Automation in Anesthesia: A Look at $\mathcal{L}_1$ Adaptive and PID Controllers

by

Kousha Talebian

Bachelor of Engineering Physics - Electrical Option, The University of British Columbia, 2011

A THESIS SUBMITTED IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE DEGREE OF

Master of Applied Science

in

THE FACULTY OF GRADUATE AND POSTDOCTORAL STUDIES
(Biomedical Engineering)

The University Of British Columbia
(Vancouver)

December 2016

Abstract

Control of anesthesia is one of the many tasks performed by anesthesiologists during surgery. It involves adjusting drug dosage by monitoring patient's vital and clinical signs. A control system can replace this tedious and routine task, and allow the anesthesiologists to concentrate on more life threatening procedures.

Because of large intra- and inter-variability in patients Pharmacokinetics and Pharmacodynamics responses, an adaptive controller is desirable. This thesis thoroughly investigates the $\mathcal{L}_1$ Adaptive Control by applying it on 44 simulation cases which cover a wide range of patient demographics. It is found that the controller approaches an implantable non-adaptive LTI controller as the adaptation gain increases, echoing the results found by other researches. This loss of adaptivity is shown through examples and mathematical derivations. It is concluded that the $\mathcal{L}_1$ Adaptive Control in its current form is not applicable to closed-loop control of anesthesia.

As an alternative to adaptive controller, partial adaptivity in a PID controller is investigated. iControl, a PID controller designed by us, can sometimes lead to oscillation in the control signal. It is desirable to automatically detect the oscillations and tune the controller in order to remove them. A real-time oscillation detection algorithm is discussed. It detects multiple oscillations in real-time and provides their frequency, amplitude, severity and regularity. A PID auto-tuning algorithm is developed that uses the dominant frequency metrics provided by the oscillation detection algorithm to retune the controller robustly and to guarantee stability. This technique is simulated and tested on 44 cases; the gain and the phase margin in all 44 cases are within $< 7\%$ of the optimal tuning parameters of the iControl.

Preface

All of the work of this research was conducted at BC Children's Hospital in collaboration with The University of British Columbia's Electrical and Computer Engineering in Medicine department. All clinical data used was approved by The University of British Columbia Children's and Women's Research Ethic Board (certificate H10-01174).

I was the principal investigator on analysis of the $\mathcal{L}_1$ Adaptive Control ($\mathcal{L}_1$-AC) of Chatper 3 and its application to closed-loop control of anesthesia. A version of Chapter 3 is published in a journal article:

- K. van Heusden, K. Talebian, G. A. Dumont. Analysis of $\mathcal{L}_1$ adaptive state feedback control. Why does it approximate an implementable LTI controller? In *European Journal of Control*, 2015

The theory of oscillation detection algorithm is based on Wang et a.l [67] while the implementation, parameter tuning, and MATLAb code is based on my research and I was the lead investigator on its fisibility in clinical settings. The turning rules of Chapter 5 is from Åström et al. [6], however the implementation and the MATLAB code was provided by myself.

The Appendix B was a continuation of the work originally conducted by Soltesz, G. I applied his findings on a series of clinical data to determine and compare their performance to the well known Varvel metrics. These findings were presented at the *American Society of Anesthesiologist 2013*:

- K. Talebian, K. Soltesz, G. A. Dumont, M. Ansermino. Clinical assessment of control performance in closed-loop anesthesia. In *American Society of Anesthesiologist*, 2013.

Table of Contents

List of Tables

List of Figures

Glossary

BIS	Bispectral Index
CNS	Central Nervous System
DCT	Discrete Cosine Transform
DFT	Discrete Fourier Transform
DOA	Depth of Anesthesia
DOH	Depth of Hypnosis
DWT	Discrete Wavelet Transform
EEG	Electroencephalography
ER	Emergence Phase Rise Time
GABA	Gamma-Aminobutyric Acid
GS	Global Score
IAE	Integrated Absolute Error
IDCT	Inverse Discrete Cosine Transform
ID	Induction Phase Duration
IE	Integrated Error
$\mathcal{L}_1$**-AC**	$\mathcal{L}_1$ Adaptive Control

LBM Lean Body Mass

LTI Linear Time Invariant

MAP Mean Arterial Pressure

MDAPE Median Performance Absolute Error

MDPE Median Performance Error

MIMO Multi-Input/Multi-Output

MPC Model-Predictive-Controller

OS Percent Overshoot

PD Pharmacodynamics

PEEG processed Electroencephalography

PE Percent Error

PID Proportional-Integral-Derivative

PI Proportional-Integral

PKPD Pharmacokinetics/Pharmacodynamics

PK Pharmacokinetics

SISO Single-Input/Single-Output

SNR Signal to Noise Ratio

TCI Target-Controlled Infusion

VI Variability Index

WAV Wavelet-based Anesthetic Value

Acknowledgments

This research would not have been possible without the support of the following amazing individuals.

First, I would like to express my deepest gratitude to my research supervisor, Dr. Guy Dumont, for his guidance, valuable support and insights. I would also like to extend my gratitude to Dr. Mark Ansermino who always provided the time and support to answer my questions. Lastly, I would like to thank my colleague Dr. Klaske van Heusden who always discussed problems and helped me with the challenges of the research.

Second, and most importantly, I would like to thank my parents, Nora and Mano, for their constant support of my professional career, and my life as a whole. I owe everything I have achieved and hope to achieve to their love and support. In addition, I would like to thank my dear sister, Kimia, who always showed her support without being asked.

Third and finally, I would like to thank all my colleagues at the Pediatric Anesthesia Research Team, Sara Khosravi, Nick West, Parastoo Dehkordi, Joanne Lim, Dr. Matthias Görges, and Aryannah Umedaly.

Chapter 1

Introduction

1.1 Motivation

The number of surgeries are increasing in the US and around the globe. According to a 2010 report by the Centers for Disease Control and Presentation, 51.4 million patients in the US alone went under surgery that year [17]. These surgeries range anywhere from a life saving operation, to a plastic surgery, or to correcting and restoring the physical appearance [17].

Anesthesiologists play a central role in the health and comfort of the patient pre-, intra-, and post-operatively. During the procedure, they monitor and control patient's vital functions, including breathing, blood pressure, heart rate, body temperature, and etc. They ensure the safety and comfort of the patient by controlling the hypnotic (unconsciousness) and the analgesic (sense of pain) states as well as assisting the surgeons by controlling the paralysis (relaxation and immobility of the skeletal muscle) through administration of anesthetic drugs. *General anesthesia* is the term given to this complex state induced patient.

To achieve general anesthesia, the anesthesiologist administers a variety of drugs. The combination of hypnotic, opioid, and neuromuscular drugs achieves the three functional states of anesthesia: *hypnosis*, *analgesic*, and *paralysis*.

Anesthesiologists assess adequate anesthesia and analgesia through monitoring the patient's vital and other clinical signs such as heart rate, blood pressure, eye movement, pupil diameter, respiratory rate, facial grimacing, and lacrimation [18].

Depth of Anesthesia (DOA) is a measure of the effect of the hypnotic and analgesic drugs that cause the unconsciousness and alleviate pain [52].

The paralysis state does not contribute to the DOA [66]. The muscle relaxant drugs that induce paralysis can however, effect some of the patient's clinical signs, for example the respiratory rate. Anesthesiologists may therefore need to monitor other physiological signs as the paralysis state is induced.

There is currently no single variable that can accurately measure the DOA and the search for this metric remains an active field of research. Monitors use the Electroencephalography (EEG) - the electrical activity of the brain along the scalp, to assess Depth of Hypnosis (DOH)[1]. Interpretation of the raw EEG signal is time consuming and requires a trained neurologist. Therefore, different processed Electroencephalography (PEEG) methods have been applied to create an index that quantifies the state of hypnosis. The most commonly used index, the Bispectral Index (BIS), uses the bispectral analysis of the EEG wavelengths [50]. Our research team at the University of British Columbia (UBC) has worked extensively in this field; their efforts have resulted in the introduction of the Wavelet-based Anesthetic Value (WAV) index that compares well with the standard BIS, see [12], [69], and [11].

Opioid (narcotic) analgesic drugs produce their effect through the interaction with the Gamma-Aminobutyric Acid (GABA) receptors in the Central Nervous System (CNS), though they bind at different sites [37] than the hypnotic drugs. While there are proposed indices such as the Analgesia Nociception Index [36], nociception and antinociception measurements during anesthesia have not been clinically proven, see [24] and [30].

Anesthesiologists continuously change the administration rate of the hypnotic, analgesic and relaxant drugs to account for stimuli from surgical incision. In fact, they assume the role of a feedback controller; to achieve a given clinical target, the doctors monitor the clinical signs of the patient and adjust the drugs accordingly. In many instances, a computer controlled automation system can assist the anesthesiologist by taking over this tedious and routine task, and thus allowing the doctor to only be involved with outliers events that are life-threatening. This is similar to the

[1] DOH is a measure of depth of hypnosis only, while DOA is the measure of depth of hypnosis and depth of analgesia.

role of the auto-pilot that takes over the cruising of the plane from the pilot. The pilot intervenes during unforeseen and critical events only, but otherwise adjusts minor details.

Our group has shown that a closed-loop Proportional-Integral-Derivative (PID) controller (known as iControl) can effectively control the DOH, see [10], [52], [54] and [68].

To account for the large inter- and intra-variability in the patient's Pharmacokinetics (PK) and Pharmacodynamics (PD) response, any controller implemented must be robust. To be clinically feasible, the controller must perform well despite surgical stimuli and keep the patient's DOH at the specified target.

A set of performance measures allow to assess the quality of the control of an anesthetic machine. These measurements need to be of clinical significance to provide the clinicians with details above the anesthetic state of the patient. If the measurements could also provide insight of the control architecture to an engineer, then these merits can be used as tuning objectives. Currently, a set of four measurements merits, known as Varvel measures [65] have constituted the norm (see Section 2.4.

1.2 Objectives and Scopes

This research started as an assessment of the novel $\mathcal{L}_1$ Adaptive Control ($\mathcal{L}_1$-AC) [26] which yielded limited feasibility. The focus was then turned to detecting oscillation caused by a PID controller, and to develop an algorithm to remove the oscillations. A set of metrics are also introduced to quantify the use cases of the control algorithms.

The objectives of this thesis is then to 1) assess the application of novel $\mathcal{L}_1$-AC [26] as applied to closed-loop control in anesthesia using WAV index as the control signal; 2) design an oscillation detection algorithm that can detect multi-period oscillations in real time; and 3) develop a tuning algorithm that can re-tune the PID controller used in iControl to remove the detected oscillation.

Conventional control theory establishes a trade-off between robustness and performance. For the safety of the patient, the current closed-loop controllers of anesthesia value robustness over performance. According to its developers, $\mathcal{L}_1$-AC

guarantees robustness and its fast adaptation implementation implies a near perfect performance can be achieved. However, this thesis will show that this structure in its current form does not live up to its promise and therefore cannot be used.

The most commonly closed-loop anesthetic controller is a PID system. In [2], it was found that many of these controllers are incorrectly tuned and resulted in inferior performance as well as observed oscillation. The implemented PID controller at our facility is tuned to be robust with no oscillation. However in practice, some oscillations have been seen due to patient variability. Partial adaptation that can detect and remove oscillation in real-time is discussed. The retuning of the controller must consider the following 4 design criteria:

1. It must detect the dominant oscillation in real-time.
2. It must re-tune the PID controller according to the dominant frequency and ensure the new system follows the guidelines of a robust controller.
3. It must reject output disturbances and compensate for surgical stimuli.
4. It must remove the oscillation and provide adequate performance and set-point response.

To sufficiently compare different control schemes, the standard means of measuring the performance of closed-loop anesthesia are explored. The Varvel measures were introduced in 1992 for target-controlled-infusion systems but their adequacy for closed-loop control is debated. A proposed set of measures introduced by Soltesz et al. [53] is assessed on real clinical data.

1.3 Thesis Organization

This thesis is organized into 6 Chapters, with this Chapter contributing as one. Chapter 6 provides the closing remarks, conclusions and future works. The supporting materials for this thesis are organized into 4 Appendices. The Chapters and Appendices are:

Chapter 2: Background

This Chapter gives a brief overview of current practices in closed-loop anesthesia. It will introduce the currently used monitoring systems for depth of hypnosis, a review of the pharmacokinetics and pharmacodynamics of the propofol drug, as well as the metrics for performance measures in closed-loop anesthesia.

Chapter 3: $\mathcal{L}_1$ Adaptive Control

This Chapter uses an $\mathcal{L}_1$-AC to simulate the control of the propofol in patients. The results are shown to be in-line with claims that $\mathcal{L}_1$-AC fails to provide an adaptive algorithm and at best behaves as an *implementable* Linear Time Invariant (LTI) controller. The loss of this adaptivity is mathematically proven.

Chapter 4: Real-Time Oscillation Detection

This Chapter provides an off-line oscillation detection algorithm that is capable of detecting multiple oscillation frequencies, along with their fitness [2] and magnitude. An extension to real-time is also introduced where a dominant frequency can be measured.

Chapter 5: Re-Tuning of a PID Controller

This Chapter provides a tuning methodology to re-tune a PID controller when an oscillation is detected. The data from the previous Chapter is used to tune the controller. Simulation results show the robustness and performance of the re-tuned system agree with the current implementation of the iControl system.

Appendix A: Propofol PKPD Modeling

This Appendix provides an overview of the propofol Pharmacokinetics/Pharmacodynamics (PKPD) model introduced in Bibian [10]. The mathematical model and parameters are also included. These models are used for simulation examples in Chapter 3 and 5.

[2] A measure of energy of the oscillation as a percentage of the total energy of the signal

Appendix B: Control Performance in Closed-Loop Anesthesia

This Appendix provides the mathematical description of Varvel and the proposed alternative measures. The proposed measures are used as tuning objectives in Chapter 5.

Appendix C: Limiting Behavior of $\mathcal{L}_1$ Adaptive Control

This Appendix provides the mathematical proof for the loss of adaptivity in the $\mathcal{L}_1$-AC.

Appendix D: Robustness and Performance of iControl

This Appendix provides the complete robustness and performance comparison of the iControl and the re-tuned controller from Chapter 5. 44 simulation examples using the PKPD models are used in this study.

Chapter 2

Background

The field of Biomedical Engineering is on demand. In 2013, this profession was rated as the #2 with respect to overall satisfaction [16]. Interestingly enough, anesthesiologists were rated #1 with regard to income [51]. Many of the complex problems faced by clinicians can only be answered through the eyes of an biomedical engineer. A few examples of engineering solutions for medical diagnostics are given.

Control engineers traditionally have been focused on the aerospace and process industries, but have recently applied their knowledge to medical devices. The use of closed-loop control to administer drugs has been shown to improve the quality and safety as well as reducing the total administrated dosage (see [20] and [2]). More specifically, closed-loop control of drugs delivery has been an active area of research in anesthesia (see [39] and [39] as well as our own research group [52] and references within).

A closed-loop anesthesia system measures the DOH from a PEEG signal (such as BIS or WAV) and controls the infusion rate of the hypnotic drug. The most common hypnotic drug for closed-loop control is intravenous propofol due to its short-acting mechanism. The controller can take advantage of the short-acting, fast metabolic, and fast elimination of the drug and provide a much smoother infusion titration than an anesthesiologist would be able to do manually. By transferring the responsibility of these routine tasks to a computer, an anesthesiologist can concentrate on more vital tasks and the safety of the patient.

In the next few Sections, a review of the required components of a closed-loop control system for anesthesia is provided. First, the methods for measuring the DOH is provided. Second, the modeling of the drug effect in a patient is described. Third, prior attempts at the automatic control of anesthesia are reviewed. Fourth, causes and concerns for oscillations in closed-control are discussed. Finally, a review of the current performance and new proposed measures are provided before closing this Chapter.

2.1 Monitoring Depth of Hypnosis

To fully control anesthesia, a measurement of DOA[1] is required. To this date, no such index has been developed. Recent studies have shown that the δ wave in the EEG signal correlates well with depth of hypnosis. Bispectral, wavelet, time domain, frequency domain, and evoke potential analysis are a few examples of techniques applied to the raw EEG signal to extract a single index from it.

The most commonly used metric uses the bispectral analysis and is appropriately called the Bispectral Index (BIS). A more recent approach uses the wavelet analysis and is called Wavelet-based Anesthetic Value (WAV). In the next two Subsections, a brief overview of each method is provided.

2.1.1 Bispectral Index

The BIS monitor was first introduced in 1994 by the Aspect Medical Systems, Inc and was marketed as a "novel measure" of level of consciousness from the EEG signal [50]. The monitor provides a single index that measures the DOH in the scale of 0 (iso-electric EEG) to 100 (fully awake). The BIS monitor was the first FDA approved monitoring system [8].

BIS is statistically based and empirically derived. A large group of volunteers' EEG were collected and using a proprietary statistical methodology, a model was fitted to the data. Since the dynamics of the system are unknown, it is difficult to design an optimal controller with this output signal. Moreover, the BIS Monitor has a time delay between the changes in the patient's anesthetic state and the changes in

[1] DOA is a measure of both the hypnotic and the analgesic states. DOH on the other hand which is a measure of the depth of hypnosis only.

the BIS value [8]. These two problems together are the motivation to have another monitor whose dynamics are known, and whose response has no delay.

2.1.2 Wavelet-based Anesthetic Value

The Wavelet-based Anesthetic Value (WAV) is an alternative to BIS that is based on the wavelet decomposition of the raw EEG signal. Proposed by Bibian et. al [11], this hypnotic monitoring value correlates well with the BIS. The dynamics of the system are described by a simple transfer function $1/(8s^2+1)$ and responds much faster to the changes in anesthetic state than the BIS. There is minimal to no delay in the signal response [69]. These two advantages of the WAV index make it appealing to be used as a control signal. It has been shown to lead to an improved performance in closed-loop control of anesthesia [10].

2.2 Drug Effect Modeling

The physiological effect of an administrated drug on a patient is typically described with two models: the pharmacokinetic and the pharmacodynamic model. The pharmacokinetic model (PK) relate the administrated drug dosage to the drug plasma concentration. The pharmacodynamic model (PD) then relate the drug plasma concentration to the physiological effect. These models describe the distribution, metabolism, and the clearance of the drug in the body to the resulting physiological effect.

In this literature review, an overview of the PK and the PD as described in [10] is introduced and briefly reviewed. A more detailed discussion can be found in [43] and [10].

2.2.1 Pharmacokinetics of Propofol

Pharmacokinetic model represents the drug uptake, distribution and elimination. The mathematical model then relates the infusion rate to the drug plasma concentration. The first significant investigation to study effect of sampling site (venous vs arterial) and the method of drug administration (bolus vs infusion) was conducted in 1998 by Schnider et al. [47]. A more recent study in 2000 was conducted by Schüttler and Ihmsen [48] and is discussed in this Section.

The Schüttler and Ihmsen study was a large collaboration between 5 institutes, where 4,112 samples from 270 individuals (150 men, 120 women) of the ages 2-88 years, body weights of 12-100 Kg were studied. The objectives of this study was:

1. Estimate the pharmacokinetics of propofol with respect to the covariates age, body weight, and gender.
2. Evaluate the inter- and intra-patient variability.
3. Study the effect of the mode of administration (bolus vs infusion).
4. Study the effect of the sampling site (venous vs arterial).

The result showed that the pharmacokinetics of propofol is best described by a 3-compartment model, see Figure A.1 in the Appendix A. Weight was determined to be the most prominent factor; age, gender and mode of administration were also positively correlated. The sample site had a little influence. The intra-patient variability in this study was found to be less than 20%.

The mathematics of the modeling can be found in Appendix A. The PK parameters can be found in Table A.3.

2.2.2 Pharmacodynamics of Propofol

Pharmacodynamic model is the observed effect of the drug as a function of the drug plasma concentration. A single drug interacts with multiple organs in the body and has multiple pharmacological effects. Here, the model for the depth of hypnosis from the EEG is considered.

There have been a limited number of studies on quantifying the effect of propofol on the EEG. A detailed discussion on these studies can be found in [10]. Many of the studies show large inter-patient variability in the PD model [10]. Moreover, most of these studies derive thePK model for the BIS.

In Bibian [10], the dynamics of propofol vs the WAV was modeled through the analysis of 44 patients. Using least-squares identification, Bibian estimated a PD model consisting of a Hill function followed by a first-order time delayed transfer function. The Hill function models the drug-receptor binding interaction. The first-order transfer function was proposed by Sheiner et al. [49] to model the

temporal aspect of the pharmacodynamics. The time delay was added to represent the arm-to-brain circulation time.

The details of the modeling can be found in Appendix A. The *PD* parameters can be found in Table A.3.

2.3 Automatic Control of Anesthesia

During surgery, the anesthetic and opioid titration are constantly adjusted to prevent under- and over-shoot of the drug plasma concentration and to keep the anesthetic state constant. An automated system that regulates the administration of these drugs thus seems appealing to the anesthesiologists.

The idea of closed-loop control of anesthesia has been investigated for half a century now. The performance and robustness of these controllers depend strongly on the mathematical model (PKPD) of the patients, the monitoring devices (BIS or WAV) as well as the tuning of the controller itself. The ideal controller should measure the three functional states hypnosis, analgesic and paralysis and regulate the administration of hypnotic, opioid and neuromuscular drugs. This Multi-Input/Multi-Output (MIMO) system is currently not available due to the limitation of monitoring systems as well as the mathematical models that govern the drug administration.

In recent years, the number of published studies on this field has increased significantly. A literature review on the current attempts on closed-loop control of hypnosis is provided next.

2.3.1 Closed-Loop Control: A Review

In the following reviews, adequate anesthesia is considered as the BIS or WAV in the range of 40-60 [39].

In 1999, Frei et al. [21] used a Model-Predictive-Controller (MPC) to control the Mean Arterial Pressure (MAP) using the inhaled drug, isoflurane. The study was performed on over 100 subjects and proved a better performance than manual control. The authors initially designed a PID-like Fuzzy controller. However, the controller was unable to account for respiratory dynamics under low flow conditions. The MPC model was implemented due to this inadequacy.

In 2001 Struys et al. [56] compared the performance of an adaptive model-based control guided by the patient's BIS to manually control anesthesia using intravenous administration propofol. The study was conducted on 20 female subjects aged 34-50 years undergoing gynecologic laparotomy. Subjects were randomized with half under closed-loop control and the other half were manually controlled. The study found that the manually controlled patient had a shorter induction time. The closed-loop controlled patients had a better maintenance performance as validated by Varvel measures, as well as a reduced recovery time. No details were provided on the controller structure.

In 2002, Absalom et al. [2] used a PID controller guided by the patient's BIS using intravenous administration of propofol in 10 patients undergoing elective hip or knee surgery. Performance was validated using Varvel measures. The authors reported clinically adequate anesthesia in 9 out of the 10 patients. Three of the patients' BIS oscillated around the set-point, although none of these cases showed a sign of inadequate anesthesia. The controller used was from another study by Kenny et al. [31] where a PID controller was guided by the auditory evoked potential.

In a follow up study by Absalom et al. [1] in 2003, a revised PID controller was used. In this study, 20 adult patients (12 female, 8 males) undergoing body surface surgery were enrolled. The patients were initially controlled with an open-loop target-controlled-infusion. Once the anesthesia was clinically adequate, the system was switched to the revised PID. All 20 patients reported a clinically adequate anesthesia. There was one patient with oscillation.

A more interesting study in 2004 by Locher et al. [42] used a cascade structure with an outer Proportional-Integral (PI) and inner model-based state feedback controller guided by the patient's BIS using isoflurane. The study was performed on 23 patients undergoing decompressive spinal surgery who were randomized into closed-loop or manual control. The study had two conclusions: 1) the closed-loop control significantly outperformed the manual mode and, 2) the closed-loop control administrated less total drug and faster wake-up time.

In 2006 study by Liu et al. [39], 164 patients undergoing elective minor or major surgery were randomized into closed-loop and manual target control infusion groups. The closed-loop system was an empirically tuned PID controller. The

patient's BIS was used as the control signal and propofol and remifentanil were administrated intravenously. Propofol consumption was lower in the closed-loop group, but the induction time was longer. Adequate anesthesia was significantly better in the closed-loop group. Recovery time was also shorter for the closed-loop group.

Finally, $\mathcal{L}_1$- output feedback adaptive control was used in 2011 by Ralph et al. [46] in a simulation study using the BIS as the control signal and isoflurane as the hypnotic drug . Seven PKPD models were reconstructed using clinical trial data. A controller was designed based on one of the identified PKPD models. The same controller was then applied to the other six models. The result showed adequate reference tracking.

2.3.2 Closed-Loop Control: Oscillation

Oscillation in closed-loop control can occur as a combination of any of the following: 1) marginally stable control loops (due to aggressive control tuning or changes in process gain/phase/time delay); 2) external disturbances; 3) stiction in control valve [9]. If the controller is improperly tuned, the oscillation can cause instability. In Chapter 4, a detailed root cause analysis of oscillation is provided.

In [2], [1] and recently in our own work [57], oscillation in the patient's BIS and WAV was detected. Therefore, it is essential that oscillation be detected in real-time to both warn the anesthesiologist and to remove it by retuning the controller.

One of the first attempts at oscillation detection was by Hägglund [25] in 1995. His method computed the Integrated Absolute Error (IAE) between consecutive zero-crossings of the error. When oscillation occurs, the absolute error and the time between consecutive zero-crossings increase, leading to a higher IAE. By counting the instances of IAE larger than a threshold in a given period of time, oscillation can be detected. This method, however, can fail to detect oscillations when multiple frequencies exist. Moreover, it cannot determine all the different oscillation frequencies in a signal.

Wang et. al. [67] review a large set of different algorithms. Auto-correlation function, Discrete Wavelet Transform (DWT) method, empirical mode decomposition, and Discrete Fourier Transform (DFT) are to just name a few methods applied

since 1995. Many have limitations. DFT has the disadvantage that the default rectangular window only provides good energy compaction for frequencies that are whole fractions of the sampling frequency F_s. DWT has the disadvantage that it can be computationally expensive. Wang et al. provide a new method based on the Discrete Cosine Transform (DCT) that overcomes all the shortcomings. It is fast, independent of the sampling frequency, and can decompose oscillation into all of its period components.

An improved representation of Wang's DCT method is provided in Chapter 4. The period, magnitude and fitness of the dominant oscillation is detected in real-time. In Chapter 5, these information are used to re-tune the PID controller.

2.3.3 Closed-Loop Control: Adaptive vs PID

A closed-loop control system can be divided into two generic types: adaptive and non-adaptive (classical). An adaptive controller is a system whose parameters can *adapt* continuously to the plant it is trying to control [28]. This adaptation can be in response to initial uncertainty in the plant or the change in the plant itself (for instance, an aircraft loses weight due to fuel consumption). The non-adaptive (classical) controller is a system in which the controller is not changed once it is implemented. Adaptive controllers in theory can provide better performance and robustness as they adapt to the particular plant. From the reviews in 2.3.1, adaptive control is still not well understood for use in closed-loop anesthesia.

In this paper, the newly introduced $\mathcal{L}_1$ Adaptive Control ($\mathcal{L}_1$-AC) is reviewed [26]. The Proportional-Integral-Derivative (PID) controller, which accounts for about 90% of all the controllers used in the industry is also considered [5]. Our research group currently uses a PID closed-loop control system for controlling the DOH (see [10] and [52]). In Chapter 5, a tuning algorithm is introduced to automatically re-tune the controller in the presence of an oscillation.

2.4 Performance of Closed-Loop Anesthesia

A set of four performance measures (MDAPE, MDPE, Divergence, and Wobble), proposed by Varvel et al. [1], have constituted the standard means of assessing performance in closed-loop anesthesia. Varvel measures were developed for Target-

Controlled Infusion (TCI) anesthesia systems; they were not developed for EEG-guided closed-loop controllers. These measures are not accepted within the control community and cannot be used as control tuning parameters. Moreover, they only account for the maintenance phase of anesthesia. Varvel measures are based on the *median* of the relative error. There is no distinction between artifacts, noise, and momentary large errors. There is also no penalty for outliers when adopted for EEG-guided DOH control and the metrics are not normalized with respect to duration of the case.

Soltesz et al. [53] proposed an alternate set of measures. The key features of these measures are : 1) wide acceptance in control community; 2) consideration of clinical feasibility; 3) separation of metrics for induction, maintenance and emergence phases of anesthesia. For the *induction phase*, Induction Phase Duration (ID) and Percent Overshoot (OS) are proposed. For the *maintenance phase*, Integrated Error (IE), Integrated Absolute Error (IAE), Variability Index (VI) and percentage of time outside the adequate range are proposed. For the *emergence phase*, Emergence Phase Rise Time (ER) is proposed. The mathematical details of Varvel and the proposed measures can be found in Appendix B.

We analyzed 63 clinical cases that were collected from a study on closed-loop control DOH using the NeuroSense monitor [57]. The study was approved by UBC Childrens and Womens Research Ethic Board (H10-01174), Vancouver, Canada [61]. The population included 32 women, 31 men between the ages of 6-17 years old, body weight of 14.5 - 70 Kg, and height of 106 - 182 cm. The propose measures provided more insight about the control performance, as discussed in Appendix B.

There are certain scenarios where Varvel measures can be misleading. DOH values in the set-point ± 10 range are considered adequate. Maintenance phases like the one in Figure 2.1 should be more desirable than the one in Figure 2.2; using the error metric on the median (Varvel) has the opposite effect.

The IE punishes outliers linearly while the median-based error metric Median Performance Error (MDPE) filters out outliers. The DOH in Figure 2.3 is clearly more negatively biased than in Figure 2.4. The MDPE metric concludes the opposite, while IE reflects this bias. Furthermore, IE is used as minimization criterion in existing controller synthesis strategies.

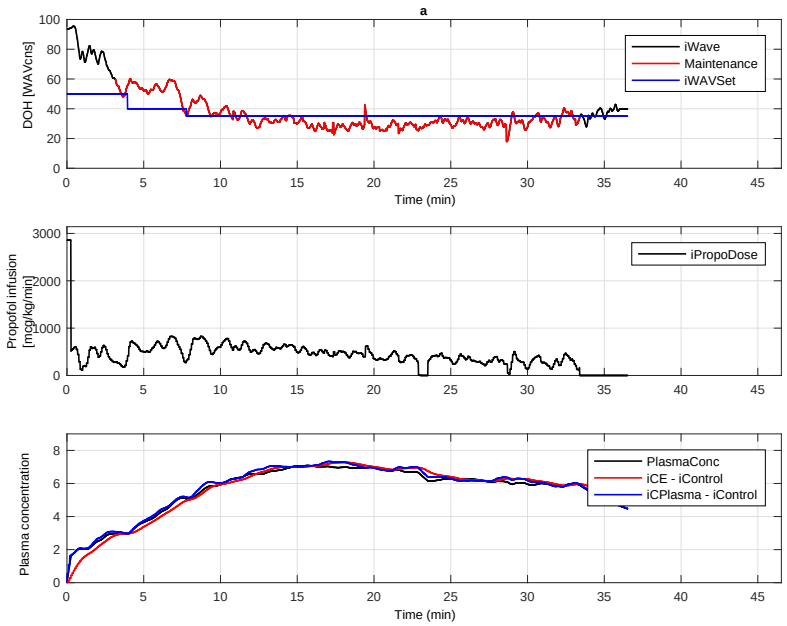


Figure 2.1: MDAPE vs IAE for a systematic small error.

Varvel metrics do not provide any measures for the induction phase. Length of ID affects the initial performance of the maintenance phase. The long ID results in a large total initial drug dosage, and an excessive overshoot of the DOH. Short ID results in low plasma concentration and signals the possibility of rapid rising in DOH. This information is available in the proposed measure as seen in Figure 2.5 and 2.6.

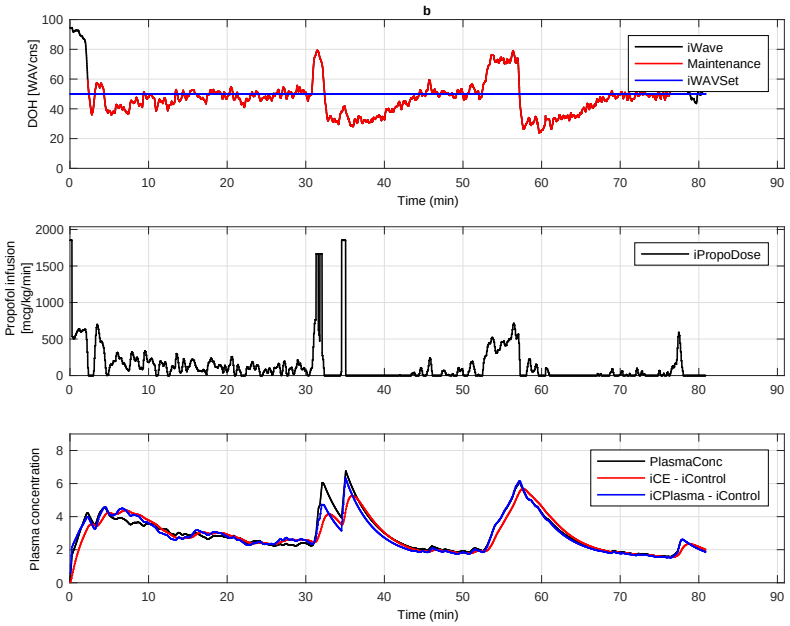


Figure 2.2: MDAPE vs IAE for a sporadic error.

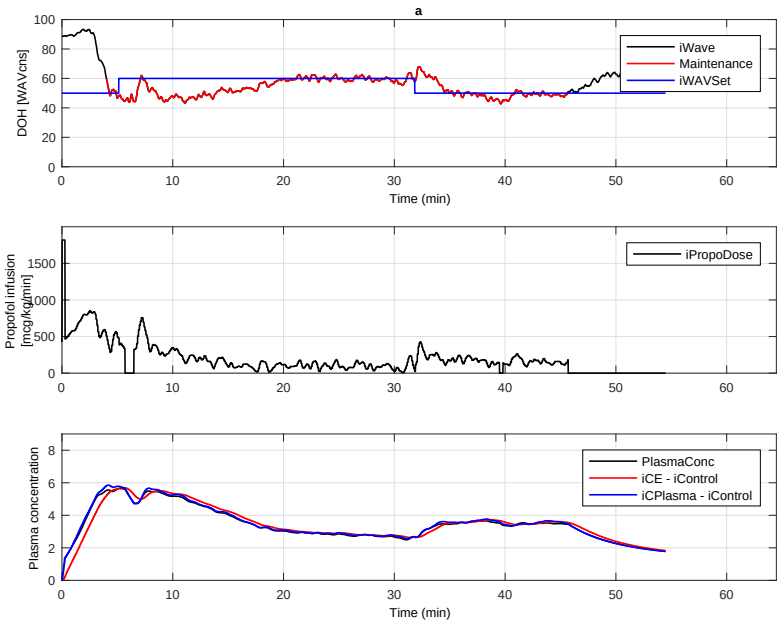


Figure 2.3: DOH is clearly negatively biased.

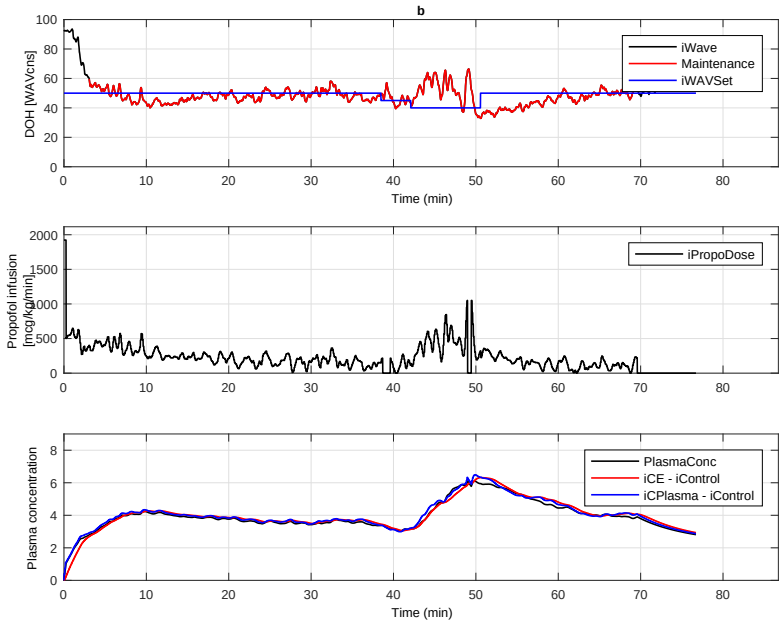


Figure 2.4: DOH is less biased.

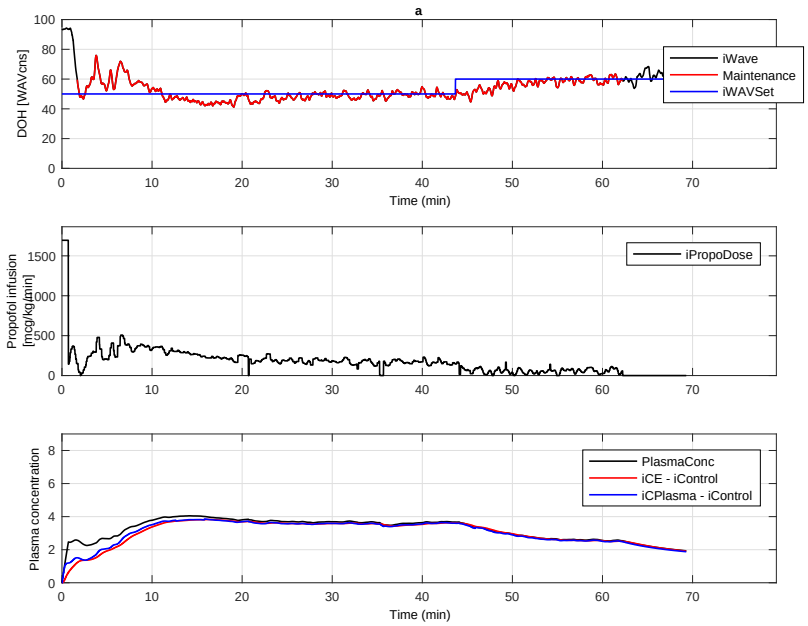


Figure 2.5: Small ID of 3.1 min translates to a small overshoot of 12.8%.

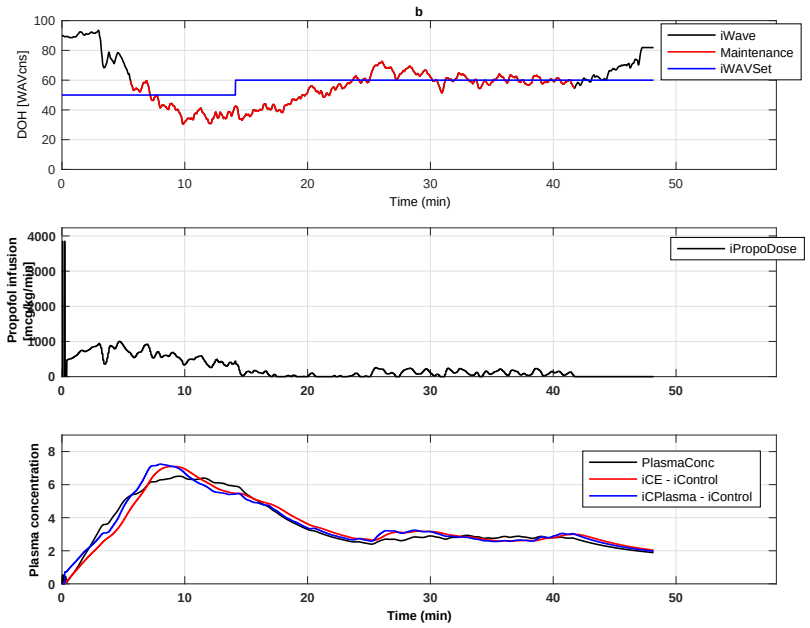


Figure 2.6: The large ID of 5.6 min translates to a larger overshoot of 48.9% and a longer DOH settling time to set-point.

Chapter 3

$\mathcal{L}_1$ Adaptive Control

There is a high inter-patient variability in the effect of the hypnotic drug on their DOH. A closed-loop system that controls the drug administration needs to guarantee robustness and performance. The rate at which an adaptive controller adapts to the patient is called the adaptation gain, Γ. It is a well known fact that high-gain in the feedback loop of a controller leads to amplified high frequency components in the control signal, reduction in phase margin, and loss of robustness [28]. Numerous authors have tried to introduce the concept of fast adaptivity with robustness as core to the control design (see for instance [28] and [35]) as classical robustness concepts are not applicable.

In an adaptive controller, Γ shows up in the *adaptation law*: a nonlinear dynamic system that identifies a known parameter related to the uncertainty of the plant. As the adaptation gain increases, the rate at which the unknown parameter is identified, also increases.

Classically, there has been a trade-off between robustness and performance: as one increases, the other decreases. In adaptive control the same trade-off exists: increasing the adaptive gain will improve the performance (by increasing the adaptivity) at the cost of reducing the robustness of the system.

This Chapter discusses $\mathcal{L}_1$ adaptive control as introduced in the book "$\mathcal{L}_1$ *Adaptive Control Theory: Guaranteed Robustness with Fast Adaptation"* [26]. The authors suggest that through their unique control structure, the adaptivity is *decoupled* from the robustness, i.e. one can increase the adaptive gain to arbitrarily large

values without effecting robustness. Robustness is then guaranteed through classical methodology. The $\mathcal{L}_1$-AC defines an *unimplementable* reference model and guarantees the difference between this reference model's output and the patient model's output decreases as the adaptation gain increases.

At the start of this research, there had been some doubts on the validity and claims of the theory [29]. In [45] it is shown that high-gain leads to system instability. In [13], the $\mathcal{L}_1$-AC shows inferior performance as compared to other well established adaptive controllers. It has also been seen in simulation that the adaptivity is lost as the gain increases [63]. Recently, the authors of $\mathcal{L}_1$-AC have proposed four different adaptation formulations which all lead to the exact same performance bounds [64]. One of these formulations is in fact LTI for all adaptation gains. All of these research however, have not proved or disproved the $\mathcal{L}_1$-AC; they have shown examples where the stability and the adpativity are lost.

The structure of this chapter is as follows: in Section 3.2, the $\mathcal{L}_1$ Adaptive Control structure is introduced. The reference system and performance bounds are discussed in Section 3.3. Simulation examples are provided in Section 3.4. Loss of adaptivity is discussed in Section 3.5.

3.1 Contribution

This chapter will review the claims made about $\mathcal{L}_1$-AC. First, it is shown that increasing the gain results in a loss of adaptivity. Second, it is shown that the limiting behavior (the case with Γ going to infinity) of the $\mathcal{L}_1$-AC can be achieved through an implementable, *non-adaptive* LTI controller. Finally, the loss of adaptivity is mathematically shown to be the direct result of inversion of the estimation loop as the gain increases. An example at the end of the chapter shows how a series of adaptive, non-adaptive, dynamic, static, linear and nonlinear laws that all lead to the exact same limiting controller as the adaptation gain increases.

3.2 The $\mathcal{L}_1$ Adaptive Control

3.2.1 Problem Formulation

Consider the following state-feedback dynamic controller $G(s)$ within the $\mathcal{L}_1$-AC architecture (see Chapter 2.2 of [26]):

$$\begin{aligned} \dot{x}(t) &= A_m x(t) + b(\omega(t)u(t) + \theta^T(t)x(t) + \sigma(t)), \quad x(0) = x_0, \\ y(t) &= c^T x(t), \end{aligned} \tag{3.1}$$

where $x(t) \in \mathbb{R}^n$ is the measured state of the system; $u(t) \in \mathbb{R}$ is the control input; $y(t) \in \mathbb{R}$ is the output; $b, c \in \mathbb{R}^n$ are assumed known constant vectors; A_m is a $n \times n$ Hurwitz matrix corresponding to the desired closed-loop dynamics; $\omega \in \mathbb{R}$ is an unknown constant but with known sign; $\theta^T(t) \in \mathbb{R}^n$ is a vector of unknown parameters; and $\sigma(t) \in \mathbb{R}$ models input disturbances. The dynamics of the desired model $M(s)$ are given by:

$$\begin{aligned} \dot{x}_m(t) &= A_m x_m(t) - k_g b r(t), \quad x_m(0) = x_0, \\ y_m(t) &= c^T x_m(t), \end{aligned} \tag{3.2}$$

where $k_g \triangleq -1/(c^T A_m^{-1} b)$ and $r(t)$ is the reference signal.

Assumption 1. Boundedness of the unknown parameters: Let the unknown parameters $\theta(t)$ and $\sigma(t)$ be bounded as:

$$\theta(t) \in \Theta, \qquad |\sigma(t)| \leq \Sigma,$$

where Θ and Σ are both known bounds of $\theta(t)$ and $\sigma(t)$ respectively. Furthermore, let the lower and upper bound of $\omega(t)$ be known:

$$\omega_{lb} \leq \omega(t) \leq \omega_{ub}, \qquad \forall t \geq 0.$$

These bounds need to be chosen from prior knowledge of the inter-variability in the patients' models and the expected input disturbances.

3.2.2 State Predictor

The state predictor in the $\mathcal{L}_1$-AC is given by:

$$\begin{aligned} \dot{\hat{x}}(t) &= A_m\hat{x}(t) + b(\hat{\omega}(t)u(t) + \hat{\theta}^T(t)x(t) + \hat{\sigma}(t)), \quad \hat{x}(0) = x_0 \\ \hat{y}(t) &= c^T\hat{x}(t). \end{aligned} \tag{3.3}$$

The predictor has the same structure as 3.1; the unknown parameters $\omega(t), \theta(t)$, and $\sigma(t)$, are replaced by their estimates $\hat{\omega}(t), \hat{\theta}(t)$, and $\hat{\sigma}(t)$.

The adaptation laws for the three unknown parameters are given by the following projection operator [34]:

$$\begin{aligned} \dot{\hat{\theta}}(t) &= -\Gamma \cdot \text{Proj}\big(\hat{\theta}(t), -\tilde{x}^T Pbx(t)\big), \quad & \hat{\theta}(t) &= \theta_0, \\ \dot{\hat{\sigma}}(t) &= -\Gamma \cdot \text{Proj}\big(\hat{\sigma}(t), -\tilde{x}^T Pb\big), & \hat{\sigma}(t) &= \sigma_0, \\ \dot{\hat{\omega}}(t) &= -\Gamma \cdot \text{Proj}\big(\hat{\omega}(t), -\tilde{x}^T Pbu(t)\big), & \hat{\omega}(t) &= \omega_0, \end{aligned} \tag{3.4}$$

where $\tilde{x}(t) = \hat{x}(t) - x(t)$, $\Gamma \in \mathbb{R}^+$ is the adaptation gain, and $P = P^T > 0$ is the solution of the algebraic Lyapunov equation $A_m^T P + PA_m = -Q$ for arbitrary $Q = Q^T > 0$.

Finally, the $\mathcal{L}_1$-AC signal is defined as:

$$u(s) = -kD(s)(\hat{\eta}(s) - k_g r(s)), \tag{3.5}$$

where $r(s)$ and $\hat{\eta}(s)$ are the Laplace transforms of $r(t)$ and $\hat{\eta}(t)$ respectively and

$$\hat{\eta}(t) \triangleq \hat{\omega}(t)u(t) + \hat{\theta}^T(t)x(t) + \hat{\sigma}(t). \tag{3.6}$$

$k > 0$ is a feedback gain and $D(s)$ is a strictly proper transfer function such that they lead to a strictly proper stable filter $C(s)$:

$$C(s) = \frac{\omega kD(s)}{1 + \omega kD(s)}. \tag{3.7}$$

The controller is shown in Figure 3.1.

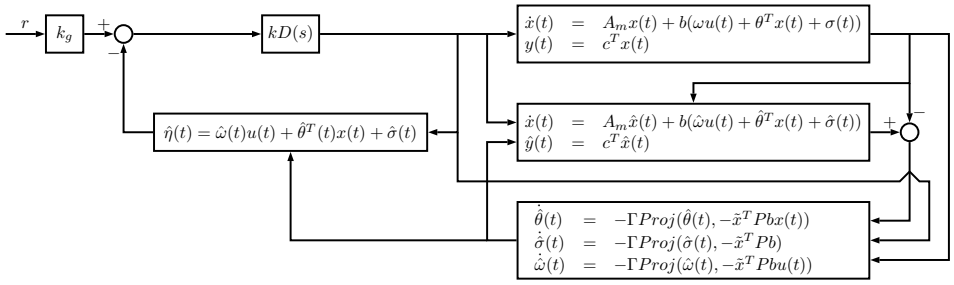


Figure 3.1: The original $\mathcal{L}_1$-AC block diagram as it appears in [26].

3.2.3 $\mathcal{L}_1$-norm Stability Condition

The $\mathcal{L}_1$-AC is subject to the following $\mathcal{L}_1$-norm condition:

$$\|L(s)\|_{\mathcal{L}_1} T < 1 \tag{3.8}$$

where $L(s)$ and T are computed as:

$$T \triangleq \max_{\theta \in \Theta} \|\theta\|, \quad H(s) = (s\mathbb{I} - A_m)^{-1} b, \quad L(s) = H(s)(1 - C(s)). \tag{3.9}$$

If the condition 3.8 in presence of Assumption 1 is satisfied, then the $\mathcal{L}_1$-AC is guaranteed to be stable. In calculating the $\mathcal{L}_1$-norm, $C(s)$ depends on the unknown parameter ω, which should be chosen as the worst expected case.

The claim for the $\mathcal{L}_1$-AC is as follows: compute a gain k and a filter $D(s)$ such that for the worst case ω, the $\mathcal{L}_1$-norm stability condition holds. This will guarantee the robustness of the system. Then increase Γ as high as computationally possible to increase the performance. The filter $kD(s)$ will act as the decoupler of robustness and performance trade-off.

3.3 Achievable Performance Bound

The controller cannot achieve the desired system dynamics $M(s)$ as a direct result of the introduction of the low-pass filter $kD(s)$ in the control loop. Instead, a reference system $G_{ref}(s)$ is introduced and the control performance of the system $G(s)$ is compared to the performance of the reference system $G_{ref}(s)$. The reference

system is defined as:

$$\begin{aligned}
\dot{x}_{ref}(t) &= A_m x_{ref}(t) + b(\omega(t)u_{ref}(t) + \theta^T(t)x_{ref}(t) + \sigma(t)),\\
y_{ref}(t) &= c^T x_{ref}(t),\\
u_{ref}(s) &= \frac{C(s)}{\omega}\Big(k_g r(s) - \eta_{ref}(s)\Big),
\end{aligned} \tag{3.10}$$

where $\eta_{ref}(s)$ is the Laplace transform of $\eta_{ref}(t) \triangleq \theta^T(t)x_{ref}(t) + \sigma(t)$. This controller is not implementable as it depends on the system unknowns $\omega(t)$, $\theta(t)$ and $\sigma(t)$.

3.3.1 Reference Controller

Assume an initial condition $x_{ref}(0) = x_0 = 0$. The reference system can be written as:

$$x_{ref} = H(s)(\omega u_{ref} + \eta_{ref}), \tag{3.11}$$

where the Laplace operator s is intentionally excluded from the signals to simplify the calculation. The above equation cannot be solved for $\eta_{ref}(s)$ since $H(s)$ is not invertible. Multiplying the equation above by $(Pb)^T$ makes the $(Pb)^T H(s)$ invertible and $\eta_{ref}(s)$ can be solved for:

$$\eta_{ref} = \frac{(Pb)^T x_{ref}}{(Pb)^T H(s)} - \omega u_{ref}. \tag{3.12}$$

Substituting η_{ref} from 3.12 into 3.10 leads to:

$$u_{ref} = \frac{C(s)}{\omega}\Big(k_g r - \frac{(Pb)^T x_{ref}}{(Pb)^T H(s)} + \omega u_{ref}\Big). \tag{3.13}$$

Isolating for u_{ref} results in:

$$u_{ref} = \frac{C(s)}{\omega(1-C(s))}\Big(k_g r - \frac{(Pb)^T x_{ref}}{(Pb)^T H(s)}\Big). \tag{3.14}$$

Taking into account the definition of $C(s)$ from 3.7, the equation above can be simplified to:

$$u_{ref}(s) = \frac{kD(s)}{(Pb)^T H(s)} (Pb)^T \Big(H(s) k_g r(s) - x_{ref}(s) \Big). \tag{3.15}$$

Even though the reference system 3.10 is not implementable, this control signal *is implementable* since it does not depend on the system's unknown parameters ($\theta(s)$, $\sigma(s)$ and $\omega(s)$). The control signal corresponds to an implementable LTI controller whose dynamics only depends on the filter $kD(s)$, the desired model dynamics $H(s)$ and the solution to the Lyapunov equation, P. Still, $\mathcal{L}_1$-AC claims to achieve this non-adaptive behavior as the $\Gamma \to \infty$ in an adaptive structure.

3.3.2 Control Performance

The $\mathcal{L}_1$-AC structure guarantees the following bounds:

Lemma 1 (From [26])**.** Let the system $G(s)$ be controlled by the $\mathcal{L}_1$ adaptive controller from Section 3.2. Assume the $\mathcal{L}_1$-norm stability condition of 3.8 is satisfied and the bounds of Assumption 1 are met. Assume the reference system 3.10 is stable, i.e. the system $G_{ref}(s)$ is stabilized through the LTI reference controller $\eta_{ref}(s)$ from equation 3.15. Then, the system state $x(t)$ and control input $u(t)$ are uniformly bounded:

$$\|x_{ref} - x\|_{\mathcal{L}_\infty} \leqslant \frac{\gamma_1}{\sqrt{\Gamma}}, \qquad \|u_{ref} - u\|_{\mathcal{L}_\infty} \leqslant \frac{\gamma_2}{\sqrt{\Gamma}}, \tag{3.16}$$

where γ_1 and γ_2 are constants. The full details of the calculation is provided in Chapter 2.2, pages 40-41 of the book [26].

The details of the calculation of the Lemma is not important. Rather, the inverse relationship of $x_{ref} - x$ and $u_{ref} - u$ to the adaptation gain Γ is significant. These performance criteria motivate the use of high adaptation gain.

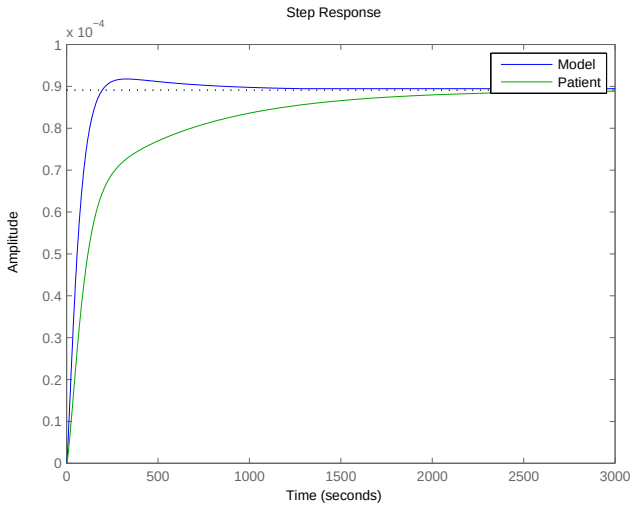


Figure 3.2: The step response of patient #7 and the predictor model

3.4 Case Studies of the $\mathcal{L}_1$ Controller

In this Section, the PKPD models from Appendix A are used for two case studies. Since the PK model depends on the demographic information of the patient only, the predictor's PK model is chosen as the patient's PK. The PD is chosen as the average of all 44 cases, with the delay set to zero. The predictor's model PKPD is also slightly modified to have a faster response than the patient's. While this predictor model is not ideal, it suffices for the purpose of showing the limitation of the $\mathcal{L}_1$-AC. Figure 3.2 shows the step response of the predictor model and a patient (case #7).

Let $D(s) = \frac{1}{10s^2+s}$ and $k = 1$ so that the filter $C(s)$ is given as:

$$C(s) = \frac{\omega}{10s^2 + s + \omega}. \tag{3.17}$$

This satisfies the $\mathcal{L}_1$-norm condition of 3.8: the value of $\|L(s)\|_{\mathcal{L}_1} T$ is between 0.0168 and 0.0673 for all 44 models. The bounds of the unknown parameters are $\Theta = 1$, $\Sigma = 1$ and $-5 \leq \omega \leq 5$ and are chosen as the maximum of the 44 PKPD models.

The system is simulated for 3 different adaptation gains 1, 5×10^3 and 1×10^5. The results of the simulation are shown in Figure 3.3. The patient's output for the three different gains is shown in the upper plot, while the absolute error between the patient's output and the reference's output is shown in the lower plot.

For an adaptive algorithm, it is expected that the error between the reference and the patient decreases throughout the case. For the small gain of $\Gamma = 1$ the result clearly shows a decrease in error as the case progresses. However, for the higher gain, specially $\Gamma = 1 \times 10^5$, while the initial error is lower, it does not improve as the case progresses, i.e. the adaptivity of the system is lost. An almost identical behavior is shown for another case (patient model #2) as shown in Figure 3.4. In this case, even for the intermediate gain $\Gamma = 5 \times 10^3$ the adaptivity is almost lost as the error does not decrease as time continues.

The claim of achieving the *non-implementable* reference model in an adaptive architecture is invalid, since adaptivity is lost. The exact same performance can be achieved using the *implementable* reference controller defined in 3.15. This controller is LTI, does not depend on any unknown parameters, and provides the same performance.

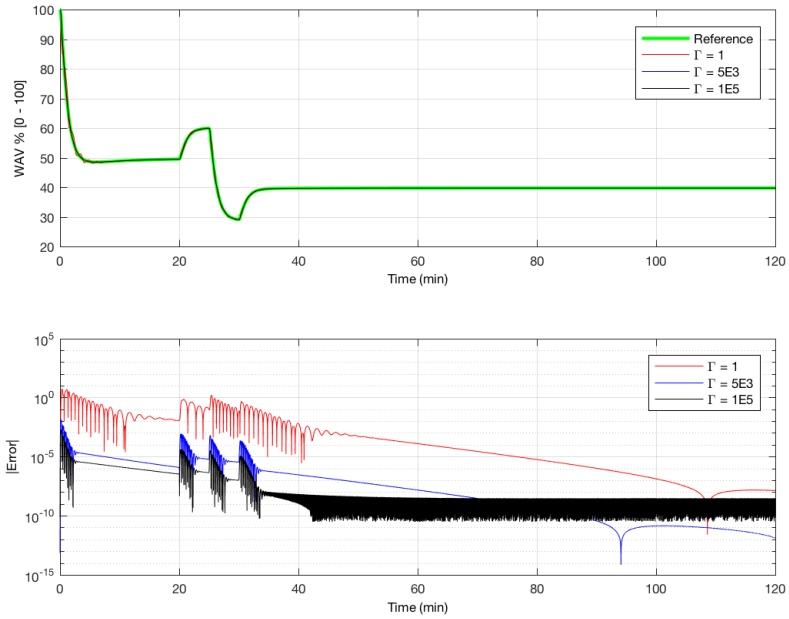


Figure 3.3: Simulated system output for the patient #7 controlled by the $\mathcal{L}_1$ controller. The upper plot shows the patient's output for the three adaptation gains. The reference output is shown in the thick green line. The lower plot shows the absolute error of patient's output to the reference's output.

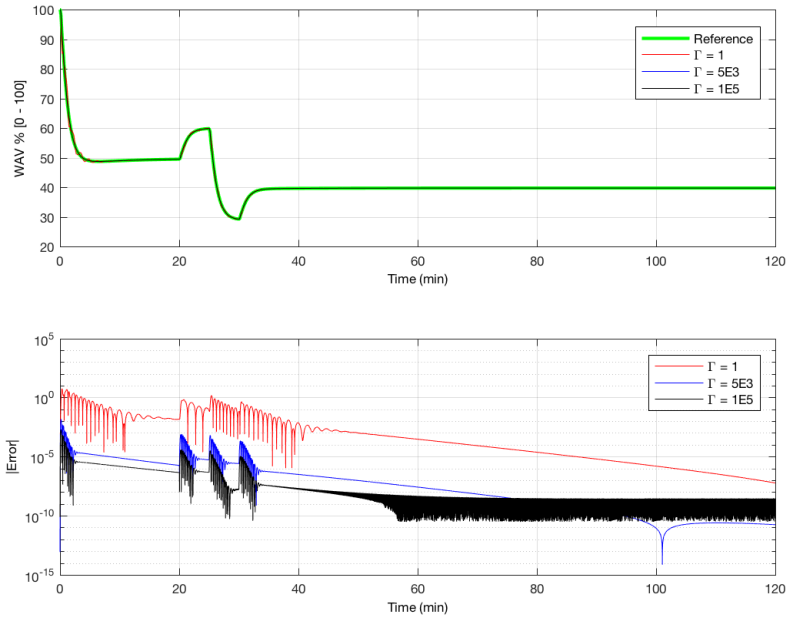


Figure 3.4: Simulated system output for the patient #2 controlled by the $\mathcal{L}_1$ controller. The upper plot shows the patient's output for the three adaptation gains. The reference output is shown in the thick green line. The lower plot shows the absolute error of patient's output to the reference's output.

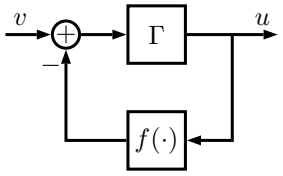


Figure 3.5: Simple feedback for a nonlinear function $f(\cdot)$

3.5 Loss of Adaptivity

The previous section provided two case study examples that showed the loss of adaptivity of the $\mathcal{L}_1$-AC as the gain increased. This is in-line with the previous claims that this controller does not achieve a better performance than an implantable LTI controller (see [13], [60] and [29]). The loss of this adaptivity is due to the high adaptation gain, and will be discussed in this Section [63].

In a simple feedback gain system, shown in Figure 3.5, it is easy to show that increasing the gain Γ leads to the inversion of the nonlinearity dynamic $f(\cdot)$ [23]. Straight forward calculation shows that $u = \Gamma(v - f(\cdot)u)$. Solving for u gives:

$$u = \frac{\Gamma}{1 + f(\cdot)\Gamma} v. \tag{3.18}$$

When the gain Γ increases to infinity, the system dynamic are inverted, i.e.

$$u_{lim} = \lim_{\Gamma \to \infty} \frac{\Gamma}{1 + f(\cdot)\Gamma} v = \frac{\Gamma}{f(\cdot)\Gamma} v = \frac{1}{f(\cdot)} v = f(v)^{-1}. \tag{3.19}$$

In the $\mathcal{L}_1$ controller Figure 3.1, there is a similar high-gain feedback over the predictor's nonlinearity and it is expected that increasing the gain will also invert the predictor, albeit the loop over the signal $\eta(s)$ makes this observation challenging. However, the $\mathcal{L}_1$ architecture is also inverted as the gain goes to infinity. The details of this calculation is available in Appendix C for reference.

Figure 3.6 shows the linearized system around some equilibrium points θ_Q^T, ω_Q^T, and σ_Q^T. Here, $G(s)$ and $H(s)$ refer to the dynamics of the patient and the predictor model. F_x, $F_{\hat{x}}$ and F_u are some LTI functions related to the linearized components of the adaptation laws as shown in C.14.

After linearizing the projection adaptive laws of 3.4, the transfer function be-

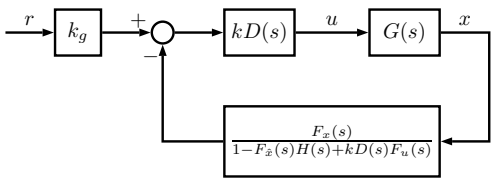


Figure 3.6: Final form of the linearized adaptation laws for $\mathcal{L}_1$ adaptive control with generic adaptation laws.

tween u and x can then be written as:

$$u(s) = -\frac{kD(s)\left[\frac{\Gamma}{s}(x_Q^T x_Q + u_Q^2 + 1)(Pb)^T + \hat{\theta}_Q^T\right]}{1 + \frac{\Gamma}{s}(x_Q^T x_Q + u_Q^2 + 1)(Pb)^T H(s) + kD(s)\omega_Q} x(s), \tag{3.20}$$

where x_Q and u_Q are the equilibrium values of the signal x and u. Here, it is assumed that at equilibrium the states x and $\hat{x}$ are the same (i.e. $\tilde{x} = \hat{x} - x = 0$). The phrase $(x_Q^T x_Q + u_Q^2 + 1)$ is just a constant. The effect of increasing the gain is now abundantly clear - the limiting controller is:

$$\begin{aligned} u_{lim}(s) &= \lim_{\Gamma \to \infty} -\frac{kD(s)\left[\frac{\Gamma}{s}(x_Q^T x_Q + u_Q^2 + 1)(Pb)^T + \hat{\theta}_Q^T\right]}{1 + \frac{\Gamma}{s}(x_Q^T x_Q + u_Q^2 + 1)(Pb)^T H(s) + kD(s)\omega_Q} x(s) \\ &= \frac{-kD(s)(Pb)^T}{(Pb)^T H(s)} x(s). \end{aligned} \tag{3.21}$$

This *limiting* controller has the exact same control signal as the *reference* signal 3.15 that was derived from the reference system. Note that unlike the reference signal, this result is not derived from the mathematical description of the plant $G(s)$ and holds true for any kind of plant $G(s)$, given the stability condition is met.

This limiting controller derivation indicates that for high adaptation gains:

- The integral effect in the adaptation law is canceled.
- The predictor is inverted.
- The effect of nonlinearity of the controller is canceled.
- The choice of the equilibrium point x_Q, $\hat{x}_Q$ and u_Q does not affect the controller. Moreover, the exact formulation of the adaptive laws is irrelevant.

In a follow up paper by the authors of the $\mathcal{L}_1$-AC, it was shown that four different adaptations laws resulted in similar performance bounds [64]. Above result explains why this is so. In fact, the result above suggests that *any* law, adaptive or non-adaptive, linear or nonlinear, dynamic or static, will result in the same performance bounds, provided that the stability condition and bounds are satisfied. An example will now follow: Six laws are shown to provide the exact same performance bound as the gain increases. The example used is the one used by the authors' themselves in [15] and [14].

3.5.1 Simple Example of Loss of Adaptivity

Consider the system 3.1 with $A_m = \begin{bmatrix} 0 & 1 \\ -1 & -1.4 \end{bmatrix}$, $b = \begin{bmatrix} 0.5 \\ 1 \end{bmatrix}$, $c = \begin{bmatrix} 1 & 0 \end{bmatrix}$, $\omega = 1$, $\theta = \begin{bmatrix} 2 \\ 2 \end{bmatrix}$ and $\sigma = 1$. This corresponds to the system $G(s) = \frac{0.5s+1.7}{s^2-1.6s-1.4}$ and the predictor $(Pb)^T H(s)$:

$$\begin{aligned} (Pb)^T H(s) &= \frac{N(s)}{D(s)} \\ N(s) &= 1.6s^7 + 8.2s^6 + 20.5s^5 + 31.4s^4 + 31.4s^3 \\ &\quad + 20.5s^2 + 8.2s + 1.6, \\ D(s) &= s^8 + 5.6s^7 + 15.8s^6 + 27.8s^5 + 33.4s^4 \\ &\quad + 27.8s^3 + 15.8s^2 + 5.6s + 1, \end{aligned} \tag{3.22}$$

where P solves the Lyapunov equation with $Q = \mathbb{I}_{2\times 2}$. With $k = 60$ and $D(s) = \frac{1}{s(1+0.1s)}$, the filter $C(s)$ is defined as:

$$C(s) = \frac{60}{0.1s^2 + s + 60}. \tag{3.23}$$

The $\mathcal{L}_1$-norm condition with the choice of this filter is 0.0858. The following 6 adaptation laws are considered:

A) The dynamic $\mathcal{L}_1$ projection adaptation law:

$$\begin{aligned}
\dot{\hat{\theta}}(t) &= -\Gamma \cdot \mathrm{Proj}\big(\hat{\theta}(t), -\tilde{x}^T P b x(t)\big) \\
\dot{\hat{\sigma}}(t) &= -\Gamma \cdot \mathrm{Proj}\big(\hat{\sigma}(t), -\tilde{x}^T P b\big) \\
\dot{\hat{\omega}}(t) &= -\Gamma \cdot \mathrm{Proj}\big(\hat{\omega}(t), -\tilde{x}^T P b u(t)\big) \\
\hat{\eta}(t) &= \hat{\omega}(t)u(t) + \hat{\theta}^T(t)x(t) + \hat{\sigma}(t).
\end{aligned}$$

B) Another dynamic nonlinear adaptation law:

$$\begin{aligned}
\dot{\hat{\theta}}(t) &= -\Gamma \cdot \tilde{x}^T P b \|x(t)\| \\
\dot{\hat{\sigma}}(t) &= -\Gamma \cdot \tilde{x}^T P b \\
\dot{\hat{\omega}}(t) &= -\Gamma \cdot \tilde{x}^T P b |u(t)| \\
\hat{\eta}(t) &= \hat{\omega}(t)|u(t)| + \hat{\theta}^T(t)x\|(t)\| + \hat{\sigma}(t).
\end{aligned}$$

C) The static $\mathcal{L}_1$ projection adaptation law:

$$\begin{aligned}
\hat{\theta}(t) &= -\Gamma \cdot \mathrm{Proj}\big(\hat{\theta}(t), -\tilde{x}^T P b x(t)\big) \\
\hat{\sigma}(t) &= -\Gamma \cdot \mathrm{Proj}\big(\hat{\sigma}(t), -\tilde{x}^T P b\big) \\
\hat{\omega}(t) &= -\Gamma \cdot \mathrm{Proj}\big(\hat{\omega}(t), -\tilde{x}^T P b u(t)\big) \\
\hat{\eta}(t) &= \hat{\omega}(t)u(t) + \hat{\theta}^T(t)x(t) + \hat{\sigma}(t).
\end{aligned}$$

D) A higher order nonlinear static adaptation law:

$$\hat{\eta}(t) = -\Gamma \cdot [\hat{x}^3(t) - x^3(t)]^T P b.$$

E) A linear adaptation law as used in [64]:

$$\hat{\eta}(t) = -\Gamma \cdot \tilde{x}^T P b.$$

F) A switching adaptation law as used in [33]:

$$\begin{aligned}
\hat{\theta}(t) &= -\Delta_\theta \cdot sgn\Big(dz_{\varepsilon_\theta}\big(\hat{x}(t)-x(t)\big)^T \cdot Pbx(t)\Big) \\
\hat{\sigma}(t) &= -\Delta_\sigma \cdot sgn\Big(dz_{\varepsilon_\sigma}\big(\hat{x}(t)-x(t)\big)^T \cdot Pb\Big) \\
\hat{\omega}(t) &= -\Delta_\omega \cdot sgn\Big(dz_{\varepsilon_\omega}\big(\hat{x}(t)-x(t)\big)^T \cdot Pbu(t)\Big) \\
\hat{\eta}(t) &= \omega(t)u(t) + \hat{\theta}^T(t)x(t) + \hat{\sigma}(t),
\end{aligned}$$

where $sgn(\cdot)$ is the sign function; $dz(\cdot)$ is the dead-zone function; $\varepsilon_\theta \in \mathbb{R}^+$, $\varepsilon_\sigma \in \mathbb{R}^+$, and $\varepsilon_\omega \in \mathbb{R}^+$ are the dead-zone intervals; Δ_θ, Δ_σ, and Δ_ω are the bounds of the unknown parameters from Assumption 1.

Laws A), E), and F) are 3 of the 4 laws which the authors of the $\mathcal{L}_1$-AC have themselves introduced as alternatives and have shown that they achieve the same performance bounds [64] (the forth law is only applicable for a different class of $\mathcal{L}_1$ controllers). Also, only Laws A) and B) are dynamic and adaptive; the other 4 laws are static and non-adaptive. For the Laws A) to E), the gains used are $\Gamma = 1, 1e3, 1e4$. For Law F), the dead-zone intervals used are $dz = 1, 0.1, 0.01$. Figures 3.7 to 3.12 show the simulation results for the 6 laws.

For Laws A) and B), the error between the reference output and the plant output decreases as the case progresses for low adaptation gain, as expected of an adaptive controller. However, as the gain increases, while the initial error is lower, it does not improve over time, suggesting that the system has lost its adaptivity.

In Law C) the integral action of the projection law is removed and the adaptation law is static. None of the gains result in an adaptive controller, yet the limiting case is identical to the original projection law. This law shows that a static, non-adaptive "adaptation law" provides the same plant output.

Law D) is an unnecessary and computationally heavy law that is not practical for any application, and is only intended for demonstration purposes. For low adaptation gain, the system has sustained oscillation. However, for higher gains, again the system approximates the LTI reference model and has the same output as the original adaptive projection law.

Law E) is a very simple error feedback, non-adaptive law that was suggested as an alternative solution by the authors of the $\mathcal{L}_1$-AC [64]. For low adaptation gain,

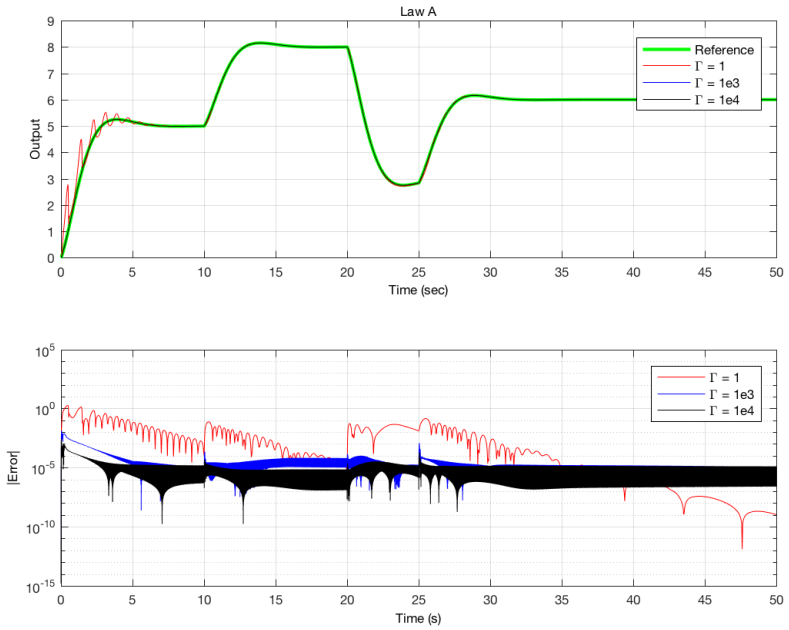


Figure 3.7: Simulated output for the plant $G(s)$ controlled by the $\mathcal{L}_1$-AC for Law A). The top figure is the output of the plant, with the thick green line being the output of the reference. The lower plot is the absolute difference of model output and reference output. The controller is adaptive for low adaptation gain, however it becomes static for higher gains.

the system has a steady-state error; since there is no integral action or adaptivity, the error does not reduce as the case progresses. The steady-state error reduces as the adaptation gain increases.

Law F) is another law suggested by the authors of $\mathcal{L}_1$-AC [33]. This law is the most computationally exhaustive algorithm and has no improvements over the other laws. It is again non-adaptive and the plant's model approximates the reference model as the gain increases.

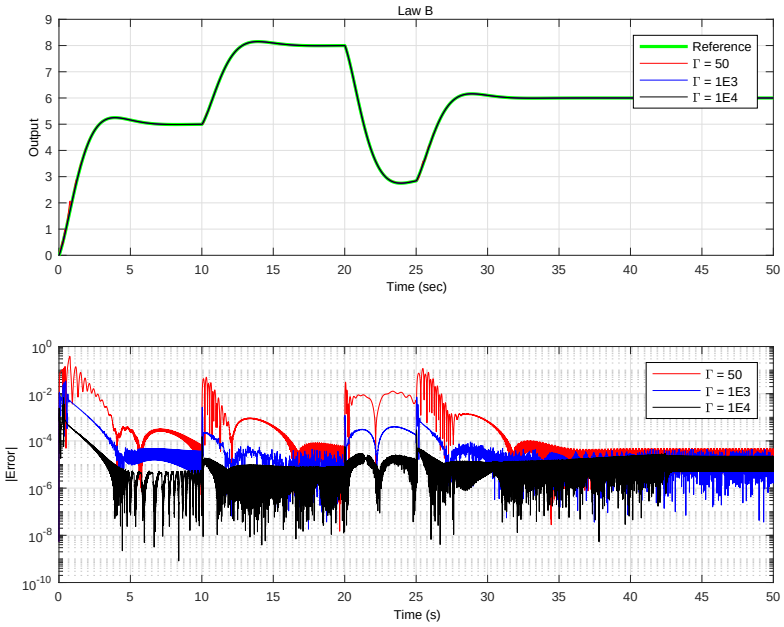


Figure 3.8: Simulated output for the plant $G(s)$ controlled by the $\mathcal{L}_1$-AC for Law B). The top figure is the output of the plant, with the thick green line being the output of the reference. The lower plot is the absolute difference of model output and reference output. The controller is adaptive for low adaptation gain, however it becomes static for higher gains.

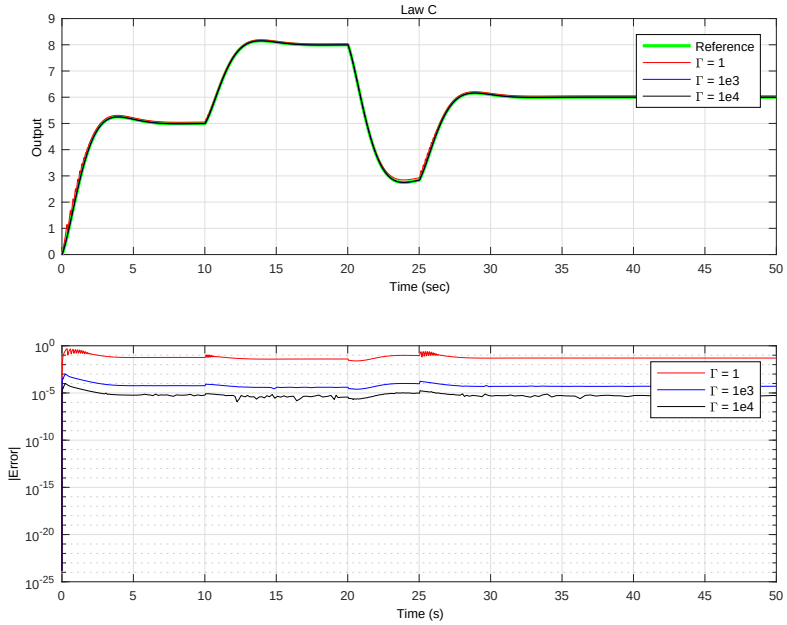


Figure 3.9: Simulated output for the plant $G(s)$ controlled by the $\mathcal{L}_1$-AC for Law C). The top figure is the output of the plant, with the thick green line being the output of the reference. The lower plot is the absolute difference of model output and reference output. The controller is non-adaptive for all gains.

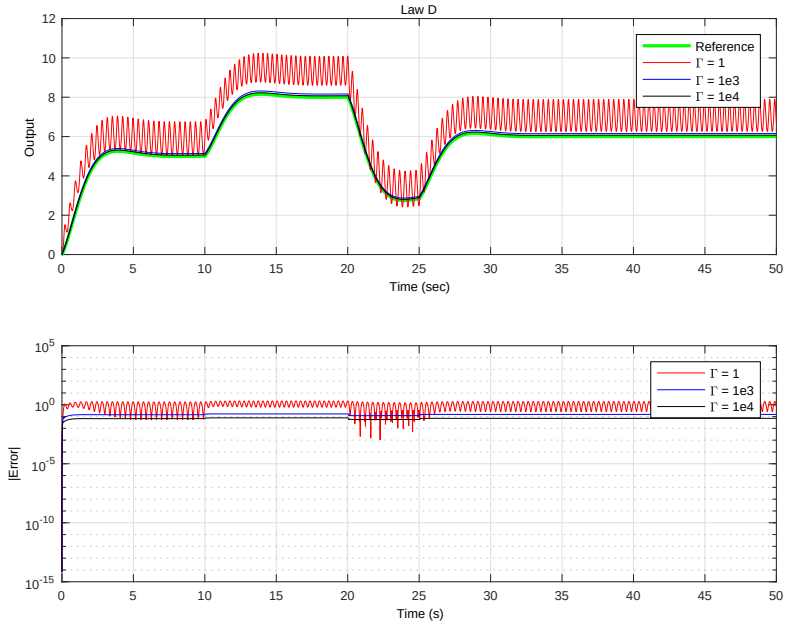


Figure 3.10: Simulated output for the plant $G(s)$ controlled by the $\mathcal{L}_1$-AC for Law D). The top figure is the output of the plant, with the thick green line being the output of the reference. The lower plot is the absolute difference of model output and reference output. The controller is nonadaptive for all gains.

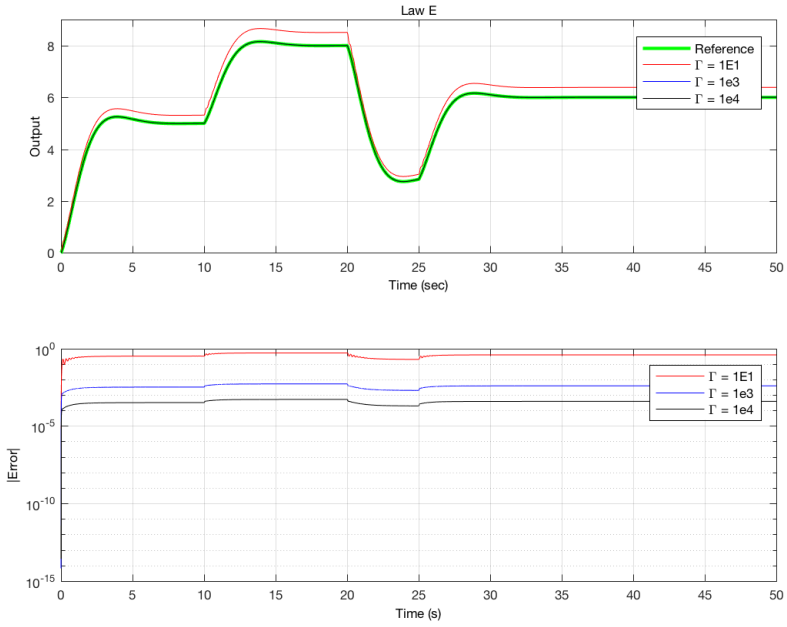


Figure 3.11: Simulated output for the plant $G(s)$ controlled by the $\mathcal{L}_1$-AC for Law E). The top figure is the output of the plant, with the thick green line being the output of the reference. The lower plot is the absolute difference of model output and reference output. The controller is nonadaptive for all gains.

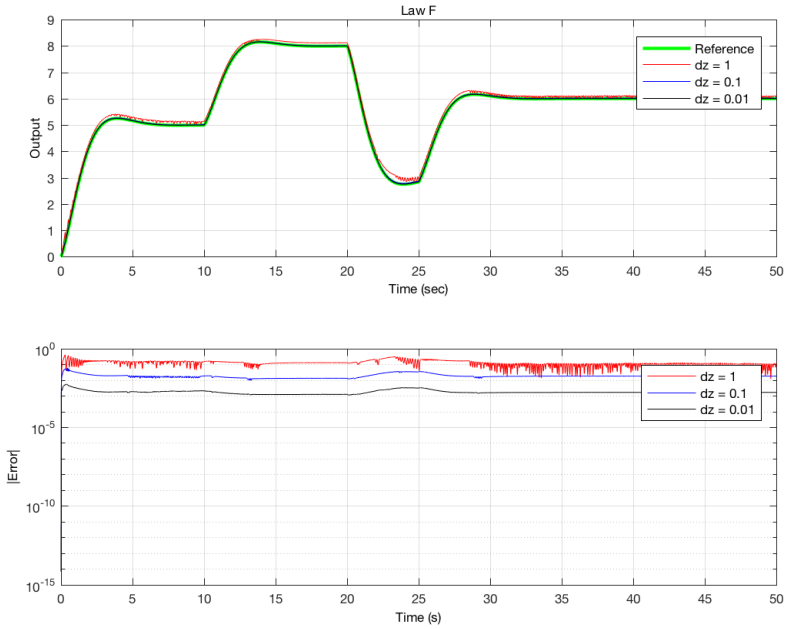


Figure 3.12: Simulated output for the plant $G(s)$ controlled by the $\mathcal{L}_1$-AC for Law F). The top figure is the output of the plant, with the thick green line being the output of the reference. The lower plot is the absolute difference of model output and reference output. The controller is nonadaptive for all dead-zone intervals.

3.6 Conclusion

The $\mathcal{L}_1$-AC claims fast adaptation while maintaining the robustness. The fast adaptation is achieved through the use of high-gain feedback while robustness is achieved through the use of a low-pass filter that filters out any noise amplification caused by the high-gain feedback.

In Section 3.4, it was shown that a PKPD model's output approximates a reference system with an *implementable* LTI controller as the gain increases. It was also shown that the system is adaptive for low gains, but non-adaptive for higher gains. In Section 3.5 the loss of adaptivity was mathematically shown. The feedback gain shows up in the loop transfer function, which when taken to infinity, causes the inversion of the predictor model. This inversion of the predictor's model is a well known concept in classical control theory. This Chapter then showed a simple example for 6 different adaptation laws, some of which were adaptive for low gains and some of which were non-adaptive for all gains. The output of all cases approximates the implantable LTI controller of the reference system. This rules out the use of those $\mathcal{L}_1$-AC schemes to address the problem of patient variability in closed-loop control of anesthesia.

Chapter 4

Real-Time Oscillation Detection

Oscillation is a common problem in control-loop systems. There are three types of oscillations: damped oscillation, undamped sustained oscillation, unstable oscillation. Unstable oscillation will lead to increased deviation from the set-point and can compromise safety and stability. While damped oscillation and the sustained oscillation will not cause instability, they will lead to a lower quality control signal.

Other than safety and stability concerns, oscillation will also result in a higher control action [58]. This translates into a higher drug dose to the patient, which can lead to post-surgical complications and unnecessary increase in cost.

There are several factors that can cause oscillation. A few are:

- Marginally stable control loops (due to aggressive control tuning or changes in process gain/phase/time delay).
- External disturbances.
- Dead-band, also known as hysteresis, of the controller valve (hardware/equipment issues).
- Stiction in the control valve.
- Hitting the upper and/or lower bound limit of the controller valve.

Studying and analyzing the oscillation can be performed in three stages:

1. Identifying sustained oscillations, and where possible, detecting the different frequency components of the oscillation(s).

2. Detecting and quantifying the root-cause of the oscillation(s).

3. Correcting the root-cause of the oscillation(s).

It would be beneficiary for the detection to be performed in real-time. A flag can be raised to alert the anesthesiologist. If the problem is due to poor-tuning, an auto-tuning method (see Chapter 5) can re-tune the controller and remove the oscillation. If the problem is due to a mechanical failure, the system can be taken into manual mode to prevent the escalation of the problem.

There have been numerous techniques to address the problem of detecting oscillation in a signal that contains multiple oscillation frequencies. In more-or-less chronological order, these methods include the integrated of absolute error [25], the auto-correlation function methods [32], the spectral peaks-based method [32], the wavelet-based method [44], the modified empirical mode decomposition method [55], the DCT-based method [67] and many more. Among all, the detection method based on the DCT proposed by Wang et. al. [67] is one of the most advanced methods. It can detect multiple oscillations in off-line and real-time and it can determine the frequency, magnitude and fitness (percent energy of the oscillation) of these components.

In this Chapter, an algorithm for detecting oscillations is discussed. In Section 4.2 an off-line method based on the DCT analysis is introduced. In Section 4.3, this methodology is extended to real-time. In Section 4.4, examples are provided.

4.1 Contribution

This Chapter discusses a new method for determining the multiple oscillations in a control signal, based on the method proposed by Wang et. al. [67]. First, the off-line algorithm is developed. The extension to real-time is discussed subsequently. The algorithm is able to also determine the dominant oscillation signal, characterized by its frequency, magnitude and fitness.

4.2 The DCT Off-line Oscillation Detection

Traditionally, the *Fourier Transform* and its discrete algorithm, DFT has been applied to frequency-related problems. Therefore, it may seem natural to pick the DFT rather than the DCT, which is related to the complex portion of the DFT signal. The main advantage of DCT is its strong "energy compaction" property [4]: most of the signal information is concentrated in a few coefficients of the low-frequency components of the transformed signal and it approaches the Karhunen-Loéve transform (which optimally decorrelates the frequency components, but is extremely slow to compute).

More importantly, the default DFT rectangular windows only provides good energy compaction for frequencies that are whole fractions of the sampling frequency F_s, i.e. a DFT analysis of a signal with sampling frequency of 240Hz can effectively detect frequencies that are exact (or close to) multiples of 240/N (such as 120Hz, 80Hz, or 60Hz). Applying a non-rectangular DFT window (or a moving window), will produce less broadband leakage, but will be lossy near the window's edge. DCT addresses all the issues above, and is therefore used as the basis for oscillation detection.

Given a time series $x(t)$, its associated frequencies will be distributed separately in the signal's DCT counterpart $y(k)$. That is, the different frequency components of the signal $x(t)$ can be studied by observing different segments of $y(k)$. In the following Section, the *oscillation detection algorithm* is discussed. Noise is also considered in this discussion.

4.2.1 The DCT Definition

Given a time series discrete sequence $x(nT)\big|_{n=1}^{N}$, with sampling period T, its DCT counterpart is defined as:

$$y(k) = \omega(k) \sum_{n=1}^{N} x(n) cos\Big(\frac{\pi}{2N}(2n-1)(k-1)\Big), \quad k = 1,2,3,...N, \tag{4.1}$$

where

$$\omega(k) = \begin{cases} \frac{1}{\sqrt{N}} & k = 1, \\ \sqrt{\frac{2}{N}} & 2 \leq k \leq N. \end{cases} \tag{4.2}$$

Similarly, the Inverse Discrete Cosine Transform (IDCT) is defined as:

$$x_i(n) = \sum_{k=1}^{N} \omega(k) y(k) cos\Big(\frac{\pi}{2N}(2n-1)(k-1)\Big), \quad n = 1,2,3,...N. \tag{4.3}$$

The DCT signal $y(k)$ has a convenient inherent property: given a signal of $x(t) = sin(2\pi\omega t + \phi)$, $y(k)$ will always be of the form $y(k) = ...,0,...,\#,0,...$ where # stands for some non-zero value. In other words, for each frequency component in the signal $x(t)$, the counterpart $y(k)$ will start with a zero directly followed by a non-zero value (call this $0-\#$ pattern), then followed by some integer (could be zero or non-zero), and finally finished by a $\#-0$ pattern. This property of the signal will be used in the next section to extract the segment of $y(k)$ that corresponds to a specific frequency of the signal $x(t)$.

To visualize this DCT pattern, a signal with two frequency components is shown in Figure 4.1. The signal is shown on the top with its DCT shown at the bottom. The first 4 points follow the discussed pattern and contribute to one of the frequency components. The next 5 points contribute to the second frequency component. The reconstructed signals of these two frequency components are shown in Figure 4.2. The reconstruction can only provide information on the frequency of the signal, and not on the magnitude or the offset of it.

4.2.2 The DCT Algorithm

The segments in $y(k)$ that are within each of the pattern $0-\#$ and $\#-0$ contribute to the different frequencies in the signal $x(t)$. A signal contaminated by noise however, may have all of the $y(k)$'s component non-zero. The case for white noise is covered below and the case of colored noise is discussed in the subsequent Section.

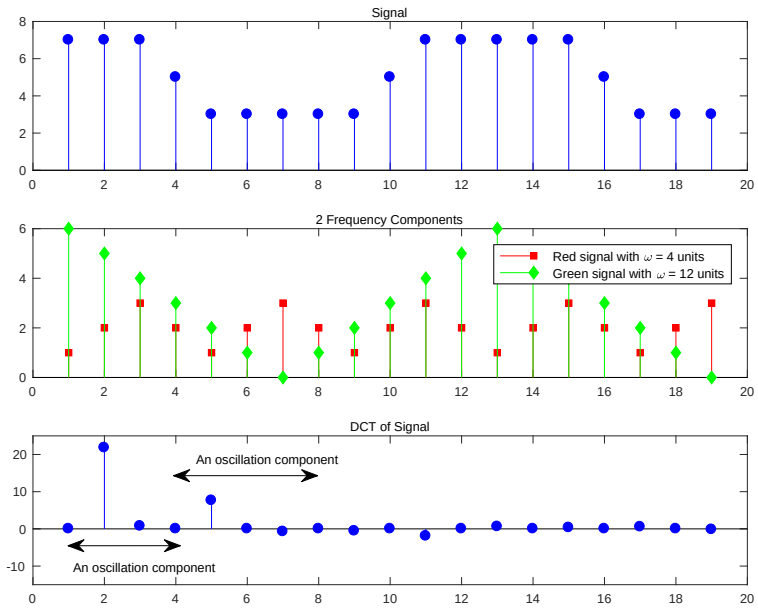


Figure 4.1: The signal blue has two components with frequencies 4 and 12 units respectively (labeled red and green signals). The DCT of the blue signal is shown on the bottom graph. The first 4 points on the DCT captures one of the frequency components, while the next 5 captures the other.

Define $\hat{\sigma}_y$ as the estimated standard deviation of the signal $y(k)$:

$$\hat{\sigma}_y = \sqrt{\frac{1}{N-1}\sum_{k=1}^{N}\left(y(k) - \frac{1}{N}\sum_{k=1}^{N} y(k)\right)^2}. \tag{4.4}$$

The white noise can be filtered by suppressing the values of $y(k)$ smaller than $3\hat{\sigma}_y$ and preserving the most significant components [67]:

$$y_h(k) = \begin{cases} y(k) & |y(k)| \geq HY, \\ 0 & |y(k)| < HY, \end{cases} \tag{4.5}$$

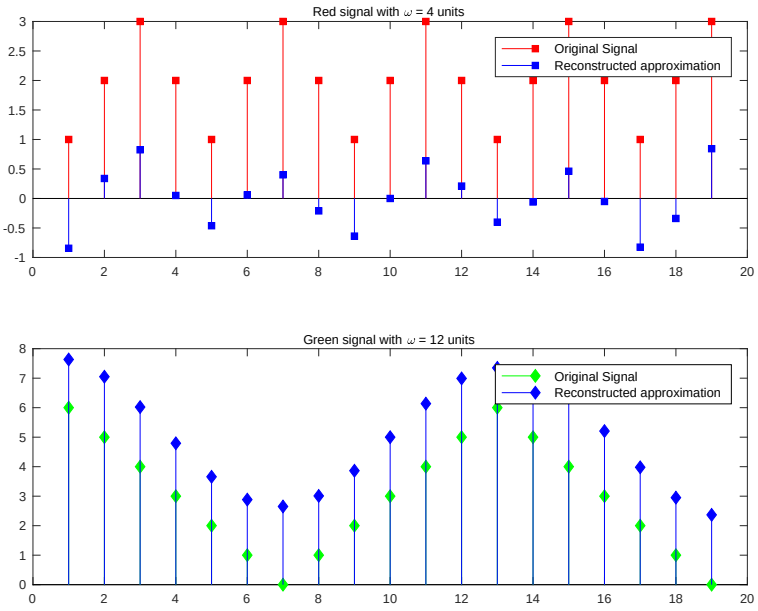


Figure 4.2: The reconstructed frequency components of Figure 4.1 is shown. The reconstruction can only preserve the frequency, but does not provide an accurate information on the magnitude and offset of the signal.

where HY is the high cut-off level:

$$HY = 3\hat{\sigma}_y. \tag{4.6}$$

Some noise may still be present in the signal y_h even after the filtering. Therefore, a segment is required to terminate with 4 consecutive zeros [67]. Define $y_i(k)$ as the i-th DCT component of the $y_f(k)$ of the same length:

$$y_i(k) = \begin{cases} y_{f_i}(k) & \text{for } k_{s,i} \leq k \leq k_{e,i}, \\ 0 & \text{otherwise}, \end{cases} \tag{4.7}$$

for $i = 1,2,3,...,I$, where $y_{f,i}$ is the i-th component of y_f with the *start* and *end*

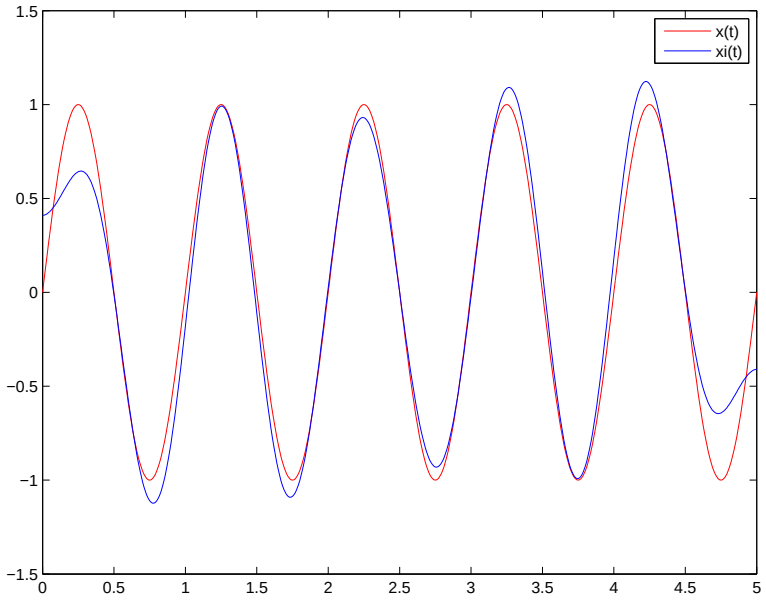


Figure 4.3: $x(t)$ and the frequency component $x_i(t)$

points $k_{s,i}$ and $k_{e,i}$ satisfying the conditions described below:

$$\begin{cases} y_f(k_{s,i}) \neq 0 \text{ and } y_f(k_{s,i}-1) = 0, \\ y_f(k_{e,i}) \neq 0 \text{ and } y_f(k_{e,i}-r) = 0 \text{ for } r = 1,2,3,4, \\ k_{s,i} \leq k_{e,i}. \end{cases} \tag{4.8}$$

The IDCT of each $y_i(k)$ will provide the specific frequency for the corresponding time-domain signal. Call this signal $x_i(t)$.

Example 4.2.1. Let $x(t) = sin(2\pi\omega t)$ with a discrete time $t = 0 : 0.01 : 5$, be a signal of size 501. The noise is due to discretization of the signal which causes the Gibbs phenomenon. The standard deviation of $y(k)$ is 0.7071 and the high cut-off level HY is 2.1213. Suppressing the values smaller than HY will yield the high component y_h that only has 6 non-zero values positioned at the indices 4, 6, 8, 10, 12, and 14. This yields only one i-th component, which will be the y_h itself. Figure 4.3 shows the plot of $x(t)$ v.s. $x_i(t)$.

4.2.2.1 Period and Regulatory Index

The zero-crossing $Z_i(l)$ of each x_i for $z = 1,2,3,...,L$, is evaluated to determine the period sequence $T_i(l)$:

$$T_i(b) = 2(Z_i(l) - Z_i(l-1)) \quad \text{for} \quad l = 2,3,4,...,L. \tag{4.9}$$

The sample mean and the standard deviation of the period is calculated using:

$$\bar{T}_i = \frac{1}{L-1}\sum_{l=2}^{L} T_i(l), \tag{4.10}$$

$$s_{T_i} = \sqrt{\frac{1}{L-2}\sum_{l=2}^{L}(T_i(l) - \bar{T}_i)^2}. \tag{4.11}$$

Define the *regulatory index* as the ratio of the sample mean and standard deviation of the signal T_i:

$$R_T = \frac{\bar{T}_i}{s_{T_i}} \tag{4.12}$$

The period signal T_i can be regular (oscillatory) or irregular (non-oscillatory, due to random arrivals). To be regular, R_T needs to be larger than 3 [59]:

$$R_T > 3 \tag{4.13}$$

To understand the rationale behind this inequality, consider the signal Z_i to be due to equally randomly distributed arrivals, i.e. a random exponential distribution:

$$f_{T_i} = \lambda e^{-\lambda \mu_T}. \tag{4.14}$$

For an exponential distribution, the mean and the standard deviation are equal, i.e. $\mu_{T_i} = \sigma_{T_i}$. A null hypothesis $H_0 : R_T = 1$ and the alternative hypothesis $H_1 : R_T > 3$ is formed. If the condition 4.13 holds, the H_0 is rejected and H_1 is accepted; T_i is then claimed to be regular and oscillatory.

The sample mean and standard deviation 4.10 and 4.11 cannot be reliably calculated with less than 4 sample sets. It is suggested to use at least 10 sample sets [59]. A *modified regulatory index* is now defined that also considers the number of

sample sets.

Define the population coefficient of variance as the ratio of population standard deviation and the mean:

$$C_v = \frac{\sigma_{T_i}}{\mu_{T_i}}, \tag{4.15}$$

and the sample coefficient of variance as:

$$\hat{C}_v = \frac{1}{R_T} = \frac{s_{T_i}}{\bar{T}_i}. \tag{4.16}$$

Let α be a small positive integer such that $(1-\alpha)100\%$ is the confidence interval for C_v:

$$\frac{\sqrt{L-1}\hat{C}_v}{\sqrt{\chi^2_{L-1,1-\alpha/2}}} < C_v < \frac{\sqrt{L-1}\hat{C}_v}{\sqrt{\chi^2_{L-1,\alpha/2}}} \tag{4.17}$$

where $\chi^2_{L-1,\alpha/2}$ is the $100\alpha/2$-th percentile of the chi-squared distribution with $L-1$ degree of freedom. The *modified regulatory index* is then given by the inverse of $\hat{C}_v$:

$$R_{T_i,\alpha} = \frac{\sqrt{\chi^2_{L-1,\alpha/2}}}{\sqrt{L-1}} \frac{\bar{T}_i}{s_{T_i}}, \tag{4.18}$$

and *the period regulatory test* is defined as the $R_{T_i,\alpha}$ that is larger than 3:

$$R_{T_i,\alpha} > 3. \tag{4.19}$$

Equation 4.19 forms the first periodic test: if no T_i passes the *period regulatory test*, then the signal $x(t)$ is concluded to be non-oscillatory.

The case for colored noise will now be discussed. If colored noise is present, the suppressed signal y_h from 4.5 will have too many of its coefficient removed and no longer resembles the noise. As a result, oscillation detection may give false results. Instead, a low cut-off value LY is defined as:

$$LY = \hat{\sigma}_y, \tag{4.20}$$

and the signal $y(k)$ is suppressed similar to y_h 4.5, but with LY, to give y_l. y_l is then segmented into its j-th DCT component, y_j, similar to 4.3. A pair of y_i and y_j

that have the same maximum value are matched. The *modified regulatory test* is performed on y_j whose pair y_i has passed the *period regulatory test* 4.19. If no y_j passes the test, the signal $x(t)$ is concluded to be non-oscillatory.

The definition of y_l preserves the colored noise. The *modified regulatory test* on y_l then filters out colored noise, while the test on y_h filters out white noise. The two cut-off values LY and HY are the two most important constants used in this derivation. Li et al. performed a series of simulations with different colored and white noises to determine their respective value of $LY = \hat{\sigma}_y$ and $HY = 3\hat{\sigma}_y$ [38].

4.2.2.2 Fitness Test

To measure the percentage energy of a component, or its *fitness*, the following equation from [41] and [67] is used:

$$F(x,x_k) = 100\big(1 - \frac{\| x_k - x \|}{\| x \|}\big), \tag{4.21}$$

where $\| x \|$ is the Euclidean norm. Of the (x_i, x_j) pair that have survived the *modified regulatory test*, the component that gives the largest *fitness*, contains the most energy in the signal and is therefore the dominant frequency. Since x_j contains more coefficients than x_i, it is used to determine the fitness of the dominant frequency of the signal x:

$$F_d = \max_j F(x,x_j). \tag{4.22}$$

If this fitness is larger than a predefined threshold, F_0, then the signal is concluded to be oscillatory. This is known as the *fitness test*:

$$F_d > F_0. \tag{4.23}$$

The dominant period is determined by comparing the *modified regulatory index* $R_{T,\alpha}$ of the x_{imax}-x_{jmax} pair that correspond to F_d:

$$\bar{T}_d = \begin{cases} \bar{T}_{imax} & R_{T_{imax},\alpha} \geq R_{T_{jmax},\alpha}, \\ \bar{T}_{jmax} & \text{otherwise}. \end{cases} \tag{4.24}$$

The signal x has a dominant period $\bar{T}_d$ with a fitness of F_d and the corresponding component x_{imax} and x_{jmax}.

4.2.2.3 Magnitude Test

The two tests *modified regulatory test* and the *fitness test* are considered adequate for most scenarios. However, sometimes the time series can pass both tests, but the magnitude of the oscillation may not be periodic; therefore a further test on the magnitude of the oscillation must be performed.

This test may seem unintuitive. An unstable oscillation (a sinusoidal that increases in magnitude) or a damped oscillation (a sinusoidal that decreases in magnitude) are both considered oscillatory, but have "irregular" magnitude. The irregular pattern is a signal that has sudden drops and peaks (for instance, due to miscommunication of the sensor) or one that has its magnitude follow an irregular pattern of high and low in no particular order. The magnitude test described below passes an unstable and damped oscillation and fails a true "irregular pattern". The paper Wang et al. [67] has an excellent example that illustrates this case.

Define the magnitude series $M(m)$ as:

$$\begin{aligned} A(m) &= \max\left(x(t)\big|_{1+(l-1)\bar{T}_d}^{1+l\bar{T}_d}\right) - \min\left(x(t)\big|_{1+(l-1)\bar{T}_d}^{1+l\bar{T}_d}\right), \\ M(m) &= A(m)/2, \end{aligned} \tag{4.25}$$

where $\bar{T}_d$ is the dominant period as determined by 4.24, $l = 1,2,3,...L$. In other words, scan the time series $x(t)$ in a window period of $\bar{T}_d$ and subtract the maximum from the minimum of the sequence.

Similar to 4.18, the *magnitude index* is defined as:

$$R_{M,\alpha} = \frac{\sqrt{\chi^2_{L-1,\alpha/2}}}{\sqrt{L-1}}\frac{\bar{M}}{s_M}, \tag{4.26}$$

where $\bar{M}$ and s_M are the sampled mean and standard deviation of the signal M respectively. The *magnitude regulatory test* is:

$$R_{M,\alpha} > 2.73, \tag{4.27}$$

where the threshold is determined as follows: the signal $M(m)$ is approximately half the size of $T(l)$. The ratio of $\frac{\sqrt{\chi^2_{L-1,\alpha/2}}}{\sqrt{L-1}}$ to $\frac{\sqrt{\chi^2_{L/2-1,\alpha/2}}}{\sqrt{L/2-1}}$ is approximately 1.1 for different values of α and L. The value 2.73 is $3/1.1$ [67].

4.2.3 Summary of the DCT Algorithm

There are 3 tests: *modified regulatory test* 4.19, *fitness test* 4.23, and *magnitude regulatory test* 4.27. The tuning parameters are α and F_0.

The algorithm is summarized in the following 13 steps (from here on called the *oscillation detection algorithm*):

Step 1. Remove the mean from the signal $x(t)$ and compute the DCT $y(k)$ from the definition 4.1.

Step 2. Suppress the elements of $y(k)$ that are smaller than the *high* cut-off value HY 4.6 to generate y_h as per 4.5.

Step 3. Compute i-th DCT components of y_h using 4.7 to get y_i for $i = 1,2,3,...,I$.

Step 4. Generate the inverse DCT $x_i(t)$ for each $y_i(t)$ using 4.3.

Step 5. Compute the period sequence $T_i(n)$ for each $x_i(t)$ and perform the *modified regulatory test*. If no signal passes the test, then $x(t)$ is concluded to be *non-oscillatory*.

Step 6. Suppress the elements of $y(k)$ that are smaller than the *low* cut-off value LY 4.20 to generate y_l.

Step 7. Compute j-th DCT components of y_l using 4.7 to get y_j for $j = 1,2,3,...,J$.

Step 8. Select the y_j that have the same maximum value as the y_i whose x_i passed the regulator test from step 5.

Step 9. Generate the inverse DCT $x_j(t)$ for each $y_j(t)$ that was selected form the previous step.

Step 10. Perform the same *modified regulatory test* as step 5. If none pass the test, then $x(t)$ is concluded to *not oscillatory*.

Step 11. Calculate F_d as the dominant of the fitness of x_{jmax} and perform the *fitness test* of 4.23. If the test fails, then $x(t)$ is concluded to be *non-oscillatory*.

Step 12. Determine the dominant period $\bar{T}_d$ using 4.24.

Step 13. Determine the magnitude sequence from $\bar{T}_d$ and perform the *magnitude test* of 4.27. If the test fails, then $x(t)$ is concluded to be *non-oscillatory*.

The oscillating signal is characterized by a dominant period $\bar{T}_d$, magnitude of $\bar{M}$ and fitness of F_d. This test has two tunable parameters: α and F_0. Suggested values are $\alpha = 2.7\%$ and $F_0 = 25\%$.

4.3 Extension to Real-Time

Oscillation can lead to instability and excessive actuator action. Detecting oscillation in real-time allows us to raise a flag and notify the anesthesiologist to quickly modify the setting and to stabilize the patient's DOH. In Chapter 5 an auto-tuning method is proposed that will re-tune the controller to remove the oscillation.

The basis for extending the *oscillation detection algorithm* from Section 4.2.2 is as follows: select an *adaptive* window range and perform the *oscillation detection algorithm*. This window range is dependent on the predicted dominant period and will change in real-time to adapt to the case.

In Thornhill et al (1997) [58], it was suggested to use a window range of 50 times of the presumed oscillation when applying the Hägglund method. This value was then disputed by Wang et al (2013) [67] when applied to the DCT method since their method specifically determines the oscillation period whereas the Thornhill method only detects large *IAE* errors and therefore requires a larger window range.

The window range best be large enough to produce sufficient sample sets to compute the sample mean and standard deviation of the period and magnitude sequences $T(l)$ and $M(m)$ accurately. A window range of 10 times the presumed period will produce a period sequence of 20 sample sets, and a magnitude sequence of 10 sample sets, allowing for an accurate measurement of the sample mean and standard deviation of $\bar{T}$ and $\bar{M}$ [67]. The starting and ending positions of the time

window are:

$$\begin{aligned} n_e &= t, \\ n_s &= n_e - 10T_p, \end{aligned} \tag{4.28}$$

where t is the current time.

The presumed period, T_p should be adaptive to allow the system to identify oscillation of any period. If a dominant period T_d is found after applying the *oscillation detection algorithm*, then it is set to be the presumed period T_p. If no oscillation is detected, the component pair x_i and x_j with the maximum F_{x_j} is selected. There are 3 scenarios:

Scenario 1. The pair x_i and x_j only contain one zero crossing. In this case, no period can be determined, and so the previous presumed period is kept.

Scenario 2. The pair x_i and x_j contain exactly 2 zero crossing. In this case, the *modified regulatory index* cannot be calculated and so the method of 4.24 cannot be applied. Since x_i is contaminated less by noise, then $\bar{T}_{x_i}$ is selected as the presumed period.

Scenario 3. There are enough zero-crossing points to perform the *modified regulatory test*. In this case, the method of 4.24 is used.

In Wang et al (2013), it is suggested to allow the presumed period to change freely. This can cause an issue: assume the system is initially oscillating with a small period T_s. The oscillation then stops and at a later time an oscillation with much higher period $T_h > 10T_s$ is formed. The presumed frequency of the algorithm is now stuck at T_s and the system only scans a period of $10T_s$ and may never be able to capture this new oscillation.

Instead, it is suggested to create multiple parallel instances of the algorithm to run simultaneously. Each instance has a predefined minimum and maximum allowed period, $[T_{lower}, T_{upper}]$ and T_p is allowed to freely adapt in this period range. The range of each instance and the number of these instances will be the tuning parameter and is related to the problem at hand. This approach also allows us to ignore certain periods that are expected to exists in the system, for instance a known background noise that might be present in the signal.

4.3.1 Summary of the Real-Time DCT Algorithm

The real-time algorithm can now be summarized in the following 5 steps. The following steps should be executed for the number of instances that have been selected to run.

Step 1. Specify an initial presumed oscillation to create the starting and ending positions of the time window 4.28. Here T_p would be a priori knowledge of the presumed oscillation. If this information is not available, then the bandwidth of the system can be used.

Step 2. Wait until sufficient time has passed and there are sufficient data to perform the *oscillation detection algorithm*.

Step 3. Perform the off-line *oscillation detection algorithm* on the segment of $x(n_s)$ to $x(n_e)$.

Step 4. Update the T_p according to the 3 scenarios 4.3 and the $[T_{lower}, T_{upper}]$ limits of the instance.

Step 5. Repeat Step 2-4 for all the instances. You may need to wait for more data if the new T_p is larger than the old one.

4.4 Oscillation Detection Examples

Two examples are provided to highlight the *oscillation detection algorithm*. The first example will be a simplified simulation example. The second example will be the Depth of Hypnosis from a surgical case performed by iControl system.

Example 4.4.1. Consider the following signal with an Signal to Noise Ratio (SNR) value of 10^{-1} shown in Figure 4.4:

$$x(t) = \sin\frac{2\pi}{1.3}t + 2\sin\frac{2\pi}{3.4}t$$

The off-line algorithm on the system determines two oscillation periods of 1.3 min and 3.397 min. Based on the *fitness* of the two signals, the component with period of 3.397 min is chosen as the dominant table. The magnitude of this signal is determined to be 2.002. Table 4.1 summarizes the result.

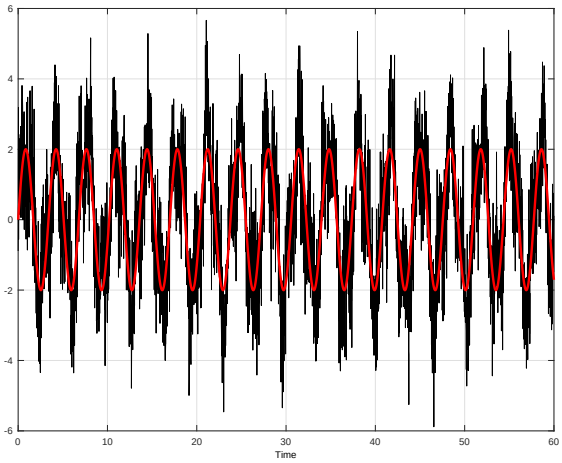


Figure 4.4: Signal from Example 4.4.1 contains two oscillations of periods 1.3 min and 3.4 min. Dominant oscillation period is detected at 3.397 min. The signal is shown in black and the dominant oscillation is shown in red.

Table 4.1: High and Low components of Example 4.4.1

High Cut-Off			Low Cut-off			Selection		
$\overline{T}$	$R_{\overline{T},\alpha/2}$	F	$\overline{T}$	$R_{\overline{T},\alpha/2}$	F	$\overline{T}$	$\overline{M}$	$R_{\overline{M},\alpha/2}$
1.3	346.99	6.43	1.29	56.855	6.43	1.3	0.885	4.08
3.421	52.59	34.19	3.397	77.76	35.72	3.397	2.002	29.2718

Dominant Oscillation		
$\overline{T}$	$\overline{M}$	F
3.397	2.002	35.72

Example 4.4.2. The following case is taken from one of the 61 surgical cases conducted at Royal Columbian Hospital in New Westminster using iControl system. Written consent was taken before the surgery from the patient. This particular patient (case 6 from the database) underwent a Laparoscopic hemicolectomy.

The oscillation starts at time 63*min*, and lasts until time 84*min*. The surgery had started at time 37*min*; there was a stimulation at time 63*min*, as recorded by the anesthesiologist. At time 80*min*, the patient moved. Immediately after, *Rocuronium* was administrated and the oscillation was damped out. The dominant

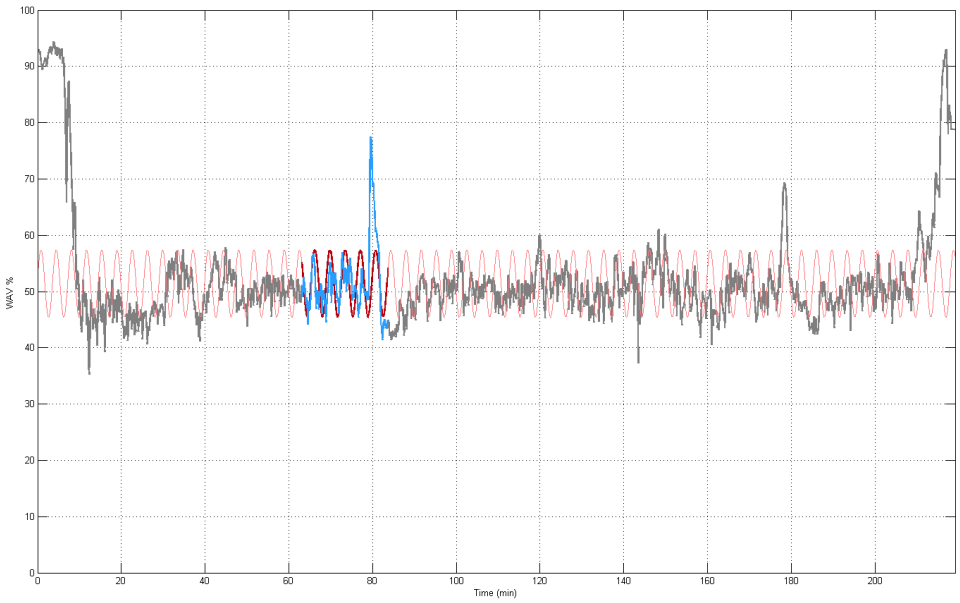


Figure 4.5: On-line oscillation detection shows a detected dominant signal of $T_p = 3.63$ min, $\bar{M} = 5.96$ and $F = 89.31\%$.

Table 4.2: Case example from Example 4.4.2. The *magnitude regulatory index* is 3.785.

High Cut-Off			Low Cut-off		
$\overline{T}$	$R_{\overline{T},\alpha/2}$	F	$\overline{T}$	$R_{\overline{T},\alpha/2}$	F
3.439	3.206	52.468	2.993	3.1132	53.479

Dominant Oscillation		
$\overline{T}$	$\bar{M}$	F
3.44	6.81	53.48

oscillation is $T_p = 3.63$ min, $\bar{M} = 5.96$ and $F = 89.31\%$. The DOH is shown in Figure 4.5.

4.5 Conclusion

Algorithms for detection of oscillation for both off-line (see Section 4.2.3) and on-line/real-time (see Section 4.3.1) were discussed. Unlike existing methods, the algorithm can detect multiple oscillations, ignore specific oscillation frequencies,

and determine the dominant oscillation. The frequency, magnitude and fitness of all measured oscillations is also provided. The fitness of the oscillation can be used to reject small oscillations. The limitation of the algorithm discussed in this chapter, however, is that it requires a signal length of 10 times the presumed oscillation period. However, this is much less than Thornhill's method that requires 50 times the presumed oscillation period. In the next Chapter, the frequency of the dominant will be used to auto-tune a PID controller.

Chapter 5

Re-Tuning of a PID Controller

The closed loop feedback mechanism of PID controllers have found use in a variety of systems, such as process control, motor, and vehicle control, to name a few [6]. The controller contains only three tunable parameters (the *proportional* k, the *integral* k_i, and the *derivative* k_d), yet in many situations it can provide a robust solutions with good performance [6]. Furthermore, this feedback system is well understood and there are numerous implementations and theories to guarantee the robustness and performance.

Our research group has worked extensively on PID control, see [10], [61], [62], [52], [54], [43] and [10]. PID controllers are not only of interest to our group, and other researchers have also investigated them, see [19], [31], [2], and [40].

With the patient's safety in mind, our robust controller is tuned to ensure a reliable and safe drug administration, see all references above. However, there have been cases where some oscillations have been observed in the clinical trials.

It is therefore beneficial to have a system that would be able to detect oscillation in real-time and automatically alert the anesthesiologist. Oscillation provides valuable insight into the plant and the control loop [5]. It is possible to use this new information to re-tune the controller in real-time and remove oscillation.

In this Section, a retuning mechanism that follows the guidelines for a robust PID controller design is discussed. The retuning mechanism is simulated with the 44 PKPD models from Appendix A and compared with the original tuning of iControl. In addition to removing the oscillation, the tuned system has met the following

objectives:

- The gain margin, phase margin, and peak sensitivity should be the same or better than the original design. This translates to a gain margin of more than 2 and phase margin of $30^\circ - 60^\circ$.
- The system should have an overshoot of less than 10% for set point change and disturbance rejection.
- A rise time of 5-10 minutes is considered appropriate. However, even with the current implementation of the closed-loop control, rise time performance criteria is a secondary objective. The rise time of the original parameters should be comparable to the tuned parameters.

This Chapter presents a robust PID tuning method for the currently implemented iControl. The feedback controller structure with all the components of iControl is shown in Section 5.2. The robustness and performance design requirement is examined in Section 5.3. The tuning rules and optimization are discussed in Section 5.4. Finally, simulation results and comparisons are presented in Section 5.7.

5.1 Contribution

A robust PID auto-tuning algorithm is presented in this Chapter. Using the frequency of a measured oscillation in real-time, the patient is identified. Oscillation is generally due to an aggreesive controller. The controller is then tuned to be less aggressive and the sustained oscillation is removed. The tuning rule follows the guidelines of a robust controller. IE optimization is used as a performance criterion.

5.2 Overview of Controller Structure

Consider the 2-degree-of-freedom PID controller shown in Figure 5.1. r is the reference DOH and y is the output WAV. l and d are the input and output disturbances respectively, and n is the measurement noise. The surgical stimulus is represented

by d. The block diagram G_p is the patient, and G_c and G_{ff} are the two LTI controller transfer functions that together describe the PID controller of the form:

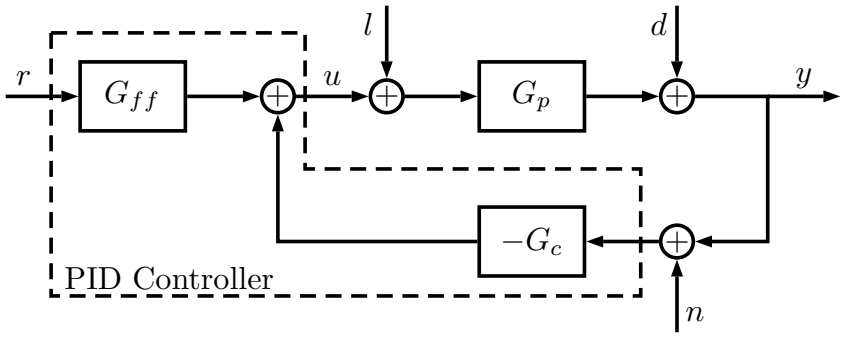


Figure 5.1: A 2-degree-of-freedom PID controller

$$u(t) = k(br(t) - y(t)) + k_i \int_0^t (r(\tau) - y(\tau))d\tau + k_d\big(-\frac{dy(t)}{dt}\big), \tag{5.1}$$

where k, k_i, and k_d are the proportional, integral, and derivative control respectively. b is a step response weighting parameter between 0 and 1. The derivative term does not act on the set-point since that can cause spikes during step changes [6].

Comparing the PID definition 5.1 to the Figure 5.1, it is easy to realize that G_c acts on the signal y and G_{ff} acts on the signal r:

$$\begin{aligned} G_{ff}(s) &= bk + \frac{k_i}{s}, \\ G_c(s) &= k + \frac{k_i}{s} + k_d s. \end{aligned} \tag{5.2}$$

The goal of a PID controller is to track the reference signal r while rejecting any load disturbance, measurement noise and process uncertainty. The relationship between the four signal r, l, d and n to y are:

$$y(s) = \frac{G_{ff}G_p}{1+G_cG_p}r(s) + \frac{G_p}{1+G_cG_p}l(s) + \frac{1}{1+G_cG_p}d(s) - \frac{G_p}{1+G_cG_p}n(s) \tag{5.3}$$

The sensitivity function $S(s)$ describes the transfer function from $d(s)$ to $y(s)$ and the complimentary function $T(s)$ describes the transfer function from r to $y(s)$.

They provide valuable insight: one can design G_c to provide a reasonable disturbance rejection and robustness to process uncertainty. G_{ff} can then be used so the controller meets the performance design criteria. The specification of the robustness and performance will be discussed in Section 5.3.

5.2.1 iControl Design Structure

An overview of iControl is provided below. More information on iControl can be found in [43] and [54]. The control structure is shown in Figure 5.2.

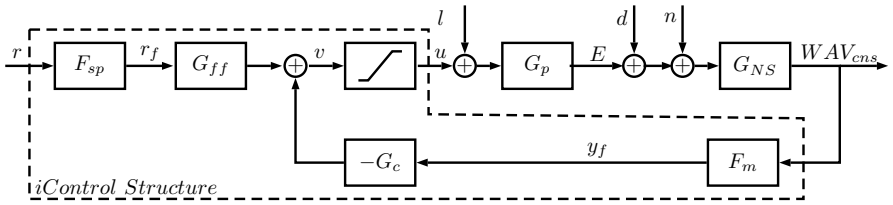


Figure 5.2: The iControl Structure

5.2.1.1 Measurement and Reference Filters

The DOH of the patient in the iControl structure, WAV, is measured by the NeuroSense monitor. To attenuate the high frequency noise, WAV is passed through a second order low-pass measurement filter with time constant $T_m = 15s$:

$$F_m = \frac{1}{1 + sT_m + (sT_m)^2/2}. \tag{5.4}$$

The reference signal is passed through a first-order low-pass set-point filter with time constant $T_{sp} = 25s$ to smooth out any step-like changes:

$$F_{sp} = \frac{1}{1 + sT_{sp}}. \tag{5.5}$$

The filtered signals r_f and y_f are given by:

$$\begin{aligned} r_f(s) &= G_{sp} r(s), \\ y_f(s) &= G_m y(s). \end{aligned} \tag{5.6}$$

5.2.1.2 Saturation and Integrator Anti-Windup

The infusion pump has a lower bound $u_{min} = 0ml/h$ and an upper bound $u_{max} = 600ml/h$. The controller's output v is limited to the saturation values u_{min} and u_{max}.

The saturation block diagram from Figure 5.2 is a non-linear dynamic. It can however be modeled by an ideal *describing function N* as defined in [7]:

$$N = \frac{1}{2}\Big(f_1\big(\frac{a_u + \delta}{a_v}\big) + f_1\big(\frac{a_u - \delta}{a_v}\big)\Big), \tag{5.7}$$

where the function f_1 is given by

$$f_1(\rho) = \begin{cases} 1 & \rho > 1 \\ \frac{2}{\pi}\big(\arcsin\rho + \rho\sqrt{1-\rho^2}\big) & -1 \leq \rho \leq 1 \\ -1 & \rho < -1, \end{cases} \tag{5.8}$$

where the constants a_u, a_v, and δ are given by:

$$\begin{aligned} a_u &= 0.5(u_{max} - u_{min}), \\ a_v &= 0.5(v_{max} - v_{min}), \\ \delta &= u_0 - v_0, \\ u_0 &= 0.5(u_{max} + u_{min}), \\ v_0 &= 0.5(v_{max} + v_{min}), \end{aligned} \tag{5.9}$$

where v_{max} and v_{min} denote the maximum/minimum of the controller's action output v during an oscillation. This dynamic is only active when the controller is saturated.

To prevent the windup that will result from the saturation, a classical tracking anti-windup scheme is implemented. The time constant T_t of the tracking anti-windup is 60 second.

5.2.1.3 NeuroSense Monitor

The EEG signal $E(t)$ can be translated to the DOH index WAV by the NeuroSense monitor. The signal $E(t)$ runs from 0 to 1 and the WAV spans from 0 to 100, with 0 corresponding to an iso-electric EEG, and 100 corresponding to the fully awake state. The dynamics of this monitor are described in a very simple LTI transfer function [54]:

$$G_{NS} = \frac{1}{(8s+1)^2}. \tag{5.10}$$

5.2.1.4 Patient Model

The patient G_p can be replaced by the PKPD model from Appendix A. The block diagram is shown again for reference in Figure A.2. The PK and PD models are LTI functions. The Hill function however, is a non-linear sigmoid function. For control design purposes, it needs to be linearized around the reference point. The linearized PKPD model of the patient is described in Appendix A and is given by:

$$G_p(s) = \frac{k_d \cdot \gamma}{4 \cdot V_1 \cdot EC_{50}} \cdot \frac{(s+k_{12}) \cdot (s+k_{31})}{(s+p_1) \cdot (s+p_2) \cdot (s+p_3) \cdot (s+k_d)} \cdot e^{-T_d s}. \tag{5.11}$$

5.2.1.5 Reference Weighting

In the first version of iControl, suppression of oscillations and rejection of disturbance were prioritized over the performance of the system. The reference weighting b was set to zero. The reference signal only entered the control signal law through the integrator action [54]. To improve performance and reduce induction time, the system was redesigned with a unity reference weighting (i.e. $b = 1$)[61].

5.2.1.6 Current iControl Parameters

The iControl structure has 7 design parameters: k, k_i, k_d, b, T_t, T_{sp} and T_m. The current implementation of iControl has constant values for the anti-windup and the

filter constants:

$$\begin{aligned} T_t &= 60, \\ T_{sp} &= 25, \\ T_m &= 15. \end{aligned}$$

The reference weighting parameter b is set to 1. The PID parameters are based on the age and weight of the patient. The Lean Body Mass (LBM) is defined in [27] as:

$$LBM(w,h) = \begin{cases} 0.3281 \cdot w + 0.33929 \cdot h - 29.5336, & \text{if Male,} \\ 0.29569 \cdot w + 0.41813 \cdot h - 43.2933, & \text{if Female,} \end{cases} \tag{5.12}$$

where w is the weight in kilogram and h is the height in centimeter. The PID parameters are defined as:

$$\begin{aligned} cf &= LBM * 0.03, \\ k &= 0.081 \cdot cf, \\ k_i &= 0.0055 \cdot cf, \\ k_d &= 45 \cdot cf. \end{aligned} \tag{5.13}$$

5.3 Robustness and Performance Design

Following the iControl structure in Section 5.2, the loop transfer function is defined as:

$$L(s) = G_c \cdot G_p \cdot G_{NS} \cdot F_m \cdot N. \tag{5.14}$$

The controller parameters are *matched* to the patient model G_p. The actual patient model may be different to the modeled PKPD G_p. It is important that the controller parameters not to be too sensitive to this process variability.

The Nyquist plot of a loop function is shown in Figure 5.3 in blue. From the Figure, the amplitude (or gain) margin A_m describes how much the *gain* of the loop function $L(s)$ can change before the system become unstable. The phase margin ϕ_m describes how much the *phase* of the $L(s)$ can change before instability is seen.

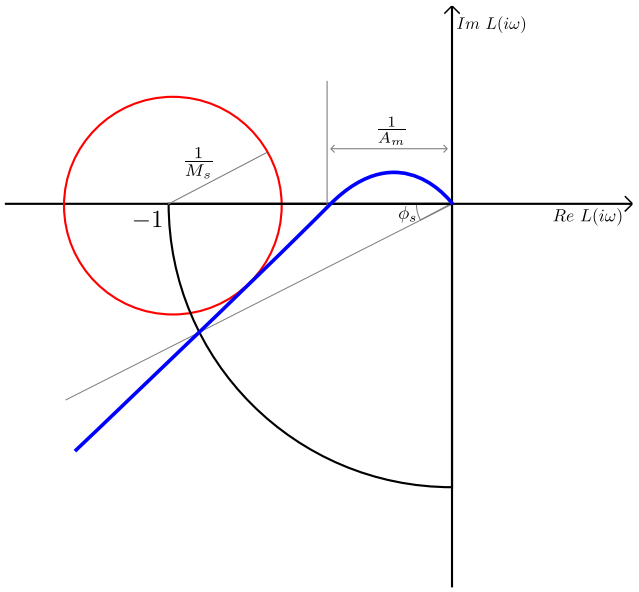


Figure 5.3: The Nyquist Plot with region of stability

The only uncertain function of the loop $L(s)$ is the patient. Therefore, A_m and ϕ_m quantify the upper bounds of how much the patient model can change before the system becomes unstable.

The relationship from d to y is called the *sensitivity function*:

$$S(s) = \frac{1}{1+L(s)}. \tag{5.15}$$

It describes the amplification of the disturbance as a function of frequency. The maximum, or the peak, of the $S(s)$,

$$M_s = \max_{0\leq\omega\leq\infty} |\, S(i\omega)\,|, \tag{5.16}$$

quantifies the worst-case amplification of the disturbance. This quantity is related to the gain margin A_m and phase margin ϕ_m and is provided shortly.

The Nyquist stability criterion defines the point where the function $L(s)$ crosses the negative x-axis at -1 as the instability point. To account for the uncertainty of the patient model, a circle of radius $1/M_s$ (the red circle) centered at -1 is intro-

duced. If the loop function is kept outside of this red circle, then the closed-loop system is guaranteed to have the specified gain margin A_m and phase margin ϕ_m.

There is a relationship between the gain and phase margin and the peak sensitivity:

$$\begin{aligned} A_m &> \frac{M_s}{M_s - 1}, \\ \phi_m &> 2 \arcsin \frac{1}{2M_s}. \end{aligned} \tag{5.17}$$

Typical values of M_s are between 1.3-2. The typical values of A_m are then between 2-5, and the ϕ_m is between 30° and 60° [5].

5.4 PID Auto-Tuning Rules

Given an oscillation of frequency and magnitude ω_u and M_u, where the subscript u stands for *unstable*, following Nyquist's Stability Theorem, the loop function $L(s)$ from Equation 5.14 is crossing the negative real-axis:

$$L(i\omega_u) = -1 \tag{5.18}$$

The idea for retuning is simple: the patient model is identified at this oscillation frequency. Call this point P_u. The retuning of the PID controller is used to shape the loop function to a stable point at the same frequency. Call this point P_s. In Figure 5.4, the dashed line represents the unstable loop function (prior to retuning). The controller tuning then shapes the loop to the blue line, outside of the red circle regime.

The tuning mechanism that follows is motivated by Åström et al. in [6]. In their book, the authors assume the plant is known at a given point, and the controller is tuned to shape the loop to the desired stable point. In the implementation here, the plant is identified at a point from the oscillation.

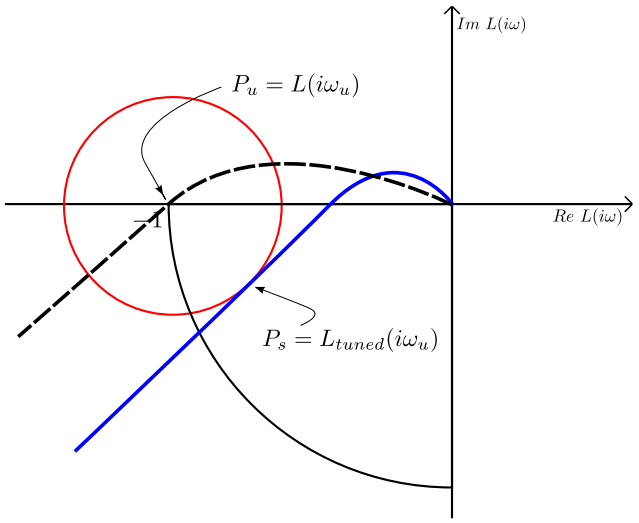


Figure 5.4: The model is identified at the unstable point P_u and the controller is tuned to take the loop function to the stable point P_s.

5.4.1 Auto-Tuning for Robustness

Let the points P_u and P_s be given by their polar representations:

$$\begin{aligned} P_u &= 1e^{i\pi}, \\ P_s &= r_s e^{i\phi_s}. \end{aligned} \tag{5.19}$$

The point P_s is outside of the red circle from Figure 5.3 of radius $1/M_s$. The magnitude and the phase of the stable point relate to the gain A_m and phase ϕ_m margins by the following relationships [22]:

$$\begin{aligned} A_m &= \frac{1}{r_s}, \\ \phi_m &= \phi_s \end{aligned} \tag{5.20}$$

where the gain and phase margins are defined in Section 5.3 and are related to M_s via Equation 5.17. The values used will be discussed in Section 5.5.

Let the transfer functions of the loop function be described by the polar repre-

sentation of a complex system at the oscillation frequency ω_u:

$$\begin{aligned} G_c(i\omega_u) &= r_c e^{i\phi_c}, \\ G_p(i\omega_u) &= r_p e^{i\phi_p}, \\ G_{NS}(i\omega_u) &= r_{NS} e^{i\phi_{NS}}, \\ F_m(i\omega_u) &= r_m e^{i\phi_m}, \\ N(v) &= r_v. \end{aligned} \tag{5.21}$$

The ideal describing function N has zero phase [7]. It is assumed that the control action v will also oscillate and is saturated (i.e. the describing function's dynamic needs to be considered).

The magnitudes and phases of G_c (from 5.2), G_{NS} (from 5.10), F_m (from 5.4), and $N(v)$ (from 5.7) can be computed. The identified r_p and ϕ_p is given by:

$$\begin{aligned} r_p &= \frac{1}{r_c \cdot r_{NS} \cdot r_m \cdot r_v}, \\ \phi_p &= -\big(\phi_c + \phi_{NS} + \phi_m\big). \end{aligned} \tag{5.22}$$

A new PID controller G_{c_t} is determined such that it moves the loop function to the point P_s at the same frequency ω_u:

$$L_{tuned}(i\omega_u) = P_s = r_s e^{i\phi_s} = G_{c_t}(i\omega_u) \cdot G_p(i\omega_u) \cdot G_{NS}(i\omega_u) \cdot F_m(i\omega_u). \tag{5.23}$$

There are two observations:

1. The tuned PID controller action will no longer oscillate. The non-linear saturation dynamics N does not need to be considered.

2. Since the new loop is still computed at the frequency ω_u, the functions $G_p(i\omega_u)$, $G_{NS}(i\omega_u)$, and $F_m(i\omega_u)$ have the same magnitude and phase as 5.21.

Substituting the values of $G_p(i\omega_u)$, $G_{NS}(i\omega_u)$, and $F_m(i\omega_u)$ from 5.21 into Equation 5.23 allows us to solve for the new controller's magnitude and phase:

$$\begin{aligned} r_{c_t} &= \frac{r_s}{r_p \cdot r_{NS} \cdot r_m \cdot r_v} = \frac{r_s}{r_c \cdot r_v}, \\ \phi_{c_t} &= \phi_s - \big(\phi_p + \phi_c + \phi_{NS} + \phi_m\big) = \phi_s - \phi_c. \end{aligned} \tag{5.24}$$

The PID controller has three tuning parameters, k, k_i and k_d. The tuning rule above provides two equations; to get a unique solution, a third equation is needed. In Åström et al, it is suggested to define a ratio between k_i and k_d and set the value of the ratio by trial and error [6]. In the next section, a condition on the performance of the system is imposed on the tuning rule instead.

5.4.2 Auto-Tuning for Performance

In the preceding section, a set of robust rules were defined for auto-tuning the controller. By defining an appropriate gain and phase margin, the tuned controller will be robust and will have good output disturbance rejection.

A common performance criterion is the ability to reject the load disturbance. The *Integrated Error* (IE) can be used as the metric to measure this performance. A small value of IE indicates a fast load disturbance rejection and a small steady state error. It has also been shown that a step-like reference applied at the process's input is directly related to the PID's integrator gain, k_i [6]:

$$IE = \int_0^\infty \big(r(t) - y(t)\big)dt \; = \frac{1}{k_i}. \tag{5.25}$$

The third tuning rule is to maximize the integrator gain k_i so as to minimize the IE and obtain a good load disturbance rejection.

5.4.3 Bumpless Parameter Change

Upon system retuning, parameter change will naturally change the controller's output. This would cause a *bump* as the system's states prior and after the parameters change may not coincide. Care must be taken if a *bumpless* parameter change is required.

To ensure a *bumpless* controller action, it is shown in [6] that is it sufficient to ensure the controller output due to the proportional and the integral component (labeled $P+I$) is invariant to the parameter change. This can be achieved by requiring the state of the integrator to change as:

$$I_{new} = I_{old} + k_{old}(r-y) - k_{new}(r-y), \tag{5.26}$$

where k_{old} and k_{new} are the old and new proportional gain respectively [6].

5.5 Auto-Tuning Implementation

The auto-tuning problem is a minimization problem:

> Minimize Equation 5.25 given the constraints 5.24 for the given gain A_m and phase ϕ_m margins and/or the peak sensitivity M_s.

To allow flexibility for this optimization problem, the gain and phase requirements are provided as a range: an M_s range of 1.3 to 2 is used to guarantee a gain margin of 2.11 to 4.33 and a phase margin of 31° to 45°.

The optimization also needs to be bounded. Otherwise, the tuned parameters can become unbounded (k_i is maximized to minimize IE). Root causes of oscillation were discussed in Chapter 4. It is assumed that the PID is initially properly tuned. Further, it is assumed there is no pump stagnation. The cause of oscillation is assumed to be due to unmatched model uncertainty: the PID controller is simply too aggressive for the patient.

Let k, k_i and k_d be the current PID parameters, as defined by 5.13. Let the k', k'_i and k'_d be the tuned parameters. The tuned parameters are expected to be smaller than the original parameters. This is certainly true for the proportional gain k, however the integral and the derivative gain may need to increase slightly to satisfy the constraints 5.24. The upper and lower bounds of the new PID parameters are given as:

$$\begin{aligned}(k'_{lower}, k'_{upper}) &= (0.75k, k),\\ (k'_{i_{lower}}, k'_{i_{upper}}) &= (0.75k_i, 1.1k_i),\\ (k'_{d_{lower}}, k'_{d_{upper}}) &= (0.75k_d, 1.1k_d).\end{aligned} \tag{5.27}$$

The lower bounds are set to prevent a slow response to stimulation and rise time. The exact values for these parameters were determined via simulation for the 44 PKPD models of [10].

The optimization is solved using MATLAB's *fmincon interior-point* algorithm using $MaxFunEvals = 1e10$, $MaxIter = 1e3$,

5.6 Summary of Auto-Tuning Algorithm

The steps below summarizes the auto-tuning algorithm discussed in this chapter. Prior to running the algorithm, two stability points based on M_s of 1.3 and 2 are defined. These will be the upper and lower acceptable stability points.

1. Run the real-time oscillation detection algorithm discussed in Section 4.3.1 to determine the dominant oscillation frequency.

2. The magnitude and phase of the controller, filters and dynamics of the NeuroSense monitor can now be computed using Equations 5.2, 5.10, 5.4 and 5.7.

3. Identify the patient model at the oscillation frequency using Equation 5.22.

4. Solve for the new PID parameters by minimizing the function 5.25 subject to the conditions of 5.24. Set the upper and lower bounds of the PID parameters as defined in 5.27.

5.7 Simulation Examples and Results

To assess the tuning robustness and performance, the system is simulated. A step change to 50 WAV is applied at time zero. Measurement noise modeled by Soltesz et al. from [52] is applied at time 75 minutes until time 85 minutes. Disturbance, also modeled by Soltesz et al. is applied from time 95 minutes to time 135 minutes. The disturbance models a surgical stimulus. The system initially starts with the tuned parameters. After induction of anesthesia is complete, the system is tuned to be unstable. The *oscillation detection algorithm* from Chapter 4 detects the oscillation and the algorithm from this Section is used to re-tune the system.

Figures 5.5 to 5.8 shows a simulation examples of one patient model from each of the four groups in the PKPD models of Bibian [10]. The black line shows the output of the controller with the original PID tuning and the blue line shows the output of the controller with auto-tuning. The blue line starts with the same PID parameters as the black line. After induction is complete, the tuning is turned to be unstable, causing instability in the output. The controller is auto-tuned and the blue

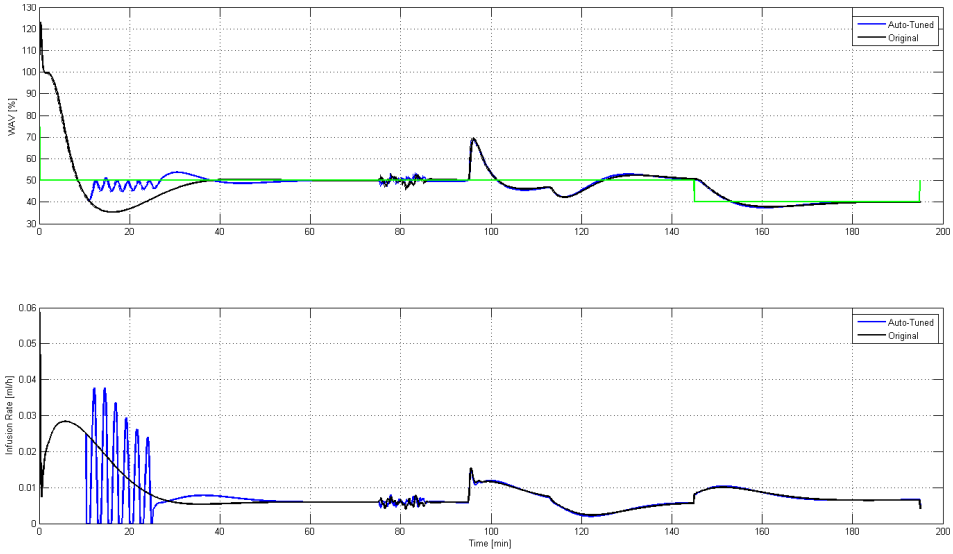


Figure 5.5: Group 1, Case 10: The original tuning has $A_m = 8.72$ and $\phi_m = 60.29$. The re-tuned system has $A_m = 8.62$ and $\phi_m = 53.81$.

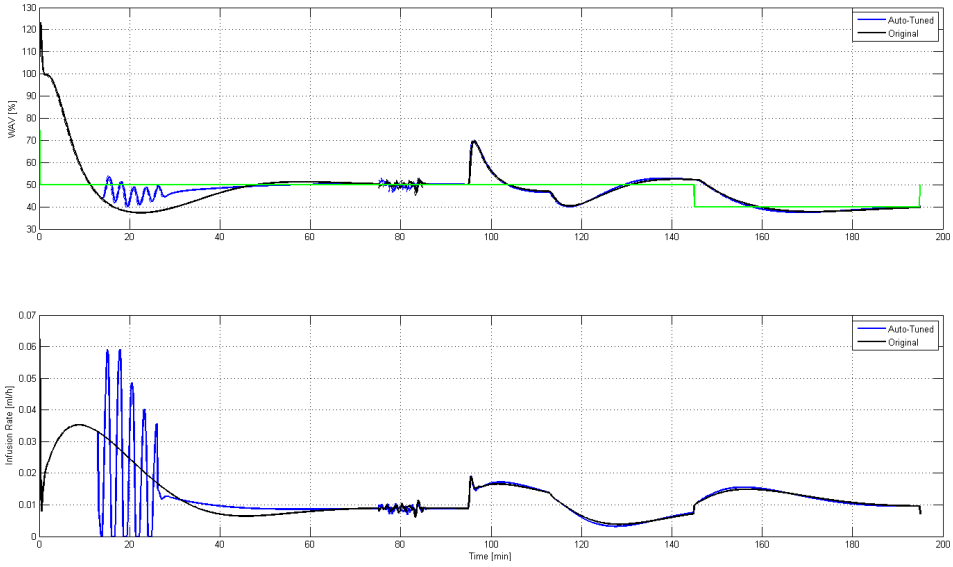


Figure 5.6: Group 2, Case 17: The original tuning has $A_m = 6.27$ and $\phi_m = 61.20$. The re-tuned system has $A_m = 6.13$ and $\phi_m = 55.64$.

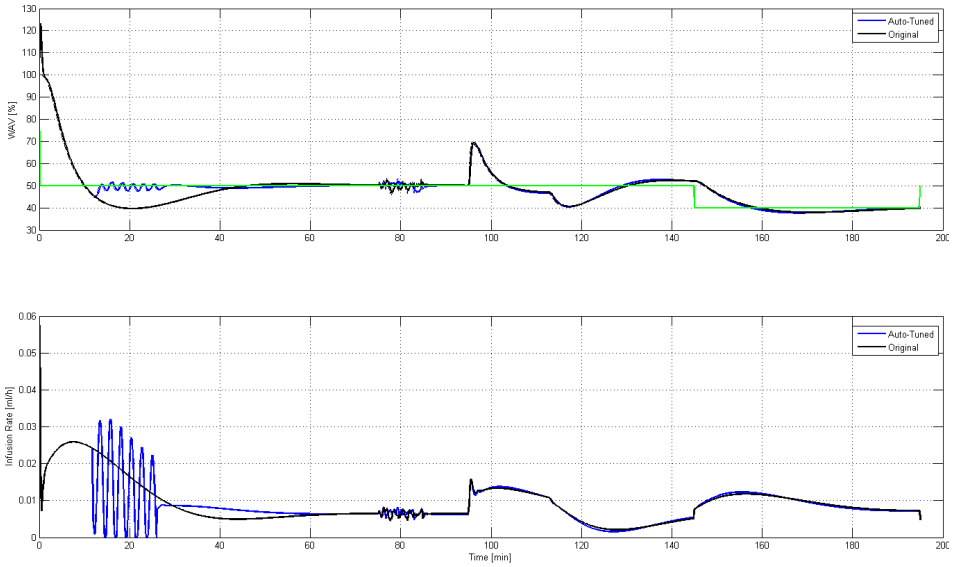


Figure 5.7: Group 3, Case 33: The original tuning has $A_m = 7.57$ and $\phi_m = 65.21$. The re-tuned system has $A_m = 7.43$ and $\phi_m = 58.96$.

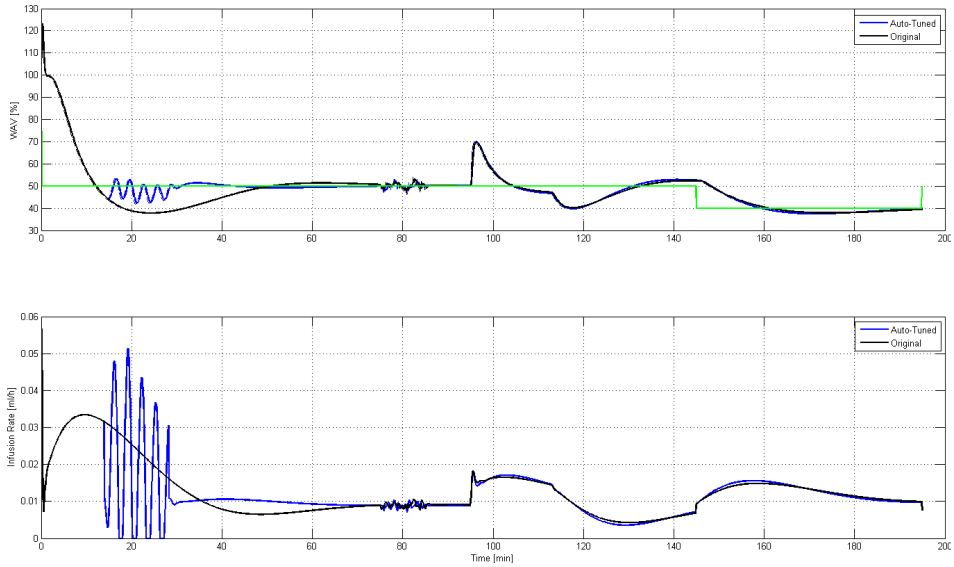


Figure 5.8: Group 4, Case 38: The original tuning has $A_m = 6.89$ and $\phi_m = 61.14$. The re-tuned system has $A_m = 6.86$ and $\phi_m = 55.71$.

signal approaches the black signal. The behavior of all four examples is similar and it highlights the effectiveness of the auto-tuning algorithm to effectively tune the controller and achieve the same disturbance and noise cancellation as well as the same performance as the original tuning.

Tables D.1 to D.3 in the Appendix show the robustness, output disturbance rejection, and step-change response for all 44 models. For *robustness* (Table D.1), the amplitude and phase margins are compared. For *output disturbance rejection* (Table D.2), the maximum overshoot assesses controller's initial response to a 20% WAV disturbance. The settling time T_s measures the time it takes for the WAV to reach within 10% of the set-point following the overshoot. Finally, the IAE measures how well the system rejects the disturbance. For *setpoint-change response* (Table D.3), rise time T_r measures the time it takes for the WAV to reach 80% of the set-point change. Settling time T_s is similar to the output disturbance rejection.

The response of the auto-tuned cases is slightly slower than the original tuning, but otherwise follows them very closely. The slower response is to counter the aggressive tuning that was imposed on the original tuning. Table D.4 shows the tuning parameters. The proportional gain k in all cases is lower than the original tuning.

In the tuning algorithm used, no prior knowledge of the patient model was used. The only assumption is that the initial PID parameters are properly tuned using the prior knowledge of the patient. The measured oscillation is due to unmatched patient uncertainty, though the unmatched parameter is unknown; there is no new information about the patient model.

The median (min, max) of the amplitude margin of the original tuning is 8.55 (4.72, 17.36). The median (min, max) of the phase margin is 61.6° (50.3°, 67.2°). The median (min, max) of the amplitude margin of the re-tuned system is 8.46 (4.75, 12.21). The phase margin is 56.4° (48.8°, 62.1°). The re-tuned system is well within the minimum robustness requirement and agrees with the original tuning [5].

The retuning does not come at a great cost to the output disturbance rejection. The median (min, max) of the IAE of the original tuning is 202.7 (137.5, 304.9). The median (min, max) of the re-tuned system is 209.7 (143, 293.0). The disturbance rejection has slightly increased.

The median (min, max) of the overshoot of the original tuning is 5.26% (3.0%, 6.57%). For the re-tuned system, it is 5.99% (3.01%, 7.63%). All overshoots are still under 10% and is in-line with the original design criteria.

5.8 Conclusion

In this Chapter, oscillation is measured in real time. The frequency of the dominant component is used to automatically re-tune the controller to remove the oscillation. The tuning rule is inspired by Åström et al. [6] is based on a robust design. The plant is identified at the oscillating frequency. A new controller is tuned to shape the loop function to a stable region with a gain margin of more than 2 and a phase margin of $30^\circ - 60^\circ$. Disturbance rejection is guaranteed by minimizing the IE. Percent Overshoot (OS) is kept under 10%. Using the PKPD models of Bibian, the tuned controller is shown to be comparable to the original iControl.

The tuned system guarantees stability at the measured frequency only. The system may still be unstable and oscillate at another frequency. The tuning algorithm should therefore keep the record of all recorded oscillations. On each successive retuning, the optimization constraints should include the list of all previously recorded oscillations.

Chapter 6

Conclusions

6.1 Summary and Contributions

The objectives of this thesis were to 1) assess the applicability of the novel $\mathcal{L}_1$-AC as applied to closed-loop control in anesthesia using the WAV index as the control signal; 2) design an oscillation detection algorithm that can detect multi-period oscillation in real time; and 3) develop a tuning algorithm that can re-tune the PID controller used in iControl to remove the detected oscillation.

Following the requirements for a fast adaptive algorithm with guaranteed robustness, the $\mathcal{L}_1$ Adaptive Control ($\mathcal{L}_1$-AC) was reviewed. This controller claims fast adaptation while maintaining robustness. The fast adaptation is achieved by using a high gain feedback and robustness is achieved by filtering out the high-frequency components of the feedback law using a low-pass filter. It was shown that $\mathcal{L}_1$-AC loses its adaptivity as the gain of the adaptation law increases. Further, the resulting limiting controller can be achieved using an implementable LTI system whose dynamics depend only on the reference model, and not on the patient's unknown parameters. Furthermore, the loss of adaptivity was mathematically shown to be a consequence of the well-known inversion of nonlinearity due to high-gain feedback.

The majority of oscillation detection algorithms currently in practice cannot guarantee the detection of oscillation if multiple oscillation frequency exists. Moreover, most of these algorithms are boolean and can only determine whether or not

an oscillation exists. In a complex systems, multiple oscillations can develop simultaneously. The algorithm of Chapter 4 can 1) determine multiple oscillations; and 2) determine the frequency, magnitude and fitness of each measured oscillation. The fitness of the oscillation can be used to reject insignificant oscillations. The frequency and magnitude of the oscillations can identify the plant. The algorithm requires signal for at least 10 times the presumed oscillation period, which may limit its feasibility for short surgeries. For instance, a presumed oscillation period of 3 minutes can only be accurately measured after 30 minutes. While this is not clinically relevant, it is still better than what is required by other methods - some require 50 times the length of the presumed oscillation period.

One of the biggest challenges of closed-loop control of anesthesia is the inter-patient drug-response variability. The uncertainty in the PKPD model of patients is a challenge for designing a controller than can be both robust and perform well by rejecting surgical stimuli as well as following step-responses. This may lead to an aggressive controller that can cause oscillations. The iControl system was modified to automatically re-tune itself when oscillation was detected subject to the following design objectives:

- The gain margin, phase margin, and peak sensitivity are the same or better than the original design. This would be a gain margin of more than 2 and phase margin of $30^\circ - 60^\circ$.
- The system must have an overshoot of less than 10% for set point change and disturbance rejection.
- A rise time of 5-10 minutes is considered appropriate. However, even with the current implementations of a closed-loop control, rise time performance criteria is a secondary objective. The rise time of the original parameters with the tuned parameters should be comparable.

The robustness and performance of the re-tuned PID controller was compared with the original iControl tuning. The PID re-tuning was applied to 44 PKPD models by Bibian. In all cases, the controller was initially properly tuned according to the latest iControl version. After induction was completed, the system was re-tuned to be unstable and cause oscillation. The dominant oscillation was automatically

measured and the system was then re-tuned. In all 44 cases, it was shown that the re-tuned controller had similar robustness and performance behavior as the original iControl tuning.

The current standard of performance evaluation of closed-loop control is the set of Varvel measures. These measures are not adequate for the current structures of closed-loop control since they cannot be used design criteria. A set of proposed measures by Soltesz et al. were shown to correlate with the Varvel measures. Unlike Varvel, however, these proposed measures are accepted within the control community and are used as performance criteria. The Integrated Error (IE), Percent Overshoot (OS), and Induction Phase Duration (ID) were used as design objectives for re-tuning the PID controller.

6.2 Future Work

The possible future directions based on the work of the thesis are outlined below. The works are separated as ”Imminent” and ”Distant” directions.

6.2.1 Imminent Future Direction

A limitation of the oscillation detection is the required length data of 10 times the presumed oscillation period. However, the system can moderately predict and warn the anesthesiologist with less data. For example, the system could alert the anesthesiologist by monitoring only 3 or 4 times the presumed oscillation period. This could be displayed as a ”probable” oscillation. As more data is gathered, the confidence on the measured oscillation can increase to ”certain”. Visually, the DOH can be in green when no oscillation is present. It can turn to yellow when there is a probable oscillation, and finally red when oscillation is detected.

The re-tuning algorithm for the PID controller can only guarantee robustness for the measured oscillation frequency. Another oscillation may occur at a different frequency at a later time. The tuning algorithm should therefore keep a record of all measured oscillations and on every successive oscillation, include all the recorded oscillations as constraints on the re-tuning optimization. Such a system can be beneficial for use in the ICU, where a patient may be sedated for a few days.

The oscillation detection algorithm can form the basis of a measurement that

can give a score to how oscillatory a signal is. This can complete the proposed Soltesz alternatives to Varvel metrics. The proposed measures currently does not quantify oscillation and Varvel's Wobble metric does not have a substitute.

The proposed alternative measures to the Varvel metrics requires more study and verification. This is needed to create a set of measures that is acceptable by both the clinicians and control engineers. Without these, no two closed-loop controllers can be compared. These measures can also facilitate communication between the clinicians and the control engineers and can act as excellent diagnostic metrics.

6.2.2 Distant Future Direction

The $\mathcal{L}_1$-AC in its current form cannot guarantee the fast adaptation it claims, nor can it guarantee adaptivity at a high adaptation speed. However, adaptive controllers may be the only feasible solution to a completely autonomous closed-loop controller that is truly robust and performs well despite surgical stimuli. Model-Predictive-Controller (MPC) is a promising adaptive controller that has been studied by other researchers. Our own research team is also working on MPC.

Finally, the phrase "closed-loop control of anesthesia" has been loosely used to describe systems that only control the Depth of Hypnosis (DOH). A true control of Depth of Anesthesia (DOA) requires monitoring several physiological signals including the EEG, blood pressure, heart rate, respiratory rate, heart rate, etc. There are also a variety of hypnotic, opioid, and neuromuscular drugs that are administrated to the patient to achieve a full state of anesthesia. From the controller's point of view, this corresponds to a MIMO system. The ideal controller should measure DOA (both hypnotic and analgesic state) as well as level of paralysis, and automatically control the infusion of all anesthetic drugs. This MIMO controller is currently not available, but should be the holy grail of the closed-loop control of anesthesia.

Bibliography

[1] A. Absalom and G. Kenny. Closed-loop control of propofol anaesthesia using bispectral index: performance assessment in patients receiving computer-controlled propofol and manually controlled remifentanil infusions for minor surgery. *British Journal of Anaesthesia*, 90(6):737–741, 2003. → pages 12, 13

[2] A. Absalom, N. Sutcliffe, and G. Kenny. Closed-loop control of anesthesia using bispectral index: Performance assessment in patients undergoing major othopedic surgery under combined general and regional anesthesia. *Anesthesiology*, 96:67–73, 2002. → pages 4, 7, 12, 13, 63

[3] G. Agrawal, S. Bibian, and T. Zikov. Recommended clinical range for wav_{CNS} index during general anesthesia. In *Annual Meeting of the American Society Anesthesiologists*, 2010. → pages 100

[4] N. Ahmed, T. Natarajan, and K. Rao. Discrete consine transform. *IEEE Transactions on Computers*, 1:90–93, 1974. → pages 47

[5] K. J. Åström and T. Hägglund. *Advanced PID Control*. ISA - The Instrumentation, Systems, and Auomation Society, 2006. → pages 14, 63, 71, 79

[6] K. J. Åström and R. M. Murray. *Feedback Systems: An Introduction for Scientists and Engineers*. Princeton University Press, 2008. → pages iii, 63, 65, 71, 74, 75, 80

[7] D. Atherton. *Nonlinear Control Engineering*. Van Nostrand Reinhold London, 1982. → pages 67, 73

[8] M. Avidan, L. Zhang, B. Burnside, K. Finkel, A. Searleman, J. Selvidge, L. Saager, M. Turner, S. Rao, M. Bottros, C. Hantler, E. Jacobsohn, and A. Evers. Anesthesia awareness and the bispectral index. *New England Journal of Medicine*, 2008. → pages 8, 9

[9] S. Babji, U. Nallasivam, and R. Rengaswamy. Root cause analysis of linear closed-loop oscillatory chemical process systems. *Industrial & Engineering Chemistry Research*, 51:13712–13731, 2012. → pages 13

[10] S. Bibian. *Automation in Clinical Anesthesia*. PhD thesis, University of British Columbia, 2006. → pages viii, xii, 3, 5, 9, 10, 14, 63, 75, 76, 92, 94, 96, 97, 98

[11] S. Bibian, T. Zikov, G. A. Dumont, C. Ries, H. Puil, H. Ahamdi, M. Huzmezan, and B. A. MacLeod. Estimation of anesthetic depth using wavelet analysis of electroencephalography. In *EMBC International Conference*, 2001. → pages 2, 9

[12] S. Bibian, T. Zikov, G. A. Dumont, C. Ries, H. Puil, H. Ahamdi, M. Huzmezan, and B. A. MacLeod. Method and apparatus for the estimation of anesthetic depth using wavelet analysis of the electroencephalogram, 2004. URL http://www.google.ca/patents/US20040010203. → pages 2

[13] J. D. Boskovic and R. K. Mehra. Performance analysis of a simple $\mathcal{L}_1$ adaptive control. In *American Control Conference (ACC)*, 2013. → pages 23, 33

[14] C. Cao and Hova. Stability margins of $\mathcal{L}_1$ adaptive controller. *IEEE Transactions on Automatic Control*, 55(2):480–487, 2010. → pages 35

[15] C. Cao and N. Hovakimyan. $\mathcal{L}_1$ adaptive controller for systems with unknown time-varying parameters and disturbances in the presence of non-zero trajectory initialization error. *International Journal of Control*, 81 (2):586–591, 2008. → pages 35

[16] CareerCast.com. Jobs rated 2013: Ranking 200 jobs from best to worst. http://www.careercast.com/jobs-rated/best-worst-jobs-2013, 9 2013. Accessed: 2014-09-04. → pages 7

[17] CDC. National hospital discharge survey: 2010 table, procedures by selected patient characteristics - number by procedure category and age. Technical report, Centers for Disease Control and Prevention, 2010. → pages 1

[18] D. J. Cullen, E. Eger, W. C. Stevens, N. T. Smith, T. H. Cromwell, B. F. Cullen, G. A. Gregory, S. H. Bahlman, W. M. Dolan, R. K. Stoelting, and H. Fourcade. Clinical signs of anesthesia. *Anesthesiology*, 36:21–37, 1972. → pages 1

[19] K. Ejaz and J. S. Yang. Controlling depth of anesthesia using pid tuning: A comparative model-based study. In *International Conference on Control Application*, 2004. → pages 63

[20] B. D. Franklin, K. OGrady, P. Donyai, A. Jacklin, and N. Barber. The impact of a closed-loop electronic prescribing and administration system on prescribing errors, administration errors and staff time: a before-and-after study. *International Journal of Healthcare Improvement*, 2007. → pages 7

[21] C. W. Frei, A. Gentilini, M. Derighetti, A. H. Glattfelder, M. Morari, T. Schnider, and A. M. Zbinden. Automation in anesthesia. In *American Control Conference*, 1999. → pages 11

[22] M. Friman. Automatic re-tuning of pi controllers in oscillating control loops. Technical report, Abo Akademi University, 1997. → pages 72

[23] G. Goodwin, S. Graebe, and M. Salgado. *Control System Design*. Prentice Hall, NJ, USA, 2001. → pages 33, 108

[24] M. Gruenewald, C. Ilies, J. Herz, T. Schoenherr, A. Fudickar, J. Hcker, and B. Bein. Influence of nociceptive stimulation on analgesia nociception index (ani) during propofolremifentanil anaesthesia. *British Journal of Anaesthesia*, 110:1024–1030, 2013. → pages 2

[25] T. Hägglund. A control-loop performance monitor. *Control Engineering Practice*, 3:1543–1551, 1995. → pages 13, 46

[26] N. Hovakimyan and C. Cao. *L1 Adaptive Control Theory: Guaranteed RobuRobust with Fast Adaptation*. Society for Industrial & Applied Mathematics, 2010. → pages ix, xii, 3, 14, 22, 24, 26, 28, 104, 106

[27] R. Hume. Prediction of lean body mass from height and weight. *Journal of Clinical Pathology*, 19:389–391, 1966. → pages 69

[28] P. A. Ioannou and J. Sun. *Robust Adaptive Control*. Dover Publications, Inc, 2012. → pages 14, 22

[29] S. Jafari, P. Ioannou, and L. Rudd. What is $\mathcal{L}_1$ adaptive control. In *AIAA Guidance, Navigation, and Control (GNC) Conference*, 2013. → pages 23, 33

[30] M. Jeanne, C. Clement, J. de Jonckheere, R. Logier, and B. Tavernier. Variations of the analgesia nociception index during general anaesthesia for laparoscopic abdominal surgery. *Journal of Clinical Monitoring and Computing*, 26:289–294, 2012. → pages 2

[31] G. Kenny and H. Mantzaridis. Closed-loop control of propofol anaesthesia. *British Journal of Anaesthesia*, 83:223–228, 1999. → pages 12, 63

[32] S. Kerra, M. Jelali, M. N. Karim, and A. Horch. *Detection and diagnosis of stiction in control loops*, chapter Detection of oscillating control loops, page Chapter. 14. Springer, 2009. → pages 46

[33] E. Kharisov and N. Hovakimyan. Generalization of $\mathcal{L}_1$ adaptive control architecture for switching estimation laws. In *American Control Conference*, 2012. → pages 37, 38

[34] E. Lavretsky and T. E. Gibson. Projection operator in adaptive systems. Technical report, Cornell University Library, 2011. → pages 25

[35] E. Lavretsky and K. A. Wise. *Robustness and Adaptive Control*. Springer, 2013. → pages 22

[36] T. Ledowski, W. Tiong, C. Lee, B. Wong, T. Fiori, and N. Parker. Analgesia nociception index: evaluation as a new parameter for acute postoperative pain. *British Journal of Anaesthesia*, 111:627 to 629, February 2013. → pages 2

[37] T. L. Lemke, D. Williams, R. V. F., and S. W. Zite. *Foye's Principles of Medicinal Chemistry*. Lippincott Williams & Wilkins, 2012. → pages 2

[38] X. Li, J. Wang, B. Huang, and S. Lu. The dct-based oscillation detection method for a single time series. *Journal of Process Control*, 20:609–617, 2010. → pages 54

[39] N. Liu, T. Chazot, T. Chazot, A. Gentr, A. Landais, A. Restoux, K. McGee, P. Laloe, B. Trillat, L. Barvis, and M. Fischler. Titration of propofol for anesthetic induction and maintenance guided by the bispectral index: closed-loop versus manual control: A prospective, randomized, multicentral study. *Anesthesiology*, 2006. → pages 7, 11, 12, 97, 100, 101, 102

[40] N. Liu, T. Chazot, S. Hamada, A. Landais, N. Boichut, C. Dussaussoy, B. Trillat, L. Beydon, E. Samain, D. Sessler, and M. Fischler. Closed-loop coadministration of propofol and remifentanil guided by bispectral index: a randomized multicenter study. *Anesthesia & Analgesia*, 112:546–557, 2011. → pages 63

[41] L. Ljung. *System identification toolbox 7: Getting started guide*. MathWorks, 2010. → pages 54

[42] S. Locher, K. Stadler, T. Boehlen, T. Bouillon, D. Leibundgut, P. Schumacher, R. Wymann, and A. M. Zbinden. A new closed-loop control system for isoflurane using bispectral index outperforms manual control. *Anesthesiology*, 101(3):591–602, 2004. → pages 12

[43] A. Martinez. Robust control: Pid vs fractional control design, a case study. Master's thesis, University of British Columbia, 2005. → pages 9, 63, 66, 92, 97

[44] T. Matsuo, H. Sasaoka, and Y. Yamashita. Detection and diagnosis of oscillations in process plants. In *Process of Seventh Int. Conf. Knowledge-Based Intelligent Information and Engineering Systems*, pages 1258–1264, UK, 2003. → pages 46

[45] R. Ortega. On the effect of input filtering and fast adaptation in model reference adaptive control. In *American Control Conference*, 2013. → pages 23

[46] M. Ralph, L. Beck, and M. Bloom. $\mathcal{L}_1$-adaptive mmethod for control of patient respones to anesthesia. In *American Control Conference*, 2011. → pages 13

[47] T. Schnider, C. Minto, P. Gambus, C. Andersen, D. Goodale, S. Shafer, and E. Youngs. The influence of method of administration and covariates on the pharmacokinetics of propofol in adult volunteers. *Anesthesiology*, 88: 1170–1182, 1998. → pages 9

[48] J. Schüttler and H. Imsen. Population pharmacokinetics of propofol: A multicenter study. *Anesthesiology*, 92:727–738, 2000. → pages viii, 9, 94, 95

[49] L. Sheiner, S. D.R, S. Vozeh, R. Miller, and J. Ham. Simultaneously modeling of pharmacokinetics and pharmacodynamics: application to d-tubocurarine. *Clinical Pharmacology and Therapeutics*, 25:358–371, 1979. → pages 10, 96

[50] J. Sigl and N. G. Chamoun. An introduction to bispectral analysis for the electroencephalogram. *Journal of Clinical Monitoring*, 10(6):392–404, 1994. → pages 2, 8

[51] J. Smith. The best- and worst-paying jobs in america. http://www.forbes.com/sites/jacquelynsmith/2013/05/13/the-best-and-worst-paying-jobs-in-america-2/, 5 2013. Accessed: 2014-09-04. → pages 7

[52] K. Soltesz, K. van Heusden, G. A. Dumont, T. Hägglund, C. L. Peterson, N. West, and M. Ansermino. Closed-loop anesthesia in children using a pid controller: A pilot study. In *IFAC*, 2012. → pages 2, 3, 7, 14, 63, 76

[53] K. Soltesz, G. A. Dumont, and M. Ansermino. Assessing control performance in closed-loop anesthesia. In *Mediterranean Conference on Control & Automation*, 2013. → pages 4, 15, 101

[54] K. Soltesz, J. Hahn, T. Hägglund, G. A. Dumont, and M. Ansermino. Individualized closed-loop control of propofol anesthesia: A preliminary study. *Biomedical Signal Processing and Control*, 8:500–508, 2013. → pages 3, 63, 66, 68

[55] R. Srinivasana, R. Rengaswamy, and R. Miller. A modified empirical mode decomposition(emd) process for oscillation characterization in control loops. *Control Engineering Practice*, 15:1135–1148, 2007. → pages 46

[56] M. Struys, T. Smet, L. F. M. Versichelen, S. Van de Velde, R. Van den Broecke, and E. Mortier. Comparison of closed-loop controlled administration of propofol using bispectral index as controlled variable versus "standard practice" controlled administration. *Anesthesiology*, 95: 6–17, 2001. → pages 12

[57] K. Talebian, K. Soltesz, G. A. Dumont, and M. Ansermino. Clinical assessment of control performance in closed-loop anesthesia. In *American Society of Anesthesiologists*, 2013. → pages 13, 15

[58] N. Thornhill and T. Hägglund. Detection and diagnosis of oscillation in control loops. *Control Engineering Practice*, 5:1343–1354, 1997. → pages 45, 57

[59] N. Thornhill, B. Huang, and H. Zhang. Detection of multiple oscillations in control loops. *Journal of Process Control*, 13:91–100, 2003. → pages 52

[60] K. van Heusden and G. A. Dumont. Analysis of linear $\mathcal{L}_1$ output feedback control: Equivalent lti controllers. In *16th IFAC Symposium on System Identification*, pages 1472–1477, Brussels, Belgium, 2012. → pages 33

[61] K. van Heusden, G. A. Dumont, K. Soltesz, C. L. Peterson, A. Umedaly, N. West, and M. Ansermino. Design and clinical evaluation of robust pid control of propofol anesthesia in children. *IEEE Transactions on Control Systems Technology*, 22:491–499, 2014. → pages 15, 63, 68

[62] K. van Heusden, N. West, A. Umedaly, M. Ansermino, R. N. Merchant, and G. A. Dumont. Safety, constraints and anti-windup in closed-loop anesthesia. In *IFAC*, 2014. → pages 63

[63] K. van Heusden, K. Talebian, and G. A. Dumont. Analysis of l1 adaptive state feedback control. why does it approximate an implementable lti controller? *European Journal of Control*, 23:1– 7, May 2015. → pages 23, 33

[64] J. Vannes, E. Kharisov, and Hov. $\mathcal{L}_1$ adaptive control with proportional adaptation. In *American Control Conference*, 2012. → pages 23, 35, 36, 37

[65] J. R. Varvel, D. L. Donoho, and S. Shafer. Measuring the predictive performance of computer-controlled infusion pumps. *Journal of Pharmacokinetics and Biopharmaceutics*, 20:63–94, 1998. → pages 3, 99

[66] F. C. Vasella, P. Frascarolo, D. R. Spahn, and L. Magnusson. Antagonism of neuromuscular blockade but not muscle relaxation affects depth of anaesthesia. *British Journal of Anaesthesia*, 94:742–747, 2005. → pages 2

[67] J. Wang, B. Huang, and S. Ku. Improved dct-based method for online detection of oscillations in univariate time series. *Control Engineering Practice*, 21:622–630, 2013. → pages iii, 13, 46, 49, 50, 54, 55, 56, 57

[68] N. West, G. A. Dumont, K. van Heusden, C. L. Peterson, S. Khosravi, K. Soltesz, A. Umedaly, E. Reimer, and M. Ansermino. Robust closed-loop control of induction and maintenance of propofol anesthesia in children. *Pediatric Anesthesia*, 23:712–719, 2013. → pages 3

[69] T. Zikov. Monitoring the anesthetic-induced unconsciousness (hypnosis) using wavelet analysis of the electroencephalography. Master's thesis, University of British Columbia, 2002. → pages 2, 9

Appendix A

Propofol PKPD Modeling

The pharmacokinetic (PK) and pharmacodynamic (PD) models are described here. PK models the distribution of the drug in the body and predicts the blood plasma concentration (C_p) of the drug. The PD models the effect of the drug from the drug plasma concentration. Propofol is the fastest anesthetic agent currently available [10]. A closed-loop control system with a fast reacting agent is more ideal. The PK and PD models discussed here are for this hypnotic agent.

In this Appendix, a quick summary and the mathematical models are provided. More in depth discussion may be found in [43] and [10].

A.1 Pharmacokinetics

The drug uptake, distribution and elimination can be expressed mathematically by a pharmacokinetic model. There are a few models available; the exponential and mamillary compartment models are the most common [10]. The mamillary compartment model is discussed below.

The body is divided into three compartments: 1) a central compartment consisting of the blood, brain and liver; 2) a larger compartment consisting of muscle and viscera; and 3) a third compartment consisting of bones and fat. The 3-compartment model is shown in Figure A.1.

The drug is administrated into the central compartment intravenously. It is eliminated from the body according to the rate k_{10} through hepatic and/or renal

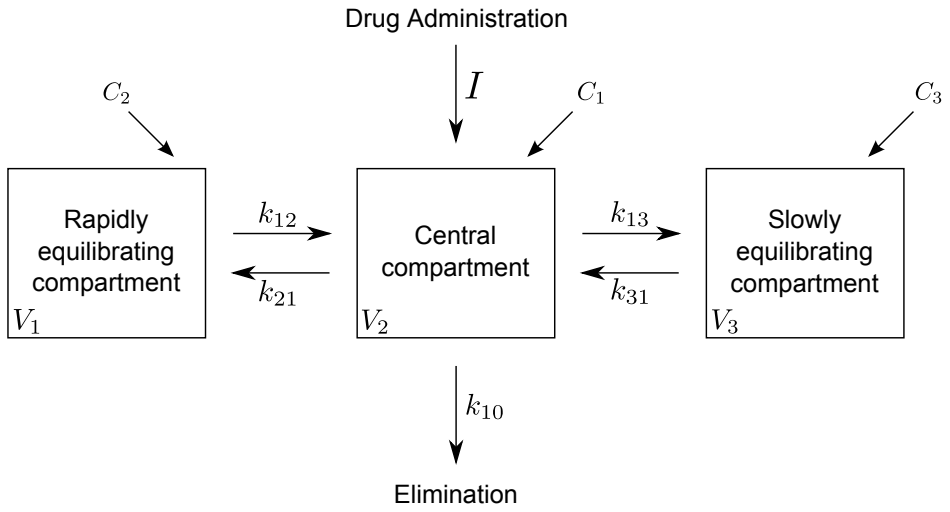


Figure A.1: The 3-compartment pharmacokinetic model. The rapidly equilibrating compartment models the muscles and viscera. The central compartment models the blood, brain and liver. The slowly equilibrating compartment models the bones and fat.

extraction. The concentration in the central compartment comes to an equilibrium with the muscle-viscera compartment through the rate constant k_{21} (and the reverse rate k_{12}). The concentration in the central compartment also comes to an equilibrium with the bones-fat compartment through the rate constant k_{13} (and the reverse rate k_{31}). These rates are usually provided in the units of min^{-1}. The concentration of the central compartment (C_1) increases following the bolus but rapidly decreases as the concentration of the muscle-viscera (C_2) and bones-fat (C_3) increases to balance the equilibrium.

The blood plasma concentration of the drug is the concentration of the central compartment; the mass-balance representation of this compartment in the state-

space is given by [10]:

$$\begin{aligned}
\begin{bmatrix} \dot{C}_1(t) \\ \dot{C}_2(t) \\ \dot{C}_3(t) \end{bmatrix} &= \begin{bmatrix} -(k_{10}+k_{12}+k_{13}) & k_{21} & k_{31} \\ k_{12} & -k_{21} & 0 \\ k_{13} & 0 & -k_{31} \end{bmatrix} \cdot \begin{bmatrix} C_1(t) \\ C_2(t) \\ C_3(t) \end{bmatrix} + \begin{bmatrix} \frac{1}{V_1(t)} \\ 0 \\ 0 \end{bmatrix} \cdot I(t), \\
C_p(t) &= \begin{bmatrix} 1 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} C_1(t) \\ C_2(t) \\ C_3(t) \end{bmatrix}
\end{aligned} \tag{A.1}$$

where V_1 is the volume of the central compartment; $I(t)$ is the drug infusion rate; and by definition $C_p(t) = C_1(t)$ is the concentration of the central compartment.

The clearance is the rate at which the drug is removed from a compartment and is expressed in $[ml \cdot min^{-1}]$. The total body clearance Cl_1 is given by:

$$Cl_1 = V_1 \cdot k_{10} \tag{A.2}$$

Likewise, the inter-compartmental clearances $Cl_{12}, Cl_{21}, Cl_{13}$, and Cl_{31} are given by $Cl_{ij} = V_i \cdot k_{ij}$. It is easy to realize that $Cl_{12} = Cl_{21} = Cl_2$ and $Cl_{13} = Cl_{31} = Cl_3$.

The parameters k_{ij} of the equation A.1 are computed according to the study published by Schüttler et al. [48]. This population-based study relates the 3-compartment clearances and the volumes to the patient's body weight and age as well as the administration type (bolus vs infusion) and the sampling site (venous vs arterial). The values of the clearance and the volumes are shown in Table A.1. The estimates of the intermediates parameters of the Table are given in the Table A.2. The relationship of 3-compartment clearances and the volumes to the plasma concentration parameters are given in Equation A.3.

$$\begin{cases}
k_{10} = \frac{Cl_1}{V_1} \\
k_{12} = \frac{Cl_2}{V_1} \\
k_{21} = \frac{Cl_2}{V_2} \\
k_{13} = \frac{Cl_3}{V_1} \\
k_{31} = \frac{Cl_3}{V_3}
\end{cases} \tag{A.3}$$

Table A.1: Propofol PK parameters from [48]. BW stands for body weight, $ven = 0$ is for arterial sampling, $ven = 1$ is for venous sampling, $bol = 0$ is for infusion administration, and $bol = 1$ is for bolus administration.

PK Parameter	Value	Unit
Cl_1	$\theta_1 \cdot (BW/70)^{\theta_7}$ if age ≤ 60	l·min^{-1}
	$\theta_1 \cdot (BW/70)^{\theta_7} - (age - 60) \cdot \theta_{10}$ otherwise	
Cl_2	$\theta_3 \cdot (BW/70)^{\theta_8} \cdot (1 + ven \cdot \theta_{14}) \cdot (1 + bol \cdot \theta_{16})$	l·min^{-1}
Cl_3	$\theta_5 \cdot (BW/70)^{\theta_{11}} \cdot (1 + bol \cdot \theta_{18})$	l·min^{-1}
V_1	$\theta_2 \cdot (BW/70)^{\theta_{12}} \cdot (age/30)^{\theta_{13}} \cdot (1 + bol \cdot \theta_{15})$	l
V_2	$\theta_2 \cdot (BW/70)^{\theta_{12}} \cdot (1 + bol \cdot \theta_{17})$	l
V_3	θ_6	l

Table A.2: Parameter estimates of the the PK model of Table A.1 from [48].

Parameter Estimate	Value	SE	Units
θ_1	1.44	0.09	l·min^{-1}
θ_2	9.3	0.9	l
θ_3	2.25	0.31	l·min^{-1}
θ_4	44.2	6.1	l
θ_5	0.92	0.15	l·min^{-1}
θ_6	266	43	l
θ_7	0.75	0.06	l·min^{-1}
θ_8	0.62	0.09	l
θ_9	0.61	0.11	l·min^{-1}
θ_{10}	0.045	0.012	l
θ_{11}	0.55	0.13	l·min^{-1}
θ_{12}	0.71	0.26	l
θ_{13}	-0.39	0.15	l·min^{-1}
θ_{14}	-0.40	0.10	l
θ_{15}	1.61	0.36	l·min^{-1}
θ_{16}	2.02	0.41	l
θ_{17}	0.73	0.23	l·min^{-1}
θ_{18}	-0.48	0.12	l

The state-space representation of the PK model of A.1 can be given as a Single-Input/Single-Output (SISO) transfer function $PK(s)$:

$$PK(s) = \frac{C_p(s)}{I(s)} = \frac{1}{V_1} \cdot \frac{(s+k_{12}) \cdot (s+k_{31})}{(s+p_1) \cdot (s+p_2) \cdot (s+p_3)} \tag{A.4}$$

where p_i are the poles of the system.

A.2 Pharmacodynamics

The pharmacological response of a drug as a function of the drug plasma concentration can be expressed mathematically by a pharmacodynamic model. Any drug can target multiple organs in the body, resulting in multiple effects; there is not a single unique pharmacological response for a given plasma concentration of a drug. Here, the intake is the hypnotic drug propofol and the pharmacological effect modeled is the depth of hypnosis.

The full pharmacological model can be represented by a LTI transfer function $PD(s)$ and a non-linear sigmoid-type function known as Hill-equation [10].

The LTI element models the effect site concentration $C_e(s)$ from the dynamics of the drug-receptor interaction. There will also be a delay for the drug to reach the effect site from the plasma concentration. This LTI model is given by the first order time-delayed transfer function $PD(s)$:

$$PD = \frac{C_e(s)}{C_p(s)} = \frac{1}{2EC_{50}} \cdot \frac{k_d}{s+k_d} \cdot e^{-T_d s}, \tag{A.5}$$

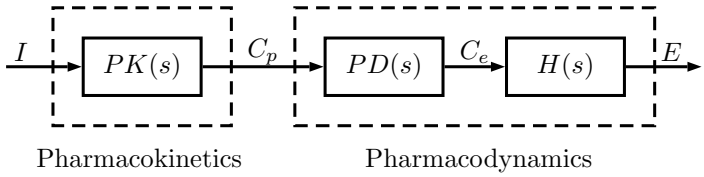
where EC_{50} is the plasma concentration which yields 50% of the maximum effect; k_d expresses the rate of the transfer of plasma concentration to the effect site introduced in [49]; and T_d is the arm-to-brain delay.

The non-linear Hill function models the dynamics of observed effect $E(s)$ to the effect-site concentration $C_e(s)$. The observed effect $E(s)$ runs from 0 (no hypnotic effect) to 1 (fully awake). This model is given by:

$$H(s) = \frac{E(s)}{C_e(s)} = E_0 + E_{max} \cdot \frac{C_e^{\gamma}(s)}{EC_{50}^{\gamma} + C_e^{\gamma}}, \tag{A.6}$$

where E_0 and E_{max} are the minimum and maximum effects, and γ is a measure of the steepness of the dose-response curve.

Figure A.2: The full PKPD model introduced in [10].



A.3 The PKPD Model

The complete model dynamics relating the drug administration $I(s)$ of propofol to the observed effect $E(s)$ is shown in Figure A.2 can now be introduced:

$$PKPD(s) = \frac{E(s)}{I(s)} = PK(s) \cdot PD(s) \cdot H(s) \tag{A.7}$$

This structure represents a nonlinear SISO transfer function that relates the infusion of Propofol $I(s)$ to the observed effect $E(s)$. This effect can then be converted to a DOH index through a EEG monitor, such as the NeuroSense Monitor.

The nonlinearity $H(s)$ can be linearized around the DOH of interest. In most cases, a DOH of 50 is considered adequate [39]. This value represents an $E(s) = 0.5$ and is a logical operating point to linearize the Hill function around it. This assumption is only valid for the maintenance phase of anesthesia. The linearized Hill function around $E = 0.5$ is $\gamma/2$. The full detail of the linearizion can be found in [43].

The linearized $PKPD(s)$ of A.7 can now be expressed as:

$$PKPD(s) = K_{pkpd} \cdot \frac{(s+k_{12}) \cdot (s+k_{31})}{(s+p_1) \cdot (s+p_2) \cdot (s+p_3) \cdot (s+k_d)} \cdot e^{-T_d s} \tag{A.8}$$

where the patient's model gain K_{pkpd} is:

$$K_{pkpd} = \frac{k_d \cdot \gamma}{4 \cdot V_1 \cdot EC_{50}} \tag{A.9}$$

Bibian [10] analyzed the induction of 44 patients with a single bolus administration of propofol and measured the WAV. The PK and PD parameters of this study are provided in Table A.3 and are used in Chapters 3, 4, and 5 as the model.

Table A.3: PK and PD parameters from the Bibian study [10].

PATIENT	DEMOGRAPHIC PARAMETERS				PD PARAMETERS			
	$Weight$ [Kg]	$Height$ [cm]	Age [yr]	$Gender$	T_d [s]	K_d [s^{-1}. 10^{-3}]	EC_{50} [µg/mL]	γ
1	100	178	21	M	22	133.5	3.2	4.7
2	59	168	28	F	4	44.4	3.1	2.5
3	90	190	26	M	44	25.0	2.4	1.9
4	53	157	21	F	45	51.5	3.8	1.2
5	90	185	19	M	39	85.7	3.8	2.3
6	60	162	28	F	18	82.5	3.9	2.1
7	78	170	24	M	32	44.4	2.9	2.8
8	68	164	19	F	12	26.7	1.9	2.3
9	70	170	25	M	7	35.2	3.4	1.9
10	81	180	23	M	9	32.8	2.8	2.8
11	83	178	18	M	17	46.4	2.8	2.3
12	67	163	21	F	4	26.2	2.4	2.5
13	88	183	22	M	9	50.4	2.5	2.6
14	59	162	21	M	18	160.5	3.6	3.9
15	72	176	19	M	20	75.0	2.5	1.9
16	73	180	34	M	44	54.8	3.2	2.7
17	71	178	34	M	29	83.1	4.0	2.3
18	91	189	35	M	18	34.4	3.7	2.1
19	85	180	38	M	1	29.6	3.3	1.2
20	86	168	33	F	1	24.9	3.1	1.5
21	75	176	38	M	12	35.2	3.9	1.8
22	91	178	39	M	4	24.8	2.7	2.0
23	78	178	34	M	12	28.7	2.8	2.2
24	65	163	36	F	5	27.0	2.8	2.1
25	58	157	34	F	4	67.2	3.6	2.0
26	77	180	33	M	12	29.3	3.1	1.8
27	105	185	39	M	13	29.1	3.7	1.8
28	66	168	46	F	11	36.6	6.1	1.3
29	99	182	45	M	2	32.6	4.7	1.3
30	77	173	48	M	12	35.0	4.5	1.4
31	98	189	47	M	10	28.7	3.9	2.0
32	79	172	46	M	12	34.8	3.9	1.8
33	96	193	40	M	9	36.6	3.2	1.9
34	77	176	40	M	8	35.6	3.4	2.3
35	97	190	48	M	13	30.0	3.0	1.9
36	80	177	53	M	35	38.0	3.3	1.8
37	63	165	60	F	3	31.5	3.5	2.3
38	91	185	52	M	29	42.0	4.4	2.2
39	100	182	52	M	2	21.8	4.7	1.4
40	97	176	52	M	16	28.8	3.7	1.1
41	95	183	50	M	10	26.4	4.0	1.9
42	56	173	52	F	6	58.0	5.0	1.5
43	100	180	54	M	6	32.2	4.2	1.5
44	95	176	60	M	12	24.3	3.1	1.8

Appendix B

Control Performance in Closed-Loop Anesthesia

The definition of Varvel and the proposed measures are discussed. In Figure B.1, an example of the DOH from a closed-loop control system is shown. The black line represents the induction phase, the blue represents the maintenance phase, and the magenta represents the emergence phase. Table B.1 displays the numerical value of all the discussed measures for reference.

B.1 Varvel Measures

The Varvel performance measures is constituted of 4 metrics: MDPE, MDAPE, *Divergence, Wobble* [65]. They are all based on the Percent Error (PE), defined as:

$$PE = 100\frac{C_m - C_p}{C_p}, \tag{B.1}$$

where C_m is the measured plasma concentration and C_p is the corresponding estimate of the C_m. Vectors could be in real-time (function of t) or in discrete (step unit of h). The four Varvel metric measurements are now defined. In the context of closed-room control, C_m is usually replaced with the measured DOH y and the C_p is replaced by the set-point r.

1. *Median Performance Error (*MDPE*)* measures the *bias* and is calculated as

the median of all the PE:

$$MDPE = median(PE). \tag{B.2}$$

2. *Median Performance Absolute Error (*MDAPE*)* measures the *inaccuracy* and is calculated as the median of the absolute of PE:

$$MDPE = median(|PE|). \tag{B.3}$$

3. *Divergence* measures whether the error is getting bigger or smaller as time progresses and is calculated as the slope of the linear regression of the absolute PE against time:

$$MDPE = \frac{t^T|PE| - n\bar{t}\overline{|PE|}}{\|t\|^2 - N\bar{t}^2}, \tag{B.4}$$

where N is the size of the signal; $\bar{t}$ is the mean of the signal; t^T is the transpose of the vector; and $\|t\|^2$ is the square of the norm defined as $t^T t$. A positive Divergence signals an unstable control system.

4. *Wobble* measures the *variability* of the estimator and is calculated as the median of the absolute difference between PE and $MDPE$:

$$MDPE = median(|PE - MDPE|). \tag{B.5}$$

In [39] a fifth parameter was introduced in an attempt to provide a single scalar score to the overall performance of an EEG-guided DOH control system. The Global Score (GS) is then defined as:

$$GS = \frac{MDPA + Wobble}{\text{fraction of time DOH} \in (40,60)}. \tag{B.6}$$

The interval $(40,60)$ for the DOH is clinically recommended [3] for maintenance phase of anesthesia.

B.2 Proposed Control Performance Measures

The proposed performance measures by Soltesz et al. [53] provide different metrics for the three temporal phases of anesthesia: *induction*, *maintenance*, and *emergence*.

A. Induction Phase Metrics

1. *Induction Phase Duration (*ID*)* is adopted from [39]. Traditionally, *ID* is defined as the time it takes from the beginning of administration of drug to the time the DOH falls and remains below 60 for a duration of 30 seconds. This definition only applies if the set-point (r) 50 is used. Instead, 60 is replaced by requiring the DOH value to be between $r \pm 10$ for 30 seconds.

2. *Percent Overshoot (*OS*)* is defined as:

$$OS = 100 \cdot min \frac{r - y}{E_0 - y}, \tag{B.7}$$

where r is the reference, y is the measured DOH and E_0 is the awake baseline DOH. Typical value of E_0 are in the range of $90 < E_0 < 100$. If E_0 is not available, then the value 100 can be used.

The maximum overshoot usually occurs after the end of the induction phase. The signals r and y are then taken as the signals from the start of induction to 10 minutes after the end of induction phase.

B. Maintenance Phase Metrics

1. *Integrated Error (*IE*)* is introduced to replace MDPE (or the *bias*) of the system. It is calculated using the trapezoid approximation rule:

$$IE = \sum_{k=1}^{N} \frac{t_{k+1} - t_k}{t_N - t_1} \cdot \frac{(r_{k+1} - y_{k+1}) - (r_k - y_k)}{2}, \tag{B.8}$$

where the signals are for the duration of the maintenance phase only. The quantity $t_N - t_1$ is the length of the maintenance phase; IE is normalized to the length of the maintenance phase.

2. *Integrated Absolute Error (*IAE*)* is introduced to replace MDAPE (or the *inaccuracy*) of the system. It is calculated using the trapezoid approximation rule:

$$IE = \sum_{k=1}^{N} \frac{t_{k+1} - t_k}{t_N - t_1} \cdot \frac{|r_{k+1} - y_{k+1}| - |r_k - y_k|}{2}. \tag{B.9}$$

3. *Variability Index (*VI*)* is introduced to replace Divergence of the system. It is calculated as the relative difference between IAE and IE:

$$VI = \frac{IAE - IE}{IAE}. \tag{B.10}$$

4. *Percentage of Time Outside Adequate Range* is calculated as the percentage of the time the signal y is outside of the adequate range. Adequate range is defined as $r \pm 10$ [39]. The sign of an error is of clinical importance. It is justified to provide two percentages, E^+ for the time when $r - y$ is more than $+10$, and E^- when it is less than -10.

C. Emergence Phase Metrics

1. *Emergence Phase Rise Time (*ER*)* is the time it takes for the DOH to exceed $r_1 + (1 - e^{-1})(E_0 + r_1)$, where r_1 is the set-point when the administration of the hypnotic drug was terminated. If the awake baseline E_0 is not available, then $E_0 = 100$ can be used as the default value.

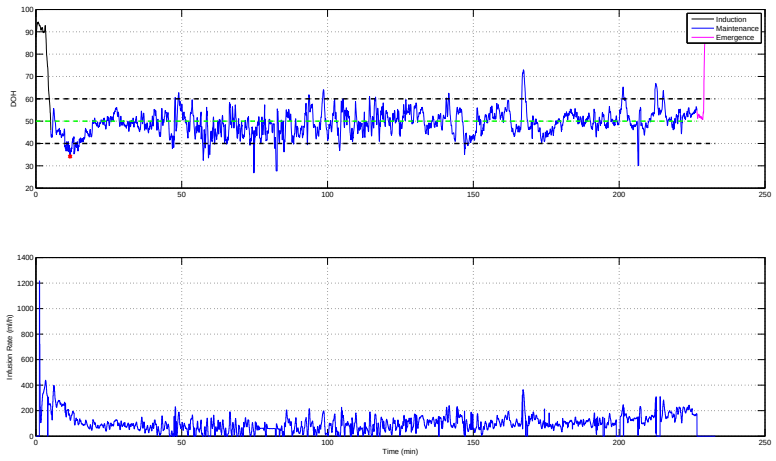


Figure B.1: Representative example from a closed-loop DOH control system. The induction phase is shown in solid black, the maintenance phase is shown in solid blue, the emergence phase is shown in solid magenta, the reference is shown in thick green, and the $r \pm 10$ bounds are shown in dashed black. The red dot represents the overshoot.

Table B.1: Varvel and proposed measures of the example from Figure B.1.

VARVEL MEASURES					PROPOSED MEASURES							
$MDPE$ [%]	$MDAPE$ [%]	$DIVERGENCE$ [%/min]	$WOBBLE$ [%]	GS	ID [min]	OS [%]	IE [%]	IAE [%]	VI	E^+ [%]	E^- [%]	ER [min]
-1.4	6.6	-0.0123	6.8	14.31	5.12	41.69	-0.789	4.0768	1.194	2.39	3.98	2.5

Appendix C

Limiting Behavior of $\mathcal{L}_1$ Adaptive Control

In Chapter 3, it was claimed that the closed-loop response of the system $G(s)$ approaches that of the reference system $G_{ref}(s)$ as $\Gamma \to \infty$. Moreover, it was claimed that the limiting controller is an implantable, non-adaptive LTI system. This implantable LTI system is independent of the system's unknown parameters ω, θ, σ. This Section will show the proof of these claims.

C.1 Problem Formulation and The $\mathcal{L}_1$ Adaptive Controller

This section will provide a summary of the $\mathcal{L}_1$-AC. A more detailed description of the control structure is available in Chapter 3.

Consider the following dynamics state-feedback controller $G(s)$ (see Chapter 2.2 of [26]):

$$\begin{aligned} \dot{x}(t) &= A_m x(t) + b(\omega(t)u(t) + \theta^T(t)x(t) + \sigma(t)), \\ y(t) &= c^T x(t), \end{aligned} \tag{C.1}$$

where $x(t) \in \mathbb{R}^n$ is the measured state of the system; $u(t) \in \mathbb{R}$ is the control input; $y(t) \in \mathbb{R}$ is the output; $b, c \in \mathbb{R}^n$ are assumed known constant vectors; A_m is a

$n \times n$ Hurwitz matrix corresponding to the desired closed-loop dynamics; $\omega \in \mathbb{R}$ is an unknown constant but with known sign; $\theta^T(t) \in \mathbb{R}^n$ is a vector of unknown parameters; and $\sigma(t) \in \mathbb{R}$ models input disturbances. The dynamics of the desired system $M(s)$ are given by:

$$\begin{aligned} \dot{x}_m(t) &= A_m x_m(t) - k_g b r(t), \quad x_m(0) = x_0, \\ y_m(t) &= c^T x_m(t) \end{aligned} \tag{C.2}$$

where $k_g \triangleq -1/(c^T A_m^{-1} b)$ and $r(t)$ is the reference.

The state predictor is given by:

$$\begin{aligned} \dot{\hat{x}}(t) &= A_m \hat{x}(t) + b(\hat{\omega}(t)u(t) + \hat{\theta}^T(t)x(t) + \hat{\sigma}(t)), \\ \hat{y}(t) &= c^T \hat{x}(t). \end{aligned} \tag{C.3}$$

The adaptation laws are given by:

$$\begin{aligned} \dot{\hat{\theta}}(t) &= -\Gamma \cdot \mathrm{Proj}\big(\hat{\theta}(t), -\tilde{x}^T P b x(t)\big), \quad \hat{\theta}(t) = \theta_0, \\ \dot{\hat{\sigma}}(t) &= -\Gamma \cdot \mathrm{Proj}\big(\hat{\sigma}(t), -\tilde{x}^T P b\big), \quad \hat{\sigma}(t) = \sigma_0, \\ \dot{\hat{\omega}}(t) &= -\Gamma \cdot \mathrm{Proj}\big(\hat{\omega}(t), -\tilde{x}^T P b u(t)\big), \quad \hat{\omega}(t) = \omega_0, \end{aligned} \tag{C.4}$$

where $\tilde{x}(t) = \hat{x}(t) - x(t)$, $\Gamma \in \mathbb{R}^+$ is the adaptation gain, and $P = P^T > 0$ is the solution of the algebraic Lyapunov equation $A_m^T P + P A_m = -Q$ for arbitrary $Q = Q^T > 0$.

The $\mathcal{L}_1$ control signal is defined as:

$$u(s) = -kD(s)(\hat{\eta}(s) - k_g r(s)), \tag{C.5}$$

where $r(s)$ and $\hat{\eta}(s)$ are the Laplace transforms of $r(t)$ and $\hat{\eta}(t)$ respectively and

$$\hat{\eta}(t) \triangleq \hat{\omega}(t)u(t) + \hat{\theta}^T(t)x(t) + \hat{\sigma}(t). \tag{C.6}$$

$k > 0$ is a feedback gain and $D(s)$ is a strictly proper transfer function such that

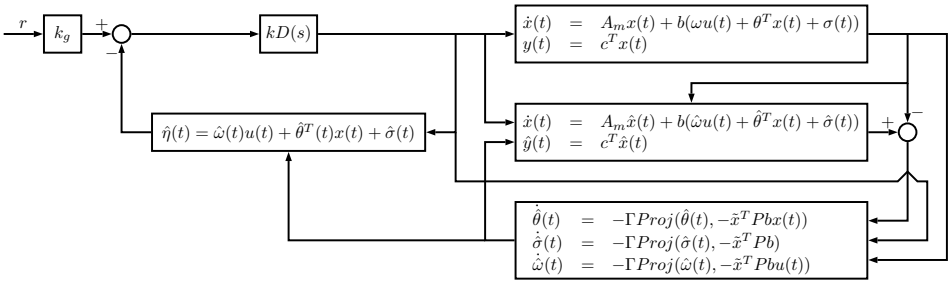


Figure C.1: $\mathcal{L}_1$-AC as formulated in [26].

they lead to a strictly proper stable filter $C(s)$:

$$C(s) = \frac{\omega kD(s)}{1 + \omega kD(s)}. \tag{C.7}$$

C.2 Removal of the Internal Feedback over $\hat{\eta}(t)$

Figure C.1 shows the $\mathcal{L}_1$-AC architecture as shown in [26]. The internal feedback over the signal $\hat{\eta}(t)$ is confusing and unnecessary. To analyze the limiting behavior of the system, this loop needs to be taken out. The multiplication of the system's adaptive parameters $\hat{\omega}$, $\hat{\theta}$, and $\hat{\sigma}$ with the state feedback $x(t)$ and the controller's output $u(t)$ is the nonlinearity that is present in the control architecture. Using the formulation of $\hat{\eta}(t)$ from the Figure C.1, it follows that this signal be defined as the output of the following dynamic system:

$$\begin{aligned}
\dot{\hat{\theta}}(t) &= -\Gamma Proj(\hat{\theta}(t), -\tilde{x}^T Pbx(t)),\\
\dot{\hat{\sigma}}(t) &= -\Gamma Proj(\hat{\sigma}(t), -\tilde{x}^T Pb),\\
\dot{\hat{\omega}}(t) &= -\Gamma Proj(\hat{\omega}(t), -\tilde{x}^T Pbu(t)),\\
\hat{\eta}(t) &= \hat{\omega}(t)u(t) + \hat{\theta}^T(t)x(t) + \hat{\sigma}(t).
\end{aligned} \tag{C.8}$$

The control architecture is a continuous system. The feedback signal $u(t)$ over $\hat{\eta}(t)$ from Figure C.1 is the same signal $u(t)$ that feeds into the adaptation laws block. The same is true for the signal $x(t)$ that feeds into the predictor and the adaptation laws block. It follows that $\hat{\eta}(t)$ is the only input signal to the predictor.

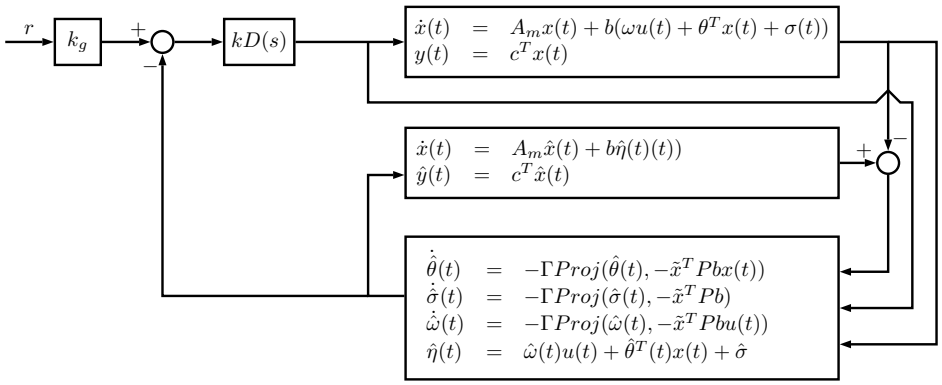


Figure C.2: Equivalent architecture of the $\mathcal{L}_1$-AC with removed internal feedback over $\hat{\eta}(t)$.

This leads to a more compact block diagram shown in Figure C.2. The state-space representation of the predictor can now be replaced by its single input transfer function representation $H(s)$ since $\hat{\eta}(t)$ is the only input, and $H(s) = (s\mathbb{I} - A_m)^{-1}b$. The same is true for the plant whose only input is $u(t)$, and the state-space representation of it can be replaced by its transfer function $G(s)$. The block diagram is further simplified to Figure C.3.

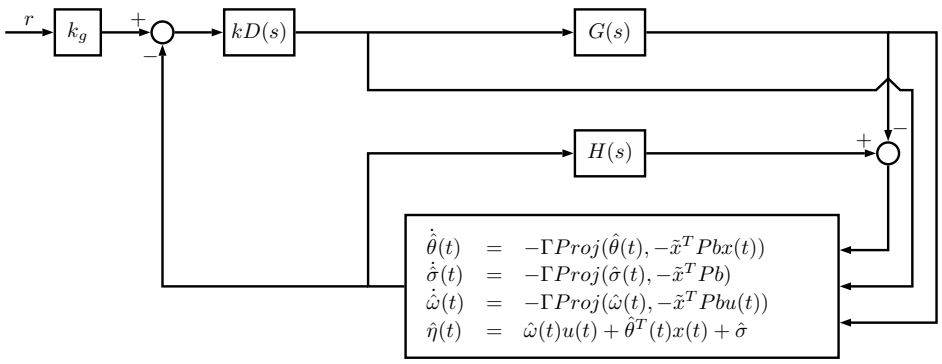


Figure C.3: Simplified architecture of the $\mathcal{L}_1$-AC to a more coherent structure.

C.3 Linearizing the $\mathcal{L}_1$ Controller with Generic Adaptation Laws

The $\mathcal{L}_1$-AC control structure creates an internal feedback loop in the controller. The forward loop consists of nonlinear functions with the adaptive gain Γ, and the feedback loop consists of the LTI function $H(s)$. High-gain feedback over a plant results in the approximate inversion of LTI function (see [23], Chapter 2.6). This inversion can be approximated with a linear model. The dynamic system in (C.8) is rewritten for generic adaptation laws and is shown in Figure C.4:

$$\begin{aligned}
\dot{\hat{\theta}}(t) &= \Gamma f_1(x,\hat{x}),\\
\dot{\hat{\sigma}}(t) &= \Gamma f_2(x,\hat{x}),\\
\dot{\hat{\omega}}(t) &= \Gamma f_3(x,\hat{x},u),\\
\hat{\eta}(t) &= g(\hat{\theta},\hat{\sigma},\hat{\omega},x,\hat{x},u).
\end{aligned} \tag{C.9}$$

Taking the gain Γ out of the function f_i clarifies the effect of increasing it. Let $\{x_Q,\hat{x}_Q,u_Q,\hat{\theta}_Q,\hat{\sigma}_Q,\hat{\omega}_Q,\hat{\eta}_Q;t\in\mathbb{R}\}$ correspond to any set of equilibrium points of the closed-loop system, i.e. $\dot{\hat{\theta}}(t) = f_1(x_Q,\hat{x}_Q) = 0$. Define:

$$\begin{aligned}
\Delta x(t) &= x(t) - x_Q,\\
\Delta \hat{x}(t) &= \hat{x}(t) - \hat{x}_Q,\\
\Delta u(t) &= u(t) - u_Q,\\
\Delta \hat{\theta}(t) &= \hat{\theta}(t) - \hat{\theta}_Q,\\
\Delta \hat{\sigma}(t) &= \hat{\sigma}(t) - \hat{\sigma}_Q,\\
\Delta \hat{\omega}(t) &= \hat{\omega}(t) - \hat{\omega}_Q,\\
\Delta \hat{\eta}(t) &= \hat{\eta}(t) - \hat{\eta}_Q,
\end{aligned} \tag{C.10}$$

where $\hat{\eta}_Q \triangleq g(\hat{\theta}_Q,\hat{\sigma}_Q,\hat{\omega}_Q,x_Q,\hat{x}_Q,u_Q)$.

The linearization of nonlinear adaptation laws in equation (C.9) in close vicin-

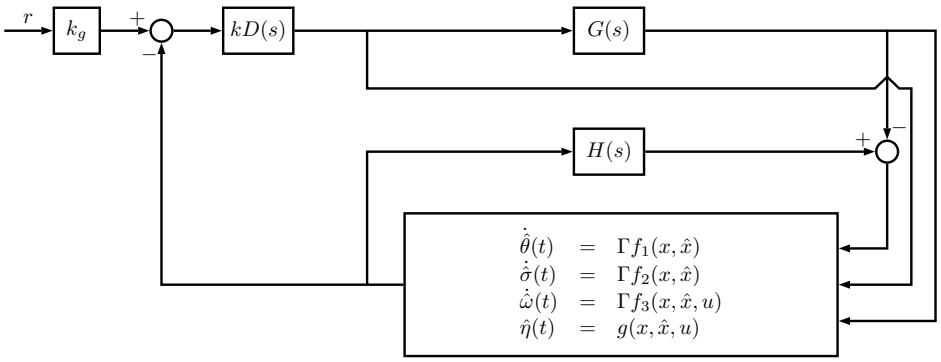


Figure C.4: Simplified architecture of the $\mathcal{L}_1$-AC with generic adaptive laws.

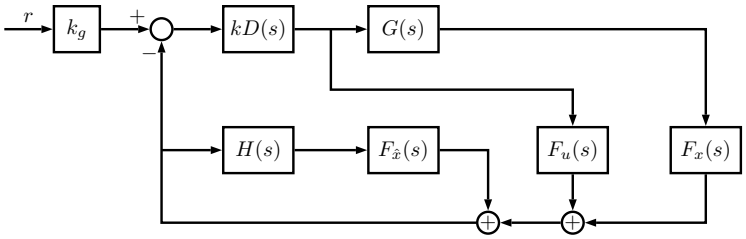


Figure C.5: Linearized $\mathcal{L}_1$-AC with generic adaptation laws.

ity of the equilibrium states are given by a first order Taylor series:

$$
\begin{aligned}
\Delta\dot{\hat{\theta}}(t) &= \Gamma\Big[\frac{\partial f_1}{\partial x}\Big|_{x_Q}\Delta x(t) + \frac{\partial f_1}{\partial \hat{x}}\Big|_{x_Q}\Delta\hat{x}(t)\Big],\\
\Delta\dot{\hat{\sigma}}(t) &= \Gamma\Big[\frac{\partial f_2}{\partial x}\Big|_{x_Q}\Delta x(t) + \frac{\partial f_2}{\partial \hat{x}}\Big|_{x_Q}\Delta\hat{x}(t)\Big],\\
\Delta\dot{\hat{\omega}}(t) &= \Gamma\Big[\frac{\partial f_3}{\partial x}\Big|_{\substack{x_Q\\u_Q}}\Delta x(t) + \frac{\partial f_3}{\partial \hat{x}}\Big|_{\substack{x_Q\\u_Q}}\Delta\hat{x}(t) + \frac{\partial f_3}{\partial u}\Big|_{\substack{x_Q\\u_Q}}\Delta u(t)\Big],\\
\Delta\hat{\eta}(t) &= \frac{\partial g}{\partial \hat{\theta}}\Big|_{\hat{\eta}_Q}\Delta\hat{\theta}(t) + \frac{\partial g}{\partial \hat{\sigma}}\Big|_{\hat{\eta}_Q}\Delta\hat{\sigma}(t) + \frac{\partial g}{\partial \hat{\omega}}\Big|_{\hat{\eta}_Q}\Delta\hat{\omega}(t)\\
&\quad + \frac{\partial g}{\partial x}\Big|_{\hat{\eta}_Q}\Delta x(t) + \frac{\partial g}{\partial u}\Big|_{\hat{\eta}_Q}\Delta u(t).
\end{aligned}
\tag{C.11}
$$

Let the notation F_{ix} represent the Laplace transform of $\frac{\partial f_{ix}}{\partial x}$. Then:

$$\begin{aligned}
\Delta\hat{\theta}(s) &= \Gamma F_{1x}\Delta x(s) + \Gamma F_{1\hat{x}}\Delta\hat{x}(s),\\
\Delta\hat{\sigma}(s) &= \Gamma F_{2x}\Delta x(s) + \Gamma F_{2\hat{x}}\Delta\hat{x}(s),\\
\Delta\hat{\omega}(s) &= \Gamma F_{3x}\Delta x(s) + \Gamma F_{3\hat{x}}\Delta\hat{x}(s) + \Gamma F_{3u}\Delta u(s),\\
\Delta\hat{\eta}(s) &= G_{\hat{\theta}}\Delta\hat{\theta}(s) + G_{\hat{\sigma}}\Delta\hat{\sigma}(s) + G_{\hat{\omega}}\Delta\hat{\omega}(s)\\
&\quad + G_x\Delta x(s) + G_u\Delta u(s),
\end{aligned} \tag{C.12}$$

where $G_{\hat{\theta}}$, $G_{\hat{\sigma}}$, $G_{\hat{\omega}}$, G_x, and G_u are the Laplace transform of $\frac{\partial g}{\partial\hat{\theta}}\Big|_{\hat{\eta}_Q}$, $\frac{\partial g}{\partial\hat{\sigma}}\Big|_{\hat{\eta}_Q}$, $\frac{\partial g}{\partial\hat{\omega}}\Big|_{\hat{\eta}_Q}$, $\frac{\partial g}{\partial x}\Big|_{\hat{\eta}_Q}$, and $\frac{\partial g}{\partial u}\Big|_{\hat{\eta}_Q}$ respectively. Substituting the intermediate signals $\Delta\hat{\theta}(s)$, $\Delta\hat{\sigma}(s)$, and $\Delta\hat{\omega}(s)$, the LTI system $\Delta\hat{\eta}(s)$ can now be written as:

$$\Delta\hat{\eta} = F_x(s)\Delta x(s) + F_{\hat{x}}(s)\Delta\hat{x}(s) + F_u(s)\Delta u(s), \tag{C.13}$$

where $F_x(s)$, $F_{\hat{x}}(s)$, and $F_u(s)$ are the linearized transfer functions between Δx, $\Delta\hat{x}$, Δu around the equilibrium points $x_Q, \hat{x}_Q, u_Q, \hat{\theta}_Q, \hat{\sigma}_Q, \hat{\omega}_Q, \hat{\eta}_Q$ and correspond to:

$$\begin{aligned}
F_x(s) &= \Gamma(G_{\hat{\theta}}F_{1x} + G_{\hat{\sigma}}F_{2x} + G_{\hat{\omega}}F_{3x}),\\
F_{\hat{x}}(s) &= \Gamma(G_{\hat{\theta}}F_{1\hat{x}} + G_{\hat{\sigma}}F_{2\hat{x}} + G_{\hat{\omega}}F_{3\hat{x}}),\\
F_u(s) &= \Gamma G_{\hat{\omega}}F_{3u} + G_u.
\end{aligned} \tag{C.14}$$

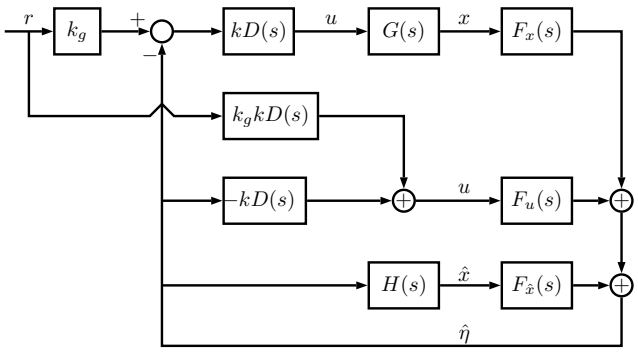


Figure C.6: Equivalent form of Figure C.5 of the linearized $\mathcal{L}_1$-AC with generic adaptation laws.

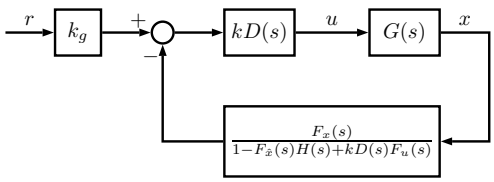


Figure C.7: Final form of the linearized adaptation laws for $\mathcal{L}_1$-AC with generic adaptation laws.

The block diagram for this linearized controller is shown in Figure C.5. This block diagram is equivalently shown in Figure C.6. This structure was realized by taking out the signal u that feeds to F_u out of the internal loop. It now follows that the feedback controller, i.e. the relation between Δx and Δu, is derived by realizing that $\Delta\hat{x} = H(s)\Delta\hat{\eta}$ and $\Delta u = -kD(s)\Delta\hat{\eta}$ and substituting $\Delta\hat{\eta}$ from C.13:

$$\Delta u(s) = K_{\Delta x}(s)\Delta x(s) = \frac{-F_x(s)kD(s)}{1 - F_{\hat{x}}(s)H(s) + F_u(s)kD(s)}\Delta x(s). \tag{C.15}$$

This linearized LTI controller corresponds to a two-degree of freedom LTI controller. The transfer function between $\Delta r(t)$ and $\Delta u(t)$ has been omitted for simplicity, but it can also be derived easily. Figure C.7 shows the block diagram for this LTI controller. The transfer functions F_x, $F_{\hat{x}}$, F_u can be replaced by their definition C.14; the gain Γ can then be taken to infinity to yield the limiting behavior for the $\mathcal{L}_1$ controller with generic adaptation laws.

In the next Section, the projection operator in the adaptation laws used in the Hovakimyan's implementation of the $\mathcal{L}_1$-AC is linearized and the limiting behavior is computed

C.4 Linearization of the Projection Operator in the $\mathcal{L}_1$ Adaptive Control

The previous Section, the $\mathcal{L}_1$-AC with generic adaptation laws, was linearized. In this Section, the case of projection operator for the adaptation laws is linearized.

Following the linearization definition C.11, the adaptation laws C.4 are linearized:

$$\begin{aligned}
\Delta\dot{\hat{\theta}}(t) &= \Gamma\Big[-\tilde{x}_Q(Pb)^T\Delta x(t) + x_Q(Pb)^T\Delta x(t) - x_Q(Pb)^T\Delta\hat{x}(t)\Big],\\
\Delta\dot{\hat{\sigma}}(t) &= \Gamma\Big[(Pb)^T\Delta x(t) - (Pb)^T\Delta\hat{x}(t)\Big],\\
\Delta\dot{\hat{\omega}}(t) &= \Gamma\Big[u_Q(Pb)^T\Delta x(t) - u_Q(Pb)^T\Delta\hat{x}(t) - \tilde{x}_Q(Pb)^T\Delta u(t)\Big],\\
\Delta\hat{\eta}(t) &= u_Q\Delta\hat{\omega}(t) + \omega_Q\Delta u(t) + \hat{\theta}_Q^T\Delta x(t) + x_Q^T\Delta\hat{\theta}(t) + \Delta\hat{\sigma}(t).
\end{aligned} \tag{C.16}$$

At the equilibrium points, it follows that $f_1(x_Q,\hat{x}_Q) = 0$, $f_2(x_Q,\hat{x}_Q) = 0$, and $f_3(x_Q,\hat{x}_Q,u_Q) = 0$. This leads to $x_Q = \hat{x}_Q$ and $\tilde{x}_Q = \hat{x}_Q - x_Q = 0$. The system above simplifies to:

$$\begin{aligned}
\Delta\dot{\hat{\theta}}(t) &= \Gamma\Big[x_Q(Pb)^T\Delta x(t) - x_Q(Pb)^T\Delta\hat{x}(t)\Big],\\
\Delta\dot{\hat{\sigma}}(t) &= \Gamma\Big[(Pb)^T\Delta x(t) - (Pb)^T\Delta\hat{x}(t)\Big],\\
\Delta\dot{\hat{\omega}}(t) &= \Gamma\Big[u_Q(Pb)^T\Delta x(t) - u_Q(Pb)^T\Delta\hat{x}(t)\Big],\\
\Delta\hat{\eta}(t) &= u_Q\Delta\hat{\omega}(t) + \omega_Q\Delta u(t) + \hat{\theta}_Q^T\Delta x(t) + x_Q^T\Delta\hat{\theta}(t) + \Delta\hat{\sigma}(t).
\end{aligned} \tag{C.17}$$

Assume the initial conditions of these differentials are all zero, i.e. $\Delta\hat{\theta}_0 = 0$, $\Delta\hat{\sigma}_0 = 0$, and $\Delta\hat{\omega}_0 = 0$. The system above can be written in the form of (C.13) as:

$$\begin{aligned}
\Delta\hat{\eta}(s) = &\Big[\frac{\Gamma}{s}(x_Q^Tx_Q + u_Q^2 + 1)(Pb)^T + \hat{\theta}_Q^T\Big]\Delta x(s)\\
&-\frac{\Gamma}{s}(x_Q^Tx_Q + u_Q^2 + 1)(Pb)^T\Delta\hat{x}(s) + w_Q\Delta u(s).
\end{aligned} \tag{C.18}$$

The transfer function between Δx and Δu is then written as:

$$\Delta\hat{u}(s) = -\frac{kD(s)\Big[\frac{\Gamma}{s}(x_Q^Tx_Q + u_Q^2 + 1)(Pb)^T + \hat{\theta}_Q^T\Big]}{1 + \frac{\Gamma}{s}(x_Q^Tx_Q + u_Q^2 + 1)(Pb)^TH(s) + kD(s)\omega_Q}\Delta x(s). \tag{C.19}$$

This controller is stable if and only if the $\mathcal{L}_1$-norm condition is satisfied and the limit of the controller as $\Gamma \to \infty$ exists. In this case, the limit of this controller

yields:

$$
\begin{aligned}
u_{lim}(s) &= \lim_{\Gamma\to\infty} -\frac{kD(s)\left[\frac{\Gamma}{s}(x_Q^T x_Q + u_Q^2 + 1)(Pb)^T + \hat{\theta}_Q^T\right]}{1+\frac{\Gamma}{s}(x_Q^T x_Q + u_Q^2 + 1)(Pb)^T H(s) + kD(s)\omega_Q} x(s) \\
&= \frac{-kD(s)(Pb)^T}{(Pb)^T H(s)} x(s).
\end{aligned}
\tag{C.20}
$$

Appendix D

Robustness and Performance of iControl

The following data are from the 44 simulation case studies of Chapter 5. The *robustness*, *output disturbance rejection*, and *set-point response* of the auto-tuned PID controller in response to a detected oscillation is compared to the current implementation of iControl. The PID *parameters* for both the auto-tuned controller and the iControl are also provided.

Table D.1: Robustness comparison of the iControl vs the auto-tuned algorithm of Chapter 5 for the 44 PKPD models.

PATIENT	ORIGINAL TUNING		AUTO-TUNED	
	A_m [-]	φ_m [degree]	A_m [-]	φ_m [degree]
1	6.62	60.10	6.40	54.22
2	9.52	64.24	9.36	58.93
3	4.72	50.27	4.96	52.14
4	7.94	52.93	7.89	51.37
5	7.06	54.56	6.96	48.88
6	8.31	60.75	8.12	56.09
7	6.05	56.86	6.05	50.29
8	7.37	57.48	7.39	50.16
9	10.75	60.60	10.61	54.92
10	8.72	60.29	8.62	53.89
11	7.35	58.08	7.23	51.67
12	11.27	59.21	11.23	52.85
13	6.77	63.15	6.60	56.57
14	7.55	60.83	7.33	55.90
15	5.44	60.87	5.30	54.08
16	4.81	58.13	4.75	57.27
17	6.27	61.20	6.13	55.58
18	8.00	60.58	7.94	54.50
19	10.99	65.08	10.79	59.26
20	14.41	61.42	14.27	56.09
21	9.26	62.14	9.15	56.59
22	9.57	63.90	9.48	57.26
23	7.55	62.19	7.51	55.34
24	10.27	63.23	10.21	57.16
25	10.13	64.71	9.74	61.14
26	8.24	61.57	8.19	55.20
27	9.51	60.94	9.44	54.98
28	14.65	58.53	14.48	55.05
29	14.64	62.52	14.35	58.13
30	10.11	62.30	9.98	57.62
31	9.74	62.85	9.64	57.20
32	9.03	63.22	8.91	57.76
33	7.57	65.21	7.43	58.89
34	8.68	64.58	8.54	58.90
35	6.78	64.82	6.71	57.95
36	5.08	61.58	5.05	60.05
37	10.28	67.18	10.16	62.07
38	6.89	61.14	6.86	55.71
39	17.36	61.01	17.21	56.28
40	8.42	62.39	8.38	56.55
41	10.45	62.35	10.37	56.65
42	9.99	65.19	9.76	61.37
43	11.00	64.21	10.79	59.50
44	7.88	64.41	7.86	57.82

Table D.2: Output disturbance rejection comparison of the iControl tuning vs the auto-tuned algorithm of Chapter 5 for the 44 PKPD models.

PATIENT	ORIGINAL TUNING			AUTO-TUNED		
	T_s [min]	$Maximum$ [WAV]	IAE [min]	T_s [min]	$Maximum$ [WAV]	IAE [-]
1	2.67	69.1	137.5	2.75	69.18	143.05
2	3.83	69.0	177.2	3.83	69.02	183.80
3	3.75	70.0	192.5	4.50	69.99	213.59
4	8.17	71.6	304.9	8.83	69.67	292.98
5	4.42	69.8	211.9	4.50	69.94	221.25
6	4.83	69.5	217.6	4.92	69.55	218.51
7	3.42	69.8	181.5	3.58	69.90	193.04
8	3.00	67.7	232.9	3.08	67.82	239.74
9	4.58	69.3	209.8	4.67	69.45	214.58
10	3.33	69.2	169.2	3.42	69.27	178.25
11	3.83	69.5	191.0	3.92	69.57	201.76
12	3.58	69.2	169.5	3.67	69.26	174.04
13	3.25	69.0	163.3	3.33	69.05	171.38
14	3.25	69.1	159.4	3.25	69.10	166.42
15	3.92	69.5	190.6	4.00	69.54	194.46
16	3.33	70.0	180.2	4.17	70.09	197.35
17	4.25	69.7	197.5	4.33	69.75	204.20
18	4.33	69.6	200.4	4.42	69.63	207.70
19	5.67	69.6	244.2	5.67	69.49	242.00
20	5.33	69.5	236.0	5.33	69.51	235.14
21	4.92	69.5	220.3	5.00	69.60	221.09
22	3.92	69.2	183.6	4.00	69.26	191.93
23	3.67	69.4	181.0	3.75	69.46	190.90
24	4.08	69.3	188.9	4.17	69.30	192.73
25	4.83	69.1	211.9	5.50	68.50	239.04
26	4.33	69.5	201.8	4.42	69.57	208.49
27	4.83	69.5	217.3	4.92	69.65	220.36
28	11.75	71.9	298.6	8.33	69.63	284.49
29	7.58	70.8	278.4	6.83	69.61	266.55
30	6.75	70.5	267.0	6.42	69.84	259.93
31	4.50	69.4	203.5	4.58	69.51	207.80
32	4.92	69.5	220.7	5.00	69.60	220.49
33	4.25	69.3	190.5	4.25	69.33	196.93
34	4.00	69.2	183.0	4.08	69.27	189.75
35	4.00	69.5	187.4	4.08	69.50	195.37
36	4.33	69.9	204.7	5.42	69.91	230.31
37	4.17	69.1	183.5	4.17	69.11	187.80
38	4.50	69.7	206.4	4.67	69.95	210.94
39	6.92	70.6	272.0	6.58	69.69	263.06
40	7.08	70.6	275.8	6.75	70.05	271.34
41	4.75	69.5	213.9	4.83	69.58	216.28
42	6.67	70.3	260.8	6.25	69.63	252.44
43	5.83	69.9	247.7	5.75	69.63	243.82
44	4.33	69.5	197.6	4.42	69.59	203.58

Table D.3: Set-point response comparison of the iControl tuning vs the auto-tuned algorithm of Chapter 5 for the 44 PKPD models.

PATIENT	ORIGINAL TUNING			AUTO-TUNED		
	T_r [min]	T_s [min]	$Overshoot$ [%]	T_r [min]	T_s [min]	$Overshoot$ [%]
1	5.0	21.1	5.27	5.08	19.92	6.54
2	8.3	34.4	5.20	7.75	32.08	5.84
3	8.0	32.8	6.36	10.67	41.00	6.64
4	24.4	50.0	4.77	29.17	50.00	3.01
5	10.6	41.3	6.57	9.75	38.33	7.63
6	13.5	48.9	5.53	12.33	45.83	6.25
7	6.9	29.5	5.93	6.75	27.33	6.99
8	3.8	18.9	3.00	4.33	19.42	3.28
9	12.0	45.4	5.73	11.00	42.42	6.52
10	6.5	27.8	5.51	6.33	25.92	6.45
11	8.1	33.8	6.08	7.58	31.42	6.99
12	7.4	30.4	5.50	7.25	28.42	6.45
13	6.3	27.3	5.26	6.17	25.50	6.12
14	6.3	27.1	5.24	6.17	25.17	6.07
15	8.8	36.0	5.75	8.25	33.50	6.49
16	7.1	30.2	5.64	9.50	38.00	6.20
17	10.6	41.3	5.52	9.75	38.58	6.32
18	10.6	41.6	5.62	9.75	38.75	6.46
19	17.5	50.0	4.90	16.08	50.00	5.48
20	15.4	50.0	5.13	14.25	50.00	5.58
21	13.8	50.0	5.35	12.67	47.00	6.08
22	8.6	35.8	5.24	8.00	33.25	5.97
23	7.9	33.4	5.36	7.42	31.00	6.19
24	9.4	38.3	5.35	8.83	35.58	5.93
25	13.6	49.1	5.12	18.08	50.00	5.09
26	10.8	42.3	5.57	9.92	39.42	6.35
27	13.2	48.4	5.49	12.08	45.33	6.29
28	26.3	50.0	3.11	26.75	50.00	3.71
29	22.6	50.0	4.32	21.33	50.00	4.95
30	20.7	50.0	4.62	19.33	50.00	5.24
31	11.8	44.8	5.30	10.83	41.83	6.02
32	14.2	50.0	5.19	13.00	47.75	5.88
33	10.1	40.2	5.18	9.25	37.50	5.87
34	9.2	37.3	5.14	8.50	34.75	5.81
35	9.4	38.3	5.14	8.75	35.75	5.87
36	11.3	43.8	5.41	15.42	50.00	5.49
37	10.2	40.0	4.83	9.33	37.42	5.38
38	12.0	45.1	5.39	11.00	42.17	6.16
39	21.2	50.0	4.61	19.83	50.00	5.27
40	22.4	50.0	4.39	21.08	50.00	5.02
41	13.2	48.2	5.29	12.00	45.17	6.03
42	20.4	50.0	4.42	19.08	50.00	4.95
43	18.2	50.0	4.75	16.75	50.00	5.33
44	11.0	42.7	5.15	10.08	40.00	5.86

Table D.4: PID Parameters of the iControl tuning and the auto-tuned algorithm of Chapter 5 for the 44 PKPD models.

PATIENT	ORIGINAL TUNING			AUTO-TUNED		
	k	k_i	k_d	k	k_i	k_d
1	1.547	0.01051	85.95	1.314	0.01156	92.68
2	1.079	0.00733	59.94	0.946	0.00806	63.72
3	1.566	0.01064	87.02	1.160	0.00798	95.72
4	0.924	0.00627	51.33	0.684	0.00471	56.47
5	1.525	0.01036	84.73	1.286	0.01139	91.99
6	1.025	0.00696	56.95	0.882	0.00766	60.75
7	1.306	0.00887	72.55	1.101	0.00975	78.42
8	1.103	0.00749	61.27	0.937	0.00824	65.71
9	1.242	0.00843	69.00	1.061	0.00928	74.09
10	1.412	0.00959	78.45	1.210	0.01055	84.41
11	1.412	0.00959	78.42	1.201	0.01054	84.53
12	1.086	0.00737	60.31	0.933	0.00811	64.45
13	1.493	0.01014	82.93	1.286	0.01115	89.27
14	1.088	0.00739	60.47	0.945	0.00813	64.39
15	1.307	0.00888	72.64	1.115	0.00977	78.10
16	1.348	0.00916	74.91	0.999	0.00687	82.40
17	1.316	0.00894	73.11	1.112	0.00983	78.86
18	1.566	0.01063	87.01	1.319	0.01170	94.24
19	1.527	0.01037	84.81	1.327	0.01140	91.27
20	1.273	0.00864	70.72	1.107	0.00951	75.69
21	1.331	0.00904	73.97	1.130	0.00994	79.70
22	1.475	0.01002	81.97	1.264	0.01102	88.28
23	1.372	0.00931	76.21	1.163	0.01025	82.18
24	1.071	0.00727	59.51	0.921	0.00800	63.57
25	0.960	0.00652	53.33	0.711	0.00489	58.66
26	1.380	0.00937	76.68	1.178	0.01031	82.56
27	1.645	0.01117	91.38	1.383	0.01228	99.06
28	1.129	0.00767	62.73	0.960	0.00843	67.32
29	1.572	0.01068	87.34	1.356	0.01174	94.13
30	1.323	0.00898	73.48	1.135	0.00988	78.95
31	1.622	0.01101	90.11	1.381	0.01211	97.35
32	1.330	0.00903	73.90	1.129	0.00994	79.64
33	1.639	0.01113	91.05	1.402	0.01224	98.28
34	1.347	0.00915	74.85	1.163	0.01006	80.38
35	1.622	0.01101	90.12	1.373	0.01212	97.51
36	1.379	0.00937	76.64	1.022	0.00703	84.30
37	1.077	0.00731	59.84	0.937	0.00805	63.73
38	1.533	0.01041	85.17	1.291	0.01145	92.22
39	1.580	0.01073	87.79	1.347	0.01180	94.79
40	1.507	0.01023	83.71	1.272	0.01125	90.56
41	1.549	0.01051	86.03	1.304	0.01157	93.16
42	1.108	0.00752	61.56	0.973	0.00828	65.53
43	1.564	0.01062	86.87	1.357	0.01168	93.55
44	1.491	0.01012	82.82	1.261	0.01114	89.52

Appendix E

Oscillation Detection MATLAB

OscillationDetectionAlgorithm

```
% Input's order of signals:
%    1. Output WAV
%    2. Time (second). Will convert to min; oscillation measured in minutes
function p_m_f = OscillationDetectionAlgorithm(input)

% We require to save all the output y
% So we can go over it and detect the oscillation when enough data points
% are found
persistent signal;
global toRunParam;

% Reset on time = 0;
if input(2) == 0
    signal.Data = [];
    signal.Time = [];
end

% Store the time and output y
y = input(1);
signal.Data = [signal.Data;y];
signal.Time = [signal.Time;input(2)/60];

% This will run the patient case in "real time" to determine all
% oscillatory components of the signal.
% Please set all parameters in the next section.
% Time is in minutes!
%% Test Parameters %%
test.period = 2.5;                  % Period Test
```

```
test.magnitude = test.period/1.1;   % Magntinude Test
test.fitness = 25;                  % Fitness Test (%)
%% Define the Parameters
toRunParam.testParameters = test;
toRunParam.windowPeriod = [1:30];   % Period ll/ul bound (in minute)
toRunParam.periodInc = 2;           % The increment in period from LL to UL
toRunParam.windowSize = 10;         % Number of periods to be analyzed

toRunParam.mergePeriod = 0.5;       % The difference between two consecutive periods to be considered same
toRunParam.mergeTime = 10;          % The difference between two consecutive time to be considered same
%% Loop through all requested oscillation bounds to detect oscillation
% We need at least 2 data points to calculate sampling time (T_2 - T_1)
p_m_f = [0 0 0];
if (length(signal.Time) > 1)
    for windowPeriod=toRunParam.windowPeriod(1):toRunParam.periodInc:toRunParam.windowPeriod(2)
        r = segment_Oscillation(windowPeriod, signal);
        if sum(abs(r)) ~= 0
            p_m_f = r;
        end
    end
end

end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%% Segment Oscillation
function p_m_f = segment_Oscillation(windowPeriod, signal)
% This will find ALL oscillations upto current time
% Oscillations need to be merged after to remove repetition
% NOTE: We will only go up until current time. If an oscillation is missed,
% it will no longer be detected.
% WindowPeriod is lower bound of the oscilation that we need to detect. The
% upper bound is +toRunParam.periodInc

global toRunParam sampleSet patientModel;
persistent lastSet;

if isempty(lastSet)
    lastSet = -1;
end
```

```
73
74  Ts = signal.Time(2) - signal.Time(1);
75  oscillationSignals = [];
76  periodInIndexLL = round(windowPeriod/Ts);
77  periodInIndexUL = round((windowPeriod+toRunParam.periodInc)/Ts);
78
79  iEnd = length(signal.Time);
80  iStart = iEnd - periodInIndexLL * toRunParam.windowSize;
81
82  p_m_f = [0 0 0];
83
84 % Enough data point to analze
85  if iStart > 0
86
87      sPartial = [];
88      sPartial.Time = signal.Time(iStart:iEnd);
89      sPartial.Data = signal.Data(iStart:iEnd);
90
91     % This runs the ODA and finds the oscillatory pair
92      OscillationPair = ODA(sPartial, windowPeriod);
93
94     % An oscillation was obsorved
95      if (~isempty(OscillationPair))
96          test = OscillationPair.Fitness > toRunParam.testParameters.fitness;
97          rangeLL = windowPeriod <= OscillationPair.Period;
98          rangeUL = OscillationPair.Period <  windowPeriod + toRunParam.periodInc;
99
100         % The regulartory tests are passed
101         % The period is within the LL & UL range
102         if (test && rangeLL && rangeUL)
103             % Save the start/end time indices
104             % And save the data to the master signal
105             OscillationPair.iStart = iStart;
106             OscillationPair.iEnd = iEnd;
107
108             % Return the result
109             p_m_f = [OscillationPair.Period OscillationPair.Magnitude
                        OscillationPair.Fitness];
110
111             %disp([OscillationPair.Period OscillationPair.Magnitude
                        OscillationPair.Fitness]);
112             % New patient, need to create a blank struct
113             if (lastSet ~= sampleSet)
114                 lastSet = sampleSet;
115
116
```

```
                disp([OscillationPair.Period OscillationPair.Magnitude
                    OscillationPair.Fitness]);


                % Save this instance of the model for future use
                load('OscillatoryModels.mat');
                writeVar = strcat('ModifiedPatient_', num2str(patientModel));
                patientVar = strcat('PKPDPatient_', num2str(patientModel));
                eval(['global ' patientVar ';']);
                readVar = strcat(patientVar, '(', num2str(sampleSet),')');



                if ~exist(writeVar)
                    eval([writeVar ' = [];']);
                end
                eval([writeVar '(end+1).tf = ' readVar '.tf;']);
                eval([writeVar '(end).gamma = ' readVar '.gamma;']);
                eval([writeVar '(end).E0 = ' readVar '.E0;']);
                eval([writeVar '(end).bwt = ' readVar '.bwt;']);
                eval([writeVar '(end).age = ' readVar '.age;']);
                eval([writeVar '(end).bht = ' readVar '.bht;']);
                eval([writeVar '(end).gdr = ' readVar '.gdr;']);
                eval([writeVar '(end).study = ' readVar '.study;']);
                eval([writeVar '(end).PKtype = ' readVar '.PKtype;']);
                eval([writeVar '(end).Td = ' readVar '.Td;']);
                eval([writeVar '(end).EC50 = ' readVar '.EC50;']);
                eval([writeVar '(end).Kd = ' readVar '.Kd;']);

                save('OscillatoryModels.mat', writeVar, '-append');
            end
        end
    end
end
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%% ODA %%
function oscillatoryPair = ODA(signal, windowPeriod)
global toRunParam;
% Computes the oscillatory pair
% First it detects a high/low pair
% Then it will perform the test to determine if it is oscillatory.
```

```
163 % This will return the highest fitness as the main oscillation
164
165 Ts = signal.Time(2) - signal.Time(1);
166 N = length(signal.Time);
167
168 x = signal.Data - mean(signal.Data);
169 y = dct(x);
170
171 %% Get the SL Components
172 Sy = std(y);
173 yh = seaLevel(3*Sy, y);
174 yl = seaLevel(Sy, y);
175 Yi = ithDCT(yh);
176 Yj = ithDCT(yl);
177 Xi = idct(Yi);
178 Xj = idct(Yj);
179 [dump,I] = size(Yi);
180 [dump,J] = size(Yj);
181
182 %% Find the pairs of xi and xj that match up
183 maxFitness = -Inf;
184 maxPair = null(1);
185 for i=1:I
186     xi = Xi(:,i);
187     yi = Yi(:,i);
188     mi = max(abs(yi));
189
190     high = null(1);
191     low = null(1);
192     for j=1:J
193         xj = Xj(:,j);
194         yj = Yj(:,j);
195         mj = max(abs(yj));
196
197         if( mi == mj )
198
199             high = generateSignal(signal.Time, xi, x);
200             low = generateSignal(signal.Time, xj, x);
201             break;
202         end
203     end
204
205     % Now perform the tests to see if this is oscillatory
206     if ~isempty(low)
207
208         % Only test for the oscillations that are within the limit
209         rangeHighLL = windowPeriod <= high.period.mean;
```

```
210         rangeHighUL = high.period.mean <  windowPeriod + toRunParam.periodInc;
211         rangeLowLL = windowPeriod <= low.period.mean;
212         rangeLowUL = low.period.mean <  windowPeriod + toRunParam.periodInc;
213
214         if (rangeHighLL && rangeHighUL && rangeLowLL && rangeLowUL)
215             hTest = high.period.test >= toRunParam.testParameters.period;
216             lTest = low.period.test >= toRunParam.testParameters.period;
217             if (hTest && lTest)
218
219                 % We only want the maximum fitness value.
220                 % So only select this pair if the fitness is the highest value.
221                 if (low.fitness > maxFitness)
222                     pair = [];
223                     pair.high = high;
224                     pair.low = low;
225
226                     maxFitness = low.fitness;
227                     maxPair = pair;
228                 end
229             end
230         end
231     end
232 end
233
234 % If a maximum pair was found, then perform the magnitude test.
235 % We need to determine which component (high or low) to use for the
236 % period/magnitude.
237 % Whichever pair has the higher periodic regulatory value, will then be
238 % selected as the candidate.
239 oscillatoryPair = null(1);
240 if (~isempty(maxPair))
241     high = maxPair.high;
242     low = maxPair.low;
243
244     % Use the low component
245     if (low.period.test > high.period.test)
246         prd = low.period;
247         mag = low.magnitude;
248     % Use the high component
249     else
250         prd = high.period;
251         mag = high.magnitude;
252     end
253
254     % Final test: magnitude regulator test must also be satisfied.
255     mTest = mag.test >= toRunParam.testParameters.magnitude;
256     if (mTest)
```

```
257             oscillatoryPair = maxPair;
258             oscillatoryPair.Fitness = low.fitness;
259             oscillatoryPair.Magnitude = mag.mean;
260             oscillatoryPair.Period = prd.mean;
261         end
262     end
263     end
264
265
266
267
268
269
270
271
272
273
274
275
276
277     %% Sea Level %%
278     function suppressed = seaLevel(SL, func)
279     % This function supresses the values below Sy and returns a vector of
280     % same dimention as y, but with supressed values.
281         N = length(func);
282         tmp = zeros(1,N);
283         index = find(abs(func) >= SL);
284         tmp(index) = func(index);
285         suppressed = tmp;
286     end
287
288     %% ith Discrete Cosine Transform
289     function output = ithDCT(yf)
290     %   Generates the ith DCT component of the vector subject to the following
291     %   criteria:
292     %       yi(k) = yf,i(k) for ks,i <= k <= ke,i ; otherwise 0
293     %           where
294     %           yf(ks,i) != 0 && yf(ks,i-r) = 0 for r = 1
295     %           yf(ke,i) ~= 0 && yf(ke,i+r) = 0 for r = 1,2,3,4
296     %           ks,i <= ke,i
297     %   It returns a matrix of k by N where k is the number of segments that
298     %   match the criteria.
299         N = length(yf);
300         Yi = [];
301         s = 2;
302         while( s ~= N )
303             if( yf(s) ~= 0 && yf(s-1) == 0 )
```

```
304             for e=s:N-4
305                 % match found. Find iDCT for the ith component
306                 if( yf(e) ~= 0 && length(find(yf(e+1:e+4) == 0)) == 4 )
307                     yi = zeros(N,1);
308                     yi(s:e) = yf(s:e);
309                     Yi = [Yi yi];
310                     s = e + 1;
311                     break;
312                 end
313             end
314         else
315             s = s+1;
316         end
317     end
318     output = Yi;
319 end
320
321 %% Period Sequence
322 function T = periodSequence(time, func)
323 % Calculates the period sequence from the original signal
324     z = zeroCrossingSequence(time, func);
325     L = length(z);
326     period = [];
327     for l=1:L-1
328         period(l) = 2*(z(l+1)-z(l));
329     end
330     T = period;
331 end
332
333 %% Period Test
334 function R = periodTest(periodSequence)
335 % Calculates the period of a partial (iDCT).
336 % Based on the work Wang 2013
337     alpha = 0.0027;
338     N  = length(periodSequence);
339     N = 8;
340     CV = std(periodSequence)/mean(periodSequence);
341     x  = chi2inv(1-alpha/2,N-1);                    % We have df = L-1, and L =
            N+1
342     f  = sqrt(x/(N-1));
343     R  = f/CV;
344 end
345
346 %% Magnitude Sequence
347 function magnitude = magnitudeSequence(Ts, func, period)
348 % Calculates the Fitness of a partial (iDCT).
349 % Based on the work Wang 2013
```

```
350     N = length(func);
351     magnitude = [];
352     Interval = round(period/Ts);
353     for l=1:Interval:N-Interval
354         m = max(func(l:l+Interval)) - min(func(l:l+Interval));
355         magnitude = [magnitude m];
356     end
357 end
358
359 %% Magnitude Test
360 function R = magnitudeTest(magnitudeSequence)
361     %   Calculates the modified regulator index.
362     %   R value > 2.73 denotes an oscillation.
363     alpha = 0.0027;
364     N = length(magnitudeSequence);
365     CV = std(magnitudeSequence)/mean(magnitudeSequence);
366     x = chi2inv(1-alpha/2,N-1);                    % We have df = L-1, and L =
            N
367     R = sqrt(x)/(sqrt(N-1)*CV);
368 end
369
370 %% Zero Crossing Sequence
371 function z = zeroCrossingSequence(time, func)
372 % Calculates the zero-crossing of a function.
373     N = length(func);
374     z = time(find( func(1:N-1).* func(2:N) < 0 ));
375 end
376
377 %% Fitness Test
378 function F = fitnessTest(partial, x)
379 % Calculates the Fitness of a partial (iDCT).
380 % Based on the work Wang 2013
381     F = 100*(1-norm(partial - x)/norm(x));
382 end
383
384
385 %% Generates a signal with all the needed components
386 function signal = generateSignal(time, partial, x)
387 % Characterizes the signal by defining
388 %       signal.x                % time-domain signal
389 %       signal.y                % DCT signal
390 %       signal.time
391 %       signal.maxDct
392 %       signal.magnitude            % time-domain signal
393 %           signal.magnitude.signal
394 %           signal.magnitude.mean
395 %           signal.magnitude.std
```

```
396 %       signal.period               % time-domain signal
397 %           signal.period.signal    % time-domain signal
398 %           signal.period.mean      % time-domain signal
399 %           signal.period.std       % time-domain signal
400 %       signal.zeroCrossing         % time-domain signal
401 %       signal.index
402 %           signal.index.p
403 %           signal.index.m
404 %           signal.index.f
405
406
407     period = [];
408     period.signal = periodSequence(time, partial);
409     period.mean = mean(period.signal);
410     period.std = std(period.signal);
411     period.test = periodTest(period.signal);
412
413
414
415     magnitude = [];
416     magnitude.signal = [];
417     magnitude.mean = [];
418     magnitude.std = [];
419     magnitude.test = [];
420     if( ~isnan(period.mean) )
421         magnitude.signal = magnitudeSequence((time(2)-time(1)), partial, period.
                mean)/2;
422         magnitude.mean = mean(magnitude.signal);
423         magnitude.std = std(magnitude.signal);
424         magnitude.test = magnitudeTest(magnitude.signal);
425     end
426
427     signal.period = period;
428     signal.magnitude = magnitude;
429     signal.fitness = fitnessTest(partial, x);
430 end
```

Appendix F

PID Tuning Algorithm

PID Controller

```
function u_r_uP_uI_uD_v  = PIDforSwitching(signals)
% Two DOF PID - same implementation as in iControl
% Strange anti windup implementation ...
% Outputs the infusion rate in ml/hr


persistent I y1 y2 r1 ysp1
persistent K a dKick dKickmag ra1 rb1 rb2 K0 Ki0 Kd0;
persistent unstableParams switchedToUnstable;

period    = signals(1);
magnitude = signals(2);
fitness   = signals(3);

ysp = signals(4);
y   = signals(5);
ub  = signals(6);
lb  = signals(7);

inductionComplete2min = signals(8);

r = ysp;
h = 5;

% Initialization
if isempty(K)
    load PIDparams.mat K a dKick ra1 rb1 rb2 unstable

```

```
29      % Keep the instance of the original stable values for later comparison
30      K0 = K;
31      Ki0 = a(3);
32      Kd0 = a(4);
33
34      I    = 0;
35      y1   = y;
36      y2   = 0;
37      r1   = y;
38      ysp1 = y;
39      r    = y;
40      ysp  = y;
41
42      % Unstable parameter switch
43      switchedToUnstable = false;
44      unstableParams = unstable;
45  end
46  if dKick > 1
47      dKickmag = dKick;
48      dKick    = 0.5;
49  end
50  if dKick == 0.5;
51      if abs(ysp - r1) > 0.01
52          dKick = 0;
53          y2    = dKickmag;
54      end
55  end
56
57  % reference filter
58  r    = -ra1*r1+rb1*ysp+rb2*ysp1;
59  r1   = r;
60  ysp1 = ysp;
61
62  % Measurement filter
63  y2 = a(1)*y2 + a(2)*(y-y1);
64  y1 = y1 + y2;
65
66  % Wait until induction is complete for 2min,
67  % Then switch to unstable
68  if (inductionComplete2min && ~switchedToUnstable)
69      switchedToUnstable = true;
70      K    = unstableParams(1);
71      a(3) = unstableParams(2);
72      a(4) = unstableParams(3);
73  end
74
75  % CLP Dec 2 2009: Increased Cp limit from 7 to 8
```

```
% if Cp>8, I=0;end  % July 25th 2012, Klaske: This needs to be taken out!!!
uP = K*(r-y1);
uI = I;
uD = -a(4)*y2;

% July 25th 2012, Klaske: Use unfiltered y for proportional error?
v = uP + uI + uD;
u = v;

% Upper/lower limit
if v < lb
    u = lb;
    display('Lower Bound');


end
if v > ub
    u = ub;
    display('Upper Bound');

end

I= I + a(3)*(r-y1) + a(5)*(u-v);

% Oscillation Detected
% Retune controller
if period > 0
    w = 2*pi /(period * 60);

    % Describing Function magnitude
    delta = (ub - lb)/2;
    vv = v - delta;
    if abs(vv) <= delta
        N = 1;
    else
        alpha = asin(delta/abs(vv));
        N = 1/pi*(2*alpha + sin(2*alpha));
    end

    % New Stable point
    Ms = 1.3;
    r_s = (Ms-1)/Ms;
    phi_s = 2*asin(1/(2*Ms));

    % Current PID params
    P0 = -K;
    I0 = -a(3)/h;
```

```
123     D0 = -a(4)*h;
124     init_state = [P0 I0 D0];
125
126     % This calculates what the PID is at this state!
127     [dump, out] = PIDTuningNLConstraints(init_state, w, 5, [0 0]);
128     r_p   = 1/(out(1)*N);
129     phi_p = - out(2);
130
131     r_c = r_s / r_p;
132     phi_c = (phi_s - phi_p);
133
134     options = optimset('MaxFunEvals', 1e10, 'MaxIter', 1e3, 'Algorithm', '
            interior-point');
135     NLC = @(arg) PIDTuningNLConstraints(arg, w, 5, [r_c phi_c]);
136     result = fmincon(@PIDTuningObjective, init_state, [], [], [], [],[0.5 0.01 h
            ],[10 0.05 200], NLC, options)
137
138     disp(result);
139     pause
140 end
141
142 disp('-------');
143
144 % assemble output vector
145 u_r_uP_uI_uD_v = [u r uP I uD v y1 y2];
146
147 end
```

PIDTuningNLConstraints

```
1  function [c, ceq] = PIDTuningNLConstraints(arg, w, N, condition)
2  %function [c, ceq] = PIDTuningNLConstraints(arg)
3  %w = 2*pi/(38);
4  %N = 5;
5  %condition = [2.8982 4.0464];
6  % This will return the inequality (c) and equality(ceq) constraints
7  %
8  % The input args are Kp, Ki, Kd parameters
9  % We will redefine K, Ti, Td to work with
10 % The PID solving is of the form:
11 % U(s) = K(1 + 1/(Ti * s) + H(s) * Td * s)
12 % where
13 % H(s) = 1/(1 + Kd/(Kp * N) * s) = Kp * N/(Kp * N + Kd * s)
14
15 Kp = arg(1);
16 Ki = arg(2);
17 Kd = arg(3);
```

```
18
19 K = Kp;
20 Ti = Kp/Ki;
21 Td = Kd/Kp;
22
23 alpha = N*w^2*Td^2 / (N^2+w^2*Td^2);
24 beta = w*Td*N^2 / (N^2+w^2*Td^2);
25
26 rPart = 1+alpha;
27 iPart = beta - 1/(w*Ti);
28
29 cmp = rPart + iPart*sqrt(-1);
30
31 gain  = K*abs(cmp);
32 phase = angle(cmp);
33
34 c = [];
35 ceq(1) = gain - condition(1);
36 ceq(2) = phase - condition(2);
37 end
```