, a general way of relating stimulation to the parameter changes of the model could not be found (Bobet *et al.* 1993).

### Modelling of Non-isometric Contraction

When muscle is allowed to change its length the contraction is said to be non-isometric. In such a setup, the force developed by the muscle does not only depend on the muscle activation, but also on its length (the force—length relationship) and on the velocity of the contraction (the force—velocity relationship). A common approach for the modelling of non-isometric contraction is to work with empirical model structures which are based on a Hill-type model, cf. Figure 4.2. In (Durfee and Palmer 1994) and (in a slightly different form) in (Shue *et al.* 1995) a modified Hill-type model is used which is schematically shown in Figure 4.4.

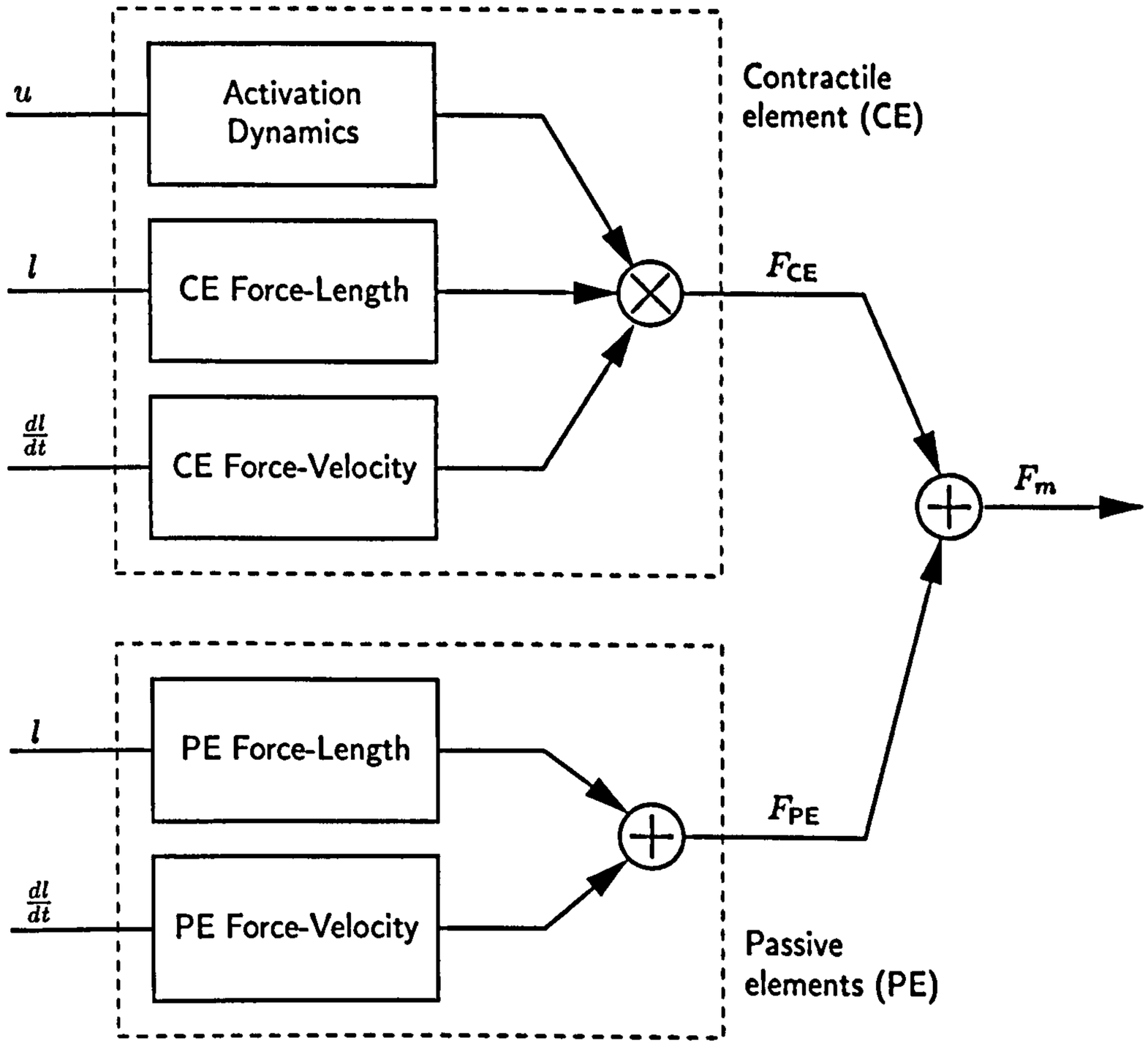



Figure 4.4: Model of non-isometric muscle contraction based on a Hill-type structure. The electrical stimulation of the muscle is denoted by $u$, $l$ is the muscle length, $dl/dt$ the change of the length, and $F_m$ denotes the muscle output force.

In this model the force–length and the force–velocity characteristics in the contractile and in the passive elements are approximated by non-linear static curves. A model of isometric contraction is used to describe the activation dynamics in the contractile element. In (Durfee and Palmer 1994, Shue *et al.* 1995), Hammerstein structures are employed here. This limits the validity of the model to muscle stimulation with varying pulse energy. To take advantage of variation of the stimulation frequency a more complex non-linear model structure of isometric contraction is needed to describe the activation dynamics under such general conditions.

### 4.3.3 Discussion

In the review of modelling approaches for muscle given above, two main strategies can be distinguished, analytical (or physiological) and empirical (or "black-box") modelling. In this section we will summarise the properties of both approaches and discuss their relevance to muscle modelling.

The analytical approach to modelling of muscle is based on first principles. Depending

on the intended application, muscle characteristics are analysed either on a microscopic or on a macroscopic level. The structure and parameters of analytical models can therefore be related to known properties of the muscle. Thus the behaviour of the model is known with respect to the muscle characteristics which have been considered for the model development. This can be regarded as the main advantage of analytical models. Analytical models can be useful to study the characteristics of muscle in simulation experiments, avoiding the need for expensive experiments.

Analytical models are, however, expensive to develop as they require a detailed analysis of the muscle. Only characteristics which are known and for which an analytical description exist can be incorporated. The models are often relatively complex which complicates their use in real-time with implantable devices where the available resources are limited. They are usually not controller oriented in the sense that a direct controller design which is based on such model is not possible.

The empirical approach to modelling is not based on an analysis of the underlying system. The parameters of the model are generally not related to characteristics of the muscle. All information used to build the model is extracted from experimental data (usually input–output data) obtained from the muscle. Thus the design of the experiments to collect these data is of great importance to ensure that all relevant characteristics of the muscle are represented in the data. Once the data is available, the identification of the model structure and its parameters is relatively straightforward and inexpensive compared to analytical models. The structure of empirical models is often simpler than that of a corresponding analytical model. Empirical models can therefore usually be simulated more easily and are often suitable for real-time applications. Their structure can be controller orientated which facilitates the design and analysis of control strategies.

For empirical models it can generally not be ensured that their behaviour for operating conditions which are not covered by the experimental data corresponds to the behaviour expected from muscle. The analysis of the model properties is more complicated as the parameters are not directly related to system characteristics. This makes the validation of empirical models more complicated.

The Local Model Network approach introduced in Chapter 2 represents a model structure which is aimed to overcome some of the problems associated with pure empirical modelling. In particular, it is possible to interpret their parameters to some degree with respect to the characteristics of the underlying system. This makes this approach more attractive for real applications than pure "black-box" approaches.

# 5 Modelling of Muscle Contraction

In this chapter we will introduce an empirical approach to the modelling of electrically stimulated skeletal muscle. The structure of the model is non-linear and based on the Local Model Network introduced in Chapter 2. Thus, it is a controller orientated model, as the design of a Local Controller Network can be directly based on this structure, cf. Chapter 3.

The chapter is structured as follows. After a brief description of the experimental setup used to collect the data, the data obtained for isometric contraction of rabbit muscle are introduced and analysed. Analysis of the data provides guidelines for the choice of the initial model structure such as the sampling period, the dynamic order and the time delay.

To assess non-linearities of the system and to obtain further information on the selection of the optimal model structure, linear models of muscle contraction are identified. Based on the results obtained in these experiments, Local Model Network structures with linear local models are used to identify models of muscle contraction. Different LMN structures of various sizes are compared to obtain a structure which is best suited to model the data. The models are analysed thoroughly and their properties are related to the characteristics of the real muscles. An algorithm is presented which allows for the local modification of LMNs to incorporate *a priori* knowledge. Finally, the results obtained are discussed and suggestions for further work are presented.

## 5.1 Experimental Setup

The data used in this thesis are obtained from experiments with rabbit *tibialis anterior* muscles. The experiments were carried out by M. M. N. Kwende and J. C. Jarvis at the Department of Human Anatomy and Cell Biology, University of Liverpool. The muscle is stimulated by irregular supramaximal pulse trains using flap electrodes which are placed around both common peroneal nerves. The term *supramaximal* refers to the fact that the amplitude and length of the stimulation pulses are chosen such that all motoneurons of the muscle are recruited. The activation of the muscle is varied by changes of the inter-pulse intervals (IPIs) of the stimulation pulses.

The experimental protocol is described in detail in (Kwende *et al.* 1995). The following conditions apply to all experiments and are particularly relevant for our studies:

- The muscle is stimulated using electrical impulses of 200 microseconds duration and an amplitude three times the threshold for muscle stimulation, which ensures supramaximal stimulation.
- The inter-pulse intervals vary randomly between 1 and 70 milliseconds.
- The duration of each pulse train does not exceed 300 milliseconds. Together with periods of rest of 30 seconds between the pulse trains, this ensures that the influence of fatigue on the recorded data can be neglected.
- A constant-frequency burst of impulses (25 milliseconds IPI, which corresponds to a stimulation frequency of 40Hz) was delivered every 5 minutes to check that the preparation did not show progressive deterioration during the experiment.
- The data are recorded with a sampling interval of $T_s = 1$ms.

### 5.1.1 Supramaximal Muscle Stimulation

With supramaximal stimulation, the amplitude and length of the stimulation pulses are chosen such that all motoneurons of the muscle are recruited by every impulse. Thus, the muscle operates at a constant (full) level of recruitment, and effects due to recruitment variations are not present. The muscle activation is varied by changes of the inter-pulse interval, i.e., by variations of the stimulation frequency. Hence non-linear force-frequency characteristics, such as the "catch-like" effect described in Section 4.3.2, are expected to be present in the data.

The use of supramaximal stimulation is motivated by the following observations:

- When a constant distance between the implanted electrode and the nerve cannot be ensured it is desirable that the muscle activation is not affected by small changes of the location of the stimulation electrode relative to the stimulated nerve. This is relevant when the muscle subjected to FES is not located at its natural position but, for instance, is transplanted to a different place in the body. With supramaximal stimulation pulses it can be ensured that all motoneurons are recruited even when the distance between the stimulation electrode and the nerve varies in a limited range.
- For long-term, repetitive muscle activation, where it is crucial to obtain the maximal power possible, it is desirable that the stimulation technique makes use of the power of all motor units. With submaximal stimulation, only a part of the motor units is recruited, but these same units will be stimulated over and over again. This leads more rapidly to fatigue without exploiting the potential power of all motor units. With supramaximal stimulation all motor units are recruited by each pulse, and thus their potential power is maximally used.

Both these objectives are relevant for the use of skeletal muscle for cardiac assistance.

## 5.2 Data of Isometric Contraction

### 5.2.1 Data Collection

For experiments with isometric muscle contraction, the contractile force of the muscle is measured and recorded while the muscle length is held constant. The experimental setup is shown in Figure 5.1. The muscle pulls a lever which is attached to a motor. The moment generated by the motor is controlled in such a way that the lever does not move, i.e. the muscle length $l$ is held constant. Thus, when the muscle is stimulated by a pulse sequence $u$, the muscle output force $F_m$ is compensated for by the moment generated by the motor. The motor moment (and thus the muscle output force) is determined by measuring the current $I$ through the motor.

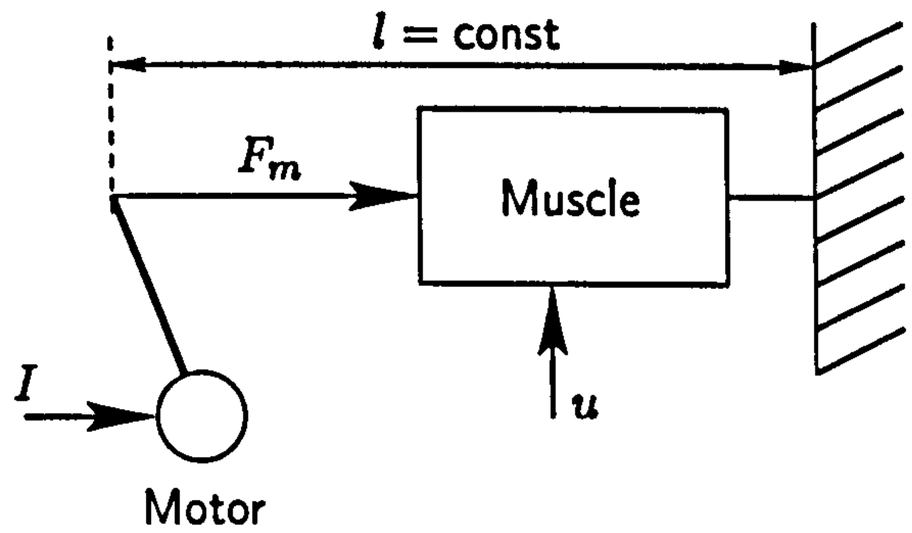



Figure 5.1: Experimental setup for the collection of isometric muscle data.

The experimental data were pre-processed such that the offset in the measured force was removed, and the input and output data sets were normalised in such a way that they lie in the range [0, 1].

Two types of muscle were used in the experiments, a control and a chronically stimulated muscle.

The *control* muscle is an unchanged rabbit tibialis anterior whose characteristics are determined by a majority of fast motor units. We will therefore refer to it as the *fast* muscle. A total of 60 data sets, containing the input pulses and the contractile force were recorded. The duration of each set is 590 milliseconds. A number of typical records are shown in Figure 5.2.

The *chronically stimulated* muscle is a rabbit tibialis anterior which was stimulated at 10Hz for 4 weeks. As outlined in (Jarvis 1993), such chronic stimulation reduces the contractile speed of the muscle. The muscle characteristics are therefore dominated by slow motor units, and we will refer to this muscle as the *slow* muscle. A total of 84 data sets, containing the input pulses and the contractile force was recorded. The duration of each set is 600 milliseconds. A number of typical records are shown in Figure 5.3.

Note that the output data of the fast and of the slow muscle were normalised individually.

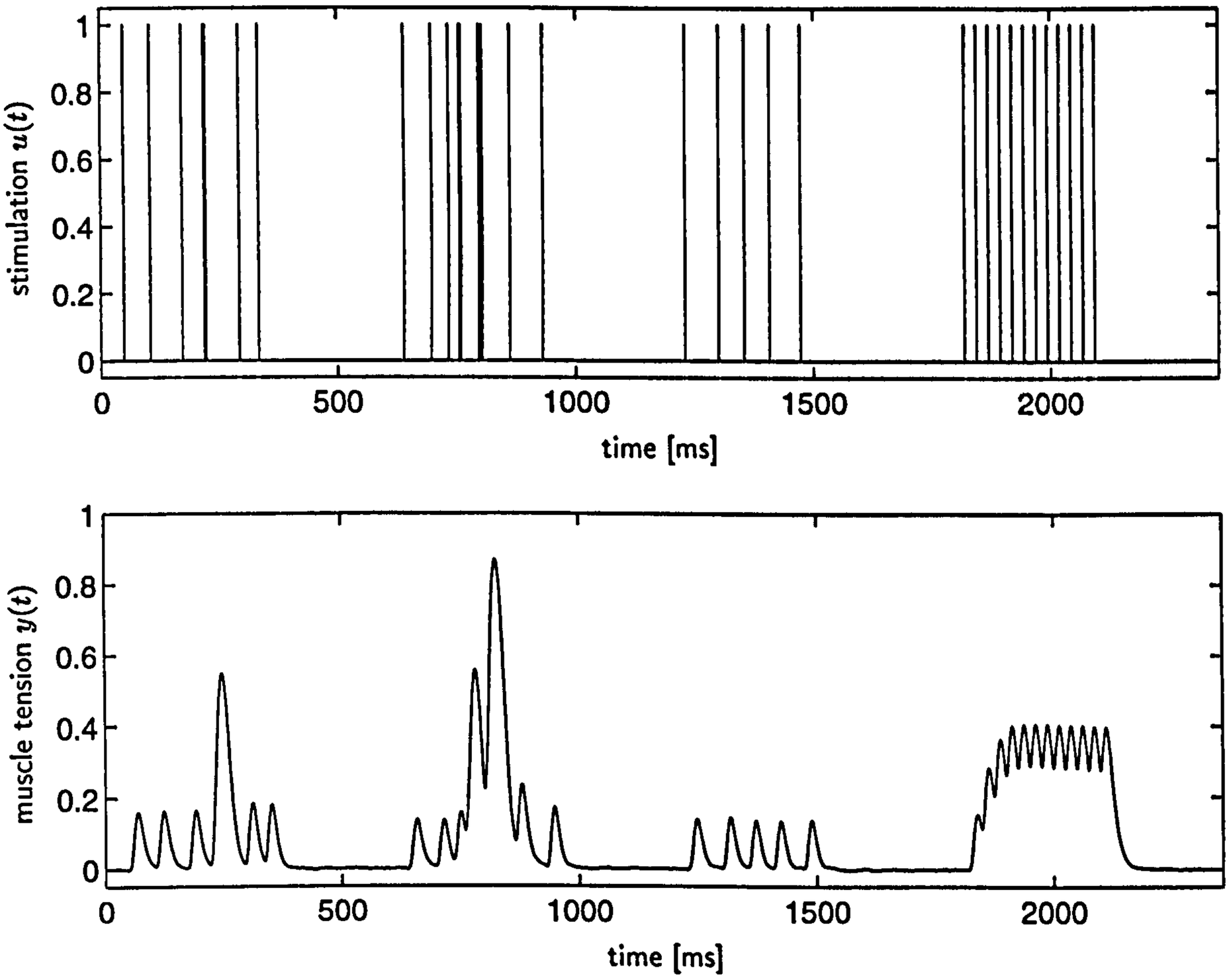



Figure 5.2: Four typical data records for the fast control muscle. The data sets are shown concatenated; the periods of rest of 30 sec between the records are omitted. The input pulses are shown in the top graph, the muscle force at the bottom. The first three data sets show records with random stimulation patterns. The last set shows a record with a constant-frequency stimulation of 40Hz.

The absolute value of the normalised muscle tensions shown in Figures 5.2 and 5.3 cannot therefore be compared with each other.

### 5.2.2 Data Analysis

A first analysis of the experimental data can give information about the frequency range of the data, time-delays and dynamic properties of the system. This information is useful for the selection of the sampling period, dynamic order and time delay of the initial model structure.

#### Sampling Period

The experimental data introduced in Section 5.2 are sampled with a period $T_{s,\mathrm{exp}} = 1\mathrm{ms}$, which is sufficiently short to represent all system dynamics in the discrete samples. As outlined

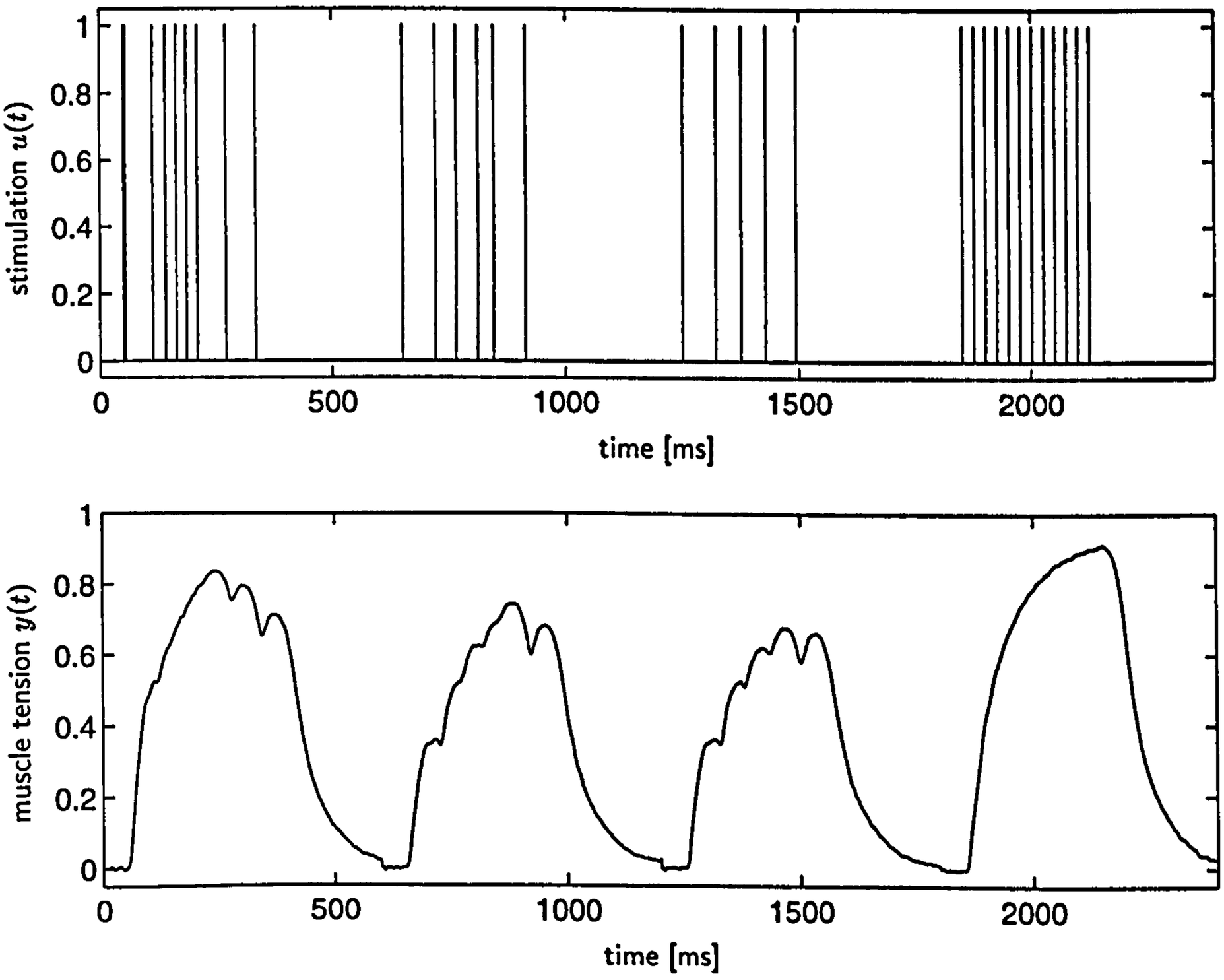



Figure 5.3: Four typical data records for the slow chronically stimulated muscle. The data sets are shown concatenated; the periods of rest of 30 sec between the records are omitted. The input pulses are shown in the top graph, the muscle force at the bottom. The first three data sets show records with random stimulation patterns. The last set shows a record with a constant-frequency stimulation of 40Hz.

in Appendix D, the representation of a system in the form of a discrete time model depends on the sampling period. In particular, for the shift operator representation (see Section D.2.1) fast sampling can lead to sensitivity problems for the identification of the parameters, cf. Section D.3. We are therefore interested in selecting an optimal sampling period which is

a) not too long, so that no information about the dynamics of the real system is lost due to a violation of the sampling theorem, and

b) not too short, to avoid sensitivity problems for the parameter estimation.

Furthermore, a larger sampling period is desirable because this goes with a reduction of the size of the data set and hence leads to a decreased computational effort.

The effects related to a changed sampling period are different for the pulse-like input signal and for the continuous-in-value output signal. In the following sections they are discussed

separately.

**Resampling of the Input Signal** For the input signal, a change of the sampling period should not affect the form of the signal, which has to remain pulse like. The sub-sampling is therefore processed in such a way that the input pulses are only shifted towards the nearest sampling point. Two objectives have to be considered here:

a) The energy of the input signal must not change, i.e. the number of pulses in the sub-sampled signal must be the same as in the original pulse train.

b) The pulses must not be shifted too far away from their original position.

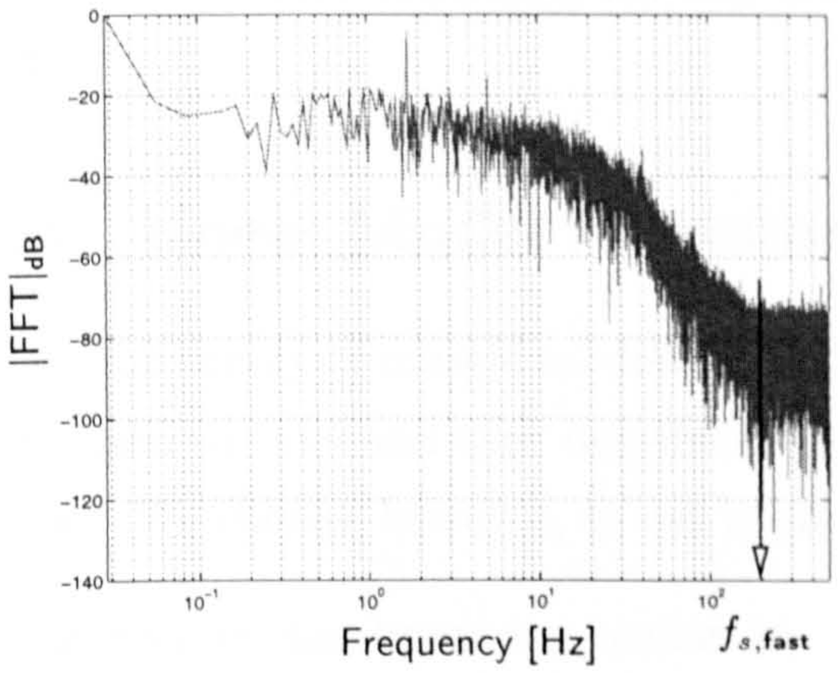
Any increase of the sampling period results in a loss of information in the input signal: by moving the pulses to the nearest sampling point they are shifted away from their original position in time. Also, the sampling frequency itself represents an upper limit to the stimulation frequency contained in the input data. If the sampling period is increased, the loss of information will increase. For the input signal we are therefore interested in a sampling period which is as close as possible to the original sampling time $T_s = 1\text{ms}$.

**Resampling of the Output Signal** The sampling period for the continuous-in-value output signal can be changed in the usual way, i.e. after low-pass filtering the signal is resampled with a larger period. Here, it is important that no significant frequency components, i.e. information, are lost.

By analysing the power spectrum of a signal, information about its frequency components can be obtained. The power spectra of the output for the fast control and the slow stimulated muscle are shown in Figure 5.4. They were calculated by applying the Fast Fourier Transformation (FFT) to all available data sets (i.e., 60 data sets for the fast muscle and 84 data sets for the slow muscle). Both graphs have a similar shape, which consists of a flat region with high magnitude at low frequencies, a descending area at medium frequencies and a flat portion with small magnitude at high frequencies. This flat region at high frequencies can be regarded as representing the noise components in the data. Thus, the "useful" information is contained in the frequency range below this region. From Figure 5.4(a) it can be surmised that the sampling frequency for the fast muscle should be chosen not lower than $f_{s,\text{fast}} = 200\,\text{Hz}$, which corresponds to a maximal sampling period of $T_{s,\text{fast}} = 5\,\text{ms}$. The sampling frequency for the slow muscle should be not lower than $f_{s,\text{slow}} = 100\,\text{Hz}$ (Figure 5.4(b)), which corresponds to an upper limit of $T_{s,\text{fast}} = 10\,\text{ms}$ for the sampling time.

### Time Delay

In order to select the structure of a model it is important to have information about the delay of the system, i.e., the time it takes until an input signal affects the system output. Such



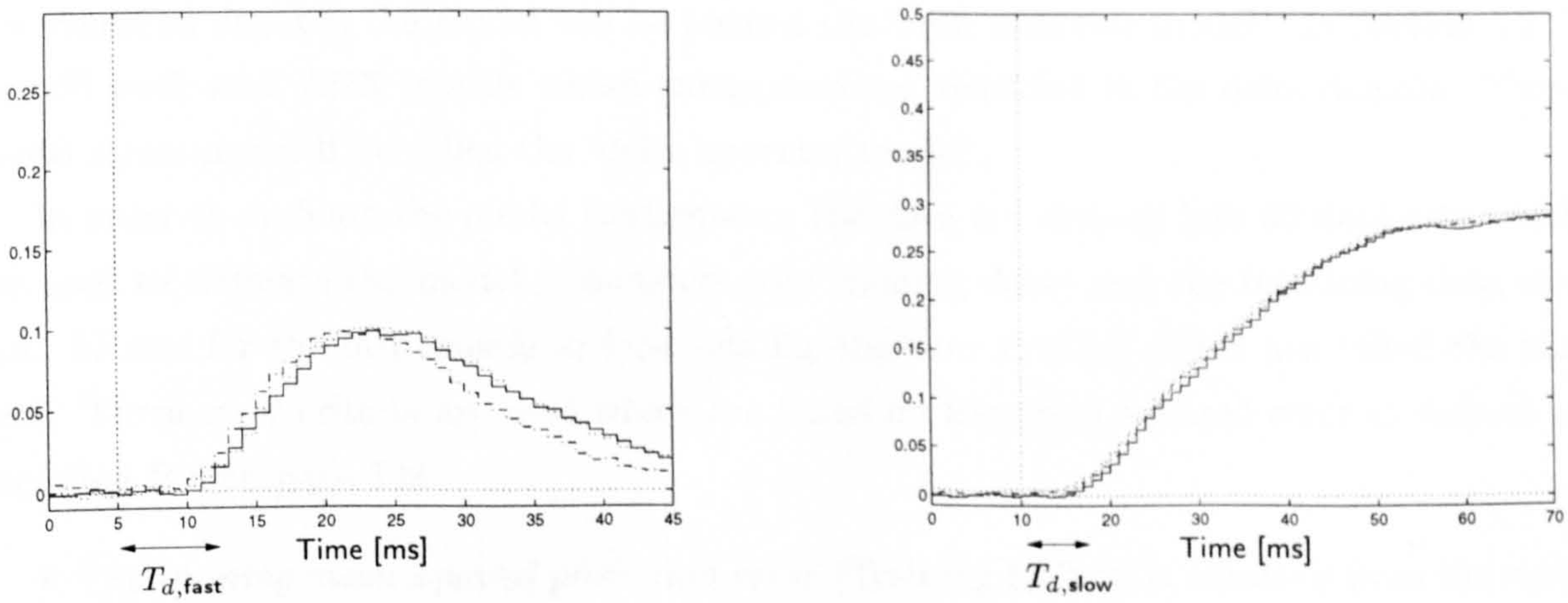

(a) Fast muscle.

(b) Slow muscle.

Figure 5.4: Power spectra of muscle output data.

information can be easily extracted from the response of the system to a single stimulation pulse.

A number of examples for the impulse responses of both the fast and the slow muscle are shown in Figure 5.5. From this, it can be surmised that the delay is approximately the same for the fast and the slow muscle and has a value of $T_d \approx 7\,\text{ms}$.




(a) Fast muscle

(b) Slow muscle

Figure 5.5: Impulse responses. For each muscle, three examples of the response to a single stimulation pulse are shown. The vertical dotted line denotes the time when the input pulse was applied. Note that for the slow chronically stimulated muscle (b) the entire response could not be shown as subsequent stimulation pulses would occur in the records.

The shapes of the pulse responses suggest that second order dynamic model structures are appropriate for modelling the response characteristics of the muscles to single stimulation pulses.

## 5.3 Linear Model for Isometric Contraction

From the survey of muscle models given in Section 4.3 it is clear that muscle is a significantly non-linear system and that non-linear modelling techniques will be needed to obtain a good model. We choose, however, to use linear model structures for initial experiments in order to

a) assess how non-linear the system is,

b) obtain guidelines for the choice of the sampling period and of the dynamic order for the model structures, and

c) provide a benchmark for the non-linear modelling experiments.

A linear input-output model structure is the ARX model introduced in Section A.2. It is defined in shift operator notation, cf. Section D.2.1. To avoid potential sensitivity problems during the estimation of the model parameters (cf. Section D.3), the parameter optimisation can be performed in a different domain, for example in the delta operator domain, cf. Section D.2.2. To simulate the model, the parameters obtained are transformed to the shift-operator notation.

Throughout this section, we will work with ARX model structures. If the parameters are identified directly, the model will be termed the "shift operator model". In Section 5.3.2, we will work with ARX models whose parameters are identified in the delta domain. These model structures will be called the "delta operator model".

In order to evaluate the model performance the data are divided into 30 data sets which are used to estimate the model parameters (the training data) and the remaining data sets (i.e., 30 sets for the fast muscle and 54 sets for the slow muscle) which are called the test data. Three error criteria are used which are based on the mean squared error as defined in Equation B.8 on page 128:

- The *training mean squared prediction error* (Training MSPE) is obtained from the error of the one-step-ahead prediction on the training data. It is only used with models trained in prediction mode.

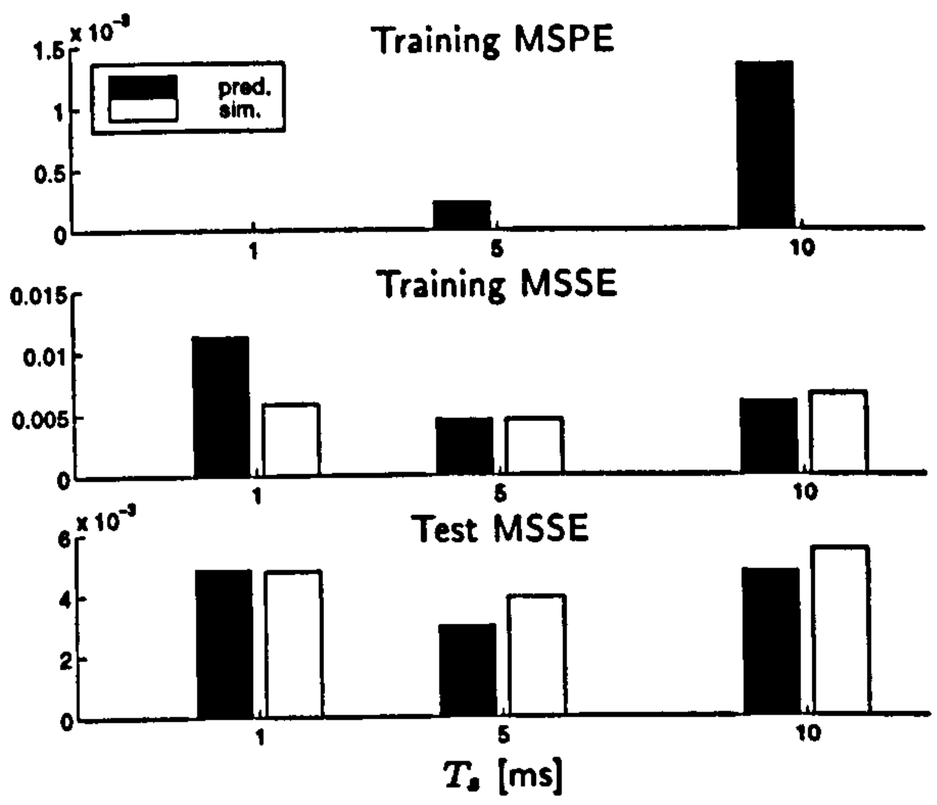
- The *training mean squared simulation error* (Training MSSE) is obtained from the error of the model simulation with infinite prediction horizon on the training data.

- The *test mean squared simulation error* (Test MSSE) is obtained from the error of the model simulation with infinite prediction horizon on the test data.

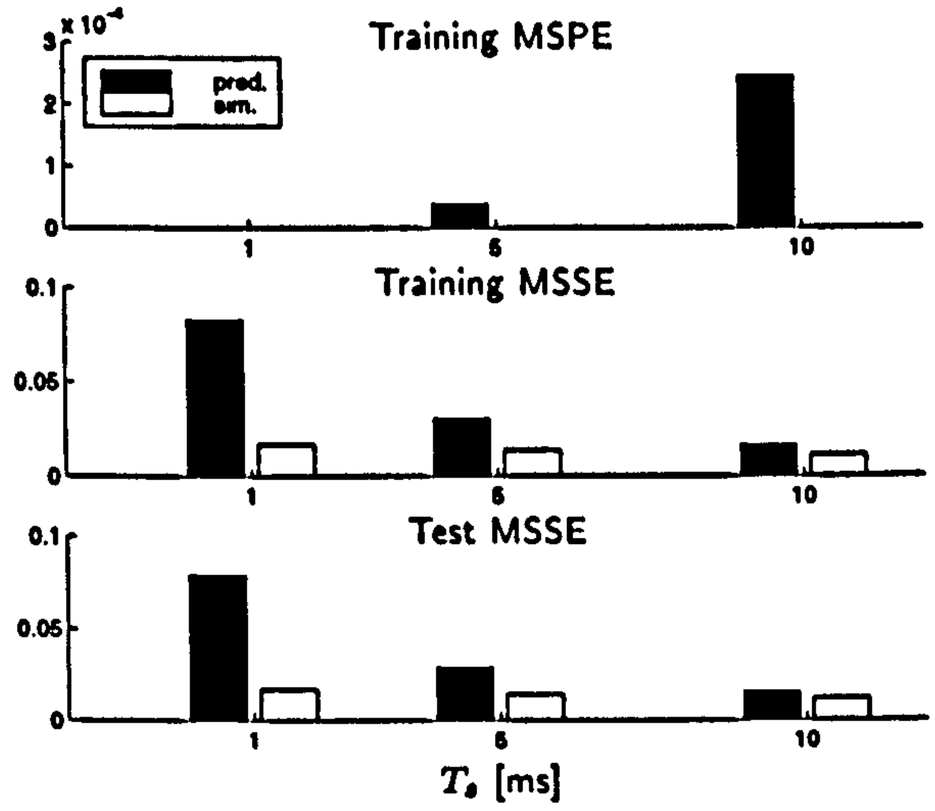
To compare the errors for different sampling periods, the outputs of the models for sub-sampled data are interpolated for the original sampling period of 1ms.

### 5.3.1 Shift Operator Model

In this section linear shift operator models are identified and evaluated. The model order is chosen to be equal to 2, as this was reported to be appropriate for various muscle modelling applications, cf. Section 4.3.2. The input delay, $T_d$, is chosen to be a multiple of the sampling period, and equal to, or larger than, $T_d = 7$ms. Models for sampling periods of $T_s = [1, 5, 10]$ms are identified. The parameters are estimated in both prediction and simulation mode. The modelling errors are shown in Figure 5.6.




(a) Fast muscle. The minimal Test MSSEs are $2.98 \times 10^{-3}$ (prediction mode) and $3.90 \times 10^{-3}$ (simulation mode) at $T_s = 5$ms.




(b) Slow muscle. The minimal Test MSSEs are $15.7 \times 10^{-3}$ (prediction mode) and $12.0 \times 10^{-3}$ (simulation mode) at $T_s = 10$ms.

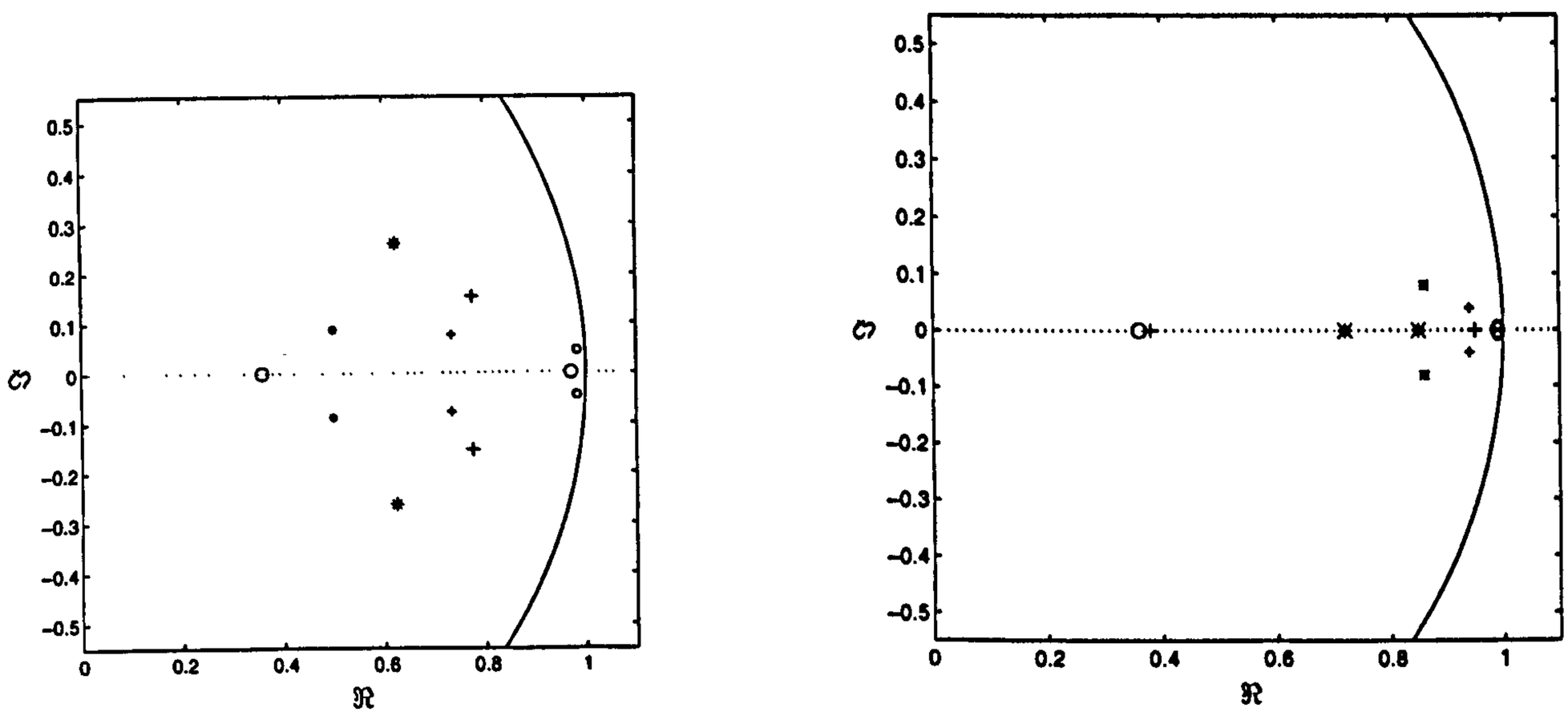
Figure 5.6: Linear shift operator model. The dark bars show errors of models trained in prediction mode, the white bars results obtained with models trained in simulation mode. The delays are 7ms for $T_s = 1$ms, and 10ms for $T_s = 5$ms and $T_s = 10$ms. Note, that for $T_s = 1$ms, the training MSPEs are in the order of $10^{-7}$ and therefore not visible in the plots.

For the models identified in prediction mode, a strong dependency of the errors on the sampling period can be observed. The Training MSPE is very small for small sampling periods, and increases for larger sampling periods. The very small MSPE for $T_s = 1$ms indicates that the identified model only propagates the previous system output. This is supported by the observation that the training and the test MSSEs are maximal for $T_s = 1$ms. The simulation errors decrease for sampling periods larger than 1ms and are minimal for $T_s = 5$ms for the fast muscle and $T_s = 10$ms for the slow muscle.

For the models identified in simulation mode, a similar dependency of the MSSEs on the sampling period, although less strong, can be observed.

The best modelling results for the fast muscle could be obtained with prediction mode at $T_s = 5$ms, although the simulation model performs similarly at this sampling period. For the slow muscle, the simulation model outperforms the prediction model at $T_s = 10$ms.

Further insight in the reason why the shift operator models do not perform well for fast sampling periods can be gained by analysing the location of the poles of the identified models. These are shown in Figure 5.7. The fast poles of the models with $T_s = 1$ms (marked by o) are located very close to the point $[1, 0j]$. Thus, sensitivity problems during the parameter estimation are likely as very small changes of the parameters will have a relatively large effect on the characteristics of the model. As the sampling period increases, the fast poles move further inside the unit circle, and the parameter estimation becomes more robust. This observation is similar for prediction and simulation modes.




(a) Fast muscle.

(b) Slow muscle.

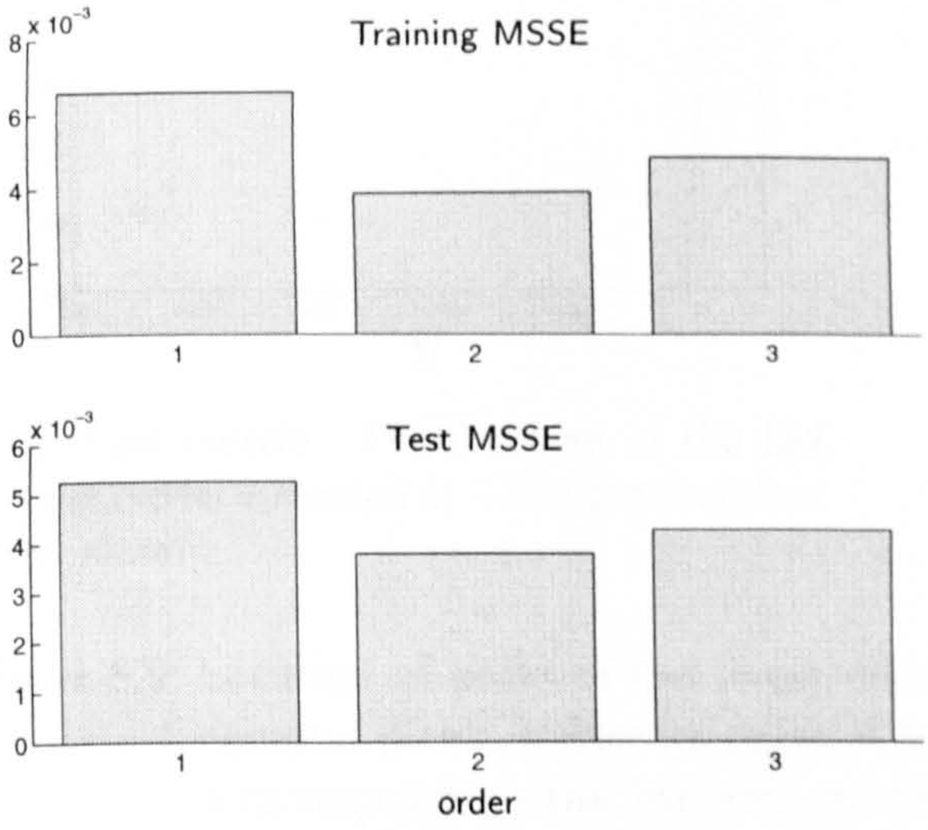
Figure 5.7: Location of the poles of the linear shift operator models in the complex plane. o denotes poles with $T_s = 1$ms, + poles with $T_s = 5$ms, and $*$ corresponds to $T_s = 10$ms. Large symbols denote models identified in simulation mode, smaller symbols correspond to models identified in prediction mode. The border of the stability region (the unit circle) is shown as a solid line.

### 5.3.2 Delta Operator Model

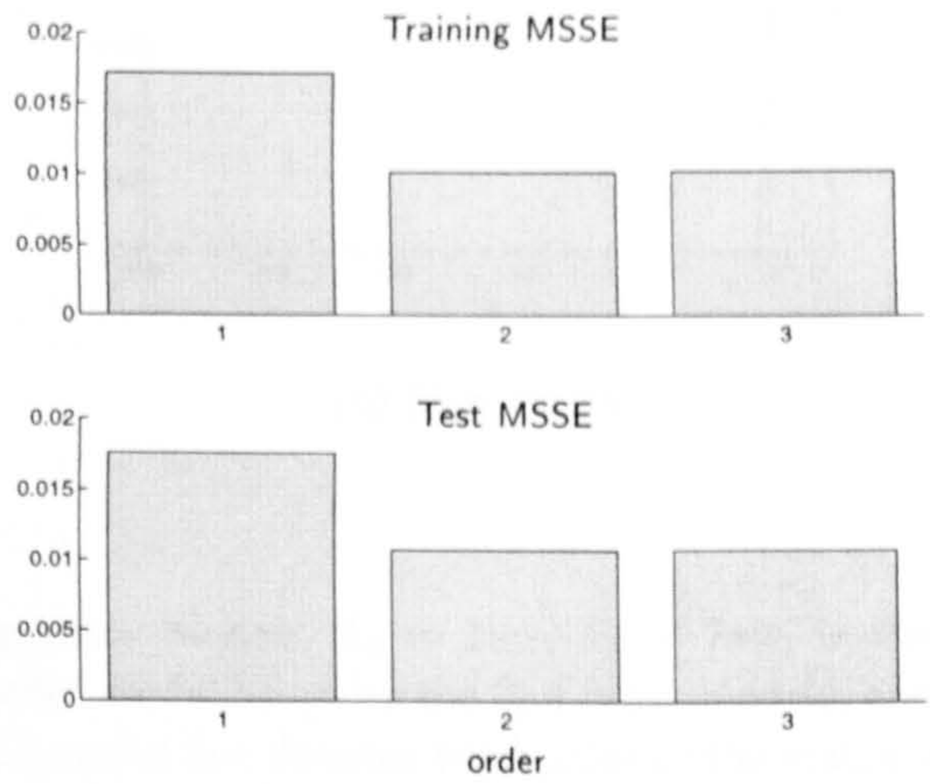
The delta operator model structure is used to identify linear models in simulation mode with the original sampling period $T_s = 1$ms and a delay of $T_d = 7$ms. As outlined in Section D.2.2,

the delta operator domain is not subject to the sensitivity problems encountered with the shift operator domain for very small sampling periods. We expect therefore that the delta operator models perform well with the original short sampling period.

To investigate the influence of the model order on the performance, models of dynamic order $n = [1, 2, 3]$ are identified. The results are shown in Figure 5.8.




(a) Fast muscle. The minimal Test MSSE is $3.80 \times 10^{-3}$ for the 2nd order model.




(b) Slow muscle. The minimal Test MSSE is $10.7 \times 10^{-3}$ for the 2nd order model.

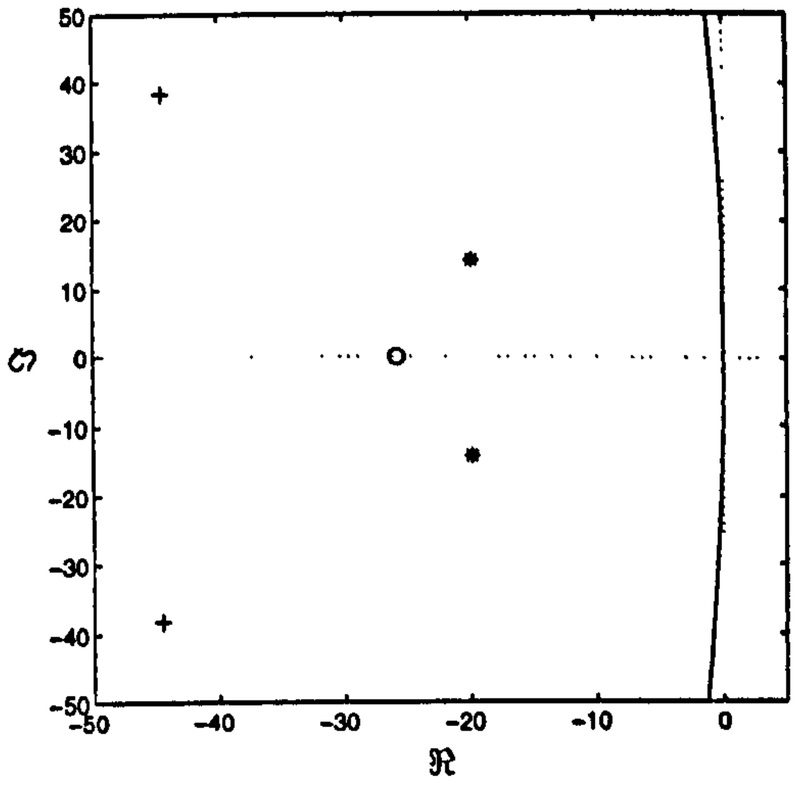
Figure 5.8: Linear delta operator model, $T_s = 1$ms, $T_d = 7$ms.

The models with the minimal training and test errors are of 2nd order for both the fast and the slow muscle. This supports the choice made for the shift operator model in the previous section.

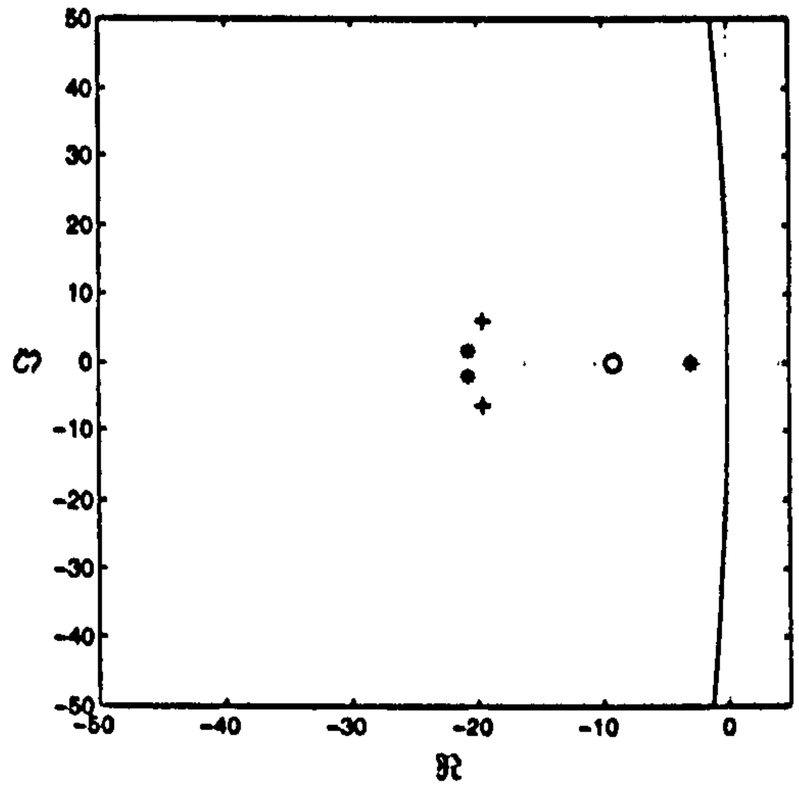
Comparing the results of the 2nd order models with the shift operator models for $T_s = 1$ms shown in Figure 5.6, the delta operator models clearly outperform the shift operator models. Even when the results obtained with the delta operator models are compared with the best shift operator model results, the delta operator models performs better. It should, however, be noted that the parameters of the delta operator model need to be estimated in simulation mode which is significantly more computationally expensive than parameter estimation using prediction mode (cf. Example E.4.2 on page 152).

The reason why the delta operator models perform well even with a very small sampling period becomes clearer when the locations of the model poles are analysed. These are shown in Figure 5.9. All poles are located clearly inside the region of stability. Thus, sensitivity problems for the estimation of the model parameters are not present here.

Simulation results for some typical data sets are shown in Figure 5.10 for the fast muscle and in Figure 5.11 for the slow muscle. The outputs of the linear models match the output of




(a) Fast muscle. The 3rd pole of the 3rd order model is located at $-165$ and therefore not shown.




(b) Slow muscle.

Figure 5.9: Location of poles of the linear delta operator models, $T_s = 1$ms, $T_d = 7$ms, in the complex plane. o denotes poles of 1st order model, + poles the 2nd order models, and $*$ corresponds to the 3rd order model. The solid line denotes the border of the stability region.

the real muscles only on average. Large modelling errors are present for very high and very low muscle activation.

### 5.3.3 Conclusions

In this section linear ARX model structures were used to model isometric contraction of muscle. Model structures with a dynamic order of 2 and an input delay of $T_d = 7 \ldots 10$ms were found to be suitable for both the fast and the slow muscle.

Experiments with models whose parameters were identified in shift operator notation show that the original sampling period of $T_s = 1$ms is not suitable. Parameter identification in prediction mode fails completely for this sampling period, and estimation of the parameters in simulation mode has convergence problems. Analysis of the location of the model poles in the complex plane shows that these problems are due to the fact that the poles are located very close to the point $[1, 0j]$ in the complex plane which leads to sensitivity problems for the parameter estimation. It was found that these problems can be overcome by increasing the sampling period, i.e. by down-sampling the data. The poles of the model are then located further inside the region of stability and the parameter estimation becomes more robust. The optimal sampling periods were found to be $T_{s,\text{fast}} = 5$ms for the fast muscle and $T_{s,\text{slow}} = 10$ms

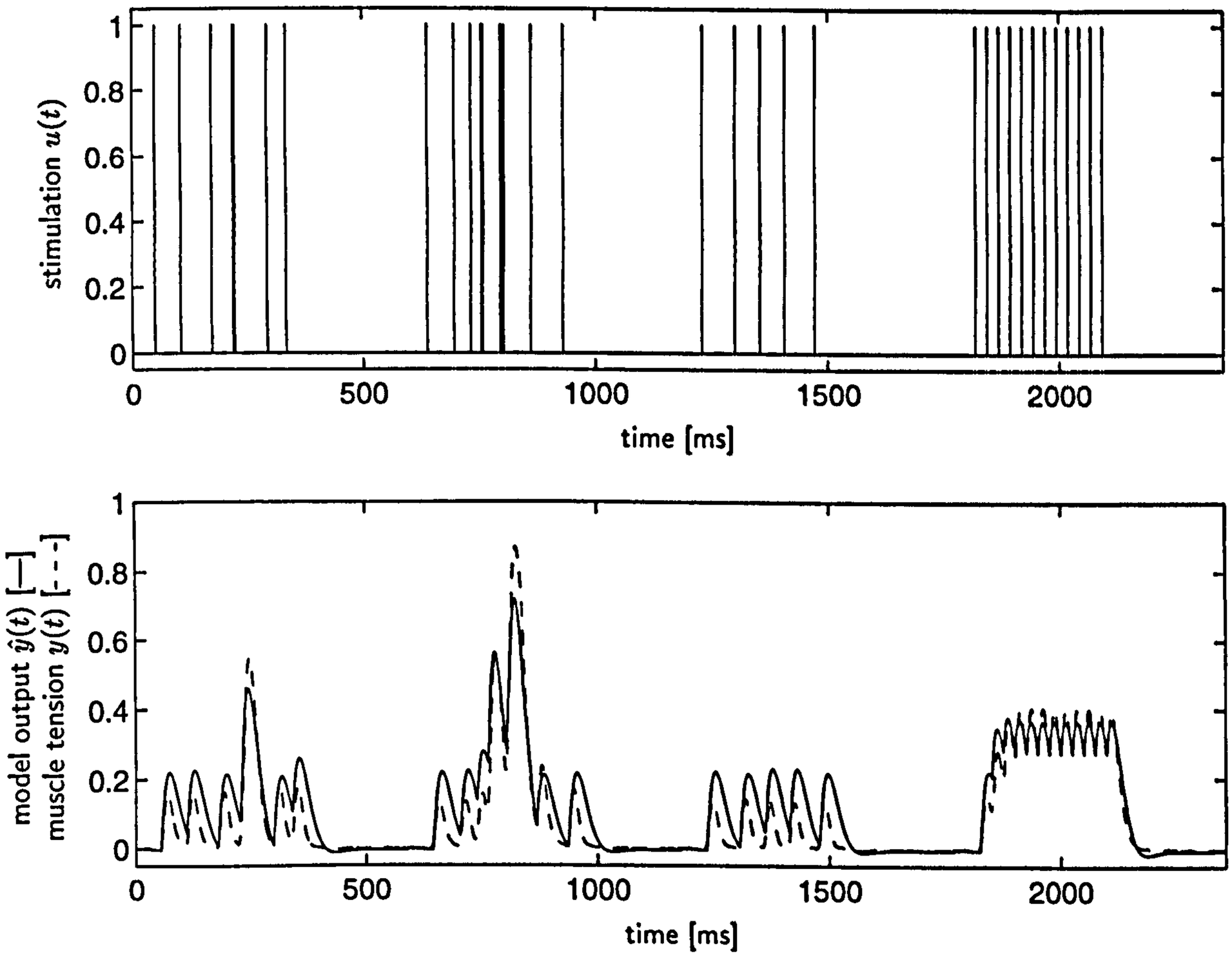



Figure 5.10: Fast muscle: Comparison of the simulated output of the linear 2nd order delta operator model (Figure 5.8(a)) with the output of the muscle for four typical data sets from the test data. The data sets are shown concatenated; the simulation is restarted after each set. For a low level of muscle activation, the output of the model is too high (e.g. 3rd data set), and for high activation of the muscle the model output is too low (e.g. 2nd data set). The initial muscle behaviour for a stimulation with a constant frequency burst cannot be correctly predicted (4th data set).

for the slow muscle. The delay is $T_d = 10$ms in both cases. Parameter identification in prediction mode gives results similar to those obtained by parameter optimisation in simulation mode, but the latter technique is much more computationally expensive.

The need to increase the sampling period can be avoided by using model structures whose parameters are identified in the delta operator domain. Model structures with a dynamic order of 2 and an input delay of $T_d = 7$ms were found to be appropriate for both the fast and the slow muscle. The modelling results are slightly better than those obtained with the shift operator model. Owing to the fast sampling, the computational cost is significantly larger than for the shift operator models with enlarged sampling periods.

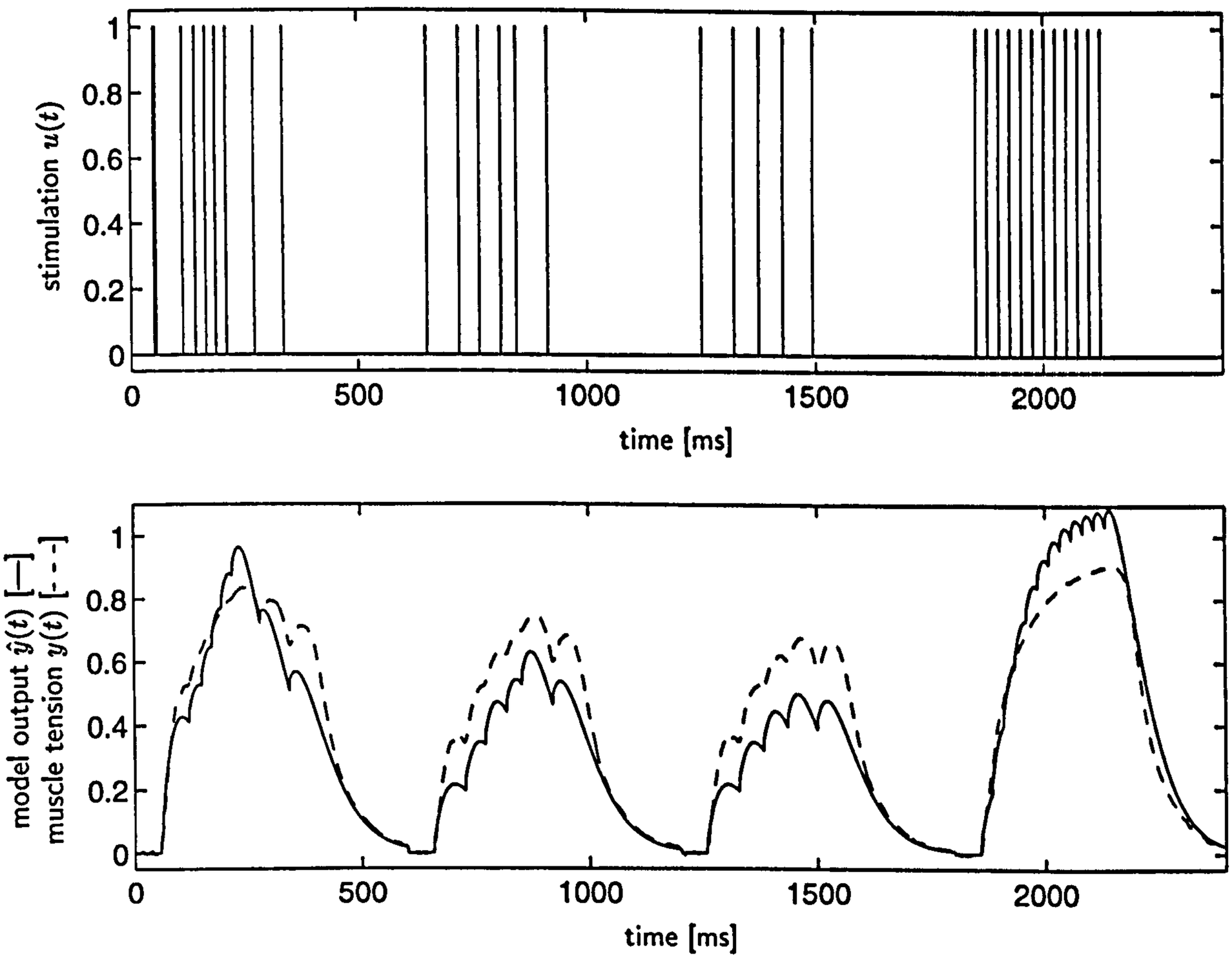



Figure 5.11: Slow muscle: Comparison of the simulated output of the linear 2nd order delta operator model (Figure 5.8(b)) with the output of the muscle for four typical data sets from the test data. The data sets are shown concatenated; the simulation is restarted after each set. For low and medium levels of muscle activation, the output of the model is too low (e.g. 2nd and 3rd data set). For high activation of the muscle the model output is too high (e.g. 1st and 4th data set).

The process of increasing the sampling period for the data gives a reduction of computational cost but involves, necessarily, a loss of information. As pointed out in Section 5.2.2, high frequency components of the input stimulation sequence are affected in particular. As the linear model does not include the force–frequency characteristics of the system, this loss does not have a significant effect here. It is, however, to be expected that down-sampling will have a larger effect with non-linear model structures.

Based on the results obtained in this section, the following model structures are chosen as initial structures for non-linear identification experiments:

- A shift operator model of 2nd order with a sampling period of 5 and 10ms. The parameter optimisation can be based on the prediction mode which is fast, or on the simulation mode which is much slower but produces slightly better results for the slow muscle.

- A delta operator model of 2nd order with the original sampling period of 1ms. With this structure, the parameter optimisation will be based on the simulation mode.

In the following section, Local Model Networks based on local models with these linear structures will be further investigated.

## 5.4 Non-linear Local Model Network for Isometric Contraction

In this section, non-linear Local Model Network structures, as introduced in Chapter 2, will be investigated as a modelling technique for isometric muscle contraction. Linear local models will be used.

The results of the previous section give guidelines for the initial choice of some model parameters, such as the dynamic order of the model, the time delay, the sampling period, and the parameter optimisation algorithm. In this section, the following model design parameters will be investigated in detail:

- the elements of the scheduling vector (cf. Section 2.2.4);

- the structure of the validity functions, i.e., the number of units (which determines the number of local models), their type and shape (cf. Section 2.2.3);

- the structure of the local models, i.e., state space or input–output (cf. Section 2.2.2) [1].

- the discrete time domain (shift or delta operator) and, related to this, the parameter estimation technique (cf. Section 2.3) and the sampling period.

Starting with the model structures suggested at the end of Section 5.3.3, we will first investigate the optimal choice of the elements of the scheduling vector and the structure of the scheduler (without optimising the shape of the validity functions). The LMN structures will be based on local ARX models as used in the previous section, and aspects of the discrete time domain, parameter estimation and the sampling period will be considered.

Based on the results obtained, structures for the local models other than the ARX form will be investigated and the shape of the validity functions will be optimised in Section 5.4.2. The Local Model Networks obtained will be analysed and validated in Section 5.4.3.

As for the single linear models, 30 data sets are used to estimate the model parameters, and the model performance is tested on the remaining data sets. The same mean squared

[1] Note that these structures are equivalent when working with a single linear model, but need to be distinguished for the Local Model Network.

error measurements as defined at the beginning of Section 5.3 are used to compare different models.

### 5.4.1 Selection of the Scheduler and Parameter Estimation Domain

Aspects of selecting an appropriate scheduling vector for a given system have been discussed in Section 2.2.4. As outlined there, the scheduling variables should represent the non-linearities of the system.

For isometric contraction of muscle stimulated with supramaximal pulses a major non-linear characteristic is the force–frequency relationship (Cooper and Eccles 1930, Binder-Macleod and Barrish 1992, Stein and Parmiggiani 1981, Parmiggiani and Stein 1981) which depends on the activation of the muscle. A straightforward approach is to characterise the activation by the muscle force. Hence, the delayed output force can be used as a scheduling variable. This setup will be called "output scheduling". Alternatively, a measurement of the current stimulation frequency can be used for scheduling. As the stimulation frequency is a characteristic of the system input this setup will be termed "input scheduling". Both alternatives will be discussed below. They result in structures where the scheduling vector has only one element. This reduces the input space for the validity functions to one dimension and thus greatly facilitates the selection of the optimal network structure.

In this section we will work with LMN structures which have uniformly distributed B-spline validity functions, and whose local models are linear ARX models. When the local model parameters are identified directly in the shift operator domain the model is referred to as a shift operator model. The term delta operator model denotes an LMN structure whose parameters are identified in the delta domain but which is implemented as an input–output LMN with local ARX models in shift operator notation.

#### Output Scheduling

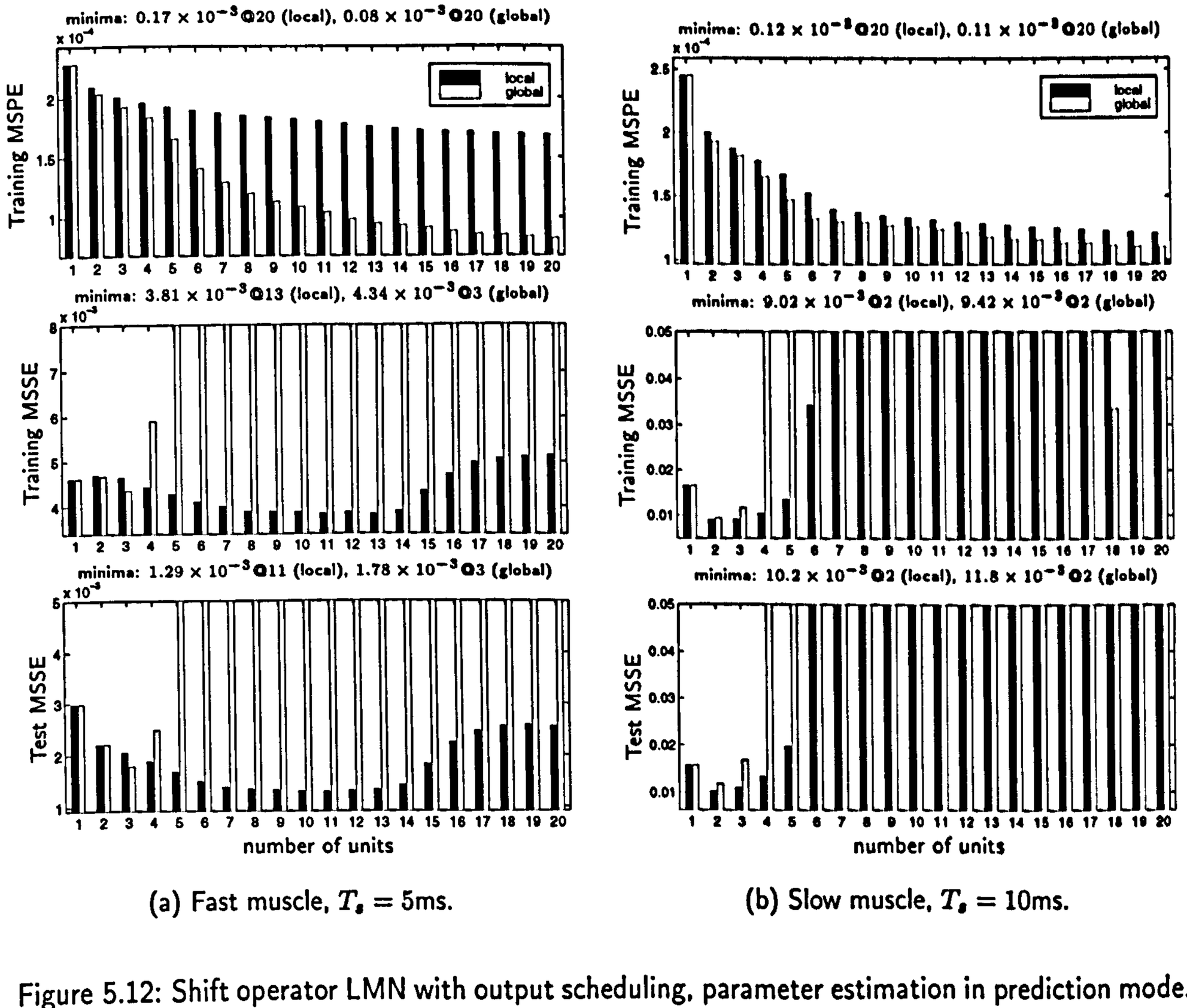
A straightforward assumption is that the non-linearities of the muscle depend on its activity, and to characterise the muscle activity by the force produced by it. Thus, the delayed system output can be chosen as the scheduling variable,

$$\phi(t_k) = y(t_{k-1})\,, \tag{5.1}$$

where $t_k$ denotes the discrete time. This setup has previously been used successfully with the fast muscle. In these identification experiments, the structure of the LMNs was based on a scheduler with Gaussian bells and local ARX models in shift operator notation with a sampling period of $T_s = 5\text{ms}$. The parameters were estimated using local optimisation in prediction mode. The results are reported in (Gollee *et al.* 1994, Gollee *et al.* 1997, Gollee and Hunt 1997).

In this section, we will investigate whether this scheduling approach is suitable for both the fast and the slow muscle, and what effect it has to estimate the model parameters in the delta domain with the original sampling period of $T_s = 1$ms. Various structures with different numbers of uniformly distributed B-spline validity functions are compared.

**Shift Operator Models** Based on the results obtained from the experiments with linear model structures, we compare models in shift operator notation with an increased sampling period. Results for LMNs with different numbers of units whose parameters were estimated in prediction mode with local and with global optimisation are shown in Figure 5.12. Parameter




(a) Fast muscle, $T_s = 5$ms.

(b) Slow muscle, $T_s = 10$ms.

Figure 5.12: Shift operator LMN with output scheduling, parameter estimation in prediction mode. Results with local optimisation are shown with dark bars, global optimisation results are printed with light bars. For the Training and Test MSSE some large errors exceed the axis range and are therefore truncated.

estimation in simulation mode was also investigated. The results obtained are, however, significantly worse than those with prediction mode and are therefore not shown.

Global parameter optimisation results in smaller one-step-ahead prediction errors on the

training data than local learning. However, when tested in simulation mode, the models obtained with global parameter optimisation perform significantly worse than models whose parameters are estimated using local learning. This is consistent with the results obtained in Example E.4.3 on page 153.

For both the fast and the slow muscle, the Training MSPE decreases monotonically for larger networks.

For the fast muscle, the best simulation error results are achieved with models of 8 to 13 units. Whereas the Training MSSE does not improve significantly compared with the linear model (i.e., an LMN with 1 unit), some improvement can be observed on the test data.

For the slow muscle, the best model has only 2 units. Its performance is only slightly better than that of a single linear model (i.e., an LMN with 1 unit). For larger networks, the performance of the models in simulation mode deteriorates drastically.

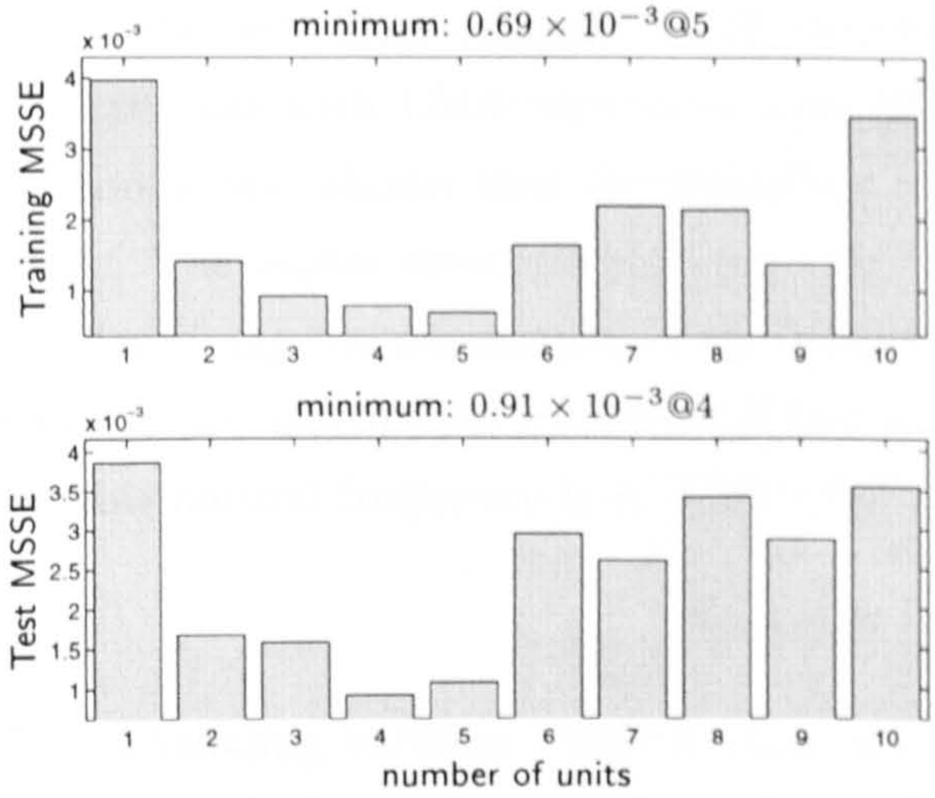
For both the fast and the slow muscle, over-fitting can be observed for large model structures: while the Training MSPE decreases for larger networks, the Training and the Test MSSE decrease only for model structures up to a certain (optimal) network size. When the network size is growing further, these errors start to increase.

**Delta Operator Models** Based on the results obtained from the experiments with linear model structures, we investigate LMNs with local ARX models whose parameters are estimated in delta operator notation. The sampling period is unchanged, $T_s = 1$ms, and the parameters are estimated in simulation mode. Modelling results of LMN structures with different numbers of units are shown in Figure 5.13.

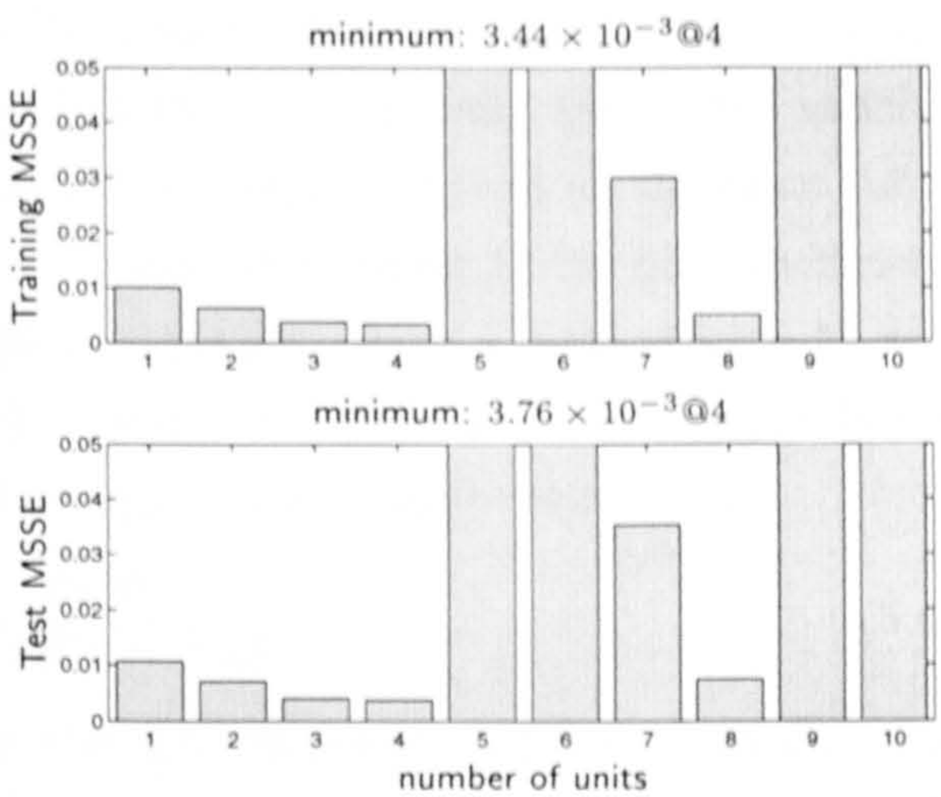
For the fast muscle, the optimal network size is 4 to 5 units. The modelling results for LMNs of this size are significantly better than the results obtained with a single linear model. Comparing with Figure 5.12(a), the results of the delta operator LMNs clearly outperform the results obtained with LMNs in shift operator notation. For networks which are larger than the optimal size, over-fitting becomes a problem, and also the parameter identification process takes a very long time.

For the slow muscle, the modelling results are better than those obtained with the shift operator LMNs (cf. Figure 5.12(b)), and outperform a single linear model structure on both training and test data sets. The optimal network size is 3 to 4 units. For larger networks, the parameter optimisation does not always converge, and the performance deteriorates drastically.

Although the LMNs in delta operator notation perform better than those in shift operator notation, the results obtained for the slow muscle are still significantly worse than the results for the fast muscle. It seems therefore that output scheduling is not the optimal way of switching between local linear models in an LMN structure for the slow muscle.




(a) Fast muscle.




(b) Slow muscle.

Figure 5.13: Delta operator LMN with output scheduling, parameter estimation in simulation mode, $T_s = 1$ms. For the slow muscle some large errors exceed the axis range and are therefore truncated.

### Input Scheduling

As an alternative to representing the muscle activity by the output it is possible to obtain a measurement for the muscle activation from the input. The muscle activation is related to the current stimulation frequency. However, the current frequency cannot be determined for a sequence of pulses with randomly varying IPI. Thus, a transformation $\Gamma : u \to [0 \ldots 1]$ must be defined which transforms the pulse-like muscle input $u(t)$ into a continuous-in-value sequence which is suitable for scheduling, i.e.

$$\phi(t) = \Gamma\{u(t)\}\,. \tag{5.2}$$

For the continuous-in-value sequence $\phi$, it is desirable that i) information about the current stimulation frequency (i.e., the IPI) is preserved, and ii) the new variable changes smoothly.

**Pre-processing of the Input Pulses** A simple second-order low-pass filter with the transfer function

$$G_{pre}(s) = \frac{K}{s^2 + 2\xi\omega_n\, s + \omega_n^2} \tag{5.3}$$

can be used to obtain the scheduling variable from the input pulses. The order of the numerator is chosen to be zero to ensure that the initial value of the pulse response is zero. Thus, an input pulse does not cause the filtered variable to jump. The damping factor $\xi$ is set equal to one to ensure a critically damped response. The factor $K$ is selected such that the filtered

variable is approximately in the input range of the validity functions, i.e. between 0 and 1. Thus, the only filter constant which remains to be determined is the natural frequency $\omega_n$. Experiments with LMN structures with different numbers of uniformly distributed validity functions have shown that the modelling results do not strongly depend on the exact value of $\omega_n$. The model structure can compensate for a non-optimal choice of the filter coefficient by identifying the parameters of the local models accordingly. Thus, a value of $\omega_n = 50$ was found to be appropriate for both the fast and the slow muscle. The corresponding gain factor for this natural frequency is $K = 25 \times 10^3$. The resulting filter transfer function is

$$G_{pre}(s) = \frac{25000}{s^2 + 100\,s + 2500}. \tag{5.4}$$

The scheduling variable which is obtained with this filter is shown in Figure 5.14 for some characteristic input patterns. In addition, the output of the fast muscle (which is the scheduling variable for the LMNs with output scheduling of the fast muscle) is shown in the bottom plot. Note that the obtained scheduling variable, $\phi$, resembles a scaled linear approximation

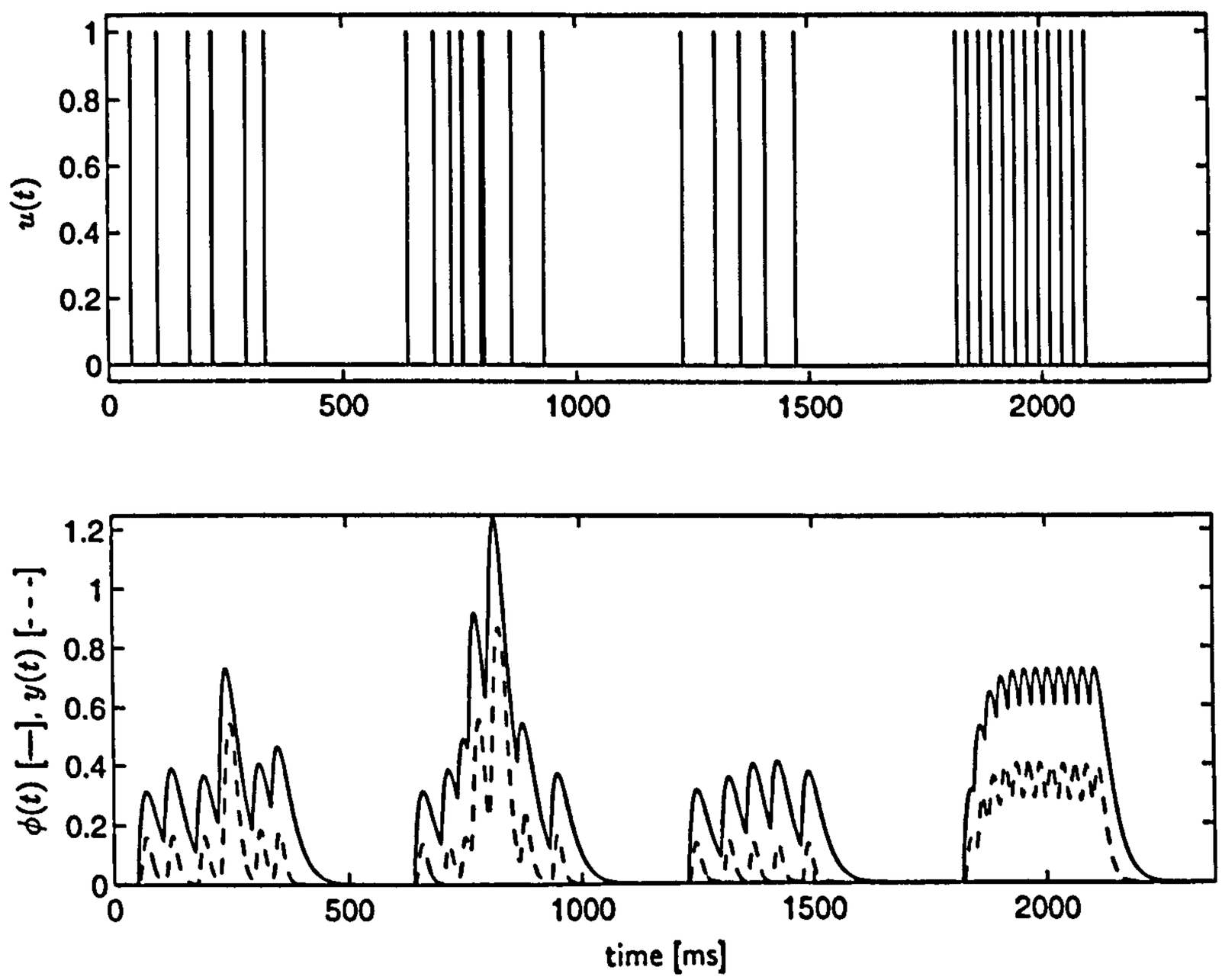


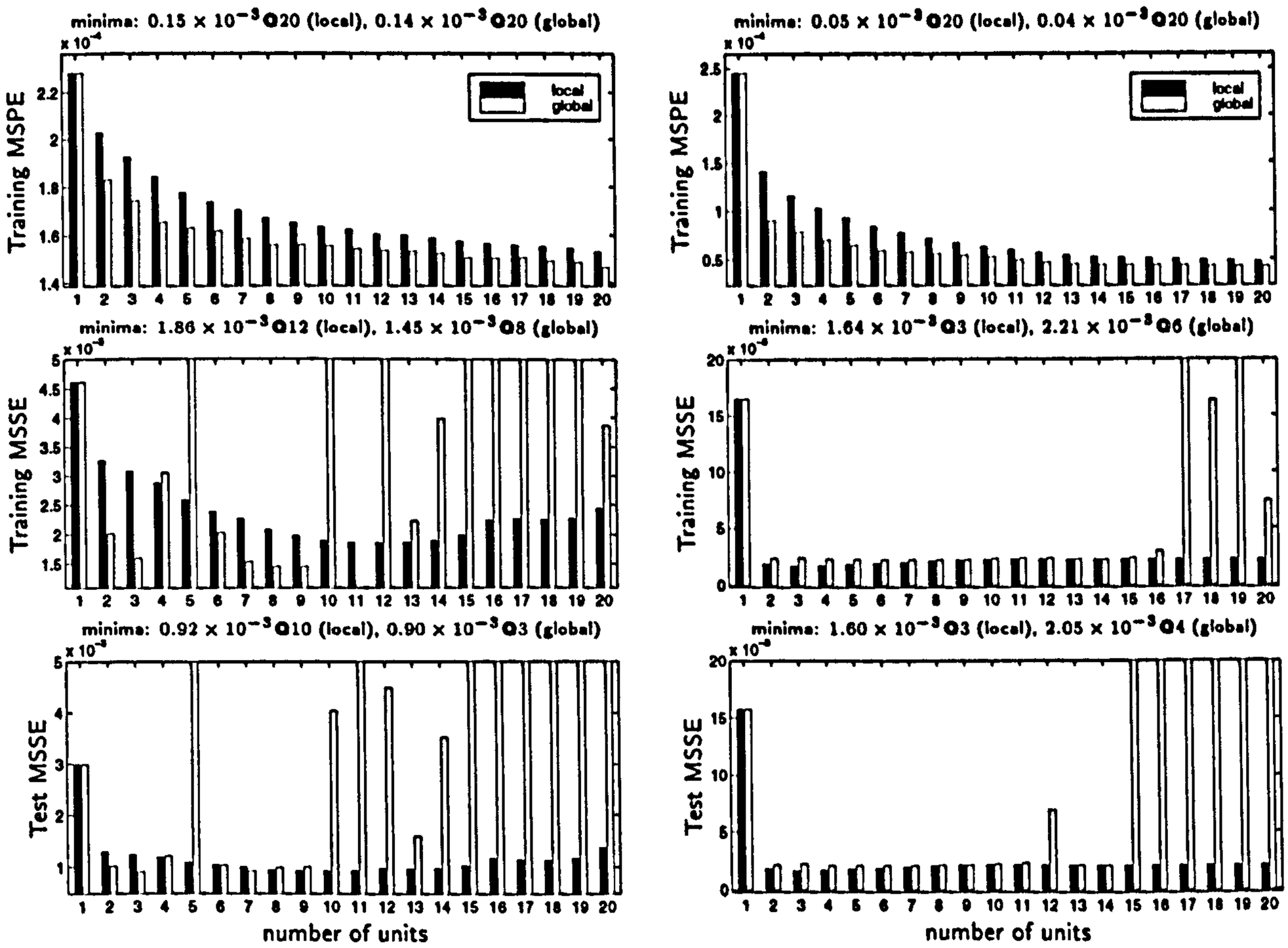

Figure 5.14: Scheduling variable $\phi$ (bottom) obtained from some characteristic input pulse sequences $u$ (top). The output of the fast muscle, $y$, is shown as the dotted line in the bottom plot.

of the output of the fast muscle, $y$. Although this was not intended when the filter parameters were selected, it is a direct results of choosing to work with a 2nd order filter. As scheduling

on the system output worked well for the fast muscle, using a scheduling variable which has similar characteristics as the output of the fast muscle is expected to be suitable for both the fast and the slow muscle.

In this section, we will investigate the performance of Local Model Networks in shift and in delta operator notation which use this filtered input for scheduling.

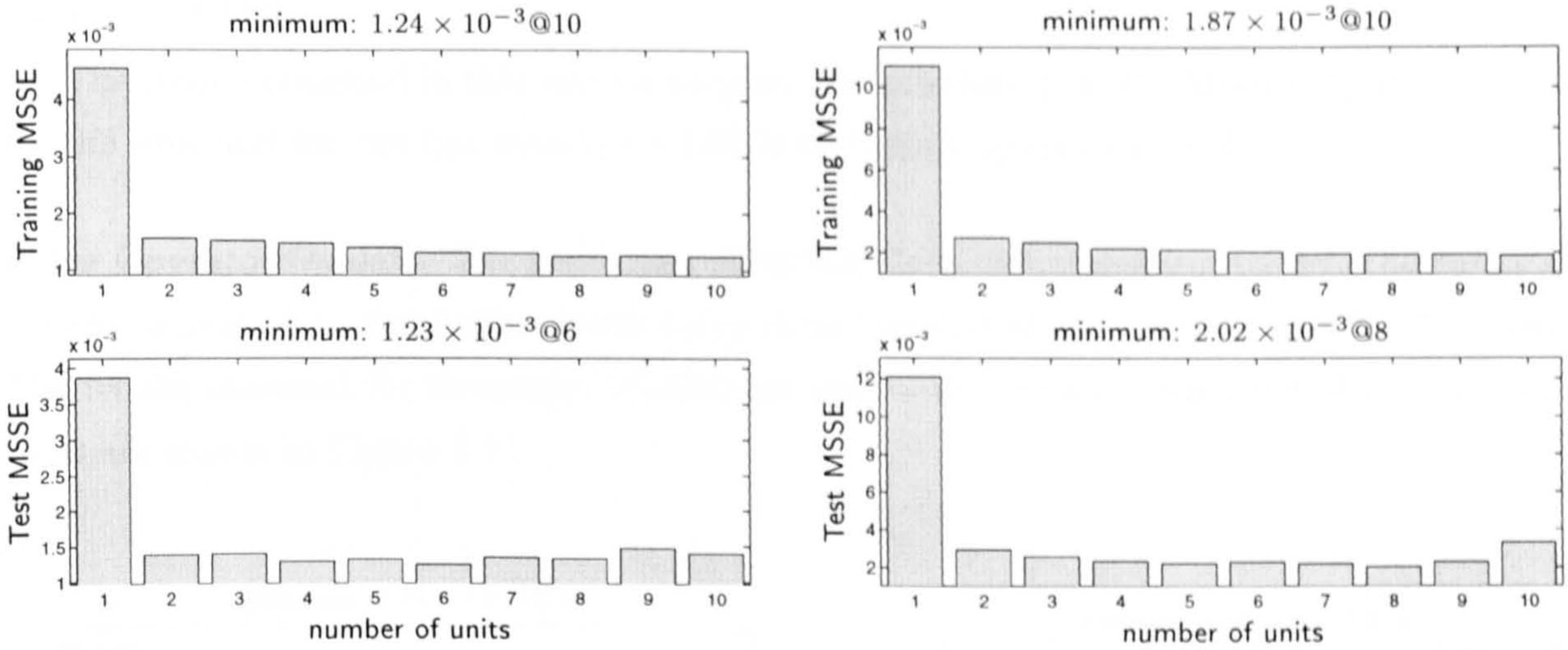
**Shift Operator Models** Based on the results obtained from experiments with linear model structures, Local Model Networks with local ARX models in shift operator form and uniformly distributed validity functions are used with a sampling period of $T_s = 5$ms for the fast muscle and $T_s = 10$ms for the slow muscle. The modelling results for structures with different numbers of local models are shown in Figure 5.15 for parameter estimation in prediction mode and in Figure 5.16 for parameter estimation in simulation mode.




(a) Fast muscle, $T_s = 5$ms.

(b) Slow muscle, $T_s = 10$ms.

Figure 5.15: Shift operator LMN with input scheduling, parameter estimation in prediction mode. Results with local optimisation are shown with dark bars, global optimisation results are printed with light bars. For the Training and Test MSSE some large errors obtained with global optimisation exceed the axis range and are therefore truncated.




(a) Fast muscle, $T_s = 5$ms.

(b) Slow muscle, $T_s = 10$ms.

Figure 5.16: Shift operator LMN with input scheduling, parameter estimation in simulation mode.

When the parameters are estimated in prediction mode, global learning results in smaller Training MSPEs than local learning. The Training and Test MSSEs are, however, more consistent for local learning. For some specific network sizes, models whose parameters are identified with global learning fail to generalise satisfactorily from the one-step-ahead prediction used for parameter estimation to the infinite prediction horizon employed for the simulation test (e.g. structures with 5, 10...12 and 14...20 units for the fast muscle and with 12, 15...20 units for the slow muscle). When the parameters are optimised in simulation mode, the Training MSSEs decrease with larger networks, whereas the Test MSSEs start to decrease initially but increase for networks which are larger than the optimal size.

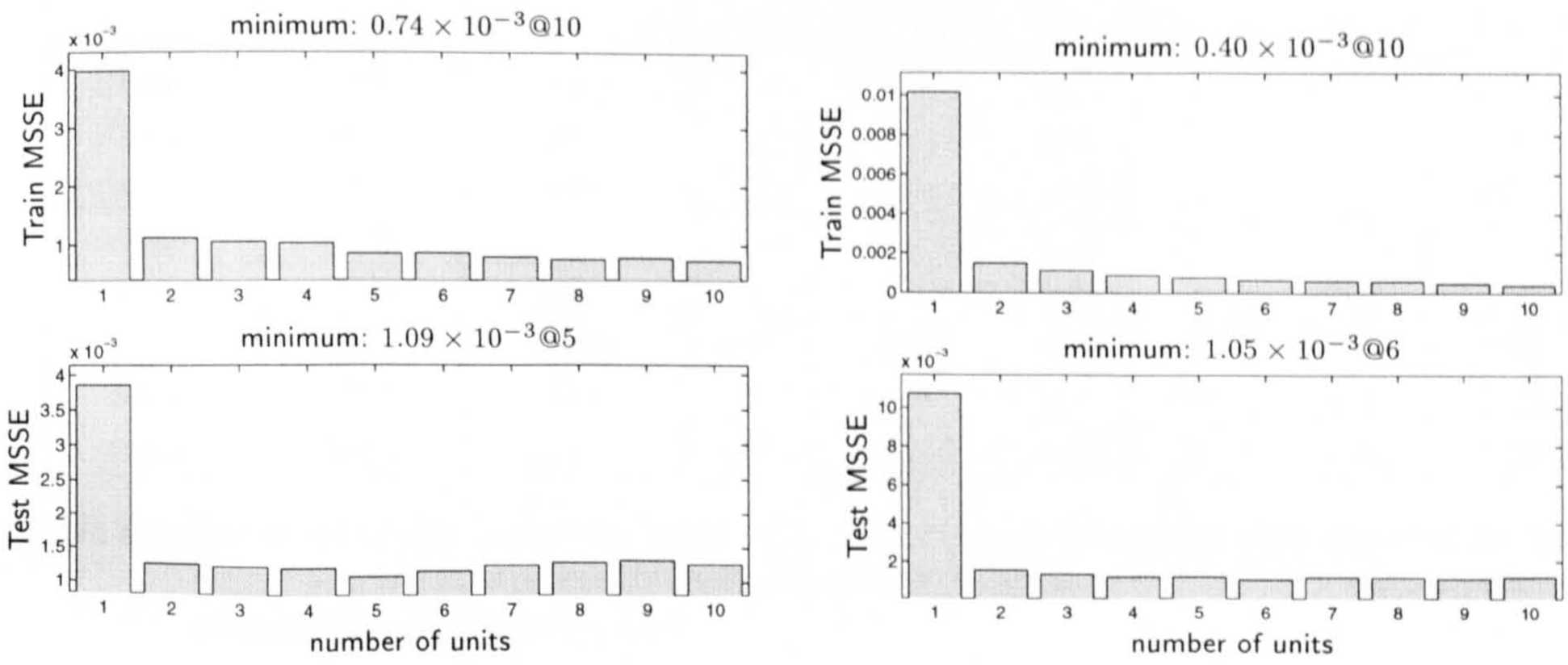
For modelling the fast muscle, the optimal network size is 10 to 14 units for LMNs whose parameters are optimised in prediction mode, and 4 to 8 units for models whose parameters are estimated in simulation mode. The Training and Test MSSEs are smaller than those obtained with shift operator models with output scheduling, cf. Figure 5.12(a). However, the models perform less well than the best delta operator models with output scheduling, cf. Figure 5.13(a). The variation of the results for models of different sizes is significantly smaller for LMNs with input scheduling than for models with output scheduling. The models whose parameters are estimated in prediction mode perform similarly to LMNs whose parameters are estimated in simulation mode.

For the slow muscle, the optimal network size is 2 to 4 units for models whose parameters are optimised in prediction mode, and 6 to 9 units for LMNs whose parameters are identified in simulation mode. The results are significantly better than those obtained with output scheduling in shift operator form, cf. Figures 5.12(b). The shift operator model results

clearly outperform also the models obtained in delta operator form with output scheduling, see Figure 5.13(b).

The results obtained in this section suggest that scheduling on the filtered input is suitable for the slow and for the fast muscle for LMNs with shift operator models.

**Delta Operator Models** For LMN structures with local delta operator models the parameters are optimised in simulation mode using data sampled at the original interval $T_s = 1$ms. The results obtained for structures of different sizes with uniformly distributed validity functions are shown in Figure 5.17.




(a) Fast muscle.

(b) Slow muscle.

Figure 5.17: Delta operator LMN with input scheduling, parameter optimisation in simulation mode, $T_s = 1$ms.

For the fast muscle, the optimal networks consist of 4 to 6 units. The best results for the slow muscle are obtained with structures of 5 to 8 units. For both muscles, the test errors deteriorate for structures with more units than the optimal size while the training error decreases further.

The modelling errors for the fast muscle are only slightly better than those obtained with the shift operator models with input scheduling. However, they are slightly larger than the errors obtained with delta operator models with output scheduling, cf. Figure 5.13(a).

For the slow muscle, the modelling errors are smaller than those obtained with shift operator models with input scheduling. The results obtained with input scheduling clearly outperform the delta operator models with output scheduling, cf. Figure 5.13(b).

The results show that scheduling on the filtered input is adequate for both the fast and the slow muscle.

### Discussion

In this section, two approaches for the selection of the scheduling variable for LMN structures were evaluated. Each approach was tested with networks of varying size and structure. Parameter identification in the shift operator and in the delta domain using prediction and simulation mode were compared. The simulation results on the training and test data for the different network structures are summarised in Table 5.1. The results with single linear models are included for comparison.

| Scheduling | Parameter domain | Estimation mode | Fast muscle | | | Slow muscle | | |
|---|---|---|---|---|---|---|---|---|
| | | | # units | MSSE ($\times 10^{-3}$) Training | Test | # units | MSSE ($\times 10^{-3}$) Training | Test |
| (linear) | shift | pred. | 1 | 4.61 | 2.98 | 1 | 16.5 | 15.7 |
| (linear) | shift | sim. | 1 | 4.55 | 3.90 | 1 | 10.1 | 12.0 |
| (linear) | delta | sim. | 1 | 3.87 | 3.80 | 1 | 10.2 | 10.7 |
| output | shift | pred. | 11 | 3.83 | 1.29 | 2 | 9.02 | 10.2 |
| output | delta | sim. | 4 | 0.78 | 0.91 | 4 | 3.44 | 3.76 |
| input | shift | pred. | 10 | 1.89 | 0.92 | 3 | 1.64 | 1.60 |
| input | shift | sim. | 6 | 1.31 | 1.23 | 8 | 1.91 | 2.02 |
| input | delta | sim. | 5 | 0.86 | 1.09 | 6 | 0.66 | 1.05 |

Table 5.1: Summary of the modelling results. For each setup, the network with the smallest Test MSSE is selected. For parameter estimation in prediction mode only the results obtained with local learning are shown.

Scheduling on the delayed output of the system works well with the fast muscle, but was found to fail when used with data from the slow muscle where the best model obtained is only slightly better than a single linear model. In a second approach, a filtered version of the input was used for scheduling. The filtering was aimed at obtaining a smooth scheduling variable which provides information about the frequency of the stimulation. It was found that the optimal filter gives a scheduling variable which is similar to a linear approximation of the output of the fast muscle. Scheduling on the filtered input works well for the fast and for the slow muscle and is therefore thought to be a scheduling approach which is suitable for both muscles[2].

Models identified in delta operator notation (which operate at the original sampling period of $T_s = 1$ms) perform better than similar structures in shift operator notation which operate at an increased sampling period. In particular for the slow muscle, models operating at $T_s = 1$ms clearly outperform the shift operator models operating at $T_s = 10$ms. This indicates that

[2]Note that LMNs with output scheduling in delta operator notation perform slightly better than the corresponding models with input scheduling. However, we aim to find a modelling technique which can be used with both the fast and the slow muscle. As output scheduling fails for the slow muscle, this scheduling approach is found unsuitable for a general model structure.

significant input information is lost when the sampling period is increased.

A straightforward extension to the approaches discussed here is to use a combination of output and input scheduling which results in a 2-dimensional scheduling space. Experiments with this setup do not indicate an improvement of the modelling performance. Owing to the higher dimensionality of the scheduling space, the scheduling space now includes areas with sparse data which leads quickly to difficulties with parameter identification, and the models obtained do not generalise in a satisfactory way.

### 5.4.2 Local Model Structure and Optimisation of the Validity Functions

In the previous section LMN structures with uniformly distributed B-spline validity functions and 2nd order local models in ARX form were used to identify the optimal scheduling variable and the best way of estimating the local model parameters. Local Model Networks with input scheduling whose parameters are optimised in the delta operator domain using a sampling interval of $T_s = 1$ms were found to perform best for the fast and for the slow muscle.

In this section, LMN structures with local models in state space form will be investigated. The parameters of the B-spline validity functions will be optimised to obtain model structures which are adapted to the complexity of the system.

#### Local Models in State Space Form

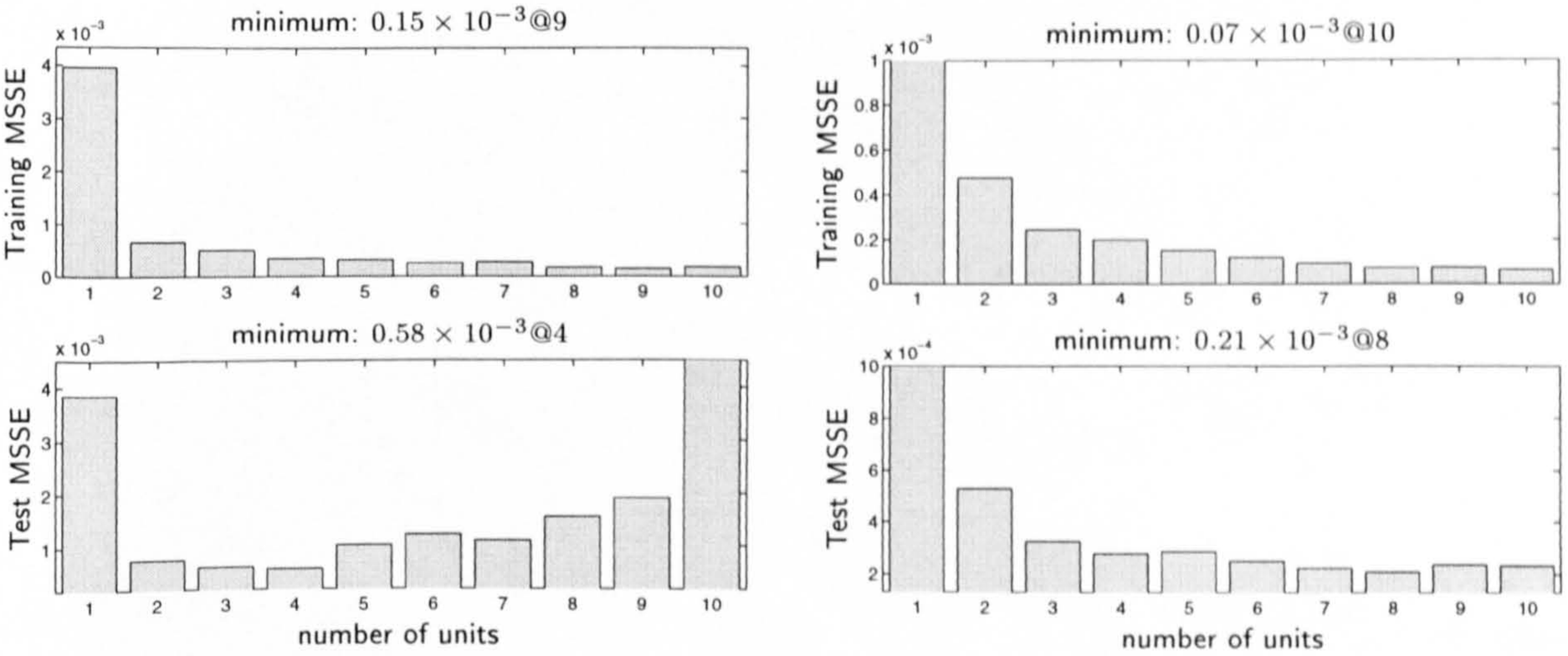
A general description of an LMN in state space form is given by equation (2.14) on page 14 for the continuous time domain. An equivalent delta domain structure (cf. Section D.2.2) will be used here to model the characteristics of the fast and the slow muscle. LMN structures with 2nd order local models in state space form will be investigated. The model parameters will be estimated in the delta domain, using simulation mode, with a sampling period of $T_s = 1$ms. The models have an input delay of $T_d = 7$ms. Structures with 1 to 10 uniformly distributed B-spline validity functions will be used. The filtered input, equation (5.4), is employed as the scheduling variable.

**Direct State Space Description** Local Model Networks in state space form are used where all model parameters $\{A_i, \underline{b}_i, \underline{d}_i^x; \underline{c}_i, d_i^y\}_{i=1}^M$ are optimised directly. The results for networks of different sizes are shown in Figure 5.18.

For the fast muscle, the optimal network size is 3 to 4 units. The performance on the test data deteriorates quickly for larger models. The results are significantly better than those obtained with local ARX models, cf. Figure 5.17(a).

For the slow muscle, networks with 7 to 9 units perform best. As for the fast muscle, the state space LMNs clearly outperform the LMNs with local ARX models, cf. Figure 5.17(b).

The results indicate that the higher number of free parameters in the more general local state space description leads to a significant improvement of the model performance compared




(a) Fast muscle.

(b) Slow muscle.

Figure 5.18: Direct state space LMN with input scheduling. Large errors which exceed the axis range are truncated.

to LMNs with local input–output models: for LMNs in input–output notation each second order local model has 5 parameters[3] which can be adjusted, while in LMNs in state space notation each second order local model has 11 free parameters[4].

**Eigensystem State Space Description** In Section 2.4.1, aspects of the stability of linear parameter varying systems, i.e., Local Model Networks, were discussed. In particular, it could be shown that models with stable local real eigensystems and common eigenvectors are stable. One way to use this property to identify Local Model Networks which are guaranteed to be stable was shown in Example E.4.5 on page 154. This approach seems to be well-suited for the modelling of muscle contraction, as muscle can generally be regarded as a well-damped system, making a restriction to real eigensystems a straightforward choice. We hope that by restricting the degrees of freedom of the model structure while considering the characteristics of the underlying system, the modelling performance can be improved, and the identification of the model parameters will be more robust. Note that, although the poles of the single linear muscle models shown in Figures 5.7 and 5.9 are complex, we expect that *locally* the system behaves well-damped.

For comparison, we first identify LMNs with local eigenvectors. As in Example E.4.5 on page 154, the parameters of the local state feedback matrices $\boldsymbol{A}_i$ are not identified directly. Instead, the local eigenvectors $\boldsymbol{V}_i \in \mathbb{R}^{2\times 2}$, and the local eigenvalues, $\underline{e}_i \in R^2$ are estimated

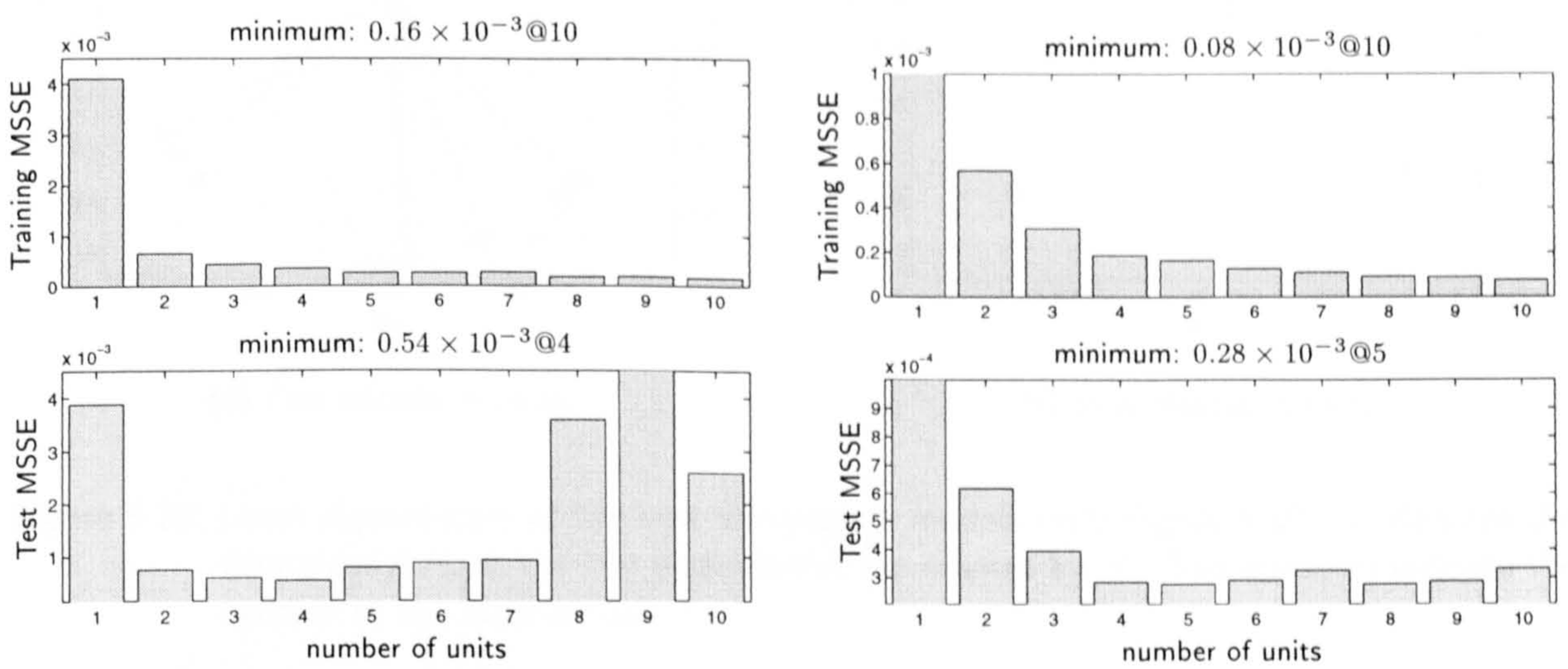
[3]The numerator and the denominator have two free parameters each, and the offset term has one parameter.

[4]The $\boldsymbol{A}$ matrix has four parameters, the $\underline{b}$ and the $\underline{c}$ vectors and the state offset term $\underline{d}^x$ have two parameters each, and the output offset $d^y$ has one parameter.

which ensures that the local eigensystems are real. The state feedback matrices are then obtained using the equation

$$\boldsymbol{A}_i = \boldsymbol{V}_i \operatorname{diag}\underline{e}_i \boldsymbol{V}_i^{-1}, \quad i = 1 \ldots M . \tag{5.5}$$

The modelling results are shown in Figure 5.19.




(a) Fast muscle.

(b) Slow muscle.

Figure 5.19: State space LMN with local eigenvectors with input scheduling. Large errors which exceed the axis range are truncated.

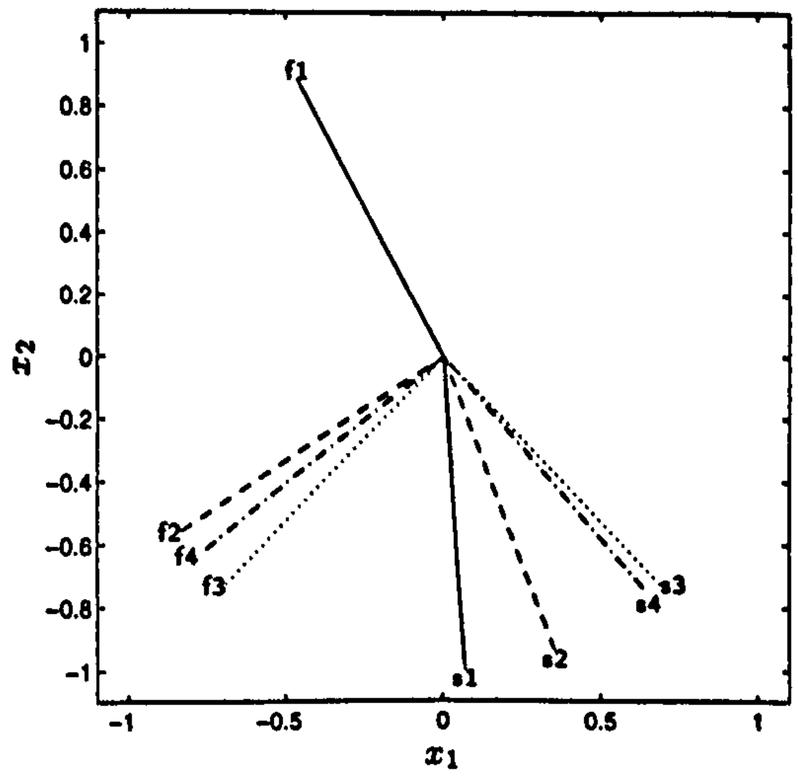
The results are very similar to those obtained with LMNs in direct state space form, cf. Figure 5.18. The optimal network size for the fast muscle is 3 to 4 units. The optimal network for the slow muscle is smaller than in Figure 5.18(b), consisting of only 4 to 6 units. We can therefore conclude that restricting the local state space models to real eigensystems is appropriate for the given application.

The eigenvector configurations of the best models (in terms of a minimal Test MSSE) from Figure 5.19 are shown in Figure 5.20. The eigenvectors vary significantly over a large range, and it is possible that unstable eigenvector configurations as described in Section 2.4.1 are present.

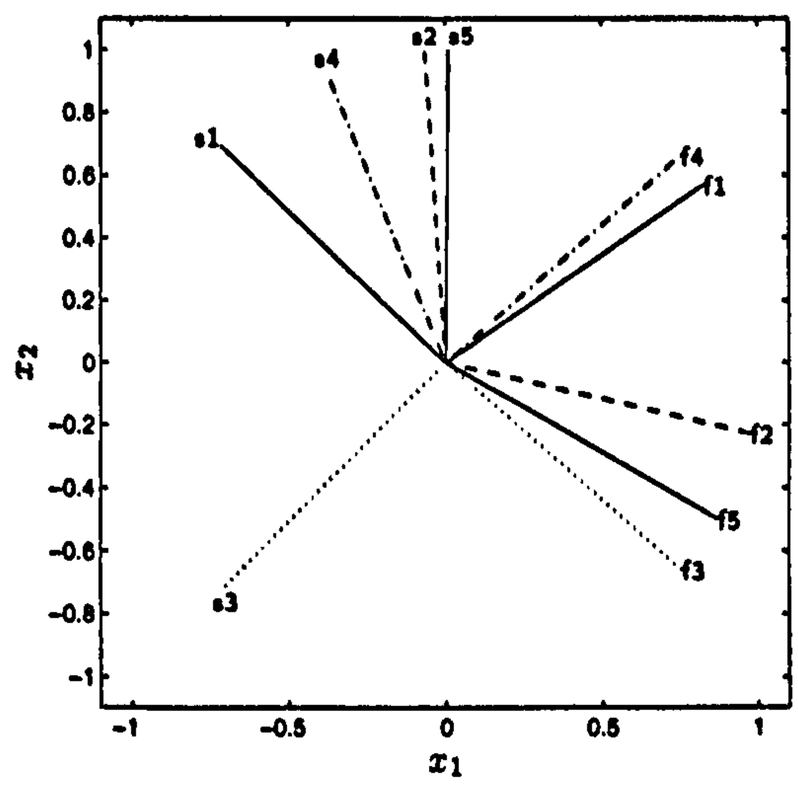
To ensure stability of the global model when all local eigensystems are stable, we restrict the LMN to have a set of common eigenvectors for all local models. Equation (5.5) is modified such that all local models now share a common set of eigenvectors, $\boldsymbol{V} \in \mathbb{R}^{2\times 2}$,

$$\boldsymbol{A}_i = \boldsymbol{V} \operatorname{diag}\underline{e}_i \boldsymbol{V}^{-1}, \quad i = 1 \ldots M . \tag{5.6}$$

The modelling results are shown in Figure 5.21.




(a) Fast muscle. 4 units.



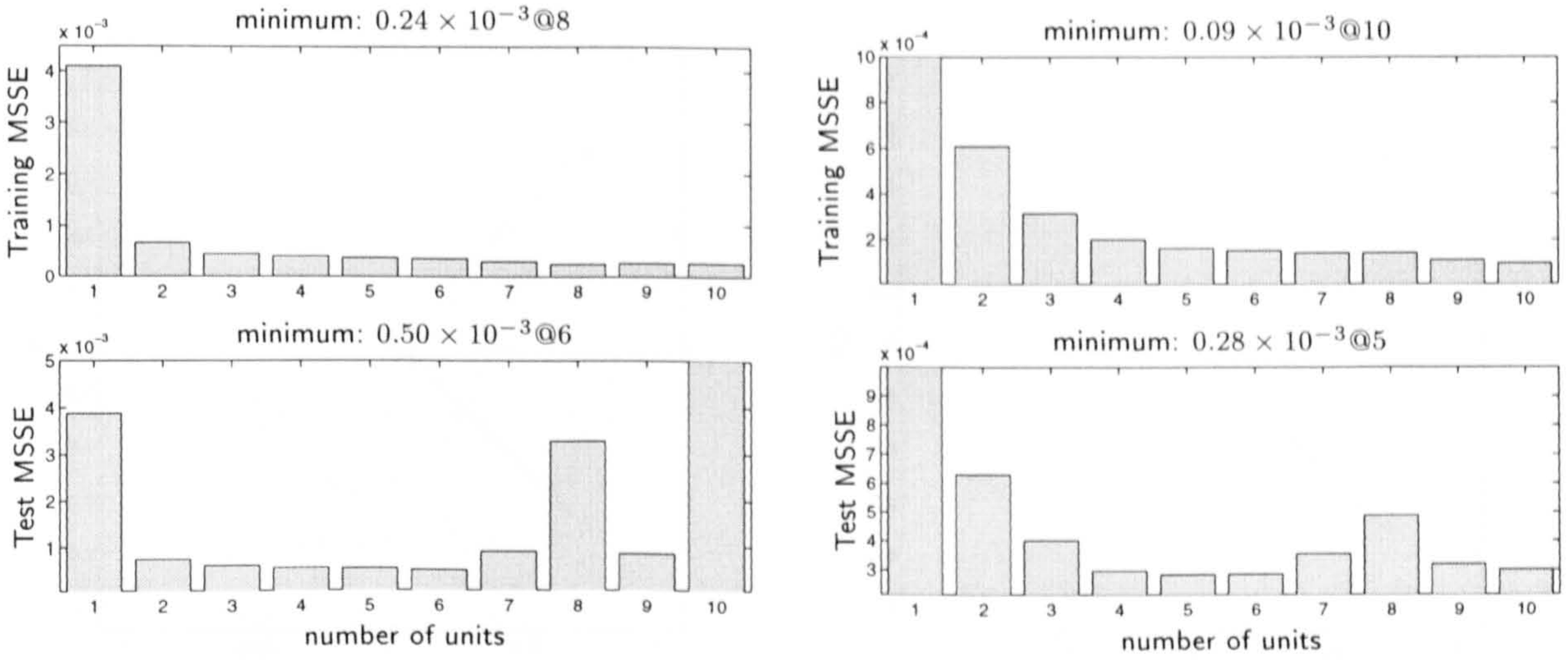

(b) Slow muscle. 5 units.

Figure 5.20: Local eigenvectors of the best state space models from Figure 5.19. "s" denotes the slow eigenvectors, the fast eigenvectors are marked by "f". The numbers indicate the number of the local model.

The results are very similar to both the results obtained with direct state space description (cf. Figure 5.18) and the results obtained with real eigensystems with local eigenvectors (cf. Figure 5.19). This indicates that the restriction to common eigenvectors and real eigensystems does not lead to deterioration of the model performance. The results for the fast muscle are even better than those obtained with LMNs with local eigenvectors. The optimal network size is 3 to 6 units for the fast muscle and 5 to 6 units for the slow muscle.

The eigenvector configurations of the best models (in terms of a minimal Test MSSE) from Figure 5.21 are shown in Figure 5.22.

**Discussion** It was found that LMN structures with local state space models perform significantly better than LMNs with local input–output models for both the fast and the slow muscle. Models with general local state space models, with state space models with real eigensystems, and with local state space models with real eigensystems and common eigenvectors were found to perform similarly. The reduction of the degrees of freedom introduced by restricting the local models does not lead to a deterioration of the performance. Thus, the limitation to real eigensystems and common eigenvectors which ensures stability of the global model provided that all local models are stable is adequate for the given application.




(a) Fast muscle.

(b) Slow muscle.

Figure 5.21: State space LMN with common eigenvectors with input scheduling. Large errors which exceed the axis range are truncated.

### Optimisation of the Validity Function Parameters

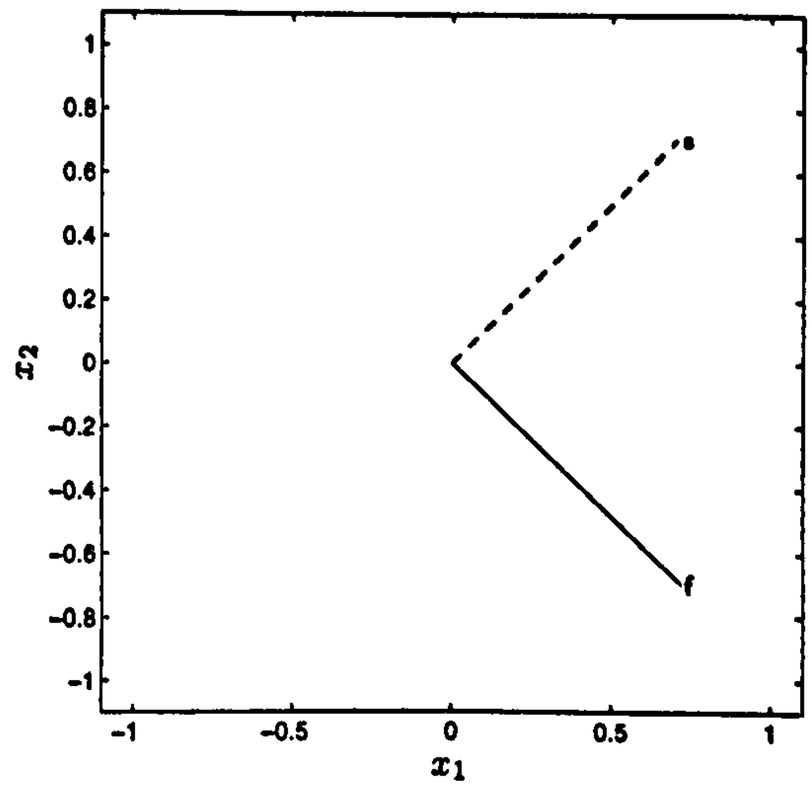
In the identification experiments carried out so far, we worked with LMN structures with uniformly distributed B–spline validity functions. The optimal number of units was found by comparing the performance of models of different size.

As outlined in Section 2.3.1, it is also desirable to adapt the shape of the validity functions to the complexity of the system. When working with B–splines, this can be easily achieved by optimising the location of the knots, i.e. by applying a parameter optimisation algorithm to the knot vector. After the knots have been changed, the parameters of the local models are updated, and the knot vector is then optimised again. This process is iterated until the network has converged.

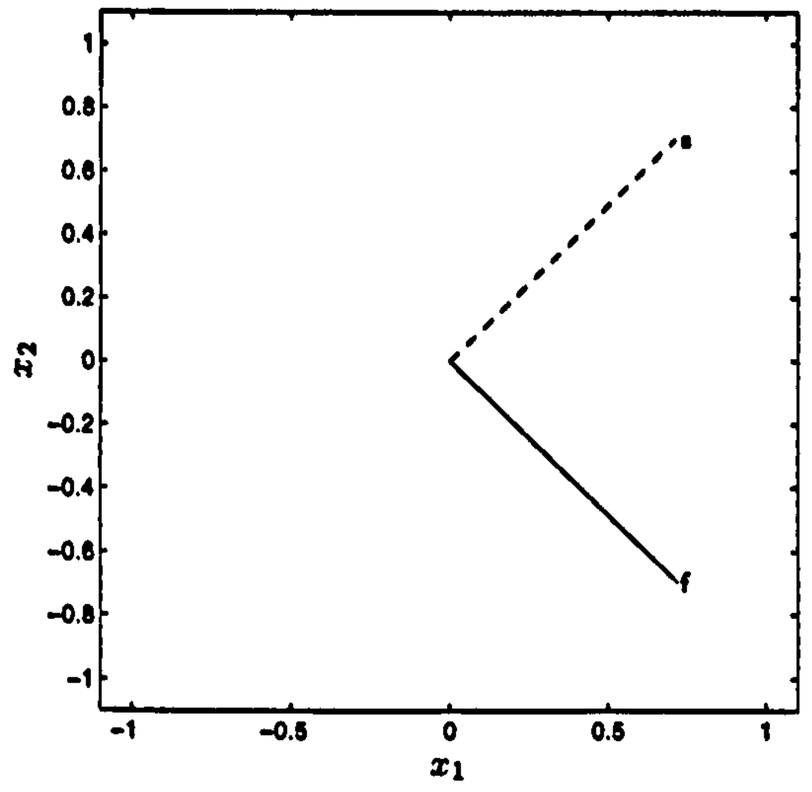
This structure optimisation algorithm is applied to the LMNs in state space form with common eigenvectors which have 6 units (fast muscle) and 5 units (slow muscle), cf. Figure 5.21. The shape of the uniform and of the optimised validity functions are shown together with the modelling errors in Figure 5.23.

The optimised shapes of the validity functions differ only slightly from the shapes of the uniform validity functions. For both muscles, the performance on the training set has improved. For the fast muscle, however, the performance on the test data deteriorates for the network with optimised validity functions. This indicates that over-fitting takes place. For the model of the slow muscle the test error is slightly lower for the optimised network.

Both the over-fitting for the model of the fast muscle and the improvement for the model

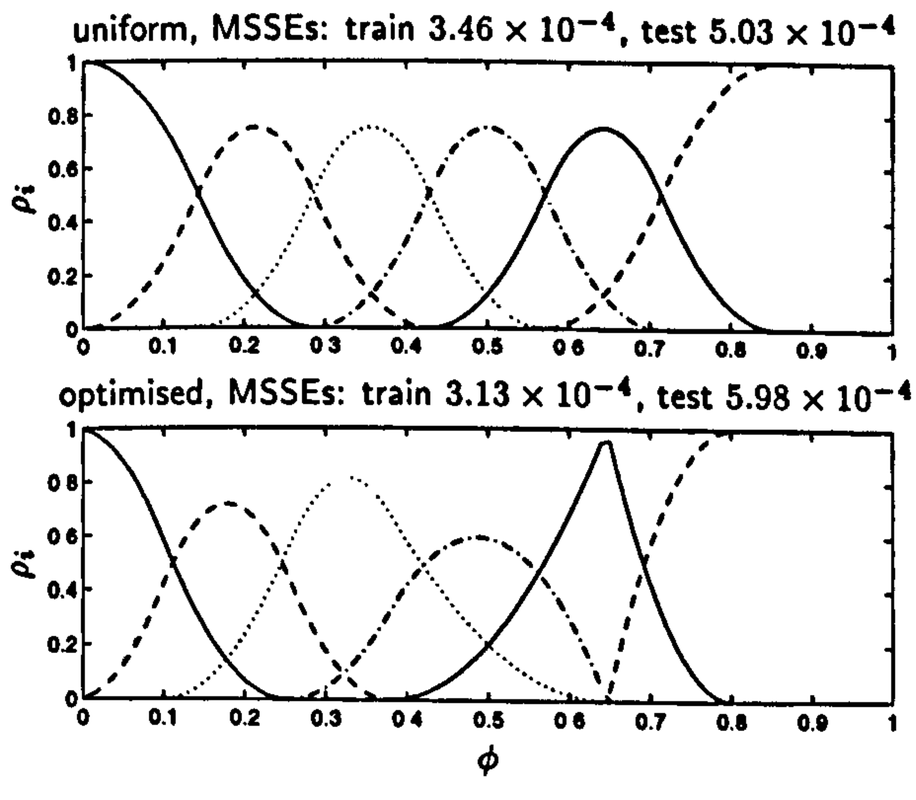



(a) Fast muscle. 6 units.

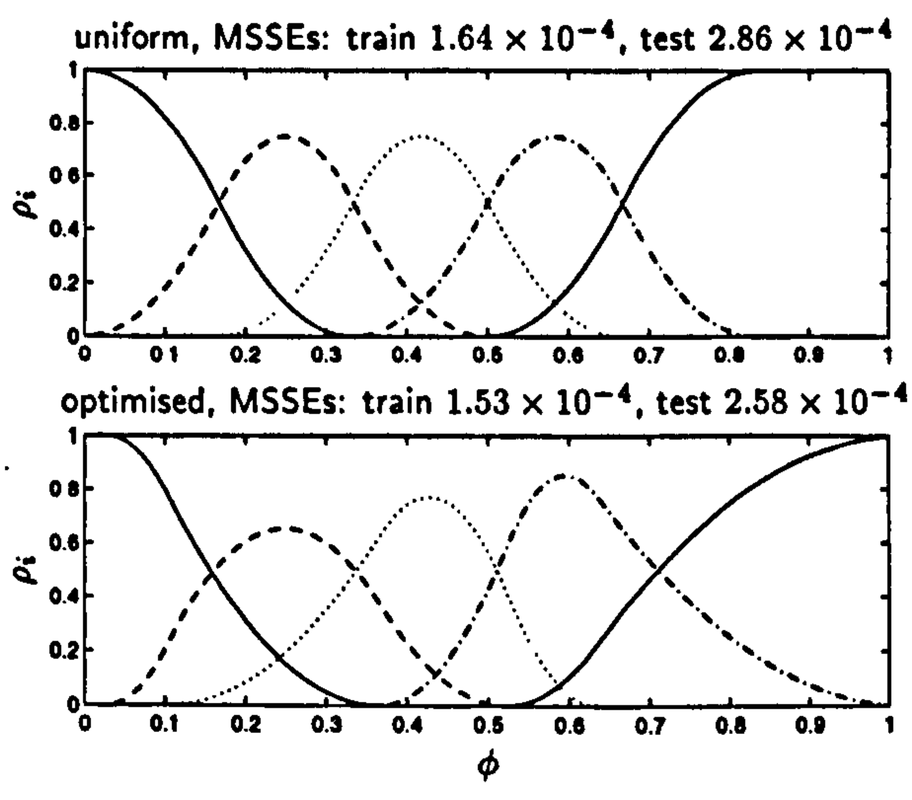



(b) Slow muscle. 5 units.

Figure 5.22: Common eigenvectors of the best state space models from Figure 5.21. "s" denotes the slow eigenvectors, the fast eigenvectors are marked by "f".




(a) Fast muscle, 6 units.




(b) Slow muscle, 5 units.

Figure 5.23: Shape of the optimised validity functions for the best models from Figure 5.21. $\rho_i$ denotes the activation of the $i$-th validity function.

of the slow muscle could be observed consistently when the shape of validity functions was optimised for networks of different sizes and with different local models.

Owing to the small effect of the optimisation of the shape of the validity functions, and the possible deterioration of the model performance on the test data, we choose to work with

models with uniformly distributed validity functions for the remaining parts of this chapter.

### 5.4.3 Analysis of the Model Performance and Properties

In the previous section, different model structures and parameter optimisation algorithms were compared in order to obtain the best models to describe the characteristics of the fast and of the slow muscle. As a criterion for the comparison of the different results, various measurements of the MSE which are defined at the beginning of Section 5.3 were used.

In this section, a more detailed analysis of the models will be carried out. The results presented will focus mainly on the best LMNs in state space form with real eigensystems with input scheduling, as shown in Figures 5.19 and 5.21. The modelling results for these LMNs are summarised in Table 5.2.

| Eigenvectors | Fast muscle | | | Slow muscle | | |
|---|---|---|---|---|---|---|
| | # units | MSSE ($\times 10^{-3}$) Training | Test | # units | MSSE ($\times 10^{-3}$) Training | Test |
| local | 4 | 0.34 | 0.54 | 5 | 0.17 | 0.28 |
| common | 6 | 0.35 | 0.50 | 5 | 0.16 | 0.28 |

Table 5.2: Modelling results of the LMNs in state space notation with real eigensystems, input scheduling.

It should be noted that the analysis techniques described below have also formed an integral part in the experiments described in the Sections 5.4.1 and 5.4.2.

We will first present model simulation results, before some properties of the models are analysed. In the final part of the section aspects of model modification and restriction are presented.

#### Model Performance

The performance of the best LMNs with local models in state space form with real eigensystems and common eigenvectors (cf. Figure 5.21) on some typical test data sets are shown for the fast muscle in Figure 5.24 and for the slow muscle in Figure 5.25. Note that the performance of the LMNs with local eigenvectors (cf. Figure 5.19) and the LMNs in direct state space form (cf. Figure 5.18) is very similar.

From the resulting plots it can be concluded that the LMNs model the behaviour of the real muscle with great accuracy for stimulation with randomly varying IPIs present in the experimental data.

The model performance can be further analysed by investigating the relationship between the stimulation frequency and the muscle output force. Long bursts (1000ms) of constant

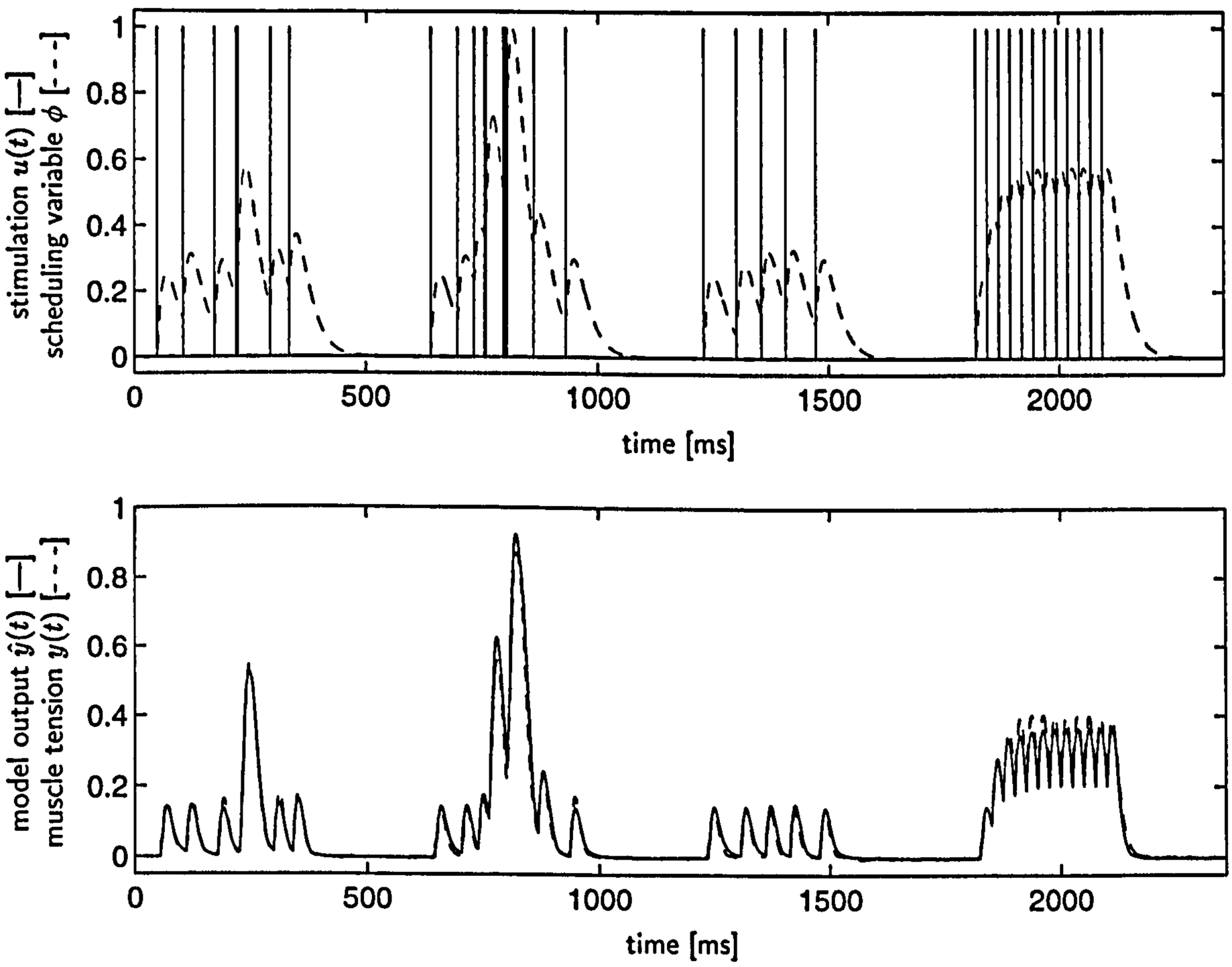



Figure 5.24: Fast muscle, LMN with local models in state space form with real eigensystems and common eigenvectors, 6 units: Comparison of the simulated model output (bottom, solid) and the output of the muscle (bottom, dashed) for four typical test data sets. The data sets are shown concatenated; the simulation is restarted after each set. The model output matches the muscle output for almost all operating conditions. A small model error is present in the first part of the 4th data set. The "catch-like" effect which is present in this muscle is modelled accurately (middle of 1st data set at 250ms, 4th and 5th pulse of the 2nd data set at 750ms).

frequency trains were applied to the muscle models, and the resulting model output was recorded. The "output force" was measured as the average model output after the initial rising portion of the response. The force–frequency curves obtained for LMNs with real eigensystems and local and common eigenvectors are shown in Figure 5.26.

The shapes of the force-frequency curves are as expected up to a frequency of approximately 100Hz: the model output increases as the stimulation frequency is increased. This relationship is non-linear, with some saturation at almost 100Hz. For frequencies above 100Hz it would be expected that the force saturates. However, for the fast muscle it continues to grow, whereas for the slow muscle some saturation can be observed. This indicates that the

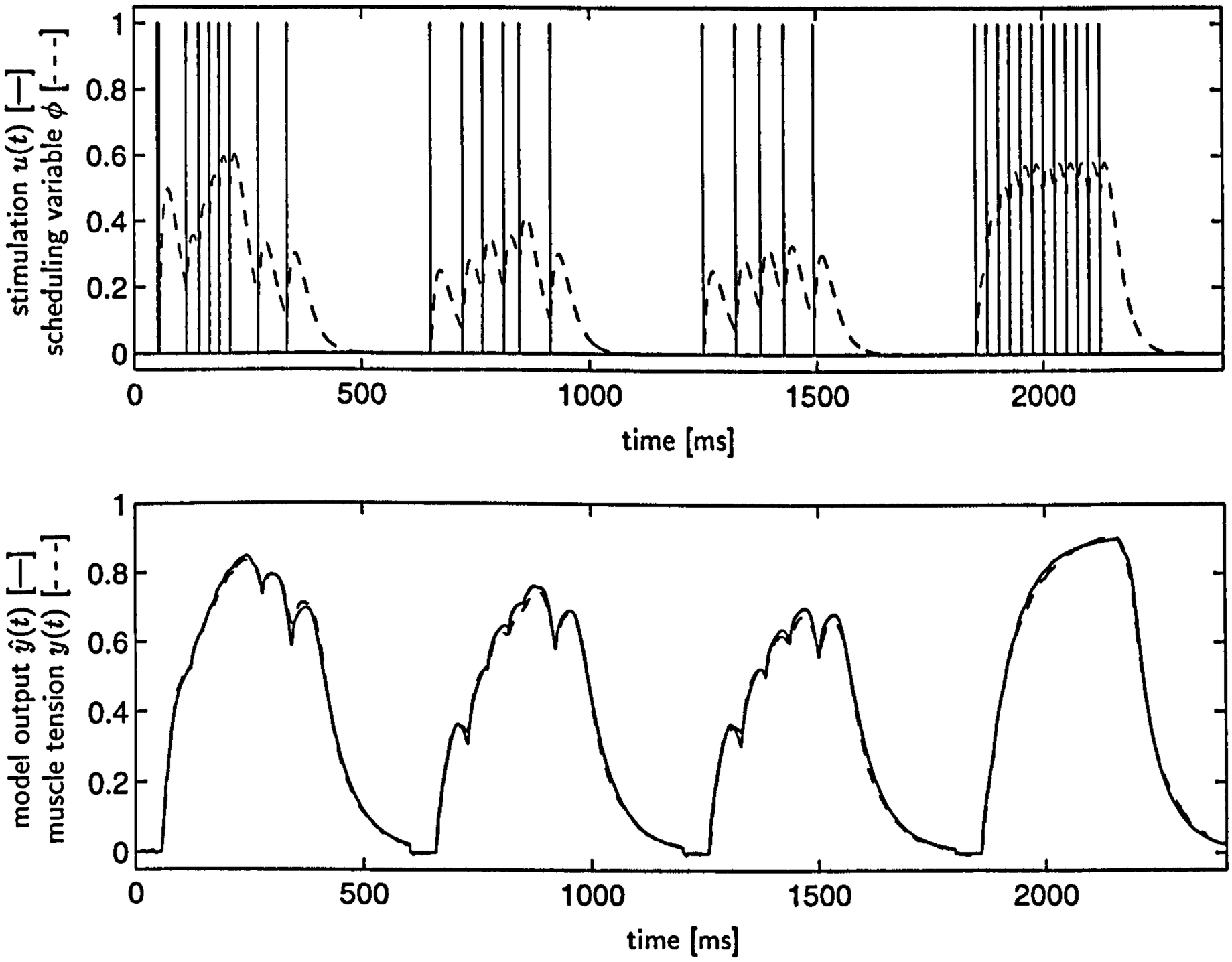
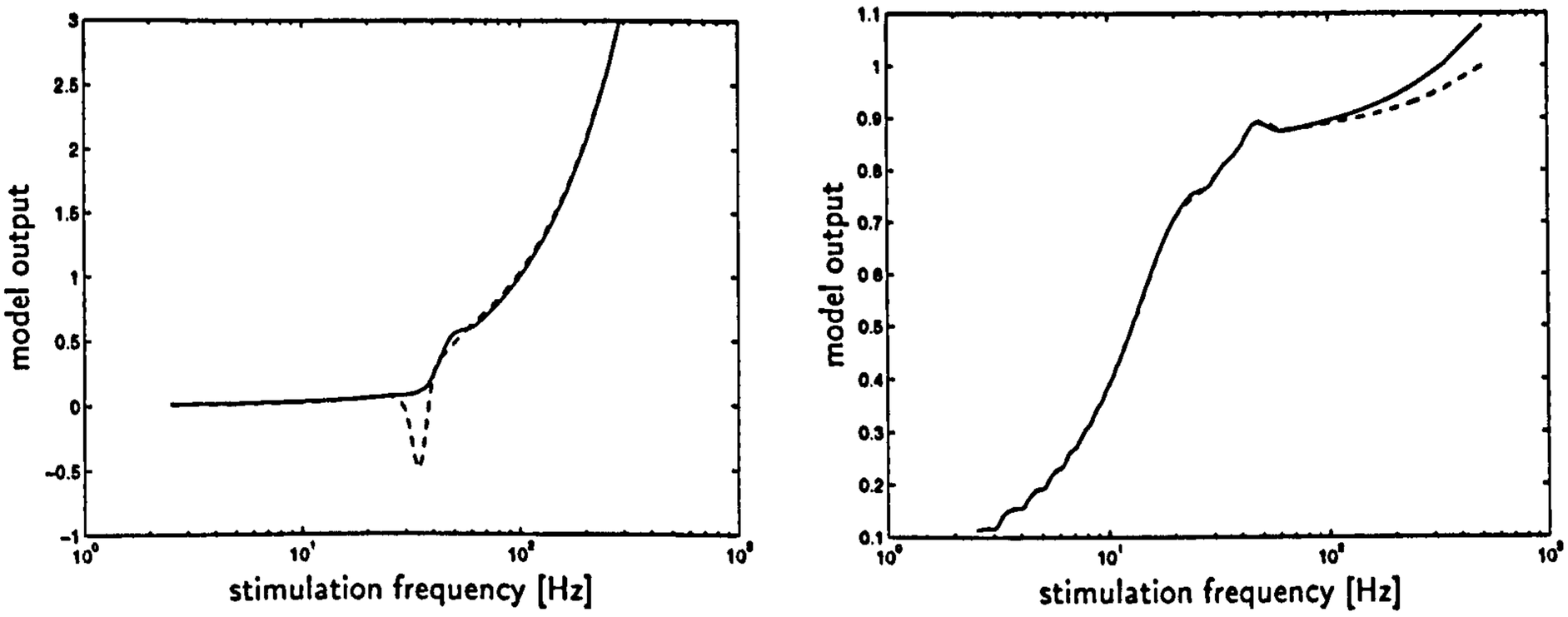



Figure 5.25: Slow muscle, LMN with local models in state space form with real eigensystems and common eigenvectors, 5 units: Comparison of the simulated model output (bottom, solid) and the output of the muscle (bottom, dashed) for four typical test data sets. The data sets are shown concatenated; the simulation is restarted after each set. The model output matches the muscle output for all operating conditions. Note that the "catch-like" effect is not present in this muscle.

model of the fast muscle does not include saturation for long bursts of impulses with a high stimulation frequency. Such stimulation patterns were not present in the data used to identify the model, where only short high frequency bursts could be observed.

The dashed line in Figure 5.26(a), which corresponds to the LMN with local eigenvectors, shows an unexpected negative bump at 30...40Hz. This indicates that the model does not perform well for long input bursts of this frequency range. Again, stimulation pulses with such a characteristic were not present in the data used to identify the models. The analysis of the properties of the models in the next section will provide more insight into the reason for this unexpected behaviour.




(a) Fast muscle.

(b) Slow muscle.

Figure 5.26: Force-frequency relationship for best LMNs with local state space models with real eigensystems with common eigenvectors (solid) and with local eigenvectors (dashed).

## Analysis of Model Properties

Aspects of the static analysis of the properties of a Local Model Network were discussed in Section 2.4.2. Applying the technique described there to the models obtained from the experiments on muscles will provide further insight into the properties of the modelling approach.

Before the analysis results are presented we will briefly discuss how the model properties can be interpreted with respect to the characteristics of the muscles.

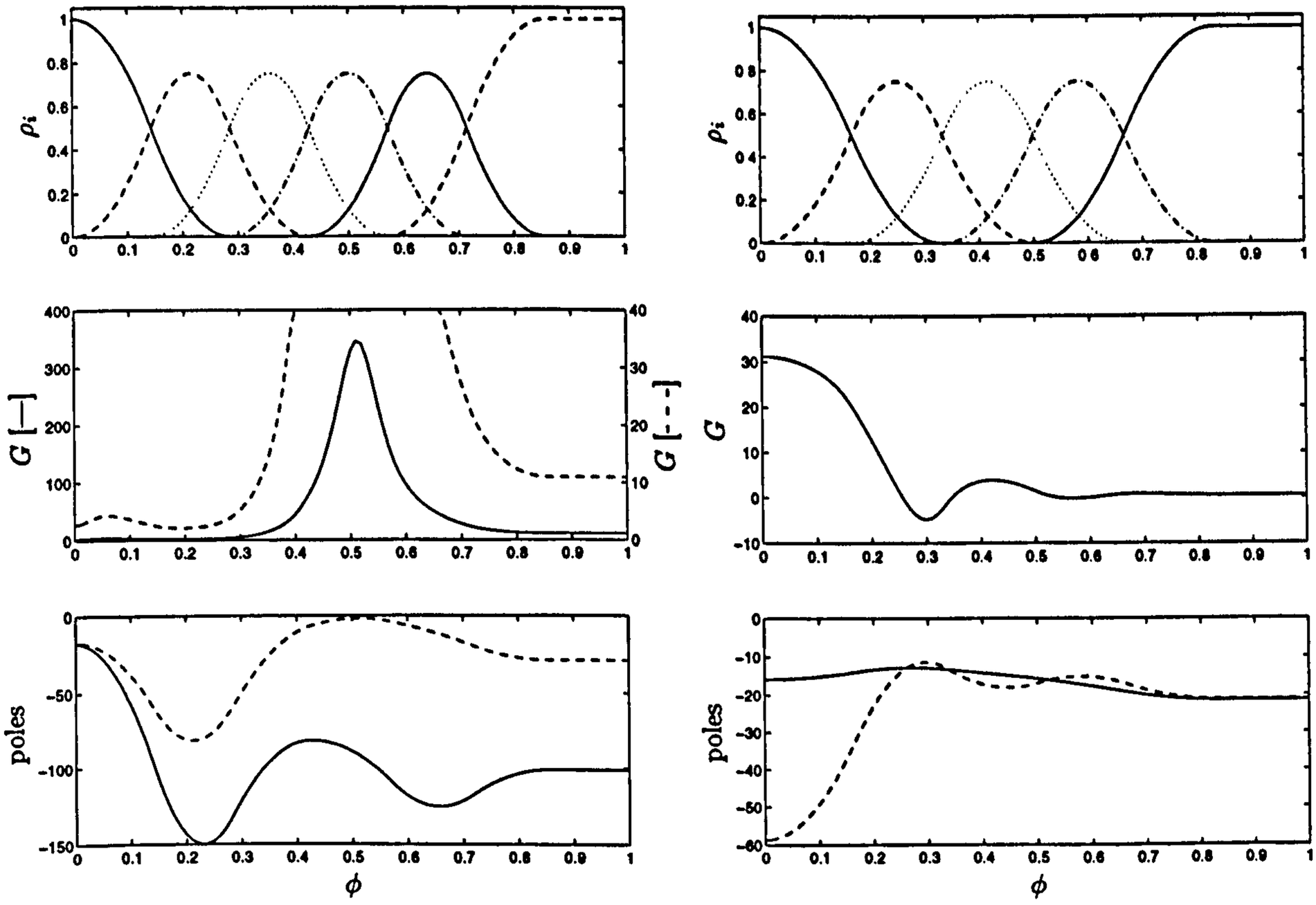
**Interpretation** The (steady-state) gain of the model is the steady state value of the model response to a step input. It is equivalent to the integral of the impulse response. As the inputs to the muscle model are always impulses, the latter interpretation is more appropriate for our application. As the model output is a prediction of the force produced by the muscle, the value of the gain for a certain activation can be interpreted as the force-time integral (FTI) per pulse for this activation. The relevance of the FTI for muscle stimulation is discussed in more detail in Chapter 6.

The location of the poles gives an indication of stability properties of the model: if the poles are located outside the region of stability (cf. Section D.2.2 for a definition of this region for models in delta operator notation) the model is unstable.

The absolute value of the poles is related to the time-constants of the model for the corresponding activation. The further away the poles are from the real axis (from zero), the smaller is the corresponding time-constant. Thus, poles with a large absolute value indicate a fast model response, whereas poles with a small absolute value correspond to slow modes

of the model.

**Results** The results of the static analysis for the best LMNs with local models in state space notation with real eigensystems and common eigenvectors are shown in Figure 5.27.




(a) Fast muscle. In the middle plot the solid line corresponds to the left y-axes and the dashed line relates to the right y-axes.

(b) Slow muscle.

Figure 5.27: Static analysis for LMN structures with local models in state space form with real eigensystems and common eigenvectors. The top plots show the activations of the validity functions, $\rho_i$. In the middle plots the steady state gains, $G$, of the interpolated models are depicted. In the bottom plots the values of the two poles of the interpolated models are shown.

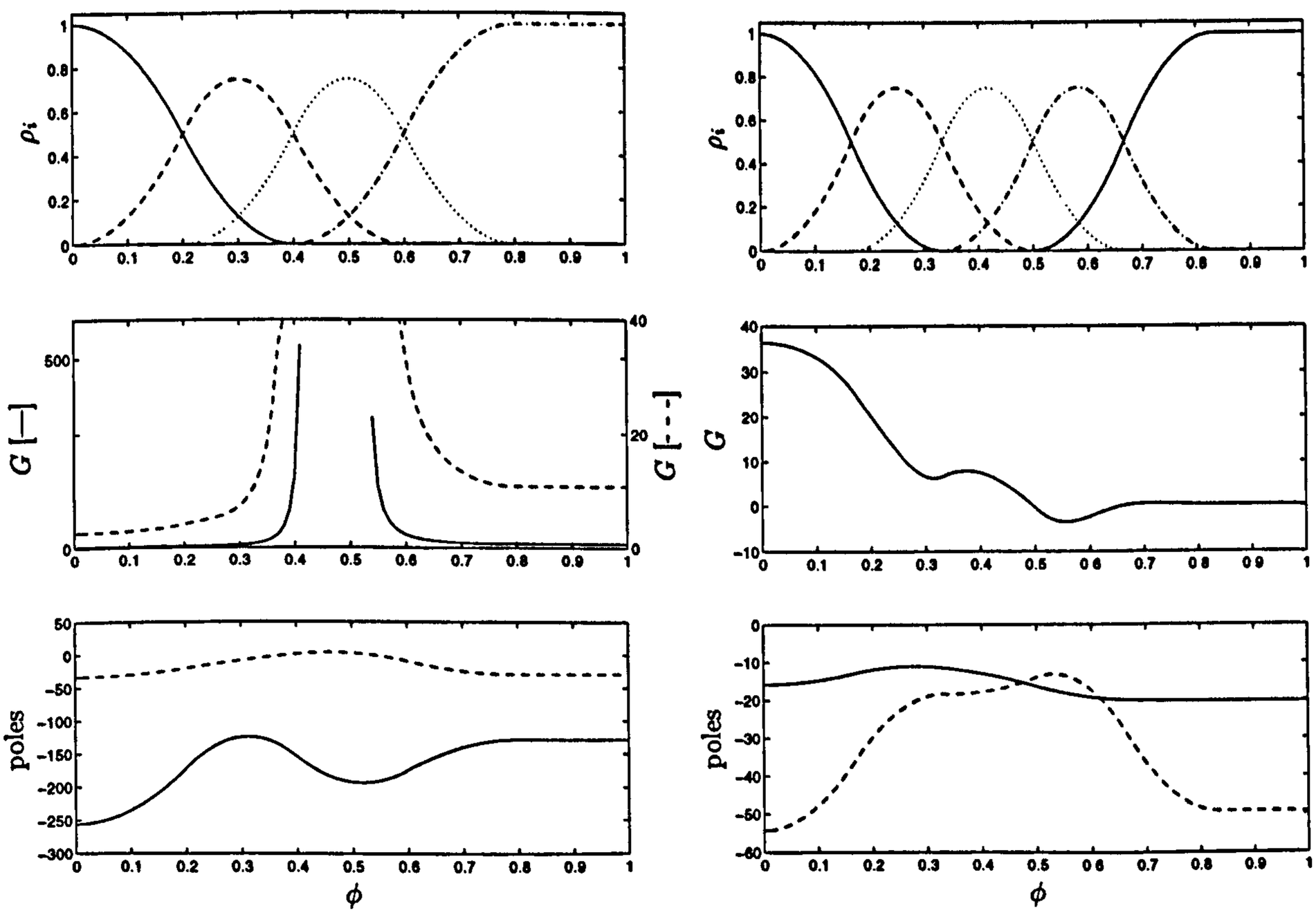
For both models the characteristics vary smoothly with the scheduling variable. This indicates that the models do not over-fit the data.

For the model of the fast muscle, the gain varies in a wide range, with a distinct maximum at $\phi \approx 0.5$. The gain for small activation is relatively small. For high activation the gain remains constant non-zero. Thus, saturation is not present for large activations. The poles vary in a wide range. A fast and a slow pole can be distinguished. The value of the slow pole

approaches zero for $\phi \approx 0.5$.

For the model of the slow muscle, the gain varies in a much smaller range than the gain of the model for the fast muscle. It has a distinct maximum for small activation, and quickly decreases as the activation becomes larger. The gain becomes slightly negative for $\phi \approx 0.3$ which indicates that for this activation, stimulation will decrease the model output. For large activation the gain is close to zero which corresponds to saturation. The poles are significantly slower than the poles of the model for the fast muscle. The slow pole changes only slightly, whereas the second pole becomes close to the slow pole for increased activation.

The results of the static analysis for the best LMNs with local models in state space notation with real eigensystems and local eigenvectors are shown in Figure 5.28.




(a) Fast muscle. In the middle plot the solid line corresponds to the left y-axes and the dashed line relates to the right y-axes.

(b) Slow muscle.

Figure 5.28: Static analysis for LMN structures with local models in state space form with real eigensystems and local eigenvectors. The top plots show the activations of the validity functions, $\rho_i$. In the middle plots the steady state gains, $G$, of the interpolated models are depicted. In the bottom plots the values of the two poles of the interpolated models are shown.

The analysis results are generally very similar to the characteristics shown in Figure 5.27. A few important differences should, however, be noted.

For the model of the fast muscle, the gain not only increases in the area around $\phi \approx 0.5$ but the model is unstable for this activation, as the slow pole becomes slightly positive. The negative bump in the force-frequency curve shown in Figure 5.26(a) can be related to this characteristic. The instability is certainly not a property of the real system. We will therefore consider this characteristic in more detail below.

For the model of the fast muscle, the shape of the change of the fast pole is different for low activation levels from that shown in Figure 5.27(a). This can be explained by the different location of the corresponding eigenvectors. Referring to Figure 5.20(a) on page 75 confirms that the location of the eigenvectors of the first local model of the LMN with local eigenvectors differs significantly from the location of the common eigenvectors shown in Figure 5.22(a) on page 77.

Similarly, the difference between the pole characteristics of the model for the slow muscle for large activations and the pole characteristics shown in Figure 5.27(b) can be attributed to the different eigenvector configurations. Additionally it should be noted that the gain of the model is small for large activations ($\phi > 0.7$). The location of the poles for that region of activation will therefore not influence the global model characteristics strongly.

**Discussion** The limited and smooth variation of the gain for the models of the slow muscle indicate that the force-time integral (FTI) does not change significantly depending on the activation. We expect such behaviour for this muscle as it does not show the "catch-like" effect. The fact that the gain is almost zero for activations above $\phi \approx 0.7$ corresponds to a decrease of the FTI per pulse for intensive stimulation. This will be verified in the experiments in Chapter 6.

The large variation of the gain for the models of the fast muscle indicates that, for this muscle, the FTI per pulse depends strongly on the activation. This is expected as the fast muscle shows the "catch-like" effect. The large value of the gain for $\phi \approx 0.5$ for the model with common eigenvectors, and the instability of the model with local eigenvectors in the same range, can be related to the initial response of the model to two closely spaced input pulses (the response to a "doublet"). In Figure 5.29, a doublet stimulation pattern is depicted together with the obtained scheduling variable. It shows that the maximal value reached by the scheduling variable for this input pattern is $\phi \approx 0.5$.

Comparing the location of the poles, it can be observed that the poles of the models of the slow muscle are significantly slower than the poles of the models of the fast muscle. This corresponds to the known characteristics of the muscles.

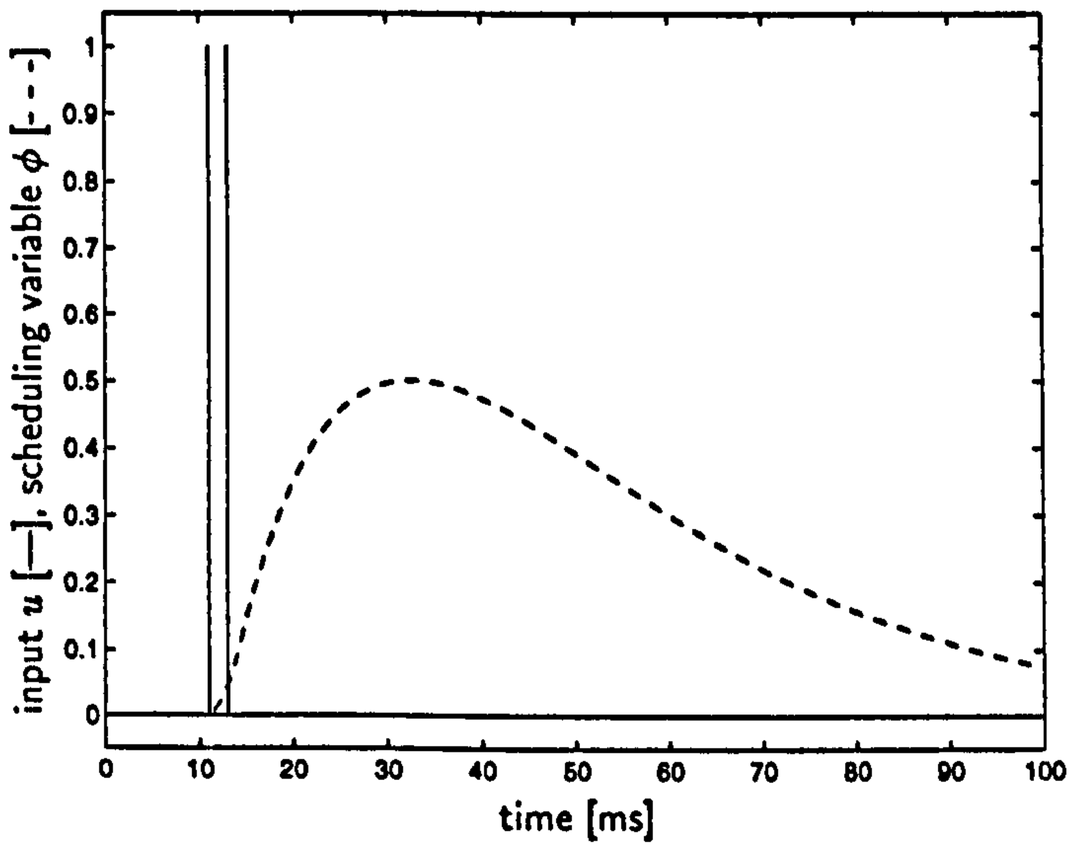



Figure 5.29: Scheduling variable obtained from a doublet input with IPI=2ms.

### Modification of a Model

In the previous section it was found that the LMN with real eigensystems and local eigenvectors which was identified for the fast muscle is unstable for a region of activation around $\phi \approx 0.5$, cf. Figure 5.28(a). The eigenvalues of the local models of this LMN are shown in Table 5.3. Note that the slow eigenvalue of the 3rd local model is slightly positive and therefore unstable.

| # unit | eigenvalues | |
|---|---|---|
| 1 | −256.27 | −33.24 |
| 2 | −87.07 | −7.71 |
| 3 | −220.68 | 9.96 |
| 4 | −128.48 | −29.61 |

Table 5.3: Eigenvalues of the local models of the LMN in state space form with real eigensystems and local eigenvectors for the fast muscle.

The unstable 3rd local model does not lead to a deterioration of the results for the experimental data used in Section 5.4.2 where the simulation errors obtained with this LMN are very small, cf. Figure 5.19(a). In these data sets, the stimulation varies randomly in such a way that the muscle is never in the same state of activation for a longer period of time. As the unstable pole is relatively slow, it has no effect on the output for these data.

When the model is, however, stimulated with longer bursts of constant stimulation frequency which cause a single local model to be active for a longer period of time, the unstable

pole influences the model behaviour. An example of this are the experiments to obtain the force-frequency relationship, where the model was stimulated with constant frequency bursts of 1000ms duration. As depicted in Figure 5.26(a), the LMN with local eigenvalues shows a negative force bump. We will show that this can be attributed to the unstable local model.

In this section we aim to modify the unstable local model such that the model is stabilised, without deterioration of the overall model performance. The results presented are included in (Shorten *et al.* 1998).

**Modification of the Local Model** The unstable local model is modified in such a way that its positive eigenvalue is replaced by a slow, but negative eigenvalue. It was found that the exact value of this changed eigenvalue does not have a significant influence on the performance of the model on the experimental data as long as its absolute value is small. We therefore choose to replace the unstable eigenvalue with $-1$. To ensure that the local dynamics are similar to those obtained with the unstable local model, the location of the operating point corresponding to this local model needs to be readjusted. As this operating point is contained in the bias terms $d_3^x$ and $d_3^y$ (cf. equation (A.7) on page 120), these terms need to be re-estimated using the same optimisation technique as for the original identification of the LMN parameters. Note that only the unstable eigenvalue and the bias terms of the corresponding local model are modified.

**Results and Analysis** The Training and Test MSSE results for the original and for the modified LMNs are summarised in Table 5.4. The modelling performance of two LMNs for a typical set from the experimental data is shown in Figure 5.30.

| LMN | MSSE ($\times 10^{-3}$) | |
|---|---|---|
| | Training | Test |
| original | 0.34 | 0.54 |
| modified | 0.39 | 0.54 |

Table 5.4: Mean squared error results for the original and the modified LMN on the experimental data.

The two models perform very similarly. The MSE error results on training and test data sets are almost identical. Thus, the modification of the local model does not lead to a deterioration of the model performance with the experimental data. As the 3rd local model is only active for a short period of time, the slow time-constants of the modified pole do not influence the model output significantly. This becomes even more obvious in the phase trajectories of the two LMNs which are shown in Figure 5.31. The trajectories of the two models are almost identical.

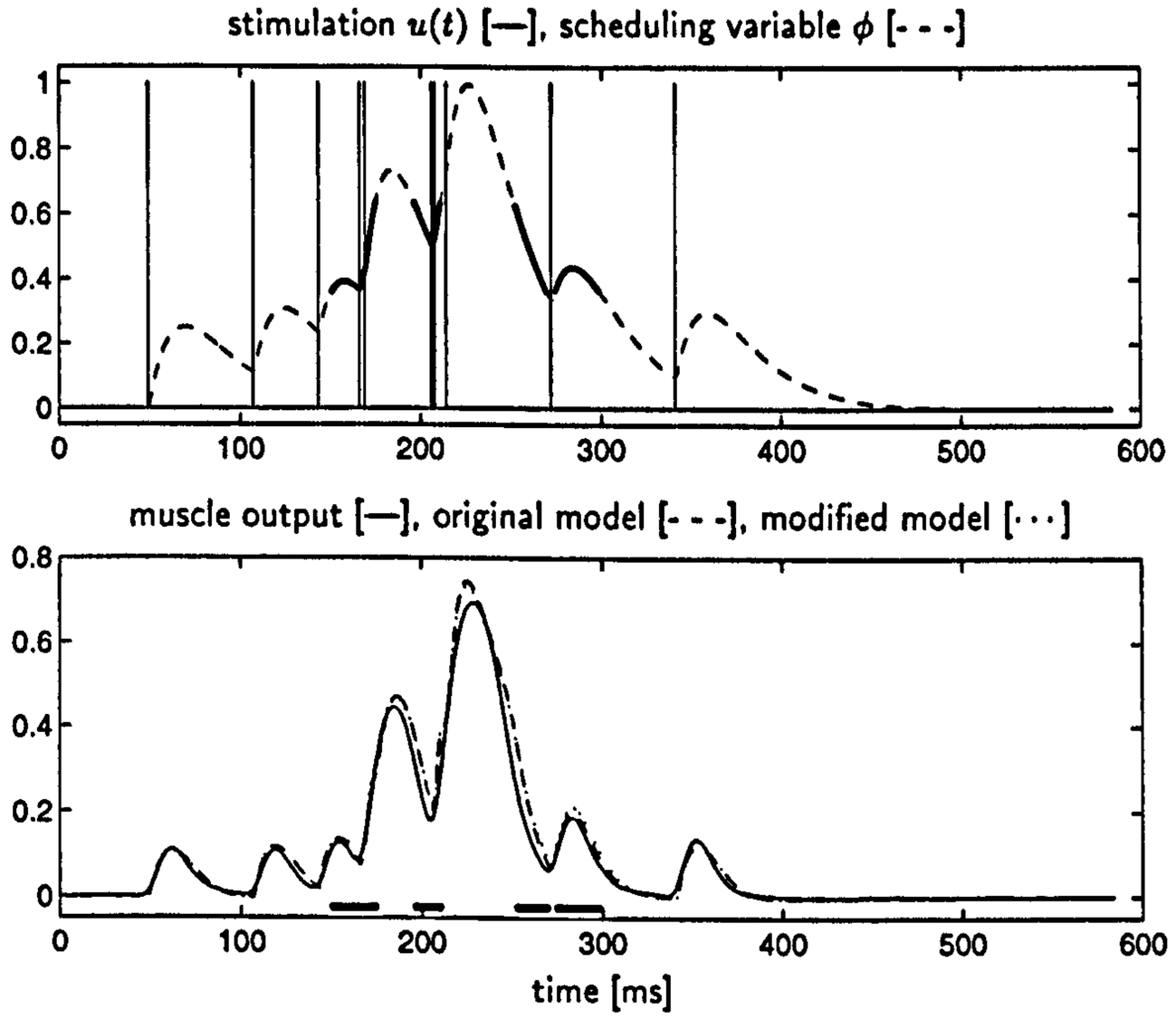



Figure 5.30: Experimental data and model responses over time. The bold lines indicate that the third local model is active ($\rho_3 > 0.3$).

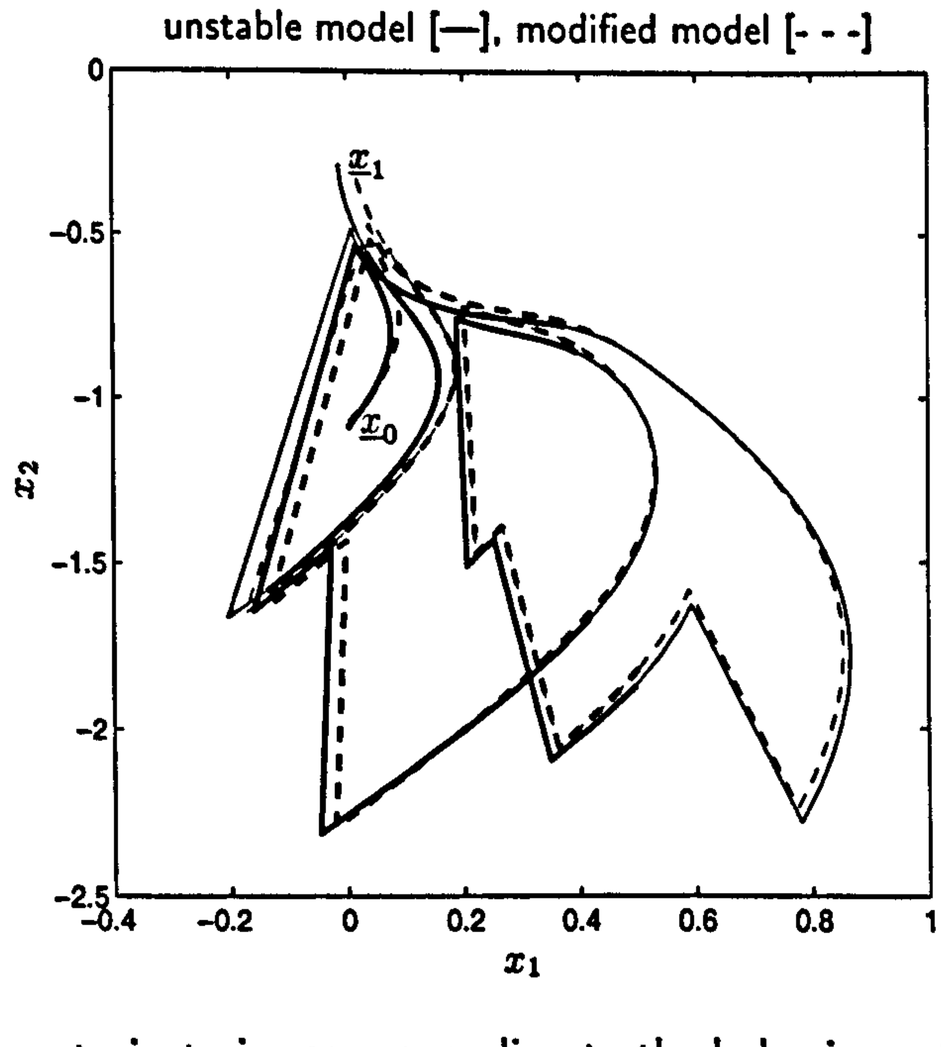
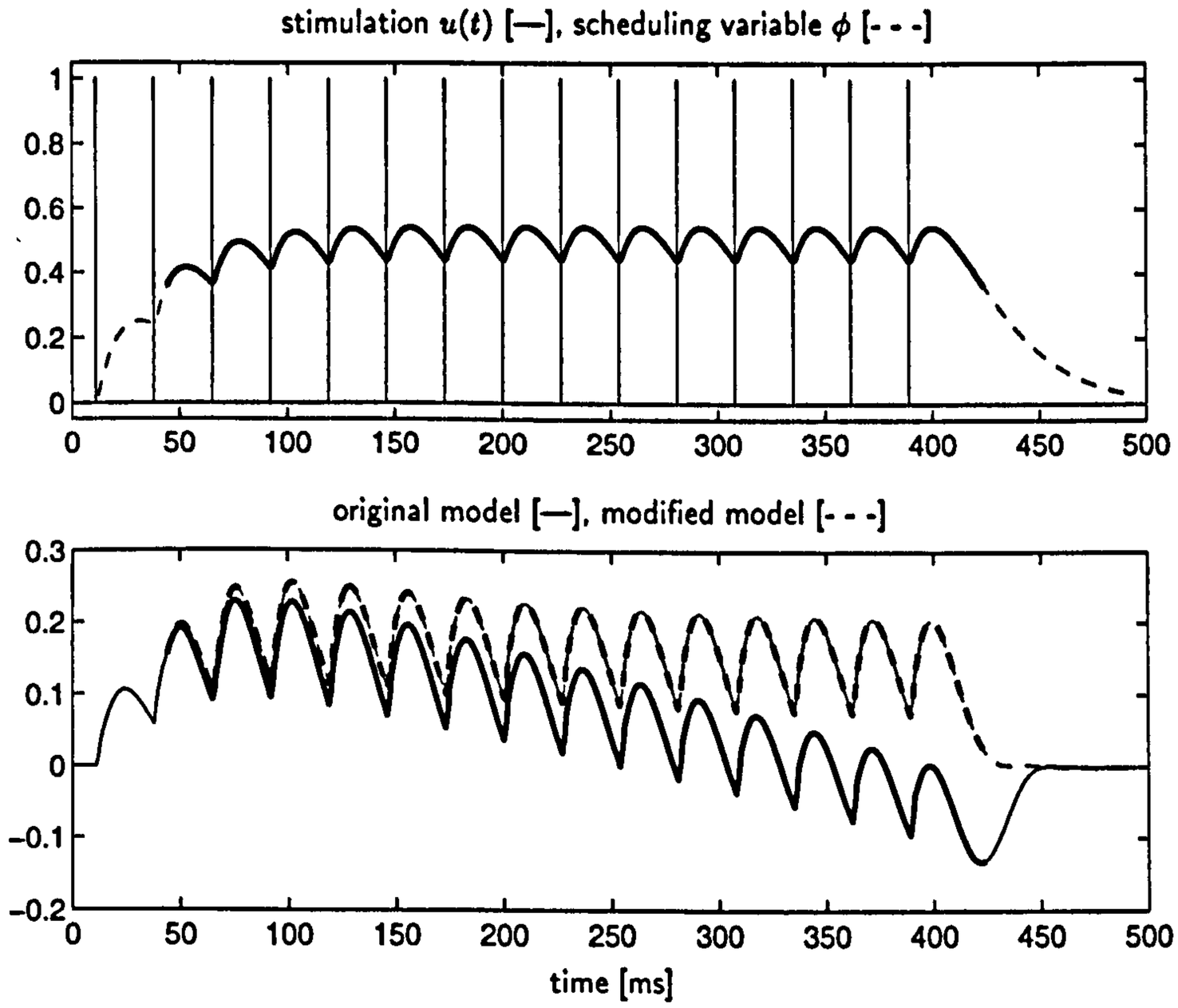
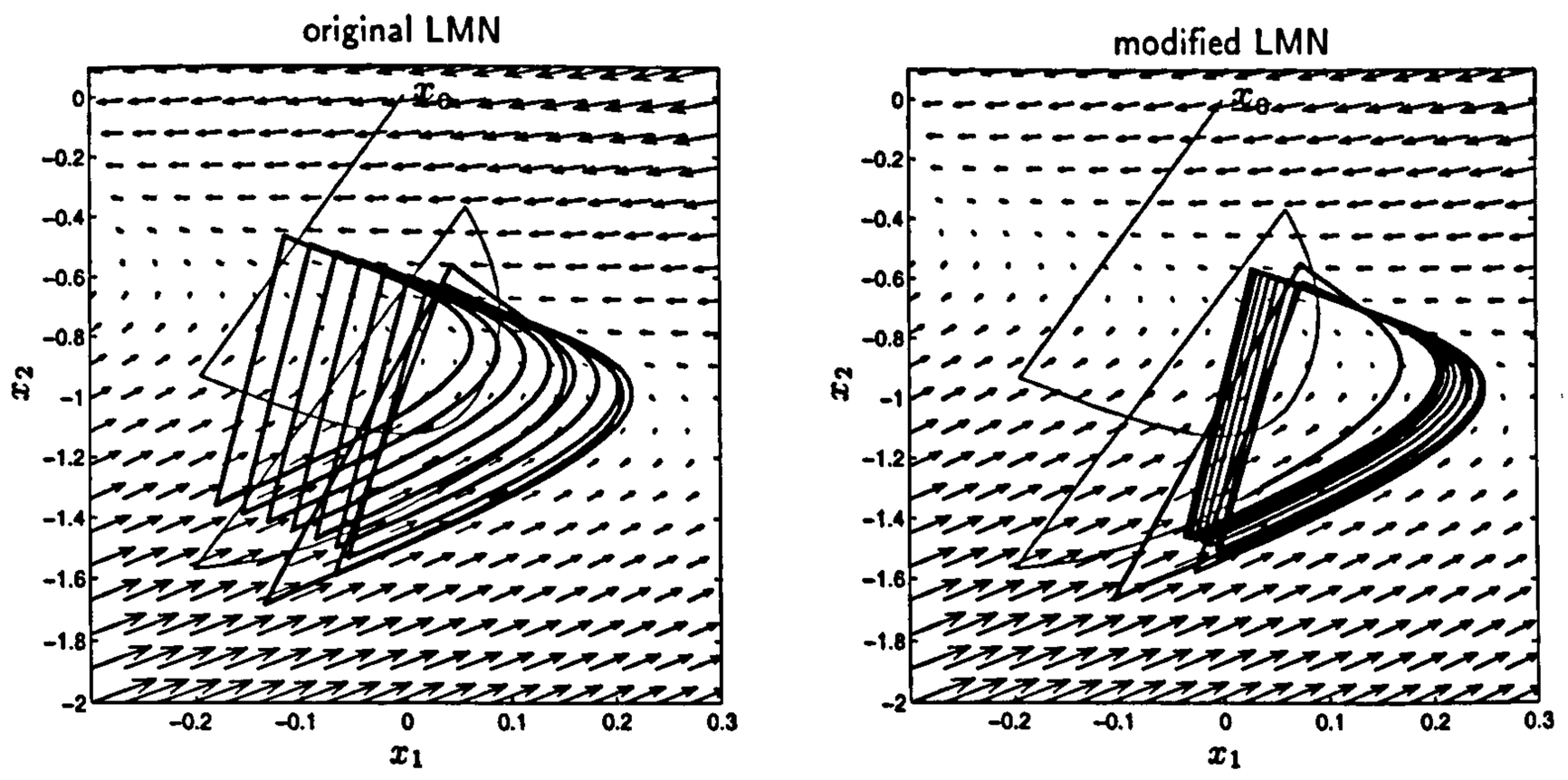



Figure 5.31: Phase plane trajectories corresponding to the behaviour of the unstable and the modified LMNs shown in Figure 5.30 from 150 to 310ms. $\underline{x}_0$ marks the beginning, $\underline{x}_1$ the end of the trajectories. The sudden vertical jumps are due to the pulse-like input $u(t)$. The bold lines indicate that the third local model is active ($\rho_3 > 0.3$).




(a) Responses in time.

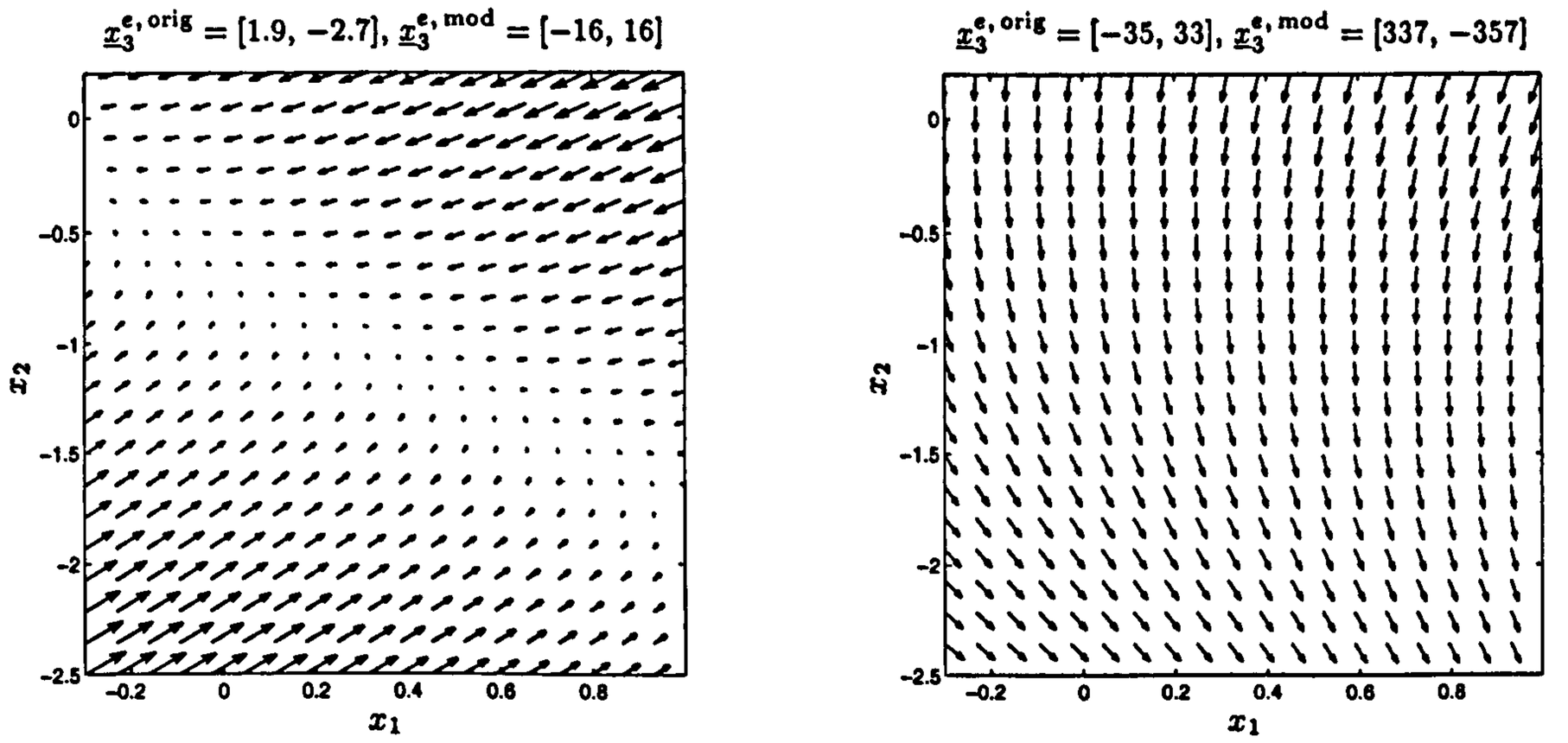



(b) Phase planes of the autonomous system for the 3rd local models, and state trajectories of the LMNs. The trajectories are shown from 1 to 300ms.

Figure 5.32: Time responses of the LMNs, and phase planes and state trajectories of the unstable and the stable local model for an input stimulation with a constant pulse frequency of 37Hz.

The responses of the original and of the modified LMN to stimulation with a constant frequency burst of 37Hz are shown in Figure 5.32(a). The stimulation frequency has been chosen to demonstrate the effect of the unstable local model in the original LMN by driving the model to a state where the 3rd local model is constantly activated. Note that a similar stimulation pattern is not present in the data used to identify the models. The initial responses of both LMNs are identical. After the 3rd local model was active for a considerable time, the output of the original LMN starts to decrease, and becomes negative unbounded as the simulation continues (not shown). The modified LMN behaves as expected from the real muscle, i.e., it reaches a constant trajectory and remains there.

The differences in the phase trajectories of the 3rd local models of the two LMNs are shown in Figure 5.32(b). The trajectory of the unstable local model of the original LMN drifts away towards minus infinity, whereas the trajectory of the stable local model of the modified LMN reaches a limit cycle.




(a) Autonomous model. $\underline{x}_3^e = -A_3^{-1}\,\underline{d}_3^x$.

(b) Input $u = 1$ applied. $\underline{x}_3^e = -A_3^{-1}\,(\underline{b}_3 + \underline{d}_3^x)$.

Figure 5.33: Phase planes of the 3rd local models of the original and the modified LMN, without and with input applied. The phase planes of the two models are almost identical.

In Figure 5.33, the phase planes of the 3rd local models are shown for the autonomous case and when an input $u = 1$ is applied, and the corresponding values of the equilibria are given. The phase planes of the two local models are almost indistinguishable. However, they differ substantially in the location of the equilibrium points which define whether the models are stable or unstable. The slow time constants associated with this difference of stability only become significant when the local model is activated for a longer period of time.

**Discussion** Local instability of an LMN model for the fast muscle was investigated in the previous section by analysing the properties of the model structure. Such instability represents a basic mismatch between the model properties and the properties of the real muscle. This mismatch could not be found by analysing the model performance with the experimental data. It was, however, shown that the instability leads to unexpected model behaviour for stimulation with long bursts of constant frequency trains.

A method to modify the model locally in such a way that the global model matches the properties of the real system more closely was introduced. The LMN was stabilised by modifying the parameters of the unstable local model. The global model performance did not deteriorate after the modification. It was shown that the modified LMN performs as expected from the real muscle for stimulation patterns of long bursts with constant frequency.

The example shows that analysis of the properties of an LMN can give useful insight concerning the behaviour of the model outside the operating regions covered by the experimental data. Restricting the properties of the local models by using *a priori* knowledge about the system (i.e., the knowledge that the system is stable) can improve the model performance in operating regions not covered by data.

It was found that the experimental data can be approximated locally by two models with completely different stability properties to the same degree of accuracy. Thus one has to be careful when relating local properties of the model to global model characteristics. The phase plots in Figures 5.32(b) and 5.33 show that although two local models have very similar characteristics locally inside the operating region, their properties differ substantially outside this region as a result of the different location of the equilibrium point.

## 5.5 Discussion

In this chapter, a novel approach to modelling of electrically stimulated muscle under conditions of isometric contraction was presented. The model is non-linear and its structure is based on a network of locally valid linear models which are blended together by a scheduler. The model accounts for non-linear effects due to variations of the stimulation frequency, such as the "catch-like" effect. It was shown that this modelling technique is suitable for modelling the contraction of muscles with very different characteristics, such as muscle with a majority of fast motor units and muscle with mainly slow motor units.

Data from experiments with rabbit tibialis anterior muscles were used. The muscles were stimulated with supramaximal impulses, i.e. all motor units were activated by every stimulation pulse. Thus, effects due to recruitment variations were not present. The muscle activation was varied by changing the IPI which leads to non-linear effects due to changing stimulation frequency.

The modelling approach uses only input–output data of the muscle from simple standard experiments which are aimed at exciting all dynamic modes of the system. No explicit knowledge of the physiological processes of muscle contraction was used. However, the modelling technique differs from a pure black-box approach in that *a priori* knowledge about the expected muscle behaviour was used for the selection of the scheduling variable and for the analytical validation of the models obtained. Thus, the approach described here could be termed a "grey-box" technique.

### 5.5.1 Modelling Technique

An initial analysis of the experimental data and identification experiments with linear model structures gave guidelines for the selection of the optimal model structure. This includes the dynamic order, the time delay, the sampling period, and the parameter identification technique. Simulation results obtained with linear model structures confirmed that the muscle characteristics are significantly non-linear and that a simple linear approach does not describe the muscle behaviour correctly for varying operating conditions.

Local Model Network structures were therefore used as a non-linear modelling technique. Based on the results of the linear experiments, various approaches for the identification of the model parameters, the choice of the scheduling vector, and the number and structure of the local models of the network were evaluated until optimal structures were found. *A priori* knowledge was used for the selection of the scheduling variable, and for selecting real eigensystems for the local models in state space notation.

Experiments with single linear models and with LMNs with local models in input-output notation showed that the sampling period needs to be increased when the parameters are identified in the shift operator domain. The original sampling period of $T_s = 1\text{ms}$ could be used when the parameters were optimised in the the delta operator domain. Parameter estimation in the delta domain, although computationally more expensive owing to the shorter sampling period, was found to give the best results.

Local Model Networks with output scheduling were found to perform well with data from the fast muscle, see also (Gollee *et al.* 1994, Gollee *et al.* 1997, Gollee and Hunt 1997). This scheduling approach failed, however, for the slow muscle. Using pre-processed input pulses for scheduling was found to yield very good modelling results for both the fast and the slow muscle, and is therefore thought to be a scheduling technique generally applicable for modelling of muscle contraction under isometric conditions with supramaximal stimulation.

LMNs with local models in input-output and in state space notation were compared. The state space LMNs were found to outperform the LMNs in input-output notation, owing to the fact that the former have a larger number of free parameters than the latter.

The local state space descriptions were then restricted to real eigensystems. Models with local and with common eigenvectors were compared. They were found to perform equally

well on the experimental data.

Optimising the shape of the validity functions did not improve the model performance significantly. For the models of the fast muscle it even led to an increase of the test error. Networks with uniform validity functions were therefore used.

### 5.5.2 Model Analysis

The optimal Local Model Network structures were analysed and validated both using data and analytically. Validation based on data similar to those used to identify the model parameters showed that the models perform well for all operating conditions represented in the data. In particular, it could be observed that the model of the fast muscle can describe the "catch-like" effect which is one of the main non-linear characteristics of this muscle.

In order to evaluate whether the models can predict the force–frequency characteristics correctly, pulse trains with constant IPIs of different value were applied. The force-frequency curves were estimated correctly by the models for small and medium stimulation frequencies. For high stimulation frequencies, the models of the fast muscle did not show saturation as expected from the properties of the real muscle. Thus, the models could not generalise correctly from the kind of data used in the identification experiments to data which contain long trains of constant stimulation frequencies.

Analysis of the model properties showed that the characteristics of the models can be related to known properties of the real muscles. It explains for the failure of some models to predict the force-frequency curve correctly.

The analysis of the model properties showed that the LMN with real eigensystems and local eigenvectors which was identified for the fast muscle is unstable for a certain region of activation. Owing to the small time-constant associated with this instability, it did not degrade the modelling performance with the experimental data. However, it could be shown that the LMN becomes unstable when it is stimulated with a long input pulse train of constant frequency which causes the unstable local model to be active for a longer period of time. Using the *a priori* knowledge that muscle is a stable system, a simple technique was developed to modify the unstable local model in such a way that the overall LMN performs well on both the experimental data and pulse trains with constant IPI. The properties of the original and of the modified LMN were analysed for various stimulation patterns.

### 5.5.3 Conclusions

Local Model Networks provide a form of model structure which is well suited to the modelling of isometric contraction of electrically stimulated muscle. This is supported by the fact that linear time-varying systems have been found to be suitable for describing muscle contraction under varying conditions (Bobet *et al.* 1993), as LMNs with linear local models are time-varying linear systems. The technique to change the model parameters depending on the

muscle activation using a scheduling variable obtained from the model inputs can be used for muscles with very different characteristics.

The identification of the parameters of the Local Model Networks is very simple, and is based only on input–output data obtained from the muscle during standard experiments. As discussed in Chapter 3, the model structure is controller orientated. An application of local control techniques which is based on the muscle model introduced here will be described in Section 6.5. The simplicity of the model from the system theoretical point of view is a great advantage compared to more complex muscle models which are based on physiological properties.

The Local Model Networks can be implemented in a relatively simple way: the validity functions can be stored in form of a lookup table. The models can be evaluated in a discrete-time simulation which can be easily implemented in digital computing hardware. In contrast to various models which are based on physiological properties of muscle (Dorgan and O'Malley 1997, Riener *et al.* 1996), no numerical solution of continuous differential equations is necessary. The system can therefore easily be simulated in real-time[5].

A potential disadvantage of the modelling technique described here is that knowledge of the physiological properties of muscle is not used. All information necessary to describe the characteristics of the muscle needs to be presented in the experimental input–output data. It is therefore necessary to ensure that all operating conditions of interest for the intended application are represented in the data. The limitations of the modelling technique become obvious for the models of the fast muscle which cannot predict the saturation present in real muscle for stimulation with long trains of constant high frequency and which are, in some cases, unstable for certain stimulation conditions. One way to over-come this limitation would be to include input stimulation of different constant frequencies in the experimental data. The problem to obtain sufficient information for the identification of the model parameter from experimental input–output data will become more important for more general stimulation conditions. The experiments to obtain the data need then to be planned carefully to ensure that all relevant operating conditions are covered by data.

### Summary of the Modelling Approach

The description of the results in this chapter is very comprehensive and this may obstruct the basic steps of the Local Model Network approach when used in practice. We will therefore briefly summarise some guidelines for the modelling of muscle contraction with the approach described here.

[5]In the implementation used in this work which is based on C-MEX files running under MATLAB, the Local Model Networks of the muscles can be simulated in approximately three times real-time on a SPARC 2 (Sun 4/75).

- By analysing twitch response data and frequency components of the data, initial information about the time constants and delays of the muscle can be obtained.
- Simple identification experiments with 2nd order linear models provide information about how non-linear the muscle characteristics are under the stimulation conditions investigated. If the model parameters are estimated in the delta operator domain, a fast sampling period is not critical.
- The scheduling variable should be obtained from the stimulation input to the model. This can be either a filtered version of the stimulation pulses, or other properties of the input such as pulse-width or current if these parameters vary, or a combination of those.
- LMNs with local models in state space form should be used as these give significantly better modelling results than LMNs with local models in input-output notation. The parameters can be identified in simulation mode using the Levenberg-Marquardt algorithm.
- Starting with a small network, the number of units can be increased until an optimal network size is found, i.e. until the model performance on test data starts to deteriorate.
- The model analysis consists of three steps:
  - The model is tested against experimental test data which have not been used to optimise the model parameters.
  - Known characteristics of muscle such as the force-frequency and the recruitment curve can be verified with the model in simulation experiments.
  - Model properties such as steady state gain, the location of the poles and the shape of validity functions are analysed for varying scheduling variables and can be related to known characteristics of muscle. In particular, the change of time-constants with varying muscle activation, saturation and possibly model instabilities can be detected.

It can be expected that the application of this modelling technique for isometric contraction of different muscles, e.g. human muscle, is straightforward. For more general stimulation conditions, the modelling cost, in particular the effort to collect sufficient experimental data, could increase significantly. It will also become more difficult to relate model properties to known characteristics of the muscle. However, a basic model mismatch such as instability or lack of saturation will still be easily detectable.

### Further Work

In its current form, the model does not include varying motor unit recruitment. The model structure can be extended in a straightforward way by either adding an explicit recruitment

model, or by including the recruitment effects in form of a varying amplitude of the stimulation pulses. The scheduling vector may need to be extended to account for non-linear recruitment effects.

To model non-isometric muscle contraction, the non-linear model described here can be used as the active force-generating element of a Hill-type model structure. In the model structure used by (Durfee and Palmer 1994) which is shown in Figure 4.4 on page 46, the non-linear model of isometric contraction introduced in this chapter can replace the Hammerstein model to describe the activation dynamics of the muscle. Local Model Networks can also be used to approximate the force–length and the force–velocity characteristics which are modelled by piecewise linear curves in (Durfee and Palmer 1994).

Initial experiments with non-isometric data give promising results. Due to lack of experimental resources it was, however, impossible to collect sufficient data for a given muscle to obtain results which could be presented as part of this thesis. Additional effort will be required to develop a more systematic approach to collect input–output data for non-isometric contraction.

# 6 Generation of Optimal Stimulation Patterns

When discussing aspects of artificial stimulation of muscle in Section 4.2.3, it was pointed out that owing to the differences of artificial stimulation from the way the muscle is stimulated by the central nervous system, the characteristics of the muscle tissue can change over time. We discussed that, in order to retain the power of the muscle and to minimise fatigue, it is desirable to obtain the same mechanical response by fewer stimulation pulses. We referred to such stimulation patterns as "optimal".

This chapter is aimed at analysing the properties of *optimal stimulation patterns* (OSPs) for the muscles investigated in Chapter 5, and at developing algorithms to generate such patterns.

## 6.1 Introduction

We define optimal stimulation patterns as stimulation sequences which maximise the force produced by the muscle over time per stimulation pulse. The muscle force over time is described by the *force-time integral* (FTI),

$$FTI_p = \int_{t=t_0}^{t_1} F[u_p(\cdot), t]\, dt\,, \tag{6.1}$$

where $F[u_p(\cdot), t]$ is the output force when the muscle is stimulated with the stimulation pattern $u_p(\cdot)$, and $t_0$ and $t_1$ are the beginning and the end of the recorded interval. The index $_p$ denotes the number of pulses in the stimulation pattern. All stimulation pulses have the same amplitude and width. For a sampled output force sequence with sampling period $T_s$, the integral is approximated by the sum,

$$FTI_p = \sum_{i=1}^{N} F[\{u\}_p, \tau_i] T_s\,, \tag{6.2}$$

where $\tau_i$ denotes the discrete time, and the force is recorded in the interval $[\tau_1 = t_0, \ldots, \tau_N = t_1 - T_s]$. The discrete stimulation sequence containing $p$ pulses is denoted by $\{u\}_p$. As a measurement of the force generated *per pulse*, the *normalised force-time integral per pulse*

(FTIpP) is introduced,

$$FTIpP_p = \frac{FTI_p}{FTI_1 \times p}. \tag{6.3}$$

Here $p$ denotes the number of impulses in the stimulation pattern, $FTI_p$ is the force–time integral for a sequence of $p$ pulses, $\{u\}_p$, and $FTI_1$ is the force–time integral of the response to a single impulse.

An optimal stimulation pattern of $p$ stimulation pulses, $\{u\}_p^*$, can now be defined as a sequence of impulses which maximises the force–time integral per pulse for the given muscle,

$$\{u\}_p^* = \arg\max_{\{u\}_p} FTIpP\left(\{u\}_p\right). \tag{6.4}$$

In this chapter different methods to generate optimal stimulation patterns are introduced. The non-linear models of isometric muscle contraction developed in Section 5.4 are used in these algorithms. Unless otherwise stated, we work with the LMNs with local models in state space notation with real eigensystems and common eigenvectors which were analysed in Section 5.4.3.

The methods discussed in Sections 6.2 to 6.4 are adapted from (Kwende *et al.* 1995), where they were used in experiments with the real muscles from which the data used in Chapter 5 to identify the models were obtained. The methods described in Sections 6.2 and 6.3 can not be used directly to generate optimal stimulation patterns. However, they give valuable insight concerning the different properties of optimal stimulation patterns for the two muscles. By showing that the results obtained with the *models* of the muscles are equivalent to those obtained with the real muscles, the validity of the models is further demonstrated.

The iterative method introduced in Section 6.4 is suitable for *generating* optimal stimulation patterns. In Section 6.5, a more general way of obtaining optimal stimulation patterns for specific desired contractions is described. This method is based on a non-linear control structures using Local Controller Networks.

## 6.2 Random Pattern Method

The random pattern method is aimed at finding from a number of randomly generated stimulation patterns those patterns which maximise the FTIpP. It is used here to verify that the models developed in Section 5.4 produce results which are equivalent to those reported in (Kwende *et al.* 1995), and to analyse characteristic properties of optimal stimulation patterns.

### 6.2.1 Experimental Setup and Results

Stimulation patterns with randomly varying IPIs, which were generated according to the setup outlined in Section 5.1, are applied as inputs to the models of the fast and of the slow muscles.

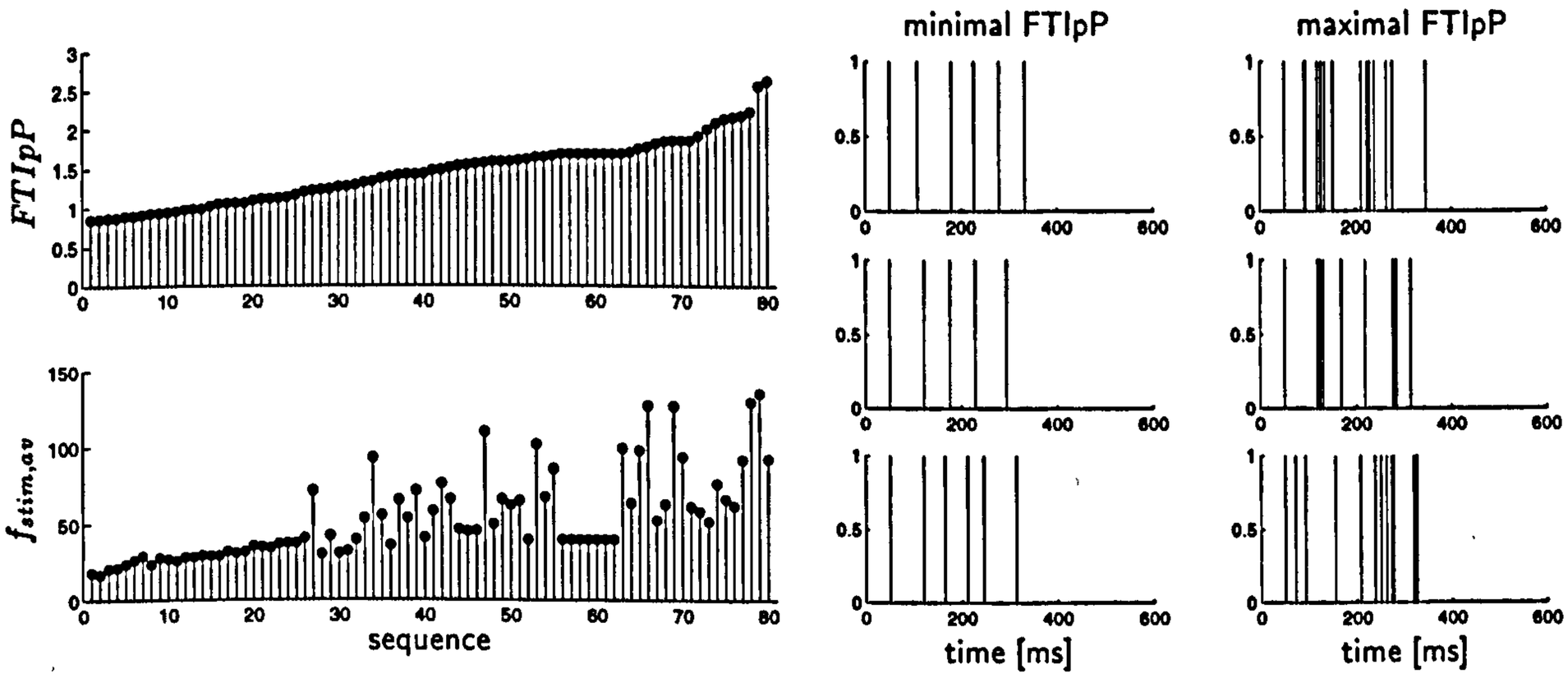
For each input pattern, a sequence of stimulation frequencies is determined as

$$f_{stim}(k) = \frac{1}{t_{k+1} - t_k}, \quad k = 1 \ldots (p-1) \tag{6.5}$$

where $p$ is the total number of pulses in the input pattern, $t_k$ denotes the time when the $k$-th impulse is delivered, and $f_{stim}(k)$ is the corresponding stimulation frequency. Thus, an *average stimulation frequency*, $f_{stim,av}$, can be obtained to describe each stimulation pattern,

$$f_{stim,av} = \frac{1}{p-1} \sum_{k=1}^{p-1} f_{stim}(k)\,. \tag{6.6}$$

The FTIpPs for each stimulation pattern for both muscle models are calculated according to equation (6.3). The results are shown in Figure 6.1 for the fast muscle and in Figure 6.2 for the slow muscle.



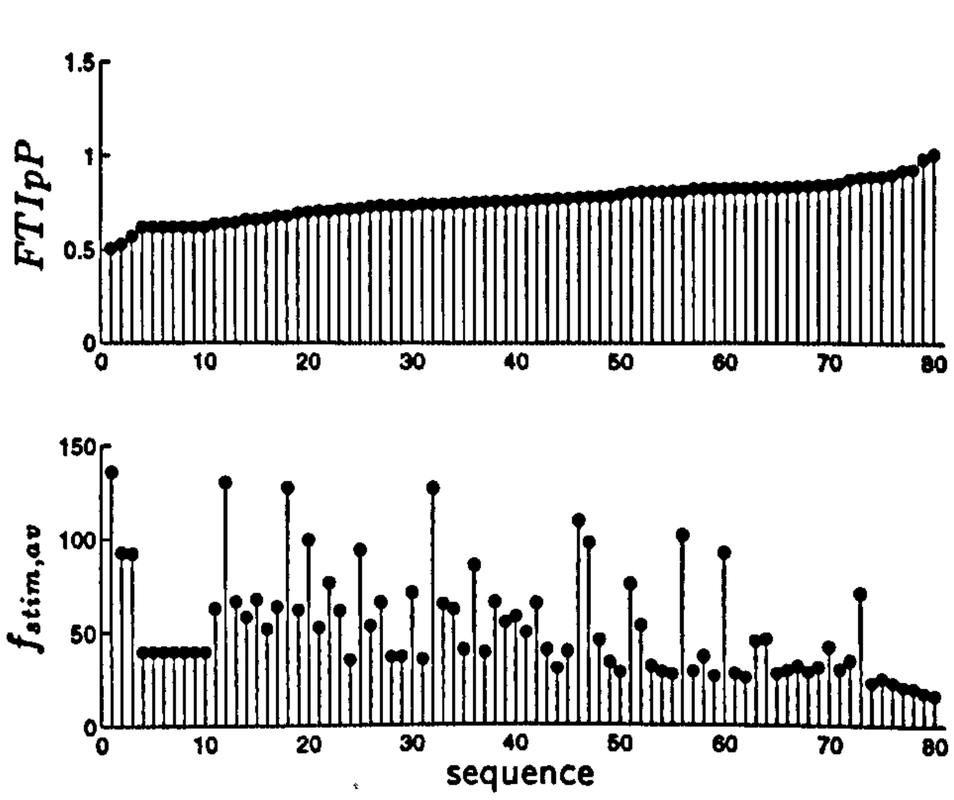

(a) Normalised FTIpP and average stimulation frequency for stimulation patterns sorted by FTIpP.

(b) Stimulation patterns corresponding to minimal and maximal FTIpP.

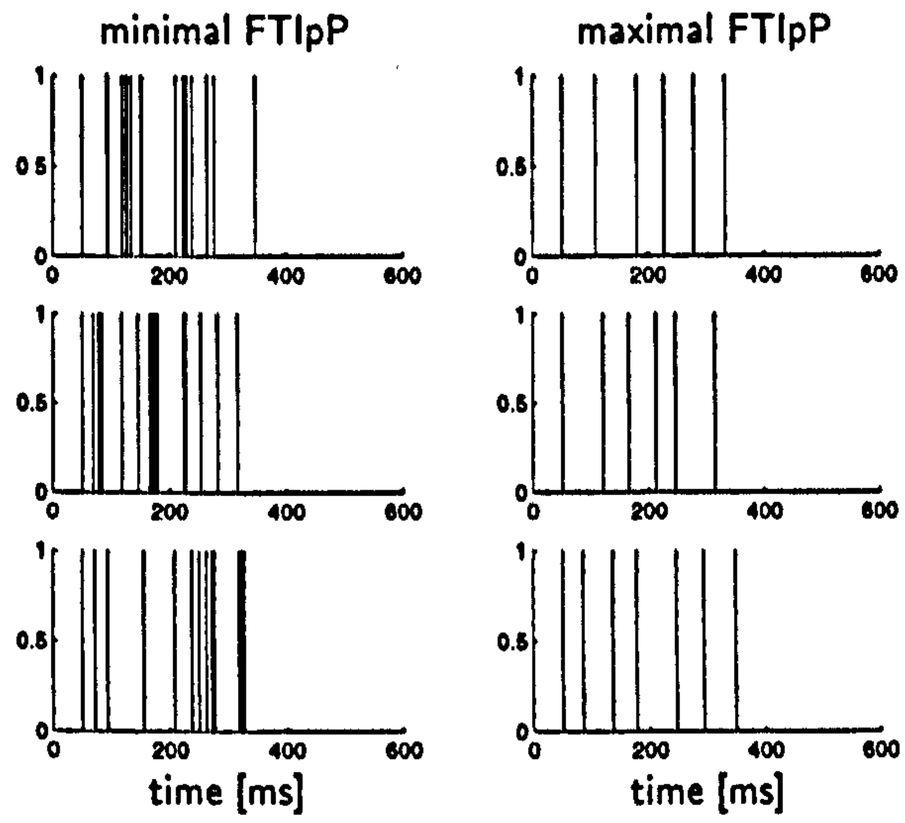
Figure 6.1: Random pattern method applied to the fast muscle model.

### 6.2.2 Conclusions

For the model of the fast muscle the stimulation patterns which result in the largest FTIpP contain pulses with high average frequency. The patterns consist of pulses with generally small and irregular IPI, cf. Figure 6.1(b). The lowest value of the FTIpP can be observed for pulse patterns with low average frequencies. These stimulation patterns consist of regularly spaced impulses with large IPIs.




(a) Normalised FTIpP and average stimulation frequency for stimulation patterns sorted by FTIpP.




(b) Stimulation patterns corresponding to minimal and maximal FTIpP.

Figure 6.2: Random pattern method applied to the slow muscle model.

For the model of the slow muscle, the FTIpP is large for stimulation patterns with a low average frequency. The best stimulation patterns in terms of maximal FTIpP have low frequency components, the worst contain components of high frequency, cf. Figure 6.2(b).

Comparing the patterns shown in Figure 6.1(b) with those depicted in Figure 6.2(b) shows that the stimulation patterns which result in the maximal FTIpP for the fast muscle give minimal FTIpP for the slow muscle, and vice versa. This supports the observation that the "catch-like" effect is only present in the fast muscle, and that this characteristic has been lost in the slow muscle.

The results are equivalent to those described in (Kwende *et al.* 1995) for experiments with real muscles. This confirms that the muscle models describe non-linear effects due to stimulation with irregular pulse trains correctly.

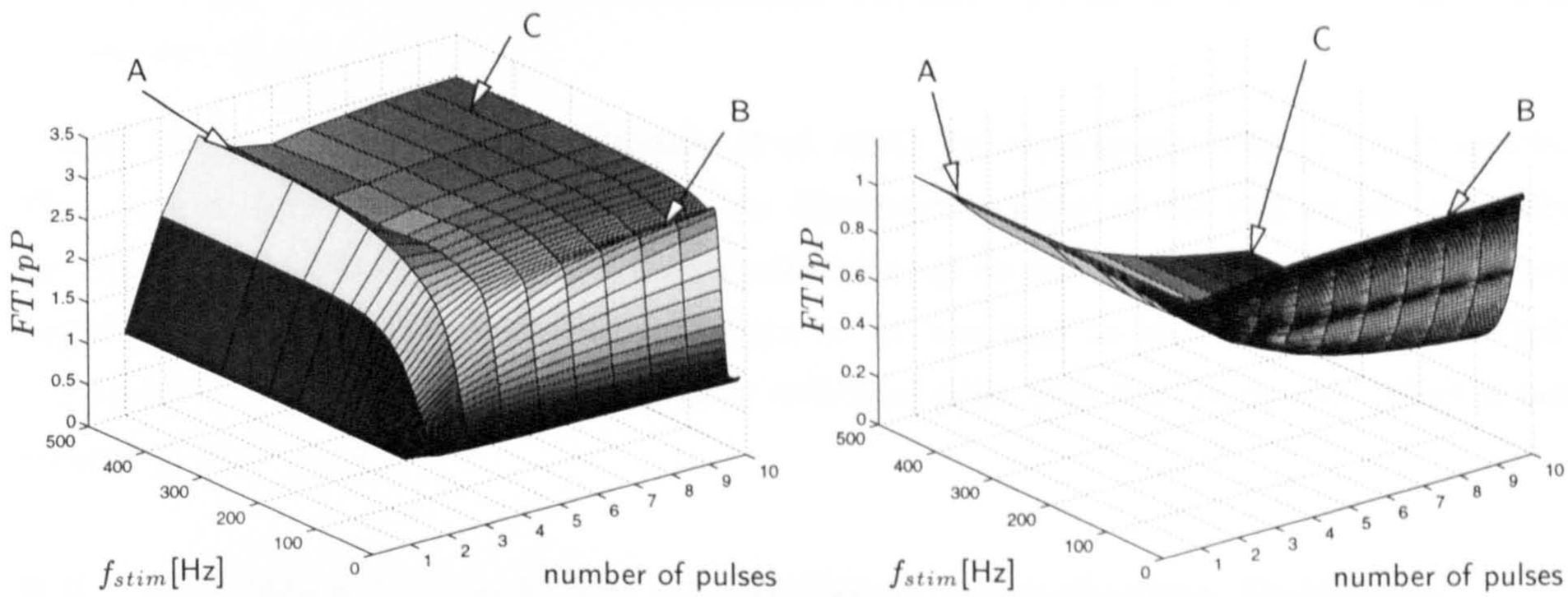
## 6.3 Constant-Frequency Train Method

In the method described in this section, short constant frequency trains with 1 to 10 impulses are generated for different IPIs. The FTIpP of the two muscle models for these input patterns are analysed.

### 6.3.1 Experimental Setup and Results

Impulse trains with constant IPIs of 2 to 100ms (equivalent to a range of stimulation frequencies, $f_{stim}$, from 10 to 500Hz) are generated with 1 to 10 impulses. The FTIpPs, equation (6.3), generated by the muscle models are obtained for each stimulation sequence. The results are then plotted as three-dimensional surfaces.

The results obtained with the models of the fast and of the slow muscles are shown in Figure 6.3.




(a) Fast muscle model.

(b) Slow muscle model.

Figure 6.3: Constant-frequency train method.

### 6.3.2 Conclusions

The results for the model of the fast muscle show a significant increase of the FTIpP for stimulation pulses with more than one pulse. Three areas of high FTIpP can be distinguished which are marked with the corresponding letters in Figure 6.3(a):

A) short stimulation bursts ($\approx 3$ pulses) with high stimulation frequency,

B) longer bursts with low stimulation frequency ($f_{stim} \approx 50$Hz), and

C) long bursts with high stimulation frequency.

In the results from experiments with real muscle as reported in (Kwende *et al.* 1995), areas A) and B) were also found to have high FTIpP. In area C), however, the FTIpP decreased rapidly with the real muscle and approached $FTIpP \approx 1$ for long burst of high frequency stimulation. Thus, the model fails to predict the muscle characteristics correctly for long stimulation patterns with constant high frequency. This model deficiency was previously

observed when the models were analysed, cf. Section 5.4.3, in particular when the force-frequency curves were discussed. Note that region A) corresponds to the "catch-like" effect.

For the model of the slow muscle, the following three areas can be distinguished in Figure 6.4(b):

A) The FTIpP is maximal for stimulation with a single pulse.

B) Another maximum of the FTIpP can be observed for pulse trains of different length with low stimulation frequency ($f_{stim} \approx 10$Hz).

C) The FTIpP decreases when the stimulation frequency is increased and the pulse trains become longer.

Similar results were reported in (Kwende *et al.* 1995) for experiments with the real muscle. Thus, the model of the slow muscle predicts the characteristics of the real muscle correctly for stimulation patterns of different frequencies with up to 10 impulses. The fact that short impulse bursts with high stimulation frequencies do not lead to an increase of the FTIpP compared to stimulation with a single pulse indicates again that the "catch-like" effect is not present in this muscle.

## 6.4 Iterative Generation of Optimal Stimulation Patterns

In this section, an iterative method will be described which allows for a constructive generation of optimal stimulation patterns which maximise the FTIpP. An approach of this type has also been used in (Karu *et al.* 1995) and (Zajac and Young 1980).

### 6.4.1 Experimental Setup and Results

In the iterative method for generating optimal stimulation patterns, the model of the muscle is subjected to a growing number of impulses. Whenever a new impulse is added, its IPI to the previous pulse is optimised such that the FTIpP is maximal.

The algorithm can be described as follows:

1. The response to a single pulse (the twitch response) is recorded.

2. Pulse trains consisting of two pulses are applied. The IPI varies between 2 and 100ms in steps of 1ms. The pulse train which maximises the FTIpP is selected and the IPI between the first and second pulse is fixed to the corresponding optimal value $IPI^*_{1,2}$.

3. Pulse trains with three pulses are applied where the IPI between the first and second pulse is fixed to the optimal value $IPI^*_{1,2}$ obtained in the previous step, and the distance between the second and third pulse is varied as in step 2. The IPI which maximises the FTIpP is selected as the optimal value, $IPI^*_{2,3}$.

4. Further pulses are iteratively added to the stimulation pattern as in step 3.

The results obtained with Local Model Networks for up to five impulses are shown in Figure 6.4 for the models of the fast muscle and of the slow muscle.

### 6.4.2 Conclusions

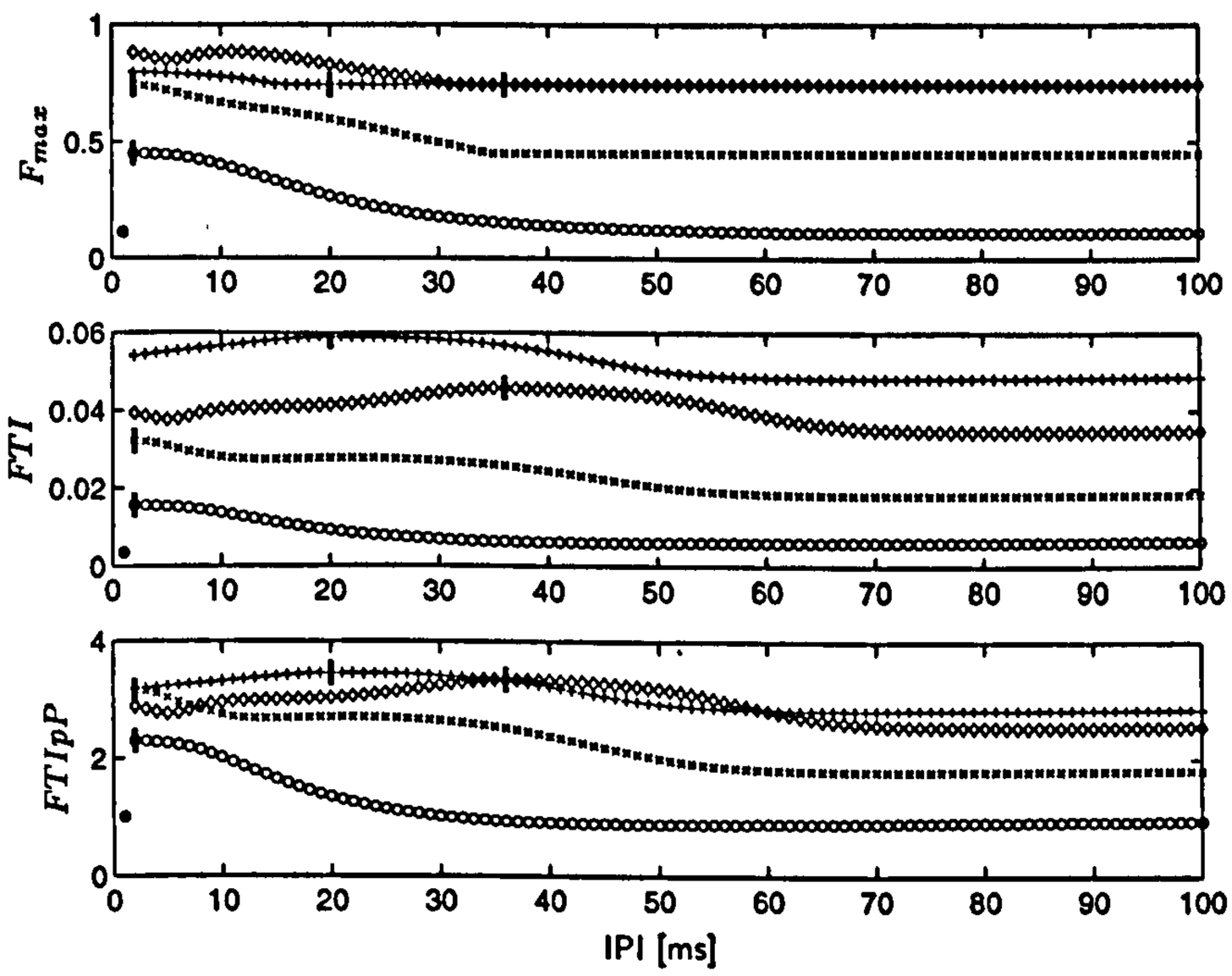
The results show that for the model of the fast muscle the FTIpP increases when few pulses are added at a very short IPI, cf. Figure 6.4(a). The optimal IPI for the 4th and 5th input is, however, relatively large. This indicates that a very short IPI is only useful at the beginning of the stimulation. As stimulation continues, the pulse frequency should be decreased to obtain the maximal FTIpP.

For the model of the slow muscle, the maximal FTIpP is obtained for the twitch response, cf. Figure 6.4(b). The FTIpP decreases as further pulses are added. Thus, the maximal normalised FTIpP is equal to one, and can only be reached for very long IPIs for which the responses to the inputs pulses are well separated.

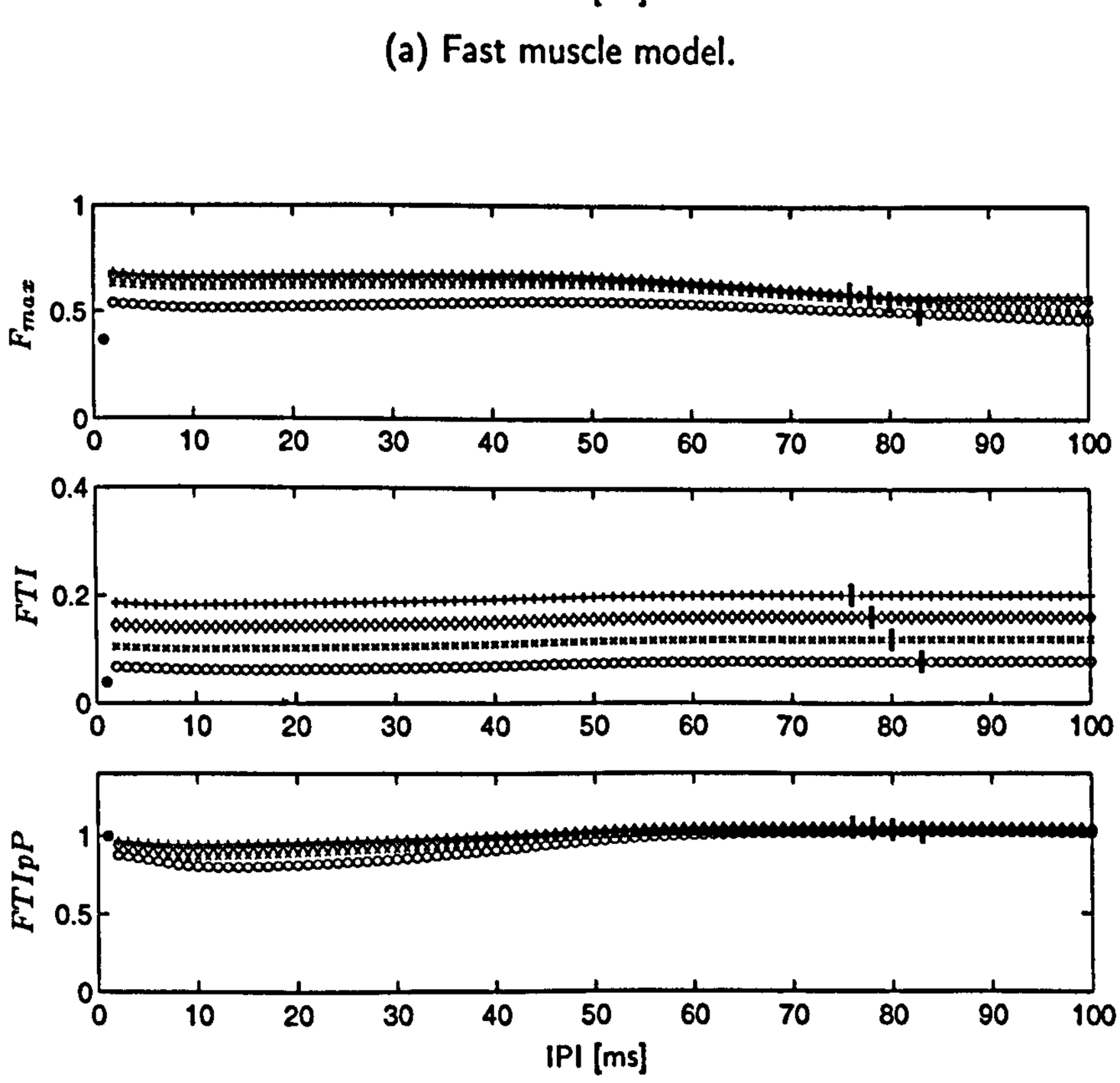
In (Kwende *et al.* 1995), the same technique was used with the real muscles. The results reported there are very similar to those obtained here with the muscle models. Thus, the models developed can be used with this iterative algorithm to constructively generate optimal stimulation patterns and to analyse their properties without the need for expensive experiments on real muscle.

The fact that the optimal stimulation pattern found for the fast muscle has pulse patterns of varying IPI shows that changing the stimulation frequency during a given contraction is essential to obtain the maximal force-time integral. Thus, stimulation with constant frequency pulse trains is not optimal for this muscle.

The iterative algorithm described is only aimed at maximising the force-time integral. It cannot account for a desired muscle contraction which should be obtained by delivering the minimum number of pulses possible. It is also computationally very expensive, as it is essentially a "trial and error" strategy where all possible IPIs are investigated before the optimal one is selected.




(a) Fast muscle model.




(b) Slow muscle model.

Figure 6.4: Iterative method: o denotes 2 pulses, x 3 pulses, ◇ 4 pulses and + stands for 5 pulses. Responses to a single pulse are shown as •. The short vertical bars denote the optimal values of the IPI which are selected for the pulse train. In each subfigure, the top plot shows the maximal force produced, the force–time integral is depicted in the middle plot, and the bottom plot shows the normalised force–time integral per pulse.

## 6.5 Non-linear Control

In this section, a more constructive algorithm for generating optimal stimulation patterns is introduced which is based on a simulated non-linear closed-loop control structure. Local Controller Networks, as introduced in Chapter 3, are used as controllers. The objective is to generate optimal stimulation patterns (in the sense of equation (6.4)) which are specific for a desired contraction.

In many FES applications a direct measurement of the muscle output is not available as sensors cannot be placed at the muscle actuator. This is, for example, the case in the application of SMVs for cardiac assistance, cf. Section 1.1. In the control structure shown in Figure 1.1 on page 2, only the outer control loop (Controller 1) has access to feedback information from the flow and pressure sensors. The inner Controller 2 is an open-loop controller as direct output information from the SMV is not available. For this reason, we aim to develop an algorithm for the generation of stimulation patterns which does not require feedback information from the muscle.

### 6.5.1 Concept

A straightforward technique for generating input patterns for the muscle to perform a desired contraction is to use an inverse model of the plant. The inverse model generates a continuous control signal $u_{cont}$ based on the reference signal $r$. In order for the control signal to be used as a stimulation pattern for the muscle, it needs to be transformed to a pattern of impulses $u_{pulse}$ using some pulse shaping algorithm. The stimulation pattern obtained is applied to the muscle in order to let it perform the desired contraction. This setup, which is an open-loop control structure, is shown in Figure 6.5.

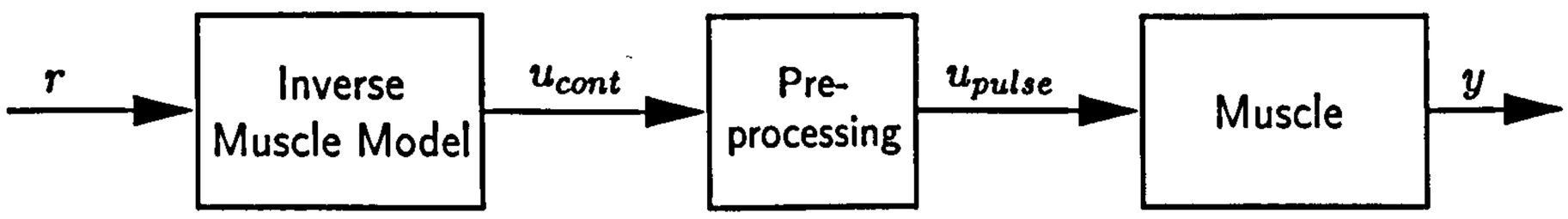


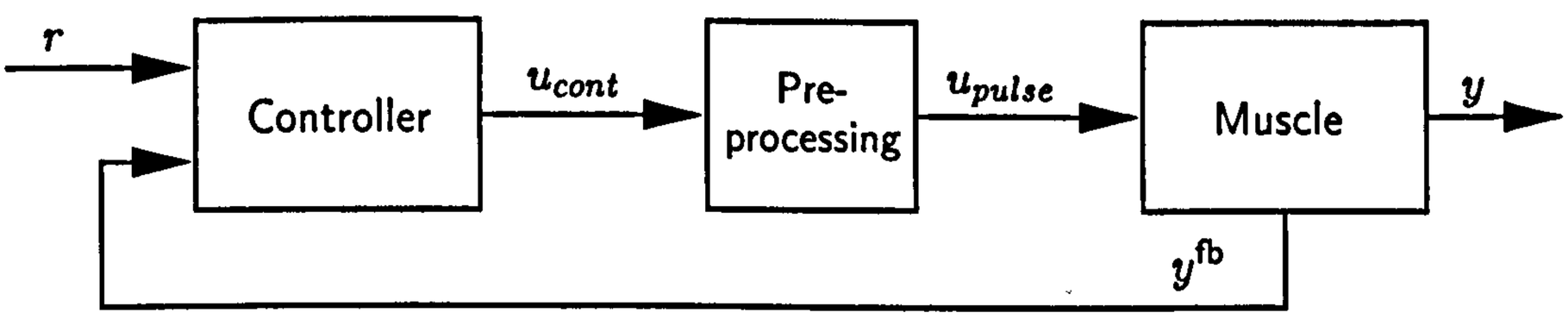

Figure 6.5: Generation of stimulation patterns with an open-loop control structure based on an inverse model.

In order to obtain an inverse model, the plant itself must be invertible which is not always the case. For electrically stimulated muscle, it is known that the same contraction can be obtained by applying different input patterns[1]. Thus, muscle performs a many-to-one mapping from the input space to the output space. A unique inverse model therefore does not exist. This observation is supported by the results obtained in Section 5.4 for the

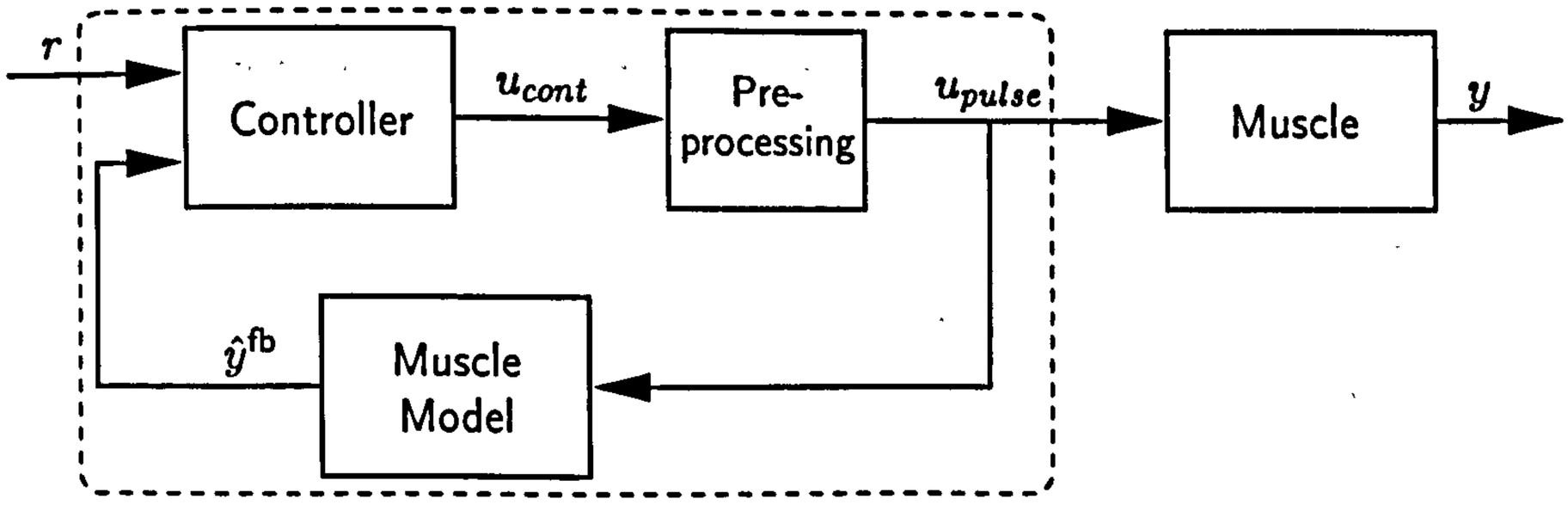
[1] In fact, the objective of this section is to find an algorithm to select from the various possible input patterns the one which is optimal in the sense of equation (6.4).

selection of the scheduling variable for LMN model structures. There it was found that the pre-processed input needs to be included in the scheduling vector to obtain a model structure which is suitable for the fast and for the slow muscle. As we discussed in Section 2.2.4, this implies that the model performs a many-to-one mapping from the input to the output. Thus, the obtained LMN model cannot be inverted as a unique inverse does not exist. The open-loop structure cannot therefore be used to generate stimulation patterns with the LMN models available. However, it should be noted that the structure shown in Figure 6.5 has the advantage that no feedback information of the muscle output is necessary.

The open-loop control setup can be extended to a closed-loop control structure where the controller is based on a forward model of the muscle and which provides additional degrees of freedom when designing the controller to ensure that the generated stimulation patterns are optimal. A possible setup is shown in Figure 6.6(a).




(a) Closed-loop control with feedback from the real muscle.




(b) Open-loop control with a controller based on simulated closed-loop control with a feedback signal estimated by a model of the muscle. The dashed line denotes the elements of the open-loop controller as seen by the muscle.

Figure 6.6: Control structures for the generation of stimulation patterns.

The control loop consists of the controller which generates the control signal based on the command signal $r$ and a feedback signal $y^{fb}$. As described above, the continuous control signal $u_{cont}$ is transformed to a pulse pattern $u_{pulse}$ which is applied to the muscle. As the pre-processing of the control signal is now part of the closed loop, non-linear effects introduced by it can be rejected by the controller. The control activity is defined by the controller design.

For example, the controller can be designed in such a way that the control signal changes rapidly, or the design can be aimed at obtaining a smoothly varying control signal. Thus, the properties of the stimulation patterns can be influenced by the design of the controller.

The main disadvantage of this structure is that it requires the feedback signal $y^{fb}$ which is normally a measurement of the muscle output, $y$. As outlined earlier we assume that such a feedback signal is not available. The closed-loop controller can be modified in such a way that the feedback signal is estimated by a model of the muscle. This setup is shown in Figure 6.6(b). The control-loop now consists of the controller, the pre-processing and, instead of the real muscle, a model of the plant. The controller can be an LCN which is designed based on the LMN model of the muscle. By using a model of the muscle in the control loop, we are able to exploit the extra freedom gained by closed-loop control and to take account of the effects introduced by the pre-processing of the control signal without the need for sensor information of the output of the real muscle. An advantage from the controller design point of view is that the feedback signal does not necessarily have to be the output of the model. It can, for instance, be the entire state vector of the model, avoiding the need for a state estimator.

For the plant, the structure shown in Figure 6.6(b) represents an open-loop controller which consists of the loop formed by the controller, the pre-processing and the muscle model. Note that, unlike for the linear case, the simulated closed loop cannot be simply replaced by a single forward controller, owing to the non-linearities in this loop.

### 6.5.2 Pre-processing of the Control Signal

When using controller techniques as shown in Figure 6.5 and 6.6 in the muscle application, the problem arises that the control signal $\{u_{cont}(t)\}$ given by the controller is continuous both in value *and* in time, i.e., the control signal can have any value at any sampling interval. For physiological reasons, the muscle can only be stimulated using impulse trains. In the application considered in this thesis, the muscle is stimulated with supramaximal pulses which implies that all pulses have the same amplitude. Thus, an algorithm has to be designed which transforms the continuous signal patterns $\{u_{cont}(t)\}$ into a train of binary impulses $\{u_{pulse}(t)\}$. The following objectives have to be taken into account:

1. The energy fed into the system by the pulse signal $\{u_{pulse}(t)\}$ needs to be related to the energy of the original continuous signal $\{u_{cont}(t)\}$, i.e.,

$$\sum_{i=1}^{N} u_{cont}(t_i)\, T_s \approx \sum_{i=1}^{N} u_{pulse}(t_i)\, T_s\,, \tag{6.7}$$

where $N$ is the number of samples, $T_s$ denotes the sampling period, and $t_i$ is the discrete time.

2. In the pulse signal, $\{u_{pulse}(t)\}$, a pulse must be followed by a zero signal. Otherwise, a large continuous signal would be transformed into a continuous series of ones.

In order to preserve the energy of the signal, the *energy since the last pulse*, $E$, is introduced[2],

$$E(t_k, t_{lp}) = \sum_{i=lp+1}^{k} u_{cont}(t_i)\,. \tag{6.8}$$

Here, $t_k$ denotes the current time instant and $t_{lp}$ is the time when the last impulse was delivered. If the energy described by equation (6.8) exceeds a threshold $\eta$, and if $k > lp + 1$, an impulse is delivered, and $t_{lp}$ is reset to the current time $t_k$. To ensure that the continuous and the pulse signal have the same energy the threshold needs to be chosen equal to one. An example of a transformation of a continuous signal to a series of pulses is shown in Figure 6.7.

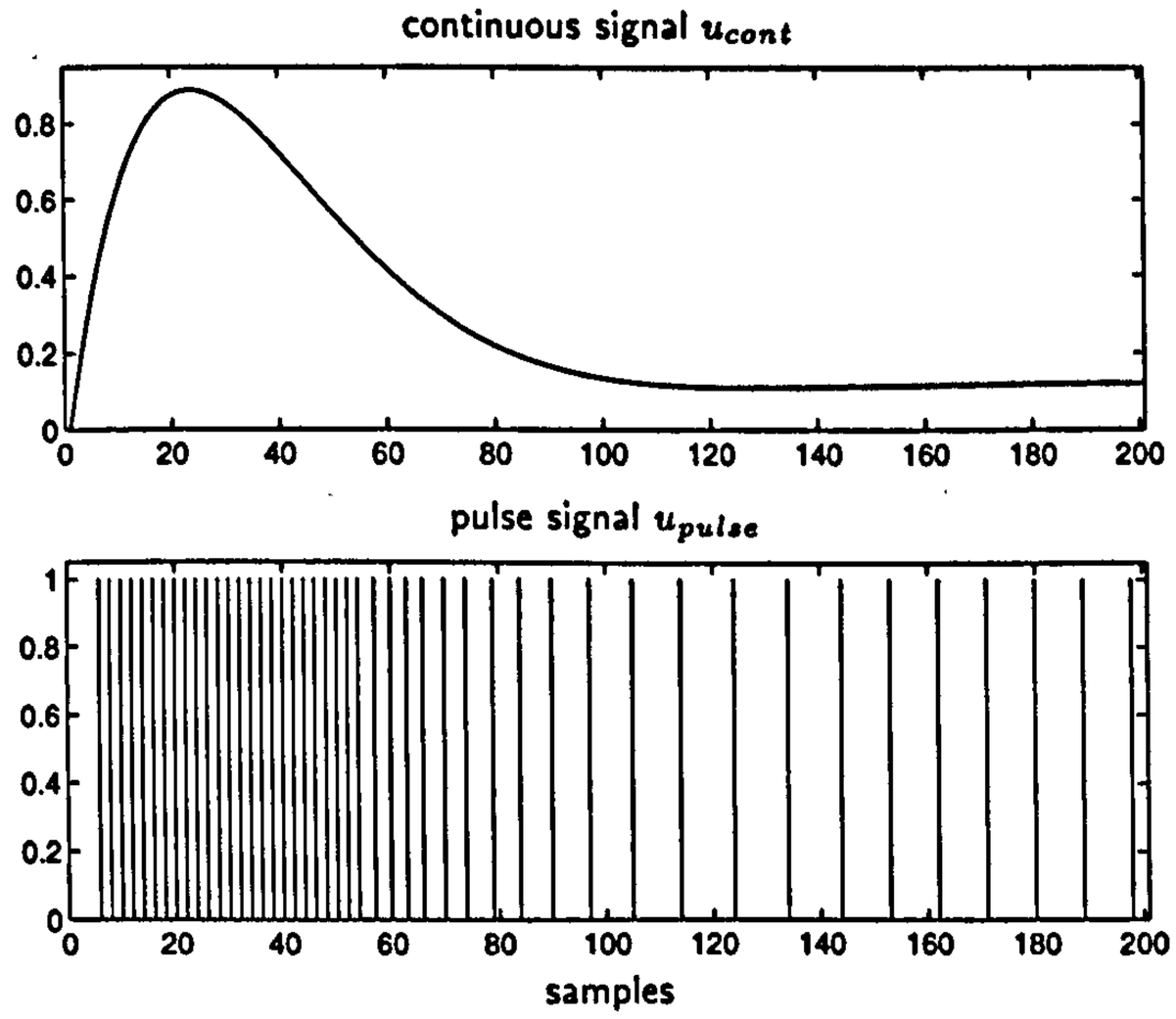



Figure 6.7: Pre-processing of the continuous control signal to obtain a pulse train.

### 6.5.3 Controller Design and Results

Based on the results obtained in Chapter 5, Local Model Networks in state space notation with common eigenvectors will be used as the model of the muscle in the control structure shown in Figure 6.6(b). Thus, it is straightforward to use the concept of Local Controller Networks as introduced in Chapter 3 for the controller. The suitability of LCNs to control the fast muscle was shown in (Gollee and Hunt 1997). The approach described there is, however, based on LMNs in input–output notation, and the controller designed is not particularly aimed at generating optimal stimulation patterns.

[2]Note, that the sampling period $T_s$ has been dropped for convenience.

As the muscle model is in state-space form, we choose to work with a state-feedback controller. Aspects of this control structure and the controller design were introduced in Sections 3.3 and C.2. Note that a state estimator is not needed in the setup used here as the full state is available from the muscle model.

The complete plant in the closed-loop consists of the LMN as the design model, augmented by the pre-processing algorithm which transforms the continuous control signal into a series of pulses. From the controller design point of view, the effects introduced by the pre-processing will be treated as modelling uncertainties. Integral action will therefore be included in the controller to enhance robustness, in particular to ensure zero steady state error. The resulting control structure which is adapted from Figure 6.6(b), is shown in Figure 6.8. The structure of the local controllers of the LCN is depicted in Figure C.3 on page 139.

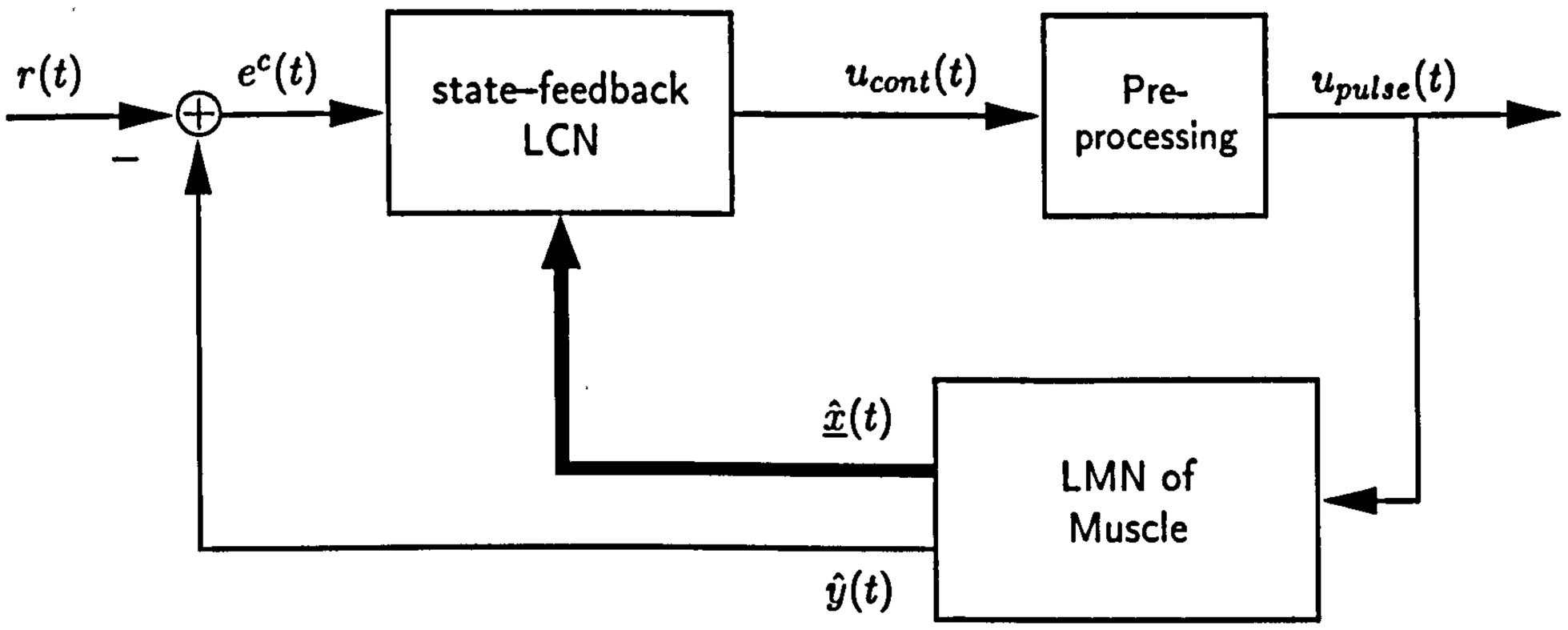



Figure 6.8: Closed-loop state-feedback control structure for the generation of optimal stimulation patterns. The muscle is modelled by an LMN, the controller has the form of an LCN.

We use pole-placement to design the local controllers of the LCN. The time-domain parameters rise-time $t_r$ and the damping factor $\xi$ are selected to define a desired dominant pair of closed-loop poles, $p_{1/2}$. The control parameters are calculated as described in Section 3.3. Owing to the integral action, the closed loop has the order $n+1$, where $n$ is the order of the plant, cf. equation (C.27) on page 138. With a second order plant, we obtain a closed loop with an extra pole, $p_3$, in addition to the pair of poles $p_{1/2}$ defined above. To retain the dominance of the pole pair, the extra pole needs to be faster than $p_{1/2}$. Thus, we choose to set $p_3 = -1.9/T_s$, where $T_s$ is the sampling period.

### Design Objectives

The overall goal is to generate optimal stimulation patterns which make the muscle perform a desired action given by the command signal $r(t)$. Thus, one design objective is to make the model output $\hat{y}(t)$ follow the command signal $r(t)$, i.e., to minimise the control error

$e^c(t) = \hat{y}(t) - r(t)$. The second objective is to make the pulse command signal $u_{pulse}(t)$ an optimal stimulation pattern in the sense of equation (6.4). The different properties of optimal stimulation patterns for the fast and for the slow muscle have been investigated in the Sections 6.2 to 6.4. These results will be considered for the controller design.

Owing to the fact that we are working with a model of the muscle, no measurement noise can enter the closed loop. Thus, the controller design does not need to be robust with respect to such disturbances.

The effects introduced by the pre-processing of the control signal need to be accounted for. This is ensured by including integral action in the controller.

The results are evaluated with the design model as both the muscle model and the muscle itself in Figure 6.6(b).

### Fast Muscle

We choose to work with the optimal LMN with local models in state space notation with real eigensystems and common eigenvectors (see Table 5.2 on page 78) as the design plant. The damping factor is chosen to be $\xi = 1.2$, i.e. the closed-loop response should have no overshoot. The rise time is selected individually for each operating regime. The values shown in Table 6.1 were found to be optimal with respect to the design objectives.

| # unit | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| $t_r[ms]$ | 100 | 30 | 30 | 10 | 20 | 30 |
| $p_1$ | −57.9 | −180.2 | −180.2 | −449.1 | −257.8 | −180.2 |
| $p_2$ | −17.0 | −55.6 | −55.6 | −157.8 | −82.3 | −55.6 |
| $K_{x,1}$ | 83.0 | 3.3 | 9.1 | 28.2 | 9.0 | 11.7 |
| $K_{x,2}$ | −5.8 | −1.7 | −2.0 | −1.9 | −1.1 | −0.3 |
| $K_I$ | 2340 | 680.5 | 597.7 | 250.1 | 517.2 | 599.5 |

Table 6.1: Fast muscle: closed-loop rise times for the LCN design, dominant closed-loop poles and state feedback gains.

The control results for step-like command signals of different amplitude are shown in Figure 6.9.

### Slow Muscle

In a first approach, the optimal LMN with local models in state space notation with real eigensystems and common eigenvectors (see Table 5.2 on page 78) was used as the design plant. However, the performance of the resulting controller was not satisfactory. Analysis shows that the gains of some of the local models of the design plant are negative, cf. Figure 5.27(b) on

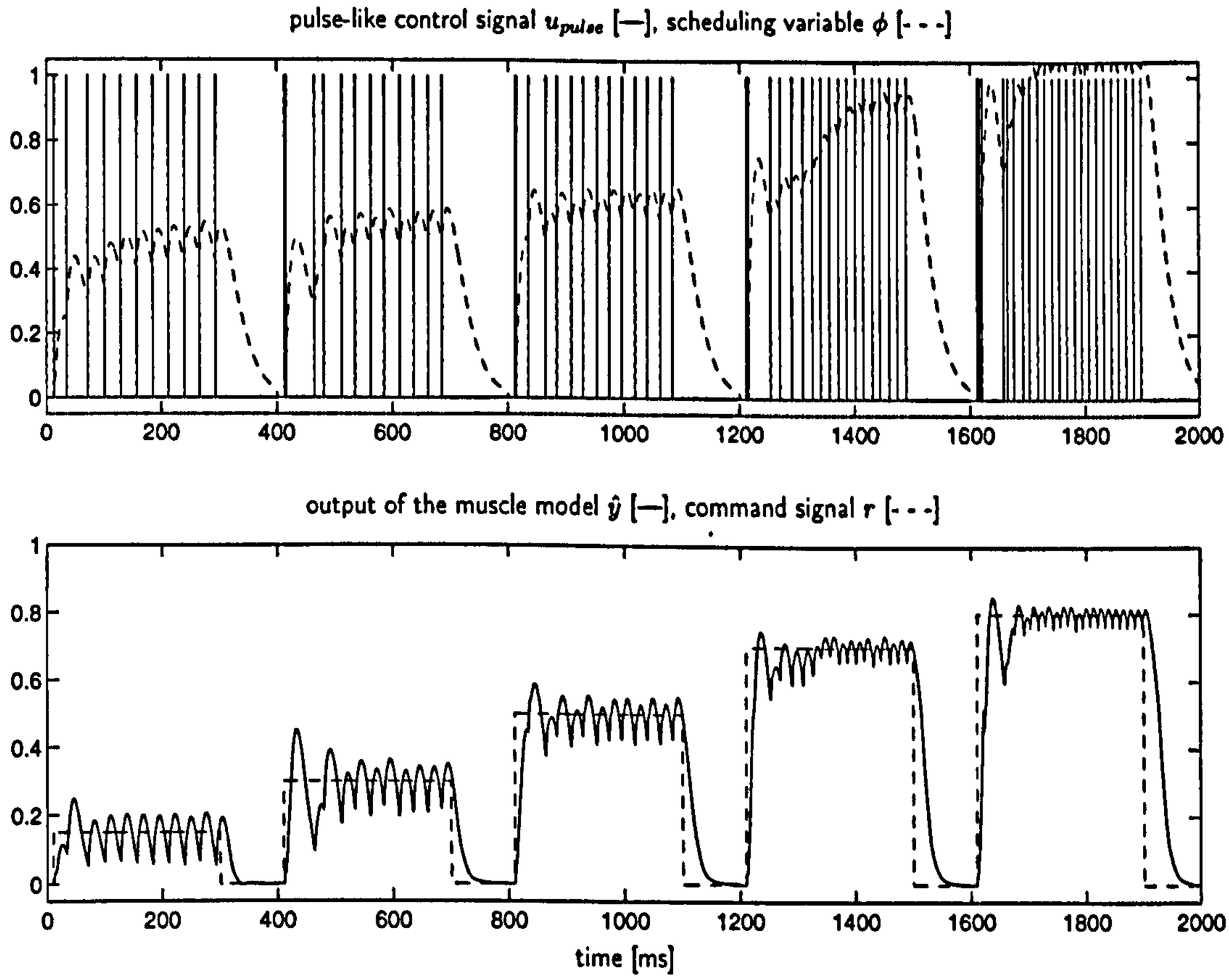



Figure 6.9: Generation of optimal stimulation patterns by non-linear control, fast muscle model.

page 82. A controller designed with this LMN can get into a state where it switches between the controller associated with a local model with positive gain and a neighbouring controller which is associated with a local model with negative gain. The resulting characteristics of the closed loop are therefore not satisfactory.

Thus, the controller design needs to be based on a LMN with local models which do not have negative gains. The gain of the overall muscle system is obviously non-negative for all operating conditions. Thus, the gains of the local models can only become negative when the local offset terms are non-zero. By fixing the offset terms $\{d_i^x, d_i^y\}_{i=1}^M$ to zero, an LMN can be identified for the slow muscle where the gains of all local models are non-negative. The results of a static analysis of such a model with 5 units, together with the training and test errors, are shown in Figure 6.10. Although the training and test errors of this model are larger than the errors for a model with non-zero offset terms, cf. Table 5.2 on page 78, the performance on experimental data is similar to the simulation results shown in Figure 5.25 on page 80.

When this LMN is used as the design model for the controller, the resulting closed-loop behaviour is satisfactory. As before, the damping factor is chosen to be $\xi = 1.2$, i.e. the closed-loop response should have no overshoot. The rise time is selected individually for each operating regime. Good results could be obtained when the values were equally spaced in the

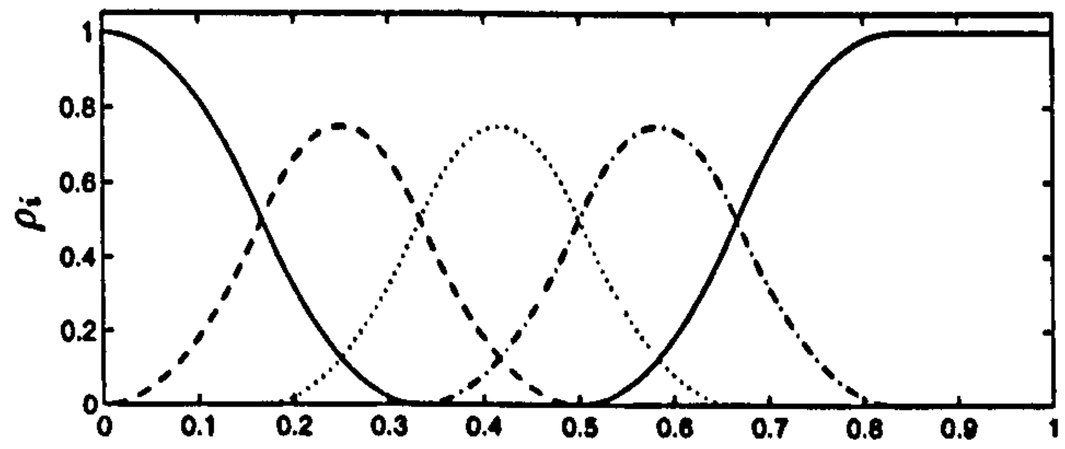



Figure 6.10: Analysis of the LMN (state space notation with real eigensystems and common eigenvectors) with no offset terms for the slow muscle. Training MSSE $= 0.36 \times 10^{-3}$, Test MSSE $= 0.48 \times 10^{-3}$. The top plot shows the activations of the validity functions, $\rho_i$. In the middle plot the steady state gain, $G$, of the interpolated model is depicted. In the bottom plot the values of the two poles of the interpolated model are shown.

interval [30 ... 200]ms. The individual values are shown in Table 6.2.

The control results for step-like command signals of different amplitude are shown in Figure 6.11. For comparison, the response to the generated stimulation patterns of the LMN with offset terms is included in this figure.

Note that the outputs of the design model (which is used in the closed loop) and of the LMN with offset terms (which was used in place of the real muscle outside the loop, cf. Figure 6.6(b)) are very similar.

### 6.5.4 Conclusions

For both muscles, the output of the model follows the command signal, i.e., the average steady state control error becomes zero. The characteristics of the responses are consistent for command signals of different amplitude.

| # unit | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| $t_r[ms]$ | 30 | 72.5 | 115 | 157.5 | 200 |
| $p_1$ | −180.2 | −79.0 | −50.5 | −37.2 | −29.4 |
| $p_2$ | −55.6 | −23.4 | −14.8 | −10.8 | −8.6 |
| $K_{x,1}$ | 10.3 | 13.3 | 5.8 | 3.2 | 3.5 |
| $K_{x,2}$ | 1.4 | 1.1 | 1.0 | 1.0 | 1.4 |
| $K_I$ | 608.6 | 267.2 | 106.4 | 59.2 | 73.4 |

Table 6.2: Slow muscle: closed-loop rise times for the LCN design, dominant closed-loop poles and state feedback gains.

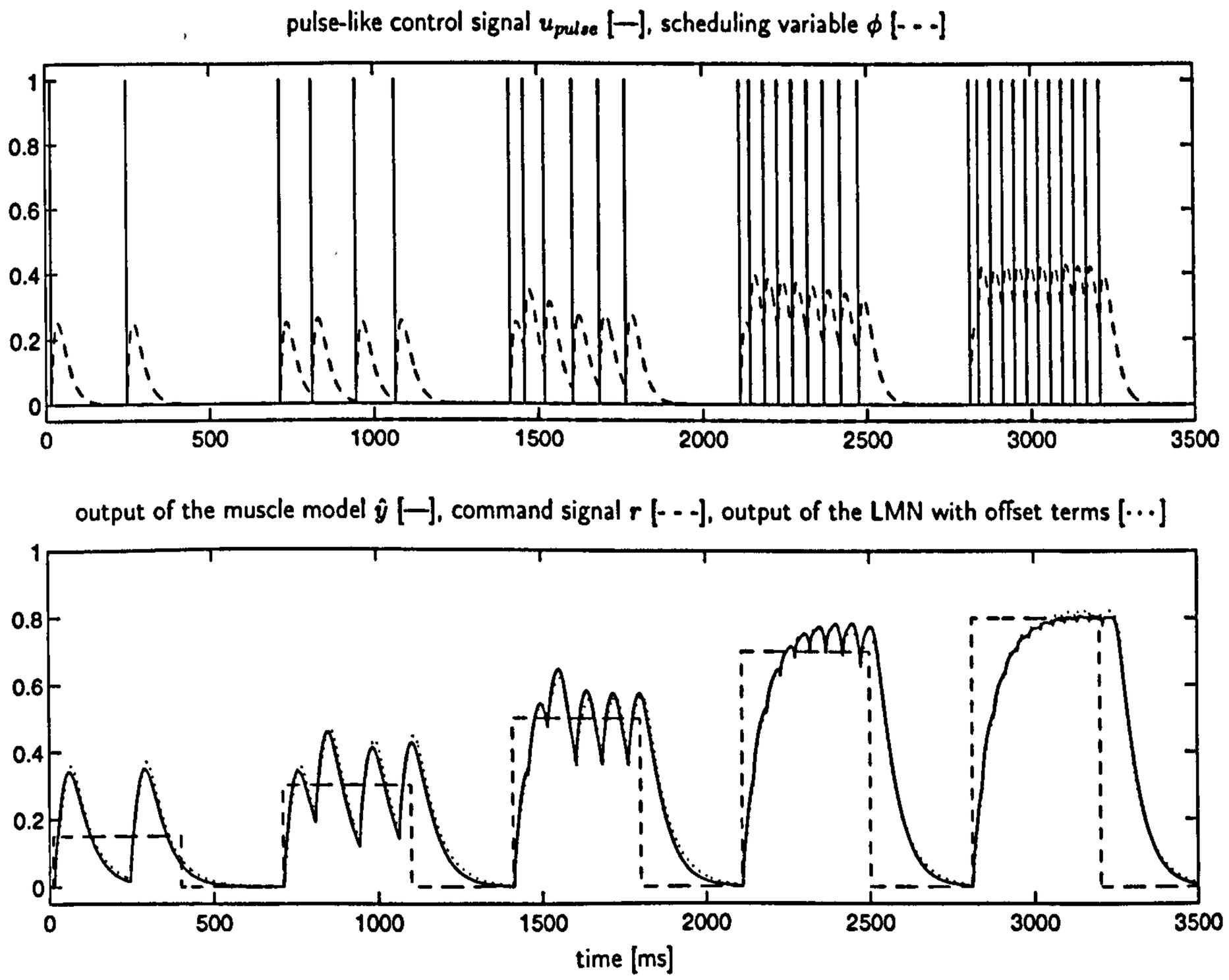



Figure 6.11: Generation of optimal stimulation patterns by non-linear control, slow muscle model.

For the fast muscle, three phases can be distinguished in the generated stimulation patterns:

a) The pattern starts with two or three closely spaced impulses ("doublets" or "triplets").

b) This is followed by a short period without stimulation.

c) After that, the pattern consists of relatively evenly spaced pulses, i.e., the stimulation frequency is almost constant.

Phase a) is not present for a small command signal (e.g., $r = 0.15$ in Figure 6.9), where a single pulse suffices to achieve the desired contraction. For larger command signals, the initial doublet or triplet (Phase a)) causes the plant to rapidly reach the desired output. After a stimulation pause (Phase b)), the contraction is retained by stimulation with constant frequency (Phase c)).

The stimulation pause in Phase b) indicates that the controller aims at maintaining the desired contraction by delivering the minimal number of pulses. It causes, however, the plant output to drop below the desired value before the following constant frequency stimulation ensures that the output recovers in Phase c). If this drop is not desired, it can be avoided by selecting the controller design parameters differently. The generated stimulation patterns will then contain more pulses and may therefore only be sub-optimal.

For the slow muscle, the stimulation pulses are almost evenly spaced for a constant command signal. Only the initial IPI is smaller than the following IPIs; it is, however, significantly larger than the IPIs in Phase a) for the fast muscle and therefore does not represent a "doublet".

The characteristics of the stimulation patterns which were generated by the technique described in this section are similar to the properties of the optimal stimulation patterns obtained in Sections 6.2 to 6.4. Although equation (6.4) was not formally minimised over the set of all possible control design parameters, it can be concluded that the non-linear control approach is suitable for the generation of stimulation patterns specific for a desired contraction which have properties similar to those of optimal stimulation patterns.

The control approach employs a simulated closed-loop controller as part of a feedforward control technique. By using a simulated closed-loop which consists of the non-linear controller, the pre-processing of the control signal and a non-linear model of the plant no inverse of the muscle model is required. The dynamic properties of the controller can be defined by an appropriate selection of the closed-loop dynamics, and the non-linear influence of the pre-processing can be taken account of. Note that disturbances and modelling uncertainties cannot be rejected by this control approach as no feedback signal from the muscle is used.

Compared to the iterative method introduced in Section 6.4, this technique has the advantage that it is not computationally expensive: after the controller parameters are calculated (which is done off-line), the simulation of the closed-loop involves only the computation of the Local Model Network, augmented by the integrator, the state–feedback gain vector and the pre-processing algorithm. Thus, this technique is suitable for real-time applications with implantable micro-controllers.

# 7 Conclusions and Recommendations for Future Work

In this thesis, a novel approach to the modelling of electrically stimulated muscle under isometric conditions was developed. The modelling technique is empirical, and the structure of the model is controller orientated. It was shown how the design of muscle stimulation controllers can be based on the model structure presented.

## 7.1 Local Modelling and Control

The modelling and control techniques introduced in this work are based on a "divide and conquer" strategy. In the modelling approach, the operating space is decomposed into smaller sub-regions which are then described by local models of simple, possibly linear structure. The local models are blended together by a scheduler, and the resulting non-linear model is called a Local Model Network (LMN). The approach was put into context with similar strategies such as fuzzy logic, hierarchical modelling, etc. Aspects of the structure of the scheduler and of the local models were discussed. Local linear models in input–output and in state space form were considered. For the scheduler, we introduced normalised Gaussian bells and B–splines.

Methods for the estimation of parameters for dynamic systems were discussed, and parameter optimisation algorithms based on one-step-ahead prediction and on model simulation were compared. These algorithms were used for learning the structure of the Local Model Network, and for the estimation of the parameters of the local models.

The analysis and validation of the non-linear model structures were recognised as important aspects of the modelling process. It was shown that LMNs with linear local models can be interpreted as linear parameter-variant systems. Aspects of the stability of this class of systems were discussed, and a graphical phase plane interpretation of linear parameter-variant systems with real eigensystems was presented. It was shown that stability analysis for such systems becomes straightforward when all local models share a common set of eigenvectors.

Static analysis of the properties of the interpolated Local Model Network can provide further insight into the properties of the model. It was shown how information gained by such

analysis can be used to validate the model analytically by relating the model characteristics to known properties of the real system.

The Local Model Network approach results in a model structure which is controller orientated: linear techniques can be used to design a controller for each local linear model of the LMN. The local controllers can then be interpolated using the same scheduler as in the LMN. This approach, which was termed a Local Controller Network approach, was reviewed and put into context with gain scheduling methods of control. Linear controller design techniques which are based on pole-placement were introduced for LMNs with local models in input–output and in state space form. The design of Local Controller Networks based on these linear techniques was demonstrated.

## 7.2 Modelling and Control of Electrically Stimulated Muscle

The Local Model Network introduced in the first part of the thesis was applied to the modelling of the contraction of electrically stimulated muscle under isometric conditions. Data from experiments with two rabbit *tibialis anterior* muscles were used. The characteristics of the two muscles were very different owing to the fact that one has a majority of fast motor units whereas the other has mainly slow motor units. The muscles were stimulated with supramaximal stimulation pulses, and the force of the isometric muscle contraction was recorded. The muscle activation was varied by changing the stimulation frequency. Non-linear characteristics due to variations of the stimulation frequency such as the "catch-like" effect could be observed.

Data analysis gave first estimates for model structure parameters such as the dynamic order, the time delay and time constant of the system. A linear modelling approach was used to assess the degree of non-linearity of the system and to obtain initial guidelines for the selection of the non-linear model structure. Local Model Networks of different sizes and structures were then identified and the modelling results compared and analysed. LMNs with local linear models in state space form with the pre-processed input as the scheduling variable were found to perform best for both muscles. Using *a priori* knowledge about the properties of muscle, the local models were restricted to be real eigensystems. This restriction did not lead to a deterioration of the performance of the models. A further limitation to common eigenvectors for all local models of the network ensured that the global LMN is stable, provided that all local linear models are stable.

A detailed analysis and validation of the models was carried out:

- A very good match between the simulated model output and the output of the real muscles was found. Non-linear effects such as the "catch-like" effect (where present) could be approximated. Thus, the models were found to perform very well on experimental data similar to those used to identify the model parameters.

- The force–frequency relationships were obtained for the models. They show that the model characteristics diverge from the properties expected in the real muscles for stimulation patterns involving long bursts of high frequency. Such data were not present in the experimental data.

- Static analysis shows that the model parameters can be related to properties of real muscles.

- For one LMN, the analysis showed that one local model was unstable. This did not influence the model performance with the experimental data used to identify the model parameters as the unstable local model was activated for short periods of time only. However, the instability could be detected when the model was stimulated with an input pattern of a constant frequency which activated the unstable local model. A method was developed to modify the unstable local model in such a way that the global model is stabilised without deteriorating the overall model performance.

The modelling approach is empirical in the sense that only input–output data from the experiments were used to identify the model. No explicit knowledge about the physiological properties and characteristics of muscle was used. However, the modelling technique differs from a pure black–box approach in that *a priori* knowledge was used for the selection of the scheduling variable, the choice of the structure of the local models, and for the analytical validation of the models obtained. The approach can therefore be termed a "grey–box" technique.

The model structure can be implemented easily and the simulation is not computationally expensive. It is therefore suitable for real-time applications with implanted stimulation devices.

It was shown how the non-linear muscle models can be used to obtain stimulation patterns which are optimal in the sense that they deliver the smallest number of pulses for a desired force output. Such patterns are thought to minimise muscle fatigue and to have a positive influence on the long term changes of the muscle properties which take place due to artificial stimulation. Results from experiments with real muscle were available for the first three methods investigated. The good match between the experimental results and the simulation results assured in the validity of the models for a wide range of operating conditions. Mismatches between experiments and simulation could be observed for some operating conditions, in particular when the systems were stimulated with long burst of high stimulation frequency. This indicates the limitations of the models.

A closed loop control approach was developed which can be used to obtain stimulation patterns specific for a desired contraction which have similar properties to optimal stimulation patterns. Local Controller Network structures whose design was based on the LMN models of the muscles were used as non-linear controllers. The muscles were simulated by Local Model

Networks. This method was found to be a constructive technique to generate stimulation patterns for real-time application.

## 7.3 Outlook

The modelling approach presented can be extended in a number of ways:

- For stimulation with sub-maximal input pulses a model of motor unit recruitment needs to be added. This can be either a separate recruitment model, or an additional scheduling dimension in the Local Model Network structure.

- The approach presented here can be used to model muscle activation as part of a Hill-type model structure for non-isometric muscle contraction. Initial experiments with such a setup gave promising results.

- In addition to the activation model, Local Model Networks can be used in a non-isometric model to approximate the non-linear force–length and force–velocity characteristics.

- The experiments to collect the data need to be optimised to cover all operating regions of interest. This will help to avoid model mismatch for some important operating conditions. In particular, data obtained under conditions of constant stimulation should be included in the experiment design. The importance of the data collection experiments will increase for modelling of more general stimulation conditions.

The non-linear control algorithm for the generation of optimal stimulation patterns requires experimental validation. Based on the results of these experiments the technique can be analysed and developed further.

# Appendix

# A Linearisation

In this appendix, linearisations of non-linear systems are introduced which are based on the expansion of the system into a Taylor series around some operating point. Note that the operating point is not required to be an equilibrium point. For further discussion see (Hunt and Johansen 1997).

## A.1 Linearisation of State Space Model

We consider the following time-invariant non-linear system in state space form,

$$\dot{\underline{x}}(t) = f(\underline{x}(t), u(t - T_d)); \quad \underline{x}(0) = \underline{x}_0 \tag{A.1a}$$

$$y(t) = g(\underline{x}(t)). \tag{A.1b}$$

where $f()$ and $g()$ are non-linear functions, $f()$ is continuous differentiable. For simplicity we restrict ourselves to single–input single–output system, i.e. $u \in \mathbb{R}$ and $y \in \mathbb{R}$ are the input and the output of the system, respectively. The dimensionality of the state vector $\underline{x} \in \mathbb{R}^n$ defines the dynamic order of the system, and $\dot{\underline{x}}(t)$ denotes the derivative of the state with respect to time, $d\underline{x}/dt$. The scalar $T_d \in \mathbb{R}$ is a time-delay, and the continuous time is denoted by $t$.

### A.1.1 Taylor Series Expansion

Equations (A.1) can be linearised around an operating point $[u^o, \underline{x}^o, y^o]$. The Taylor series expansion results in

$$\dot{\underline{x}}(t) = f(\underline{x}^o, u^o) + \left.\frac{\partial f}{\partial \underline{x}}\right|_{\underline{x}^o, u^o} \underline{x}^d(t) + \left.\frac{\partial f}{\partial u}\right|_{\underline{x}^o, u^o} u^d(t - T_d) + \text{h.o.t.}; \quad \underline{x}(0) = \underline{x}_0 \tag{A.2a}$$

$$y(t) = g(\underline{x}^o) + \left.\frac{\partial g}{\partial \underline{x}}\right|_{x^o} \underline{x}^d(t) + \text{h.o.t.}, \tag{A.2b}$$

where h.o.t. denotes higher order terms which will be ignored. The deviation variables are defined as

$$\begin{aligned} \underline{x}^d(t) &= \underline{x}(t) - \underline{x}^o \\ \dot{\underline{x}}^d(t) &= \dot{\underline{x}}(t) - f(\underline{x}^o, u^o) \\ u^d(t - T_d) &= u(t - T_d) - u^o \\ y^d(t) &= y(t) - y^o \,. \end{aligned} \tag{A.3}$$

When introducing the constants

$$\boldsymbol{A} = \left.\frac{\partial f}{\partial \underline{x}}\right|_{\underline{x}^o, u^o} \in \mathbb{R}^{n \times n}; \qquad \underline{b} = \left.\frac{\partial f}{\partial u}\right|_{\underline{x}^o, u^o} \in \mathbb{R}^n; \qquad \underline{c} = \left.\frac{\partial g}{\partial \underline{x}}\right|_{x^o} \in \mathbb{R}^n \,, \tag{A.4}$$

equation (A.2) can be rewritten as

$$\dot{\underline{x}}^d(t) = \boldsymbol{A}\underline{x}^d(t) + \underline{b}u^d(t - T_d); \quad \underline{x}^d(0) = \underline{x}_0 - \underline{x}^o \tag{A.5a}$$

$$y^d(t) = \underline{c}^T \underline{x}^d(t) \,. \tag{A.5b}$$

The state equation (A.5a) can be reformulated in terms of the original variables $\underline{x}$ and $u$,

$$\dot{\underline{x}}(t) = \boldsymbol{A}\,\underline{x}(t) + \underline{b}\,u(t - T_d) + [f(\underline{x}^o, u^o) - \boldsymbol{A}\,\underline{x}^o - \underline{b}\,u^o]\,; \qquad \underline{x}(0) = \underline{x}_0 \,. \tag{A.6a}$$

Similarly, the output equation (A.5b) can be rewritten in terms of the original output $y$,

$$y(t) = \underline{c}^T\,\underline{x}(t) + [y^o - c^T\,\underline{x}^o] \,. \tag{A.6b}$$

With the offset terms

$$\underline{d}^x = f(\underline{x}^o, u^o) - \boldsymbol{A}\,\underline{x}^o - \underline{b}\,u^o \qquad \in \mathbb{R}^n \tag{A.7a}$$

$$d^y = y^o - c^T\,\underline{x}^o \qquad \in \mathbb{R}, \tag{A.7b}$$

system (A.6) becomes

$$\dot{\underline{x}}(t) = \boldsymbol{A}\,\underline{x}(t) + \underline{b}\,u(t - T_d) + \underline{d}^x\,; \qquad \underline{x}(0) = \underline{x}_0 \tag{A.8a}$$

$$y(t) = \underline{c}^T\,\underline{x}(t) + d^y \,. \tag{A.8b}$$

The system (A.8) is depicted in Figure (A.1).

### A.1.2 Linear Transfer Function

The Laplace transformed system corresponding to (A.8) has the form

$$\underline{X}(s) = (s\boldsymbol{I} - \boldsymbol{A})^{-1}\,\underline{x}_0 + (s\boldsymbol{I} - \boldsymbol{A})^{-1}\left(\underline{b}U(s) + \underline{d}^x\,\frac{1}{s}\right) \tag{A.9a}$$

$$Y(s) = \underline{c}^T\,\underline{X}(s) + d^y\,\frac{1}{s} \,. \tag{A.9b}$$

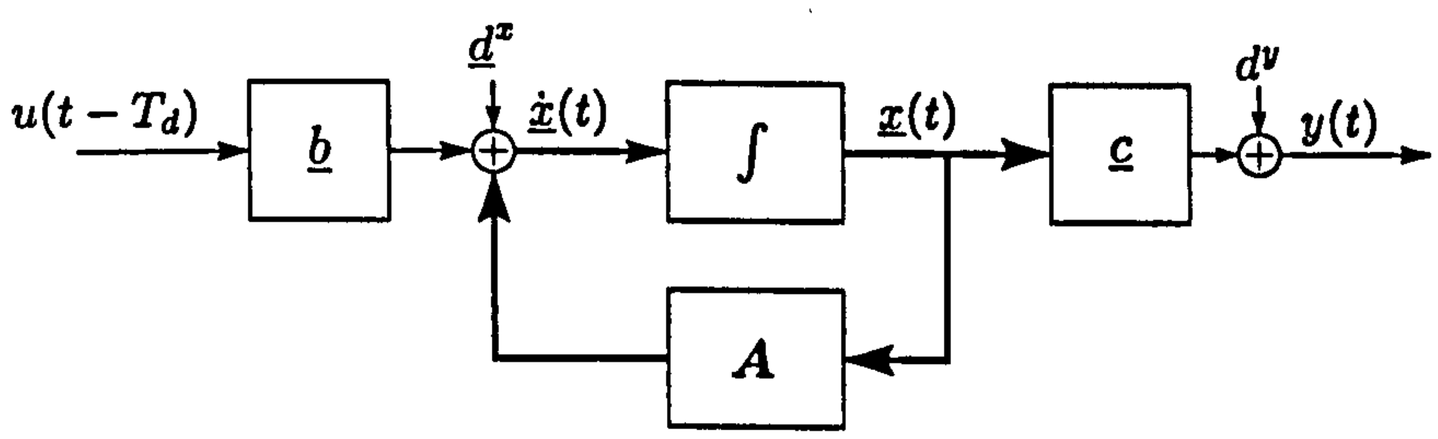



Figure A.1: Linear state space description.

For zero initial condition, equations (A.9a) and (A.9b) can be combined to

$$Y(s) = \underline{c}^T \, (sI - A)^{-1} \left( \underline{b} U(s) + \underline{d}^x \frac{1}{s} \right) + d^y \frac{1}{s} . \tag{A.10}$$

We can now substitute

$$\underline{c}^T \, (sI - A)^{-1} \, \underline{b} = \frac{\underline{c}^T \operatorname{Adj} (sI - A) \, \underline{b}}{\det (sI - A)} = \frac{b(s)}{a(s)} , \tag{A.11a}$$

and

$$\underline{c}^T \, (sI - A)^{-1} \, \underline{d}^x = \frac{\underline{c}^T \operatorname{Adj} (sI - A) \, \underline{d}^x}{\det (sI - A)} = \frac{d^x(s)}{a(s)} . \tag{A.11b}$$

When we introduce the transfer function (Kailath 1980) in terms of the polynomials $a(s)$ and $b(s)$,

$$H(s) = \frac{b(s)}{a(s)} , \tag{A.12a}$$

and combine the effects of the bias terms in $D(s)$,

$$D(s) = \frac{d^x(s) + a(s) \, d^y}{a(s)} \frac{1}{s} , \tag{A.12b}$$

equation (A.10) can be rewritten as

$$\begin{aligned} Y(s) &= H(s) \, U(s) + D(s) \\ &= \frac{b(s)}{a(s)} U(s) + D(s) . \end{aligned} \tag{A.13}$$

When the bias terms, $\underline{d}^x$ and $\mathrm{d}^y$, are zero, $D(s)$ vanishes and we obtain the classical transfer function description of a linear state-space system. The representation $\{b(s), a(s)\}$ can be called a polynomial (fractional) representation of the state space realisation $\{A, \underline{b}, \underline{c}\}$. A polynomial representation of a system can have an unlimited number of state space realisations (Kailath 1980), and the state space realisation can be regarded as the more general description. (Note that transfer function and state space descriptions are not necessarily equivalent for linear parameter varying systems.) The polynomial representation (or transfer function) can be directly obtained from its state space realisation (A.8a)–(A.8b) if the system

is in observer canonical form (Franklin *et al.* 1994). The elements of the polynomials $b(s)$ and $a(s)$ are then related to $\{A, \underline{b}, \underline{c}\}$ as follows,

$$A = \begin{bmatrix} -a_1 & 1 & 0 & \cdots & 0 \\ -a_2 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ -a_n & 0 & 0 & \cdots & 0 \end{bmatrix}, \quad \underline{b} = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{bmatrix}, \quad \underline{c}^T = [1, 0, \ldots, 0] \tag{A.14}$$

### A.1.3 Steady State Gain and Dynamics

To investigate the steady state value of the step response, we apply an input to the system which is a step with amplitude $u$, $U(s) = u\frac{1}{s}$. Equation (A.10) becomes now,

$$Y(s) = \underline{c}^T (sI - A)^{-1} \left( \underline{b}\, u \frac{1}{s} + \underline{d}^x \frac{1}{s} \right) + d^y \frac{1}{s} . \tag{A.15}$$

Using the final value theorem for the Laplace transform, which says that provided that all poles of $sY(s)$ are in the left half-plane,

$$\lim_{t \to \infty} y(t) = \lim_{s \to 0} sY(s) , \tag{A.16}$$

the steady state value for (A.15) becomes

$$\lim_{t \to \infty} y(t, u) = y_{\infty|u} = -\underline{c}^T A^{-1} (\underline{b}\, u + \underline{d}^x) + d^y . \tag{A.17}$$

Thus, the steady state gain can be defined as

$$G = \frac{y_{\infty|u_2} - y_{\infty|u_1}}{u_2 - u_1} = -\underline{c}^T A^{-1} \underline{b}, \tag{A.18}$$

where $u_2$ and $u_1$ are close in value.

The dynamics of the system are determined by the eigenvalues of the state feedback matrix $A$,

$$p = \mathrm{eig} A . \tag{A.19}$$

These eigenvalues are also the poles of the characteristic polynomial, $\det(sI - A)$.

## A.2 Linearisation of NARX Model

We consider the general description of the input–output characteristics of a non-linear system as a NARX model, cf. Section 2.2.2,

$$y(t_k) = f(\underline{\psi}(t_k)) + e(t_k) . \tag{A.20}$$

with the data vector

$$\underline{\psi}(t_k) = [u(t_{k-d}), \dots u(t_{k-d-n_u}), y(t_{k-1}), \dots y(t_{k-n_y})]^T \in \mathbb{U}^{n_u+1} \times \mathbb{Y}^{n_y} \subset \mathbb{R}^{n_u+n_y+1} . \tag{A.21}$$

Here, $t_k$ expresses the $k$-th discrete time step, $t_k = k T_s$ with $T_s$ as the sampling period, and the index $_d$ denotes the input delay, $T_d = dTs$.

### A.2.1 Taylor Series Expansion

The classical approach to find a linear model is based on the expansion of this general NARX description into a Taylor series around some operating point $[u^o, y^o]$. For small deviations around this point higher order terms can be ignored and we obtain a linear approximation,

$$\begin{aligned} y(t_k) = f(\underline{\psi}^o) + \sum_{j=0}^{n_u} \left.\frac{\partial f}{\partial u(t_{k-d-j})}\right|_{u^o,y^o} (u(t_{k-d-j}) - u^o) + \\ \sum_{i=1}^{n_y} \left.\frac{\partial f}{\partial y(t_{k-i})}\right|_{u^o,y^o} (y(t_{k-i}) - y^o) + e(t_k) . \end{aligned} \tag{A.22}$$

When we define the deviation variables as

$$\begin{aligned} y^d(t_k) &= y(t_k) - f(\underline{\psi}^o) = y(t_k) - y^o , \\ u^d(t_k) &= u(t_k) - u^o , \\ \text{and} \quad \underline{\psi}^d(t_k) &= \underline{\psi}(t_k) - \underline{\psi}^o , \\ \text{where} \quad \underline{\psi}^o &= [u^o, \dots u^o, y^o, \dots y^o]^T , \end{aligned} \tag{A.23}$$

and introduce the constants

$$a_i = \left.\frac{-\partial f}{\partial y(t_{k-i})}\right|_{u^o,y^o} , \qquad i = 1, \dots n_y \tag{A.24a}$$

$$b_j = \left.\frac{\partial f}{\partial u(t_{k-d-j})}\right|_{u^o,y^o} , \qquad j = 0, \dots n_u \tag{A.24b}$$

equation (A.27) can be rewritten as

$$y^d(t_k) = -a_1 y^d(t_{k-1}) - \dots - a_{n_y} y^d(t_{k-n_y}) + b_0 u^d(t_{k-d}) + \dots + b_{n_u} u^d(t_{k-d-n_u}) + e(t_k) . \tag{A.25}$$

Introducing the parameter vector $\underline{\Theta} \in \mathbb{R}^{n_u+n_y+1}$,

$$\begin{aligned} \underline{\Theta} &= [\Theta_0, \dots \Theta_{n_u}, \Theta_{n_u+1}, \dots \Theta_{n_u+n_y}]^T \\ &= [b_0, \dots b_{n_u}, -a_1, \dots - a_{n_y}]^T , \end{aligned} \tag{A.26}$$

equation (A.25) becomes,

$$y^d(t_k) = \underline{\psi}^{dT}(t_k)\underline{\Theta} + e(t_k) . \tag{A.27}$$

An equivalent model expressed in terms of the absolute input-output signals $u$ and $y$ can be obtained when substituting (A.23) in equation (A.27). This results in

$$y(t_k) = \underline{\psi}^T(t_k)\,\underline{\Theta} + d^y + e(t_k) = \underline{\psi}'^T(t_k)\,\underline{\Theta}' + e(t_k)\,, \tag{A.28}$$

where the offset term $d^y$ is expressed in terms of the operating point as

$$d^y = -\sum_{j=0}^{n_u} b_j u^o - \sum_{i=1}^{n_y} a_i y^o \tag{A.29}$$

and

$$\underline{\psi}'(t_k) = \left[u(t_{k-d}), \ldots u(t_{k-d-n_u}), y(t_{k-1}), \ldots y(t_{k-n_y}), 1\right]^T\,, \tag{A.30a}$$

$$\underline{\Theta}' = \left[b_0, \ldots b_{n_u}, -a_1, \cdots - a_{n_y}, d^y\right]^T\,. \tag{A.30b}$$

Often, it can be assumed that the number of delayed inputs corresponds to the number of delayed outputs,

$$n = n_u + 1 = n_y\,, \tag{A.31}$$

which simplifies the data vector (2.18) to

$$\underline{\psi}_{d,n}(t_k) = \left[u(t_{k-d}), \ldots u(t_{k-d-n+1}),\, y(t_{k-1}), \ldots y(t_{k-n})\right]^T\,. \tag{A.32}$$

Using data vector (A.32) the linear ARX structure is defined by only two design parameters, $n$ and $d$.

### A.2.2 Linear Transfer Function

Defining the polynomials $A$ and $B$ in $q^{-1}$ as

$$A(q^{-1}) = 1 + a_1 q^{-1} + \cdots + a_{n_y} q^{-n_y} \tag{A.33a}$$

$$B(q^{-1}) = b_0 + b_1 q^{-1} + \cdots + b_{n_u} q^{-n_u}\,, \tag{A.33b}$$

equation (A.27) can be rewritten as a transfer function,

$$A(q^{-1})\,y^d(t_k) = q^{-d}\,B(q^{-1})\,u^d(t_k) + e(t_k)\,. \tag{A.34}$$

Here $q^{-1}$ is the time shift operator, i.e. $q^{-n}\,y(t_k) = y(t_{k-n})$.

Describing an ARX model in the form of a transfer function (A.34) facilitates the analysis of the system, e.g. statements about its stability can be given by observing the location of the poles and zeros, and techniques for controller design are often based on this form.

An equivalent model expressed in terms of the absolute input-output signals $u$ and $y$ can be obtained when applying substitutions (A.23) to equation (A.34). This results in

$$A(q^{-1})\,y(t_k) = q^{-d}\,B(q^{-1})\,u(t_k) + d^y + e(t_k)\,, \tag{A.35}$$

where the offset term $d^y$ is expressed in terms of the operating point as

$$d^y = A(1)\,y^o - B(1)\,u^o\,. \tag{A.36}$$

### A.2.3 Steady State Gain and Dynamics

From equations (A.25) and (A.33), the steady state response of the ARX system (A.34) to a step input with amplitude $u$ can be derived as

$$\lim_{k\to\infty} y(t_k, u) = y_{\infty|u} = \frac{B(1)\,u + d^y}{A(1)}\,. \tag{A.37}$$

Thus, the steady state gain can be formulated as

$$G = \frac{y_{\infty|u_2} - y_{\infty|u_1}}{u_2 - u_1} = \frac{B(1)}{A(1)}\,, \tag{A.38}$$

where $u_2$ and $u_1$ are close in value.

The dynamics of the system are determined by the poles of the polynomial $A(q^{-1})$.

# B Parameter Optimisation

The objective of parameter optimisation is to find the parameter vector $\underline{\theta} \in \mathbb{R}^n$ of a model which minimises a chosen measure of the output error. The output error is defined as

$$e_i(\underline{\theta}) = y_i - \hat{y}_i(\underline{\theta}) \,, \tag{B.1}$$

where $\{\hat{y}_i\}_{i=1}^N$ is the model output sequence, and $\{y_i\}_{i=1}^N$ is the desired model response sequence; $N$ denotes the number of data points. The weighted sum of the squared output error will be used in the formulation of the optimisation criterion,

$$J(\underline{\theta}) = f\left[e_i(\underline{\theta})\right] = \sum_{i=1}^{N} \gamma_i \left[y_i - \hat{y}_i(\underline{\theta})\right]^2 \,, \tag{B.2}$$

where $\gamma_i$ are the weighting factors for each data point, and $N$ is the number of data points. Criterion (B.2) depends only on the model parameter vector $\underline{\theta}$. Thus, the optimal parameter vector $\underline{\theta}^*$ in the sense of equation (B.2) is the one which minimises $J(\underline{\theta})$,

$$\underline{\theta}^* = \arg\min_{\underline{\theta}} J(\underline{\theta}) \,. \tag{B.3}$$

Depending on the characteristics of the function $\hat{y}(\underline{\theta})$, different techniques are available to minimise $J(\underline{\theta})$. In Section B.2, we will consider the case when the dependency of $\hat{y}$ in $J(\underline{\theta})$ on the parameters $\underline{\theta}$ is linear. In Section B.3 we will present techniques to solve (B.3) when $\hat{y}$ depends on its parameter in a non-linear way.

Before approaches for linear and non-linear parameter estimation are introduced, we will discuss some general aspects of parameter optimisation for dynamic systems.

## B.1 Parameter Optimisation for Dynamic Systems

For dynamic systems the way in which the parameter vector influences the model output can depend on how $\hat{y}$ is calculated. For discrete time dynamic model structures, two modes of calculating $\hat{y}$ can be distinguished, depending on the way the feedback information is obtained:

- **Prediction model:** The model predicts the new system output (or state) based on past outputs (or states) obtained from the real system. For a general input–output

model structure, the model has the form,

$$\hat{y}(t_{k+1}) = f(y(t_i), u(t_{i-d+1}), i \leq k, t_k)\,. \tag{B.4}$$

This setup is illustrated in Figure B.1(a). For the discrete time state space description, the relevant state equation becomes

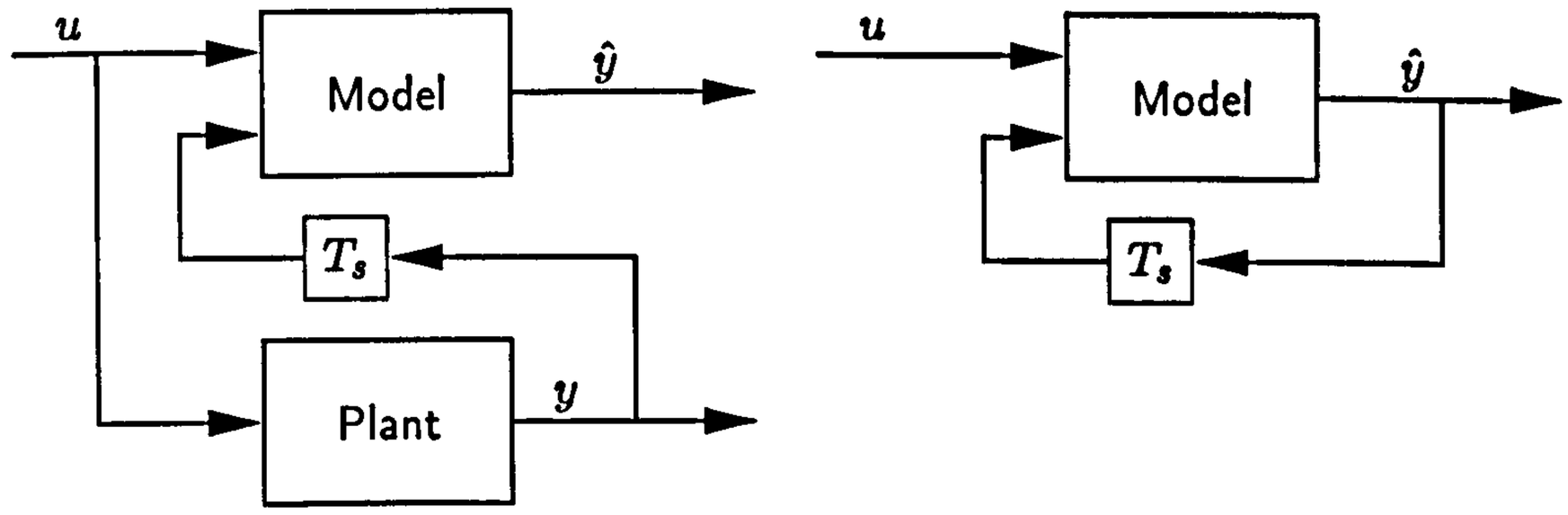
$$\hat{\underline{x}}(t_{k+1}) = h(\underline{x}(t_k), u(t_{k-d}))\,. \tag{B.5}$$

- **Simulation model:** The model predicts the new system output (or state) based on past outputs (or states) predicted by the model. In input–output form we obtain,

$$\hat{y}(t_{k+1}) = f(\hat{y}(t_i), u(t_{i-d+1}), i \leq k), t_k)\,. \tag{B.6}$$

This setup is illustrated in Figure B.1(b). The corresponding description for the discrete state space model has the form

$$\hat{\underline{x}}(t_{k+1}) = h(\hat{\underline{x}}(t_k), u(t_{k-d}))\,. \tag{B.7}$$




(a) Prediction model. (b) Simulation model.

Figure B.1: Calculation of the model output for input–output structures. $u$ is the input, $\hat{y}$ denotes the model output and $y$ is the output obtained from the real system. $T_s$ denotes a delay of one sampling interval.

The prediction model uses a one-step ahead prediction, whereas the simulation model works with an infinite prediction horizon. Intermediate setups which use $n$-step ahead prediction are possible. With respect to the identification of the parameters they are equivalent to the simulation model. Thus, we restrict our discussion to prediction and simulation models.

In order to use the prediction model mode with state space descriptions, the entire state of the real system must be available. This often restricts the use of this mode to input–output models, where measured past outputs are readily available. However, the correct choice of

the sampling period becomes crucial, as this mode will fail to represent important system dynamics when the sampling period is very short. The model then learns just to propagate the past measured value as the new prediction. For example, for the input-output structure, a model which just propagates the past measured output as the model predicted output, $\hat{y}(t_{i+1}) = y(t_i)$, can give a very small prediction error, as $y(t_{i+1}) \approx y(t_i)$ for $t_{i+1} - t_i$ very small.

The simulation model approach is less sensitive to the choice of the sampling period. It has the further advantage that for the state space model structure, no measurements of the states are necessary. Parameter optimisation with this mode is, however, often much more complex and expensive than with the prediction model.

In order to compare the performance of different models easily, a single measurement of the error is needed. Based on the optimisation criterion (B.2), the mean squared error (MSE) can be formulated[1]

$$MSE = \frac{1}{N}\sum_{i=1}^{N}[y(t_i) - \hat{y}(t_i)]^2 \tag{B.8}$$

where $N$ denotes the number of data points, $y$ is the desired output and $\hat{y}$ is the model output. The advantage of the $MSE$ compared to the sum of squared errors is that it can be compared for different numbers of data points. To distinguish the way in which the model output was obtained, the result of equation (B.8) will be called *mean squared prediction error* (MSPE) if $\hat{y}$ is the result of a prediction model, equations (B.4) and (B.5), or *mean squared simulation error* (MSSE) if the model output was obtained from a simulation model, equations (B.6) and (B.7).

Aspects of parameter identification with prediction and with simulation mode are illustrated in Example E.4.2 on page 152.

## B.2 Linear Least Squares Optimisation

In this section we consider $\hat{y}$ to be a linear function with respect to its parameters, $\underline{\theta}$,

$$\hat{y}_i = \underline{\psi}_i^T \underline{\theta}, \tag{B.9}$$

with $\underline{\psi} \in \mathbb{R}^n$ as the input vector.

We aim to solve equation (B.3), i.e. to find the global minimum of the function (B.2). A necessary condition for this is that its derivative with respect to the parameter vector vanishes,

$$\frac{\partial J(\underline{\theta})}{\partial \underline{\theta}} = 0 = 2\sum_{i=1}^{N}\gamma_i \underline{\psi}_i \left[y_i - \underline{\psi}_i^T \underline{\theta}^*\right], \tag{B.10}$$

[1]Note that in a strict statistical notation, equation (B.8) denotes the *average* squared error which converges to the *mean* squared error for $N \to \infty$. According to the notation common in the literature, we will, however, use the term *mean*.

or,

$$\left[\sum_{i=1}^{N} \gamma_i \underline{\psi}_i \underline{\psi}_i^T\right] \underline{\theta}^* = \sum_{i=1}^{N} \gamma_i \underline{\psi}_i y_i \,. \tag{B.11}$$

Equation (B.11) can be rewritten in terms of matrices and vectors as

$$\left[\Psi_\gamma^T \Psi_\gamma\right] \underline{\theta}^* = \Psi_\gamma^T \underline{y}_\gamma \,, \tag{B.12}$$

where

$$\begin{aligned} \underline{y}_\gamma &= [\sqrt{\gamma_1}\, y_1, \ldots, \sqrt{\gamma_N}\, y_N]^T && \in \mathbb{R}^N \\ \Psi_\gamma &= \begin{bmatrix} \sqrt{\gamma_1}\, \underline{\psi}_1^T \\ \vdots \\ \sqrt{\gamma_N}\, \underline{\psi}_N^T \end{bmatrix} && \in \mathbb{R}^{N \times n} \end{aligned} \tag{B.13}$$

The optimal parameter vector can now be calculated as

$$\underline{\theta}^* = \Psi_\gamma^+ \underline{y}_\gamma \,, \tag{B.14}$$

where the matrix $\Psi_\gamma^+ \in \mathbb{R}^{n \times N}$ denotes the Moore–Penrose pseudo inverse of $\Psi_\gamma$. It can be calculated as

$$\Psi_\gamma^+ = \left[\Psi_\gamma^T \Psi_\gamma\right]^{-1} \Psi_\gamma^T \,. \tag{B.15}$$

The direct computation of the pseudo inverse using the above equation is problematic when $\left[\Psi_\gamma^T \Psi_\gamma\right]$ is ill-composed. A robust way to calculate the pseudo inverse which avoids taking the inverse of the matrix $\left[\Psi_\gamma^T \Psi_\gamma\right]$ is to use Singular Value Decomposition (Press *et al.* 1992).

## B.3 Non-linear Least Squares Optimisation

A standard algorithm to solve (B.3) for non-linear systems is the Levenberg–Marquardt method (Marquardt 1963). It is an elegant way of switching smoothly between the (robust, but slow) steepest descent method and the (fast, but sensitive) Newton–Raphson algorithm (Unbehauen and Rao 1987). Gradient information from the Jacobian and the Hessian of (B.2) is used. The implementation presented here is adapted from (Press *et al.* 1992).

We first introduce the Jacobian and the Hessian of (B.2). We then discuss the Newton–Raphson algorithm, and finally present the Levenberg–Marquardt method.

### B.3.1 Jacobian and Hessian

One way to find a minimum of (B.2) is to expand the non-linear function $J(\underline{\theta})$ into a Taylor series around some nominal parameter vector $\underline{\theta}^o$, ignore higher order terms, and minimise the approximation of the original function.

The Taylor series expansion of equation (B.2) around the nominal parameter vector $\underline{\theta}^o \in \mathbb{R}^n$ has the form

$$J(\underline{\theta}) = J(\underline{\theta}^o) + \sum_{k=1}^{n} \left.\frac{\partial J}{\partial \theta_k}\right|_{\underline{\theta}^o} (\theta_k - \theta_k^o) + \frac{1}{2} \sum_{k=1}^{n}\sum_{l=1}^{n} \left.\frac{\partial^2 J}{\partial \theta_k \partial \theta_l}\right|_{\underline{\theta}^o} (\theta_k - \theta_k^o)(\theta_l - \theta_l^o) + \text{h.o.t.}\,. \tag{B.16}$$

When the higher order terms, which are denoted as h.o.t., are ignored, equation (B.16) can be approximated as a quadratic equation,

$$J(\underline{\theta}) \approx c + \underline{d}\,\Delta\underline{\theta} + \frac{1}{2}\,\Delta\underline{\theta}^T\, \boldsymbol{D}\, \Delta\underline{\theta}\,, \tag{B.17}$$

where

$$c \equiv J(\underline{\theta}^o)\,, \qquad \underline{d} \equiv \nabla J|_{\underline{\theta}^o}\,, \qquad D_{k,l} \equiv \left.\frac{\partial^2 J}{\partial \theta_k \partial \theta_l}\right|_{\underline{\theta}^o}\,, \qquad \Delta\underline{\theta} \equiv \underline{\theta} - \underline{\theta}^o \tag{B.18}$$

The gradient vector $\underline{d}$ is the Jacobian, and the matrix $\boldsymbol{D}$ denotes the Hessian of (B.2).

#### Calculation of the Jacobian and the Hessian

Considering $J$ in the form given by equation (B.2). The gradient vector, $\underline{d}$, then has the components

$$d_k = -2 \sum_{i=1}^{N} \gamma_i\, e_i(\underline{\theta}) \frac{\partial \hat{y}_i(\underline{\theta})}{\partial \theta_k}\,, \qquad k = 1 \ldots n\,. \tag{B.19}$$

The Hessian matrix, $\boldsymbol{D}$, can be obtained by taking an additional partial derivative,

$$D_{k,l} = 2 \sum_{i=1}^{N} \gamma_i \left[ \frac{\partial \hat{y}_i(\underline{\theta})}{\partial \theta_k} \frac{\partial \hat{y}_i(\underline{\theta})}{\partial \theta_l} - e_i(\underline{\theta}) \frac{\partial^2 \hat{y}_i(\underline{\theta})}{\partial \theta_l \partial \theta_k} \right]\,, \qquad k,l = 1 \ldots n\,. \tag{B.20}$$

Usually, it is assumed that the second derivative term in (B.20) can be ignored (Press *et al.* 1992), and the Hessian can be approximated by

$$D_{k,l} \approx \tilde{D}_{k,l} = 2 \sum_{i=1}^{N} \gamma_i \left[ \frac{\partial \hat{y}_i(\underline{\theta})}{\partial \theta_k} \frac{\partial \hat{y}_i(\underline{\theta})}{\partial \theta_l} \right]\,, \qquad k,l = 1 \ldots n\,. \tag{B.21}$$

To rewrite equations (B.19) and (B.21) in matrix form, we define a few vectors and matrices,

$$\begin{aligned}
&\underline{e}(\underline{\theta}) = [e_1(\underline{\theta}), \ldots, e_N(\underline{\theta})]^T && \in \mathbb{R}^N \\
&\boldsymbol{G} = \text{diag}\,(\gamma_1, \ldots, \gamma_N) && \in \mathbb{R}^{N \times N} \\
&\underline{\hat{y}}(\underline{\theta}) = [\hat{y}_1(\underline{\theta}), \ldots, \hat{y}_N(\underline{\theta})]^T && \in \mathbb{R}^N \\
&\nabla \underline{\hat{y}}(\underline{\theta})|_{\underline{\theta}^o} = \left[\nabla \hat{y}_1(\underline{\theta})|_{\underline{\theta}^o}, \ldots, \nabla \hat{y}_N(\underline{\theta})|_{\underline{\theta}^o}\right]^T && \in \mathbb{R}^{N \times n}
\end{aligned} \tag{B.22}$$

The gradient vector (B.19) becomes now

$$\underline{d} = -2\,\left[\nabla\underline{\hat{y}}(\underline{\theta})|_{\underline{\theta}^o}\right]^T \boldsymbol{G}\,\underline{e}\,, \tag{B.23}$$

and the simplified Hessian matrix (B.21) can be rewritten as

$$\tilde{\boldsymbol{D}} = 2\,\left[\nabla\underline{\hat{y}}(\underline{\theta})|_{\underline{\theta}^o}\right]^T \boldsymbol{G}\,\nabla\underline{\hat{y}}(\underline{\theta})|_{\underline{\theta}^o}\,. \tag{B.24}$$

### B.3.2 Finding the Minimum

#### Newton–Raphson method

The minimum of function (B.17) can be determined by setting its first derivative equal to zero,

$$\frac{\partial J(\underline{\theta})}{\partial \Delta\underline{\theta}} = 0 = \underline{d} + \boldsymbol{D}\,\Delta\underline{\theta}\,. \tag{B.25}$$

Thus, the optimal parameter vector could be found in one step, considering the initial parameter set $\underline{\theta}^o$, as

$$\underline{\theta}^* = \underline{\theta}^o - \boldsymbol{D}^{-1}\underline{d}. \tag{B.26}$$

For linear functions $J$, equation (B.26) describes the global minimum. For non-linear functions, however, the approximation in equation (B.17) is only valid for limited (typically small) values of $(\underline{\theta} - \underline{\theta}^o)$. Taking the full step towards the optimum in equation (B.26) may take us out of the region where the approximation is valid. It is then better to take only a small step in the direction given by (B.26), and to repeat this procedure iteratively.

Replacing the gradient vector by (B.23), and approximating the Hessian by (B.24) in equation (B.26), and introducing a small positive step size $h$ results in the Newton–Raphson algorithm (Unbehauen and Rao 1987), where the parameter vector is iteratively updated as follows,

$$\begin{aligned}\underline{\hat{\theta}}_{m+1} &= \underline{\hat{\theta}}_m - h\,\tilde{\boldsymbol{D}}^{-1}\underline{d} \\ &= \underline{\hat{\theta}}_m + h\,\left\{\left[\nabla\underline{\hat{y}}(\underline{\theta})|_{\underline{\hat{\theta}}_m}\right]^T \boldsymbol{G}\,\nabla\underline{\hat{y}}(\underline{\theta})|_{\underline{\hat{\theta}}_m}\right\}^{-1} \left[\nabla\underline{\hat{y}}(\underline{\theta})|_{\underline{\hat{\theta}}_m}\right]^T \boldsymbol{G}\,\underline{e}(\underline{\hat{\theta}}_m)\,.\end{aligned} \tag{B.27}$$

For each iteration, the model of the system must be simulated to evaluate the cost function (B.2) and to estimate the sensitivity function $\nabla\underline{\hat{y}}(\underline{\theta})|_{\underline{\theta}^o}$ in (B.23) and (B.24) which is calculated for each parameter using finite differences.

For linear systems, the Newton–Raphson algorithm always converges to the optimal parameter vector $\underline{\theta}^*$. For non-linear systems, the convergence of this iterative process depends on the degree of non-linearity of the system. For many non-linear systems the straightforward Newton–Raphson algorithm overshoots and tends to become unstable. Various modifications exist which try to eliminate this problem, see (Beck and Arnold 1977) for examples. The

basic problem is, however, that the Newton–Raphson algorithm depends on the approximation (B.17) which might be only valid near the optimum. The problem remains that this algorithm can perform poorly far away from the region where (B.17) holds.

### Levenberg-Marquardt Method

The Levenberg-Marquardt method (Marquardt 1963) provides an elegant technique to overcome the problems encountered with the Newton–Raphson algorithm by switching smoothly between the steepest descent method and Newton–Raphson.

The steepest descent method is an optimisation technique which is robust even when (B.17) is a very poor local approximation, as it follows the gradient,

$$\underline{\hat{\theta}}_{m+1} = \underline{\hat{\theta}}_m - h_s \, \underline{d}, \tag{B.28}$$

where the constant $h_s$ is small enough to exhaust the downhill direction. The steepest descent method does not converge very fast, especially close to the optimum. This can make it computational very expensive if it is used exclusively to minimise the criterion (B.2).

Smooth switching between the steepest descent method (B.28) and Newton-Raphson (B.26) is a way to combine the advantages of both techniques. It can be achieved by multiplying the diagonal elements of the Hessian matrix $\tilde{D}$ by a positive factor $(1+\lambda)$,

$$\tilde{D}'_{k,l} = \begin{cases} (1+\lambda)\,\tilde{D}_{k,l} & \text{for} \quad k = l \\ \tilde{D}_{k,l} & \text{for} \quad k \neq l \end{cases} \qquad k,l = 1 \ldots n \tag{B.29}$$

This modified Hessian is now used to update the parameter vector,

$$\underline{\hat{\theta}}_{m+1} = \underline{\hat{\theta}}_m - [\tilde{D}']^{-1} \, \underline{d} \, . \tag{B.30}$$

For small values of $\lambda$, this is equivalent to equation (B.27) with a step size of $h = 1$. For large $\lambda$, the matrix $\tilde{D}'$ is diagonally dominant, and equation (B.30) becomes equivalent to the steepest descent method (B.28).

The factor $\lambda$ is adapted in the following way:

- If $J(\underline{\theta}_{m+1}) \geq J(\underline{\theta}_m)$ then the new parameter vector is rejected, and $\lambda$ is increased by a factor of 10 (i.e., the optimisation behaves more like steepest descent).

- If $J(\underline{\theta}_{m+1}) < J(\underline{\theta}_m)$ then the new parameter vector is accepted, and $\lambda$ is decreased by a factor of 10 (i.e., the optimisation behaves more like Newton–Raphson).

The Levenberg-Marquardt method is a standard tool for non-linear parameter optimisation, and it was found to perform very robust for the parameter optimisations carried out in this work.

# C Linear Controller Design

In this appendix linear controller design techniques are introduced. These techniques can be used for the design of local controllers in LCN structures, cf. Chapter 3.

We will first discuss the design of a linear controller based on a model in input–output notation. In Section C.2, a controller design technique which is based on a linear model in state space form is introduced.

## C.1 Design based on Input–Output Models

We consider the linear plant in discrete-time[1] (shift operator) transfer function form as described by equation (A.35) on page 124,

$$A(q^{-1})y(t) = q^{-d}B(q^{-1})u(t) + d^y + e(t)\,, \tag{C.1}$$

where $A$ and $B$ are polynomials of degree $n$ in $q^{-1}$, $d^y$ is a constant offset and $e(t)$ denotes a white noise term. This description can be reformulated in $q$ by multiplying (C.1) with $q^{n+d}$ which results in

$$A'(q)y(t) = B'(q)u(t) + q^{n+d}d^y + q^{n+d}e(t)\,. \tag{C.2}$$

Here $A'$ and $B'$ are polynomials of degree $n+d$ in $q$.

Given a measurement of the plant output $y(t)$ and some command signal $r(t)$, the dynamic properties of the control loop can be determined by the feedback component $u(t)$. A general two-degrees-of-freedom structure can be defined as,

$$u(t) = \frac{1}{H(q)}\left[S(q)r(t) - G(q)y(t)\right]\,, \tag{C.3}$$

where the polynomials $G$, $H$ and $S$ have the forms

$$G(q) = g_0q^{n_g} + g_1q^{n_g-1} + \ldots + g_{n_g}\,, \tag{C.4a}$$

$$H(q) = q^{n_h} + h_1q^{n_h-1} + \ldots + h_{n_h}\,, \tag{C.4b}$$

$$S(q) = s_0q^{n_s} + s_1q^{n_s-1} + \ldots + s_{n_s}\,, \tag{C.4c}$$

[1]Note that $t$ denotes the discrete time throughout this section as we drop the index $k$ for convenience.

and these are to be determined in the control design.

Owing to the offset term $d^y$, the plant described by equation (C.2) does not have the form required by most linear design techniques. Thus, the offset has to be explicitly taken account of by the control strategy. A possible approach is to include integral action in the controller. The controller polynomials are now restricted to include an integral component,

$$G(q) = q\,G'(q) \tag{C.5a}$$

$$H(q) = (q-1)\,H'(q) \tag{C.5b}$$

$$S(q) = q\,S'(q) \tag{C.5c}$$

Integral action can also be desirable to compensate for a mismatch between the real plant and the model used for the controller design, and to enhance controller robustness. The overall control structure is shown in Figure C.1.

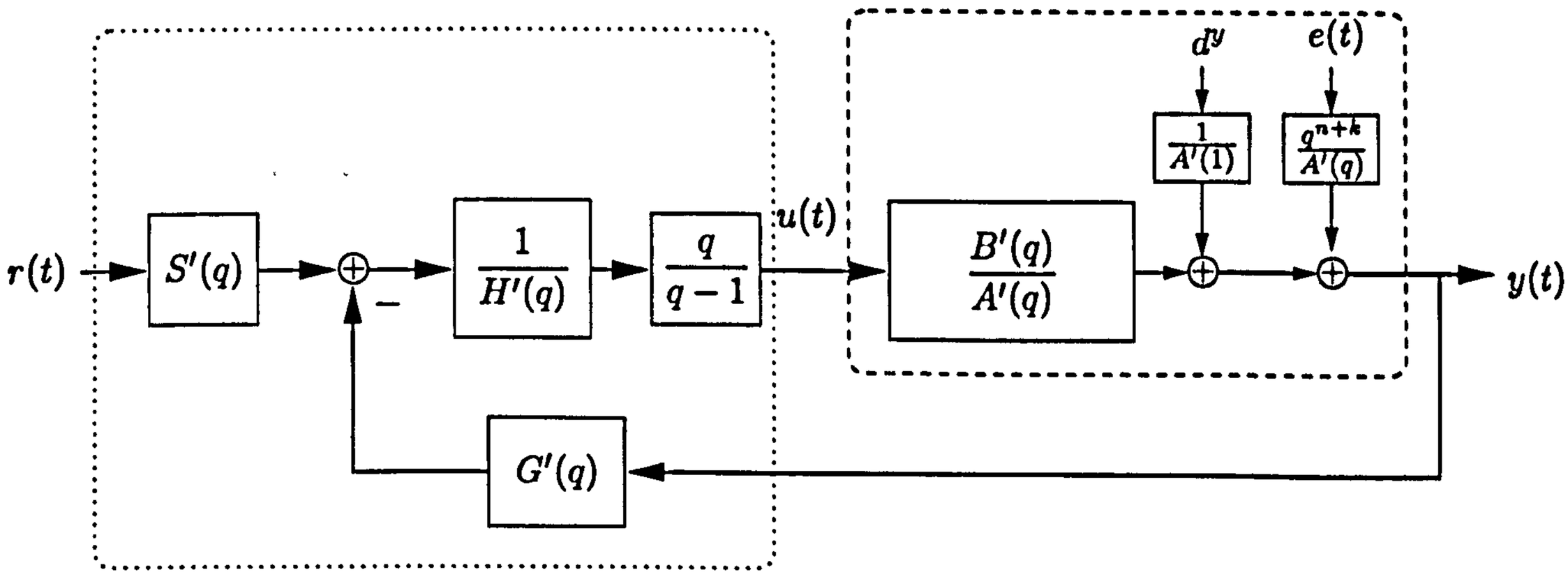



Figure C.1: Closed-loop control with integral action: single linear transfer function model. The dashed line marks the plant, the controller components are surrounded by the dotted line.

### C.1.1 Pole-placement Design

An approximate way to meet control specifications is to aim for a dominant pair of poles for the transfer function of the closed loop from $r$ to $y$. The specification consists of a desired nominal rise time, $t_r$, and a damping factor, $\xi$, for which the corresponding pair of poles can be directly set using standard time-domain formulae, see for instance (Franklin *et al.* 1994).

Combining equations (C.2)—(C.4) and compensating for the offset $d^y$ by (C.5), the closed loop characteristic polynomial can be easily found to be $A'H + B'G$. For a given desired closed loop characteristic $A_{cl} = A_o A_m$, the following equation is solved,

$$A'(q)H(q) + B'(q)G(q) = A_o(q)A_m(q)\,. \tag{C.6}$$

Here, $A_o$ corresponds to a desired observer polynomial and $A_m$ to the remaining desired closed loop poles which will be assigned to the dominant poles. The observer poles need to be faster than the poles of $A_m$ to obtain a dominant pair of poles. We choose to work with a dead-beat observer, i.e., the observer poles are set as fast as possible, $A_o = q^{n_{cl}-2}$.

In order to solve equation (C.6) for a given $A_m(q)$ of degree 2, the degree of the closed loop, $n_{cl}$, has to be equal to $2n_p - 1$, where $n_p$ is the degree of the system's denominator[2]. Hence, the degree of the controller polynomials must be chosen as $n_p - 1$, i.e. $n_h = n_g = n_s = n_p - 1$. Note that, if integral action is included, the integrator is considered to be part of the plant for the design, i.e. $n_p = n + d + 1$, and $n_h$, $n_s$ and $n_g$ refer to the degrees of the polynomials $H'$, $S'$ and $G'$, respectively. Without integral action, the degree of the system's denominator is $n_p = n + d$.

The $S$ polynomial in equation (C.3) is chosen to achieve a desired servo response, in particular zero steady state error,

$$S(q) = \frac{A_{cl}(1)}{B'(1)} q^{n_s} . \tag{C.7}$$

## C.2 State-space Design

We now consider a linear plant in state-space notation as described by equation (A.8) on page 120,

$$\dot{\underline{x}}(t) = A\,\underline{x}(t) + \underline{b}\,u(t - T_d) + \underline{d}^x \tag{C.8a}$$

$$y(t) = \underline{c}^T\,\underline{x}(t) + d^y . \tag{C.8b}$$

We choose to work with a description of the system in the continuous time domain for this section. The results obtained can be easily adapted for discrete time shift and delta operator notations by replacing the time derivative by the appropriate operator, cf. Appendix D. For simplicity, we further assume that the system has no input delay, $T_d = 0$. Extension to systems with input delay is straightforward when working in shift operator notation and was discussed in the previous section for controller design with input–output models.

A closed–loop control structure which is based on a state-space notation is shown in Figure C.2. The control law uses full state-feedback via the controller gain $\underline{K}$.

If only the output of the plant is available, an estimator is used to reconstruct the state using input–output data from the plant. If the full state of the plant is available, the estimator is not needed, and the real state $\underline{x}$ will be fed back instead of the estimate $\hat{\underline{x}}$.

We will first introduce a design method for the control law, assuming that the full state $\underline{x}$ is available. Then, design aspects of the estimator will be discussed.

[2]Note that due to the input delay $d$ the *degree* of the system's denominator will generally be larger than the *order* $n$ of the system.

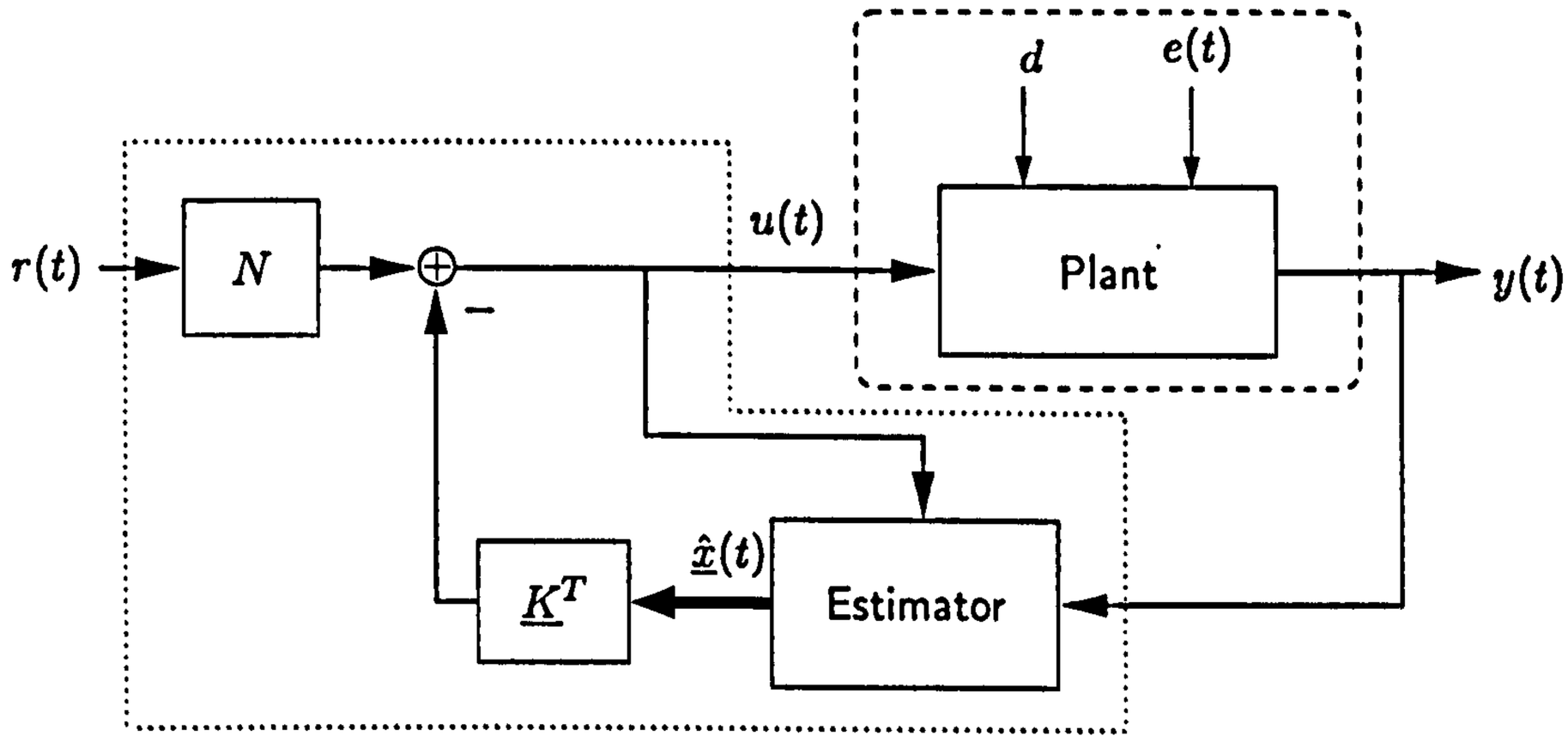



Figure C.2: Linear state feedback control with state estimator. The dashed line marks the plant, the controller components are surrounded by the dotted line.

### C.2.1 State-Feedback Controller Design

Given a measurement of the full state of the plant $\underline{x}(t) \in \mathbb{R}^n$, knowledge of the offset terms $\underline{d}^x$ and $d^y$, and some command signal $r(t)$, a general state feedback controller can be defined to consists of a feedforward control $u^{\text{ff}}(t)$ and a feedback control $u^{\text{fb}}$,

$$u(t) = u^{\text{ff}}(t) + u^{\text{fb}}(t)\,. \tag{C.9}$$

The feedforward component compensates the effect of the constant offset terms,

$$u^{\text{ff}}(t) = -Nd^y - \frac{\underline{b}^T}{\underline{b}^T\underline{b}}\,\underline{d}^x\,, \tag{C.10}$$

where $N \in \mathbb{R}$ is a scalar factor. The feedback component determines the dynamic properties of the control loop,

$$u^{\text{fb}}(t) = Nr(t) - \underline{K}^T\underline{x}(t)\,. \tag{C.11}$$

Provided that the plant is controllable, the eigenvalues of the closed loop can be assigned by selecting the feedback gain vector $\underline{K} \in R^n$ appropriately, as can be seen from the resulting closed-loop state equation,

$$\dot{\underline{x}}(t) = (A - \underline{bK}^T)\,\underline{x}(t) + \underline{b}\,N[r(t) - d^y]\,. \tag{C.12}$$

Using pole-placement design as introduced in the previous section, we define a desired characteristic polynomial of the closed loop,

$$\alpha_{\text{cl}}(s) = \det[sI - (A - \underline{bK}^T)] = s^n + a_1 s^{n-1} + a_2 s^{n-2} + \cdots + a_n\,. \tag{C.13}$$

A number of techniques exist which relate the coefficients of the polynomial $\alpha_{cl}(s)$ to the elements of the feedback gain $\underline{K}$. A compact solution is given by Ackermann's formula (Franklin *et al.* 1994). The feedback gain vector can be obtained as

$$\underline{K}^T = [0 \quad \dots \quad 0 \quad 1]\, \boldsymbol{F}^{-1}\, \alpha_{cl}(\boldsymbol{A}) \qquad \in \mathbb{R}^{1\times n}\,, \tag{C.14}$$

where $\boldsymbol{F} \in \mathbb{R}^{n\times n}$ is the controllability matrix,

$$\boldsymbol{F} = [\underline{b} \quad \boldsymbol{A}\underline{b} \quad \boldsymbol{A}^2\underline{b} \quad \dots \quad \boldsymbol{A}^{n-1}\underline{b}] \tag{C.15}$$

and the matrix equation $\alpha_{cl}(\boldsymbol{A}) \in \mathbb{R}^{n\times n}$ is defined as

$$\alpha_{cl}(\boldsymbol{A}) = \boldsymbol{A}^n + a_1\boldsymbol{A}^{n-1} + a_2\boldsymbol{A}^{n-2} + \cdots + a_n\boldsymbol{I}\,, \tag{C.16}$$

with the coefficients from equation (C.13).

The scalar factor $N$ in equation (C.11) ensures a steady state gain of one from the command $r(t)$ to the output signal, $y(t)$. Using equation (A.18) with equations (C.12) and (C.8b), $N$ is chosen as,

$$N = -\frac{1}{\underline{c}^T(\boldsymbol{A} - \underline{b}\underline{K}^T)^{-1}\underline{b}}\,. \tag{C.17}$$

### C.2.2 Estimator Design

A straightforward approach of estimating the full state of the plant would be to use an exact model of the plant dynamics with the same input signal as the plant. The estimated state $\hat{\underline{x}}(t)$ will, however, only be equal to the state of the plant, $\underline{x}(t)$, if the initial conditions $\hat{\underline{x}}(0)$ and $\underline{x}(0)$ are identical. Furthermore, small disturbances and modelling uncertainties will cause the estimated state to diverge from the true state.

A robust method of estimating the state is to include feedback information of the plant output in the estimator model,

$$\dot{\hat{\underline{x}}}(t) = \boldsymbol{A}\,\hat{\underline{x}}(t) + \underline{b}\,u(t) + \underline{d}^x + \underline{L}\,[y(t) - d^y - \underline{c}^T\,\hat{\underline{x}}(t)]\,, \tag{C.18}$$

where $\underline{L} \in \mathbb{R}^n$ is the estimator gain vector. Using equation (C.8) and defining the state estimation error as $\tilde{\underline{x}}(t) = \underline{x}(t) - \hat{\underline{x}}(t)$, the estimation error dynamics can be obtained as,

$$\dot{\tilde{\underline{x}}}(t) = (\boldsymbol{A} - \underline{L}\underline{c}^T)\,\tilde{\underline{x}}(t) \tag{C.19}$$

Thus, the dynamics of the state estimation error are defined by the characteristic polynomial,

$$\alpha_e(s) = \det[s\boldsymbol{I} - (\boldsymbol{A} - \underline{L}\underline{c}^T)] = s^n + a_1^e s^{n-1} + a_2^e s^{n-2} + \cdots + a_n^e\,. \tag{C.20}$$

Provided that the plant is observable, the poles of $\alpha_e(s)$ can be completely defined by an appropriate selection of $\underline{L}$. As in equations (C.14)—(C.16), we can use Ackermann's formula (Franklin *et al.* 1994) to obtain the estimator gain vector,

$$\underline{L} = \alpha_e(\boldsymbol{A})\,\boldsymbol{O}^{-1}\,[0 \quad \dots \quad 0 \quad 1]^T \qquad \in \mathbb{R}^{n\times 1}\,. \tag{C.21}$$

Here, $\boldsymbol{O} \in \mathbb{R}^{n \times n}$ is the observability matrix,

$$\boldsymbol{O} = [\underline{c}^T \quad \underline{c}^T A \quad \underline{c}^T A^2 \quad \dots \quad \underline{c}^T A^{n-1}]^T , \tag{C.22}$$

and the matrix equation $\alpha_e(A) \in \mathbb{R}^{n \times n}$ is defined as,

$$\alpha_e(A) = A^n + a_1^e A^{n-1} + a_2^e A^{n-2} + \cdots + a_n^e I , \tag{C.23}$$

with the coefficients from equation (C.20).

To ensure that the state estimation error $\underline{\tilde{x}}$ decays quickly to zero, it is desirable to have fast estimator dynamics. The consequence of increasing the speed of the estimator is that the bandwidth of the estimator becomes higher. In the presence of sensory noise, this causes more noise to pass on to the control actuator which is generally not desirable. Thus, good estimator design is a balance between fast estimator response and low-enough bandwidth.

### C.2.3 Integral Action

Instead of using a feedforward control action $u^{\mathrm{ff}}(t)$ which is obtained using explicit knowledge of the offset terms, the steady state effect of the offset terms can be eliminated by including integral action in the controller.

We aim to ensure that the control error, $e^c(t) = y(t) - r(t)$, decays to zero in the steady state. Thus, an additional state is defined,

$$\dot{x}_I(t) = e^c(t) = y(t) - r(t) . \tag{C.24}$$

Replacing $y(t)$ by equation (C.8b), and augmenting equation (C.8a) with the extra state results in

$$\begin{bmatrix} \dot{x}_I(t) \\ \dot{\underline{x}}(t) \end{bmatrix} = \begin{bmatrix} 0 & \underline{c}^T \\ \underline{0} & A \end{bmatrix} \begin{bmatrix} x_I(t) \\ \underline{x}(t) \end{bmatrix} + \begin{bmatrix} 0 \\ \underline{b} \end{bmatrix} u(t) - \begin{bmatrix} 1 \\ \underline{0} \end{bmatrix} r(t) + \begin{bmatrix} d^y \\ \underline{d}^x \end{bmatrix} \tag{C.25}$$

Similar to equation (C.11), a control law which is based on the feedback of the plant state can be defined,

$$u(t) = -\begin{bmatrix} K_I & \underline{K}_x^T \end{bmatrix} \begin{bmatrix} x_I(t) \\ \underline{x}(t) \end{bmatrix} = -\underline{K} \begin{bmatrix} x_I(t) \\ \underline{x}(t) \end{bmatrix} . \tag{C.26}$$

The corresponding control structure, using an estimator to approximate the state of the plant, is shown in Figure C.3.

The differential equation of the closed-loop becomes,

$$\begin{bmatrix} \dot{x}_I(t) \\ \dot{\underline{x}}(t) \end{bmatrix} = \begin{bmatrix} 0 & \underline{c}^T \\ -\underline{b} K_I & A - \underline{b}\,\underline{K}_x \end{bmatrix} \begin{bmatrix} x_I(t) \\ \underline{x}(t) \end{bmatrix} - \begin{bmatrix} 1 \\ \underline{0} \end{bmatrix} r(t) + \begin{bmatrix} d^y \\ \underline{d}^x \end{bmatrix} . \tag{C.27}$$

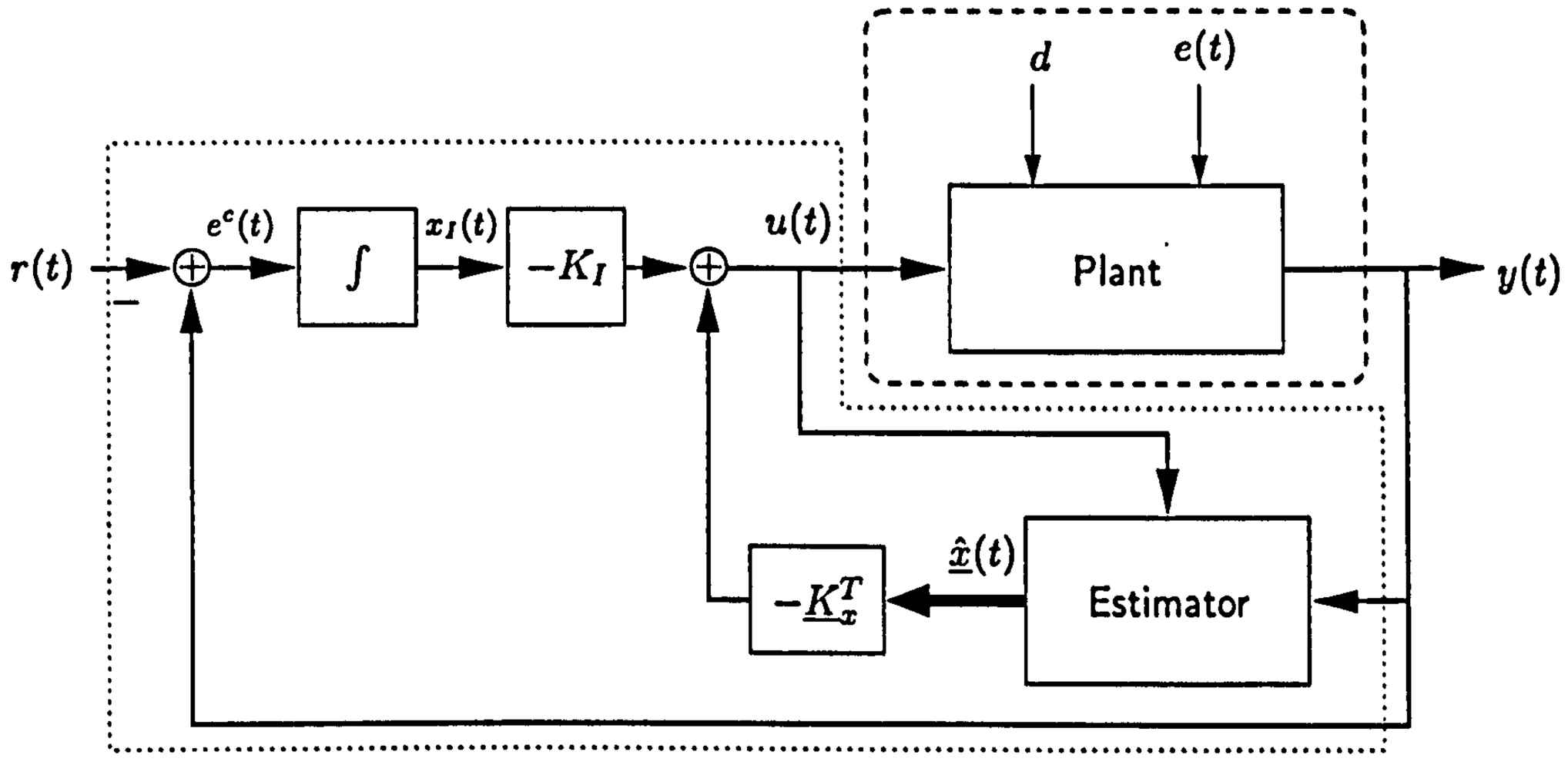



Figure C.3: Linear state feedback control including integral action, with state estimator. The dashed line marks the plant, the controller components are surrounded by the dotted line.

Provided that the augmented plant is controllable, the eigenvalues of the closed loop can be assigned by selecting the feedback gain $\underline{K} \in \mathbb{R}^{n+1}$ accordingly. The same algorithm as described by equations (C.13)—(C.16) can be applied.

If $\underline{K}$ is chosen such that the closed loop is stable, the first equation in (C.27) has the following form for the steady state,

$$\lim_{t\to\infty} \dot{x}_I(t) = 0 = \lim_{t\to\infty} [\underline{c}^T \underline{x}(t) + d^y - r(t)]\,. \tag{C.28}$$

With equation (C.8b) and (C.24), we obtain

$$\lim_{t\to\infty} e^c(t) = 0 = \lim_{t\to\infty} [y(t) - r(t)]\,, \tag{C.29}$$

i.e., the steady state control error is zero. Note that no explicit knowledge of the values of the offset terms is used in the control law (C.26).

# D Simulation

The way a model is simulated can have a significant influence on the choice of the parameter estimation algorithm and on the identification results. In this chapter, we discuss aspects of system simulation in continuous and in discrete time. The time-shift and the delta operator are introduced as discrete-time operators. For a review of other discrete-time operators, see (Back *et al.* 1997). A detailed discussion of discrete and continuous time system representation with particular emphasis on the delta operator can be found in (Middleton and Goodwin 1990).

## D.1 Continuous Time

Continuous-time simulation is the most straightforward and natural way to describe a system, as physical systems are always continuous. Models derived from first principles can often be represented as a set of ordinary differential equations (ODEs),

$$p\underline{x}(t) = f(\underline{x}(t), u(t)) \tag{D.1a}$$

$$y(t) = g(\underline{x}(t)\,, \tag{D.1b}$$

where $f()$ and $g()$ are non-linear functions, $f()$ is continuously differentiable. For simplicity we restrict ourselves to single input – single output systems, i.e. $u \in \mathbb{U} \subset \mathbb{R}$ and $y \in \mathbb{Y} \subset \mathbb{R}$ are the input and the output of the system, respectively. The dimensionality of the state vector $\underline{x} \in \mathbb{X} \subset \mathbb{R}^n$ defines the dynamic order of the system. For consistency throughout this chapter, we denote the derivative with respect to time, $d/dt$ by the operator $p$, i.e. $p\underline{x}(t) = \dot{\underline{x}}(t) = d\underline{x}(t)/dt$. For simplicity the initial condition, $\underline{x}(0) = \underline{x}_0$, is omitted and no input delay is considered throughout this appendix.

The corresponding linear form of a continuous time system can be written as

$$p\underline{x}(t) = A\underline{x}(t) + \underline{b}u(t) \tag{D.2a}$$

$$y(t) = \underline{c}^T\underline{x}(t)\,. \tag{D.2b}$$

Note that in equation (D.2), the variables $\underline{x}$, $u$ and $y$ denote deviations from the operating point for which the system was linearised.

The linear system is stable if all the eigenvalues of the Matrix $A$ (i.e., the poles of the characteristic polynomial) are located in the left half of the complex plane or on the imaginary

axis, and asymptotically stable if the eigenvalues are located in the left half plane. This region of stability is shown in Figure D.1.

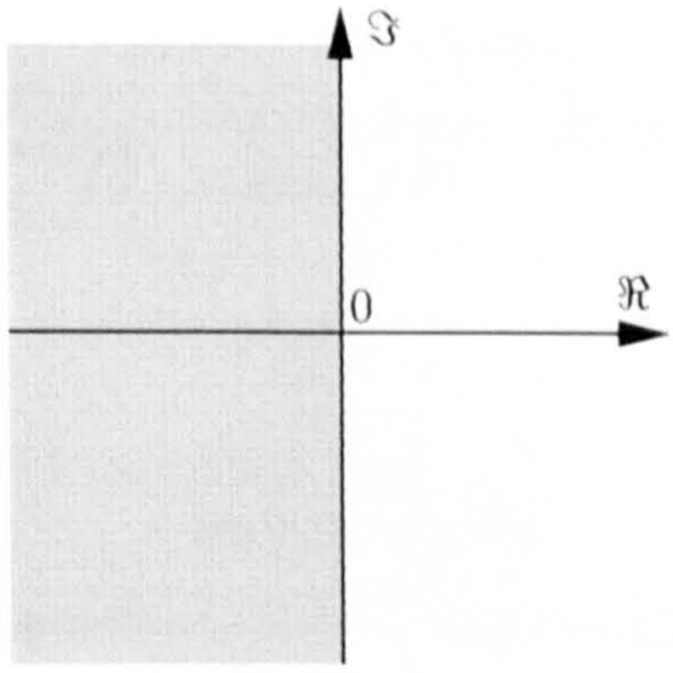



Figure D.1: Region of stability for continuous system.

Continuous time systems can only be simulated directly by analogue devices. When digital computing hardware is used, numerical integration algorithms such as Runge–Kutta have to be used to solve the system of ODEs. This can be computationally expensive and represents one of the major drawbacks of continuous system simulation.

## D.2 Discrete Time

When using digital computers for system analysis, identification or control, we deal with sampled signals instead of continuous signals. The relationship between the ('real') continuous system and its discrete approximation which is obtain from the computer is shown in Figure D.2. Here, the time-continuous input signal $u(\cdot)$ is transformed into a sampled data sequence $\{u(k)\}$, using an A-D converter. The sequence $\{u(k)\}$ forms the input of the discrete model, the output sequence $\{y(k)\}$ of which is interpolated by the D-A converter to the approximated continuous output $\hat{y}(\cdot)$. Both converters are synchronised by the sampling interval $T_s$.

Owing to the sampling process, the interpolated time-continuous output $\hat{y}(\cdot)$ is generally only an approximation of the output of the real continuous time system, $y(\cdot)$. The quality of this approximation depends on both the sampling interval and the interpolation process. We consider the interpolation to be of order zero (sample-and-hold).

For the transformation of the continuous input to a discrete sequence, aliasing effects have to be considered, and low pass filtering may be necessary prior to sampling.

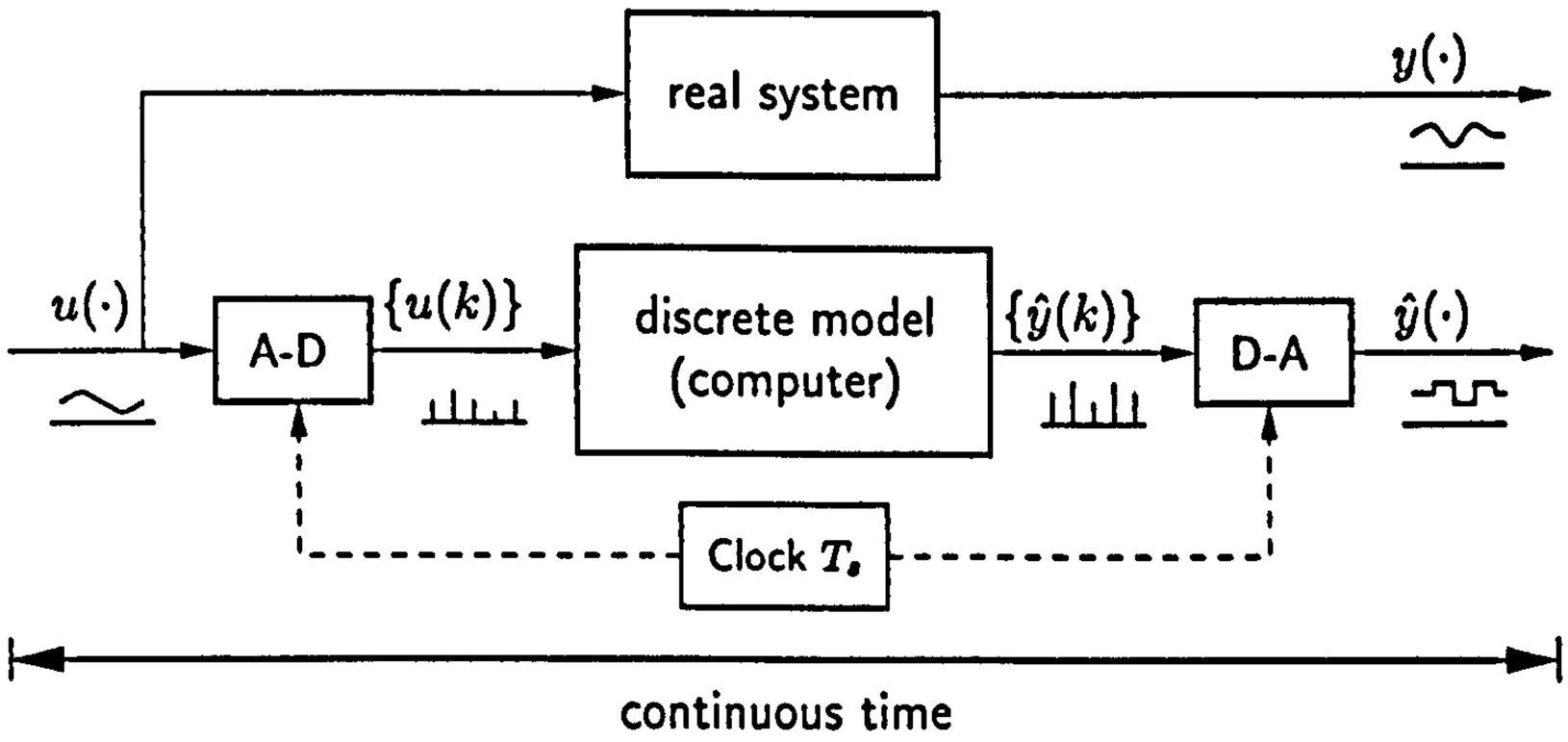



Figure D.2: Relationship between continuous-time and discrete-time system.

The continuous time system can now be represented by its discrete approximation as

$$\frac{\underline{x}((k+1)T_s) - \underline{x}(kT_s)}{T_s} = f^d(\underline{x}(kT_s), u(kT_s), T_s) \tag{D.3a}$$

$$y(kT_s) = g^d(\underline{x}(kT_s), T_s)\,, \tag{D.3b}$$

where $f^d()$ and $g^d()$ relate to the corresponding continuous functions in equation (D.1), but take the effect of the A-D and D-A converters into account.

For linear systems, we can find representations of the system (D.3) which match the continuous system exactly at the sampling instances. We introduce two such representations, the first using the traditional time-shift operator, the second the delta operator, which has superior numerical properties if the sampling interval is small.

For simplicity, we replace the discrete time argument $kT_s$ by the index $k$, i.e., $x_k$ denotes $x(kT_s)$.

### D.2.1 Shift Operator

When we introduce the time-shift operator $q$,

$$qx_k = x_{k+1}\,, \tag{D.4}$$

the system (D.3) can be rewritten as

$$q\underline{x}_k = \underline{x}_k + T_s f^d(\underline{x}_k, u_k, T_s) \tag{D.5a}$$

$$y_k = g^d(\underline{x}_k, T_s)\,. \tag{D.5b}$$

For the linear system (D.2), the corresponding discrete system in shift-operator notation is

$$q\underline{x}_k = A^q\underline{x}_k + \underline{b}^q u_k \tag{D.6a}$$

$$y_k = \underline{c}^q\underline{x}_k\,, \tag{D.6b}$$

where $\{\boldsymbol{A}^q, \underline{b}^q, \underline{c}^q\}$ can be directly obtained from the continuous representation.

The discrete system in shift-operator notation is stable if all the eigenvalues of the matrix $\boldsymbol{A}^q$ are located inside or on the unit circle in the complex plane, and asymptotically stable if the eigenvalues are located inside the unit circle. This region of stability is shown in Figure D.3.

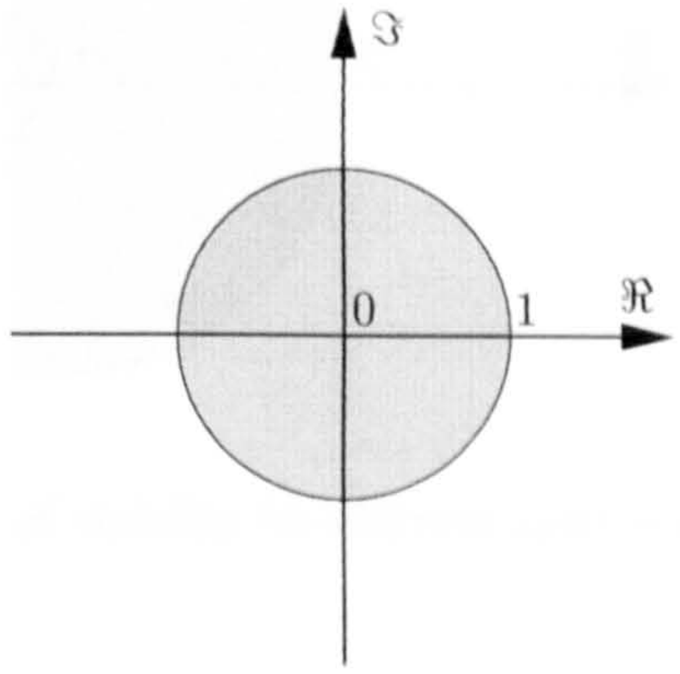



Figure D.3: Region of stability for discrete system with shift operator.

The imaginary axis from the continuous complex plane, Figure D.1, is mapped to the border of the unit circle in Figure D.3. When the sampling time tends to zero, $T_s \to 0$, the poles of the system migrate towards the point $[1, 0]$ in the discrete complex plane. Zeros of the discrete system which have been introduced by the sampling process migrate to fixed positions inside the unit circle.

### D.2.2 Delta Operator

When we introduce the delta operator $\delta$,

$$\delta = \frac{q-1}{1} \qquad \delta\underline{x}_k = \frac{\underline{x}_{k+1} - \underline{x}_k}{T_s}\,, \tag{D.7}$$

the system (D.3) can be rewritten as

$$\delta\underline{x}_k = f^d(\underline{x}_k, u_k, T_s) \tag{D.8a}$$

$$y_k = g^d(\underline{x}_k, T_s)\,. \tag{D.8b}$$

For the linear system (D.2), the corresponding discrete system in delta operator notation is

$$\delta\underline{x}_k = \boldsymbol{A}^\delta \underline{x}_k + \underline{b}^\delta u_k \tag{D.9a}$$

$$y_k = \underline{c}^\delta \underline{x}_k\,, \tag{D.9b}$$

where $\{\boldsymbol{A}^\delta, \underline{b}^\delta, \underline{c}^\delta\}$ can be directly obtained from the continuous representation.

The discrete system in delta operator notation is stable if all eigenvalues of the matrix $\boldsymbol{A}^\delta$ are located inside or on a circle of the radius $1/T_s$ with centre at $[-1/T_s, 0]$ in the complex

plane, and asymptotically stable if the eigenvalues are located inside the circle. This region of stability is shown in Figure D.4.

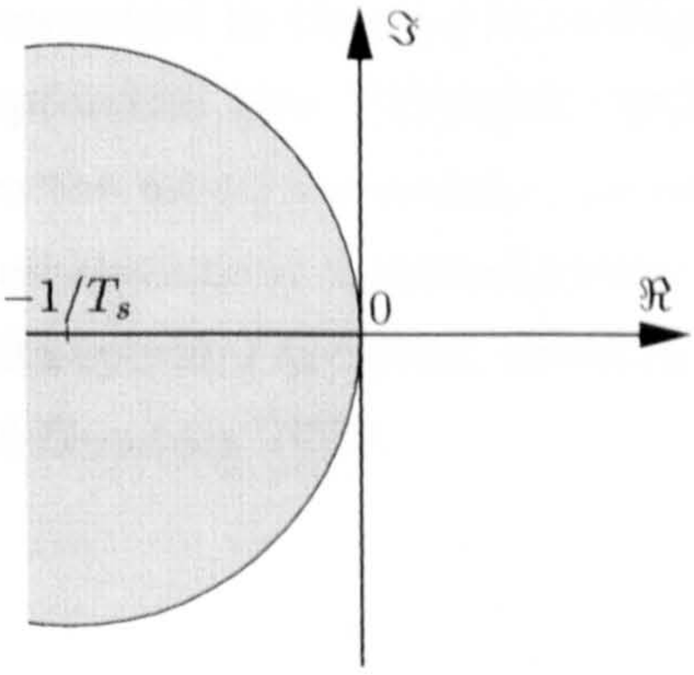



Figure D.4: Region of stability for discrete system with delta operator.

The imaginary axis of the continuous complex plane, Figure D.1, is mapped to the border of the stability region in Figure D.4. When the sampling interval tends to zero, $T_s \to 0$, the region of stability becomes the entire left half plane. The poles of the discrete system migrate to the location of the corresponding poles of the continuous system. Zeros which have been introduced by the sampling process go to $-\infty$. Thus, for $T_s \to 0$, the discrete system in delta operator notation converges towards the corresponding continuous system.

## D.3 Discussion

The simulation of a system in discrete-time has computational advantages compared to continuous simulation, as the model can be computed directly using digital techniques. Continuous-time simulation requires expensive integration algorithms to solve the system of ODEs which are often difficult to implement in real-time.

On the other hand, continuous time is the natural domain in which to represent a physical system. The approximations introduced by discrete representations must be carefully taken account of, especially when dealing with non-linear systems.

We introduced the shift and the delta operator as possible domains for the representation of a system in discrete time. The shift operator is very easy to implement, and a variety of model structure, such as the (N)ARX model (cf. Section 2.2.2), are directly based on it. However, when a system is represented in the shift operator domain, any relation to the continuous time domain is lost. In particular, the shift operator representation does not converge to the continuous representation when the sampling interval approaches zero. This can have significant effects for the identification of systems which are sampled very fast. The poles of the model in shift operator representation then tend to converge to the point $[1, 0j]$

in the complex plane. This can lead to sensitivity problems for the parameter estimation.

The delta operator is directly based on the continuous representation in the sense that it represents a discrete approximation of the differential operator $p$. Thus, the representation of a system in the delta domain converges to the corresponding continuous-time representation when the sampling interval approaches zero. Sensitivity problems due to fast sampling are therefore not encountered when the model parameters are estimated.

For linear systems, the representations in continuous time, equation (D.2), in shift operator form, equation (D.6), and in delta operator notation, equation (D.9), can be easily transformed into each other (Middleton and Goodwin 1990).

# E The Heat Transfer Process

The heat transfer process is used in examples throughout Part I of this thesis to demonstrate basic concepts of the operating regime based approaches to modelling and control presented in Chapter 2 and 3. A similar process has been studied in (Johansen and Foss 1995).

We first introduce the heat transfer system from which experimental data were obtained. Then the data collection experiments are described. Finally, some system properties, like dynamic order and time delays, and non-linearities of the plant are discussed in a first analysis of the data.

## E.1 The System

The experimental setup of the heat transfer process is illustrated in Figure E.1.

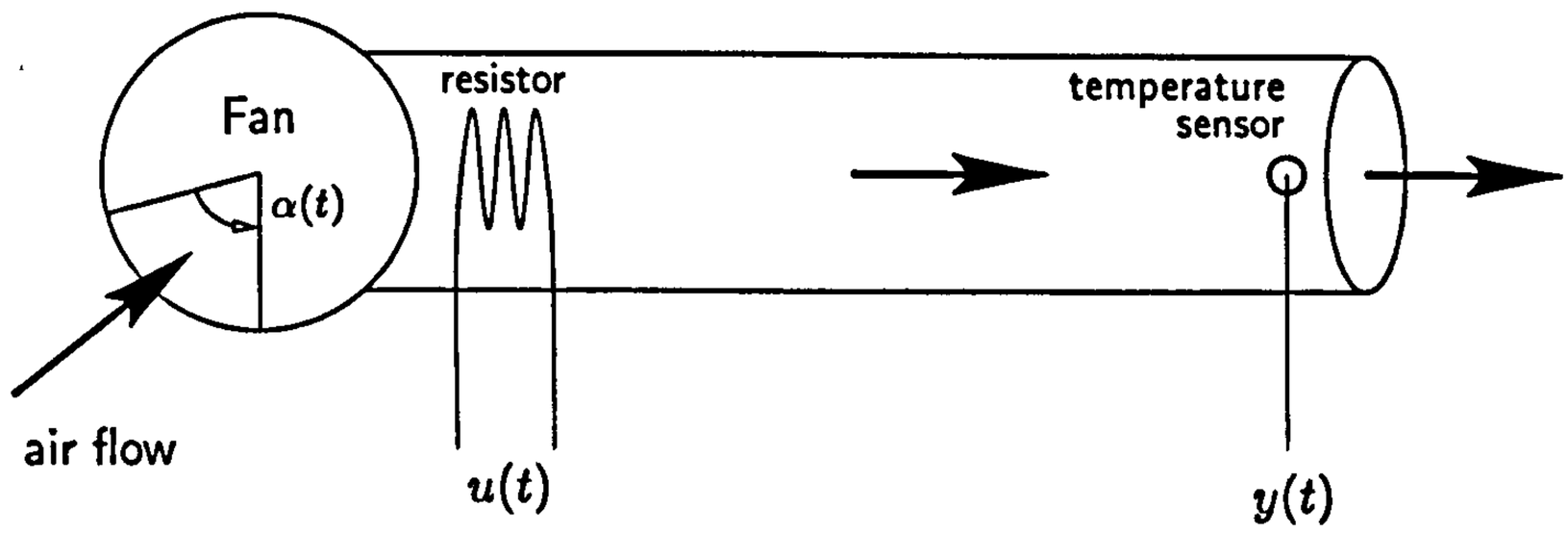



Figure E.1: Heat transfer process.

Air is driven by a fan through a tube. The air is heated by a resistor at the tube input, and its temperature is measured at the outlet. The air flow can be adjusted by changing the valve angle $\alpha$ of the fan inlet.

The heat transfer process can be interpreted as a system with two inputs, the valve angle $\alpha(t)$ and the voltage at the resistor, $u(t)$, and one output, the voltage $y(t)$ at the temperature detecting element at the tube outlet.

The system is non-linear in a number of ways:

- The gain and time-delay of the heat transfer process $u(t) \rightarrow y(t)$ vary with changing valve angle $\alpha(t)$. A larger angle causes the air to flow faster which decreases the delay

$u(t) \rightarrow y(t)$. At the same time a larger angle increases the volume of air pulled through the tube and heated by the resistor. Hence, the temperature at the outlet will decrease.

- The heat transmitted from the heating element will generally depend on the voltage in a non-linear way. Only for small changes in $u$, it can be approximated by a linear relationship.

When modelling this process, we are only interested in the short term effect of heating the air moving through the tube. Throughout the experiments, the tube itself will be heated up. However, this process has much larger time constants than the heating of the moving air and is therefore neglected in this study.

## E.2 Experimental Data

The experiments for recording data are aimed at exciting all dynamic modes of the system for all operating conditions. Examples of the data sets recorded are shown in Figures E.2 and E.3. The sampling period was chosen at $T_s = 0.05$ s. Four different experiments have been performed to obtain data:

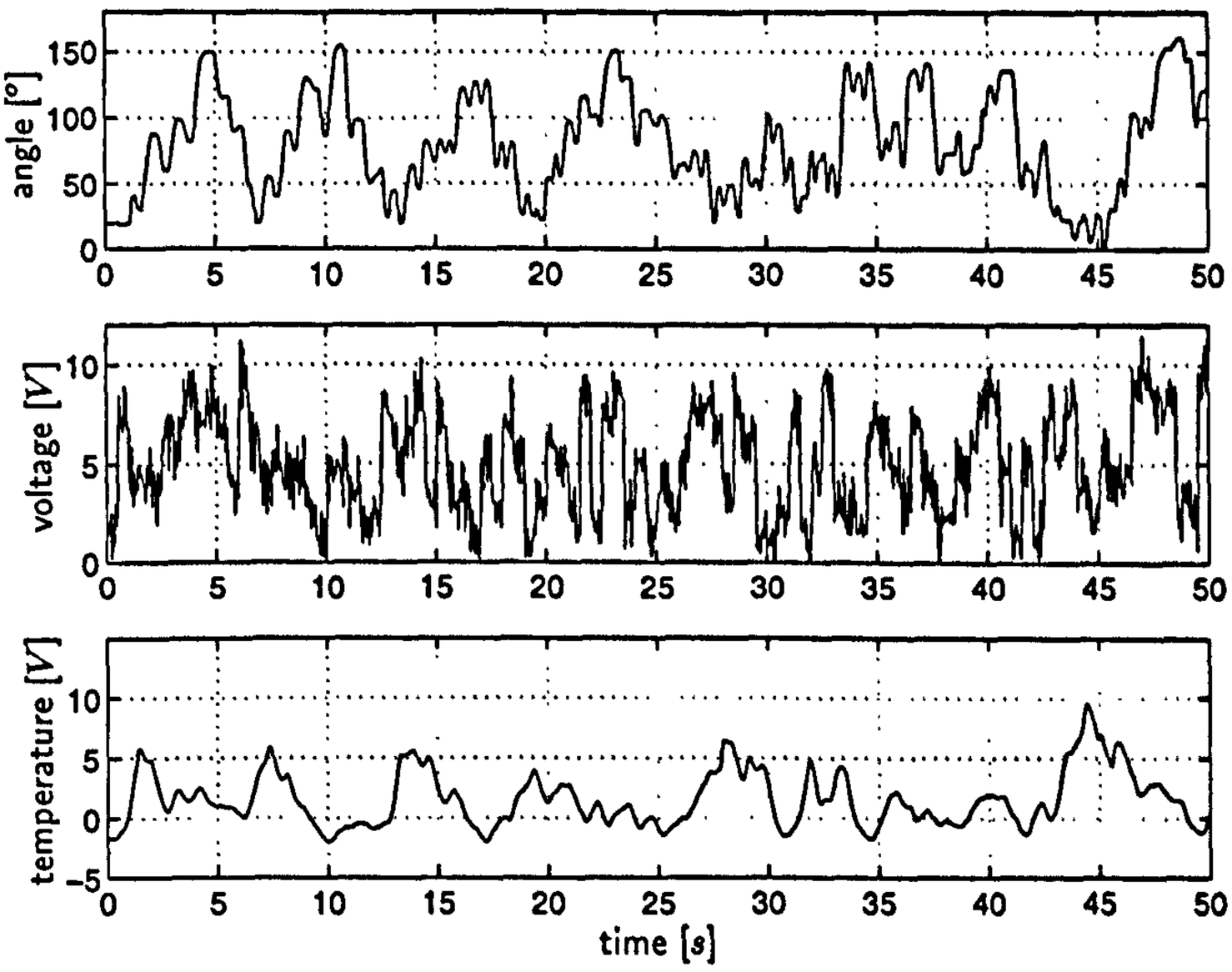
- "Fast pseudo–random data", Figure E.2(a), are aimed to excite all modes of the system. The voltage changes fast in a pseudo-random way, around a slower pseudo-random variations, to excite all dynamics of the system for different amplitudes. The angle changes also in a pseudo-random way. Three data sets of 50 seconds each were recorded.

- In the "slow pseudo–random data", Figure E.2(b), the voltage changes in a pseudo–random fashion, but slower than in the fast pseudo–random data, thus exciting different modes than those data. The angle varies again in a pseudo-random way. Two data sets of 50 seconds each were recorded.

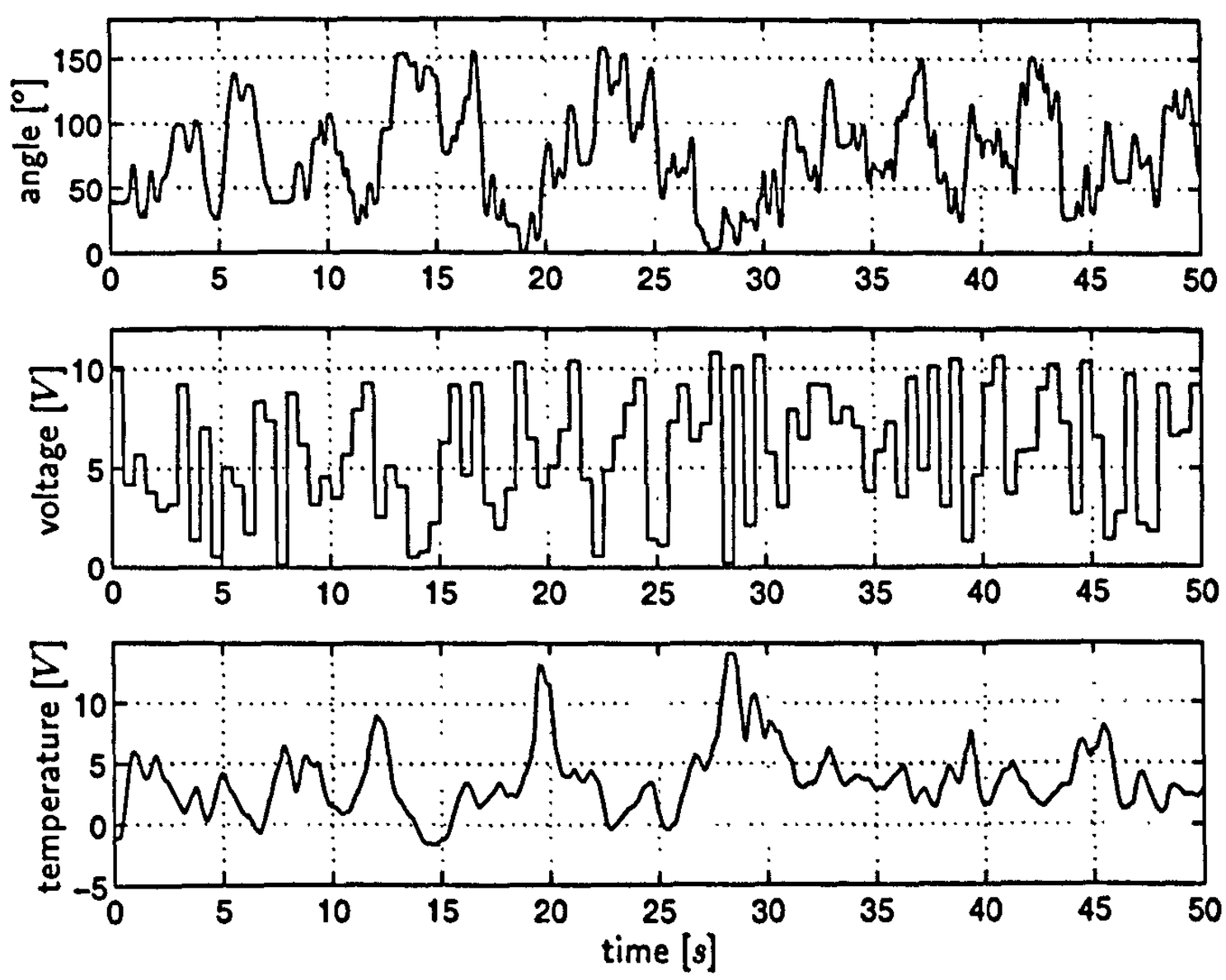
- In the "test–angle data", Figure E.3(a), the voltage is held constant for 25 seconds, thus obtaining a response to changes of the angle only. The angle varies randomly every few seconds. Two data sets of 50 seconds each were recorded.

- In the "test–voltage data", Figure E.3(b), the angle is held constant at values of $\alpha = [20^o, 40^o, 60^o, 90^o, 120^o]$ for 50 seconds each. For every angle, the same random voltage sequence is applied. Two data sets of 250 seconds each were recorded.

The recorded data contain noise and are therefore filtered by a second order FIR filter (mean filter). All data are normalised in such a way that their values are mapped to the region [0, 1].

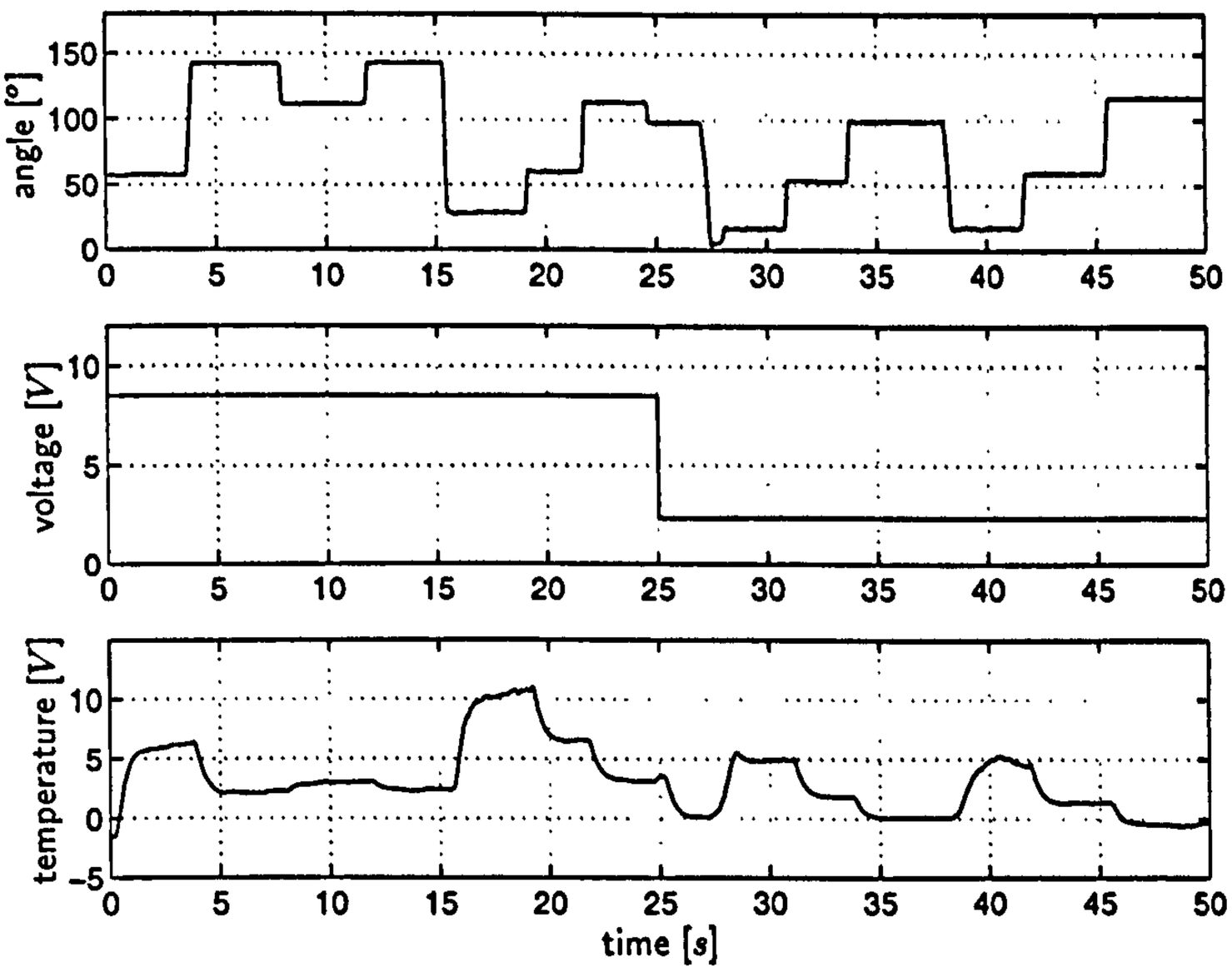



(a) Fast pseudo–random.

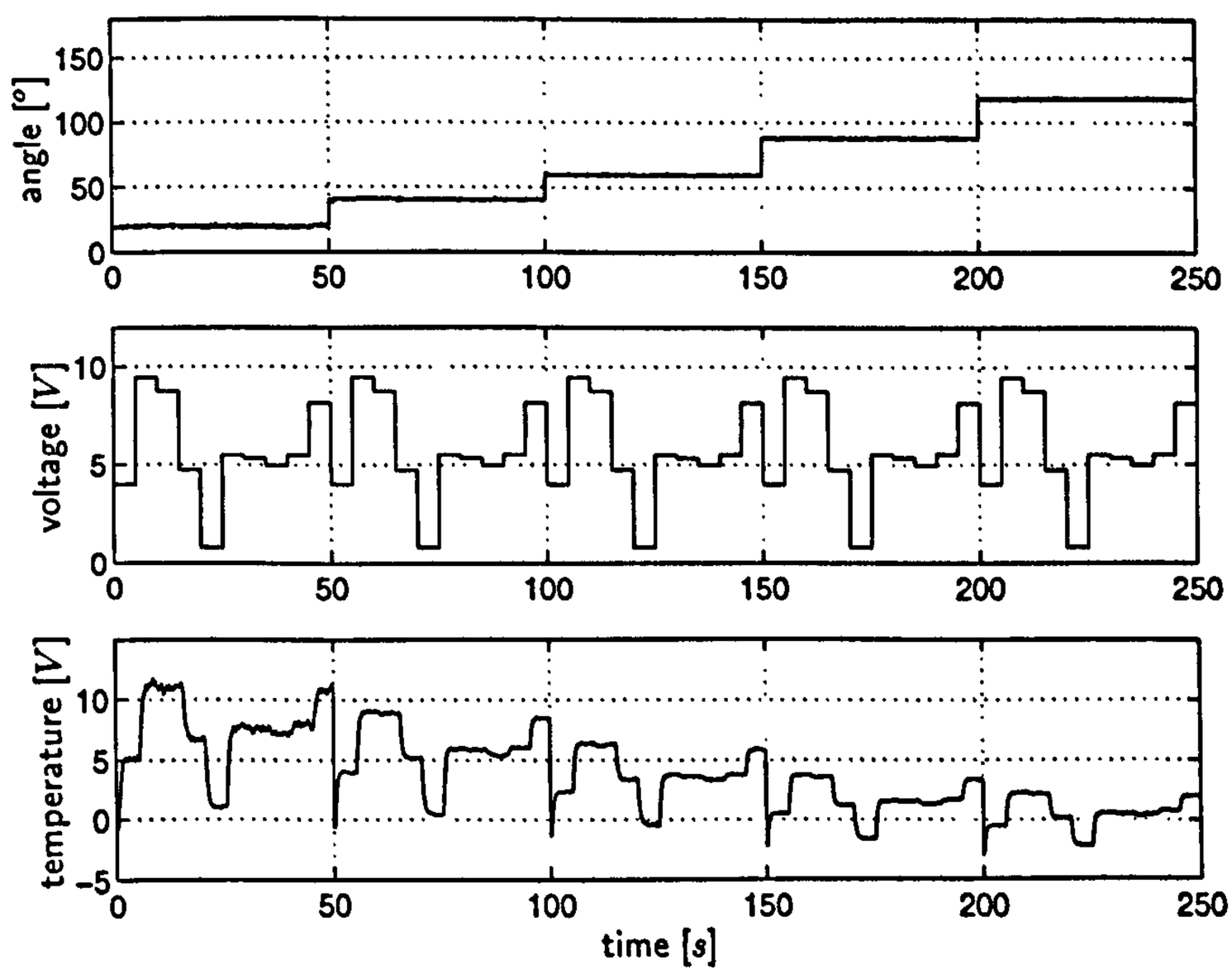



(b) Slow pseudo–random.

Figure E.2: Examples of experimental data from the heat transfer process.




(a) Test–angle



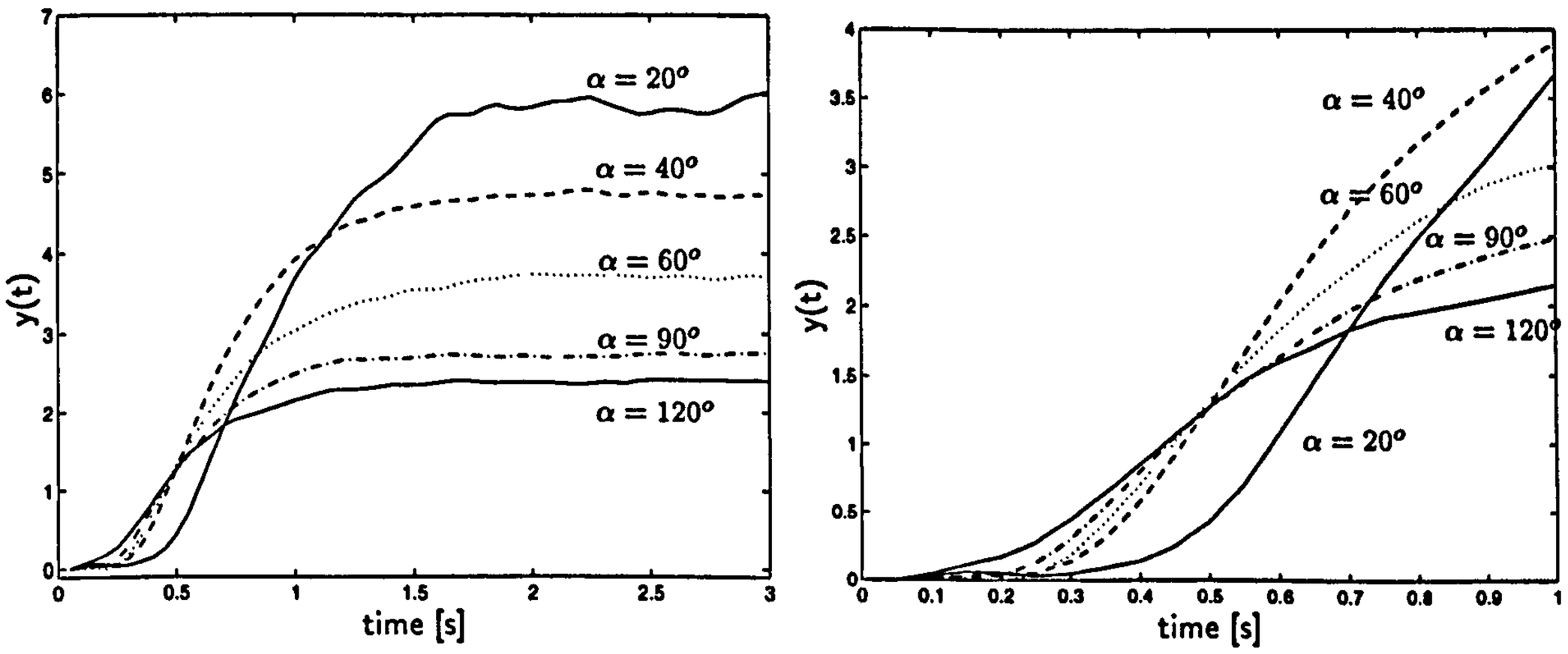

(b) Test–voltage

Figure E.3: Examples of experimental data from the heat transfer process (continued).

## E.3 First Analysis

The analysis of the step response of a system under varying operating conditions can provide information about the dynamic order, time-delays and non-linear characteristics of the process. In Figure E.4, responses to a 4 volt input step are shown for various valve angles. The following observations can be made:

- In the step responses shown in Figure E.4(a), a change of the gain from the voltage $u(t)$ to the output $y(t)$ with varying valve angle $\alpha$ can be observed. The gain is large for small valve angles (small airflow) and decreases as the valve is opened further (airflow increases).




(a) Step response

(b) Step response (shorter time-scale)

Figure E.4: Response to a 4 volt step input.

- From the short time-scale step response shown in Figure E.4(b), the change of the delay of $u(t) \rightarrow y(t)$, $T_{d,u}$, with varying valve angle can be determined. The delay is small for large valve angles, as the air flows fast. It increases when the valve is being closed, as the air flows slower. The delays estimated from Figure E.4(b) for various angles are shown in Table E.1.

Two responses to steps of the valve angle for constant voltages are shown in Figure E.5. It is difficult to determine the exact value of the delay, as the change of the angle is not exactly step-like.

Additional linear identification experiments confirm that the optimal value for the delay of the system $\alpha(t) \rightarrow y(t)$, $T_{d,\alpha}$ is approximately 0.1s smaller than the corresponding delay of the system $u(t) \rightarrow y(t)$, $T_{d,u}$. These results are summarised in Table E.1.

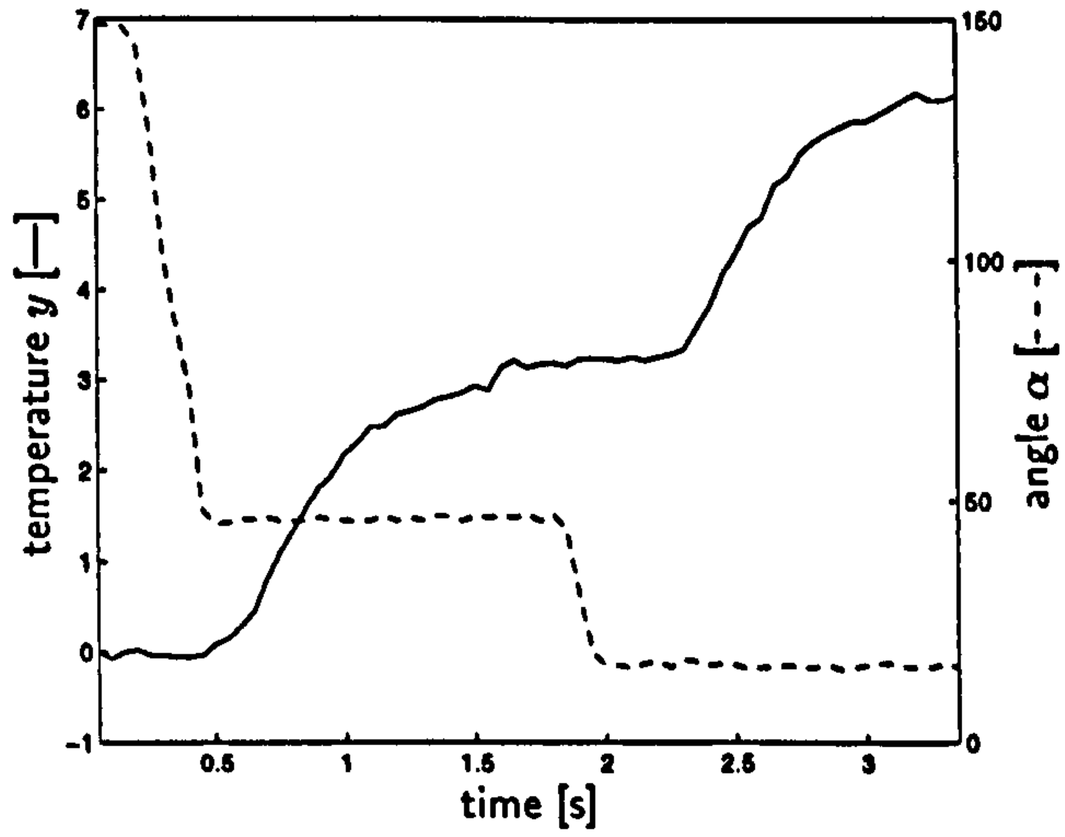



Figure E.5: Responses to steps of the valve angle.

| angle $\alpha$ [$^o$] | delay $u \rightarrow y$ | | delay $\alpha \rightarrow y$ | |
|---|---|---|---|---|
| | $T_{d,u}$ [s] | $T_{d,u}/T_s$ | $T_{d,\alpha}$ [s] | $T_{d,\alpha}/T_s$ |
| 20 | 0.30 | 6 | 0.20 | 4 |
| 40 | 0.25 | 5 | 0.15 | 3 |
| 60 | 0.25 | 5 | 0.15 | 3 |
| 90 | 0.20 | 4 | 0.10 | 2 |
| 120 | 0.10 | 2 | $< 0.05$ | $< 1$ |

Table E.1: Delays for various valve angles $\alpha$.

From the shapes of the step responses in Figure E.4 and E.5, it can be surmised that the system is of 1st or 2nd order.

## E.4 Examples

The heat transfer process is used as a simple system to illustrate the modelling and control principles throughout Part I of this thesis. The examples in this section are referred to in Chapter 2 and 3.

**Example E.4.1** *The inputs to the heat transfer process are the valve angle $\alpha$ and the voltage at the resistor $u$, the output is the temperature at the outlet, $y$. As discussed in Section E.1, the characteristics of the system strongly depend on the valve angle. For significant changes of the voltage $u$, this input will also have a non-linear influence on the system. There is no evidence to suggest that the system characteristics will change significantly with the temperature at the outlet (as long as this temperature remains in a reasonable range.).*

*The most straightforward choice is therefore to include the valve angle and the voltage in the scheduling vector. This results in a 2-dimensional scheduling space,*

$$\underline{\phi}(t) = [\alpha(t - T_{d,\alpha}),\, u(t - T_{d,u})]^T \qquad \in \mathbb{R}^2 \,. \tag{E.1}$$

*This setup has been used in (Johansen and Foss 1995).*

*For small changes of $u$, or if the system non-linearities with respect to the voltage are not significant, it might be sufficient to include only the valve angle in the scheduling vector,*

$$\underline{\phi}(t) = \alpha(t - T_{d,\alpha}) \qquad \in \mathbb{R}. \tag{E.2}$$

*This reduces the dimensionality of the scheduling space significantly, and, although it is not as general as (E.1), it may help to reduce the variance in the estimate of the model parameters.*

**Example E.4.2** *We attempt to identify the parameters of a linear 1st order ARX structure, equation (2.20), with the data vector chosen as*

$$\underline{\psi}(t) = [\alpha(t-3),\, u(t-5),\, y(t-1)]^T \qquad \in \mathbb{R}^3 \tag{E.3}$$

*Employing the prediction model simulation, the elements of the parameter vector can be identified using the linear least squares algorithm described in Section B.2.*

*For simulation model mode, not all elements of the data vector $\underline{\psi}$ are known in advance. Thus, a recursive optimisation algorithm must be used. For simplicity, we choose to work with the Levenberg-Marquardt method described in Section B.3.2. The modelling results together with the model parameters are shown in Table E.2.*

*The performance of the simulation model in terms of the training and test errors is significantly better than that of the prediction model. However, the parameters of the two models are very similar. The fact that the pole is located close to the unit circle (*0.911 *and* 0.945 *respectively) indicates that the system has been sampled rather fast. The parameter optimisation for the simulation model takes approximately 10 times longer than the optimisation of the parameters of the prediction model.*

| | *Prediction model* | *Simulation model* |
|---|---|---|
| *Train MSE* | $5.30 \times 10^{-3}$ | $4.35 \times 10^{-3}$ |
| *Test MSE* | $6.69 \times 10^{-3}$ | $5.32 \times 10^{-3}$ |
| $B(q^{-1})$ | $-0.0458q^{-3}\alpha(t) + 0.0365q^{-5}u(t)$ | $-0.0697q^{-3}\alpha(t) + 0.0537q^{-5}u(t)$ |
| $A(q^{-1}$ | $1 - 0.945q^{-1}$ | $1 - 0.911q^{-1}$ |
| $d$ | 0.0292 | 0.0481 |
| $G_\alpha$ | $-0.837$ | $-0.785$ |
| $G_u$ | 0.667 | 0.605 |

*Table E.2: ARX modelling (1st order).*

**Example E.4.3** *We will compare local and global optimisation of the local model parameters for models of the heat transfer process. Following the results obtained in Example E.4.2, we choose to work with local 1st order ARX models. The valve angle $\alpha$ is selected as the scheduling variable which results in an one-dimensional scheduling space. We compare networks with 5 and with 10 uniformly distributed B-spline validity functions.*

*The parameters of the local models are estimated using prediction mode. The mean squared prediction error (MSPE) and mean squared simulation error (MSSE) results are shown in Table E.3. Note that the training MSPE is the criterion used for the estimation of the local model parameters. The training MSSE is given for the prediction model for comparison to the training MSSE of the simulation model in Example E.4.4.*

| | *5 units* | | | *10 units* | | |
|---|---|---|---|---|---|---|
| | *training* | | *test* | *training* | | *test* |
| | *MSPE* | *MSSE* | *MSSE* | *MSPE* | *MSSE* | *MSSE* |
| *global learning* | $4.96 \times 10^{-5}$ | $1.80 \times 10^{-3}$ | $1.79 \times 10^{-3}$ | $4.88 \times 10^{-5}$ | $1.74 \times 10^{-3}$ | $1.88 \times 10^{-3}$ |
| *local learning* | $5.34 \times 10^{-5}$ | $1.87 \times 10^{-3}$ | $1.91 \times 10^{-3}$ | $5.12 \times 10^{-5}$ | $1.77 \times 10^{-3}$ | $1.82 \times 10^{-3}$ |

*Table E.3: Modelling results for prediction model.*

*For both model structures, the training MSPE is larger for local learning than for global learning, i.e. local learning leads to a larger bias in the estimate. The simulation error results are better with global learning only for the smaller network. With the network of 10 units, simulation error results with local learning outperform the global learning results. This suggest that the regularisation effect introduced by local learning leads to better generalisation results of the network when the structure is large and probably locally over-parameterised. Note that when trained in prediction mode, the model has to generalise from the one-step-ahead prediction*

*used for training to an infinite prediction horizon for simulation.*

**Example E.4.4** *We now compare local and global optimisation of the local model parameters for the simulation model mode. We choose to work with the same setup as in Example E.4.3. The mean squared simulation error (MSSE) results are shown in Table E.4.*

| | *5 units* | | *10 units* | |
|---|---|---|---|---|
| | *training MSSE* | *test MSSE* | *training MSSE* | *test MSSE* |
| *global learning* | $1.28 \times 10^{-3}$ | $1.27 \times 10^{-3}$ | $1.10 \times 10^{-3}$ | $1.50 \times 10^{-3}$ |
| *local learning* | $2.17 \times 10^{-3}$ | $1.78 \times 10^{-3}$ | $1.29 \times 10^{-3}$ | $1.54 \times 10^{-3}$ |

*Table E.4: Modelling results for simulation model.*

*For both model structures, the results obtained with global learning outperform those achieved using local learning. This was expected for the training error, as local learning gives a larger bias in the estimate. The observation that the global learning test errors are smaller than those with local learning suggest that the necessity for generalisation of the networks when faced with the test data is not as great as in Example E.4.3. This is to be expected as we use an infinite prediction horizon for both the parameter estimation and the model validation.*

**Example E.4.5** *To illustrate the effect of constant eigenvectors, we consider again the modelling of the heat transfer process. The model structure is similar to the one employed in Example E.4.4, with only the network with 5 units being considered here. Instead of local 1st order ARX models, we choose to work with local linear 2nd order delta-domain (cf. Section D.2.2) state space descriptions, equivalent to equations (2.14a)-(2.14b) on page 14.*

*In order to ensure that all eigenvalues are real, the parameters of the local state feedback matrices $A_i$ are not identified directly. Instead, the local eigenvectors $V_i \in \mathbb{R}^{2\times 2}$, and eigenvalues $\underline{e}_i \in \mathbb{R}^2$ are estimated. The state feedback matrices of the local models, $A_i$, are then obtained using the equation*

$$A_i = V_i \operatorname{diag}\underline{e}_i \, V_i^{-1}, \qquad i = 1 \ldots M. \tag{E.4}$$

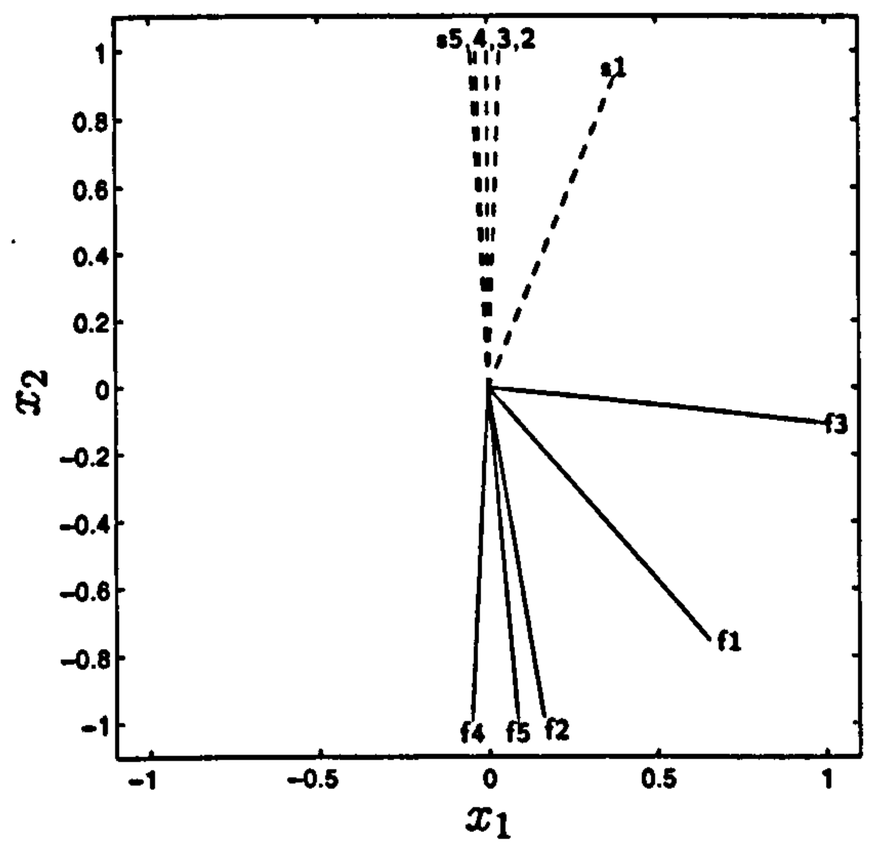
*In a first identification experiment, the eigenvectors and eigenvalues for all local models are estimated without restrictions, using global parameter estimation in simulation mode. We will refer to this model as the* general model. *In a second experiment with similar setup, a common set of eigenvectors $V$ is identified such that $\{V_i = V\}_{i=1}^{M}$, and only the eigenvalues are identified individually for each local model. This model will be referred to as the* restricted model.

*The identification results are summarised in Table E.5. The performance of the restricted model is only slightly inferior to the performance of the general model.*

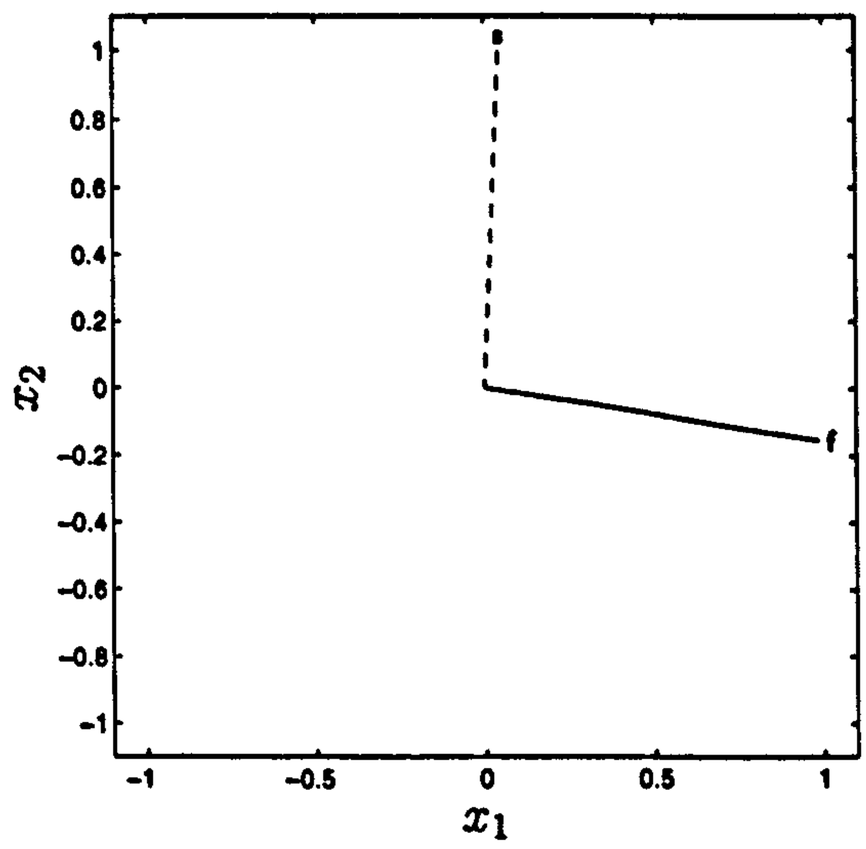
| *Model* | *Training MSE* | *Test MSE* |
|---|---|---|
| *general* | $0.93 \times 10^{-3}$ | $1.04 \times 10^{-3}$ |
| *restricted* | $0.99 \times 10^{-3}$ | $1.08 \times 10^{-3}$ |

*Table E.5: Modelling results for state space LMN structures with global learning.*

*The eigenvectors are depicted in Figure E.6. The fast eigenvectors of the general model vary significantly over a large range whereas the slow eigenvectors are very similar. Unstable eigenvector configurations as shown in Figure 2.6(b) are possible. Using only one eigenvector configuration for all local models, Figure E.6(b), excludes such instabilities, without significant loss of model performance.*




*(a) General LMN. The numbers correspond to the number of the local model.*




*(b) Restricted LMN.*

*Figure E.6: Eigenvector configuration. The slow eigenvectors are marked by 's' and plotted as dashed lines, the fast eigenvectors are denoted 'f' and plotted solid.*

**Example E.4.6** *In Example E.4.4, we compared local and global optimisation results for LMNs with 5 and with 10 units for modelling the heat transfer process. The results for global learning are repeated in Table E.6.*

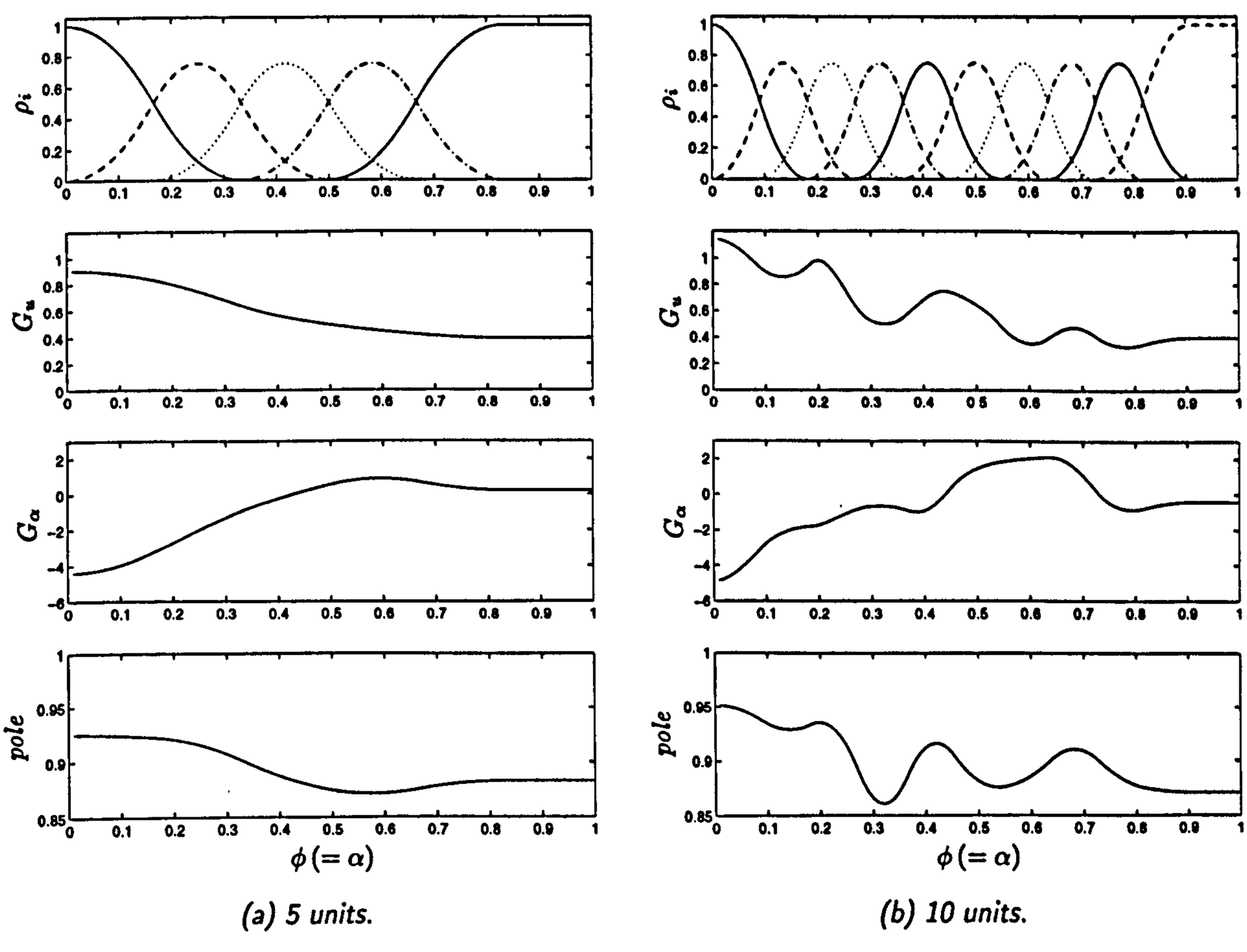
*For the model with 5 units, the training error is slightly larger than the test error, which suggests that the model performs as well on unknown data as it does on known data, i.e. it generalises well.*

| *No. units* | *Training MSE* | *Test MSE* |
|---|---|---|
| *5* | $1.28 \times 10^{-3}$ | $1.27 \times 10^{-3}$ |
| *10* | $1.10 \times 10^{-3}$ | $1.50 \times 10^{-3}$ |

*Table E.6: Modelling results for simulation model with global learning.*

*For the model with 10 units, the training error is significantly smaller than the test error (and also smaller than the training error of the smaller network). This suggests that the model does not generalise satisfactorily on unknown data.*

*Analysis of the properties of the LMNs for constant values of the scheduling variable $\phi$ provides further insight. The analysis results are shown in Figure E.7.*




*(a) 5 units.* *(b) 10 units.*

*Figure E.7: Static analysis for LMN structures. The top plots show the activations of the validity functions, $\rho_i$. In the middle plots, the steady state gains of the interpolated models from the voltage input to the output, $G_u$, and from the valve angle to the output, $G_\alpha$, are depicted. The bottom plots show the values of the pole of the interpolated models.*

*For the LMN with 5 units, Figure E.7(a), the properties change smoothly over the range of the scheduling variable which is in this case the valve angle. The gain of the plant from the voltage to the output, $G_u$, is large for small valve angles and decreases for larger angles. The gain from the angle to the output, $G_\alpha$, decreases with larger angles, i.e. the influence of the angle on the process is smaller for larger angles. The pole moves further inside the unit circle for increasing valve angle, i.e. the plant becomes faster. All these observations correspond to properties of the real system as discussed in Section E.1.*

*For the LMN with 10 units, Figure E.7(b), the properties change in a similar way to those of the LMN with 5 units. However, the changes are not smooth, and are sometimes rapid. These non-smooth changes certainly do not reflect properties of the real system. They suggest that the model is over-parameterised, as pointed out in the discussion on training and test errors.*

**Example E.4.7** *We illustrate the design concept of the LCN with the heat transfer process. The model structure is chosen as in Example E.4.4 on page 154. To simplify the controller structure, we include only the voltage as an input to the local models, i.e. the data vector of the local ARX model has the form*

$$\underline{\psi}_i(t) = [u(t - T_{d,i}),\, y(t - T_s)]^T . \tag{E.5}$$

*The valve angle is used as the scheduling variable. A model with 5 uniformly distributed validity functions is identified, using global optimisation and simulation model mode.*

*The local models of this LMN are then used to design a local controller network. The control specification is*

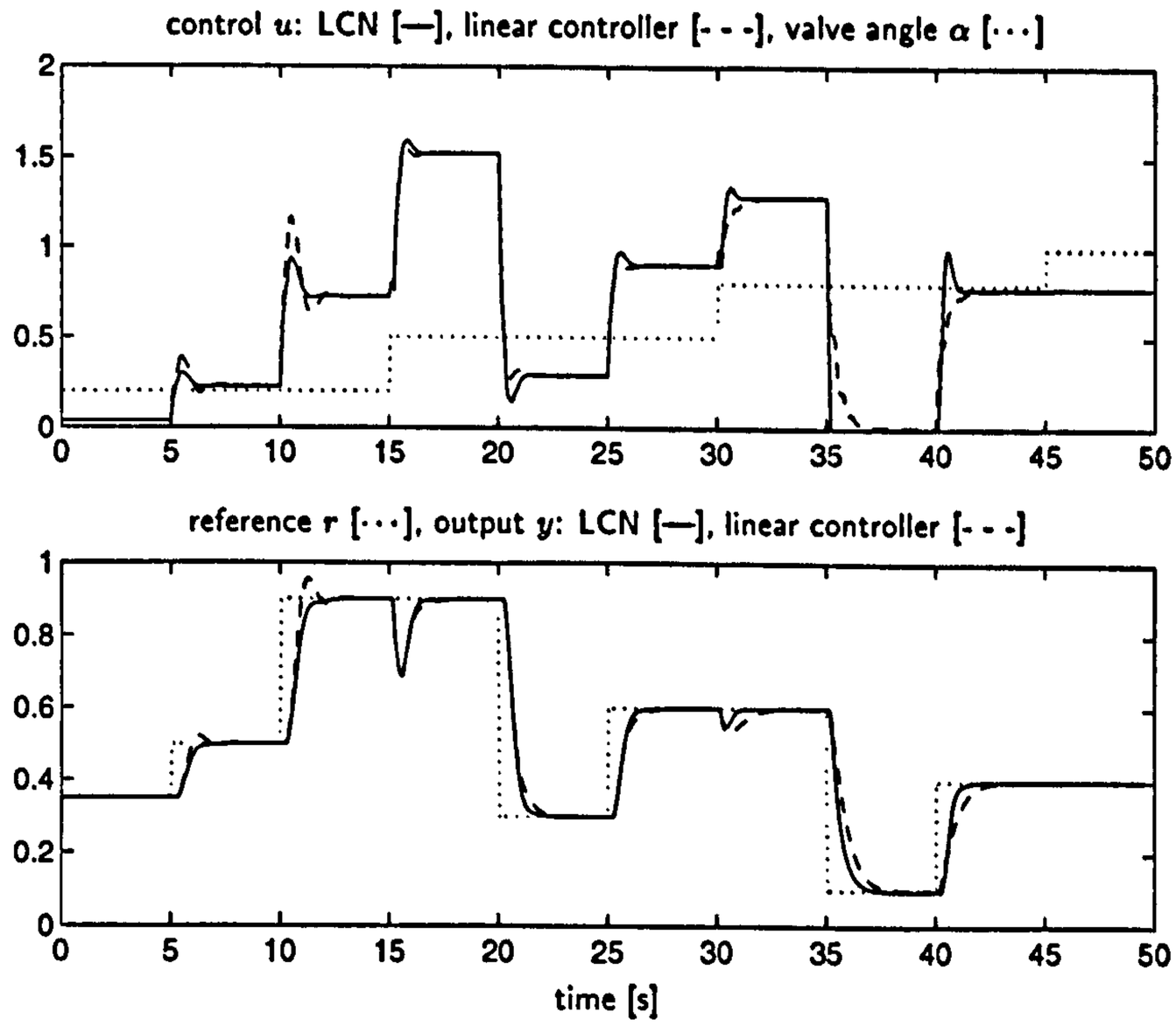
1. *no steady state error,*
2. *zero overshoot, and*
3. *a varying rise-time from* 700*ms for fully open valve (large angle) to* 1000*ms for minimal valve opening (small angle).*

*This specification takes account of the fact that the plant is faster for a larger valve angle. To meet the specification of zero steady state error, integral action is included in the controller.*

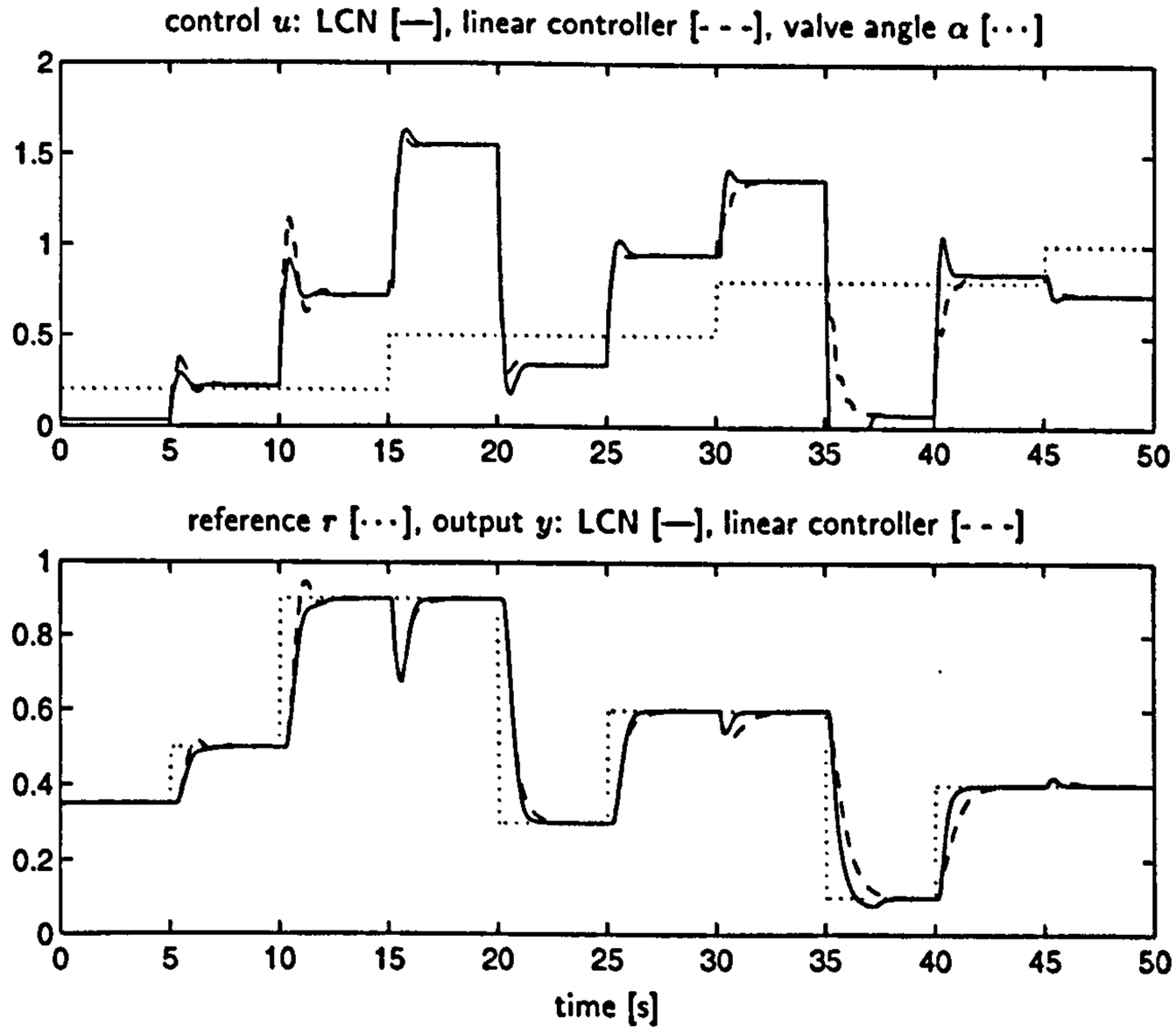
*To compare the performance of the non-linear LCN with a standard linear controller design, a single linear controller is used as a benchmark. The linear controller was designed for the interpolated plant at* $\alpha = 0.5$, *with a rise-time of* 900*ms.*

*The controllers are tested in simulations with the design model as the plant. Results are shown in Figure E.8(a).*

*To evaluate the robustness of the design to changes of the plant, the same controllers are tested in simulations to control the LMN model with 5 units from Example E.4.4. Note that*




(a) Control of the design model.




(b) Control of a changed plant.

Figure E.8: Simulation of LCN and linear controller to control a model of the heat transfer process.

*this model differs from the design model in that the valve angle $\alpha$ is also an input of the local models. The results are shown in Figure E.8(b).*

*The results show clearly that the Local Controller Network performs consistently for all operating conditions, and that the controller specifications are met for varying valve angles. The single linear controller performs well for medium and large valve angles. Its performance deteriorates when the valve angle is small as a significant overshoot can be observed for these operating conditions.*

*The performance of both controllers does not change significantly when a slightly different plant than the design plant is controlled. This indicates that the controller design is robust with respect to modelling uncertainties. A slight overshoot can be encountered at 37s for the LCN which is due to saturation of the control signal (which cannot become negative). The overshoot could be avoided by implementing anti-reset windup in the controllers (Franklin* et al. *1994).*

# F Nomenclature

## F.1 Notation

| | |
|---|---|
| $u$ | scalar variable $u$ |
| $\underline{u}$ | vector $u$ |
| $\{u\}$ | sequence $u$ |
| $A$ | matrix $A$ |
| $A^{-1}$ | inverse of matrix $A$ |
| $A^{+}$ | pseudoinverse of matrix $A$ |
| $A^{T}$ | transpose of matrix $A$ |
| $t$ | continuous time |
| $t_k$ | discrete time, $k$-th sampling point |
| $T_d$ | time delay |
| $T_s$ | sampling period |
| $q$ | time shift operator |
| $u$ | system input |
| $y$ | system output |
| $\hat{y}$ | estimate of $y$ |
| $y^o$ | value of $y$ at an operating point |
| $y^d$ | deviation of $y$ from the operating point |
| $\underline{x}$ | state vector |
| $\dot{\underline{x}}$ | derivative of $\underline{x}$ with respect to time |
| $\underline{x}_0$ | initial state at $t = 0$ |
| $\underline{x}^e$ | equilibrium state |
| $\mathbb{R}^n$ | Euclidean $n$-dimensional space |
| $\mathbb{R}^{n \times n}$ | Euclidean $n$-by-$n$-dimensional space |
| $\mathbb{C}$ | Complex space |

| | |
|---|---|
| $\Re$ | real part of a complex number |
| $\Im$ | imaginary part of a complex number |
| $J$ | optimisation criterion |
| $\gamma$ | weighting factor |
| $\underline{\theta}$ | parameter vector |
| $\underline{\theta}^*$ | optimum of $\underline{\theta}$ |
| $\underline{\hat{\theta}}$ | estimate of the parameter vector $\underline{\theta}$ |
| $\underline{\phi}$ | scheduling vector |
| $\underline{\psi}$ | input vector containing delayed system inputs and outputs |
| $\rho_i$ | $i$-th validity function |
| $f_i$ | $i$-th local model |
| $\xi$ | damping factor |
| $t_r$ | rise time |

## F.2 Abbreviations

| | |
|---|---|
| ARMAX | Auto-Regressive Moving Average model with eXogenous inputs |
| ARX | Auto-Regressive model with eXogenous inputs |
| ASMOD | Adaptive Spline Modeling of Observation Data |
| CE | Contractile Element |
| FES | Functional Electrical Stimulation |
| FTI | Force-Time Integral |
| FTIpP | (normalised) Force-Time Integral per Pulse |
| IPI | Inter-Pulse Interval |
| KBF | Kernel Basis Functions |
| LCN | Local Controller Network |
| LMN | Local Model Network |
| LPI | Linear Parameter–Invariant |
| LPV | Linear Parameter–Variant |
| MARS | Multivariate Adaptive Regression Splines |
| MSE | Mean Squared Error |
| MSPE | Mean Squared Prediction Error |
| MSSE | Mean Squared Simulation Error |
| NARX | Non-linear ARX |

| | |
|---|---|
| PE | Passive Element |
| SE | Series elastic Element |
| SMV | Skeletal Muscle Ventricle |
| ODE | Ordinary Differential Equation |
| OSP | Optimal Stimulation Pattern |

# Bibliography


Åström, K. J. and B. Wittenmark (1989). *Adaptive Control.* Addison–Wesley Publishing Company.

Åström, K. J. and B. Wittenmark (1990). *Computer-Controlled Systems. Theory and Design.* 2nd ed.. Prentice–Hall, Inc.. Englewood Cliffs, New Jersey.

Allin, J. and G. F. Inbar (1986). FNS control schemes for the upper limb. *IEEE Trans. Biomed. Eng.* **33**(9), 818–828.

Babuška, R. and H. B. Verbruggen (1997). Fuzzy set methods for local modelling and identification. Chap. 2. In: Murray-Smith and Johansen (1997*a*).

Back, A. D., A. C. Tsoi, B. G. Horne and C. L. Giles (1997). Alternative discrete-time operators and their application to nonlinear models. Technical Report UMIACS-TR-97-03. Institute for Advanced Computer Studies. University of Maryland, USA.

Bawa, P., A. Mannard and R. B. Stein (1976*a*). Effects of elastic loads on the contractions of cat soleus muscle. *Biol. Cybern.* **22**, 129–137.

Bawa, P., A. Mannard and R. B. Stein (1976*b*). Predictions and experimental test of a visco-eleastic muscle model using elastic and intertial loads. *Biol. Cybern.* **22**, 139–145.

Beck, J. V. and K. J. Arnold (1977). *Parameter Estimation in Engineering and Science.* John Wiley and Sons.

Bernotas, L. A., P. E. Crago and H. J. Chizeck (1986). A discrete-time model of electrically stimulated muscle. *IEEE Trans. Biomed. Eng.* **33**(9), 829–838.

Billings, S. A. and W. S. F. Voon (1987). Piecewise linear identification of non-linear systems. *Int. J. Control* **46**(1), 215–235.

Binder-Macleod, S. A. and C. B. Barker (1991). Use of a catch-like property of human skeletal muscle to reduce fatigue. *Muscle and Nerve* **14**, 850–857.

Binder-Macleod, S. A. and W. J. Barrish (1992). Force response of rat soleus muscle to variable-frequency train stimulation. *J. Neurophysiology* **68**(4), 1068–1078.

Bobet, J., R. B. Stein and M. N. Oguztoreli (1993). A linear time-varying model of force generation in skeletal muscle. *IEEE Trans. Biomed. Eng.* 40(10), 1000–1006.

Brockett, R. W. (1970). *Finite Dimensional Linear Systems.* John Wiley and Sons.

Brown, M. and C. Harris (1994). *Neurofuzzy Adaptive Modelling and Control.* Prentice Hall. Hemel-Hempstead, UK.

Chagas, A. C. P., L. F. P. Moreira, P. L. da Luz, G. P. Camarano, A. Leirner, N. A. G. Stolf and A. D. Jatene (1989). Stimulated preconditioned skeletal muscle cardiomyoplasty - an effective means of cardiac assist. *Circulation* 80(5), 202–208.

Chen, S. and S. A. Billings (1989). Representation of non-linear systems: the NARMAX model. *Int. J. Control* 49, 1013–1032.

Cooper, S. and J. C. Eccles (1930). The isometric response of mammalian muscles. *J. Physiol.* 69, 377–385.

de Boor, C. (1978). *A Practical Guide to Splines.* Vol. 27 of *Applied Mathematical Science Series.* Springer-Verlag. New York, Berlin.

Donaldson, N. de N., H. Gollee, K. J. Hunt, J. C. Jarvis and M. K. N. Kwende (1995). A radial basis function model of muscle stimulated with irregular inter-pulse intervals. *Medical Engineering & Physics* 17(6), 431–441.

Dorgan, S. J. and M. J. O'Malley (1996). A model for an artificially activated musculoskeletal system. pp. 91–94. In: Pedotti *et al.* (1996).

Dorgan, S. J. and M. J. O'Malley (1997). Nonlinear mathematical model of electrically stimulated skeletal muscle. *IEEE Trans. Rehab. Eng.* 5(2), 179–194.

Duchateau, J. and K. Hainaut (1986). Nonlinear summation of contractions in straited muscle. I. Twitch potentiation in human muscle; II. Potentiation of intracellular ca movements in single barnacle muscle fibres. *J. Muscle Research and Cell Motility* 7, 11–24.

Durfee, W. K. (1992). Model identification in neural prosthesis systems. pp. 58–87. In: Stein *et al.* (1992).

Durfee, W. K. and K. E. MacLean (1989). Methods for estimating isometric recruitment curves of electrically stimulated muscle. *IEEE Trans. Biomed. Eng.* 36(7), 654–667.

Durfee, W. K. and K. I. Palmer (1994). Estimation of force-activation, force-length, and force-velocity properties in isolated, electrically stimulated muscle. *IEEE Trans. Biomed. Eng.* 41(3), 205–216.

Franklin, G. F., J. D. Powell and A. Emami-Naeini (1994). *Feedback Control of Dynamic Systems*. 3rd ed.. Addison–Wesley Publishing Company.

Friedman, J. H. (1991). Multivariate adaptive regression splines. *The Annals of Statistics* **19**(1), 1–67.

Gawthrop, P. J. (1995). Continuous-time local state local model networks. In: *Proceedings of 1995 IEEE Conference on Systems, Man and Cybernetics*. Vancouver, British Columbia. pp. 852–857.

Gawthrop, P. J. (1996). Continuous-time local model networks. In: *Neural Adaptive Control Technology* (R. Żbikowski and K. J. Hunt, Eds.). Vol. 15 of *World Scientific Series in Robotics and Intelligent Systems,*. Chap. 2. World Scientific. Singapore.

Geman, S., E. Bienenstock and R. Doursat (1992). Neural networks and the bias/variance dilemma. *Neural Computation* 4(1), 1–58.

Gollee, H. and D. J. Murray-Smith (1997*a*). Analytical validation and restriction of local model networks for electrically stimulated muscle. In: *Workshop on Multiple Model Approaches to Modelling and Control*. Trondheim, Norway.

Gollee, H. and D. J. Murray-Smith (1997*b*). Validation of local model networks for electrically stimulated muscle. In: *Proc. International Conference on Engineering Applications of Neural Networks*. Stockholm, Sweden.

Gollee, H. and K. J. Hunt (1997). Nonlinear modelling and control of electrically stimulated muscle: a local model network approach. *Int. J. Control* **68**(6), 1259–1288.

Gollee, H., K. J. Hunt, N. de N. Donaldson and J. C. Jarvis (1997). Modelling of electrically stimulated muscle. Chap. 3. In: Murray-Smith and Johansen (1997*a*).

Gollee, H., K. J. Hunt, N. de N. Donaldson, J. C. Jarvis and M. K. N. Kwende (1994). A mathematical analogue of electrically stimulated muscle using local model networks. In: *Proc. 33rd IEEE Conf. on Decision and Control*. Vol. 2. Lake Buena Vista, Florida. pp. 1879–1880.

Graupe, D. and K. H. Kohn (1994). *Functional Electrical Stimulation for Ambulation by Paraplegics: twelve years of clinical observations and system studies*. Krieger Publishing Company.

Gray, H. (1995). *Gray's Anatomy: The Anatomical Basis of Medicine and Surgery*. 38th ed.. Churchill Livingstone. New York, Edinburgh.

Haber, R. and H. Unbehauen (1990). Structure identification of nonlinear dynamic systems — a survey on input/output approaches. *Automatica* **26**(4), 651–677.

Hahn, W. (1963). *Theory and Application of Liapunov's Direct Method.* Prentice-Hall, Inc.

Hambrecht, F. T. (1992). A brief history of neural prostheses for motor control of paralyzed extremities. pp. 3–14. In: Stein *et al.* (1992).

Hatze, H. (1977). A myocybernetic control model of skeletal muscle. *Biol. Cybern.* **25**, 103–119.

Hatze, H. (1978). A general myocybernetic control model of skeletal muscle. *Biol. Cybern.* **28**, 143–157.

Hatze, H. (1990). The charge-transfer model of myofilamentary interaction: Prediction of force enhancement and related myodynamic phenomena. pp. 24–45. In: Winters and Woo (1990).

Hill, A. V. (1938). The heat of shortening and the dynamic constants of muscle. *Proc. R. Soc. Lond. [Biol.]* **126**, 136–195.

Hlaváčková, K. and R. Neruda (1993). Radial basis function networks. *Neural Network World* **1**, 93–101.

Hlaváčková, K. (1995). An upper estimate of the error of approximation of continuous multivariable functions by KBF networks. In: *Proc. 3rd European Symposium on Artificial Neural Networks.* Brussels. pp. 333–340.

Hooper, T. L. and L. W. Stephenson (1991). Using skeletal muscle to assist the heart. *Br. Heart J.* **66**(4), 261–263.

Hunt, K. J. and T. A. Johansen (1997). Design and analysis of gain-scheduled control using local controller networks. *Int. J. Control* **66**(5), 619–651.

Hunt, K. J., M. Munih and N. Donaldson (1997). Feedback control of unsupported standing in paraplegia — Part I: optimal control approach. *IEEE Trans. Rehab. Eng.* **5**(4), 331–340.

Hunt, K. J., M. Munih, N. Donaldson and F. M. D. Barr (1998*a*). Investigation of the Hammerstein hypothesis in the modelling of electrically stimulated muscle. *IEEE Trans. Biomed. Eng.* To appear August.

Hunt, K. J., M. Munih, N. Donaldson and F. M. D. Barr (1998*b*). Optimal control of ankle joint moment: towards unsupported standing in paraplegia. *IEEE Trans. Automatic Control* **43**(6), 819–832.

Huxley, A. F. (1957). Muscle structure and theories of contraction. *Prog. Biophys. and Biophys. Chem.* **7**, 257–318.

Jacobs, R. A. (1995). Methods for combining experts' probability assessments. *Neural Computation* **7**(5), 867–888.

Jang, J. S. R. (1993). ANFIS - adaptive-network-based fuzzy inference system. *IEEE Trans. Systems, Man and Cybernetics* **23**(3), 665–685.

Jarvis, J. C. (1993). Power production and working capacity of rabbit tibialis anterior muscles after chronic electrical stimulation at 10 Hz. *J. Physiol.* **470**, 157–169.

Jarvis, J. C., H. Gollee, M. M. N. Kwende, S. Salmons and D. J. Murray-Smith (1996*a*). Towards an optimized muscle controller. pp. 123–128. In: Pedotti *et al.* (1996).

Jarvis, J. C., H. Sutherland and S. Salmons (1991). The effect of chronic electrical stimulation on the speed, power output and fatigue resistance of fast twitch skeletal muscle. *J. Muscle Research Cell Motility* **12**(1), 80–81.

Jarvis, J. C., H. Sutherland, C. N. Mayne, S. J. Gilroy and S. Salmons (1996*b*). Induction of a fast-oxidative phenotype by chronic muscle stimulation — mechanical and biochemical studies. *Am. J. Physiol.-Cell Physiol.* **39**(1), C306–C312.

Johansen, T. A. (1994). Operating regime based process modeling and identification. PhD thesis. Department of Engineering Cybernetics, Norwegian Institute of Technology, University of Trondheim. Norway.

Johansen, T. A. (1995). On the optimality of the Takagi-Sugeno-Kang fuzzy inference mechanism. In: *Proc. IEEE International Conference On Fuzzy Systems.* Vol. 1. pp. 97–102.

Johansen, T. A. and B. A. Foss (1992). A NARMAX model representation for adaptive control based on local models. *Modelling, Identification and Control* **13**(1), 25–39.

Johansen, T. A. and B. A. Foss (1993). Constructing NARMAX models using ARMAX models. *Int. J. Control* **58**, 1125–1153.

Johansen, T. A. and B. A. Foss (1995). Empirical modeling of a heat transfer process using local models and interpolation. In: *Proc. Amer. Control Conference.* Vol. 5. pp. 3654–3658.

Johansen, T. A. and R. Murray-Smith (1997). The operating regime approach to nonlinear modelling and control. Chap. 1. In: Murray-Smith and Johansen (1997*a*).

Johansen, T. A., K. J. Hunt, P. J. Gawthrop and H. Fritz (1998). Off-equilibrium linearisation and design of gain scheduled control with application to vehicle speed control. *Control Engineering Practice* **6**(2), 167–180.

Johansson, M. and A. Rantzer (1998). Computation of piecewise quadratic lyapunov functions for hybrid systems. *IEEE Trans. Automatic Control* **43**(4), 555–559.

Jordan, M. I. and R. A. Jacobs (1994). Hierarchical mixtures of experts and the EM algorithm. *Neural Computation* **6**(2), 181–214.

Jutten, Ch. and O. Fambon (1995). Pruning methods: A review. In: *Proc. 3rd European Symposium on Artificial Neural Networks*. Brussels. pp. 129–140.

Kailath, T. (1980). *Linear Systems*. Prentice–Hall, Inc.

Kalman, R. E. and J. E. Bertram (1960). Control system analysis and design via the 'second method' of Lyapunov. I Continuous-time systems. *Trans. ASME / J. Basic Eng.* pp. 371–393.

Karu, Z. Z., W. K. Durfee and A. M. Barzilai (1995). Reducing muscle fatigue in FES applications by stimulating with N-let pulse trains. *IEEE Trans. Biomed. Eng.* **42**(8), 809–817.

Kavli, T. (1993). ASMOD - an algorithm for adaptive spline modeling of observation data. *Int. J. Control* **58**(4), 947–967.

Kavli, T. and E. Weyer (1995). On ASMOD — an algorithm for empirical modelling using spline functions. In: *Neural Network Engineering in Dynamic Control Systems* (K. J. Hunt, G. R. Irwin and K. Warwick, Eds.). pp. 83–104. Advances in Industrial Control. Springer–Verlag. Berlin.

Kaymak, U. and R. Babuška (1995). Compatible cluster merging for fuzzy modeling. In: *Proceedings FUZZ–IEEE/IFES 95*. Yokohama, Japan. pp. 897–904.

Keynes, R. D. and D. J. Aidley (1991). *Nerve and Muscle*. 2nd ed.. Cambridge Univ. Press.

Kralj, A. R. and T. Bajd (1989). *Functional Electrical Stimulation: Standing and Walking after Spinal Cord Injury*. CRC Press.

Krishnapuram, R. and Chin-Pin Freg (1992). Fitting an unknown number of lines and planes to image data through compatible cluster merging. *Pattern Recognition* **25**(4), 385–400.

Kwende, M. M. N., J. C. Jarvis and S. Salmons (1995). The input–output relations of skeletal muscle. *Proc. R. Soc. Lond. [Biol.]* **261**, 193–201.

La Salle, J. and S. Lefschetz (1961). *Stability by Liapunov's Direct Method (With Applications)*. Vol. 4 of *Mathematics in Science and Engineering*. Academic Press.

Leontaritis, I. J. and S. A. Billings (1985*a*). Input output parametric models for non-linear systems. Part I: Deterministic non-linear systems. *Int. J. Control* **41**(2), 303–328.

Leontaritis, I. J. and S. A. Billings (1985*b*). Input output parametric models for non-linear systems. Part II: Stochastic non-linear systems. *Int. J. Control* **41**(2), 329–344.

Liberson, W. T., H. J. Holmquest, D. Scot and M. Dow (1961). Functional electrotherapy: Stimulation of peroneal nerve synchronized with the swing phase of the gait of hemiplegic patients. *Arch. Phys. Med. Rehabil.* **42**, 101–105.

Mannard, A. and R. B. Stein (1973). Determination of the frequency response of isometric soleus muscle in the cat using random nerve stimulation. *J. Physiol.* **229**, 275–296.

Marquardt, D. W. (1963). *Journal of the Society for Industrial and Applied Mathematics* **11**, 431–441.

Maxwell, D. J., M. H. Granat and R. H. Baxendale (1996). Novel stimulation strategies for the recruitment of paralysed muscle. pp. 115–122. In: Pedotti *et al.* (1996).

McMahon, T. A. (1984). *Muscles, Reflexes, and Locomotion.* Princeton University Press. Princeton, New Jersey.

Meilă, M. and M. I. Jordan (1997). Markov mixtures of experts. Chap. 5. In: Murray-Smith and Johansen (1997*a*).

Middleton, R. H. and G. C. Goodwin (1990). *Digital Control and Estimation. A Unified Approach.* Prentice–Hall, Inc.

Munih, M., N. Donaldson, K. J. Hunt and F. M. D. Barr (1997). Feedback control of unsupported standing in paraplegia — Part II: experimental results. *IEEE Trans. Rehab. Eng.* **5**(4), 341–352.

Murray-Smith, D. J. (1995). *Continuous System Simulation.* Chap. 9. Chapman & Hall. London.

Murray-Smith, R. (1994). A Local Model Network Approach to Nonlinear Modelling. PhD thesis. University of Strathclyde. Glasgow, UK.

Murray-Smith, R. and H. Gollee (1994). A constructive learning algorithm for local model networks. In: *Proc. IEEE Workshop on Computer-intensive methods in control and signal processing.* Prague, Czech Republic. pp. 21–29.

Murray-Smith, R. and Johansen, T. A., Eds. (1997*a*). *Multiple Model Approaches to Modelling and Control.* Taylor and Francis.

Murray-Smith, R. and T. A. Johansen (1995). Local learning in local model networks. In: *4th IEE Intern. Conf. on Artificial Neural Networks.* pp. 40–46.

Murray-Smith, R. and T. A. Johansen (1997*b*). Local learning in local model networks. Chap. 7. In: Murray-Smith and Johansen (1997*a*).

Parmiggiani, F. and R. B. Stein (1981). Nonlinear summation of contractions in cat muscles: II. Later facilitation and stiffness changes. *J. Gen. Physiol.* 78(3), 295–311.

Peckham, P. H. and M. W. Keith (1992). Motor prostheses for restoration of upper extremity function. pp. 162–187. In: Stein *et al.* (1992).

Pedotti, A., Ferrarin, M., Quintern, J. and Riener, R., Eds. (1996). *Neuroprosthetics: from Basic Research to Clinical Application.* Springer–Verlag. Berlin, Heidelberg, New–York.

Perkins, J., N. de N. Donaldson, A. C. Worley, V. Harper, P. Taylor, D. E. Wood and D. N. Rushton (1996). Initial results with a lumbar/sacral anterior root stimulator implant. pp. 623–634. In: Pedotti *et al.* (1996).

Pochettino, A., D. R. Anderson, R. L. Hammond, A. D. Spanta, E. Hohenhaus, H. Niinami, H. P. Lu, R. Ruggiero, T. L. Hooper, M. Baars, C. Devireddy and L. W. Stephenson (1991). Skeletal muscle ventricles - a promising treatment option for heart-failure. *J. Cardiac Surgery* 6(1), 145–153.

Press, W. H., S. A. Teukolsky, W. T. Vetterling and B. P. Flannery (1992). *Numerical Recipes (C): The Art of Scientific Computing.* 2nd ed.. Cambridge University Press. UK.

Reed, R. (1994). Pruning algorithms: A survey. *IEEE Trans. Neural Networks* 4(5), 740–747.

Riener, R., J. Quintern, E. Psaier and G. Schmidt (1996). Physiologically based multi-input model of muscle activation. pp. 95–114. In: Pedotti *et al.* (1996).

Ronco, E. (1997). Incremental Polynomial Controller Networks: two self-organising non-linear controllers. PhD thesis. Dept. of Mechanical Engineering, University of Glasgow. Scotland, UK.

Sagawa, K. (1973). The circulation and its control I: Mechanical properties of the cardiovascular system. In: *Engineering Principles in Physiology* (J. H. U. Brown and D. S. Gann, Eds.). Vol. II. Chap. 14, pp. 49–71. Academic Press.

Salmons, S. and J. C. Jarvis (1992). Cardiac assistance from skeletal muscle: a critical appraisal of the various approaches. *Br. Heart J.* 68(3), 333–338.

Schultheiss, J., L. del Re and F. Kraus (1997). Mathematical modeling of electrically stimulated muscle. In: *Proc. IMACS Symposium on Mathematical Modelling* (I. Troch and F. Breitenecker, Eds.). Technical University Vienna. Austria. pp. 383–388.

Shamma, J. S. (1988). Analysis and design of gain scheduled control systems. PhD thesis. MIT. Cambridge, Massachusetts.

Shamma, J. S. and M. Athans (1992). Gain scheduling: Potential hazards and possible remedies. *IEEE Control Systems Magazine* **12**(3), 101–107.

Shorten, R. (1996). A Study of Hybrid Dynamical Systems with Application to Automobile Control. PhD thesis. Department of Electrical and Electronic Engineering, University College Dublin. Ireland.

Shorten, R. and R. Murray-Smith (1997). Side-effects of normalising basis functions. Chap. 8. In: Murray-Smith and Johansen (1997*a*).

Shorten, R. N. and K. S. Narendra (1998). Using modal analysis to analyse the stability of a class of hybrid system. *IEE Computing and Control Engineering Journal.* to appear.

Shorten, R., R. Murray-Smith, R. Bjørgan and H. Gollee (1998). On the interpretation of local models in blended multiple model structures. *Int. J. Control.* to appear.

Shue, G., P. E. Crago and H. J. Chizeck (1995). Muscle-joint models incorporating activation dynamics, moment-angle, and moment-velocity properties. *IEEE Trans. Biomed. Eng.* **42**(2), 212–223.

Skeppstedt, A., L. Ljung and M. Millnert (1992). Construction of composite models from observed data. *Int. J. Control* **55**(1), 141–152.

Slotine, J.-J. E. and W. Li (1991). *Applied Nonlinear Control.* Prentice–Hall, Inc.. Englewood Cliffs, New Jersey.

Söderman, U. and J.-E. Strömberg (1993). The conceptual side of mode switching. In: *Proc. IEEE Conf. Systems, Man and Cybernetics.* Le Touquet, France. pp. 245–250.

Solomonow, M. (1992). Biomechanics and physiology of a practical functional neuromuscular stimulation powered walking orthosis for paraplegics. pp. 202–232. In: Stein *et al.* (1992).

Stein, R. B. and F. Parmiggiani (1981). Nonlinear summation of contractions in cat muscles: I. Early depression. *J. Gen. Physiol.* **78**(3), 277–293.

Stein, R. B., Peckham, P. H. and Popović, D. P., Eds. (1992). *Neural Prostheses. Replacing Motor Function After Disease or Disability.* Oxford University Press.

Sugeno, M. and G. T. Kang (1988). Structure identification of fuzzy model. *Fuzzy Sets and Systems* **28**(1), 15–33.

Sugeno, M. and K. Tanaka (1991). Successive identification of a fuzzy model and its applications to prediction of a complex system. *Fuzzy Sets and Systems* **42**(3), 315–334.

Takagi, T. and M. Sugeno (1985). Fuzzy identification of systems and its application to modeling and control. *IEEE Trans. Systems, Man and Cybernetics* **15**(1), 116–132.

Timmons, W. D. (1995). Cardiovascular models and control. In: *The Biomedical Engineering Handbook* (J. D. Bronzino, Ed.). Chap. 158, pp. 2386–2403. CRC Press.

Unbehauen, H. and G. P. Rao (1987). *Identification of Continuous Systems*. Vol. 10 of *North-Holland Systems and Control Series*. North-Holland.

Vidyasagar, M. (1993). *Nonlinear Systems Analysis*. 2nd ed.. Prentice–Hall, Inc.

Weiss, S. M. and C. A. Kulikowski (1991). *Computer Systems that Learn*. Morgan Kaufmann. San Mateo, California.

Westerhof, N., G. Elzinga, P. Sipkema and G. C. van den Bos (1977). Quantitative analysis of the arterial system and heart by means of pressure-flow relations. In: *Cardiovascular Flow Dynamics* (N. H. C. Hwang and N. A. Normann, Eds.). Chap. 11, pp. 403–438. Univ. Park Press. Baltimore.

Weyer, E. and T. Kavli (1997). Theoretical properties of the ASMOD algorithm for empirical modelling. *Int. J. Control* **67**(5), 767–789.

Winters, J. M. and L. Stark (1987). Muscle models: What is gained and what is lost by varying model complexity. *Biol. Cybern.* **55**, 403–420.

Winters, J. M. and Woo, S. L-Y., Eds. (1990). *Multiple Muscle Systems. Biomechanics and Movement Organization*. Springer–Verlag.

Zahalak, G. I. (1992). An overview of muscle modeling. pp. 17–57. In: Stein *et al.* (1992).

Zajac, F. E. and J. L. Young (1980). Properties of stimulus trains producing maximum tension-time area per pulse from single motor units in medial gastrocnemius muscle of the cat. *J. Neurophysiol.* **43**, 1206–1220.

Zames, G. (1966*a*). On the input-output stability of time-varying nonlinear feedback systems. Part I: Conditions derived using concepts of loop gain, conicity, and positivity. *IEEE Trans. Automatic Control* **11**(2), 228–238.

Zames, G. (1966*b*). On the input-output stability of time-varying nonlinear feedback systems. Part II: Conditions involving circles in the frequency plane and sector nonlinearities. *IEEE Trans. Automatic Control* **11**(3), 465–476.

Zhang, X.-C., A. Visala, A. Halme and P. Linko (1994). Functional state modelling approach for bioprocesses: Local models for aerobic yeast growth processes. *J. Process Control* **4**, 127–134.