Control of Depth of Anaesthesia using locally weighted learning methods

André Dionísio Falcão

Dissertação para obtenção do Grau de Mestre em Engenharia Electrotécnica e de Computadores

Júri
Presidente: Professor João M. L. de Miranda Lemos
Orientador: Professor Alexandre J. M. Bernardino
Co-Orientador: Vogais:

October 2009
Acknowledgments

I want to thank Professor João Lemos and Professor Alexandre Bernardino for all the availability and support demonstrated during the development of this work. And FCT for the research grant provided.

This dissertation was performed at INESC-ID in the framework of project PTDC/EEA-ACR/69288/2006 "IDeA - "Integrated Design for Automation of Anaesthesia"
Abstract

This dissertation addresses the control of depth of anaesthesia (DoA) with a novel approach. For control purposes DoA is modeled as a non-linear dynamic system. The approach consists in approximating the non-linear system by a set of locally linear systems obtained from one of two locally weighted learning methods: RFWR (Receptive fields weighted regression) [1] or its most recent development LWPR (Locally weighted projection regression) [2][3]. The estimated local models are then used to synthesize local controllers that through a controller fusion strategy, provide control to the the non-linear system.

The main motivation for the proposed control strategy is to solve a real life problem involving the automatic administration of a drug that induces a certain degree of unconsciousness in a patient during surgery.

The applications of the algorithms to the problem considered are explained and illustrated by simulation.

Keywords

Control, Adaptation, Learning, Anaesthesia, Biomedical Engineering, Multiple Models.
Resumo

Esta dissertação apresenta uma nova abordagem ao controlo da profundidade de anestesia (DoA). Para efeitos de controlo o problema da profundidade de anestesia é modelado como um sistema dinâmico não linear. A nova abordagem consiste em aproximar o sistema não-linear por um conjunto de sistemas lineares locais obtidos através de um método de aprendizagem local: RFWR (Receptive Fields Weighted Regression) [1] ou o seu desenvolvimento mais recente LWPR (Locally weighted projection regression) [2][3]. Os modelos locais estimados são depois usados para sintetizar controladores locais que através de uma estratégia de fusão de controladores irão controlar o sistema não-linear.

A motivação principal para a estratégia de controlo proposta, é resolver um problema da vida real relacionado com a administração de um fármaco que induz um certo grau de insconsciência num paciente durante uma cirurgia.

As aplicações dos algoritmos ao problema considerado são explicados e ilustrados através de simulação.

Palavras Chave

Controlo, Adaptação, Aprendizagem, Anestesia, Engenharia Biomédica, Modelos Múltiplos.
# Contents

1 **Introduction** .......................... 1
   1.1 Context and Motivation .................. 3
   1.2 State Of The Art ......................... 4
   1.3 Main Objectives and Contributions ........... 6
   1.4 Dissertation Outline ..................... 6

2 **The DoA Model** ......................... 7
   2.1 PK model .................................. 9
   2.2 Effect Compartment ....................... 10
   2.3 Interaction Model ......................... 10
   2.4 Global model .............................. 11

3 **Locally weighted learning methods** ...... 13
   3.1 General description ........................ 15
   3.2 LWL internal functioning ................. 18
      3.2.1 Linear model parameters learning ......... 18
         3.2.1.A RFWR ............................ 18
         3.2.1.B LWPR ............................ 20
      3.2.2 Adding and pruning RFs ................ 22
      3.2.3 Distance metric learning ............... 22
      3.2.4 Algorithm Pseudo-Code ................ 23
         3.2.4.A Update Stage ...................... 23
         3.2.4.B Prediction Stage .................. 24

4 **Control Strategy** ...................... 25
   4.1 Model estimation using LWL methods ........... 27
   4.2 Control Setup ................................ 28
      4.2.1 Controller Design ..................... 28
         4.2.1.A State Space model derivation .......... 30
         4.2.1.B Local controller synthesis .......... 30
## List of Figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1</td>
<td>General structure for the DoA model</td>
<td>9</td>
</tr>
<tr>
<td>2.2</td>
<td>Three compartment PK model</td>
<td>9</td>
</tr>
<tr>
<td>3.1</td>
<td>Linearization of a function with $D \subset \mathbb{R}^n$ through three local models (red) and their associated gaussian kernels (blue)</td>
<td>15</td>
</tr>
<tr>
<td>3.2</td>
<td>LWL prediction schematic</td>
<td>17</td>
</tr>
<tr>
<td>3.3</td>
<td>RF update stage for LWPR</td>
<td>20</td>
</tr>
<tr>
<td>3.4</td>
<td>Projection of query point $x_q$ to $r$ projection directions</td>
<td>21</td>
</tr>
<tr>
<td>3.5</td>
<td>Derivative of projections in relation to mean zero input space $I$ is the identity matrix</td>
<td>22</td>
</tr>
<tr>
<td>4.1</td>
<td>Schematic of the global model obtained with LWL methods</td>
<td>28</td>
</tr>
<tr>
<td>4.2</td>
<td>Augmented model for control</td>
<td>29</td>
</tr>
<tr>
<td>4.3</td>
<td>The controller block</td>
<td>29</td>
</tr>
<tr>
<td>4.4</td>
<td>Local control scheme</td>
<td>30</td>
</tr>
<tr>
<td>4.5</td>
<td>Overall view of the control strategy. The controller block synchronized with the learning algorithm and the common integrator with anti wind-up</td>
<td>32</td>
</tr>
<tr>
<td>4.6</td>
<td>Anti windup with back-calculation</td>
<td>33</td>
</tr>
<tr>
<td>5.1</td>
<td>Training input and output data from 29 models</td>
<td>38</td>
</tr>
<tr>
<td>5.2</td>
<td>Simulation for $d_i=1E+001 \ d_o=1E+001 \ \lambda = 1$</td>
<td>40</td>
</tr>
<tr>
<td>5.3</td>
<td>Initial manual control and the respective output results for the 8 test models</td>
<td>41</td>
</tr>
<tr>
<td>5.4</td>
<td>Simulation result for test model 1 with $d_o = 1 \times 10^{-1}$ $d_i = 1 \times 10^{-1}$ and $\lambda = 0.999$</td>
<td>42</td>
</tr>
<tr>
<td>5.5</td>
<td>Remifentanil model disturbance</td>
<td>42</td>
</tr>
<tr>
<td>5.6</td>
<td>Simulation results for plant input and output whit $R = 100 \ \ f = 3$ for test model 1</td>
<td>43</td>
</tr>
<tr>
<td>5.7</td>
<td>Normalized weights evolution for test model 1 with $R=100 \ \ f = 3$</td>
<td>44</td>
</tr>
<tr>
<td>5.8</td>
<td>Plant input and output with a wide range reference and $R = 100 \ \ f = 3$</td>
<td>44</td>
</tr>
<tr>
<td>5.9</td>
<td>Normalized weights with wide range reference with $R = 100$ and $f = 3$</td>
<td>45</td>
</tr>
<tr>
<td>5.10</td>
<td>Plant input and output with gaussian noise $\sigma = 1$ and $R = 100 \ \ f = 3$</td>
<td>45</td>
</tr>
<tr>
<td>5.11</td>
<td>Normalized weight evolution with gaussian noise $\sigma = 1$ and $R = 100 \ \ f = 3$</td>
<td>45</td>
</tr>
<tr>
<td>5.12</td>
<td>Results with controller block update turned off</td>
<td>46</td>
</tr>
</tbody>
</table>
## List of Tables

<table>
<thead>
<tr>
<th>Table</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.1</td>
<td>PKPD model fixed parameters</td>
<td>37</td>
</tr>
<tr>
<td>5.2</td>
<td>TPI(%) for different R and f´s</td>
<td>43</td>
</tr>
<tr>
<td>D.1</td>
<td>Mean squared error for $\lambda = 1$ (a -1 value indicates total prediction failure)</td>
<td>67</td>
</tr>
<tr>
<td>D.2</td>
<td>Mean squared error for $\lambda = 0.999$ (a -1 value indicates total prediction failure)</td>
<td>67</td>
</tr>
<tr>
<td>D.3</td>
<td>Mean squared error for $\lambda = 0.995$ (a -1 value indicates total prediction failure)</td>
<td>68</td>
</tr>
<tr>
<td>D.4</td>
<td>Mean squared error for $\lambda = 0.950$ (a -1 value indicates total prediction failure)</td>
<td>68</td>
</tr>
<tr>
<td>D.5</td>
<td>Number of RFs</td>
<td>69</td>
</tr>
<tr>
<td>D.6</td>
<td>Number of trustworthy RFs</td>
<td>69</td>
</tr>
<tr>
<td>D.7</td>
<td>Prediction failure (%) for $\lambda = 1$</td>
<td>70</td>
</tr>
<tr>
<td>D.8</td>
<td>Prediction failure (%) for $\lambda = 0.999$</td>
<td>70</td>
</tr>
<tr>
<td>D.9</td>
<td>Prediction failure (%) for $\lambda = 0.995$</td>
<td>71</td>
</tr>
<tr>
<td>D.10</td>
<td>Prediction failure (%) for $\lambda = 0.950$</td>
<td>71</td>
</tr>
<tr>
<td>D.11</td>
<td>Tracking performance index(%) for $\lambda = 1$</td>
<td>72</td>
</tr>
<tr>
<td>D.12</td>
<td>Tracking performance index(%) for $\lambda = 0.999$</td>
<td>72</td>
</tr>
<tr>
<td>D.13</td>
<td>Tracking performance index(%) for $\lambda = 0.995$</td>
<td>73</td>
</tr>
</tbody>
</table>
List of Acronyms

LWL  Locally Weighted Learning
RFWR  Receptive Fields Weighted Regression
LWPR  Locally Weighted Projection Regression
DoA  Depth of Anaesthesia
MUSMAR  Multi Step Multi Variable Adaptive Regulator
SMMAC  Switched Multiple Model Adaptive Control
EEG  Electroencephalogram
EMG  Electromyogram
BIS  Bi-Spectral Index
PK  Pharmaco-Kinetic
PKPD  Pharmaco-Kinetic Pharmaco-Dynamic
TCI  Target Control Infusion
PID  Proportion Integral Derivative
RF  Receptive Field
VIF  Variance Inflation Factor
PLS  Partial Least Squares
MSE  Mean Squared Error
TPI  Tracking Performance Index
# Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1 Context and Motivation</td>
<td>3</td>
</tr>
<tr>
<td>1.2 State Of The Art</td>
<td>4</td>
</tr>
<tr>
<td>1.3 Main Objectives and Contributions</td>
<td>6</td>
</tr>
<tr>
<td>1.4 Dissertation Outline</td>
<td>6</td>
</tr>
</tbody>
</table>
1.1 Context and Motivation

This dissertation is presented in the scope of project IDeA (http://ramses.inesc.pt/IDEA/), a multidisciplinary research project with the objective of developing an autonomous system to administer anaesthesia in a patient subject to surgery.

The introduction of intravenous anesthetic drugs in the mid 20th century revolutionized the medical specialty of anesthesia. Since then, a number of increasingly effective drugs where introduced that act within minutes and able to block specific human body mechanisms such as cognition, awareness, memory, stress response or muscle movement [4]. This fact allied to the computer revolution beginning in the 1980’s that brought actuators, such as computer controlled syringes, sensors such as Electroencephalogram (EEG), and computer control theory, made anaesthesia automation an active research topic.

Three main motivations arise when talking about anaesthesia automation. The first is precision: an automatic control scheme may have, a considerably faster response time than manual control and will in theory provide a better regulation behavior. Precision is essential because it allows less drug underdosage or overdosage problems which are related to a higher probability of patient complications during or after surgery.

Allied to precision comes the economical motivation since precise regulation means less unnecessary drug administration. In terms of drug cost this is not an important issue. However anaesthesia automation has potentially a major economic impact in the resolution of the periods in which surgery rooms are needed for a given patient.

The third motivation and possibly the most important is to provide an anesthetist with an "autopilot" for the repetitive and time consuming task of continuous drug administration, thereby allowing the practitioner to concentrate on other high level tasks and reducing the probability of human error.

General anaesthesia can be divided into three main components each related to a specific physical effect:

1. **Areflexia** – Reflex movement must be avoided. Usually, drugs like atracurium, are administered to induce and maintain a certain level of paralysis.

2. **Unconsciousness** – In many cases to avoid stress the patient must not be aware of what is happening during surgery. A certain level of unconsciousness is achieved and maintained through the administration of drugs like propofol. The drug is administered intravenously, being adequate to automatic control.

3. **Analgesia** – the patient should not feel any noxious stimuli, commonly known as "pain". To suppress it, analgesic drugs like remifentanil, are administered.
1. Introduction

To achieve a fully automated system all three components must be controlled. However most of the work and advances done so far are concentrated on the areflexia and unconsciousness components because they can both be measured through Electromyogram (EMG) and Bi-Spectral Index (BIS) respectively (other sensor are available as well). In opposition "pain" level measurement involves cross referencing data coming from multiple sensors, such as, EEG or cardiovascular functions and also some amount of empirical evaluation from the practitioner based on work experience.

Even for the measurable components, devising an efficient and robust control strategy for such a complex biological system as the human body constitutes a hard task. Two main factors pose a problem:

- Several mathematical patient models were derived in extensive previous work [5] [6] [7] but are, as expected, non-linear.
- High patient to patient variability exists leading to high parametric uncertainty in the patient models.

For the unconsciousness component also known as Depth of Anaesthesia (DoA) a third problem adds an additional difficulty. The drug used to suppress pain, remifentanil, interacts with propofol changing its effect.

Another aspect worth mentioning is related to the patient model structure itself. As described in compartmental models are normally used as they possess a simpler structure and thus less parameters to be determined. The downside to the previously mentioned choice is a loss in model accuracy, that could degrade performance in control strategies relying on such a structure.

This work will study a different approach to the DoA control problem using two Locally Weighted Learning (LWL) algorithms to learn the patients dynamic behavior: Receptive Fields Weighted Regression (RFWR) [1]; or its most recent development Locally Weighted Projection Regression (LWPR) [2] [3]. Using LWL methods for DoA presents two main advantages:

- It provides the possibility of online adaptation, which is essential to tackle the problem of variability from patient to patient and parameter variation due to drug interaction.
- No model structure is assumed. LWL methods constitute a non-parametric approach.

1.2 State Of The Art

Several control strategies have been devised in the past decades to tackle the problem of DoA automation. Between the first ones to appear were open loop strategies with the help of Target Control Infusion (TCI) systems [8] [9]. These approaches use the linear dynamic model that describes drug diffusion through the human body, the Pharmaco-Kinetic (PK) model, to calculate the drug infusion rate necessary to target and maintain a desired drug concentration in the blood.
1.2 State Of The Art

As seen in [10][11], TCI systems are nowadays preferred to manual drug infusion and are implemented in commercial infusion pumps for propofol and other drugs.

The major drawback of such a system is his open loop nature. Practitioner intervention is still necessary to specify the desired concentration to reach a determinate unconsciousness state, the relation between the concentration and the effect is nonlinear [5] and thus continuous readjustments are needed.

Since the drug interaction problem in DoA made closed loop control difficult, a wide range of control strategies exist ranging from simple Proportion Integral Derivative (PID) to multiple model control.

One approach made by Absalom et al. [12] used TCI as the control actuator and a PID controller to obtain the necessary drug concentration to maintain a prescribed BIS level. The results were acceptable although in some patients the control performance was far from optimal. An improved version of the algorithm was presented in [13] that proved to be more efficient, but the use of a PID continued to created some stability problems.

Other class of approaches are model based. A Pharmaco-Kinetic Pharmaco-Dynamic (PKPD) patient model structure is assumed and parameters are determined online for each patient. A good example of this type of approach is accomplished by Syrus et al. [14][15], where the model is learned individually for each patient during the initial drug infusion administered to reach a predetermined unconsciousness level (induction phase). After the determination of the initial PKPD profile, parameters continued to be readjusted during surgery. In [15] the Syrus et al. strategy is compared to manual practices performed during surgery and was concluded to be clinically acceptable for general anaesthesia.

Another two solutions were also suggested in the scope of project IDeA both using multiple model control strategies.

The first uses the Multi Step Multi Variable Adaptive Regulator (MUSMAR) control strategy [16], several simple models or predictors are identified online. The predictors are then used to synthesize controllers that drive the system during a sliding future time horizon. In this case the remifentanil infusion rate is integrated in control to be used as an accessible disturbance. Good simulation results were obtained with this strategy [16].

The second, known as Switched Multiple Model Adaptive Control (SMMAC) [17] assumes the existence of models defining linearizations around predefined setpoints for the PKPD model. These models are used to generate a bank of controllers. A supervisor is then designed to switch between controllers according to the current working point of the system.

As for the LWL methods used in this work, they have mostly been used in static mapping tasks especially in the robotics and vision research areas with success [18]. Some research has also been done in using these methods for control [19][20][21] although few publications still exist, and none associated with the anaesthesia field of research.
1. Introduction

1.3 Main Objectives and Contributions

The main objective for this work is to develop and test a control strategy using one of the LWL methods. The main idea for the control strategy is to use RFWR or LWPR to build, resorting to training, a global model constituted by several locally linear models. A controller is then derived for each locally linear model and a closed loop controller fusion strategy is used to drive the system. The main contribution consists in the application of the methods mentioned and its demonstration in simulation with realistic models.

1.4 Dissertation Outline

The dissertation is divided in five main chapters. In chapter two an introduction is done to the DoA model used to simulate patient behavior and generate training data for the LWL methods. In chapter three the general concept of LWL is explained followed by a detailed description on the internal functioning of RFWR and LWPR. Chapter four will explain the idea for using LWL for control and describe the control strategy. In chapter five tests are made on the control strategy defined in chapter four. Conclusions are drawn on chapter six.
2

The DoA Model

Contents

2.1 PK model .................................................. 9
2.2 Effect Compartment ..................................... 10
2.3 Interaction Model ........................................ 10
2.4 Global model ............................................. 11
The DoA model [5] is divided into three main parts (figure 2.1). The PK model describes how the hypnotic drug (propofol) administered with an infusion rate \( r \) spreads through the blood and tissue resulting in a plasma concentration \( C_p \). The second part, the effect compartment, models the transport of a certain plasma concentration \( C_p \) to the brain resulting in a concentration effect \( C_e \). Finally, the third part, the interaction model, describes the actual effect the drug has on the body measured by BIS. The actual effect of the hypnotic is influenced by the effect concentration of another drug remifentanil, used as analgesic.

### 2.1 PK model

![Three compartment PK model](image)

A three compartment model is used to emulate the patient dynamics. Compartment one represents blood and highly irrigated organs (heart, liver, kidney for example) and the other two represent the remaining organs and anatomy elements such as muscles or fat. As apparent in figure 2.2, compartments interact with each other, the interaction being modeled by the transfer coefficients \( k_{12}, k_{21}, k_{13}, k_{31} \). Coefficient \( k_{10} \) is added to model drug elimination due to metabolism. The model is then written in a continuous state space representation [5] as

\[
\dot{x} = A_{PK} x + B_{PK} r \\
C_p = C_{PK} x
\]

\[
A_{PK} = \begin{bmatrix}
-k_{10} - k_{12} - k_{13} & -k_{21} & k_{31} \\
k_{12} & -k_{21} & 0 \\
k_{13} & 0 & -k_{31}
\end{bmatrix},
B_{PK} = \begin{bmatrix}
10^4 \\
0 \\
0
\end{bmatrix},
C_{PK} = \begin{bmatrix}
10^4 & 0 & 0
\end{bmatrix}
\]
2. The DoA Model

where \( v_1 \) represents the volume of compartment one for a given patient, and is obtained by:

\[
v_1 = patw \cdot V_c
\] (2.2)

the parameter \( patw \) being the patient weight in Kilograms and \( V_c \) a coefficient which represents the volume of compartment one per unit weight.

2.2 Effect Compartment

The effect compartment relates the \textit{propofol} concentration in a patient \( C_p \) with the effect concentration \( C_e \). It is modeled by a low-pass filter \cite{5}

\[
C_e = \frac{1}{k_{eo} + 1} C_p
\] (2.3)

or in state-space representation

\[
\dot{x} = A_{EC} x + B_{EC} C_p
\]

\[
C_e = C_{EC} x
\]

with \( A_{EC} = [-k_{eo}] \) \( B_{EC} = [k_{eo}] C_{EC} = 1 \)

2.3 Interaction Model

The drug interaction between \textit{propofol} and \textit{remifentanil} is modeled by the Hill equation relating the normalized effect concentrations \( U_{prop} \) and \( U_{remi} \) with the level of unconsciousness \( E \)

\[
E = E_0 \left( 1 - \frac{U_{prop} + U_{remi}}{1 + U_{prop} + U_{remi}} \right)
\] (2.5)

The normalized effect concentrations are obtained dividing each concentration by the respective concentration at half the maximum effect \( U_{prop} = \frac{C_{prop}}{C_{prop_{50}}} \) and \( U_{remi} = \frac{C_{remi}}{C_{remi_{50}}} \). The parameter \( E_0 \) is the level of unconsciousness without drug effect.

Equation (2.5) shows a simplified model for the Hill equation. Usually an improved model is used that does not assume a purely additive relation between \( U_{prop} \) and \( U_{remi} \). Defining \( U_{50} = 1 - \beta \theta + \beta \theta^2 \) where \( \theta = \frac{U_{prop}}{U_{prop} + U_{remi}} \) the improved Hill equation is given by:

\[
E = E_0 \left( 1 - \frac{[(U_{prop} + U_{remi})/U_{50}\theta]^{\gamma}}{1 + [(U_{prop} + U_{remi})/U_{50}\theta]^{\gamma}} \right)
\] (2.6)

the additional parameter \( \gamma \) defines the steepness of the concentration-response relation. The improved Hill equation model (2.6) will be the one used in this work.
2.4 Global model

The global model has a Wiener structure. It is defined mathematically as a linear part resulting from the series of the PK model (2.2) and the effect compartment (2.3):

\[
\begin{align*}
\dot{x} &= \begin{bmatrix} A_{PK} & 0 \\ B_{EC}C_{PK} & A_{EC} \end{bmatrix} x + \begin{bmatrix} B_{PK} \\ 0 \end{bmatrix} r \\
C_{e} &= \begin{bmatrix} 0 & C_{EC} \end{bmatrix} x
\end{align*}
\] (2.7)

constituting a 4th order system.

The linear part that accounts for the "memory" of the system is in series with the non-linearity modeled by the Hill equation. The nonlinearity changes due to the influence of the remifentanil concentration (2.6).
2. The DoA Model
Locally weighted learning methods

Contents

3.1 General description .................................................. 15
3.2 LWL internal functioning ............................................. 18
3.1 General description

The objective for this chapter is to describe the LWL methods used in this work. They will be used to perform the non-linear system identification necessary for control.

3.1 General description

Figure 3.1: Linearization of a function with $D \subset \mathbb{R}^n$ through three local models (red) and their associated gaussian kernels (blue)

The main idea behind the two LWL methods described in this work is to approximate any given function by a variable set of locally linear functions, or models. Consider the smooth generic function:

$$y = f(x)$$

(3.1)

where $x$ is a vector of dimension $n$ $[x_1 \ldots x_n]^T$.

If the domain of this function is partitioned in a sufficient number of subsets, $f(x)$ will behave linearly, or almost linearly in each one of these subsets. The approximated, or predicted value of $f(x)$ in each subset is then given by a linear function or model:

$$\hat{y}_i = \beta_i (x - c_i) + \beta_{0i}$$

(3.2)

where $\beta_i$ is a vector $[\beta_1 \ldots \beta_n]$ , $c_i = [c_1 \ldots c_n]^T$ is the central point of a given section and $\beta_{0i}$ is a scalar corresponding to the linear model value at the central point (see figure 3.1).

As apparent in the example at figure 3.1, in order to obtain a global predictor $\hat{y} = f(x)$ valid for any domain query point $x$, it is necessary to locate to which domain subset $x$ belongs to. The associated local model is then used to make the prediction.

To ascertain to which subset a query point $x$ belongs to, LWL methods use functions denominated kernels that define each subset. Each kernel is centered around $c_i$ and is associated to a positive
definite distance matrix $D_i$ that defines its shape. The matrix $D_i$ defines the scope of the subset (see figure 3.1 for a graphical interpretation). The kernel functions used in the studied LWL methods can be of two types:

**Gaussian kernel**

$$w_i = e^{\frac{1}{2}(x-c_i)D_i(x-c_i)^T} \tag{3.3}$$

**Biquadratic kernel**

$$w_i = \begin{cases} 
(1 - d^2)^2 & : |d| < 1 \\
0 & : \text{otherwise}
\end{cases} \tag{3.4}$$

where $d = \frac{1}{2}(x - c_i)D_i(x - c_i)^T$

As apparent in (3.3) and (3.4), in response to a query point $x$ a kernel $i$ gives out a weight $w_i$ based on the relative distance of $x$ to the kernel center $c_i$ in a given direction. This relative distance can also be interpreted as the position of $x$ in the subset $i$ defined by kernel $i$. The further away the point $x$ is from the center of the subset, the lower will be its weight $w_i$, having zero (Biquadratic kernel) or almost zero (Gaussian kernel) weight when $x$ surpasses the scope defined by the kernels $D_i$.

For further reference, a local model associated with its respective kernel will be from now on denominated as a Receptive Field (RF) and the weight obtained in a particular RF in response to a query point $x$ is known as the RF activation weight for that point.

Considering the predictions obtained from each RF $\hat{y}_i$, add their associated weights $w_i$, the global prediction is defined as ($m$ is the total number of RFs):

$$\hat{y} = \frac{\sum_{i=1}^{m} w_i \hat{y}_i}{\sum_{i=1}^{m} w_i} = \frac{\sum_{i=1}^{m} w_i (\beta_i(x - c_i) + \beta_0i)}{\sum_{i=1}^{m} w_i} = \frac{\sum_{i=1}^{m} w_i^{\text{norm}} (\beta_i(x - c_i) + \beta_0)}{\sum_{i=1}^{m} w_i^{\text{norm}}} \tag{3.5}$$

where $w_i^{\text{norm}} = \frac{w_i}{\sum_{i=1}^{m} w_i}$ is the normalized weight with a range between 0 and 1. With $\sum_{i=1}^{m} w_i^{\text{norm}} = 1$.

Observing the schematic representation of (3.5) in figure 3.2 it is easier to understand the estimation process:

- A query point $x$ is fed trough all RFs. In each RF an activation weight $w_i$ is obtained by kernel $i$ based on the relative position of $x$ to its center in a given direction, the weights are then normalized.

- At the same time each local model generates a prediction based on $x$. The local predictions are then multiplied by their associated normalized weights $w_i^{\text{norm}}$ and summed obtaining the global prediction.
The normalized weights can be interpreted, from a prediction perspective, as a measure of relevance of a particular local prediction. The models defined in RFs to which \( x \) belongs the most will have greater contribution to the global prediction.

To clarify the idea the example in figure 3.1 with three RFs is used. Two particular situations are considered:

- In a first situation assume query point \( x \) is near \( c_1 \). Looking at the kernels (represented in blue) only RF\(_1\) will be activated. The distance of \( x \) to RF\(_2\) and RF\(_3\) goes beyond the border of each of their kernels. The normalized weight distribution for this case will then be \( w_{n1} \simeq 1 \), \( w_{n2} \simeq 0 \), \( w_{n3} \simeq 0 \). This means that the global prediction for query point \( x \) in this situation will only come from the local prediction at RF\(_1\).

- For an RF transition phase, for instance, query point \( x \) is in an area between \( c_1 \) and \( c_2 \). There is no single dominant RF, \( x \) is still beyond the border of RF\(_3\), but is between RF\(_1\) and RF\(_2\). The normalized weight distribution will then be \( w_{n1} \simeq 0.5 \), \( w_{n2} \simeq 0.5 \), \( w_{n3} \simeq 0 \). This means \( x \) is equally relevant to RF\(_1\) and RF\(_2\) and the global prediction will have to take into account the local predictions coming from those two models in an equally fashion. This is equivalent to say that the global prediction will be an almost arithmetic average of the two predictions.

Locally weighted learning algorithms obtain the parameters of each locally linear model through training, and in a completely independent form. They are also able to adapt the distance metric \( D_i \) of each kernel function to improve local prediction performance. This will be approached in some detail in the following section.
3. Locally weighted learning methods

3.2 LWL internal functioning

Although the general concept described in the previous section is common to RFWR and LWPR, their internal functioning is substantially different. The aim of this section is to describe the inner working in each of them: How the linear models parameters are estimated; how the distance metrics are learned for each RF and the criterium for adding or removing RFs from the global model.

One should note this is intended to be a summarized description of the algorithms focusing on the most important points. A thorough step by step description is available in [1][2][3].

3.2.1 Linear model parameters learning

In both cases the parameters \( \beta_i \) for each local model are obtained through regression. In RFWR the regression is done in the input-output space using a standard least squares approach. On the other hand LWPR uses the Partial Least Squares (PLS) implementation, where univariate regression is performed along \( r \) orthogonal projections of the input data.

For further reference, since the regression process is done the same way for all RFs, the following sections will only describe the regression for a particular RF. The subscript \( i \) used so far to define the \( i^{th} \) RF and its associated parameters will be dropped to simplify notation.

3.2.1.1 RFWR

In order to ensure online learning capability in RFWR, parameters \( \beta \) for each local model are learned through a Recursive Least Squares Algorithm with an exponential forgetting factor \( \lambda \). When an RF receives the \( k^{th} \) training pattern \( p \) at time \( n \), in the form \( (y^k, x^k) \) and with an associated weight \( w^k \). The means \( (\mu_y, \mu_x) \) for all patterns seen so far by the RF are updated to ensure mean zero variables \( \tilde{x}, \tilde{y} \).

\[
\mu_x^n = \frac{\lambda \left( \sum_{p=1}^{k-1} w^p \right) \mu_x^{n-1} + w^k x^k}{\lambda \left( \sum_{p=1}^{k-1} w^p \right) + w^k} \tag{3.6}
\]

\[
\mu_y^n = \frac{\lambda \left( \sum_{p=1}^{k-1} w^p \right) \mu_y^{n-1} + w^k y^k}{\lambda \left( \sum_{p=1}^{k-1} w^p \right) + w^k} \tag{3.7}
\]

\[
\tilde{x} = x^k - \mu_x^n \tag{3.8}
\]

\[
\tilde{y} = y^k - \mu_y^n \tag{3.9}
\]

the inverted covariance matrix \( P \) is then updated

\[
P^n = \frac{1}{\lambda} \left( P^{n-1} - \frac{P^{n-1} \tilde{x} \tilde{x}^T P^{n-1}}{w^T + \tilde{x}^T P^{n-1} \tilde{x}} \right) \tag{3.10}
\]
leading to the parameter vector update

\[ \beta^n = \beta^{n-1} + w P^n \tilde{e}_{cv}^T; \]  
(3.11)

with \( e_{cv} = \tilde{y} - \beta^{n-1} \tilde{x}^T \)

Considering \( \lambda = 1 \) the implementation previously described is the online recursive equivalent of a batch weighted Least Squares regression \[22\] with \( p = k \) training patterns.

\[ \beta = (\tilde{X}^T W \tilde{X})^{-1} \tilde{X}^T W Y = PWY \]  
(3.12)

with \( \tilde{X} = [\tilde{x}^1 \ldots \tilde{x}^k]^T, \tilde{Y} = [\tilde{y}^1 \ldots \tilde{y}^k]^T \) and \( W = diag(w^1 \ldots w^k) \).

Looking at equation (3.12) it is clearer how the independent regression is achieved for each RF. From a group of mean zero training patterns \((\tilde{X}, \tilde{Y})\) for a specific RF only the ones more relevant to it (higher \( w^p \)) will contribute to the regression.

A word of notice should be given concerning the prediction model described in (3.2). Considering that the model had seen training patterns until time \( n \) a prediction in response to a query point \( x_q \) at that time is in fact given by:

\[ \hat{y} = \beta(x_q - \mu_x^n) + \mu_y^n \leftrightarrow \hat{y} = \beta(\tilde{x}_q) + \mu_y^n \]  
(3.13)

although for general explaining purposes (3.2) is an acceptable approximation. In fact looking at (3.7) and (3.8) one can see the weight attributed to a particular pattern influences the means \((\mu_x, \mu_y)\) update, considering the weights are highest near the RF’s center \( c \) then, \( \mu_x \approx c, \mu_y \approx \beta_0 \).

To guarantee the regression model in each RF is trustworthy, that is, reliable parameter estimates \( \beta \) are obtained, two numerical safety measures are calculated. Only trustworthy RFs will contribute to the global prediction in (3.5). One measure is naturally the number of training data seen by the RF. Only RFs that have seen a number of data superior to a given threshold are considered trustworthy.

The other measure relies on the determination of the Variance Inflation Factor (VIF) to evaluate the degree of multicollinearity in the regression. Appendix A provides the definition for multicollinearity and its relation with the VIF.

Considering that matrix \( P \) is the inverted covariance matrix and using equation (A.5) the update rule for the VIF matrix in a particular RF at time \( n \) is given by:

\[ VIF^n = diag(P^n)(\sigma_x^2)^n I \]  
(3.14)

where \( I \) is the identity matrix and \( \sigma_x^2, \sigma_y^2 \) is the current RF variance updated using the result from the \( \mu_x \) update rule (3.7):

\[ (\sigma_x^2)^n = \frac{\lambda \left( \sum_{p=1}^{k-1} w^p \right) (\sigma_x^2)^{k-1} + w_k (x - \mu_x^n)^2}{\lambda \left( \sum_{p=1}^{k-1} w^p \right) + w_k} \]  
(3.15)
3. Locally weighted learning methods

Usually obtaining a VIF for any parameter higher than 10 means high multicollinearity exists [23]. Thus only RFs having all the diagonal elements in the VIF matrix smaller than 10 are considered trustworthy.

3.2.1.B LWPR

As referenced in this section’s introduction LWPR uses PLS to obtain parameter $\beta_{PLS}$ for each projection direction. As explained in appendix B PLS will perform univariate regression in orthogonal projection directions $u$ that truly explain the relation between the input data $X$ and output data $Y$ seen by each RF thus eliminating the multicolinearity problem.

To explain the parameter update process consider again a situation where a particular RF receives the $k^{th}$ training pattern $p$ at time $n$, in the form $(y^k, x^k)$ and with an associated weight $w^k$. First the mean zero variables $\bar{x}$ and $\bar{y}$ are calculated by (3.9) and (3.9) respectively. They are then used as the initialization for the iterative update process described in figure 3.3.

*For $j = 1$ to $r$ projections*

\[
\begin{align*}
(x_{res})_1 &= \bar{x} \\
(y_{res})_1 &= \bar{y}
\end{align*}
\]

\[
\begin{align*}
\lambda_{u_j} &= \lambda u_j^{n-1} + w^k (x_{res})_j (y_{res})_j \\
\phi_j &= (x_{res})_j u_j^n \\
SS_j^n &= \lambda SS_j^{n-1} + w^k s^2 \\
SR_j^n &= \lambda SR_j^{n-1} + w^k (y_{res})_j \\
SZ_j^n &= \lambda SZ_j^{n-1} + w^k (x_{res})_j s \\
\beta_j^n &= SR_j^n / SS_j^n \\
p_j^n &= SZ_j^n / SS_j^n \\
(x_{res})_{j+1} &= (x_{res})_j - \phi_j p_j^n \\
(y_{res})_{j+1} &= (y_{res})_j - \phi_j\beta_j^n \\
MSE_{j+1} &= \lambda MSE_j^{n-1} + w y_{res}
\end{align*}
\]

end

Figure 3.3: RF update stage for LWPR

Notice that the update process constitutes the weighted incremental equivalent of the batch implementation for PLS described in appendix B. The zero mean new data is used to update the first projection direction $u$ (3.18) and the new data is projected into that direction (3.19).

The projected data point $\phi$ is then regressed against the output space (3.23) and input space (3.24). To make the regression possible it is necessary to use the cumulative (or memory) variables defined in (3.20), (3.19) and (3.20) to "store" the data seen by the RF until time index $n$.

Finally the new residuals are calculated in (3.24) and (3.25). The process is then repeated for the
3.2 LWL internal functioning

The subsequent number of directions \( r \) currently in use by the RF. Similarly to RFWR, the activation weights are included to ensure independent learning between RFs.

The Mean Squared Error (MSE) in (3.26) is used to evaluate if an additional direction \( u_j \) needs to be added to the regression. This is accomplished by comparing the MSE of the last two projections being used at a given time \( n \). If the MSE from the last two projections does not decrease by more than a certain percentage

\[
\frac{MSE_r}{MSE_{r-1}} \times 100 > T(\%)
\]  

(3.27)

then LWPR stops adding directions to the regression.

The prediction of an RF in response to a query point \( x_q \) that as seen training patterns until time \( n \) is obtained by projecting \( x_q - \mu_x \) into each of the projection directions \( u \) for that RF, resulting in a vector of \( r \) projected patterns \( \Phi_q = [\phi_1 \cdots \phi_r]^T \). This is done iteratively as depicted in figure 3.4. Using the parameter vector \( \beta \) obtained from the regression in each projection the prediction

\[
(x_{res})_1 = x_q - \mu_x \tag{3.28}
\]

for \( j = 1 \) to \( r \) projections

\[
\phi_j = (x_{res})_j u_j^n \tag{3.29}
\]

\[
(x_{res})_{j+1} = (x_{res})_j + \phi_j n \tag{3.30}
\]

end

Figure 3.4: Projection of query point \( x_q \) to \( r \) projection directions

is given by:

\[
\hat{y} = \beta_{PLS}^n \Phi_q + \mu_y^n \tag{3.31}
\]

Another way of obtaining the prediction that will be important for control purposes is to convert the parameters \( \beta_{PLS} \) to input space parameters \( \beta \) in order to obtain the prediction model in a structure similar to (3.13). Considering that \( \beta = \frac{dy}{dx} \) in (3.13) and the derivative in relation to \( \tilde{x} \) for (3.31), the relation between \( \beta \) and \( \beta_{PLS} \) is obtained:

\[
\frac{dy}{dx} = \beta_{PLS} \frac{d\Phi}{dx} \leftrightarrow \beta = \beta_{PLS} \frac{d\Phi}{dx} \tag{3.32}
\]

where the derivative of each projection in relation to the input space \( \frac{d\Phi}{dx} \) at time \( n \) can also be obtained in an iterative form as depicted in figure 3.5.

In LWPR, the only numerical safety measure necessary to guarantee that the RF is trustworthy, is to ensure it has seen a sufficient number of training data. VIF determination is no longer required because, as referenced before, multicolinearity is no longer a problem.
3. Locally weighted learning methods

\[
\left( \frac{dx}{d\tilde{x}} \right)_j = I
\]

for \( j = 1 \) to \( r \) projections

\[
\left( \frac{d\phi}{d\tilde{x}} \right)_j = \left( \frac{dx}{d\tilde{x}} \right)_j w_j
\]

\[
\left( \frac{dx}{d\tilde{x}} \right)_{j+1} = \left( \frac{dx}{d\tilde{x}} \right)_j - \left( \frac{ds}{d\tilde{x}} \right)_j (\mu_j^n)^T
\]

end

Figure 3.5: Derivative of projections in relation to mean zero input space \( I \) is the identity matrix

3.2.2 Adding and pruning RFs

The adding and pruning strategies for both LWPR and RFWR are similar. A new RF is added each time a given training pattern \((y, x)\) does not activate any existing RF by more than a predefined threshold \(w_{\text{gen}}\), when creating an RF there are two possible outcomes:

Pattern \((y, x)\) still activates some RF by more than \(0.1w_{\text{gen}}\) In this case the new RF is still considered to be close to an existing RF the new RF is created with distance matrix \(D\) equal to the previous RF.

No RF is activated by more than \(0.1w_{\text{gen}}\) - In this case the new RF is initialized with the predefined default value for \(D\).

Pruning of an RF takes place when a training pattern activates two RFs by more than a predefined threshold \(w_{\text{prune}}\). In that case the RF with larger determinant for \(D\) is pruned. This is done to avoid having superimposed (or almost superimposed) RFs. This situation will not necessarily degrade prediction performance, but is computationally inefficient.

3.2.3 Distance metric learning

As seen in the previous sections, LWL performance is highly dependent on the shape and size of the kernel. A relatively small initial \(D\) for each RF leads to too few RFs which in turn implies that the function \(y = f(x)\) will not behave linearly or almost linearly in each of them, leading to poor prediction performance. On the other hand, a too large value for \(D\) ensures local linearity, but may lead to too few training points per RF which implies poor performance in the estimation of the parameters \(\beta\), degrading prediction performance once again and also increasing the computation complexity.

Both LWL algorithms should adapt to the local curvature of \(f(x)\). High curvature zones should have smaller RFs and in greater number. Low curvature zones don’t need a great number of RFs, so a smaller distance metric should be allowed.
3.2 LWL internal functioning

To try and reach an optimum solution for the shape and size of each RF, a leave-one-out cross validation cost functional in order to the decomposed distance matrix \( M(D = M^T M) \) is minimized for each of them.

\[
J(W) = \frac{1}{W} \sum_{k=1}^{P} w_k \| y_k - \hat{y}_{k,-k} \|^2 + \gamma \sum_{i,j=1}^{n} D_{i,m}^2
\]  

(3.34)

where \( W = \sum_{i=1}^{n} w_i \) and \( p \) is the number of training points seen by the algorithm.

In the cost functional (3.34), \( \hat{y}_{k,-k} \) represents the prediction of the \( k \)-th data point with the \( k \)-th data point excluded from the training set. A penalty term \( \gamma \sum_{i,j=1}^{n} D^2_{i,j} \) is added in the cost functional where \( D_{i,m} \) is an element in the distance matrix. The finality of the penalty term is to prevent RFs from becoming too small which could lead to overfitting problems (an unnecessarily large number of RFs). As the penalty factor \( \gamma \) is increased, the distance matrix will become smaller which in turn implies larger RFs. The cost functional (3.34) can be rewritten to be used with RFWR or LWPR. In RFWR is given by:

\[
J = \frac{1}{W} \sum_{k=1}^{P} \frac{w_{i,k}}{1 - w_{i,k}^2} \| y_k - \hat{y}_{i,k} \|^2 + \gamma \sum_{i,j=1}^{n} D_{i,j}^2
\]  

(3.35)

where \( P \) is the inverted covariance matrix, as in the previous sections.

For the LWPR case it can be rewritten as

\[
J = \frac{1}{W} \sum_{k=1}^{P} \frac{w_{i,k} (y_{res})_{j,k}^2}{1 - w_{i,k}^2} \frac{\sigma_{i,k}^2}{\phi_k^2 \phi_k}
\]  

(3.36)

where \( r \) is the total number of projections being used for a regression in a particular RF, \( (y_{res})_{j,k} \) is the residual of the output data for a particular projection of training point \( k \) and \( \phi_k \) is projection point. These reformulations allow for both problems to be solved by a gradient descent solution:

\[
M_{i}^{n+1} = M_{i}^{n} - \alpha \frac{\partial J_{i}}{\partial M_{i}}
\]  

(3.37)

where \( \alpha \) is the learning rate and \( \frac{\partial J_{i}}{\partial M_{i}} \) is obtained incrementally.

3.2.4 Algorithm Pseudo-Code

For a better insight in how the algorithm works, pseudo-code for the update stage and prediction stage is presented comparing LWPR and RFWR.

3.2.4.4 Update Stage

- training pattern \((x,y)\) is received
  - FOR Number of RFs
    - compute activation weight for RF
      - IF RFWR
        - update regression parameter as described in section (3.2.1.A)
      END
      - IF LWPR
        - update regression parameters as described section (3.2.1.B)
    - END
  - check if new projection needs to be added (3.27)
  - END
- check for adding new RF (section 3.2.2)
- check for pruning RF (section 3.2.2)
3. Locally weighted learning methods

3.2.4.B Prediction Stage

<table>
<thead>
<tr>
<th>FOR Number of RFs</th>
</tr>
</thead>
<tbody>
<tr>
<td>-query point ( x ) is received</td>
</tr>
<tr>
<td>-sum of predictions = 0</td>
</tr>
<tr>
<td>-sum of weights = 0</td>
</tr>
</tbody>
</table>

FOR Number of RFs
- compute activation weights for RF
  IF RF activation is high enough and RF is trustworthy (see sections 3.2.1.A 3.2.1.B)
    IF RF
      Compute prediction (eq. 3.13)
    END
  IF LWPR
    Compute prediction (eq. 3.29)
  END
  - multiply prediction by activation weight and add to sum of predictions
  - add activation weight to sum of weights
END

- divide sum of predictions by sum of weights and obtain final prediction
IF sum of predictions = 0
  Xprediction failure only happens when
  - there are no RFs in proximity or they are all untrustworthy.
  - final prediction = 0
END
Contents

4.1 Model estimation using LWL methods ........................................ 27
4.2 Control Setup .......................................................................... 28
This section explains how the LWL methods are used to estimate a non-linear dynamic system, followed by a description of the control setup.

4.1 Model estimation using LWL methods

A causal discrete non-linear dynamic system can be approximated as a linear difference equation with time variable parameters where \( n \) is the time index:

\[
y(n) = -a_1(n)y(n-1) - a_2(n)y(n-2) \cdots - a_{N_a}(n)y(n-N_a) \\
+ b_1(n)u(n-1-d) + b_2(n)u(n-2-d) + b_3(n)u(n-3-d) \\
\cdots + b_{N_b}(n)u(n-N_b-1-d)
\]  \hspace{1cm} (4.1)

with a delay \( d \geq 0, N_a \geq 0, N_b \geq 0 \) and \( N_b = N_a - 1 - d \). This difference equation can be rewritten in a compact form:

\[
y(n) = a(n)s_y + b(n)s_u \Leftrightarrow y(n) = f(s_y, s_u)
\]  \hspace{1cm} (4.2)

where \( a(n) = [a_1(n) \cdots a_{N_a}(n)] \), \( b(n) = [b_1(n) \cdots b_{N_b}(n)] \) are the time variable parameter vectors and \( s_y = [y(n-1) \cdots y(n-N_a)]^T \), \( s_u = [u(n-1-d) \cdots u(n-N_b-1-d)]^T \) are the system state vectors associated with the output and the input respectively. Following what was stated in section 3.1 about both LWL methods, if training patterns \( [y(n), (s_y, s_u)] \) are provided, \( f(s_y, s_u) \) is approximated through \( m \) local models

\[
y^*(n)_i = \beta_{ai}(s_y - c_{ai}) + \beta_{bi}(s_u - c_{bi}) + \beta_{oi}
\]  \hspace{1cm} (4.3)

with a final prediction

\[
y^*(n) = \frac{\sum_{i=1}^{m} w_i (\beta_{ai}(s_y - c_{ai}) + \beta_{bi}(s_u - c_{bi}) + \beta_{oi})}{\sum_{i=1}^{m} w_i}
\]  \hspace{1cm} (4.4)

(4.3) can be interpreted as linear incremental around the point \( f(c_{ai}, c_{bi}) = \beta_{oi} \)

\[
\Delta y^*(n)_i = \beta_{ai}\Delta s_y + \beta_{bi}\Delta s_u
\]  \hspace{1cm} (4.5)

with

\[
\Delta y^*_i = y(n)_i - \beta_{oi} \\
\Delta y_{-n_i} = s_y - c_{ai} \\
\Delta u_{-n_i} = s_u - c_{bi}
\]

substituting into (4.4) yields the global increment

\[
\Delta y = \frac{\sum_{i=1}^{m} w_i (\beta_{ai}\Delta s_y + \beta_{bi}\Delta s_u)}{\sum_{i=1}^{m} w_i}
\]  \hspace{1cm} (4.6)
4. Control Strategy

These results show that LWL methods create, from a control standpoint, a set of local linear rules of behavior (or dynamics) for the nonlinear behavior of the global system, each one specific to a certain working area defined around \((c_y, c_u)\). The normalized weight \(w_i^{\text{norm}} = \frac{w_i}{\sum_{i=1}^{m} w_i}\) defines the contribution of a given local dynamic to the global increment. In other words \(w_i^{\text{norm}}\) measures the degree of resemblance between the global non-linear behavior and a local behavior \(i\), when the global system is in a given working area defined by its state \((s_y, s_u)\).

\[
\begin{align*}
u \cdot u_{\bar{v}1} & \rightarrow \text{Model 1} \quad y_1 \cdot \beta_{01} \\
u \cdot u_{\bar{v}2} & \rightarrow \text{Model 2} \quad y_2 \cdot \beta_{02} \\
\cdots \cdots & \\
u \cdot u_{\bar{v}j} & \rightarrow \text{Model } i \quad y_i \cdot \beta_{0i} \\
\end{align*}
\]

\(w_i^{\text{norm}}\)

Figure 4.1: Schematic of the global model obtained with LWL methods

To better illustrate the concept, figure 4.1 shows a graphical representation of what was just said. At a given time index \(n\), \(i\) RFs exist, each RF being a model linearized around a point \((u_{\bar{v}1}, \beta_{01})\) and the final output for the global model is obtained from a linear combination of all local model’s outputs. Notice the role of the normalized weights to the final output: they control which local models should be considered. Remembering what was said in section 3.1 the kernel in each RF determines its weight for a current state \((s_y, s_u)\), so only the models “close” to the current state will be activated.

4.2 Control Setup

4.2.1 Controller Design

Assuming each local model (4.5) parameters are correctly estimated and also the size and number of RFs is correctly determined (the model structure is correct), then an augmented version of the model depicted in figure 4.1 can be used for control purposes. If a controller is designed for each local augmented model such that each closed loop system will track the same reference, that is:

\[
\lim_{n \to \infty} \text{ref}(n) - y_i(n) = 0 \Rightarrow \text{ref}(\infty) = y_i(\infty)
\]  

(4.7)
4.2 Control Setup

Figure 4.2: Augmented model for control

substituting in equation (4.4) it is clear the global system will also track the same reference:

\[ y = \frac{\sum_{i=1}^{n} w_i \cdot ref(\infty)}{\sum_{i=1}^{n} w_i} = ref(\infty) \]  

(4.8)

Notice the results shown are no proof of guaranteed convergence to the reference but instead, they were made under the rather big assumption that the global model structure and local parameters estimates were correct. As seen in chapter three making this assumption correct is not a trivial matter, there is no way to guarantee this conditions in any situation. Result (4.8) only shows the convergence of the control strategy for an ideal situation.

The speed of convergence to the reference will naturally depend on the current working area of the algorithm (each local model will have different characteristics) and also on the controller design method.

The global controller will be a block of multiple controllers (figure 4.3), one for each linear augmented model (figure 4.2). The local command increments from each controller \( \Delta u_i \) are fused into the global increment \( \Delta u \) resorting to the normalized activation weights obtained from the LWL method at a given time index \( n \). In other words, if the global system state at a time \( n \) is provided,
the most influential controllers to $\Delta u$ at that time are those that control the mostly activated local models for that state.

4.2.1.A State Space model derivation

As described in the previous section the main idea for control is trying to find a sequence of linear models with a fixed structure:

$$y(t) = \beta_{a1} y(n - 1) + \cdots + \beta_{a_N} y(n - N_a) + \beta_{b1} u(n - 1 - d)$$

$$+ \cdots + \beta_{b_N} u(n - N_b - 1 - d)$$

$$= \beta_a s_y + \beta_b s_u$$

where $d$ is the delay.

The training input space for both LWL methods will be the previous system’s states in the form $s = [s_y \ s_u]$ and one model with this structure exists for each RF. In RFWR the regression is performed directly in the input space (3.2.1.A) so the parameters $\beta = [\beta_a \ \beta_b]$ for each model are obtained directly. On LWPR the parameters $\beta$ are obtained using (3.32).

To synthesize the controller it is necessary to convert the linear model to a State-Space formulation:

$$x(n+1) = \Phi x(n) + \Gamma u(n)$$

$$y(n) = C x(n)$$

This is accomplished by making

$$\Phi = \begin{bmatrix} 0_{1 \times N_a-1} & \ldots & I_{N_a-1} \\ \beta_{a_{N_a}} & \ldots & \beta_{a_1} \end{bmatrix} \quad \Gamma = \begin{bmatrix} 0_{1 \times N_b-1} \\ 1 \end{bmatrix} \quad C = \begin{bmatrix} 0_{m \times 1} & \beta_{b_{N_b}} & \cdots & \beta_{b_1} \end{bmatrix}$$

where $0_{n \times m}$ is a $n$ by $m$ matrix of zeros and $I_n$ is a size $n$ identity matrix.

4.2.1.B Local controller synthesis

![Local control scheme](image)

Figure 4.4: Local control scheme

For each local model with dynamics described by (4.11) a controller like the one specified in figure 4.4 is designed. An Euler integrator $\frac{T_s}{z - 1}$ ($T_s$ is the discrete sampling time) is added to the local model, leading to the augmented dynamics:
4.2 Control Setup

\[ \Phi_a = \begin{bmatrix} \Phi & \Gamma T_s \\ 0 & 1 \end{bmatrix}, \quad \Gamma_a = \begin{bmatrix} 0 \\ T_s \end{bmatrix}, \quad C_a = [C \ 0] \]

The control law gains \( K \) are determined for the augmented system solving a discrete infinite horizon LQ optimization problem \([24][25]\)

\[ J(u) = \sum_{n=1}^{\infty} [x(n)^T Q x(n) + u(n)^T R u(n)] \tag{4.12} \]

where \( Q \) is a square matrix penalizing large \( x \) and \( R \) is a scalar penalizing large commands \( u \). In this case \( Q = C^T C \) and \( R \) is a predefined value set in the initialization of the control algorithm.

This problem has a closed form solution \( S \) obtained from the discrete algebraic Riccati equation

\[ \Phi_a^T S \Phi_a - S - (\Phi_a^T S \Gamma_a) (\Gamma_a^T S \Gamma_a + R)^{-1} (\Gamma_a^T S \Phi_a) + Q = 0 \tag{4.13} \]

The control law gains \( K \) that stabilize the closed loop system are then computed from \( S \) by:

\[ K = (\Gamma_a^T S \Gamma_a + R)^{-1} (\Gamma_a^T S \Phi_a) \tag{4.14} \]

To obtain the estimate \( \hat{x} \) a current observer \([24]\) incorporated with the system’s tracking error \( e(n) = \text{ref}(n) - y(n) \) is used.

The main difference between a regular prediction observer with observer gains \( L_p \)

\[ \hat{x}(n|n-1) = [\Phi_a - L_p C_a] \hat{x}(n-1|n-2) + \Gamma_a u(n-1) - L_p e(n-1) \tag{4.15} \]

and a current observer is the fact that the prediction observer obtains the current estimate \( \hat{x}(n|n-1) \) with the error measurements available until the previous time instant. In opposition a current observer quickly updates the estimate coming from instant \( n-1 \) with the tracking error information available at time \( n \) obtaining the current estimate \( \hat{x}(n|n) \):

\[ \hat{x}(n|n) = \hat{x}(n|n-1) + L_c [-e(n) - C_a \hat{x}(n|n-1)] \Leftrightarrow \hat{x}(n|n) = [I - L_c] \hat{x}(n|n-1) - L_c e(n) \tag{4.16} \]

where \( I \) is the identity \( L_c \) are the current observer gains and \( x(n|n-1) \) is the predicted estimate based on a model prediction from the previous time estimate

\[ \hat{x}(n|n-1) = \Phi_a \hat{x}(n-1|n-1) + \Gamma_a u(n-1) \tag{4.17} \]

The observer gains \( L_c \) are designed to make the observer a factor faster than the closed-loop system. This is done by placing the observer poles \( Op \) at a location \( f \) times smaller than the closed loop poles \( CLp \)

\[ Op = CLp^f \tag{4.18} \]

with \( f > 1 \). The factor \( f \) is set in the initialization of the algorithm.

Based on \([4.17]\) the current controller can then be defined by

\[ \Delta u = -K \hat{x}(n|n) \tag{4.19} \]
4. Control Strategy

An implementable controller is then derived substituting (4.16) into (4.19)

\[ \dot{x}(n+1|n) = [\Phi_a - \Gamma_a K][I - L_a C_a]\dot{x}(n|n-1) - [\Phi_a - \Gamma_a K]L_a e(n) \]
\[ \Delta u = -K[I - L_a C_a]\dot{x}(n|n-1) + KL_a e(n) \] (4.20)

4.2.2 Control Algorithm

A short description on how the control strategy is implemented is stated in this section followed by pseudo-code to give a more general idea.

4.2.2.A General Description

![Overall view of the control strategy](image)

Figure 4.5: Overall view of the control strategy. The controller block synchronized with the learning algorithm and the common integrator with anti wind-up

The algorithm is designed to work with one of the two methods described in this work [RFWR] or [LWPR] and can work with a pre-trained initial mean model or without any prior information. It comprises of two main stages:

- In the first stage the system is being controlled manually and training patterns \([y, (s_y, s_u)]\) are being supplied to the learning algorithm at each time index \(n\). In this stage the algorithm either builds an initial model for the case where no a priori information was known, or uses the new data to adjust the pre-trained mean model.

- In the second stage control is turned on based on the initial model obtained in the first stage. Training patterns continue to be supplied to the learning algorithm and the control algorithm checks for changes in the global model. The controller block is then updated accordingly.

4.2.2.B Updating the Controller Block

The updates for the controller block are done periodically with a period \(T\) defined in the initialization. During the update process three distinct situations may occur:
4.2 Control Setup

1. A new RF is added, in this case a controller is synthesized for the new model and added to the controller block.

2. The model parameters for a pre-existing RF change, in this case a controller for the new model is synthesized and the controller corresponding to the RF in question is substituted. To avoid constant changes in the controller block caused by small changes in the model Vinicombe (see appendix C) gap metric is used to compare the pre-existing model with the new one. The controller only changes if the gap metric between the two models is superior to a given threshold.

3. An RF is pruned, in this case the controller associated to the RF in question is eliminated from the controller block.

4.2.2.C Integrator anti-windup

The actuator used to administer propofol to the patient is nonlinear. Evidently a saturation at zero exist, because a negative infusion rate would correspond to a physical impossible situation of removing drug from the patient’s body.

Using a nonlinear actuator with integral control creates a problem commonly known as the integrator windup. If during control saturation is reached the closed loop is temporarily interrupted. A controller “sees” variations in the tracking error but is unable to see the saturated output \( v \) and consequently the integral term continues to rise. When eventually the tracking error is reduced, the controller may take more time to react because the integral value is too high. A possible solution to this problem is to use a back-calculation strategy represented in figure 4.6. The system described in figure 4.6 is defined by:

\[
u = \frac{1}{z - 1} (\Delta u + \gamma e_s)
\]  

(4.21)
4. Control Strategy

considering the integrator input must be zero at steady state.

\[ T_s \Delta u + \gamma e_s = 0 \Leftrightarrow e_s = -\frac{1}{\gamma} \quad (4.22) \]

Looking at figure 4.6 and the result in (4.22) it is clear how back-calculation works. When the integrator is working on the linear zone of the actuator \( e_s \) is zero and the integrator works normally. If saturation is reached \( e_s \) is no longer zero and the input of the integrator will be driven to zero with a rate defined by \( \gamma \). Result (4.22) determines that \( \gamma \) must be chosen to be higher than one. \( \gamma = 1 \) represents the limit case when the anti-windup strategy tries to drive the integrator input to zero instantly.

4.2.2.2 Pseudo Code

```%
\%Initializations
-define command penalty R
-define factor f
-time to begin control (BCT) is defined
-interval to update the controller is defined

\%Algorithm
FOR n=1 to end of control
  -get current output \( y(n) \) and previous state \((y, y_u)\)
  -update LNPR or RFWR with the pattern \((y(n), (y, y_u))\)
  -predict current output using previous state \((y, y_u)\) and get:
    .weight and model information from all existing RFs
  IF \( n>BCT \) \%control begins
    IF \( n-BCT \) or block update time is verified
      -update controller block with model information and weights
      END
    -feed tracking error to controller block
    -get u from controller block
  ELSE
    -get u from manual control
  END
  -feed command u to the system
END
%```
5 Simulations

Contents

5.1 Testing Conditions ................................................. 37
5.2 RFWR and LWPR for prediction and control comparative
  and parameter variation study ................................ 38
5.3 Control algorithm analysis ..................................... 42
This chapter is divided into two main parts:

- In the first part RFWR is compared to LWPR in terms of prediction capability using input and output data from several patient models. The objective for this section is also to create mean prediction models. Using those models mean controller blocks are then synthesized and their control performance is evaluated.

- In the second part the control algorithm is tested using a mean controller block obtained in the first part. The mean controller block is used in the initialization of the algorithm. New data is then integrated online in the learning algorithm to readjust the mean controller block.

5.1 Testing Conditions

To make the control tests a sampled version of the DoA model described in chapter 2 is used as control plant. The model is sampled using zero order hold with a sampling time, $T_s$, of 5 seconds. Most model parameters will be fixed and are taken from Marsh. [6]. Table 5.1 specifies the predefined parameter values for the model. The only parameter varying from model to model in this simulation is the patient weight which, as seen in (2.2), will determine the volume for the PK compartment one. Also, weight, BIS and propofol data from 37 patients is available of which 29 will be used to generate a mean model, the tests will be done with the remaining 8.

All simulations are done in MATLAB and try to resemble as much as possible a real situation. During a predefined initial time a quantity of propofol is injected "manually" to drive the BIS value close to a recommended level (usually the recommended level is 40). In medical practice this stage is commonly known as the induction phase. After the simulated induction phase control is turned on and tries to track a predefined filtered reference. To deal with the actuator saturation at zero the integrator windup (4.22) parameter $\gamma$ is set to one (instantaneous).

In all simulations the global non-linear system will be approximated by 3rd order linear models:

$$y_i(k) = a_1 y_i(n-1) + a_2 y_i(n-2) + a_3 y_i(n-3) + b_1 u_i(n-1)$$  \hspace{1cm} (5.1)

where $n$ is the time index. Considering this fixed model structure and what was referenced in chapter 4 the patterns will be in the form $(BIS(n), s)$, where $s$ is the previous global system's

<table>
<thead>
<tr>
<th>PKPD fixed parameters</th>
<th>PK</th>
<th>Effect compartment</th>
<th>Interaction Model</th>
</tr>
</thead>
<tbody>
<tr>
<td>$k_{10}$</td>
<td>$1.98E^{-4}$</td>
<td>$4.17E^{-4}$</td>
<td>$C_{\text{pk}}^{\text{ini}}$</td>
</tr>
<tr>
<td>$k_{12}$</td>
<td>$1.87E^{-3}$</td>
<td>$C_{\text{ke}}^{\text{ini}}$</td>
<td></td>
</tr>
<tr>
<td>$k_{13}$</td>
<td>$6.98E^{-4}$</td>
<td>$C_{\text{ke}}^{\text{ini}}$</td>
<td></td>
</tr>
<tr>
<td>$k_{21}$</td>
<td>$9.17E^{-5}$</td>
<td>$E_0$</td>
<td></td>
</tr>
<tr>
<td>$k_{31}$</td>
<td>$5.50E^{-7}$</td>
<td>$\gamma$</td>
<td></td>
</tr>
<tr>
<td>$V_c$</td>
<td>$2.28E^{-1}$</td>
<td>$\beta$</td>
<td></td>
</tr>
</tbody>
</table>

Table 5.1: PKPD model fixed parameters
5. Simulations

state defined by:

\[
\begin{bmatrix}
BIS(n - 1) & BIS(n - 2) & BIS(n - 3) & prop(n - 1)
\end{bmatrix}
\]

In the previous expression \( BIS \) and \( prop \) represent the \( BIS \) values between 0 and 100 and the \( propofol \) infusion rate in [ml/h].

The initial distance matrix for each RF is considered to be defined by two parameters: the distance associated with the output part of the training patterns (the \( BIS \) values) and a distance associated with the input part (the \( propofol \) values). In this case, since \( s \) is size 4 the initial distance matrix is defined by:

\[
D = \begin{bmatrix}
d0 & 0 & 0 & 0 \\
0 & d0 & 0 & 0 \\
0 & 0 & d0 & 0 \\
0 & 0 & 0 & dI \\
\end{bmatrix}
\]

5.2 RFWR and LWPR for prediction and control comparative and parameter variation study

The main objective for this section is to use previously obtained \( propofol \) and \( BIS \) data from 37 patient models to do a comparative analysis between LWPR and RFWR in prediction and control performance when large amounts of data are available. The objective is also to obtain an acceptable mean controller block to be used as the initial block for the control algorithm.

5.2.1 Prediction performance

![Training input and output data from 29 models](image)

Figure 5.1: Training input and output data from 29 models

To test the prediction performance of both algorithms a simple validation test is used. The data is split in a training set with 29 models and a test set with 8 models, the test is done in two stages:
5.2 RFWR and LWPR for prediction and control comparative and parameter variation study

- Both learning algorithms are fed with the same training patterns \((BIS(n), s)\) generated from each of the 29 models, represented in figure 5.1. The patterns are given sequentially in time. This is done to mimic the way the learning algorithm is fed during control.

- For each of the 8 test models, previous states \(s\) are obtained and fed to both algorithms to make predictions. The predictions obtained for a model are then compared to the real \(BIS\) value for the same model.

The criterium used to compare the real \(BIS\) with the predicted \(BIS\) is the \(MSE\). Considering prediction failures may occur because no RF is activated when a test pattern is given, which implies \(BIS_p = 0\), the \(MSE\) is only calculated with valid predictions, that is, time samples where \(BIS_p > 0\).

Defining \(BIS_{pv}\) as the valid predictions the \(MSE\) is given by:

\[
MSE = \frac{1}{N_{\text{pred}}} \sum_{k=1}^{N_{\text{pred}}} (BIS_{pv}(k) - BIS(k))
\]

where \(N_{\text{pred}}\) is the number of valid predictions.

To evaluate the extent of prediction failures for each test model the percentage of failures \(PredF(\%)\) is also calculated:

\[
PredF(\%) = \frac{N - N_{\text{pred}}}{N} \times 100
\]

where \(N\) is the total number of predictions.

The overall objective for the test consists in evaluating the prediction performance with the variation of three parameters: the distance matrix parameters \((d_i, d_o)\) and the forgetting factor \(\lambda\).

All tests are made with the RF distance metric adaptation described in (3.37) turned off.

For each parameter configuration of \((d_i, d_o, \lambda)\) the simple validation test described previously is performed. The average results for the 8 test models with each parameter configuration are presented in appendix D.1.

Observing table D.5 it is concluded that the influence of the \(d_i\) and \(d_o\) with the number of RFs is clear. Choosing larger distance matrices, will imply smaller RFs and thus a larger number of RFs is needed to cover the entire trained input-output space. The \(\lambda\) parameter has no influence on the RF size, it only controls the rate at each local model “forgets” old data. Thus the number of RF’s was the same for all the \(\lambda\) values used in the test.

The simulations from table D.1 to D.4 concerning the prediction performance show some interesting results. Considering only the \(MSE\) RFWR outperforms LWPR, this was expected since PLS does not use all the information present in a training set, only the more relevant (see appendix B).

Ignoring some almost irrelevant projection directions of the input-output space will mean larger local model errors and thus poorer prediction performance in relation to a method that used the entire input-output space.

However RFWR denotes a larger number of prediction failures (see figure 5.2). The situation is
particularly serious when $\lambda$ is decreased, where failure is almost complete. Comparing tables D.5 and D.6 the difference between the number of RF's allocated by RFWR and the number of trustworthy RF's is notorious. This happens because the VIF for almost all local models is higher than 10, which indicates high multicolinearity in the local regressions (see section 3.2.1.A). LWPR, due to its PLS implementation, when subjected to the same situation is able to make predictions overcoming the problem. In an overall perspective LWPR performance is better than RFWR where the tradeoff is prediction error by prediction reliability.

A relation between the distance matrix, LWPR prediction performance and $\lambda$ is also apparent on tables D.1 to D.4. The MSE seems to improve when a larger number of RF's is used at the cost of having an increase on prediction failures, this is to be expected also, when a large amount of RF's is used the input-output training set is partitioned in a large number of subsets, this implies less data is available for each RF and some might have insufficient data to perform regression, thus becoming untrustworthy. The MSE performance increase with a larger number of RF's is notorious when smaller $\lambda$'s are chosen.

5.2.2 Mean Controller Block Performance

The control performance test will use mean global models built by LWPR in the training stage of the previous section to synthesize the mean controller blocks as explained in sections 4.2.1.A and 4.2.1.B. The control simulations use the DoA model for each of the 8 test patients (with parameters defined in section 5.1) as the control plant.

Only LWPR is tested since, as it was seen in the previous section, outperforms RFWR in prediction performance and the assumption made in in section 4.1 for building a successful control strategy was that the global model predicts well the behavior of the real system.

So the parameter configurations that gave best prediction performance were chosen to do the test.
5.2 RFWR and LWPR for prediction and control comparative and parameter variation study

For each chosen parameter configuration a controller block is synthesized and is then used to drive each of the 8 simulation models to a predefined reference. The reference is placed in the area (BIS values between 35 and 45) where more training data exists (figure 5.1) and thus where the built LWPR model was mostly trained.

The controller blocks are synthesized with a fixed set of parameters $R = 10$, $f = 3$ and each simulation lasts 10000 seconds (2 hours).

The induction phase is simulated by setting a fixed propofol infusion rate of 200[ml/h] lasting 300 seconds. This initial action will drive the BIS value “close” to the reference (see figure 5.3) and control is then turned on.

To evaluate control performance a Tracking Performance Index (TPI) is defined:

$$TPI(\%) = \left( \sum_{n=BCT}^{tf} \frac{|ref(n) - BIS_{sim}(n)|}{ref(n)} \right) \frac{T_s}{BCT - tf} \times 100$$ (5.4)

where BCT is the begin control time (the instant where the induction phase ends) at 300 seconds. $tf$ is the final simulation time, in this case 10000 seconds, $ref$ is the reference, $BIS_{sim}$ is the simulated BIS value and $T_s$ is the sampling time of 5 seconds. The mean [TPI] for the 8 models in each parameter configuration $(d_i, d_o, \lambda)$ are presented in tables D.11 to D.12.

The results are rather inconclusive apart from the fact that obtaining good prediction performance is an insufficient criterium for obtaining a good controller. In fact, some odd results appear that prove this fact.

In the specific simulation depicted on figure 5.4 prediction error between the predicted output and BIS is small and yet control fails completely. The situation seems to be caused by model number
5. Simulations

Figure 5.4: Simulation result for test model 1 with $d_o = 1 \times 10^{-1}$, $d_i = 1 \times 10^{-1}$ and $\lambda = 0.999$.

one, the local positive command increment for controller 1 associated with its weight is of larger amplitude than that of controller 2, resulting in positive command increments being given to the plant. This is wrong considering what is known about the DoA model, to raise BIS the propofol infusion rate has to be reduced and vice-versa.

5.3 Control algorithm analysis

Figure 5.5: Remifentanil model disturbance

In this section the best mean model obtained in the previous section (with parameters $d_i = 1 \times 10^0$, $d_o = 1 \times 10^1$ and $\lambda = 0.999$) is used to synthesize a controller block. The idea is to use that controller block as the initial model for the control simulations. While the simulation is running new training data is generated and then integrated into LWPR and the controller block is readjusted if needed (see section 4.2.2.B).

The simulations are made for the 8 test patients in a similar way to the previous section. An initial induction phase is simulated like it is described in figure 5.3. Control is then turned on, the interval to update the controller being chosen to be 250 seconds.
5.3 Control algorithm analysis

Table 5.2: TPI(%) for different R and f’s.

<table>
<thead>
<tr>
<th>R/f</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2.592</td>
<td>1.725</td>
<td>1.383</td>
<td>1.269</td>
</tr>
<tr>
<td>10</td>
<td>2.825</td>
<td>1.915</td>
<td>1.464</td>
<td>1.428</td>
</tr>
<tr>
<td>100</td>
<td>7.000</td>
<td>2.431</td>
<td>2.102</td>
<td>1.869</td>
</tr>
<tr>
<td>1000</td>
<td>14.427</td>
<td>5.119</td>
<td>4.871</td>
<td>3.812</td>
</tr>
</tbody>
</table>

A test was made repeating the simulation for different combinations of R and f for the local controllers and with a remifentanil concentration acting as model disturbance (see chapter 2) represented in figure 5.5. The test was done for the 8 test models with each parameter configuration (R,f), results for the average TPI for the 8 test models in each parameter configuration (R, f) are presented in table 5.2.

Observing the results the influence of R is clearly seen, with larger R’s the tracking performance index is worse. This was to be expected since enforcing a larger command penalty in (4.12) will restrict command amplitude in each local controller, and make the controller block response slower. Table 5.2 also shows a loss of sensitivity to R as it decreases, the performance gain when R is reduced from 1000 to 100 or 100 to 10 is far greater than when R is reduced from 10 to 1. This can be explained by the existing propofol saturation at zero, which poses a limit on command amplitude, allied to the fact that the DoA model is by nature slow to respond to fast transitions in commands.

The influence of the observer poles placement factor f is also notorious in table 5.2 since the test was made in situation without noise choosing faster observer poles will improve the controller response.

![Figure 5.6: Simulation results for plant input and output with R = 100 and f = 3 for test model 1](image)

Since the use of TPI only provides a general idea on control performance, figure 5.6 shows the results for test model one that will be used as an example in this section. The tracking performance in the results shown on figure 5.6 is good although a bit slow, with a
settling time between 8 and 12 minutes. The disturbance (figure 5.5) appears to have no effect on control performance.

Figure 5.7: Normalized weights evolution for test model 1 with $R=100 \ f = 3$

Figure 5.7 shows the normalized weight evolution for each controller. Only two models are dominant in this case, this was to be expected since the working area for control is always the same. The controller block remained the same during the whole simulation. No new models were added, which confirms expectations since the mean model has seen extensive training in the working area and thus RFs were already allocated in that area. Adding new data to the mean LWPR model produced no significant changes the currently in use models, in fact, the maximum value obtained in the $\mu_{gap}$ metric comparisons (see section 4.2.2.B) during the whole simulation was very close to zero.

A test was also performed with a different reference to see the control behavior in a wider working area, the reference is set as a disturbed ramp between BIS levels 40 an 80.

Again results shown in figure 5.8 are rather good, the controller is able to drive the system in a wider working area. Looking at the weights for each controller represented in 5.9 the switching
5.3 Control algorithm analysis

Figure 5.9: Normalized weights with wide range reference with $R = 100$ and $f = 3$

mechanism between controllers is notorious, in the beginning of control controllers three and four are dominant, as the system changes working area controllers one and two start gaining more relevance.

Gaussian white noise with $\sigma = 1$ is added to the plant output in the next simulation, the reference is again set between 35 and 45.

Figure 5.10: Plant input and output with gaussian noise $\sigma = 1$ and $R = 100 \ f = 3$

Figure 5.11: Normalized weight evolution with gaussian noise $\sigma = 1$ and $R = 100 \ f = 3$

The results shown on figure are not so good, the control algorithm even fails tracking almost at the end of simulation. Three controller block updates occurred during simulation: controller four
was updated at 2245 seconds; controller 3 was updated at 3295 seconds and finally controller 6 is updated at 9045s. When each of these controllers was updated the $\mu_{gap}$ for the currently in use model associated with that controller and the new model was always 1. This means the previous models used to synthesize these controllers were completely different in a control perspective from the new ones obtained from LWPR at the indicated times, thus completely different controllers from the ones that existed before had to be synthesized.

The first update at 2245 second appears to have no influence on control despite the fact that the updated controller was dominant at that time. However this is was not the case for the two remaining updates, after the update at 3295 seconds the response becomes oscillatory, after the update at 9045 control fails completely.

To prove that the model updates caused the problem the test was repeated exactly with the same noise signal, but with controller block update turned off, the response is clearly better than in the previous case. Adding new training data corrupted with noise appears to be causing severe variations in local model estimations.

Figure 5.12: Results with controller block update turned off
Conclusions

6.1 Conclusions ......................................................... 49
6.2 Future Work ......................................................... 49
6.1 Conclusions

The objective for this dissertation was to develop a control strategy using a LWL method to identify a patient's dynamics. The basic assumption was that if a good prediction model was obtained from the LWL method it would represent the dynamic behavior of the real system and a controller based on the prediction model could be used for control purposes.

Both learning algorithms (LWPR and RFWR) were tested in terms of prediction performance in a comparative analysis. Despite having poorer prediction performance, LWPR proved to be more reliable due to its PLS implementation (more resistant to the multicollinearity problem) and thus more suitable for control.

The control strategy could be validated without noise when using a mean model for the initialization of the algorithm. The algorithm achieved satisfactory results, controlling different patient models in a wide working area with a time variant model disturbance. However when noise was added control, performance degraded significantly.

Control performance variation exists between different LWPR parameters configurations that obtain good prediction performance. The prediction performance alone does not provide guarantee of adequate control with this strategy.

6.2 Future Work

Since the control algorithm showed potential with some LWPR parameter configurations. The first step would be to continue to study the problem and try to find an heuristic to define the learning parameters. The use of distance matrix adaptation should also be explored as it has the potential of providing better prediction performance.
Bibliography


Variance inflation factor
Consider a general multiple regression problem

\[ y = b_0 + b_1 x_1 + b_2 x_2 + \cdots + b_j x_j \Leftrightarrow y = \beta X + b_0 \]  

(A.1)

with \( \beta = [b_1 \ b_2 \ \cdots \ b_j] \), \( x = [x_1 \ x_2 \ \cdots \ x_j]^T \) and \( b_0 \) scalar. Solved by obtaining data matrices \( X \) and \( Y \) of \( p \) training patterns \( x, y \) \( X = [x^1 \cdots x^p]^T \) and \( Y = [y^1 \cdots y^p]^T \) and using the batch least squares solution:

\[ \beta = \tilde{X}^T \tilde{X}^{-1} \tilde{X}^T \tilde{Y} \]  

(A.2)

where \( \tilde{X} \) and \( \tilde{Y} \) correspond to the mean zero matrices of \( X \) and \( Y \), that is, \( \tilde{X} = X - \mu_X \) and \( \tilde{Y} = Y - \mu_Y \).

Multicolinearity occurs when two or more of the explanatory variables \( x_j \) are highly correlated, that is, they exhibit a nearly linear relation between them. This constitutes a problem from a regression standpoint because it is not possible to determine the individual contribution of each of those variables to \( y \). In other words, for a well defined regression model the explanatory variables are assumed to be independent, in this case the affected variables are not or, at least, the training data \( X \) provided is insufficient to distinguish the individual contribution of some variables.

This fact translates mathematically to an ill defined \( \tilde{X}^T \tilde{X} \) matrix in equation (A.2) almost non-invertible, leading to a high variability on the parameter estimates \( \beta \) related to the affected variables if new training data is added to the regression model.

The variance inflation factor matrix for each RF, \( VIF \), is obtained using the relation between the correlation matrix and the covariance matrix for \( \tilde{X} \) where \( I \) is the identity matrix:

\[ \text{corr}(\tilde{X}, \tilde{X}) = \frac{\text{cov}(\tilde{X}, \tilde{X})}{\sigma^2_X I} \Leftrightarrow \text{corr}(\tilde{X}, \tilde{X})^{-1} = \text{cov}(\tilde{X}, \tilde{X})^{-1} \sigma^2_X I; \]  

(A.4)

substituting (A.3) in (A.4):

\[ VIF = \text{diag}(\text{corr}(\tilde{X}, \tilde{X})^{-1}) = \text{diag}(\text{cov}(\tilde{X}, \tilde{X})^{-1}) \sigma^2_X I \]  

(A.5)
A. Variance inflation factor
Partial Least Squares
Consider a general multiple regression problem

\[ y = b_0 + b_1 x_1 + b_2 x_2 + \cdots + b_j x_n \Leftrightarrow y = \beta X + b_0 \]  

(B.1)

with \( \beta = [b_1 \quad b_2 \quad \cdots \quad b_j] \), \( x = [x_1 \quad x_2 \quad \cdots \quad x_j]^T \) and \( b_0 \) scalar. Were \( p \) training patterns \( x, y \) are contained in data matrices \( X = [x^1 \cdots x^p]^T \) and \( Y = [y^1 \cdots y^p]^T \).

In PLS the main idea is to find a set of orthogonal directions \( U = [u_1 \cdots u_n] \) where the data \( X \) is projected and where univariate regression is then performed along each of the directions. The directions \( U \) are found in a recursive manner to better explain the relation between the input data \( X \) and the output data \( Y \). Looking at the batch recursive implementation for PLS the idea becomes clearer: The algorithm starts by considering the entire \( X \) data and \( Y \) data, a direction \( u_n \) is chosen as the direction of maximum correlation between \( X \) and \( Y \).

\[ u_n = X^T Y \]  

(B.2)

the data \( X \) is then projected in the \( u_n \) direction resulting in the transformed data \( \phi \).

\[ \phi = X u_n \]  

(B.3)

regression against \( Y \) is then performed in the transformed space obtaining parameter \( \beta_n \) for that direction.

\[ \beta_n = \frac{\phi^T Y}{\phi^T \phi} \]  

(B.4)

regression is also performed against \( X \) to ensure the orthogonality between the directions \( u_n \) obtaining the parameter \( p_n \)

\[ p_n = \frac{X^T \phi}{\phi^T \phi} \]  

(B.5)

finally the residuals \( X_{res} \) and \( Y_{res} \) are calculated using the \( \beta_n \) and \( p_n \)

\[ X_{res} = X - p_n \phi \]  

(B.6)

\[ Y_{res} = Y - \beta_n \phi \]  

(B.7)

(B.8)

the residuals can be interpreted as the "portion" of data form \( X \) and \( Y \) that is not correctly explained by the first direction considered. The cycle previously described is then repeated but now using the new residuals, that is, \( X = X_{res} \) and \( Y = Y_{res} \) in order to find an additional direction.
B. Partial Least Squares
The Vinnicombe norm also known as the \( \mu \text{gap} \) metric constitutes a measure of "proximity" between two models from a closed loop control perspective. It is defined as:

\[
\delta_{\nu}(M_1, M_2) = \| (I + M_2M_2^*)^{-\frac{1}{2}}(M_2 - M_1)(I + M_1^*M_1)^{-\frac{1}{2}} \|_{\infty}
\]  

(C.1)

where \( M_1 \) and \( M_2 \) constitute two models with the same input-output dimension and \( I \) is the identity matrix. If two models are "close" in the Vinnicombe norm a controller that stabilizes one of them should stabilize the other. The metric in (C.1) takes values between 0 and 1, a 1 value means \( M_1 \) and \( M_2 \) are far apart, in opposition a 0 value means \( M_1 \) and \( M_2 \) are equal. To show the usefulness of the \( \mu \text{gap} \) metric consider the following example with two simple models:

\[
M_1 = \frac{1}{z^{0.99}} \quad M_2 = \frac{1}{z^{1.01}}
\]

looking at the transfer function one can see \( M_1 \) is stable and \( M_2 \) is unstable, considering only this fact could lead to the wrong conclusion that a controller would be necessary for each model. In fact a single controller can stabilize \( M_1 \) and \( M_2 \) an example is choosing for instance a a feedback gain \( K = 1 \) for the closed loop. Using the Vinnicombe norm for this case a \( \delta(M_1, M_2) \approx 0.001 \) is obtained which confirms that the models despite completely different in response are close form a control perspective.
Contents

D.1 Prediction performance tests ........................................ 67
D.2 Control performance tests .......................................... 72
### D.1 Prediction performance tests

Table D.1: Mean squared error for $\lambda = 1$ (a -1 value indicates total prediction failure)

<table>
<thead>
<tr>
<th>d/dt</th>
<th>1.00E-003</th>
<th>1.00E-002</th>
<th>1.00E-001</th>
<th>1.00E+000</th>
<th>1.00E+001</th>
<th>1.00E+002</th>
</tr>
</thead>
<tbody>
<tr>
<td>1,00E-003</td>
<td>6.06E-01</td>
<td>6.06E-01</td>
<td>6.13E-01</td>
<td>5.73E-01</td>
<td>7.56E-01</td>
<td>7.86E-01</td>
</tr>
<tr>
<td></td>
<td>5.46E-05</td>
<td>5.46E-05</td>
<td>5.49E-05</td>
<td>5.49E-05</td>
<td>5.09E-05</td>
<td>4.81E-05</td>
</tr>
<tr>
<td>1,00E-002</td>
<td>6.09E-01</td>
<td>6.10E-01</td>
<td>6.17E-01</td>
<td>4.68E-01</td>
<td>7.53E-01</td>
<td>7.95E-01</td>
</tr>
<tr>
<td></td>
<td>5.46E-05</td>
<td>5.45E-05</td>
<td>5.45E-05</td>
<td>5.49E-05</td>
<td>5.09E-05</td>
<td>4.80E-05</td>
</tr>
<tr>
<td>1,00E-001</td>
<td>3.12E+00</td>
<td>3.07E+00</td>
<td>2.67E+00</td>
<td>8.64E-01</td>
<td>8.93E-01</td>
<td>6.17E-01</td>
</tr>
<tr>
<td></td>
<td>5.44E-05</td>
<td>5.44E-05</td>
<td>5.40E-05</td>
<td>5.09E-05</td>
<td>4.80E-05</td>
<td>4.29E-05</td>
</tr>
<tr>
<td>1,00E-000</td>
<td>1.04E+00</td>
<td>1.03E+00</td>
<td>1.02E+00</td>
<td>6.85E-01</td>
<td>7.93E-01</td>
<td>6.91E-01</td>
</tr>
<tr>
<td></td>
<td>5.19E-05</td>
<td>5.19E-05</td>
<td>5.16E-05</td>
<td>4.90E-05</td>
<td>4.63E-05</td>
<td>4.01E-05</td>
</tr>
<tr>
<td>1,00E+001</td>
<td>4.63E-01</td>
<td>4.63E-01</td>
<td>4.63E-01</td>
<td>4.63E-01</td>
<td>3.84E-01</td>
<td>3.91E-01</td>
</tr>
<tr>
<td></td>
<td>4.63E-05</td>
<td>4.63E-05</td>
<td>4.63E-05</td>
<td>4.63E-05</td>
<td>4.63E-05</td>
<td>4.63E-05</td>
</tr>
<tr>
<td>1,00E+002</td>
<td>3.46E-01</td>
<td>3.45E-01</td>
<td>3.45E-01</td>
<td>3.45E-01</td>
<td>2.78E-01</td>
<td>2.66E-01</td>
</tr>
<tr>
<td></td>
<td>4.53E-05</td>
<td>4.53E-05</td>
<td>4.52E-05</td>
<td>4.49E-05</td>
<td>4.45E-05</td>
<td>4.41E-05</td>
</tr>
</tbody>
</table>

Table D.2: Mean squared error for $\lambda = 0.999$ (a -1 value indicates total prediction failure)

<table>
<thead>
<tr>
<th>d/dt</th>
<th>1.00E-003</th>
<th>1.00E-002</th>
<th>1.00E-001</th>
<th>1.00E+000</th>
<th>1.00E+001</th>
<th>1.00E+002</th>
</tr>
</thead>
<tbody>
<tr>
<td>1,00E-003</td>
<td>1.35E-02</td>
<td>1.56E-02</td>
<td>8.96E-03</td>
<td>1.58E-01</td>
<td>6.07E-03</td>
<td>2.22E-01</td>
</tr>
<tr>
<td></td>
<td>-1.00E+00</td>
<td>-1.00E+00</td>
<td>-1.00E+00</td>
<td>-1.00E+00</td>
<td>-1.00E+00</td>
<td>-1.00E+00</td>
</tr>
<tr>
<td>1,00E-002</td>
<td>9.87E-03</td>
<td>1.12E-02</td>
<td>1.73E-02</td>
<td>1.52E-01</td>
<td>6.11E-03</td>
<td>2.19E-01</td>
</tr>
<tr>
<td></td>
<td>-1.00E+00</td>
<td>-1.00E+00</td>
<td>-1.00E+00</td>
<td>-1.00E+00</td>
<td>-1.00E+00</td>
<td>-1.00E+00</td>
</tr>
<tr>
<td>1,00E-001</td>
<td>8.68E-03</td>
<td>9.19E-03</td>
<td>1.13E-02</td>
<td>1.25E-01</td>
<td>4.93E-03</td>
<td>2.31E-01</td>
</tr>
<tr>
<td></td>
<td>-1.00E+00</td>
<td>-1.00E+00</td>
<td>-1.00E+00</td>
<td>-1.00E+00</td>
<td>-1.00E+00</td>
<td>-1.00E+00</td>
</tr>
<tr>
<td>1,00E-000</td>
<td>2.00E-01</td>
<td>2.00E-01</td>
<td>1.96E-01</td>
<td>5.39E-02</td>
<td>7.67E-02</td>
<td>2.83E-01</td>
</tr>
<tr>
<td></td>
<td>-1.00E+00</td>
<td>-1.00E+00</td>
<td>-1.00E+00</td>
<td>-1.00E+00</td>
<td>-1.00E+00</td>
<td>-1.00E+00</td>
</tr>
<tr>
<td>1,00E+001</td>
<td>2.11E-02</td>
<td>2.11E-02</td>
<td>2.16E-02</td>
<td>9.34E-02</td>
<td>3.76E-02</td>
<td>5.08E-02</td>
</tr>
<tr>
<td></td>
<td>-1.00E+00</td>
<td>-1.00E+00</td>
<td>-1.00E+00</td>
<td>-1.00E+00</td>
<td>-1.00E+00</td>
<td>-1.00E+00</td>
</tr>
<tr>
<td>1,00E+002</td>
<td>1.26E-01</td>
<td>1.26E-01</td>
<td>1.26E-01</td>
<td>1.35E-01</td>
<td>5.02E-02</td>
<td>5.73E-02</td>
</tr>
<tr>
<td></td>
<td>-1.00E+00</td>
<td>-1.00E+00</td>
<td>-1.00E+00</td>
<td>-1.00E+00</td>
<td>-1.00E+00</td>
<td>-1.00E+00</td>
</tr>
</tbody>
</table>
### D. Simulation Tables

#### Table D.3: Mean squared error for $\lambda = 0.995$ (a -1 value indicates total prediction failure)

<table>
<thead>
<tr>
<th>dldo</th>
<th>1.00E-003</th>
<th>1.00E-002</th>
<th>1.00E-001</th>
<th>1.00E+000</th>
<th>1.00E+001</th>
<th>1.00E+002</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.00E-003</td>
<td>1.21E+001</td>
<td>1.22E+001</td>
<td>1.12E+001</td>
<td>1.78E+000</td>
<td>1.33E+000</td>
<td>2.35E-001</td>
</tr>
<tr>
<td></td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>1.33E-004</td>
</tr>
<tr>
<td>1.00E-002</td>
<td>1.25E+001</td>
<td>1.29E+001</td>
<td>1.43E+001</td>
<td>1.81E+000</td>
<td>3.77E-001</td>
<td>1.40E-001</td>
</tr>
<tr>
<td></td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>1.33E-004</td>
</tr>
<tr>
<td>1.00E-001</td>
<td>1.41E+001</td>
<td>1.42E+001</td>
<td>1.42E+001</td>
<td>6.45E-002</td>
<td>1.44E-002</td>
<td>1.85E-001</td>
</tr>
<tr>
<td></td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
</tr>
<tr>
<td>1.00E+000</td>
<td>4.18E-001</td>
<td>4.17E-001</td>
<td>4.03E-001</td>
<td>8.52E-002</td>
<td>3.49E-002</td>
<td>4.36E-002</td>
</tr>
<tr>
<td></td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
</tr>
<tr>
<td>1.00E-001</td>
<td>1.54E+001</td>
<td>1.55E+001</td>
<td>1.59E+001</td>
<td>9.28E-002</td>
<td>2.11E-002</td>
<td>3.93E-002</td>
</tr>
<tr>
<td></td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>8.60E-005</td>
</tr>
<tr>
<td>1.00E+002</td>
<td>1.15E-002</td>
<td>1.14E-002</td>
<td>1.13E-002</td>
<td>1.76E-002</td>
<td>1.64E-002</td>
<td>3.47E-002</td>
</tr>
<tr>
<td></td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>9.37E-005</td>
</tr>
</tbody>
</table>

#### Table D.4: Mean squared error for $\lambda = 0.950$ (a -1 value indicates total prediction failure)

<table>
<thead>
<tr>
<th>dldo</th>
<th>1.00E-003</th>
<th>1.00E-002</th>
<th>1.00E-001</th>
<th>1.00E+000</th>
<th>1.00E+001</th>
<th>1.00E+002</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.00E-003</td>
<td>1.23E+003</td>
<td>1.23E+003</td>
<td>1.24E+003</td>
<td>4.13E-002</td>
<td>2.11E+000</td>
<td>3.94E+000</td>
</tr>
<tr>
<td></td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>3.44E-004</td>
</tr>
<tr>
<td>1.00E-002</td>
<td>1.29E+003</td>
<td>1.29E+003</td>
<td>1.25E+003</td>
<td>7.78E-003</td>
<td>4.90E+000</td>
<td>3.97E+000</td>
</tr>
<tr>
<td></td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>3.40E-004</td>
</tr>
<tr>
<td>1.00E-001</td>
<td>1.73E+003</td>
<td>1.69E+003</td>
<td>1.28E+003</td>
<td>7.14E+001</td>
<td>3.08E+02</td>
<td>2.48E+000</td>
</tr>
<tr>
<td></td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>2.48E+000</td>
</tr>
<tr>
<td>1.00E+000</td>
<td>1.01E+002</td>
<td>1.00E+002</td>
<td>8.53E+000</td>
<td>2.34E+002</td>
<td>2.31E+000</td>
<td>5.95E-001</td>
</tr>
<tr>
<td></td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
</tr>
<tr>
<td>1.00E-001</td>
<td>1.81E+002</td>
<td>1.86E+002</td>
<td>2.56E+002</td>
<td>8.62E+000</td>
<td>1.86E-001</td>
<td>3.39E-001</td>
</tr>
<tr>
<td></td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>1.55E-004</td>
</tr>
<tr>
<td>1.00E+002</td>
<td>7.32E+000</td>
<td>7.52E+000</td>
<td>8.91E+000</td>
<td>7.07E+000</td>
<td>3.32E-002</td>
<td>9.65E-001</td>
</tr>
<tr>
<td></td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>2.25E-004</td>
</tr>
</tbody>
</table>
### D.1 Prediction performance tests

#### Table D.5: Number of RFs

<table>
<thead>
<tr>
<th>dito</th>
<th>1.00E-003</th>
<th>1.00E-002</th>
<th>1.00E-001</th>
<th>1.00E+000</th>
<th>1.00E+001</th>
<th>1.00E+002</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.00E-003</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>4</td>
<td>12</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>6</td>
<td>17</td>
</tr>
<tr>
<td>1.00E-002</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>4</td>
<td>12</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>6</td>
<td>17</td>
</tr>
<tr>
<td>1.00E-001</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>3</td>
<td>6</td>
<td>17</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>4</td>
<td>10</td>
<td>29</td>
</tr>
<tr>
<td>1.00E+000</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>5</td>
<td>14</td>
<td>34</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>6</td>
<td>6</td>
<td>9</td>
<td>26</td>
<td>60</td>
</tr>
<tr>
<td>1.00E+001</td>
<td>11</td>
<td>11</td>
<td>11</td>
<td>14</td>
<td>30</td>
<td>74</td>
</tr>
<tr>
<td></td>
<td>15</td>
<td>15</td>
<td>16</td>
<td>22</td>
<td>51</td>
<td>131</td>
</tr>
<tr>
<td>1.00E+002</td>
<td>30</td>
<td>30</td>
<td>30</td>
<td>33</td>
<td>69</td>
<td>156</td>
</tr>
<tr>
<td></td>
<td>45</td>
<td>45</td>
<td>45</td>
<td>61</td>
<td>113</td>
<td>243</td>
</tr>
</tbody>
</table>

#### Table D.6: Number of trustworthy RFs

<table>
<thead>
<tr>
<th>dito</th>
<th>1.00E-003</th>
<th>1.00E-002</th>
<th>1.00E-001</th>
<th>1.00E+000</th>
<th>1.00E+001</th>
<th>1.00E+002</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.00E-003</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>4</td>
<td>12</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>3</td>
<td>14</td>
</tr>
<tr>
<td>1.00E-002</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>4</td>
<td>12</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>3</td>
<td>14</td>
</tr>
<tr>
<td>1.00E-001</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>3</td>
<td>6</td>
<td>17</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>15</td>
</tr>
<tr>
<td>1.00E+000</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>5</td>
<td>14</td>
<td>33</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>4</td>
<td>17</td>
</tr>
<tr>
<td>1.00E+001</td>
<td>11</td>
<td>11</td>
<td>11</td>
<td>14</td>
<td>27</td>
<td>59</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>3</td>
<td>6</td>
<td>22</td>
</tr>
<tr>
<td>1.00E+002</td>
<td>26</td>
<td>26</td>
<td>26</td>
<td>29</td>
<td>59</td>
<td>113</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>5</td>
<td>9</td>
<td>32</td>
</tr>
</tbody>
</table>
### Table D.7: Prediction failure (%) for $\lambda = 1$

<table>
<thead>
<tr>
<th>dlido</th>
<th>1.00E-003</th>
<th>1.00E-002</th>
<th>1.00E-001</th>
<th>1.00E+000</th>
<th>1.00E+001</th>
<th>1.00E+002</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.00E-003</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>2.7</td>
<td>0</td>
</tr>
<tr>
<td>1.00E-002</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>2.7</td>
<td>0</td>
</tr>
<tr>
<td>1.00E-001</td>
<td>0.04</td>
<td>0.04</td>
<td>0.04</td>
<td>0</td>
<td>2.16</td>
<td>0.03</td>
</tr>
<tr>
<td>1.00E+000</td>
<td>0.18</td>
<td>0.18</td>
<td>0.18</td>
<td>0.31</td>
<td>2.05</td>
<td>0.27</td>
</tr>
<tr>
<td>1.00E+001</td>
<td>2.01</td>
<td>2.01</td>
<td>2.01</td>
<td>2.02</td>
<td>3.78</td>
<td>1.26</td>
</tr>
<tr>
<td>1.00E+002</td>
<td>3.68</td>
<td>3.68</td>
<td>3.68</td>
<td>3.63</td>
<td>6.54</td>
<td>4.18</td>
</tr>
</tbody>
</table>

### Table D.8: Prediction failure (%) for $\lambda = 0.999$

<table>
<thead>
<tr>
<th>dlido</th>
<th>1.00E-003</th>
<th>1.00E-002</th>
<th>1.00E-001</th>
<th>1.00E+000</th>
<th>1.00E+001</th>
<th>1.00E+002</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.00E-003</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1.00E-002</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>84.39</td>
</tr>
<tr>
<td>1.00E-001</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1.00E+000</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1.00E+001</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1.00E+002</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1.00E+003</td>
<td>0.02</td>
<td>0.02</td>
<td>0.02</td>
<td>0.02</td>
<td>0.01</td>
<td>0.07</td>
</tr>
<tr>
<td>1.00E+002</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>88.42</td>
</tr>
</tbody>
</table>
D.1 Prediction performance tests

Table D.9: Prediction failure (%) for $\lambda = 0, 995$

<table>
<thead>
<tr>
<th>d/d/o</th>
<th>1,00E-003</th>
<th>1,00E-002</th>
<th>1,00E-001</th>
<th>1,00E+000</th>
<th>1,00E+001</th>
<th>1,00E+002</th>
</tr>
</thead>
<tbody>
<tr>
<td>1,00E-003</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
</tr>
<tr>
<td>1,00E-002</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
</tr>
<tr>
<td>1,00E-001</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
</tr>
<tr>
<td>1,00E+000</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0,01</td>
</tr>
<tr>
<td></td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
</tr>
<tr>
<td>1,00E+001</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
</tr>
<tr>
<td>1,00E+002</td>
<td>0,02</td>
<td>0,02</td>
<td>0,02</td>
<td>0,02</td>
<td>0,01</td>
<td>0,07</td>
</tr>
<tr>
<td></td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
</tr>
</tbody>
</table>

Table D.10: Prediction failure (%) for $\lambda = 0, 950$

<table>
<thead>
<tr>
<th>d/d/o</th>
<th>1,00E-003</th>
<th>1,00E-002</th>
<th>1,00E-001</th>
<th>1,00E+000</th>
<th>1,00E+001</th>
<th>1,00E+002</th>
</tr>
</thead>
<tbody>
<tr>
<td>1,00E-003</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
</tr>
<tr>
<td>1,00E-002</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
</tr>
<tr>
<td>1,00E-001</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
</tr>
<tr>
<td>1,00E+000</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0,01</td>
</tr>
<tr>
<td></td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
</tr>
<tr>
<td>1,00E+001</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
</tr>
<tr>
<td>1,00E+002</td>
<td>0,04</td>
<td>0,04</td>
<td>0,04</td>
<td>0,04</td>
<td>0,03</td>
<td>0,07</td>
</tr>
<tr>
<td></td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
</tr>
</tbody>
</table>
### D.2 Control performance tests

Table D.11: Tracking performance index(%) for $\lambda = 1$

<table>
<thead>
<tr>
<th>dïïdï</th>
<th>1,00E-003</th>
<th>1,00E-002</th>
<th>1,00E-001</th>
<th>1,00E+000</th>
<th>1,00E+001</th>
</tr>
</thead>
<tbody>
<tr>
<td>1,00E-003</td>
<td>1,949</td>
<td>1,950</td>
<td>1,828</td>
<td>87,333</td>
<td>57,810</td>
</tr>
<tr>
<td>1,00E-002</td>
<td>1,951</td>
<td>1,952</td>
<td>1,958</td>
<td>86,715</td>
<td>84,343</td>
</tr>
<tr>
<td>1,00E-001</td>
<td>1,959</td>
<td>1,975</td>
<td>2,270</td>
<td>55,879</td>
<td>87,723</td>
</tr>
<tr>
<td>1,00E+000</td>
<td>2,359</td>
<td>2,362</td>
<td>2,391</td>
<td>2,540</td>
<td>81,259</td>
</tr>
<tr>
<td>1,00E+001</td>
<td>3,755</td>
<td>3,757</td>
<td>3,779</td>
<td>3,351</td>
<td>17,219</td>
</tr>
</tbody>
</table>

Table D.12: Tracking performance index(%) for $\lambda = 0.999$

<table>
<thead>
<tr>
<th>dïïdï</th>
<th>1,00E-003</th>
<th>1,00E-002</th>
<th>1,00E-001</th>
<th>1,00E+000</th>
<th>1,00E+001</th>
</tr>
</thead>
<tbody>
<tr>
<td>1,00E-003</td>
<td>87,416</td>
<td>87,424</td>
<td>87,371</td>
<td>87,729</td>
<td>2,145</td>
</tr>
<tr>
<td>1,00E-002</td>
<td>87,403</td>
<td>87,408</td>
<td>87,428</td>
<td>87,680</td>
<td>18,328</td>
</tr>
<tr>
<td>1,00E-001</td>
<td>81,730</td>
<td>81,755</td>
<td>81,523</td>
<td>1,649</td>
<td>1,464</td>
</tr>
<tr>
<td>1,00E+000</td>
<td>2,033</td>
<td>2,033</td>
<td>2,034</td>
<td>84,673</td>
<td>2,097</td>
</tr>
<tr>
<td>1,00E+001</td>
<td>2,689</td>
<td>2,690</td>
<td>2,908</td>
<td>25,663</td>
<td>68,552</td>
</tr>
</tbody>
</table>
Table D.13: Tracking performance index(%) for $\lambda = 0.995$

<table>
<thead>
<tr>
<th>diïdo</th>
<th>1,00E-003</th>
<th>1,00E-002</th>
<th>1,00E-001</th>
<th>1,00E+000</th>
<th>1,00E+01</th>
</tr>
</thead>
<tbody>
<tr>
<td>1,00E-003</td>
<td>87.59</td>
<td>87.59</td>
<td>87.59</td>
<td>87.51</td>
<td>87.62</td>
</tr>
<tr>
<td>1,00E-002</td>
<td>87.59</td>
<td>87.59</td>
<td>87.59</td>
<td>87.53</td>
<td>87.62</td>
</tr>
<tr>
<td>1,00E-001</td>
<td>6.75</td>
<td>6.76</td>
<td>6.98</td>
<td>87.72</td>
<td>87.44</td>
</tr>
<tr>
<td>1,00E+000</td>
<td>7.83</td>
<td>7.83</td>
<td>7.8</td>
<td>1.99</td>
<td>1.6</td>
</tr>
<tr>
<td>1,00E+001</td>
<td>7.16</td>
<td>7.17</td>
<td>7.22</td>
<td>79.82</td>
<td>4.93</td>
</tr>
</tbody>
</table>
D. Simulation Tables