Automatic tuning of a MIMO PI controller of a flotation bank

by

Albertus Viljoen Richter

Submitted in partial fulfillment of the requirements for the degree

Master of Engineering (Electronic Engineering)

in the

Department of Electrical, Electronic and Computer Engineering

Faculty of Engineering, Built Environment and Information Technology

UNIVERSITY OF PRETORIA

November 2024

SUMMARY

Automatic tuning of a MIMO PI controller of a flotation bank

by

Albertus Viljoen Richter

Supervisor:	Prof J.D le Roux
Co-supervisor:	Prof I.K Craig
Department:	Electrical, Electronic and Computer Engineering
University:	University of Pretoria
Degree:	Master of Engineering (Electronic Engineering)
Keywords:	automatic tuning, Bayesian optimization, flotation, level control, proportional integral derivative

The literature on the automatic tuning of PID controllers is surveyed. The automatic tuning methods are sorted into model based and model-free methods. The methods are further subdivided into the manner the system is perturbed.

The method of Bayesian optimization is presented and discussed within the context of automatic controller tuning. The method used to constrain the Bayesian optimization is presented. The level flotation model is given and linearized. The controllers are given and discussed. The controller tuning strategies for both SISO and MIMO controllers are presented.

A Bayesian optimization automatic tuner is implemented on SISO and MIMO PI controllers used to control the pulp levels in a flotation bank. The implemented automatic tuner achieves performance improvement for both SISO and MIMO cases without any noise present. The MIMO controller tuning is also implemented on a system with measurement noise present and the Bayesian optimization automatic tuner settings performed on par with a state of the art forward-feeding controller.

The Bayesian optimization automatic tuner is constrained to ensure safety and stability. The constraints are found using a structured singular value analysis.

Acknowledgements

Various people have guided, mentored and supported me on my journey to where I am. From family and friends to teachers, I would not be here if it weren't for you!

Most directly with this work, I would like to thank Prof. Derik Le Roux and Prof. Ian Craig for their continued input and support. I appreciate the value you have added to this work.

To my father, Stephan, my mother, Marinda, grandmother, Stefanie and two brothers thank you for all the love and support I have enjoyed from you my whole life. Without your support on all levels of life, I could not have been here.

To the friends I made throughout this journey, you have made the toughest of times bearable with a smile.

To the One who gave me all I have, including these people whom have played such significant roles in my life I give the highest of praise. Thank you Lord Jesus Christ.

Psalm 32:8

"I will instruct you and teach you in the way you should go; I will counsel you with my eye upon you."-ESV Bible.

LIST OF ABBREVIATIONS

ARD	automatic relevance determination
BO	Bayesian optimization
EI	expected improvement
FF	forward feeding
FOPTD	first-order-plus-time-delay
GP	Gaussian process
IAE	integrated absolute error
IMC	internal model control
ISE	integrated squared error
ITAE	integrated time absolute error
MIMO	multiple-input-multiple-output
PID	proportional-integral-derivative
PI	proportional-integral
RS	robust stability
SIMC	Skogestad internal model control
SISO	single-input-single-output.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1 INTRODUCTION

1.1 PROBLEM STATEMENT

1.1.1 Context of the problem

In the field of process control multiple layers of control are encountered in the automation hierarchy. The hierarchy starts at the instrumentation of a plant followed by base layer control, then supervisory control, and finally optimization of the system as a whole (Shean and Cilliers, 2011). In the field of mineral processing the instrumentation is mostly related to valves and pumps (Shean and Cilliers, 2011). A major process in mineral processing is froth flotation. The base layer control for this process is mostly slurry or pulp level control, aeration control, and mass through-put control (Shean and Cilliers, 2011). Proportional-integral-derivative (PID) controllers are mainly utilized in the base layer of control, and sometimes in higher layers (Skogestad, 2023; Gous et al., 2023). Higher layers of control can only be utilized once base layers are successfully implemented. The aim of the base layer is to stabilize the closed-loop system and minimize deviations from set operating points (Shean and Cilliers, 2011). Regular tuning of the PID controllers is required to keep the PID settings optimal, as the process dynamics change over time.

The pulp level control of a flotation plant provides an intricate problem as the tanks (also called cells) are usually highly coupled (Jämsä-Jounela et al., 2003; Kämpjärvi and Jämsä-Jounela, 2003). The tuning of the base layer controllers regulating the pulp levels is critical and is complicated by the interactions between the cells. The flotation banks usually consist of a high number of interconnected cells, thus using a multivariable controller will require a large tuning effort with many parameters to consider. Generally, single-input-single-output (SISO) PID controllers are used to control the pulp level for each cell.

The main goal of flotation level control is to keep the actual pulp level as close to the desired level

(level setpoint) at all times. Thus, both setpoint tracking and disturbance rejection is of importance. The main expected disturbances are disturbances to the feed flow rate as well as volume disturbances to individual cells. Setpoint changes in other cells will cause a feed flow rate disturbance in downstream cells, and thus setpoint changes may also act as disturbances.

PID controllers can be tuned through various strategies (Somefun et al., 2021). The tuning of a controller can be improved by means of adaptive techniques such as gain scheduling, automatic tuning and adaptive control (Åström et al., 1993). Automatic tuning can broadly be done in two steps: system identification and controller design (Åström et al., 1993). From these steps two main strategies arise: model-based tuning and model-free or data driven tuning (Åström et al., 1993; Somefun et al., 2021).

In model-based tuning an explicit model of the controlled process is obtained through a system identification (SID) method. The tuning parameters are subsequently calculated from the model parameters. A classical method used to obtain a process model is the step test. The Ziegler-Nichols and Cohen-Coon open-loop step response methods obtain their PID tuning parameters from the first order process model obtained in the step test method (Cohen and Coon, 1953; Ziegler and Nichols, 1942). Open-loop methods require that the controller be taken out of the process loop, therefore the normal operation is disturbed. This is also called off-line controller tuning. This disruption in normal operation can cause significant down time.

Data driven methods are mostly seen in Reinforcement learning (RL) applications which try to learn what the optimal controller settings are through an iterative process (Boubertakh et al., 2010; Howell and Best, 2000; Wakitani et al., 2013; Wang et al., 2007). The benefits of these data driven methods are that they can easily be implemented while the plant is running and can sit directly on the current control structure. Data driven methods can also keep track of a large interconnected system when combining multiple PID controllers. However, the drawback of the data driven methods is that they require perturbation of the plant. Since the learning takes much more time than calculating simple analytic settings, the plant may be perturbed for quite some time while the automatic tuner is learning.

The use of Bayesian optimization (BO) in learning seeks to optimize the search direction of the learning process and thus reduce the number of iterations required to find the optimal PID parameters (Berkenkamp et al., 2021). It is shown in Neumann-Brosig et al. (2020) that BO can be used to tune

controllers, where the input/output data is used as inputs to a cost function together with the current controller parameters from which controller tuning parameters are adjusted. This data is appended to a dataset which the BO uses to seek new controller settings for the following iteration to maximize information gained about the system whilst minimizing the cost function. This is applied to PID control in Fiducioso et al. (2019) and van Niekerk et al. (2023). This model-free fully online implementation is well-suited for automatic tuning as it does not require a stop in operation. It requires changes in the setpoint of the system to evaluate PID settings. Therefore it can take a long time to learn in a system with small setpoint changes.

1.1.2 Research gap

The success of data driven learning based methods in controller tuning has led to the increase in applications searching for a general automatic tuner capable of tuning various processes (Berkenkamp et al., 2021; Boubertakh et al., 2010; Howell and Best, 2000; Van Niekerk et al., 2023). However, these learning methods often require a large number of iterations to reach optimal controller settings.

Learning based methods tend to be more abstract than their intuitive counter parts in the classical offline model based methods. The learning part provides most of the abstraction and extra care needs to be taken to ensure closed-loop stability of the tuned system. A major part that can be improved is the time/iterations required to learn and tune a given system. Thus, the combination of learning based methods with BO can yield improvements for automatic tuners.

Processes that are limited to only a small dataset or time to converge to optimal tuning values can benefit from automatic tuners as mentioned above. The process considered in this dissertation is the pulp level control for a froth flotation bank. Controller structures and control methodologies have been investigated as in Schubert et al. (1995), Kämpjärvi and Jämsä-Jounela (2003) and Jämsä-Jounela et al. (2003), but little has been done on the automatic tuning of these controllers. Therefore a gap exists for the research of an automatic tuner on pulp level controllers.

1.2 RESEARCH OBJECTIVE AND QUESTIONS

This dissertation aims to optimize the search for optimal tuning parameters, starting from initial tuning parameters, be it through fewer system perturbations or a decrease in iterations taken to reach optimal parameters. The research will expand the developed automatic tuner from SISO systems to multiple-input-multiple-output (MIMO) systems. The research questions for this objective is:

- How can a data driven learning based automatic tuner be applied to level controllers in flotation banks?
- How can this automatic tuner for SISO PID controllers be expanded to a MIMO controller?
- How can the data driven methods be made more sample and time efficient?
- How can it be ensured that the automatic tuner produces controller settings such that the closed-loop system remains stable?
- Can the data driven method be applied in the presence of measurement noise?

1.3 APPROACH

To reach the research goals above the following approach will be followed:

- Research the literature on PID tuning rules as well as automatic PID tuners.
- Implement BO which consists of a Gaussian process and an acquisition function.
- Research and experiment to find the required response and the appropriate cost function to obtain that response in both SISO and MIMO cases.
- Simulate the flotation level model with the appropriate SISO controllers.
- Research and implement a robust stability analysis to find safe bounds for the automatic tuner to operate in.
- Tune the controllers and assess the newly tuned controllers by subjecting the plant to reference signal changes and disturbances.
- Select appropriate performance criteria for comparison with controller settings found by other tuning rules, and evaluate the performance.
- Evaluate the automatic tuner on its convergence and efficiency.
- Expand the approach to a MIMO controller.

1.4 RESEARCH CONTRIBUTION

The research contribution of this work is firstly the implementation of a BO automatic tuner on SISO and MIMO flotation level controllers in simulation. To do that, appropriate controllers have to be used and tested to ensure the closed-loop system remains stable. Thereafter, appropriate cost functions to attain optimality have to be setup. Strategies for tuning a MIMO controller using BO is developed.

1.5 RESEARCH OUTPUTS

The following article was published from this work:

- Richter, A., Le Roux, J.D., and Craig, I.K. (2024). Automatic tuning of level controllers in a flotation bank using Bayesian optimization, IFAC-PapersOnline, Volume 58, Issue 25, 2024, pp. 13-18.

The following article was submitted for publication from this work:

- Richter, A., Le Roux, J.D., and Craig, I.K. (2024). Bayesian optimization for automatic tuning of a MIMO level controller of a flotation bank, submitted to Journal of Process Control. August 2024.

1.6 OVERVIEW OF STUDY

In Chapter 2 the relevant literature on automatic tuning of PID controllers is presented. The main categories of controller tuning is presented with the different methods categorized according to the nature in which they find the controller settings and/or models. In Chapter 3 the method of BO is presented, as well as the method to analyze a system for robust stability. Chapter 4 presents the plant model on which BO was implemented. In Chapter 5 the plant is placed under SISO PI control. The initial SISO PI settings are found using tuning rules. The plant is assessed for robust stability and the controllers are tuned using BO. The performance of the new controller settings is evaluated and compared to the initial settings. In Chapter 6 the method is expanded to a MIMO controller. The same steps are followed as in the SISO case with the added results of the MIMO controller tuned in a system with measurement noise. The MIMO results are presented and evaluated. Chapter 7 provides the concluding remarks.

CHAPTER 2 LITERATURE STUDY

2.1 CHAPTER OVERVIEW

An overview of PID controller tuning is given in Section 2.2. This is followed by a discussion of the critical concepts of Chapter 2 in Section 2.3. The model based automatic tuning methods are presented in Section 2.4 followed by the model-free methods for controller tuning in Section 2.5. Controller tuning methods specific to MIMO systems are presented in Section 2.6. Finally the literature on modelling a flotation bank is discussed in Section 2.7 followed by the chapter summary in Section 2.8.

2.2 CONTROLLER TUNING OVERVIEW

Automatic tuning of PID controllers has been extensively utilized and investigated for SISO systems with classical methods (Somefun et al., 2021). Earlier works divided the adaption of PID parameters in the controllers into three sections namely: gain scheduling, adaptive control and automatic tuning (Åström et al., 1993). Gain scheduling is employed where the control parameters are varied according to the current operating conditions of the system. Adaptive control refers to the continuous adjustment of controller parameters in accordance with changes in the process dynamics as well as disturbances. Automatic tuning refers to the automated changing of PID parameters on the demand of the user to improve performance. The process of automatic tuning can broadly be divided into two steps: SID and controller design (Åström et al., 1993). Earlier works also divided the tuning topic into direct and indirect tuning (Åström et al., 1993; Ziegler and Nichols, 1942). Direct tuning is done from the observed response via heuristics and industry experience, while indirect tuning requires model identification to define models from which the PID controller parameters can be calculated. Recently the distinction between direct and indirect controller tuning has become more convoluted with the use of machine learning and the availability of large data sets where no explicit model is required to compute the PID parameters (Somefun et al., 2021). Automatic tuning of PID controllers could therefore also

be divided into the categories of model based tuning and model free tuning. In this dissertation implicit models will be regarded as model free as no predefined model structure is chosen from which to calculate controller settings. Obtaining a model of the system via SID is a critical part of the process as it is a prerequisite for systematic control design (Somefun et al., 2021). The model can be obtained explicitly from first principle modelling or implicitly through data driven processes. The design of the controller can then be done, dependent on the identification technique, either through the direct or indirect synthesis methods. Indirect synthesis requires an explicit model as the controller settings are indirectly synthesized from the tests done on the system being controlled. The direct synthesis method produces controller parameters directly from input-output data of the system being controlled. This is also called data driven controller tuning (Åström et al., 1993). Data driven techniques were previously captured in heuristic tuning and operator experience but advancements in processing technology has given rise to the field of modelling from input-output relations (Somefun et al., 2021).

2.3 CRITICAL CONCEPTS

This section is included to provide an overview of the automatic tuning process before the focus on specific methods. It also serves to explain some critical concepts used to categorize certain designs methods into specific categories of methods.

As the process of automatic tuning first requires some sort of SID (or performance evaluation), the initial step is to obtain data from the plant in question. Different ways of SID exists that fall into two categories: online and offline identification (Tan et al., 2001). The difference between the two is that in the online identification techniques the system is kept in a closed-loop configuration with the controller still performing the control action. In the offline techniques the controller is removed from the loop while the system is perturbed from an external source to obtain the needed measurements. It is important to note that in both online and offline techniques implicit or explicit models can be obtained.

The second part of the process is to design the controller. This involves finding the design parameters of the controller for some required specifications. These specifications are usually time domain response specifications and/or stability specifications such as gain- and phase margins, and bandwidth requirements. As mentioned above, the two main categories for controller design are direct and indirect synthesis of the design parameters (Åström et al., 1993). The direct methods use no models to obtain the tuning parameters and the tuning parameters are computed directly from input-output data. The

indirect methods use models and calculate the tuning parameters from the characteristics of these models.

2.4 MODEL BASED METHODS

2.4.1 Step response methods

The step response is a method widely used to produce a response in the system outputs by stepping the inputs. The input-output data is subsequently used to compute a specific model from which the controller parameters will be calculated. These methods can be either analytical or data driven. A classical approach is the Ziegler-Nichols open-loop step response method to obtain a first-order-plus-time-delay (FOPTD) model and compute controller settings from this model (Ziegler and Nichols, 1942). In practice this is accompanied by tuning charts for fine-tuning. Another very popular tuning method based on this principle is the Cohen-Coon method (Cohen and Coon, 1953). These methods lay the groundwork for tuning PID controllers and later on for automatic tuning. Methods that still use the step response as SID usually focus on obtaining better models from the step test. In Pavković et al. (2014) the approach is still analytical, as the FOPTD model is obtained through the integrated step response. The same step test is used in Hang and Sin (1991) as pre-tuning, after which the automatic tuning begins by injecting small perturbations into the system at the controller input to excite the system and obtain frequency domain information used in designing the PID parameters. The approach in Behera et al. (2017) follows a more data driven approach in obtaining an auto regression exogenous model to find tuning parameters through particle swarm optimization. One of the main differences between both Pavković et al. (2014) and Hang and Sin (1991), and Behera et al. (2017) is the requirement of offline tests. The open-loop step tests remove the PID controller and thus causes a stop in the normal process operation.

2.4.2 Relay feedback methods

The work in Åström and Hägglund (1984) changed the need for step tests by introducing relay feedback as a way of SID. Although this method is closed-loop it is essentially also offline, as the PID controller is disconnected for tuning purposes. Nonetheless, it is used frequently in automatic tuning procedures as a replacement of the closed-loop Ziegler-Nichols ultimate gain method of increasing the controller gain in closed-loop to find the critical gain and period of a system. The relay feedback generates a sustained oscillation used to find a design point on the Nyquist diagram or to construct a frequency model of the system. A higher order model is found in Sung et al. (1998) through relay excitation, which is then reduced to a FOPTD model to use classical tuning rules for the PID parameters. In Voda and Landau (1995) extremum seeking is introduced to set the PID parameters through symmetrical

optimum principles after the frequency domain model is obtained through relay excitation. In Schei (1994) a transfer function model is developed from limit cycling which is then used in a constrained optimization problem to determine the PID parameters according to a specified minimum allowable gain and phase margin. The method in Besharati Rad et al. (1997) follows a different course, where relay excitation is used to estimate system parameters via a recursive least-squares method. The estimated system is then used in a Newton-Raphson search for the ultimate gain and period to design the PID controller via classical tuning rules. In Tan et al. (2001) an online method for relay tuning is shown where a FOPTD model is fitted from the relay-induced oscillation signals. The PID controller is then designed with analytical expressions to achieve a desired response.

2.4.3 Miscellaneous

A less common method of automatic tuning is found in Woodyatt and Middleton (1997) where sine waves are injected into an open-loop system to classify the plant. The classification of the plant is then used to design a desired closed-loop transfer function. A least-squares frequency domain approximation is used to minimize a given cost function to obtain the PID controller parameters.

Another such method is the genetic model based automatic tuning found in Jones and De Moura Oliveira (1995). A genetic model is constructed from online data, where the PID parameters are known. This model is then subjected to simulation to optimize the performance of the PID controller. Once finished the obtained PID parameters are implemented.

The common theme in Section 2.4 is the identification of the plant, be it through classical methods such as step tests or relay excitation. The approach is to introduce perturbations in the system and actively model a system based on the observed responses. This system model is then used to design a PID controller. The design is done either through classical tuning rules or an offline search on the model for a set of optimized parameters.

2.5 MODEL-FREE METHODS

2.5.1 Classical methods

The relay excitation method is seen again with frequency response tuning methods as in Tan et al. (2001) with the direct tuning method. The controller is designed using the ultimate gain and period directly where a heuristic rule for the choice of the derivative parameter is used. The same is seen in Pandey et al. (2020), where on-off control is used to generate oscillations in the system. Characteristics of these oscillations, such as minimum/maximum value and the oscillation period is used to directly

design the PID controller. The relay excitation method is also seen in Ho et al. (2003), initially in open-loop and then closed-loop as the automatic tuning starts. A phase margin, bandwidth, and desired output is chosen, the iterative feedback tuning starts after the ultimate gain and frequency is determined. The iterative feedback then adjusts the PID parameters according to the value of a cost function, searching in directions that would minimize the given function. When the minimum is found the tuning stops and the PID parameters are kept at the obtained values.

Another interesting use of relay excitation is seen in Wang (2017), where the excitation is used in combination with frequency sampling filters to obtain an estimation of the frequency response of the system. A desired closed-loop frequency response is specified and used together with the estimated frequency response to design the PID controller.

Similar to the relay excitation is the injection of sine waves into the system as in Gyöngy and Clarke (2006). This is done in an online manner with small enough amplitude to match the noise in the system. Phase/Frequency- and gain estimators are employed to tune the PID parameters to find a design point on the Nyquist curve. The iterative adjustment of the PID parameters ensure the convergence of the design. Another frequency design technique is seen in Yang et al. (2012) where poles are placed based on a dominant pole implicit reference model obtained from input-output data. The controller design is then done through a recursive least-squares adaption law in combination with a reference process.

In an adaption of the classical step test, Shamsuzzoha et al. (2010) uses a closed-loop setpoint response to determine PID controller settings directly. This is done through the correlation between the setpoint response and Skogestad internal model control (SIMC) settings, where PID parameters are computed from the overshoot, peak time and steady-state output change.

2.5.2 Learning based methods

As the focus shifts from analytical solutions of input-output data to data driven processes, databases and learning techniques are used in the iterative tuning of the PID parameters. In Wakitani et al. (2013) a database method is combined with a fictitious reference feedback tuning method. This method is used to update a database from input-output data of a plant where the database is used in the calculation of the PID parameters using a nearest neighbors method. This method requires a large database to provide an initial tuning set, which is a drawback. A more learning based method is one seen in Wang

et al. (2007), where RL is used to adapt PID parameters using an actor-critic structure. The actor uses a radial basis function neural network to provide recommended parameters used by the stochastic action modifier. The modifier selects the actual PID parameters based on the recommendations and an estimated signal from the critic.

RL is also used in Howell and Best (2000), with their continuous acting reinforcement learning automata (CARLA) where a learning subsystem is used with a random action selector and performance evaluation. The learning subsystem produces actions based on certain probability density functions, the performance evaluation then provides a performance score each iteration which increases or decreases the probability of the previously selected action's re-selection. These value based iterative learning techniques are known as Q-learning.

Q-learning is used in the development of fuzzy PD and PI controllers in Boubertakh et al. (2010), where fuzzification is used to speed up the learning process. The use of triangular membership functions to fuzzify variables such as error, change or even a combination of error and controller output help ascribe meaning to the learned controller parameters. Thus, physical knowledge of the system can be introduced in the initial controllers, which helps in speeding up the learning process.

Fuzzy logic is also used in the tuning of PID controllers as in Ahn and Truong (2009). Triangle membership functions are used in fuzzy P, I and D blocks. The use of a block for each PID parameter simplifies the approach. A robust extended Kalman filter is used to tune the weights used in the membership functions of the fuzzy blocks. This inherently tunes the PID parameters to exhibit the required behavior.

The rewarding of PID parameters that yield a desired reduction in some cost function essentially encourages the search of parameters in a certain direction. The search direction can be optimized as seen in BO described in Berkenkamp et al. (2021). BO can be used to tune controllers as shown in Neumann-Brosig et al. (2020) as well as van Niekerk et al. (2022, 2023). Input-output data of a process is used as inputs to a cost function together with controller parameters, which is then appended to a data set. BO essentially uses this data set to predict a new set of parameters that will maximize information gain of the system while minimizing the cost function. This concept is applied to PID controller tuning as seen in Fiducioso et al. (2019) and (van Niekerk et al., 2022, 2023).

2.6 MIMO SYSTEMS

Thus far in the literature that was examined automatic tuners were mostly applied to SISO systems. Here automatic tuners applied to MIMO systems are looked at.

An early implementation found in Gawthrop and Nomikos (1990) derives control laws for MIMO systems, showing that to control the system to satisfaction one needs to decouple the interactions between the multiple loops. Feedforward control is suggested in combination with normal PID feedback control. The controller parameters are calculated from a system estimator.

Another method employed in MIMO system control is using decentralized controllers. In Veronesi and Visioli (2011) decoupling is done through splitting the system into several multi-input single-output loops and then designing controllers for those loops while being fed information on the other loops. The system is identified through a series of setpoint changes, then internal model control (IMC) tuning rules are used to calculate the PID parameters.

A computationally heavy method is seen in Tamura and Ohmori (2007), where a single controller representation is designed for a system in matrix transfer function form. The system is identified, and controller parameters are designed in array format using analytical equations.

The classical relay feedback method is also seen for MIMO process in Campestrini et al. (2006). Sequential relay feedback tuning is used to design decentralized controllers. This is done by putting the control loops under relay feedback, one loop at a time. After the experiment is finished the loop is closed with the designed PID controller. The same is done for the next loop with the previous loop closed. This done a second time for all loops with all other loops closed.

An automatic tuner using learning techniques is seen in Yu et al. (2007). Multiple neural networks are used, one to resemble the process being controlled and another used to predict the optimum tuning parameters for the current state of the process.

The use of optimization techniques such as loop shaping and critical point identification can also be used to automatically tune MIMO systems, but this optimization depends on high model accuracy (Boyd et al., 2015; Ono et al., 2000). The use of data driven iterative optimization methods are attractive as they guarantee that optimality will be reached (Guardeño et al., 2019; Euzebio et al.,

2021). One such data driven iterative optimization method is BO. As previously mentioned the goal of BO is to find an optimum according to an objective function while actively optimizing the search direction (Berkenkamp et al., 2021). BO was successfully implemented in various controller tuning applications (Fiducioso et al., 2019; Neumann-Brosig et al., 2020). More recently it was implemented in mineral processing (Coutinho et al., 2023; van Niekerk et al., 2022, 2023).

Data driven methods are of particular interest for controllers with high dimensionality as they are generally difficult to tune. SISO PI controllers are attractive because of their low dimensionality, but even these PI controllers when combined into a larger MIMO controller can be difficult to tune effectively by hand. The level control of a flotation bank with 6 cells in series is an example of SISO PI loops controlling a multivariable system (Kämpjärvi and Jämsä-Jounela, 2003). There are multiple intricacies in MIMO pulp level control such as controller structure, initializing controller parameters and MIMO stability. Apart from ensuring stability and managing the interactions in the system, the challenge is to tune the different loops to ensure satisfactory performance. Robustness can be ensured using μ-analysis (Skogestad and Postlethwaite, 2007) (assuming the plant is linear and time-invariant), but this can come at the expense of sufficiently fast controller responses. Placing constraints on the sensitivity transfer function of the closed-loop is shown to be effective in bounding the controller parameters to ensure stability while still prioritizing performance (Boyd et al., 2015).

2.7 FLOTATION LEVEL CONTROL

The automatic tuning will be applied to pulp level controllers in a flotation bank. Typically, these controllers take the form of PI controllers. A flotation bank is usually set up with the cells in series with one cell flowing into the next cell. This coupling between cells causes heavy interactions in the level responses in reaction to the valve control actions that change the flow through the cells. The levels also show interactions between the cell levels. An adequate model representing these interactions can be found in Jämsä-Jounela et al. (2003). Examples of MIMO level controllers can be seen in Schubert et al. (1995) and Kämpjärvi and Jämsä-Jounela (2003). However, these controllers still need to be tuned to operate efficiently. The more cells there are in the bank the larger the controller and the bigger the tuning effort. BO can provide a significant contribution to tuning these controllers as an automatic tuning strategy. The MIMO interactions provide a challenge to operators and automatic tuners.

2.8 CHAPTER SUMMARY

As discussed in the above sections there exists various different ways to approach the issue of automatic PID controller tuning. This arises from the different ways to approach the steps inherent of PID tuning

such as SID and controller design.

It is seen that the approach of obtaining explicit models is usually accompanied by some sort of system perturbation to obtain the system dynamics. The predominant controller design here is the use of set tuning rules without subsequent optimization of the parameters.

Contrast this to the model-free implementation, where iterative optimization is a popular approach. The iterative manner of these approaches may require a longer time to develop PID parameters but may yield more optimal controller settings.

The purely data-driven and optimization approaches are of particular interest as they do not require explicit models or large system perturbations making them ideal for online PID tuning. The machine learning field of these approaches are also on the rise as seen in the RL field. As these methods require large data sets, optimizing the search direction may help to find the optimal PID settings in as few iterations as possible.

A few approaches exist for MIMO systems, where the learning approach is underrepresented, but attempts are being made in addressing this as seen in Yu et al. (2007) and van Niekerk et al. (2023). Thus, a gap exists for MIMO implementations of PID automatic tuners using some form of learning and optimization techniques.

CHAPTER 3 BAYESIAN OPTIMIZATION

3.1 CHAPTER OVERVIEW

The optimization problem for automatic controller tuning is presented in Section 3.2. The method of Bayesian optimization in the context of automatic controller tuning is given in Section 3.3. The method for determining the bounds for Bayesian optimization is presented in Section 3.4 followed by the chapter summary in Section 3.5.

3.2 OPTIMIZATION PROBLEM

Given a general system under feedback control as shown in Fig. 3.1, with the reference input $\boldsymbol{r}(t)$, error signal $\boldsymbol{e}(t)$, the controller with tuning parameters $\boldsymbol{\alpha}$, the input to the plant $\boldsymbol{u}(t)$, the plant with states $\boldsymbol{x}(t)$ and output $\boldsymbol{y}(t)$ and disturbance to the system $\boldsymbol{d}(t)$. The plant can be represented mathematically as

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{f}(\boldsymbol{x}(t), \boldsymbol{u}(t), \boldsymbol{d}(t)), \tag{3.1}$$

with output

$$\boldsymbol{y}(t) = \boldsymbol{h}(\boldsymbol{x}(t), \boldsymbol{d}(t)). \tag{3.2}$$

The performance of the system largely depends on $\boldsymbol{\alpha}$ as it governs $\boldsymbol{u}(t)$ in response to $\boldsymbol{r}(t)$ and $\boldsymbol{d}(t)$. The response of the system to $\boldsymbol{r}(t)$ and $\boldsymbol{d}(t)$ can be quantified using performance metrics. The

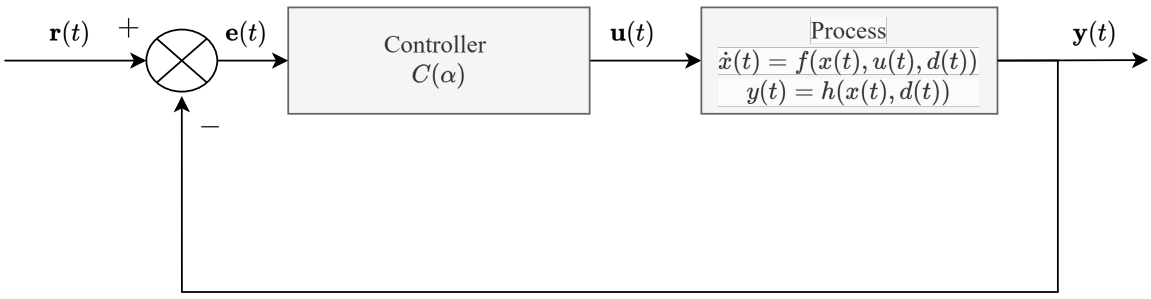


Figure 3.1. Plant with negative feedback controller.

performance metrics can be combined into an overall objective function Q from some criteria on how the system should behave. The task at hand is to select $\boldsymbol{\alpha}$ to find the best Q according to a predetermined performance criteria. In this case Q will be minimized, and thus the optimization problem is given as

$$\min_{\boldsymbol{\alpha}\in\mathcal{A}\subset\mathbb{R}^b} Q(\boldsymbol{\alpha}), \tag{3.3}$$

where b is the dimensions of $\boldsymbol{\alpha}$. The cost function that is minimized defines the desired process response. The general cost function can be expressed as

$$Q = \sum_{j}^{M} w_j \beta_j(\boldsymbol{\alpha}), \tag{3.4}$$

where β_j are the performance variables for $j = 1,\ldots,M$ and w_j is the weighting of each performance variable. For example, in this study β_j can reflect the influence of the PI gains to the settling time of a flotation cell. The objective function can be minimized in a number of ways. However, seeing as one does not necessarily have access to a mathematical form of the underlying relationship between $\boldsymbol{\alpha}$ and Q, and it is expensive to generate performance data on process plants, the minimization method should be efficient in its evaluations of the objective. A method known for its efficiency in iterative minimization is BO (Streltsov and Vakili, 1999; Jones, 2001). BO is attractive as it is a model free global optimization method (Shahriari et al., 2016). BO uses a surrogate model alongside an acquisition function to model as well as iteratively minimize the unknown objective function.

3.3 BAYESIAN OPTIMIZATION

The Bayesian way of extremum seeking is described as optimization based on the minimization of the expected deviation from the extremum (Močkus, 1975). The function that is minimized is treated as a stochastic function, where a probability distribution over functions in a parameter space is defined and adapted iteratively after making observations on the actual function being minimized. The probability distribution is used in creating a surrogate model as well as a decision-making function (the acquisition function). The BO automatic tuner will iteratively trial new controller tuning parameters $\boldsymbol{\alpha}$ to find parameters that minimize the objective function $Q(\boldsymbol{\alpha})$. The environment for the process under feedback control with the BO automatic tuner included can be seen in Fig. 3.2. The method of BO consists of two large parts: the Bayesian statistical model to create a surrogate of Q, and an acquisition function to make decisions on the best place to sample Q next (Snoek et al., 2012). One can start without any previous objective evaluations or design an initial sampling set that is collected into the dataset $\mathcal{D}(Q_n, \boldsymbol{\alpha}_n)$ to start off the BO process. The way the surrogate model is constructed can vary to fit the function being modelled, for example, the expected non-linearity and smoothness of the modelled function may influence the choice (Shahriari et al., 2016). Some selection criteria for the model are

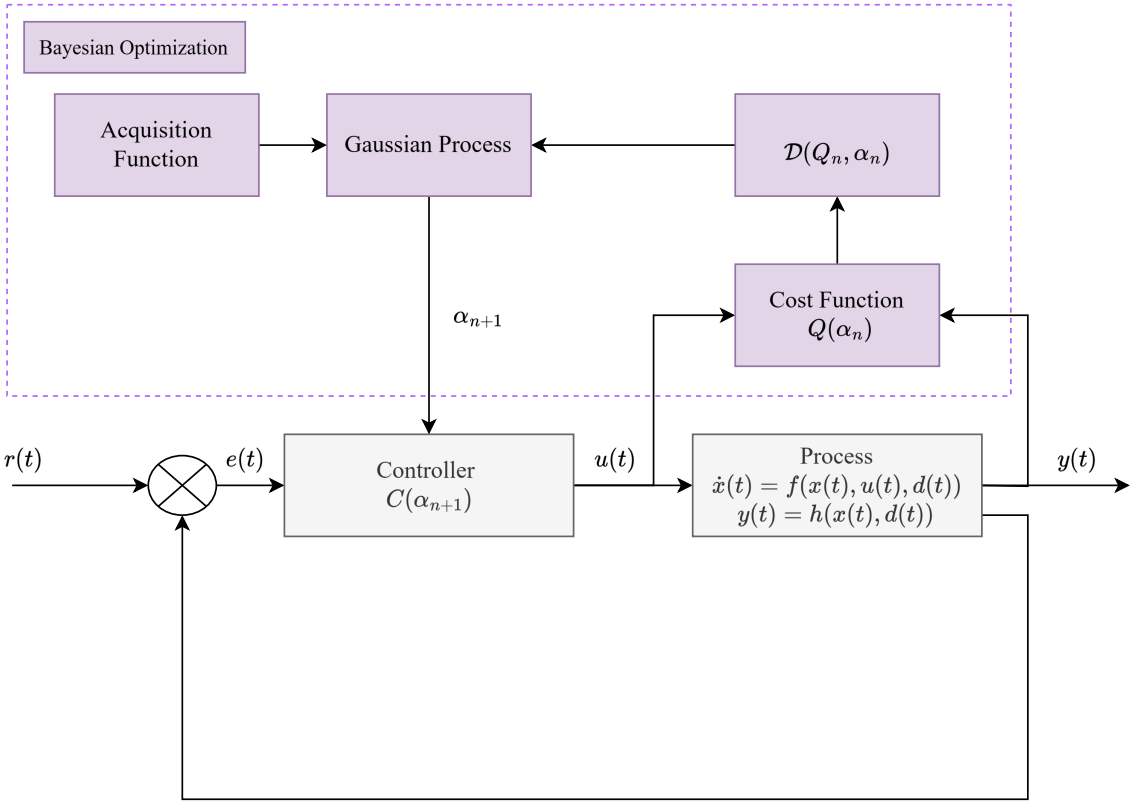


Figure 3.2. Feedback controller with BO automatic tuner.

that the surrogate model has to:

- Record observations dynamically.
- Predict samples over the whole parameter space.
- Provide some measure of model uncertainty.

The Gaussian process (GP) meets all these requirements and will be used in this dissretation. The GP is by definition a set of random variables of which any bounded set has a joint Gaussian distribution. The GP is fully described by its mean vector $\boldsymbol{m}(\cdot)$ and covariance matrix $\boldsymbol{K}(\cdot)$. In this case the random variables represent the function values of Q. As shown to in (3.3) the GP is indexed by the possible values of the controller tuning parameters $\boldsymbol{\alpha}$. Thus the GP for modelling the relationship between Q and $\boldsymbol{\alpha}$ is

$$Q(\boldsymbol{\alpha}) \sim \mathcal{GP}(\boldsymbol{m}(\boldsymbol{\alpha}), \boldsymbol{K}(\boldsymbol{\alpha}, \boldsymbol{\alpha'})), \tag{3.5}$$

where $\boldsymbol{m}(\boldsymbol{\alpha})$ is the GP mean vector subject to the vector of inputs $\boldsymbol{\alpha}$ and $\boldsymbol{K}(\boldsymbol{\alpha}, \boldsymbol{\alpha'})$ the covariance matrix between two sets of distinct input vectors $\boldsymbol{\alpha}$ and $\boldsymbol{\alpha'}$, where $\boldsymbol{\alpha'}$ is used to indicate a different set

of inputs than $\boldsymbol{\alpha}$. The GP allows the specification of the belief about the shape of Q without knowing Q itself through the GP prior. The GP also allows for statistical predictions about Q through the GP posterior. The prior is a set of arbitrary sample functions drawn in the parameter space according to the belief about the shape of Q. The BO process then samples Q and appends the new sample value Q_{n+1} at $\boldsymbol{\alpha}_{n+1}$ to the dataset $\mathcal{D}_{1:n+1}$ after $n+1$ samples. The data set is:

$$\mathcal{D} = \{\alpha_n, Q_n\}|_n^N, \tag{3.6}$$

where n is the current data point and N is the total number of data points. The posterior distribution is then calculated from the updated $\mathcal{D}_{1:n+1}$. The posterior belief about Q is then used to make statistical predictions of Q over the parameter space. It is these predictions that will be used by an acquisition function to choose the next sampling point.

As previously mentioned the sample efficiency of BO is particularly attractive, and thus it is apt to select an acquisition function with the same sample efficiency criteria in mind. Another point to keep in mind is that the better the surrogate model represents the function the easier it is to find the global optimum. Therefore, there is a trade-off to be made between exploration of the parameter space and exploitation of current knowledge (Frazier, 2018). Exploration equates to searching in places that the automatic tuner has not searched before and exploitation equates to searching more often in places where the function is known to have possible optima. Improvement based acquisition functions are well suited for this purpose. The expected improvement (EI) function provides an adjustable trade off in terms of an exploration ratio. Given that the GP gives access to the following statistics of the GP model:

- process mean $\boldsymbol{m}(\boldsymbol{\alpha})$,
- covariance between sample points $\boldsymbol{K}(\boldsymbol{\alpha}, \boldsymbol{\alpha'})$, and
- process variance,

the EI function uses this to produce the next sampling point, which then gets added to the dataset after the point is sampled, and the process continues until the number of available iterations are depleted, or some exit criteria is met. The structure of the BO algorithm can be seen in Algorithm 1 from Shahriari et al. (2016).

Algorithm 1 : Bayesian Optimization (Shahriari et al., 2016).

for $n = 1, 2, \cdots, N$ **do**

→ `select new` α_{n+1} `by optimizing acquisition function` EI`:`

$\alpha_{n+1} = \arg\max_{\alpha \in \mathcal{A}} EI(\alpha|\mathcal{D})$, (3.31)

→ `apply` $\boldsymbol{\alpha}_{n+1}$ `to the plant and observe the output.`

→ `query objective function to obtain` Q_{n+1} `(see (3.4))`

→ `augment data` $\mathcal{D}_{n+1} = \{\mathcal{D}_n, (\alpha_{n+1}, Q_{n+1})\}$, (see 3.6)

→ `update statistical model,` (see 3.27 - 3.30)

end for

3.3.1 Gaussian process model

The GP is selected for the surrogate model. There are several ways to view or interpret the GP, as given in Rasmussen and Williams (2006). Here the weight-space view will briefly be given as an introduction to the function-space view that will be used to describe the GP. Given a standard linear regression model with Gaussian noise:

$$f(\boldsymbol{\alpha}) = \boldsymbol{\alpha}^T \boldsymbol{w}, \quad y = f(\boldsymbol{\alpha}) + \varepsilon, \tag{3.7}$$

where $f(\boldsymbol{\alpha})$ is the function value at $\boldsymbol{\alpha}$, and y the observed value. The vector $\boldsymbol{w}$ is a vector of parameter weights used to describe the model. The observed values are assumed to differ from the function values through additive white Gaussian noise with zero mean and variance σ_n^2

$$\varepsilon \sim \mathcal{N}(0, \sigma_n^2). \tag{3.8}$$

The regression model coupled with the noise assumption gives rise to something called the *likelihood*, which is the probability density of the samples given the parameters factored over the prior dataset to give (Rasmussen and Williams, 2006):

$$\begin{aligned} P(\boldsymbol{y}|\boldsymbol{\Lambda}, \boldsymbol{w}) &= \prod_{i=1}^{n} P(y_i|\alpha_i, \boldsymbol{w}) \qquad (3.9) \\ &= \prod_{i=1}^{n} \frac{1}{\sqrt{2\pi}\sigma_n} \exp\left(-\frac{(y_i - \alpha_i^T \boldsymbol{w})^2}{2\sigma_n}\right) \\ &= \frac{1}{(2\pi\sigma_n^2)^{n/2}} \exp\left(-\frac{1}{2\sigma_n^2}|\boldsymbol{y} - \boldsymbol{\Lambda}^T \boldsymbol{w}|^2\right) \\ &= \mathcal{N}\left(\boldsymbol{\Lambda}^T \boldsymbol{w}, \sigma_n^2\right), \qquad (3.10) \end{aligned}$$

where the vector inputs $\boldsymbol{\alpha}$ for all n cases are aggregated in the $\mathcal{D} \times n$ design matrix $\boldsymbol{\Lambda}$. It is required to specify a *prior* over the parameter weights $\boldsymbol{w}$, that expresses the belief about the parameters. A zero

mean Gaussian prior, with a covariance of Σ_{prior} is put on the parameter weights such that

$$\boldsymbol{w} \sim \mathcal{N}(0, \Sigma_{prior}). \tag{3.11}$$

The prediction of the regression model is built on the posterior distribution over the parameter weights. The posterior makes it apparent why the method takes the name of BO, as the posterior is computed using Bayes' rule

$$\text{posterior} = \frac{\text{likelihood} \times \text{prior}}{\text{marginal likelihood}}, \quad P(\boldsymbol{w}|y,\boldsymbol{\Lambda}) = \frac{P(\boldsymbol{y}|\boldsymbol{\Lambda},\boldsymbol{w})P(\boldsymbol{w})}{P(\boldsymbol{y}|\boldsymbol{\Lambda})}, \tag{3.12}$$

where the marginal likelihood is a normalizing constant, independent of the parameter weights and given as

$$P(\boldsymbol{y}|\boldsymbol{\Lambda}) = \int P(\boldsymbol{y}|\boldsymbol{\Lambda}\,,\boldsymbol{w})P(\boldsymbol{w})d\boldsymbol{w}. \tag{3.13}$$

The likelihood of certain samples combined with the prior belief of the process contains everything known about the parameters. The posterior with dependence only on the parameter weights can be found to be a Gaussian distribution

$$P(\boldsymbol{w}|\boldsymbol{\Lambda},\boldsymbol{y}) \sim \mathcal{N}(\bar{\boldsymbol{w}} = \frac{1}{\sigma_n^2}\boldsymbol{K}^{-1}\boldsymbol{\Lambda}\boldsymbol{y}\,,\boldsymbol{K}^{-1}), \tag{3.14}$$

where $\bar{\boldsymbol{w}}$ is the mean vector and $\boldsymbol{K}$ the covariance matrix with

$$\boldsymbol{K} = \sigma_n^{-2}\boldsymbol{\Lambda}\boldsymbol{\Lambda}^T + \Sigma_{prior}^{-1}. \tag{3.15}$$

The Gaussian regression model described above can now be used to make predictions of $f(\boldsymbol{\alpha})$. The way a GP makes these predictions is by averaging over all the possible values of the parameters used to describe the sample functions according to the posterior probability. The predictive distribution for $f_*(\boldsymbol{\alpha}) \triangleq f(\boldsymbol{\alpha}_*)$ at $\boldsymbol{\alpha}_*$ can be found by averaging all the possible linear models in regard to the Gaussian posterior

$$P(f_*|\boldsymbol{\alpha}_*,\boldsymbol{\Lambda},\boldsymbol{y}) = \int P(f_*|\boldsymbol{\alpha}_*\,,\boldsymbol{w})P(\boldsymbol{w}|\boldsymbol{\Lambda},\boldsymbol{y})d\boldsymbol{w} \tag{3.16}$$

$$= \mathcal{N}\left(\frac{1}{\sigma_n^2}\boldsymbol{\alpha}_*^T\boldsymbol{K}^{-1}\boldsymbol{\Lambda}\boldsymbol{y}\,,\,\boldsymbol{\alpha}_*^T\boldsymbol{K}^{-1}\boldsymbol{\alpha}_*\right). \tag{3.17}$$

This is using linear regression. This might offer a very limited fit to real world processes but can be mitigated through projecting the inputs into a higher dimensional feature space using basis functions. The linear model can then be applied in the higher dimensional feature space. The vector function $\phi(\boldsymbol{\alpha})$ is selected to map the b-dimensional input vector into an M-dimensional feature space. The regression model then becomes

$$f(\boldsymbol{\alpha}) = \phi(\boldsymbol{\alpha})^T\boldsymbol{w}. \tag{3.18}$$

The predictive distribution is obtained as

$$P(f_*|\boldsymbol{\alpha}_*,\boldsymbol{\Lambda},\boldsymbol{y}) \sim \mathcal{N}\left(\frac{1}{\sigma_n^2}\phi(\boldsymbol{\alpha}_*)^T\boldsymbol{K}^{-1}\Phi(\boldsymbol{\Lambda})\boldsymbol{y}\,,\,\phi(\boldsymbol{\alpha}_*)^T\boldsymbol{K}^{-1}\phi(\boldsymbol{\alpha}_*)\right), \tag{3.19}$$

where $\Phi(\boldsymbol{\Lambda})$ is the combination of the columns $\phi(\boldsymbol{\alpha})$ for all the cases in the dataset and the covariance matrix $\boldsymbol{K} = \sigma_n^{-2}\Phi(\boldsymbol{\Lambda})\Phi(\boldsymbol{\Lambda})^T$. Now that there is a clear view of how the GP is set up the focus is shifted to how a GP is implemented.

The function-space view provides a convenient way with an emphasis on the basis functions mentioned earlier. Here the basis functions are used directly in the inference. The process mean vector $\boldsymbol{m}(\boldsymbol{\alpha})$ and covariance matrix $\boldsymbol{K}(\boldsymbol{\alpha}, \boldsymbol{\alpha'})$ still fully define the Gaussian process

$$Q(\boldsymbol{\alpha}) \sim \mathcal{GP}(m(\boldsymbol{\alpha}), \boldsymbol{K}(\boldsymbol{\alpha}, \boldsymbol{\alpha'})), \tag{3.20}$$

where the objective function $Q(\boldsymbol{\alpha})$ of the minimization problem is substituted for the function $f(\boldsymbol{\alpha})$ previously used, with

$$m(\boldsymbol{\alpha}) = \mathbb{E}[Q(\boldsymbol{\alpha})], \tag{3.21}$$

and

$$\boldsymbol{K}(\boldsymbol{\alpha}, \boldsymbol{\alpha'}) = \mathbb{E}[(Q(\boldsymbol{\alpha}) - m(\boldsymbol{\alpha}))(Q(\boldsymbol{\alpha'}) - m(\boldsymbol{\alpha'}))], \tag{3.22}$$

where $\mathbb{E}$ indicates the expected value. The above description of the regression model can be used to set up a GP with the model as $Q(\boldsymbol{\alpha}) = \phi(\boldsymbol{\alpha})^T \boldsymbol{w}$, with a zero mean prior $\boldsymbol{w} \sim \mathcal{N}(\boldsymbol{0}, \Sigma_{prior})$. The mean and covariance then becomes

$$\mathbb{E}[Q(\boldsymbol{\alpha})] = \phi(\boldsymbol{\alpha})^T \mathbb{E}[\boldsymbol{w}] = 0, \tag{3.23}$$

and

$$\mathbb{E}[Q(\boldsymbol{\alpha}), Q(\boldsymbol{\alpha'})] = \phi(\boldsymbol{\alpha})^T \mathbb{E}[\boldsymbol{w}\boldsymbol{w}^T]\phi(\boldsymbol{\alpha'}) = \phi(\boldsymbol{\alpha})^T \Sigma_{prior}\phi(\boldsymbol{\alpha'}). \tag{3.24}$$

The expectation $\mathbb{E}$ is that $Q(\boldsymbol{\alpha})$ and $Q(\boldsymbol{\alpha'})$ are jointly Gaussian with zero mean and covariance given by $\phi(\boldsymbol{\alpha})^T \Sigma_{prior}\phi(\boldsymbol{\alpha})$. The covariance function can be chosen from a number of different functions according to the belief of how the function changes over the parameter space. The automatic relevance determination (ARD) Matérn covariance function is attractive as it is twice differentiable and allows the user to easily adjust the function to their liking using a set of hyperparameters. The ARD Matérn52 function is given as

$$\boldsymbol{K}_{M52}(\alpha_p, \alpha_q) = \sigma^2 \left(1 + \sqrt{5}r + \frac{5}{3}r^2\right) \exp(-\sqrt{5}r), \tag{3.25}$$

with

$$r = \sqrt{\sum_{i=1}^{M} \frac{(\alpha_i - \alpha_i')^2}{\sigma_m^2}}, \tag{3.26}$$

where σ^2 is the variance and σ_m is a separate characteristic length scale for each predictor $i = 1, 2, ..., M$. This separate characteristic length scale is where the automatic relevance comes in, as σ_m determines how quickly the function can change over the parameter space. A distribution over the basis functions is implied by specifying the covariance function. This distribution will form the prior before new sample points are added. Examples of such basis functions are shown in Fig. 3.3. The joint distribution

of the prior and function values at the new sample points is given as

$$\begin{bmatrix} \boldsymbol{y} \\ Q_* \end{bmatrix} \sim \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} \boldsymbol{K}(\boldsymbol{\Lambda},\boldsymbol{\Lambda}) + \sigma_n^2 I & \boldsymbol{K}(\boldsymbol{\Lambda},\boldsymbol{\Lambda}_*) \\ \boldsymbol{K}(\boldsymbol{\Lambda}_*,\boldsymbol{\Lambda}) & \boldsymbol{K}(\boldsymbol{\Lambda}_*,\boldsymbol{\Lambda}_*) \end{bmatrix}\right), \tag{3.27}$$

with the same assumptions about noisy observations as previously in the weight-space representation. The predictive equations for Gaussian process regression are found by deriving the conditional distribution

$$Q_*|\boldsymbol{\Lambda},\boldsymbol{y},\boldsymbol{\Lambda}_* \sim \mathcal{N}(\bar{Q}_*, \mathrm{cov}(Q_*)), \tag{3.28}$$

where

$$Q_* \triangleq \mathbb{E}[\bar{f}_*|\boldsymbol{\Lambda},\boldsymbol{y},\boldsymbol{\Lambda}_*] = \boldsymbol{K}(\boldsymbol{\Lambda}_*,\boldsymbol{\Lambda})[\boldsymbol{K}(\boldsymbol{\Lambda},\boldsymbol{\Lambda}) + \sigma_n^2 I]^{-1}\boldsymbol{y}, \tag{3.29}$$

$$\mathrm{cov}(Q_*) = \boldsymbol{K}(\boldsymbol{\Lambda}_*,\boldsymbol{\Lambda}_*) - \boldsymbol{K}(\boldsymbol{\Lambda}_*,\boldsymbol{\Lambda})[\boldsymbol{K}(\boldsymbol{\Lambda},\boldsymbol{\Lambda}) + \sigma_n^2 I]^{-1}\boldsymbol{K}(\boldsymbol{\Lambda},\boldsymbol{\Lambda}_*). \tag{3.30}$$

Q_* is the predicted mean and the predicted variance is the diagonal elements of $\mathrm{cov}(Q_*)$. The predicted mean and variance will be used by the acquisition function to select the next parameters to sample.

3.3.2 Acquisition function

The acquisition function provides the next points at which to evaluate the objective function. Different functions can be used for different purposes, if the goal was to minimize model uncertainty one could use the information from the GP and sample where the predicted model variance is the largest. However, the goal is to minimize the cost function within as few samples as possible. Thus, parameters that maximize information gain but also minimize the objective function are desired. These parameters are obtained where the acquisition function $\mathcal{L}$ is maximized, with

$$\boldsymbol{\alpha}_{n+1} = \arg\max_{\boldsymbol{\alpha}\in\mathcal{A}} \mathcal{L}(\boldsymbol{\alpha}|\mathcal{D}). \tag{3.31}$$

The choice of acquisition function determines how the decision is made on which point to sample next. A number of options are available with variable features available to prioritize. As mentioned earlier, the EI function is used here. The EI function seeks the next evaluation points where the objective function is expected to show the best improvement according to the lowest posterior mean of the GP model. This is given by

$$EI(\boldsymbol{\alpha},Q) = \mathbb{E}\max[0, Q_{best}(\boldsymbol{\alpha}_{best}) - Q(\boldsymbol{\alpha})], \tag{3.32}$$

where $Q_{best}(\boldsymbol{\alpha}_{best})$ is the current best mean at the location α_{best}. This is solved analytically (Jones et al., 1998) as

$$EI(\boldsymbol{\alpha}) = (Q_{best} - \overline{\boldsymbol{Q}}_*(\boldsymbol{\alpha}))\Phi(z(\boldsymbol{\alpha})) + \sigma(\boldsymbol{\alpha})\phi(z(\boldsymbol{\alpha})), \tag{3.33}$$

where $\overline{\boldsymbol{Q}}_*$ is the predicted mean, Φ is the cumulative distribution function, ϕ is the probability density function and with

$$z(\boldsymbol{\alpha}) = \frac{Q_{best} - \overline{\boldsymbol{Q}}_*(\boldsymbol{\alpha})}{\sigma(\boldsymbol{\alpha})}, \tag{3.34}$$

where $\sigma(\boldsymbol{\alpha})$ is the predicted standard deviation at $\boldsymbol{\alpha}$.

3.3.3 Example of Bayesian Optimization

The BO process can be illustrated on a known function, for example

$$g(x) = x \cdot sin(x). \tag{3.35}$$

The GP is started without any prior knowledge, but with a random kernel with the prior distribution over the basis functions as shown in Fig. 3.3 to capture the smoothness of the function that is minimized. The function is then sampled as illustrated in Fig. 3.4. After the first sample the kernel is adjusted to fit the data in $\mathcal{D}$. The top graphs (3.4(a)-3.4(c)) illustrate the GP model with model mean (green line), variance (gray area), basis functions (gray lines) and the actual function (red line). The bottom graphs (3.4(d)-3.4(f)) show the expected improvement (EI) function with the black dots as the next sample points. In samples 1 to 4 the BO is building the GP model. From samples 5 to 9 the BO adapts this GP model but also uses it in the acquisition function to guide the search for the minimum. This is shown in Figs. 3.5 and 3.6. This guidance can be seen in the form of the EI function over the parameter space. One can see the BO making a trade-off between exploration and exploitation after sample 6 where a local minimum was found and then after sample 8 where the global minimum was found for the bounded parameter space. The BO explores the parameter space once it has exploited the current best input parameter. After sample 9 the EI suggestion is to return to try and exploit the minimum found at $x = 10$. Through the samples in Figs. 3.4 to 3.6 the unknown function is fitted by the GP and

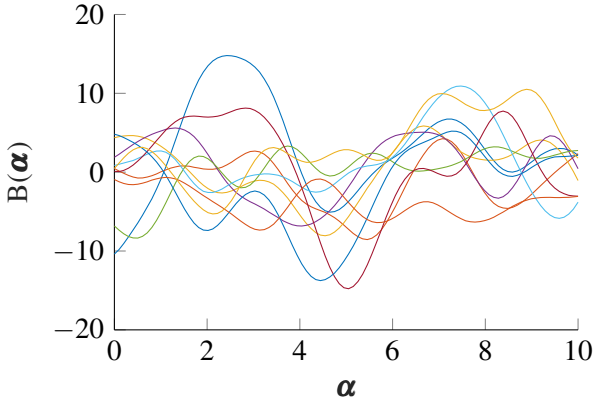
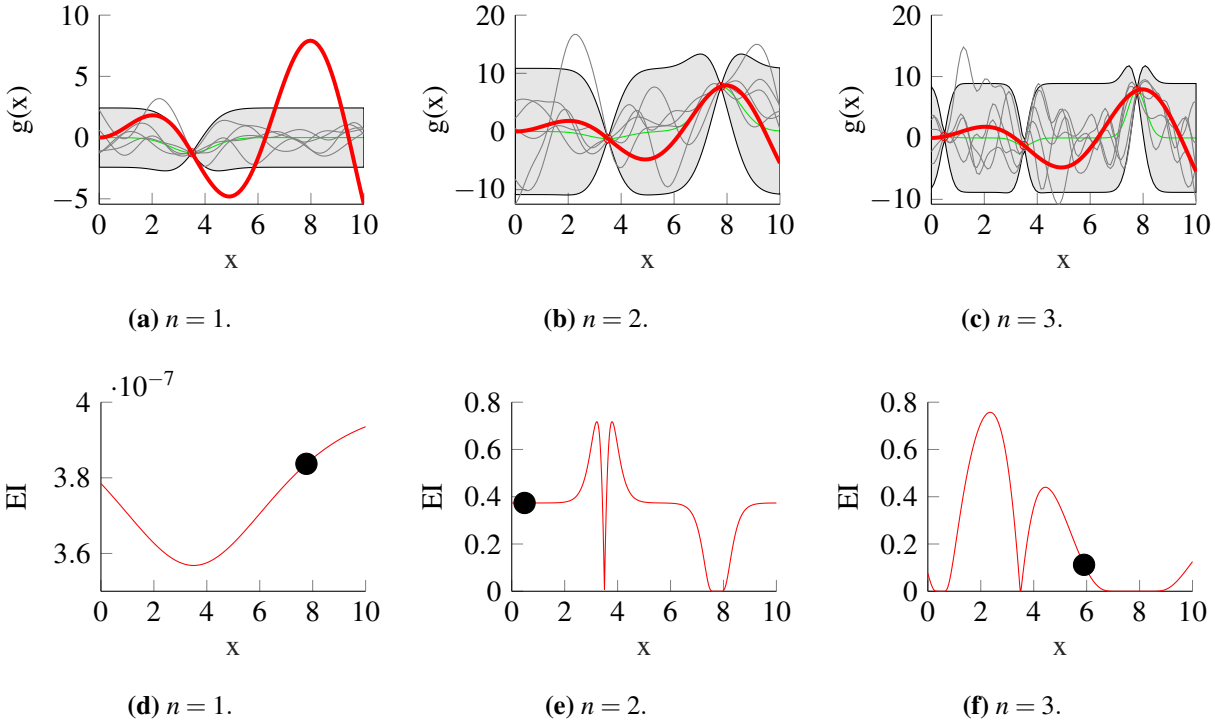


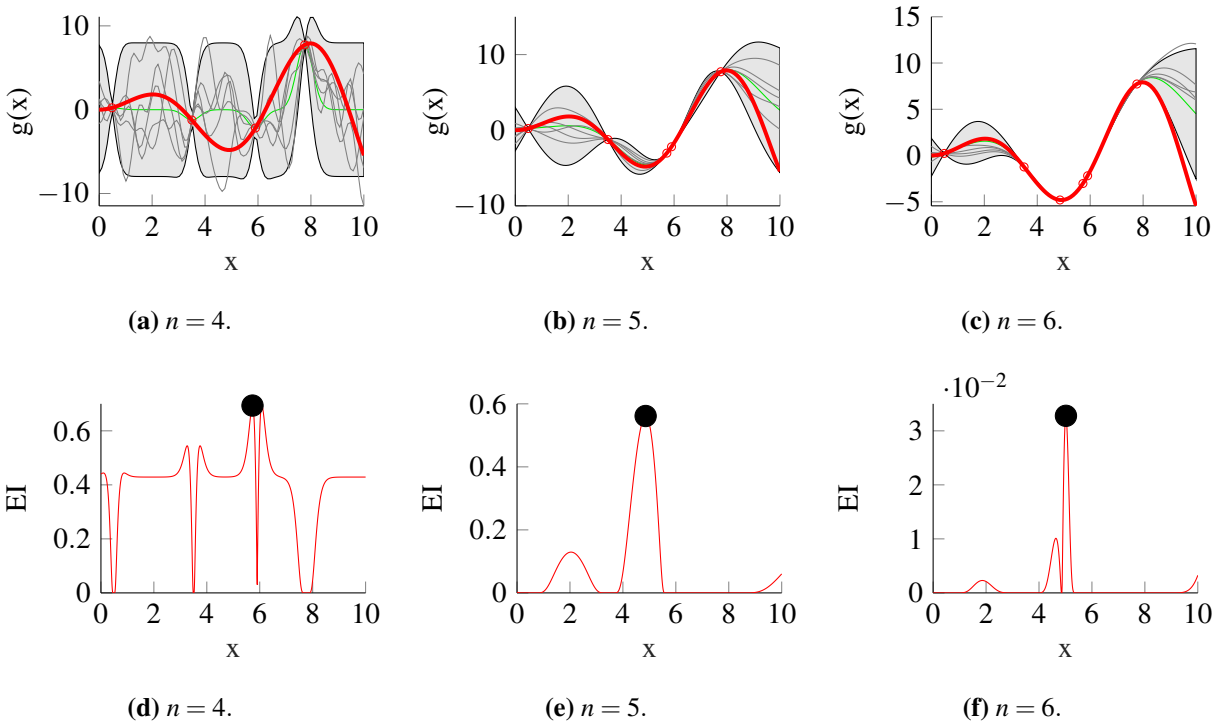
Figure 3.3. Basis functions with prior distribution.



(a) $n = 1$. **(b)** $n = 2$. **(c)** $n = 3$.

(d) $n = 1$. **(e)** $n = 2$. **(f)** $n = 3$.

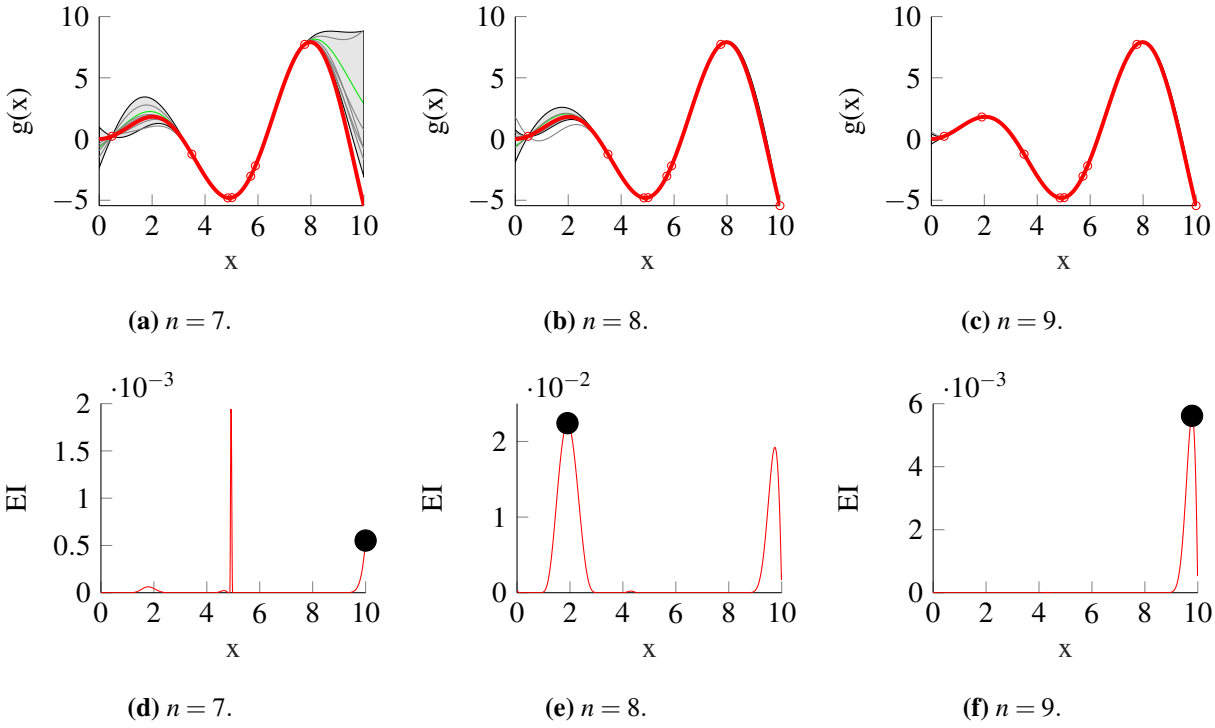
Figure 3.4. BO iterations $n = 1 : 3$.



(a) $n = 4$. **(b)** $n = 5$. **(c)** $n = 6$.

(d) $n = 4$. **(e)** $n = 5$. **(f)** $n = 6$.

Figure 3.5. BO iterations $n = 4 : 6$.



(a) $n = 7$. **(b)** $n = 8$. **(c)** $n = 9$.

(d) $n = 7$. **(e)** $n = 8$. **(f)** $n = 9$.

Figure 3.6. BO iterations $n = 7 : 9$.

explored by the EI function to both model and find the minimum of $g(x)$. This is done through very few samples and illustrates the power of having a good trade-off between exploration and exploitation. It also shows the versatility of the GP model that is able to capture the form of the unknown function over the entire parameter space.

3.4 ROBUST STABILITY FOR SISO AND MIMO SYSTEMS

The BO automatic tuning procedure has been described above, but the question of how the parameter space is defined for the automatic tuner still remains. The automatic tuner should not produce parameters that could make the closed-loop system unstable. Given the interactions in a plant, combined with multiple feedback controllers the matter of closed-loop stability can become complex.

Skogestad and Postlethwaite (2007) provides a convenient way of handling this parametric uncertainty in what is called the structured singular value analysis or μ-analysis. Given a simple feedback control loop as in Fig. 3.7, the process is assumed to be linear time-invariant, the uncertain controller parameters can be represented as a multiplicative input uncertainty to the plant. The controllers used

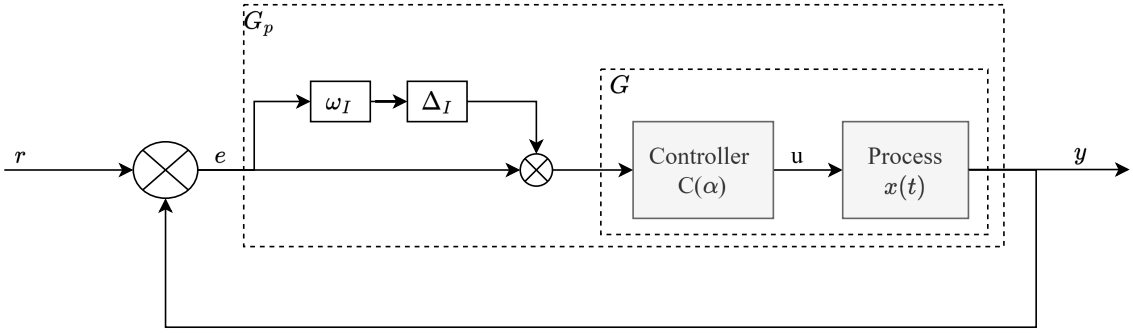


Figure 3.7. Feedback system with multiplicative uncertainty.

in this work are PI controllers of the form:

$$c_{PI} = k_c \left(1 + \frac{1}{\tau_I s}\right), \tag{3.36}$$

with the controller gain k_c within the range

$$k_{min} \leq k_c \leq k_{max}, \tag{3.37}$$

and the controller integral time constant within the range

$$\tau_{min} \leq \tau_I \leq \tau_{max}. \tag{3.38}$$

Both the parameters k_c and τ_I have parameter uncertainty. The controller is rewritten to simplify the uncertainty analysis. (3.36) becomes

$$c_{PI} = \frac{k_c}{\tau_I}\frac{1}{s}(1 + \tau_I s). \tag{3.39}$$

This can be handled as multiple gain uncertainties and a time constant uncertainty. The first gain uncertainty is taken as

$$k_c = \bar{k}(1 + r_k \Delta), \quad \bar{k} \triangleq \frac{k_{min} + k_{max}}{2}, \quad r_k \triangleq \frac{(k_{max} - k_{min})/2}{\bar{k}}, \tag{3.40}$$

where $\bar{k}$ is the average gain and r_k is the relative magnitude of the gain uncertainty. Δ is a scalar value so that $|\Delta| \leq 1$. The second gain uncertainty is taken as

$$\frac{1}{\tau_I} = \frac{1}{\bar{\tau}_I(1 + r_\tau \Delta)}, \quad \bar{\tau} \triangleq \frac{\tau_{min} + \tau_{max}}{2}, \quad r_\tau \triangleq \frac{(k_{max} - k_{min})/2}{\bar{\tau}}. \tag{3.41}$$

The time constant uncertainty can be represented as

$$(1 + \tau_I s) = (1 + \bar{\tau}_I s + r_\tau \bar{\tau}_I s \Delta). \tag{3.42}$$

The time constant uncertainty introduces the need for uncertainty in the form of a complex region around the nominal model in the frequency domain. This complex region is best described as a complex perturbation. The perturbation is represented by a weighted function $\omega_I(s)$ so that the uncertain model

is given by

$$(1+\bar{\tau}_I s + r_\tau \bar{\tau}_I s\Delta) = (1+\bar{\tau}_I s)(1+\omega_I(s)\Delta), \tag{3.43}$$

with

$$\omega_I(s) = \frac{r_{\tau\bar{\tau}_I s}}{1+\bar{\tau}_I s}. \tag{3.44}$$

The complex perturbations are very useful in representing models with multiple sources of uncertainty, such as the PI controller with uncertainty in k_c and τ_I. The complexity of the uncertainty grows with each uncertain parameter. It is attractive to lump all the parametric uncertainties into one and use a single weighted function to account for all uncertainty. This is what the *robstab* function in MATLAB does used later to compute the robust stability given system uncertainty. In the frequency domain the smallest radius $l(\omega)$ from the nominal model $G(s)$ that includes all possible models Π due to multiplicative uncertainty is found as:

$$l(\omega) = \max_{G\in\Pi} \left| \frac{G_p(j\omega) - G(j\omega)}{G(j\omega)} \right|, \tag{3.45}$$

where $G_p(j\omega)$ represents a complex region around $G(j\omega)$ at each frequency brought about by the uncertain model. The weighted function is then selected as

$$|\omega_I(j\omega)| \geq l(\omega), \forall\omega. \tag{3.46}$$

The loop transfer function with uncertainty for the system in Fig. 3.7 with the controller C is

$$L_p = G_p C = GC(1+w\Delta) = L + \omega_I L\Delta, \quad |\Delta(j\omega)| \leq 1, \forall\omega. \tag{3.47}$$

The Nyquist stability condition and L_p is used to define the robust stability (RS) in Skogestad and Postlethwaite (2007) as

$$\text{RS} \Leftrightarrow |\omega_I L| \leq |1+L|, \forall\omega, \tag{3.48}$$

and the condition on the complimentary sensitivity function (T) is found as

$$\text{RS} \Leftrightarrow |T| \leq \frac{1}{\omega_I}, \forall\omega, \tag{3.49}$$

and finally the structured singular value for RS as

$$\text{RS} \Leftrightarrow \mu = |\omega_I T| \leq 1, \forall\omega. \tag{3.50}$$

The μ-value will be used to evaluate the RS of the closed-loop system with PI controllers that have uncertainty in the gain and integral time constant. The uncertainty will be set up to act as tuning bounds for the BO automatic tuner. In doing so the BO automatic tuner will be provided with bounds prior to commencing the tuning procedure. The initial parameter uncertainty will be derived from initial controller parameters. This uncertainty will then be subjected to μ-analysis. The μ-analysis will give the final bounds for the BO automatic tuner to operate in as a bounded parameter space.

The discussion of structured and parametric uncertainty discussed above applies also the MIMO case. The main difference between the SISO and MIMO cases are that the latter introduces directionality in the uncertainty. The core of the stability analysis remains to find some boundary in the frequency domain that contains all possible plant variations. The boundary can be used as a weighted function for the uncertainty to compute the complementary sensitivity function to find the structured singular values for the MIMO system.

3.5 CHAPTER SUMMARY

The method for the BO automatic tuner is described in detail. The different subcomponents of the BO algorithm is described with examples. The manner in which the parameter space is bounded to ensure stability of the closed-loop system is discussed. This bounding of the parameter space requires a linear model of the plant and controller. The μ-analysis used to bound the parameter space provides a conservative estimate of the stability of the closed-loop system given the initial uncertainty provided. The outcome of the μ-analysis will give more information on if the parameter space can be expanded or of it needs to be reduced to ensure stability.

CHAPTER 4 THE FLOTATION PROCESS

4.1 CHAPTER OVERVIEW

The non-linear flotation model is presented in Section 4.2. The differential equations are given and the parameters of the plant are defined. The nominal operating conditions of the plant are given. The model is linearized in Section 4.3 for the μ-analysis as described in Section 3.4. This is followed by the chapter summary in Section 4.4.

4.2 FLOTATION BANK MODEL

The model used for the flotation bank with 6 cells in series can be found in Jämsä-Jounela et al. (2003). The pulp level in each cell is modelled by integrating the difference between the volume flow in and out of each cell. The SISO controlled plant is given in Fig. 4.1. The rate of change in pulp level is

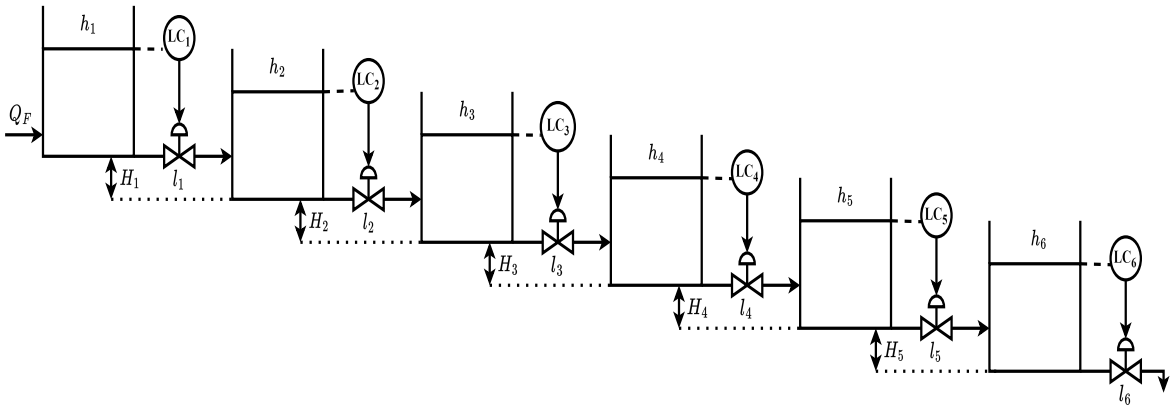


Figure 4.1. Flotation bank configuration with individual PI level controllers for each cell.

given by

$$\dot{h}_i = \frac{\dot{V}_i}{A_i}, \tag{4.1}$$

where $\dot{h}_i$ is the rate of change in the pulp level of cell i, $\dot{V}_i$ is the rate of change in the volume of cell i, and A_i is the cross-sectional area of the cell which is assumed to be constant. Note, (4.1) assumes a

constant gas holdup throughout the bank, similar to Kämpjärvi and Jämsä-Jounela (2003). The goal of flotation level control is to reject disturbances caused by any of, but not limited to, the following: feed flow changes, spillage water and/or interactions by other cells. The first two cannot be used to tune the controllers as those disturbances do not result from anything within the operator's controls and thus level setpoint changes are used to generate disturbances due to the interactions between the cells. The rate of change in volume for each of the cells $i = 1,...,6$ is given as

$$\dot{V}_i = B_{i-1}C_v f_c(l_{i-1})\sqrt{h_{i-1} - h_i + H_{i-1}} - B_i C_v f_c(l_i)\sqrt{h_i - h_{i+1} + H_i}, \tag{4.2}$$

where C_v is the valve capacity coefficient, l_i is the input signal to the valve, $f_c(\cdot)$ is the valve position, h_i is the pulp level and H_i is the physical height difference between the bottoms of the cells as shown in Fig. 4.1. H_6 is the difference in height between the bottom of the last cell and a tailings sump. The coefficient B_i is calculated with $l_i = 0.5$ and the feed flow $Q_F = 2336$ m^3/h to ensure a steady-state. The rate of change in volume in the first cell is given by

$$\dot{V}_1 = Q_F - B_1 C_v f_c(l_1)\sqrt{h_1 - h_2 + H_1}, \tag{4.3}$$

and the rate of change in volume in the last cell by

$$\dot{V}_6 = B_5 C_v f_c(l_5)\sqrt{h_5 - h_6 + H_6} - B_6 C_v f_c(l_6)\ \sqrt{h_6 + H_6}. \tag{4.4}$$

The valve position $f_c(\cdot)$ is assumed to be a linear function of the control input signal adapted around the steady-state position, thus $f_c(l_i) = l_i + 0.5$, with $f_c(l_i) = 1$ being fully open and $f_c(l_i) = 0$ being fully closed. The valve capacity coefficient C_v is given by

$$C_v = 1.17 Q_m \sqrt{\frac{\rho_p}{\Delta_p}}, \tag{4.5}$$

where $Q_m = 1.2\frac{V_{cell}}{\tau/60}$ is the mean volume flow through a cell with a volume of V_{cell}, $\Delta_p = \rho_p g H_i$ is the pressure difference across a valve, ρ_p is the pulp density, and $g = 9.81$ m/s^2 is the gravitational constant. The pulp retention time in each cell is τ (in minutes). The flotation bank parameters are shown in Table 4.1 and the initial cell level profile is given in Table 4.2.

Table 4.1. Flotation circuit parameters.

Parameter	Value	Unit	Parameter	Value	Unit
Q_F	2336	m^3/h	$B_{i=1..5}$	3.49	m$^{2.5}$/s
V_{cell}	76.0	m^3	B_6	1.41	m$^{2.5}$/s
A_i	12.0	m^2	H_i	0.85	m
τ	1.50	min			

Table 4.2. Cell level profile.

Parameter	Value	Unit	Parameter	Value	Unit
h_1	4.06	m	h_4	4.15	m
h_2	4.09	m	h_5	4.18	m
h_3	4.12	m	h_6	4.21	m

4.3 LINEAR PLANT MODEL

The plant as described above is non-linear, the plant is linearized to perform robust stability analysis around a particular operating point, as done for the SISO controller in Section 5.4 and for the MIMO controller in Section 6.3.2. The linearization is performed at the operating conditions given in Tables 4.1 and 4.2, and written as a state-space model used by the μ-analysis to find the ranges of the controller tuning parameters. Note, the non-linear plant is controlled in the simulation and the linear plant is only used for the robust stability analysis. The non-linear system of the form

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{f}\left(\boldsymbol{x}(t), \boldsymbol{u}(t)\right) \tag{4.6a}$$

$$\boldsymbol{y}(t) = I\boldsymbol{x}, \tag{4.6b}$$

is linearized at the steady-state operating conditions shown in Tables 4.1 and 4.2. For (4.6) the state vector $\boldsymbol{x}$ represents the six pulp levels h_i, the input vector is $\boldsymbol{u} = [Q_F, l_{1,\ldots,6}]^T$ with Q_F taken as an extra uncontrolled input for linearization purposes. The elements of the vector function $\boldsymbol{f}$ is given by (4.1), and the output $\boldsymbol{y}$ is a measurement of all the states. The linear representation of (4.6) is obtained by means of Taylor series expansion and is given by

$$\begin{aligned}\dot{\delta}\boldsymbol{x}(t) &\equiv \left.\frac{\partial f}{\partial \boldsymbol{x}}\right|_{\substack{\boldsymbol{x}=\bar{\boldsymbol{x}}\\ \boldsymbol{u}=\bar{\boldsymbol{u}}}} \delta\boldsymbol{x}(t) + \left.\frac{\partial f}{\partial \boldsymbol{u}}\right|_{\substack{\boldsymbol{x}=\bar{\boldsymbol{x}}\\ \boldsymbol{u}=\bar{\boldsymbol{u}}}} \delta\boldsymbol{u}(t) \\ &= A\delta\boldsymbol{x}(t) + B\delta\boldsymbol{u}(t) \end{aligned} \tag{4.7a}$$

$$\delta\boldsymbol{y}(t) = I\delta\boldsymbol{x}(t), \tag{4.7b}$$

where the deviation variables in terms of a steady-state operating condition $(\bar{\boldsymbol{x}}, \bar{\boldsymbol{u}}, \bar{\boldsymbol{y}})$ are

$$\delta\boldsymbol{x}(t) = \boldsymbol{x}(t) - \bar{\boldsymbol{x}}$$

$$\delta\boldsymbol{u}(t) = \boldsymbol{u}(t) - \bar{\boldsymbol{u}}$$

$$\delta\boldsymbol{y}(t) = \boldsymbol{y}(t) - \bar{\boldsymbol{y}},$$

and the state-space matrices for the operating conditions in Tables 4.1 and 4.2 are given by

$$A = \begin{bmatrix} -118.7 & 118.7 & 0 & 0 & 0 & 0 \\ 118.7 & -237.4 & 118.7 & 0 & 0 & 0 \\ 0 & 118.7 & -237.4 & 118.7 & 0 & 0 \\ 0 & 0 & 118.7 & -237.4 & 118.7 & 0 \\ 0 & 0 & 0 & 118.7 & -237.4 & 118.7 \\ 0 & 0 & 0 & 0 & 118.7 & -137.9 \end{bmatrix}$$

$$B = \begin{bmatrix} 0.08333 & -389.3 & 0 & 0 & 0 & 0 & 0 \\ 0 & 389.3 & -389.3 & 0 & 0 & 0 & 0 \\ 0 & 0 & 389.3 & -389.3 & 0 & 0 & 0 \\ 0 & 0 & 0 & 389.3 & -389.3 & 0 & 0 \\ 0 & 0 & 0 & 0 & 389.3 & -389.3 & 0 \\ 0 & 0 & 0 & 0 & 0 & 389.3 & -389.3 \end{bmatrix}.$$

4.4 CHAPTER SUMMARY

The flotation level model is given and the nominal operating conditions are defined. The plant is linearized around a nominal steady-state operating condition for μ-analysis. The linearized state-space representation of the plant shows the amount of interactions present in the cell levels. Both the levels and control actions will propagate through the plant in both directions. This will make the plant difficult to decouple for efficient control. The interactions will also make it difficult to independently tune the controllers for each cell, as any perturbation used to identify the level response from the valve will have dynamics from other valves and levels.

CHAPTER 5 FF CONTROL AND BO AUTOMATIC TUNING OF SISO LEVEL CONTROLLERS

5.1 CHAPTER OVERVIEW

The flotation bank model is placed under SISO level control. In Section 5.3 the controller structure and initial controller tuning is presented. The initial tuning is found using open-loop step tests to the valve position from which SIMC settings are calculated. The BO bounds are found through robust stability analysis in Section 5.4, using the SIMC settings as a starting point. The simulation of the flotation model with controller and automatic tuner is discussed in Section 5.5 alongside the tuning strategy and performance metrics. The results of the simulation presented and discussed in Section 5.6. The chapter summary is given in Section 5.7.

5.2 OVERVIEW OF BO AUTOMATIC TUNING OF SISO LEVEL CONTROLLERS

The work described in this Chapter is based on Richter et al. (2024a). The flotation bank is placed under SISO control with a PI level controller placed on each cell, with the valve position as manipulated variable and the level as the controlled variable. First SIMC settings are found for the controllers through open-loop step tests of the valves. This involves SID of the step response data and model reduction to find adequate transfer function models to calculate the SIMC settings. The BO automatic tuner bounds are found using the SIMC settings as starting point for the robust stability analysis (μ-analysis). Next the BO automatic tuner is allowed to tune the PI controllers individually given the bounds found by the μ-analysis. Details of the BO automatic tuner process is shown to gain insight into how the BO automatic tuner goes about finding the optimal controller settings. Both the SIMC controllers and BO controllers are subjected to a performance evaluation and compared using a few different performance metrics. The tuning of the SISO PI controllers on the diagonal of the controller

matrix forms part of the larger goal to tune the MIMO controller. The controller elements on the diagonal are tuned and fixed first to limit the fluctuations caused by the closed-loop perturbations used to generate performance data when tuning the controllers. The off-diagonal elements are then tuned with the diagonal elements fixed. This strategy is further discussed in Section 6.4.

5.3 CONTROLLER STRUCTURE AND INITIAL TUNING

5.3.1 PI controller

The pulp level model in Chapter 4 can be controlled with various controllers, however most plants in the industry use PI or PID controllers. PI and PID controllers are feedback controllers that compare a reference signal (or setpoint) to the manipulated variable, in this case a level setpoint and the measured pulp level. The error between the two is passed to the controller which calculates a control signal that gets sent to the plant. This structure is seen in Fig. 3.1. The manner in which the control signal is calculated depends on the controller structure and tuning parameters $\boldsymbol{\alpha}$. The PI controller structure will be used as the base control element throughout this work. The structure is the same as in (3.36) and is repeated here

$$c_{ij} = k_c(1 + \frac{1}{\tau_I s}). \tag{5.1}$$

The control signal is calculated as

$$u_i = u_{base} + k_c(1 + \frac{1}{\tau_I s})e_i. \tag{5.2}$$

where the valve position (u_i) for each cell is the manipulated variable. The error signal ($e_i = h_i^{SP} - h_i$) is the difference between the level setpoint (h_i^{SP} or SP) and the measured pulp level (h_i) for each cell. The manipulated variable is adapted around a base point $u_{base} = 0.5$, thus the control action from the PI controller is limited to between -0.5 and 0.5 to keep the manipulated variable in the required range of 0 to 1. The PI controller combines proportional action required for a fast response and integral action required for zero steady-state offset to obtain the control signal sent to the plant. To control the pulp levels in the flotation bank a PI controller is added for each cell. The individual PI controllers from (5.1) get combined into the SISO controller structure for the full plant that is given by

$$\mathbf{C_{PI}} = \begin{bmatrix} c_{11} & 0 & 0 & 0 & 0 & 0 \\ 0 & c_{22} & 0 & 0 & 0 & 0 \\ 0 & 0 & c_{33} & 0 & 0 & 0 \\ 0 & 0 & 0 & c_{44} & 0 & 0 \\ 0 & 0 & 0 & 0 & c_{55} & 0 \\ 0 & 0 & 0 & 0 & 0 & c_{66} \end{bmatrix}. \tag{5.3}$$

The initial tuning of these controllers have to be found for the robust stability analysis used to define the bounds in which the BO automatic tuner should function. The initial tuning is found using step tests from the valve position to the level. The step test data is used to calculate SIMC PI parameters (Skogestad, 2003).

5.3.2 Step tests

The step test models are obtained by stepping the valve position and measuring the level response. The valve is stepped from the steady state given in Tables 4.1 and 4.2. The valve position is stepped by 0.1 from 0.5 to 0.6, which equates to 10% of the valve's operating range. The resulting level responses for each of the cells are shown in Fig. 5.1. The level responses are a mix of fast and slow dynamics and will typically require a higher order model to accurately capture the step response. The gain from valve to level is seen to be negative and corresponds to the expectation that if the valve is opened more the level will decrease as the flow out of the cell increases and the flow into the cell does not change. The mixture of fast and slow dynamics can be seen as the level very quickly decreases in the first 5 minutes and then very slowly reaches a new steady state condition. The level responses were fitted by multiple forms of transfer function models with varying numbers of poles and zeros. The final model chosen contains two poles and one zero. The transfer functions models are given in time constant form

$$G_{ij}(s) = k_p \frac{(\frac{1}{z_1}s+1)}{(\frac{1}{p_1}s+1)(\frac{1}{p_2}s+1)}, \tag{5.4}$$

with the time constants in hours, where the subscript ij indicates from which input (i) to which output (j) the model is constructed. The models are obtained as

$$G_{11} = -2.51\frac{(\frac{1}{10.3}s+1)}{(\frac{1}{166.3}s+1)(\frac{1}{7.1}s+1)} \tag{5.5a}$$

$$G_{22} = -2.51\frac{(\frac{1}{14.8}s+1)}{(\frac{1}{237.1}s+1)(\frac{1}{6.8}s+1)} \tag{5.5b}$$

$$G_{33} = -2.51\frac{(\frac{1}{17.6}s+1)}{(\frac{1}{459.2}s+1)(\frac{1}{6.4}s+1)} \tag{5.5c}$$

$$G_{44} = -2.51\frac{(\frac{1}{18.2}s+1)}{(\frac{1}{488.3}s+1)(\frac{1}{6.4}s+1)} \tag{5.5d}$$

$$G_{55} = -2.50\frac{(\frac{1}{17.0}s+1)}{(\frac{1}{286.0}s+1)(\frac{1}{7.2}s+1)} \tag{5.5e}$$

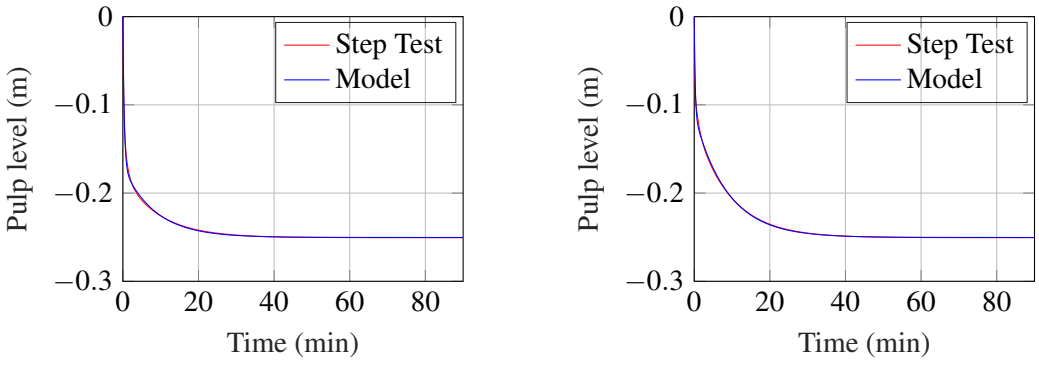
$$G_{66} = -3.22\frac{(\frac{1}{11.0}s+1)}{(\frac{1}{134.4}s+1)(\frac{1}{7.8}s+1)}. \tag{5.5f}$$

The fit of the models are determined by computing the integrated absolute error (IAE) over the period from 0 to 90 minutes. The maximum error over the period is also obtained. The goodness of fit is also

computed using the normalized root mean squared error (NRMSE) as

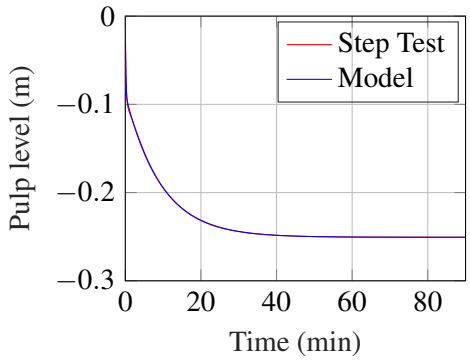
$$fit = \frac{||h_i^{SP} - h_i||}{||h_i^{SP} - mean(h_i^{SP})||} \times 100. \tag{5.6}$$

The model errors for all 6 cells can be seen in Table 5.1.

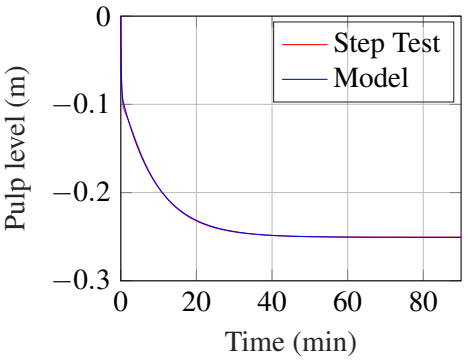


(a) Cell 1 Step Response.

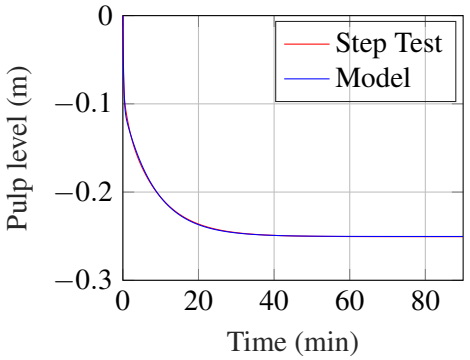
(b) Cell 2 Step Response.



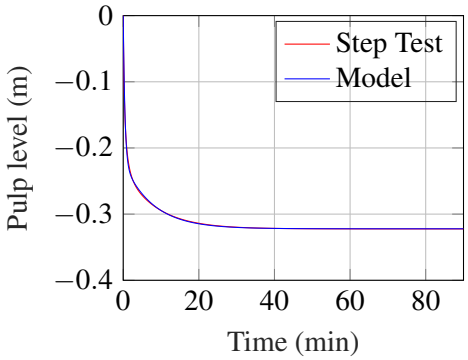
(c) Cell 3 Step Response.



(d) Cell 4 Step Response.



(e) Cell 5 Step Response.



(f) Cell 6 Step Response.

Figure 5.1. Individual cell step tests.

Table 5.1. Step test model fit errors.

	IAE	Error Max (mm)	Goodness of Fit %
Cell 1	1.33	6.86	94.85
Cell 2	1.30	7.48	96.60
Cell 3	0.73	1.72	98.58
Cell 4	1.03	2.50	98.00
Cell 5	1.16	6.32	97.23
Cell 6	1.26	6.13	96.17

5.3.3 Model reduction

The models in (5.5) are reduced to first order plus time delay (FOPTD) models of the form

$$G(s) = \frac{ke^{-\theta s}}{\tau_p s + 1}, \tag{5.7}$$

where k is the system gain, τ_p is the time constant, and θ is the time delay. This is done to obtain the tuning parameters via the SIMC tuning rules.

The models are reduced to only contain the fastest pole (the smallest time constant) and steady state gain. This is different to the Skogestad's half rule for model reduction where the largest time constant in the denominator is kept and the second largest time constant in the denominator is split into half, adding one half of it to the time constant that is kept and adding the other half as a delay. The half rule deals with right half plane zeros by adding them to the delay as well. However, left half plane zeros are cancelled out by the pole closest to the left half plane zero, where certain rules apply to the way this cancellation is applied. The effect of using the half rule in this case is that the time constant in the approximated first order model will still be slower than the fastest time constant of the original model and thus, using the SIMC tuning rules, results in a controller too conservative for tight control. The decision was made to keep only the faster pole and the steady state gain. Having a tight controller as a starting point is important for both the performance comparison as well as the initial bounds obtained from the initial controller as per Section 5.4. If the initial bounds given to the robust stability analysis is too conservative, the analysis might not expand the bounds by enough and the optimum might be excluded from the range.

The reason for requiring tight control is that the initial response of the level is quite fast and the controller will have to be adequately fast to obtain good performance. The controller will need to act to the initial level change before the dynamics of the slower pole and zero influences the response.

The reduced models, with time constants in hours, are obtained as

$$G_{11reduced} = \frac{-2.51}{(\frac{1}{166.3}s+1)} \tag{5.8a}$$

$$G_{22reduced} = \frac{-2.51}{(\frac{1}{237.1}s+1)} \tag{5.8b}$$

$$G_{33reduced} = \frac{-2.51}{(\frac{1}{459.2}s+1)} \tag{5.8c}$$

$$G_{44reduced} = \frac{-2.51}{(\frac{1}{488.3}s+1)} \tag{5.8d}$$

$$G_{55reduced} = \frac{-2.50}{(\frac{1}{286.0}s+1)} \tag{5.8e}$$

$$G_{66reduced} = \frac{-3.22}{(\frac{1}{134.4}s+1)}. \tag{5.8f}$$

The corresponding time constants, in minutes, are given in Table 5.2. The conversion to minutes is done to make the time constants easier to work with and put it into perspective with the plots showing the dynamics.

Table 5.2. Reduced models time constants.

	τ (min)
Cell 1	0.362
Cell 2	0.255
Cell 3	0.130
Cell 4	0.122
Cell 5	0.211
Cell 6	0.448

5.3.4 SIMC settings

The initial controller settings are found from the open loop step tests and transfer function models in Sections 5.3.2 and 5.3.3 where the valve positions are manipulated. The controller settings are obtained using SIMC tuning rules as given in Table 5.4. The value for τ_c is chosen as $\tau_c = 0.1\tau_p$ (min)

to provide a good trade-off between setpoint tracking and disturbance rejection (Seborg et al., 2019). The initial controller settings are given in Table 5.5.

5.3.5 Forward feeding control

Given the model in Chapter 4 a highly interactive system is expected. These interactions can be mitigated through more advanced controllers such as forward feeding (FF) controllers, where each valve position is passed to the valves downstream. The basic control element is still the PI controllers as defined in (5.1), however the structure of the combined controller changes to

$$\mathbf{C_{FF}} = \begin{bmatrix} c_{11} & 0 & 0 & 0 & 0 & 0 \\ c_{11} & c_{22} & 0 & 0 & 0 & 0 \\ c_{11} & c_{22} & c_{33} & 0 & 0 & 0 \\ c_{11} & c_{22} & c_{33} & c_{44} & 0 & 0 \\ c_{11} & c_{22} & c_{33} & c_{44} & c_{55} & 0 \\ c_{11} & c_{22} & c_{33} & c_{44} & c_{55} & c_{66} \end{bmatrix}. \tag{5.9}$$

This form of the controller is found in Schubert et al. (1995) and Kämpjärvi and Jämsä-Jounela (2003) where knowledge of the plant is incorporated into the controller design. From the structure in (5.9) it can be seen that the controller structure is a bottom lower triangular matrix. This is because the valve position is only fed forward to cells downstream of itself. One can also see that all non-zero elements in a column are the same as individual PI controllers are not added for each interaction. There are still only 2 parameters to tune per column of the controller as feeding forward the valve position corresponds to copying the individual controllers column-wise down the controller. Therefore, the control signal from each controller is added to the control signals of the downstream controllers. The effects of this is that the PI controller of the next cell does not have to adapt to the changes in the flow

Table 5.3. FOPTD model parameters.

Cell	k	τ_p (min)
$i = 1$	-2.51	0.362
$i = 2$	-2.51	0.255
$i = 3$	-2.51	0.130
$i = 4$	-2.51	0.122
$i = 5$	-2.51	0.211
$i = 6$	-3.22	0.448

Table 5.4. SIMC tuning rules (Skogestad, 2003).

Model	$G(s)$	k_c	τ_I
FOPTD	$\frac{ke^{-\theta s}}{\tau_p s+1}$	$\frac{1}{k}\frac{\tau_p}{\tau_c+\theta}$	$\min[\tau_p, 4(\tau_c+\theta)]$

Table 5.5. SIMC settings for each of the tanks.

Cell	k_c	τ_I (min)
$i=1$	-4.00	0.145
$i=2$	-4.00	0.102
$i=3$	-4.00	0.052
$i=4$	-4.00	0.048
$i=5$	-4.00	0.0845
$i=6$	-3.10	0.179

generated by the control from the previous cell as this control is already added to the valve position by the FF control. This method can be seen in Kämpjärvi and Jämsä-Jounela (2003).

5.4 ROBUST STABILITY ANALYSIS & CONTROLLER BOUNDS FOR SISO PI CONTROL

The SIMC settings are used to define initial bounds for the automatic tuning procedure. The initial lower bound of the gain (k_c) parts of the controllers are set to a factor of 2 of the SIMC value. The initial upper bounds are selected in the opposite direction as a factor of 0.2 of the initial settings. The integral time-constant (τ_I) bounds are selected inversely to the proportional part, with the upper bound selected as a factor of 0.5 of the SIMC settings and a factor of 5 in the opposite direction. The initial parameter uncertainty is then given by

$$K_{c11} \in [-8.0, -1.6] \tag{5.10a}$$

$$\tau_{I11} \in [0.0725, 0.725] \tag{5.10b}$$

$$K_{c22} \in [-8.0, -1.6] \tag{5.10c}$$

$$\tau_{I22} \in [0.051, 0.51] \tag{5.10d}$$

$$K_{c33} \in [-8.0, -1.6] \tag{5.10e}$$

$$\tau_{I33} \in [0.026, 0.26] \tag{5.10f}$$

$$K_{c44} \in [-8.0, -1.6] \tag{5.10g}$$

$$\tau_{I44} \in [0.025, 0.245] \tag{5.10h}$$

$$K_{c55} \in [-8.0, -1.6] \tag{5.10i}$$

$$\tau_{I55} \in [0.042, 0.423] \tag{5.10j}$$

$$K_{c66} \in [-6.2, -0.62] \tag{5.10k}$$

$$\tau_{I66} \in [0.0694, 0.694]. \tag{5.10l}$$

The intervals in (5.10) are only initial ranges provided to the robust stability analysis. The robust stability analysis will either expand or contract the range until the desired stability margins are reached. Thus, these ranges are only used to start the analysis. The bounds in Table 5.6 are the final bounds computed as per Section 3.4.

The closed-loop system including the controller of the form shown in (5.3) with parameter uncertainty as per (5.10) is subjected to a robust stability μ-analysis (Skogestad and Postlethwaite, 2007), where the minimal bounds for closed-loop stability are found using the MATLAB Robust Control Toolbox. The models obtained in the initial step tests in Section 5.3.2 can be used for the robust stability analysis in the case where no other suitable models are available. The bounds are given in Table 5.6. The μ-analysis tests if the system is closed-loop stable given the uncertainty in the controller, as per Section 3.4. The analysis also expands the bounds if the μ-analysis finds that there is room for expansion while still keeping $\mu < 1$. In this case the initial bounds were more conservative than needed and thus the analysis expanded the ranges for all k_c and τ_I values.

Table 5.6. Final SISO bounds for BO.

Cell	k_c		τ_I (min)	
	Lower	Upper	Upper	Lower
$i = 1$	-8.980	-0.202	5.463	0.0616
$i = 2$	-8.980	-0.202	3.843	0.0433
$i = 3$	-8.980	-0.202	1.959	0.0221
$i = 4$	-8.980	-0.202	1.8460	0.0208
$i = 5$	-8.980	-0.202	3.183	0.0306
$i = 6$	-6.960	-0.157	5.227	0.0589

The result can be visualized through the structured singular value (μ) plot of the system shown in Fig. 5.2. The frequency dependent μ-value is obtained as given in (3.50). The μ-value is smaller than 1 for all frequencies implying the closed-loop system is stable for all possible PI parameter values within the bounds provided in Table 5.6, provided that the plant is approximately linear at the operating condition around which it was linearized. These bounds are subsequently used to bound the BO tuner.

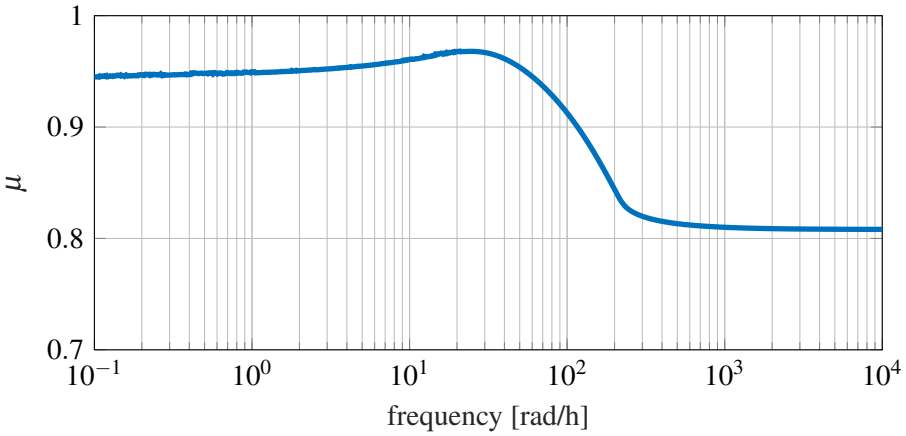


Figure 5.2. Maximum structured singular values (μ) for the constraints given in Table 5.6.

5.5 SIMULATION

The SIMC controller settings and bounds are provided to the BO tuner. Closed-loop step tests are conducted one cell at a time starting with the first cell in the series. The BO is conducted on each cell using the step tests to perturb the system and minimize the settling time of each cell level response to the change in its own valve position. In other words, only a single cell is being tuned at a time. The BO iteratively changes the controller settings as discussed in Chapter 3 after each step test. The cost function in (5.11) returns the performance of the current settings. The returned cost function value is used in the acquisition function to suggest the next set of controller parameters. For this application the cost function is chosen as the settling time (T_s) of the cell level of the cell being tuned. This provides an almost critically damped response which also minimizes the dynamics caused in the other cells by the step in the current cell. The cost function for a single cell is given by

$$Q_i = T_s(\boldsymbol{\alpha}), \tag{5.11}$$

where $\boldsymbol{\alpha}$ is the vector of the proportional and integral gain of the controller. Each controller is tuned considering only its own settling time. However, the whole structure is evaluated using performance metrics as per Section 5.5.2 at the end of the tuning.

5.5.1 Iterative approach to controller tuning

Given the iterative nature of BO, the controllers will be tuned iteratively as well. Thus, each iteration the automatic tuner will:

- select new controller tuning parameters,
- perturb the system,
- collect performance data (settling time of the cell that is perturbed), and
- append the new data to the dataset.

Closed-loop reference steps to the level reference signals are selected as the perturbation method. This generates information on both the effect of the controller on its own level as well as the effect of the controller on the other cell levels. The drawback of this perturbation method is that it limits the automatic tuner to perturbing one cell at a time, as distinguishing between the effects of different cells being stepped at the same time will be difficult. The benefit of this perturbation method is that it can easily be applied in closed-loop without any change to the structure of the control-loop. Therefore, the BO automatic tuner can sit on the process without making any structural changes to the original system.

5.5.2 Performance evaluation

The SIMC and BO tuned controllers are evaluated according to three different performance metrics: the Integral of the Absolute Error (IAE), the Integral Squared Error (ISE) and the Integral of the Time Absolute Error (ITAE) (Seborg et al., 2019). These metrics are defined as

$$ISE(i) = \int_0^T e_i^2(t)dt \tag{5.12a}$$

$$IAE(i) = \int_0^T |e_i(t)|\,dt \tag{5.12b}$$

$$ITAE(i) = \int_0^T t\,|e_i(t)|\,dt, \tag{5.12c}$$

where i refers to the specific cell evaluated and T is the time period across which the error signal is evaluated. The final performance metrics is the summation of all the different cell performances. A different metric used to define the impact of the tuning on the actuator is the Total Variation (TV) of the manipulated variable, in this case the valve position. The TV is defined as

$$TV = \sum_{i=1}^{6}\sum_{k=1}^{T} |u_i(k) - u_i(k-1)|\,, \tag{5.13}$$

where k indicates the sample indices. A higher TV will indicate more actuator usage which may be detrimental to the physical actuator.

5.6 RESULTS

The BO tuner is allowed 20 iterations (i.e., 20 step tests with a duration of 30 seconds each) to find the controller settings as this limit was found sufficient for the BO to reach the optimal value. These

Table 5.7. BO controller settings.

Cell	k_c	τ_I (min)	**Cell**	k_c	τ_I (min)
$i = 1$	−8.08	0.852	$i = 4$	−8.08	0.444
$i = 2$	−8.08	0.409	$i = 5$	−8.08	0.338
$i = 3$	−7.66	0.538	$i = 6$	−6.20	0.565

settings are applied to the PI controllers with and without FF control and compared to the performance of the SIMC settings with and without FF control. The BO settings are obtained without the FF control in the loop. As seen from Table 5.7 some of the optimal settings are found to be on the limit of the allowed boundaries, thus fast aggressive controllers are expected. The BO automatic tuner finds very similar settings for the different cells regardless of the position of the cell in the bank. Cell 3 and 6 deviate the most. This is expected, for cell 3 detuning makes sense as cell 3 is one of the middle cells experiencing more interaction with other cells. Cell 6 deviates from the rest as the flow out of cell 6 is different from the rest, only depending on its own valve position and pulp level. The similar controller settings throughout the bank is expected as the same cost function is used in the tuning. The cell dynamics should mostly also be uniform from what is defined in the simulation model. To see how the BO automatic tuner arrives at these settings the iterations through the bank needs to be discussed.

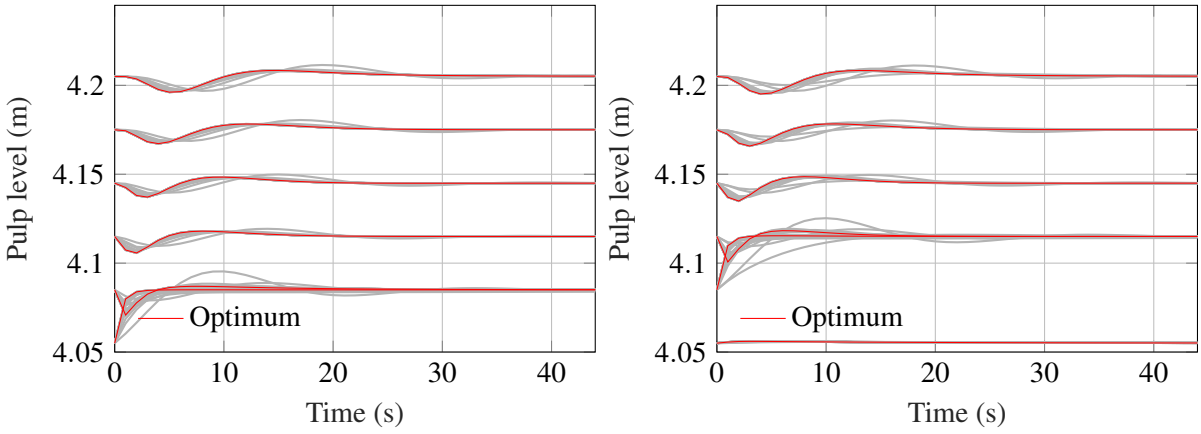
5.6.1 Iteration convergence and efficiency

The matter of iteration convergence and efficiency is addressed in this section. Convergence in this case implies that after a step change, the levels either return to their setpoint or settle to a new steady-state. The iteration convergence of the BO automatic tuner can be confirmed with the responses generated by the iterative step tests that are conducted when the different tuning parameters are trialed. All the parameter combinations trialed by the automatic tuner resulted in responses that converge to the setpoints as seen in Figs. 5.3-5.5. The responses generated during the iterations range from under-damped to over-damped. However the BO automatic tuner quickly identifies the under- and

over-damped responses as less optimal through the cost function and steers the parameter choice to a more critically damped region. This also confirms the validity of the choice of cost function.

In Fig. 5.3 the iterations of the first two cells show that the system is indeed highly interactive, with a step in the level of upstream cells propagating to the last cell as well. The is both due to the change in levels and change in valve positions to mitigate the change in flow caused by the first cell level reference change. Although the BO automatic tuner finds the optimal controller tuning values for the required level response from the valve to its own level, it does little to minimize the disturbance sent to the downstream cells. This is to be expected as a diagonal controller is used and the controllers are tuned only on data from the cell's own level. As the BO automatic tuner makes its way down the bank of cells it can be seen that the aggressively tuned cells mitigate the expected push back from the down stream levels that are changing as the other cells are tuned. The tuner finds similar controller parameters for the other cells in the bank. The bank is tuned on setpoint changes to exhibit a critically damped response. However, controllers on actual processes will mostly work at rejecting disturbances. These disturbances are typically unmeasured and largely unpredictable. Thus, one will not be able to tune the controllers using disturbances, but the disturbance response can be used to quantify the performance of the controller settings after tuning has taken place.

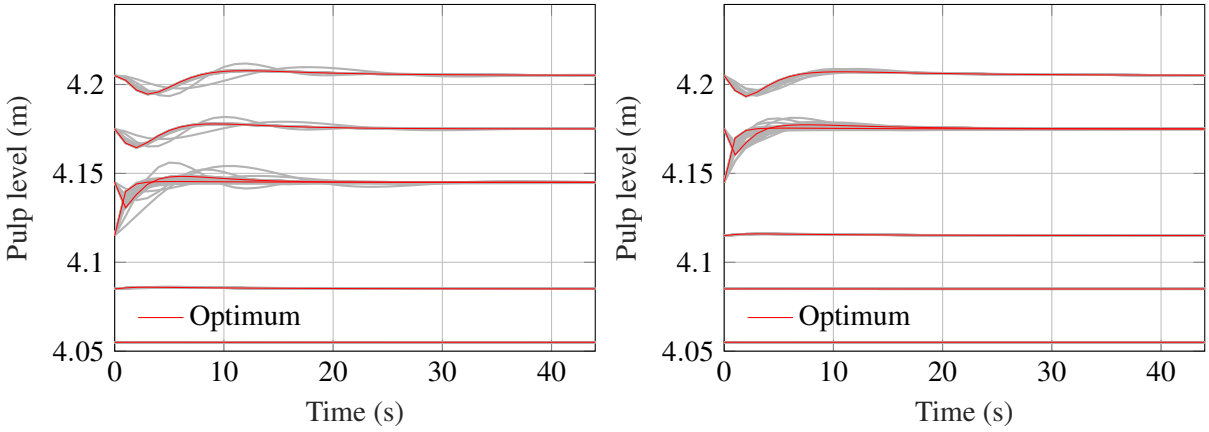
The way in which the BO automatic tuner explores the search space is further visualized through tracing the objective function values as the iterations progress. The objective function values are



(a) Cell 1.

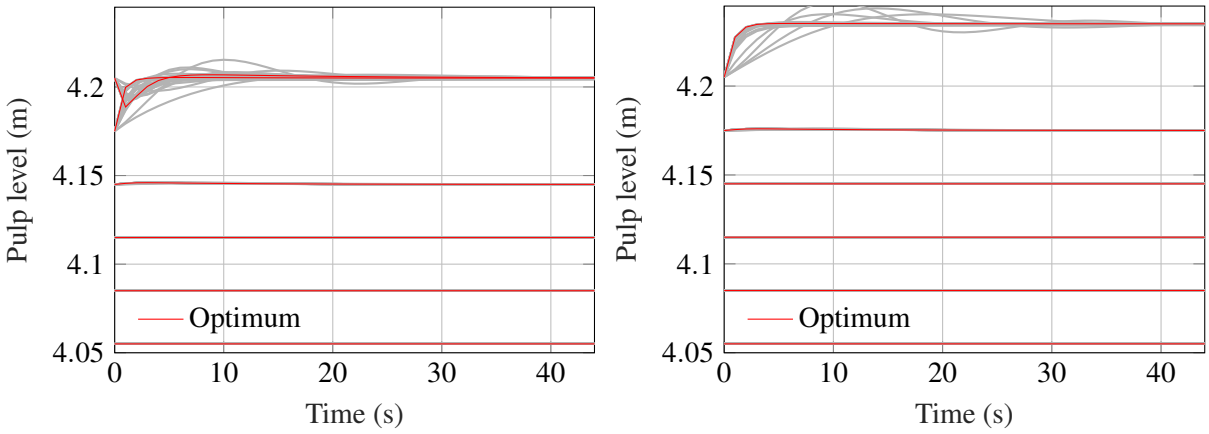
(b) Cell 2.

Figure 5.3. SISO BO iterations cell 1 and 2.



(a) Cell 3. **(b)** Cell 4.

Figure 5.4. SISO BO iterations cell 3 and 4.



(a) Cell 5. **(b)** Cell 6.

Figure 5.5. SISO BO iterations cell 5 and 6.

normalized by dividing by the objective function value produced by the SIMC settings. As seen in Fig. 5.6 the objective function values range between 0 and 8 where cell 1 and 6 have the most variability. For the iterations where the pulp level did not settle in time the iteration is discarded. The objective function values quickly settle between 0 and 1, however spikes can be seen where the BO automatic tuner explores the parameter space but is quick to return to parameters that produce a normalized objective function value of between 0 and 1. The tuner takes the most iterations to settle when tuning cell 1 and cell 6, and explores the parameter space at the end of tuning cell 6. This produces the large objective function values in the later iterations of tuning cell 6. The number of iterations in

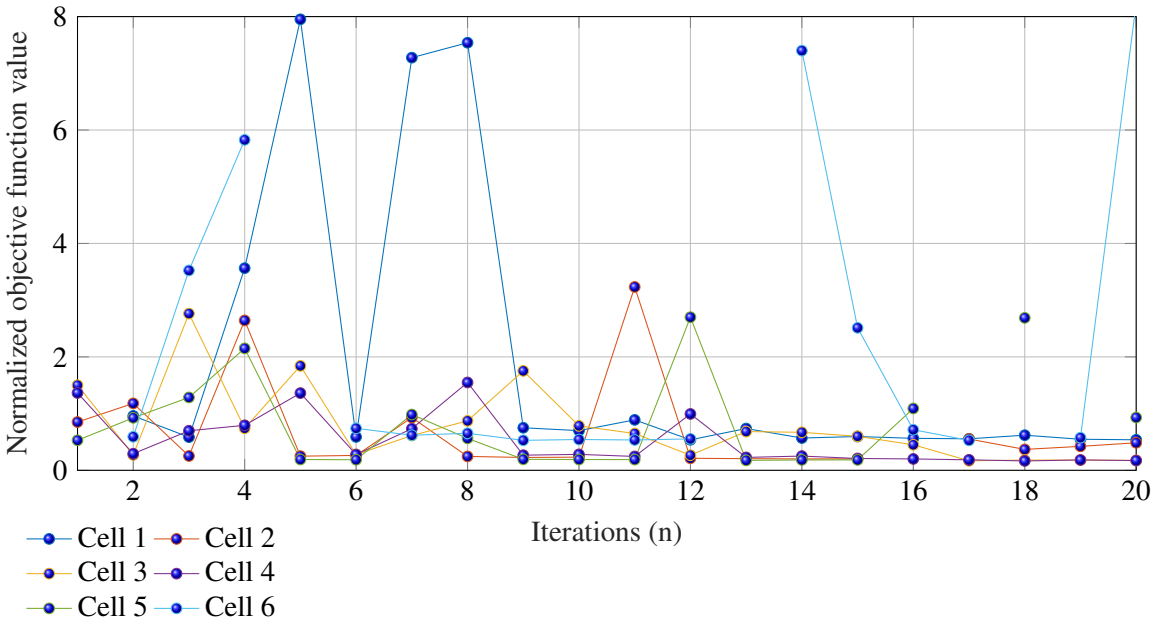


Figure 5.6. Normalized objective function value trace.

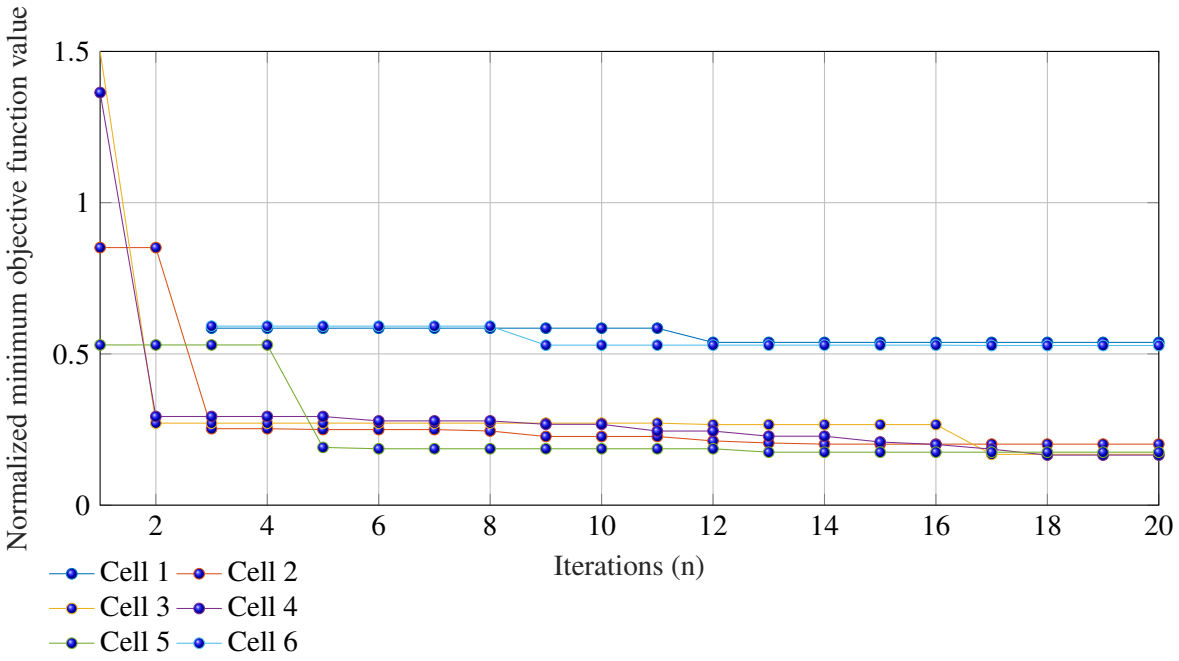


Figure 5.7. Normalized minimum objective function value trace.

which the BO automatic tuner finds the minimum can be visualized by plotting the minimum objective function value after each iteration. This is shown for each cell in Fig. 5.7. Cell 1 and 6 take the most iterations to find the minimum, these two cells also show the least improvement over the SIMC

settings as seen in the normalized objective function values of just above 0.5. The traces for the other cells show that the automatic tuner quickly finds parameters that reduce the objective function value, but for most the minimum is only found later. This shows that the BO automatic tuner could benefit from an exit function, to exit the optimization once the automatic tuner reaches a certain performance improvement.

5.6.2 SISO SIMC and BO controllers without forward feeding control

To evaluate the newly tuned controllers the plant is subjected to a sequence of setpoint changes as well as a feed flow disturbance in Q_F, combined with spillage water in cell 3. For the setpoint changes, a 3 cm increase in the level set-point of each cell is made, with the last cell being stepped first. The feed flow disturbance is a -20% step in Q_F and constant spillage water of 75 m^3/h in cell 3 from time $t = 100$ s. The results for the SIMC and BO controllers without the use of FF control are shown in Figs. 5.8 and 5.10. The BO tuned PI controllers provide a significant improvement over the SIMC settings. The error peaks are reduced the most as can be seen from the ISE in Tables 5.8 and 5.9. The time the errors persist also shows improvement as seen in the ITAE. In both the setpoint tracking and disturbance rejection results the BO settings show a much more critically damped response compared to the SIMC settings, in line with minimizing the settling time for each SISO PI controller. The faster, more aggressively tuned BO controllers reduce the effect of the interactions between the cells, reacting to the interactions between the cells much quicker. However, the interactions between the cells are still apparent, the downstream cells are affected the most. It is also seen that the BO controller has a much higher actuator usage compared to the SIMC controller as seen from the TV metric as well as Figure 5.9 and 5.11. The valve deviates much further for the steady-state value in a shorter time, this could be detrimental to the physical actuator doing the work.

Table 5.8. Setpoint change performance.

Performance Metric	**Controller**		
	BO	SIMC	Improvement
IAE	1409	2714	48%
ISE	637	1491	57%
ITAE	468705	907043	48%
TV	3.17592	1.95714	-62%

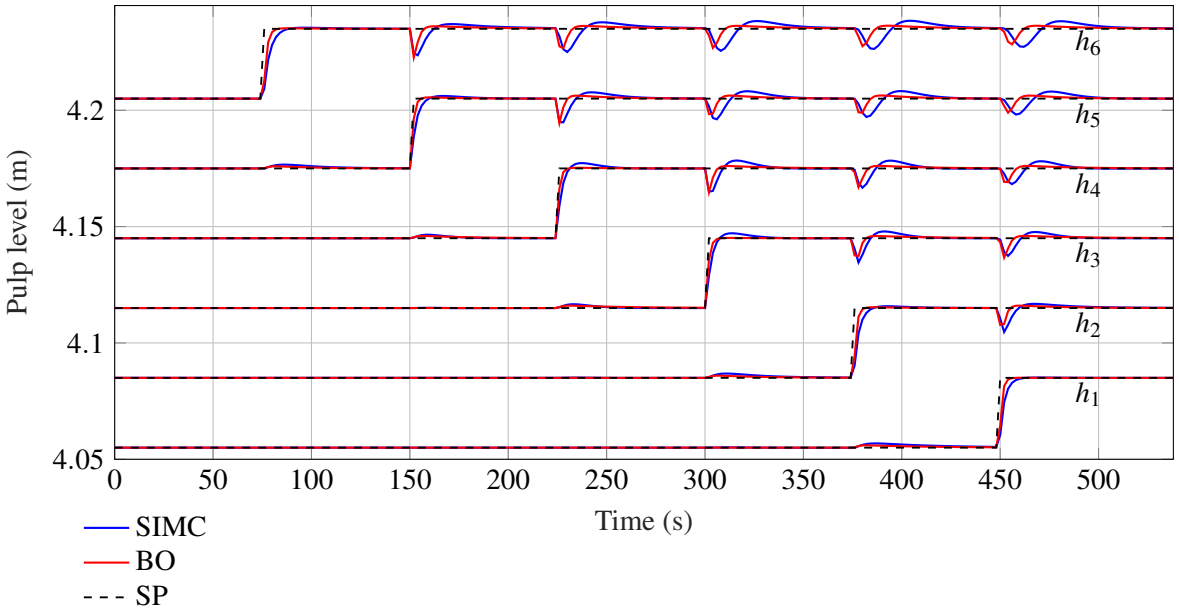
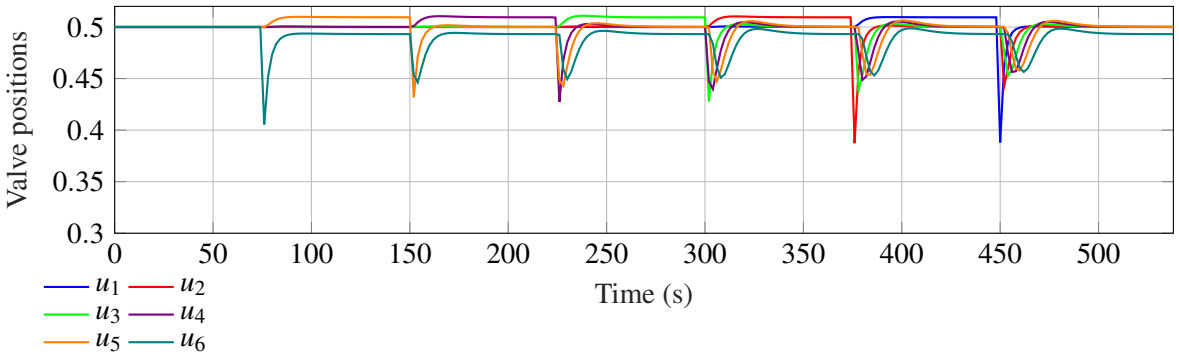
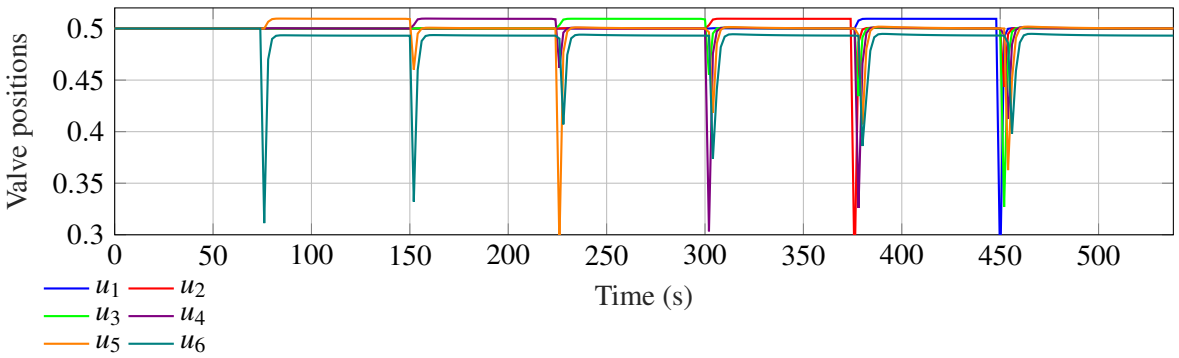


Figure 5.8. Setpoint changes with PI only.



(a) SIMC manipulated variable action.



(b) BO manipulated variable action.

Figure 5.9. Manipulated variable action for setpoint changes with PI only.

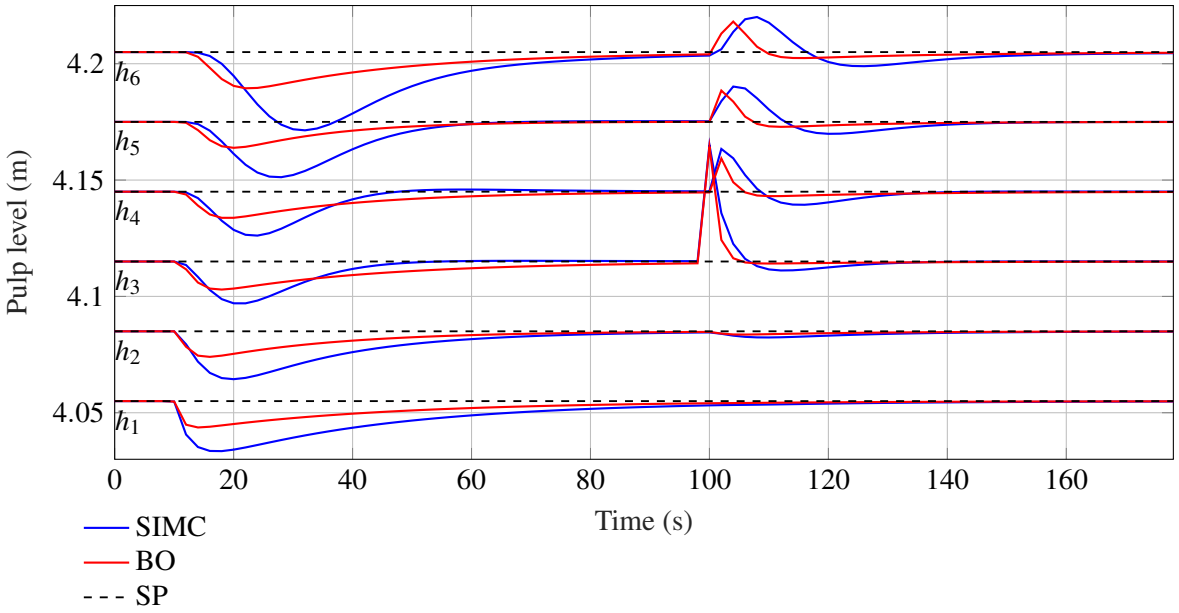
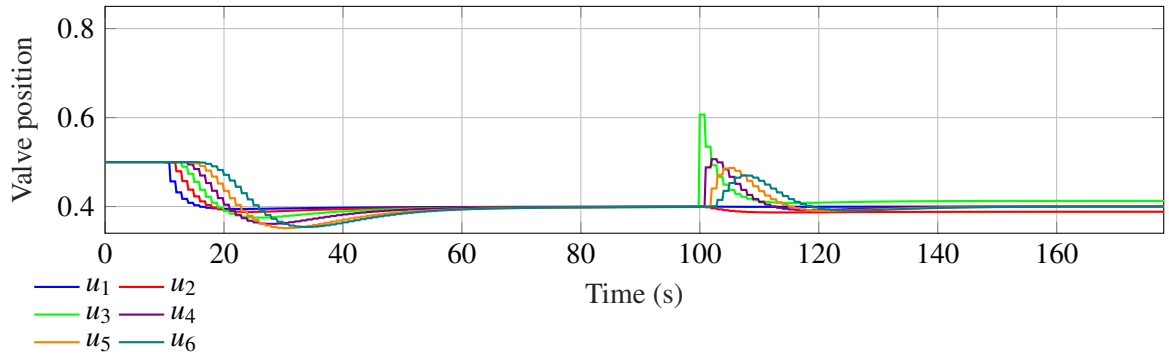
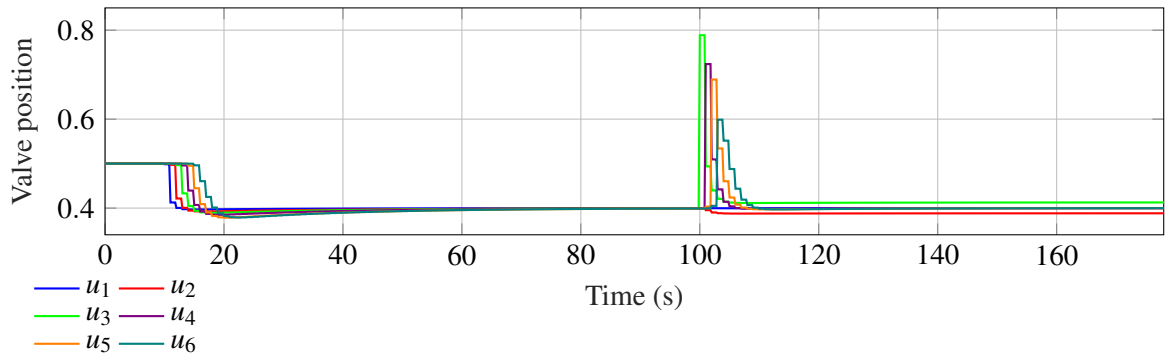


Figure 5.10. Disturbance rejection with PI only.



(a) SIMC manipulated variable action.



(b) BO manipulated variable action.

Figure 5.11. Manipulated variable action for disturbance rejection with PI only.

Table 5.9. Disturbance rejection performance.

Performance Metric	Controller		
	BO	SIMC	Improvement
IAE	2652	4603	42%
ISE	1983	6684	70%
ITAE	132173	239699	44%
TV	7.66345	3.06252	-150%

The setpoint change in one cell will act as a disturbance to the other cells as a change in flow will occur when the valve changes position. This disturbance can be seen to propagate from the second cell all the way down to the final cell when the first cell's setpoint is changed and the controller changes the valve position to get the pulp level to the new setpoint. The change in flow will propagate down the bank causing the levels to change. To reduce the effects of this series coupling between cells FF control can be added.

5.6.3 SISO SIMC and BO controllers with FF control

The plant under PI control with the added FF control is subjected to the same performance evaluation as the PI only case. In both cases, SIMC and BO, the interactions between the cells can be further reduced with simple FF control as described in Section 5.3.5. The effect this has on the effectiveness of the controller settings is shown in the performance comparison of the SIMC and BO settings in both Tables 5.10 and 5.11, as well as Figs. 5.12 and 5.14. The FF control is applied without retuning the PI controllers. Retuning is not needed as the cells and valve responses are mostly uniform. If the cells were different the off-diagonal might be require tuning when adding FF control. However, a better option is to use a multivariable controller and tune the off-diagonal elements separately from the diagonal control.

The BO settings still provide an improvement over the SIMC settings. The improvement is reduced from where no FF control was implemented. The error peaks are still the most improved as shown in the ISE, while the time taken to reduce the errors still shows significant improvement as per the ITAE. The BO controller again has much higher actuator usage compared to the SIMC controller. The results for the SIMC and BO controllers with FF control are shown in Figs. 5.12 and 5.14. The manipulated variable actions for the results are shown in 5.13 and 5.15.

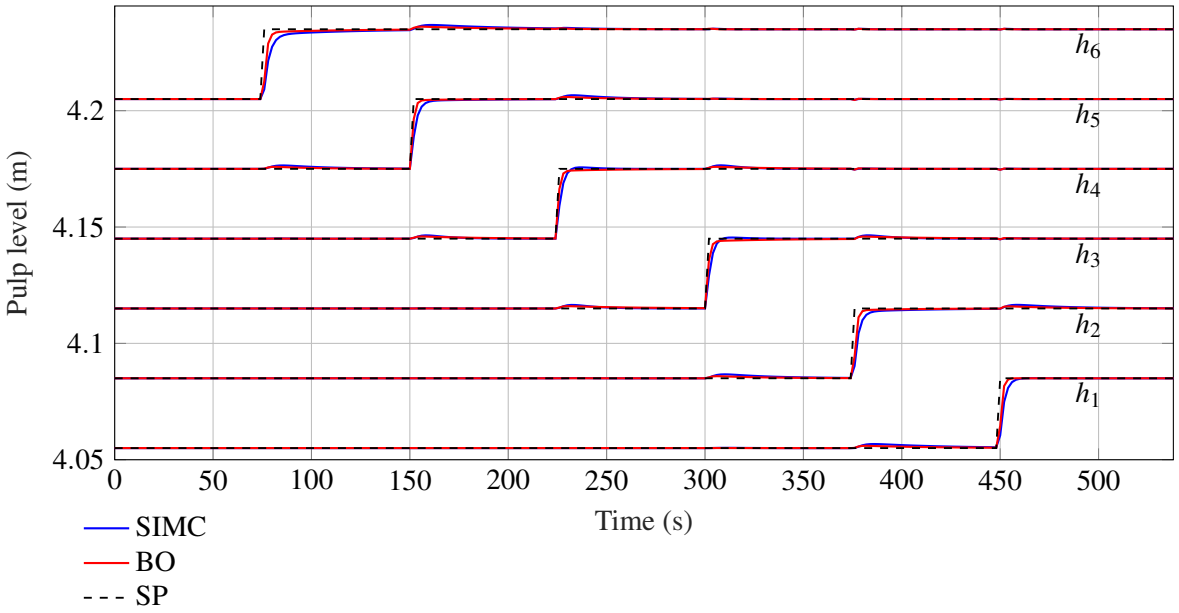
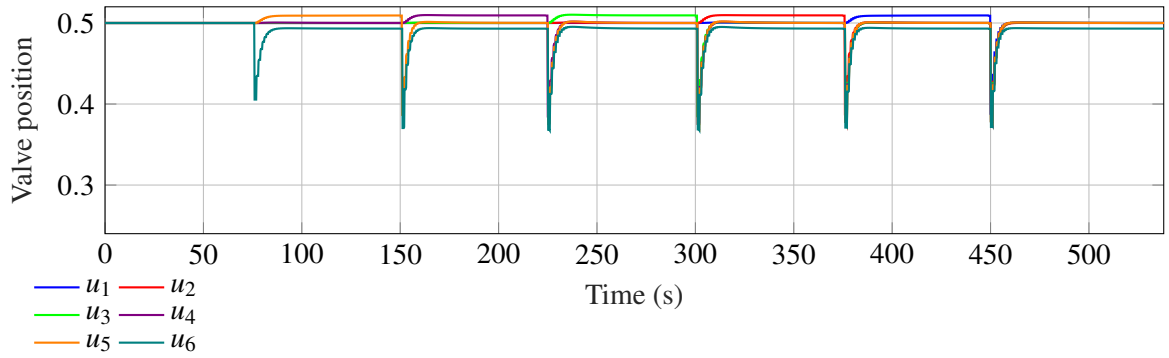
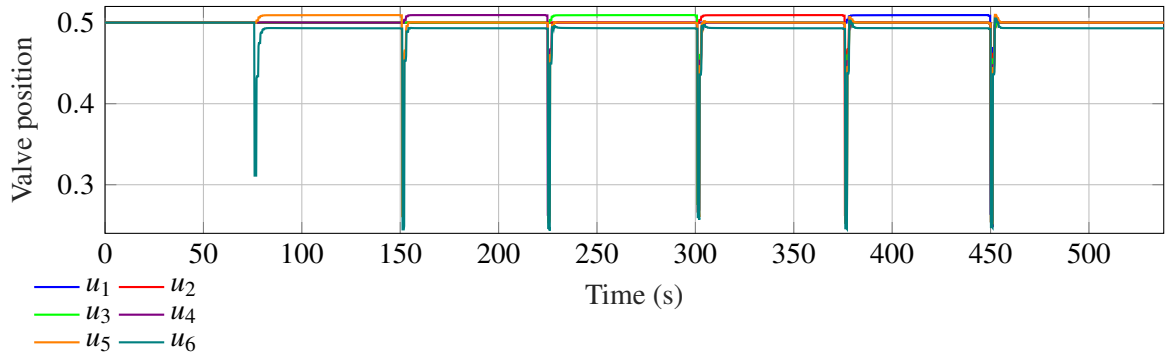


Figure 5.12. Setpoint changes with PI and FF.



(a) SIMC manipulated variable action.



(b) BO manipulated variable action.

Figure 5.13. Manipulated variable action for setpoint changes with PI and FF.

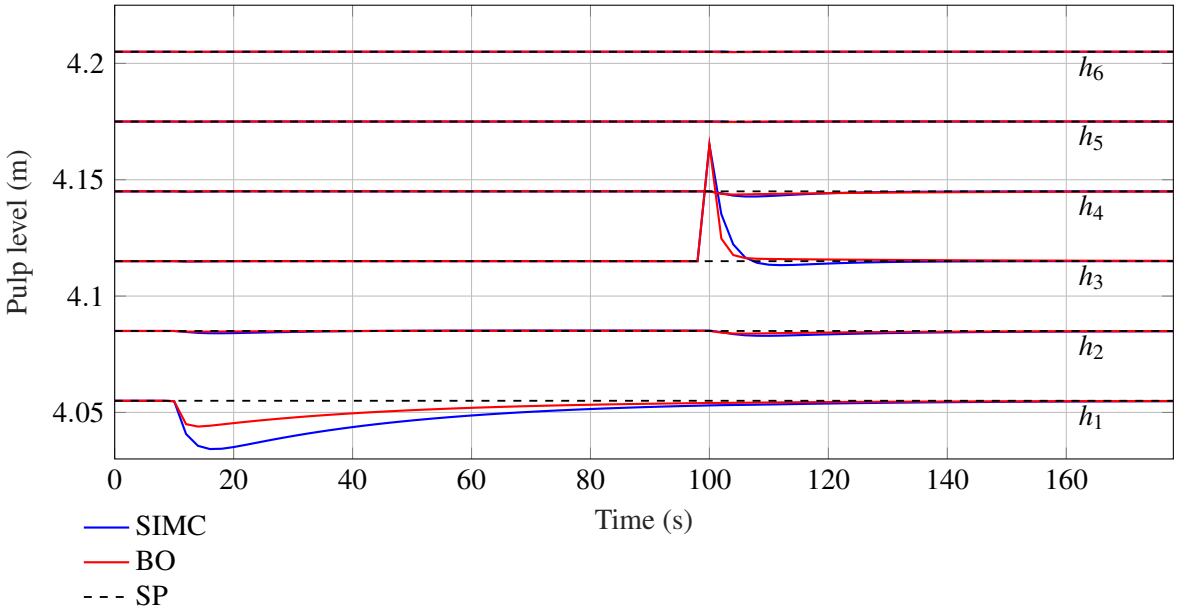
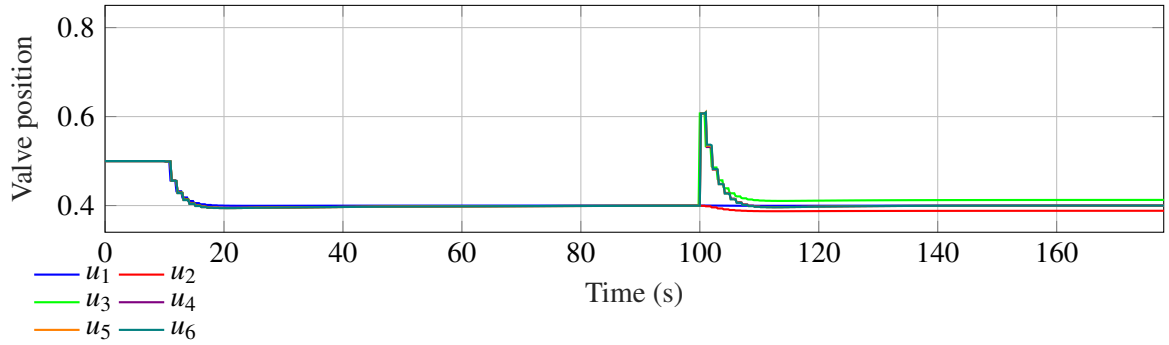
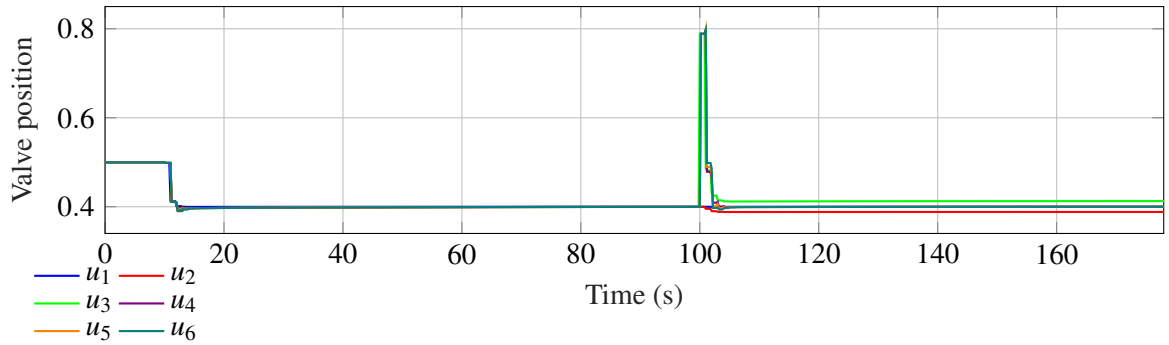


Figure 5.14. Disturbance rejection with PI and FF.



(a) SIMC manipulated variable action.



(b) BO manipulated variable action.

Figure 5.15. Manipulated variable action for disturbance rejection with PI and FF.

Table 5.10. Setpoint change performance with FF.

Performance Metric	**Controller**		
	BO	SIMC	Improvement
IAE	659	1058	37%
ISE	294	655	55%
ITAE	178623	281258	36%
TV	3.86517	2.35826	-64%

The improvement the FF control provides is best seen in the disturbance rejection of both SIMC and BO settings. The BO again has a much higher TV showing that the actuator usage of the BO controller compared to the SIMC controller is much higher. The higher usage again occurs in a shorter time than compared to the SIMC controller. The interactions between cells are greatly reduced, similar to the effect that can result from a well-designed multi-variable controller. Care should however be taken when using FF control to ensure that the plant model on which the controller is based is accurate. Otherwise FF control could degrade closed-loop performance (Seborg et al., 2019).

The BO settings provide a significant improvement on the SIMC settings with regards to the ITAE, ISE and IAE performance metrics. The SIMC settings are used as a departure point and the bounded parameter space is searched to minimize the cost function to provide improved controller settings. This improvement comes at the cost of an increased number of step tests to conduct. The number of iterations required can be reduced in a number of ways, such as: exit functions, combined step tests and combined controller tuning. The latter is of interest as seen in the fact that additional control like the FF control can significantly improve the performance of the both SIMC and BO controllers. The

Table 5.11. Disturbance rejection performance with FF.

Performance Metric	**Controller**		
	BO	SIMC	Improvement
IAE	591	1114	42%
ISE	428	1331	67%
ITAE	36631	64387	43%
TV	5.18115	10.38037	-100%

success of adding the FF control shows the need to expand the BO automatic tuner to a multivariable controller.

5.7 CHAPTER SUMMARY

The structure of the individual PI control elements, as well as the total controller used to control the level of the flotation bank is described. Step tests are conducted where the valve positions are stepped to find the level response. Transfer function models are fitted to the step responses, and are compared to the step response data to determine the goodness of the fit. These models are reduced to FOPTD models to obtain the SIMC controller settings for the diagonal controller. The SIMC settings are used to provide initial bounds for robust stability analysis. The robust stability analysis expands the initial bounds to find the final bounds in which the controllers will provide a stable closed-loop response. The bounds are passed to the BO automatic tuner that tunes the PI controllers using closed-loop setpoint changes. The iterations of the BO automatic tuner are presented and discussed. The performance of the BO settings is compared to the SIMC settings using different forms of the integrated error. The BO finds large improvements over the SIMC settings, but at the cost of an increased number of step test conducted as well as an large increase in actuator usage. FF control is added to mitigate interactions between cell levels and mitigate the effects of the change in flow when the valve positions change. The BO settings still obtain an improvement over the SIMC settings, although it is reduced slightly. The FF control is static and does not account for when the cells are not uniform. In Chapter 6 the BO automatic tuner is extended to tune the multivariable controller after the diagonal elements have been tuned and fixed as per Chapter 5.

CHAPTER 6 MIMO BO AUTOMATIC TUNING OF LEVEL CONTROLLERS

6.1 CHAPTER OVERVIEW

In this chapter the flotation model is controlled using a MIMO controller. The controller structure and initial tuning and BO bounds are given in Section 6.3. The initial tuning and bounds are found from the FF controller. The tuning strategies are presented in Section 6.4. The first adds controller elements as it progresses through the bank, while the other starts tuning with all controller elements in place. The results for both noise free and noisy simulations are presented and discussed in Section 6.4. The chapter is summarized in Section 6.5.

6.2 EXTENSION OF BO AUTOMATIC TUNER TO A INVENTORY CONTROLLER

In this Chapter the BO automatic tuner is extended to a multivariable controller similar to what is covered in Richter et al. (2024b). The controller structure is presented with the initial settings taken from the FF controller in the previous chapter. The multivariable structure is analyzed for robust stability and the bounds are provided to the BO automatic tuner. The procedure in which the multivariable inventory controller is tuned is presented. Two strategies to tune the controller is trialed. The best strategy is applied to a system with measurement noise on the simulated pulp levels. The performance of the BO settings for both tuning strategies is compared to the FF controller using the same performance evaluation as in the previous chapter. The performance of the BO settings tuned with measurement noise is also compared to the performance of the FF controller subjected to the same noise. The iteration convergence and efficiency is presented and discussed.

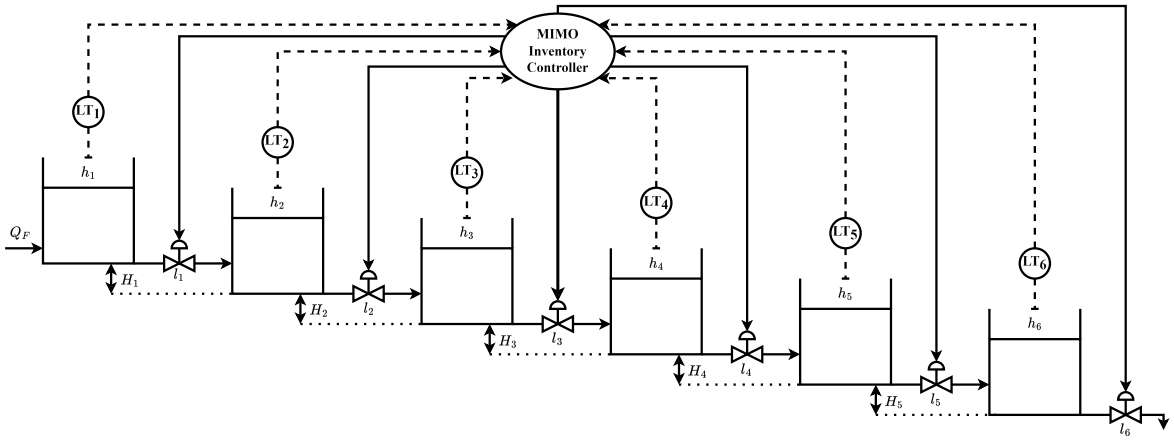


Figure 6.1. Flotation bank configuration with MIMO inventory controller.

6.3 CONTROLLER STRUCTURES, BOUNDS AND TUNING PROCEDURE

6.3.1 Controller structures

6.3.1.1 Feedback controller

The SISO controllers to control the plant in Fig. 4.1 are collected in matrix form as

$$\boldsymbol{C}_{PI} = \text{diag}([c_{11}, c_{22}, \ldots, c_{66}]), \tag{6.1}$$

where each controller has the form

$$c_{ii} = K_{Cii}\left(1 + \frac{1}{\tau_{Iii}}s\right). \tag{6.2}$$

The controllers are initialized with SIMC settings (Skogestad, 2003), and are tuned further using BO as in Chapter 5. The controller parameters obtained using BO are given in Table 6.1. Although the cells are uniform with the exception of the last cell, the interactions between the cells depend on their position in the bank. Thus, the optimum found by the BO automatic tuner for the SISO controllers might differ. The SISO controller settings and the linearized plant model of (4.7) are used to set up an initial parameter space for the μ-analysis in Section 6.3.2.

Table 6.1. Controller settings for SISO PI obtained using BO (see Section 5.6).

Cell	K_c	τ_I (min)	**Cell**	K_c	τ_I (min)
$i = 1$	-8.08	0.582	$i = 4$	-8.08	0.444
$i = 2$	-8.08	0.409	$i = 5$	-8.08	0.338
$i = 3$	-7.66	0.538	$i = 6$	-6.20	0.565

6.3.1.2 Forward feeding controller

The settings given in Table 6.1 are used to create a forward feeding (FF) controller to be used for comparison against the BO automatic tuner settings found for a multivariable controller. The FF controller copies the BO SISO PI controllers in (6.1) through the whole matrix such that the controller is a lower triangular matrix of the form

$$\mathbf{C_{FF}} = \begin{bmatrix} c_{11} & 0 & 0 & 0 & 0 & 0 \\ c_{11} & c_{22} & 0 & 0 & 0 & 0 \\ c_{11} & c_{22} & c_{33} & 0 & 0 & 0 \\ c_{11} & c_{22} & c_{33} & c_{44} & 0 & 0 \\ c_{11} & c_{22} & c_{33} & c_{44} & c_{55} & 0 \\ c_{11} & c_{22} & c_{33} & c_{44} & c_{55} & c_{66} \end{bmatrix}. \tag{6.3}$$

Since the cells within an industrial flotation bank are generally uniform, it is reasonable to use the controllers on the diagonal to complete the lower triangle of the matrix. This FF structure is similar to the controller used in Kämpjärvi and Jämsä-Jounela (2003), Jämsä-Jounela et al. (2003) and Schubert et al. (1995). The FF controller results in each valve being controlled by its own level plus the upstream level errors, i.e., the error of each cell is fed forward to the cells downstream. This assists to mitigate upstream disturbances in downstream units. The conditions of closed-loop stability using this controller are addressed in Section 6.3.2.

6.3.1.3 MIMO inventory controller

The inventory controller (cf. (Kämpjärvi and Jämsä-Jounela, 2003), (Jämsä-Jounela et al., 2003) and (Schubert et al., 1995)) in Fig. 6.1 is a MIMO controller that takes into account the total inventory of the flotation bank. Since the entry-ports of the cells are at the bottom, the level of a cell influences both the in- and outflow volume. To ensure local consistency of control loops and following the radiating rule (Aske and Skogestad, 2009; Minasidis et al., 2015), a lower triangular matrix structure is chosen for the MIMO inventory controller with the elements of the matrix individual PI controllers as per (6.2). This choice is the industry standard approach (Schubert et al., 1995). The controller structure is

$$\mathbf{C_{MIMO}} = \begin{bmatrix} c_{11} & 0 & \cdots & 0 \\ c_{21} & c_{22} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ c_{61} & c_{62} & \cdots & c_{66} \end{bmatrix}. \tag{6.4}$$

As shown by the subscripts, the difference between (6.3) and (6.4) is that in (6.3) all elements of a

column are equal, whereas in (6.4) the off-diagonal elements are allowed to differ from the diagonal elements. For (6.3), once the controller parameters for the diagonal elements are defined, the full matrix is defined. For (6.4), the controller parameters for each element in the matrix must be obtained.

The diagonal elements in (6.4) control the response of each valve to its own level, whereas the off-diagonal elements influence the interactions between cells. When the diagonal elements are tuned at the same time as the off-diagonal elements, large fluctuations in the level responses occur. The reason for this is that the BO automatic tuner searches over the whole stable parameter space given in Table 6.2, found using μ-analysis as described in Section 6.3.2, which include controller parameter combinations that result in a combination of fast and slow closed-loop responses.

The combination of a slow controller on the diagonal and a much faster off-diagonal controller means that down-stream cells might e.g. overcompensate for the level fluctuations in cell 1. For example, if the first column in the controller matrix is being tuned, in other words the controllers responsible for providing the control signal to all valves affected by changes in level 1, the control action of valve 1 will influence the flow to all down-stream cells. The off-diagonal elements are supposed to correct for the change in flow induced by valve 1. If both the diagonal and off-diagonal elements change, the plant may encounter large level fluctuation as the controllers may produce widely varying responses in the valve positions. This will slow down the learning part of the BO automatic tuner.

For these reasons a 2-step approach is chosen to first fix, i.e., determine and set, the diagonal controllers, and secondly tune the off-diagonal controllers. By fixing the diagonal controller settings first, the fluctuations are limited, and the tuning procedure also becomes more efficient and less perturbing to normal operation. This can be done by obtaining the controller settings for the diagonal elements using either a classical approach such as SIMC, or a machine learning approach such as BO.

The off-diagonal controller entries in (6.4) are chosen to be of the form shown in (6.2) (similar to van Niekerk et al. (2023)), with the initial K_{Cij} and τ_{Iij} values of each column taken from the diagonal controller element c_{ii}, i.e. initially $c_{21} = c_{31} = c_{41} = c_{51} = c_{61} = c_{11}$. This initial structure is equivalent to the FF controller in (6.3). Initializing the search in this way helps the BO automatic tuner find the optimal settings in fewer iterations, but does not influence the final settings that are found by the BO.

The outline for the tuning procedure of the MIMO inventory controller in (6.4), is given as follows:

1. determine the diagonal controller settings and set them constant,
2. find the controller bounds making use of the robust stability analysis and using the FF controller structure as shown in (6.3), and
3. tune the off-diagonal controller settings using BO.

6.3.2 Robust stability analysis & controller bounds for MIMO inventory control

The MIMO controller in (6.4) is subjected to a robust stability analysis (see e.g. Skogestad and Postlethwaite (2007)) from which stability bounds on the controller gains (K_{Cij}) and integral time constants (τ_{ij}) are obtained (The *robstab* function in MATLAB is used to obtain the controller bounds.). For the robust stability analysis the plant is represented by the linear plant model in (4.7). The controller has a lower diagonal structure with each nonzero element as given in (6.2). The robust stability analysis is performed similar to van Niekerk et al. (2023), with the resulting controller parameter ranges given in Table 6.2. The result of the robust stability analysis can be visualized through a structured singular value (μ) plot of the closed-loop system shown in Fig. 6.2. The system is robustly stable ($\mu < 1$) over the whole frequency range, meaning that the MIMO controller in (6.4) will stabilize the plant in (4.7) for any controller parameter in the ranges shown in Table 6.2. Constraining the search space of the BO in this way will ensure closed-loop stability as long as the plant is adequately represented by (4.7). Note, the non-linear plant model is controlled in the simulations shown in Section 6.4 and that the linear plant model is only used for the robust stability analysis.

Table 6.2. MIMO bounds for BO.

Cell(ij)	K_{Cij}		τ_{Iij} (min)	
	Lower	Upper	Lower	Upper
$i1$	-8.79	-1.72	2.022	0.104
$i2$	-8.78	-1.72	1.43	0.0738
$i3$	-8.33	-1.63	1.87	0.0978
$i4$	-8.78	-1.72	1.73	0.0795
$i5$	-8.78	-1.72	1.26	0.0607

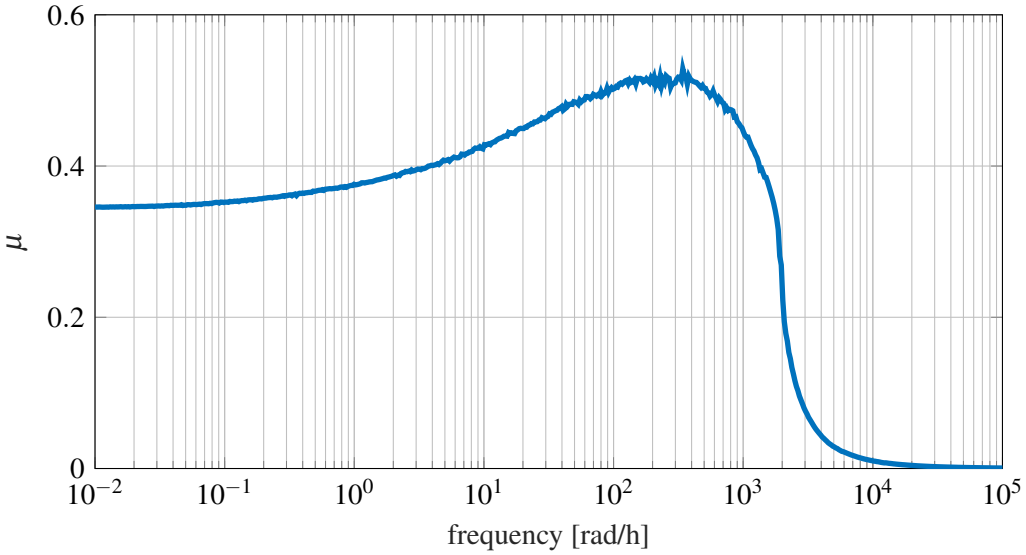


Figure 6.2. Maximum singular values (μ) for constraints given in Table 6.2.

6.4 BO CONTROLLER TUNING

The plant and controller is simulated in MATLAB and Simulink. The diagonal controllers are fixed using the BO settings in Table 6.1 (see Section 5.6). The off-diagonal elements are tuned one cell at a time, using step changes in the level reference values and evaluating the error between the reference and process variable. Only a single reference is stepped at a time to isolate the interactions to a single source to tune the controller elements that control on the difference between the actual level and the level reference being stepped. The reference errors for each cell can again be evaluated according to the ISE, IAE, or ITAE (see Section 5.5.2 and Seborg et al. (2019)).

The errors are collected in an objective function Q, shown below, expressed as a weighted sum of the ISE and ITAE for each cell i. The ISE metric was selected to penalize large errors, whereas the ITAE metric was selected to penalize errors that persist for a long time. The weights of the performance metrics were obtained in a trade-off between these two metrics with final cost function as:

$$Q = \sum_{i=1}^{6} 0.2\text{ISE}(i) + 0.8\text{ITAE}(i). \tag{6.5}$$

The BO tuner is given 80 iterations to minimize the objective function in terms of the controller parameters. The standard *bayesopt* function in MATLAB is used with the "Expected-improvement" (EI) chosen as the acquisition function and with an exploration ratio of 0.8. The exploration ratio is a parameter used to control the algorithm's propensity to explore the parameter space if little improvement is being made in subsequent iterations. The rest of the *bayesopt* parameters are kept to

their default values.

The tuning starts at cell 1 and ends at cell 5 as only the off-diagonal elements are tuned. The tuning is done using two different strategies:

- **Strategy 1**: The off-diagonal elements are added as each next cell is tuned.
- **Strategy 2**: The tuning starts with the full controller of (6.4).

For Strategy 2 the assumption is that by tuning with all controller elements in place the cost function will be more representative of the final response and the need for doing multiple passes through the columns will be mitigated. For example if the first column is tuned without the full down-stream controllers in place the ability of the down-stream cells to handle the inventory being sent down will be very limited and in turn limit the upstream controllers to more conservative settings. To implement Strategy 2, it is assumed all cells have similar dynamics and the off-diagonal controllers can be initialized safely with the diagonal controller parameters. This is not necessarily the case for an industrial plant where cells dynamics may differ. In comparison, for Strategy 1 the controller elements are added column for column as the BO automatic tuner progresses through the multivariable controller. Thus, Strategy 1 is a cautious approach where controllers are added as the automatic tuner progresses down the flotation bank. The computational complexity and optimization time remain the same for both strategies as the number of parameters optimized at a time are the same. Only the cost function changes between the two strategies.

The FF controller serves as the base case for comparison and the BO automatic tuner will aim to find a controller similar to the FF controller. The performance of the BO tuned MIMO controllers are compared to the FF controller using the following performance metrics: IAE, ISE, ITAE and TV.

6.4.1 Tuning without measurement noise

The initial controller structure is similar to (6.4)

$$\mathbf{C_{BO}} = \begin{bmatrix} c_{11} & 0 & 0 & 0 & 0 & 0 \\ c_{21} & c_{22} & 0 & 0 & 0 & 0 \\ c_{31} & c_{32} & c_{33} & 0 & 0 & 0 \\ c_{41} & c_{42} & c_{43} & c_{44} & 0 & 0 \\ c_{51} & c_{52} & c_{53} & c_{54} & c_{55} & 0 \\ c_{61} & c_{62} & c_{63} & c_{64} & c_{65} & c_{66} \end{bmatrix}, \tag{6.6}$$

with each element an individual PI controller as given in (6.2). The diagonal elements in blue are fixed and not tuned. The controller settings found by the BO tuner for Strategies 1 and 2 are shown below in terms of the PI controller gains and integral time constants.

- **Strategy 1:**

$$\mathbf{K_{ij}^{(1)}} = \begin{Bmatrix} -8.08 & 0 & 0 & 0 & 0 & 0 \\ -8.45 & -8.08 & 0 & 0 & 0 & 0 \\ -8.23 & -7.36 & -7.66 & 0 & 0 & 0 \\ -8.61 & -7.52 & -6.53 & -8.08 & 0 & 0 \\ -8.85 & -7.73 & -7.05 & -7.43 & -8.08 & 0 \\ -8.29 & -7.86 & -6.85 & -7.47 & -7.83 & -6.20 \end{Bmatrix} \tag{6.7a}$$

$$\boldsymbol{\tau}_{Iij}^{(1)} = \begin{Bmatrix} 0.579 & 0 & 0 & 0 & 0 & 0 \\ 1.45 & 0.409 & 0 & 0 & 0 & 0 \\ 1.911 & 0.678 & 0.538 & 0 & 0 & 0 \\ 1.293 & 0.653 & 0.836 & 0.444 & 0 & 0 \\ 1.035 & 0.707 & 0.778 & 0.697 & 0.338 & 0 \\ 1.941 & 0.665 & 0.837 & 0.720 & 0.596 & 0.565 \end{Bmatrix}. \tag{6.7b}$$

- **Strategy 2:**

$$\mathbf{K_{ij}^{(2)}} = \begin{Bmatrix} -8.08 & 0 & 0 & 0 & 0 & 0 \\ -8.08 & -8.08 & 0 & 0 & 0 & 0 \\ -8.08 & -7.84 & -7.66 & 0 & 0 & 0 \\ -8.08 & -7.61 & -7.42 & -8.08 & 0 & 0 \\ -8.08 & -8.40 & -7.77 & -7.10 & -8.08 & 0 \\ -8.08 & -8.06 & -7.75 & -7.34 & -7.81 & -6.20 \end{Bmatrix} \tag{6.8a}$$

$$\boldsymbol{\tau}_{Iij}^{(2)} = \begin{Bmatrix} 0.579 & 0 & 0 & 0 & 0 & 0 \\ 0.579 & 0.409 & 0 & 0 & 0 & 0 \\ 0.579 & 0.704 & 0.538 & 0 & 0 & 0 \\ 0.579 & 0.706 & 0.731 & 0.444 & 0 & 0 \\ 0.579 & 0.625 & 0.738 & 0.754 & 0.338 & 0 \\ 0.579 & 0.608 & 0.730 & 0.755 & 0.594 & 0.565 \end{Bmatrix}. \tag{6.8b}$$

The plant and MIMO controllers are subjected to two different performance tests:

- **Test 1**: a sequence of reference steps of 3 cm to the level of each cell.
- **Test 2**: a -20% disturbance to the feed flow, Q_F at time $t = 10$ s, as well as 75 m^3/h of spillage water pumped into cell 3 from time $t = 100$ s.

The performance of the BO controllers are compared to the FF controller. The level responses to the step changes in the reference signals are shown in Figs. 6.3 and 6.5. Both the BO tuned MIMO controllers show an improvement over the FF controller in the reference tracking as seen in Table 6.3. The BO controllers eliminate the error quicker than the FF controller.

The level responses to the disturbances are shown in Figs. 6.7 and 6.9. The BO controller tuned according to Strategy 2 shows an improvement over the FF controller in terms of disturbance rejection as can be seen in Table 6.4. As seen from the performance metrics in Table 6.4, Strategy 1 performs much worse than both the FF controller and the settings found by Strategy 2. This makes sense as Strategy 1 tunes the controller without all the elements in place, and might be optimal at the time it is being tuned but at the end is not optimal anymore. Once again the smallest improvement is seen in the ISE, similar to what is seen for reference tracking in Table 6.3. The actuator usage for the different controllers are similar. This can also be seen from the similar performances. However, the Strategy 1 controller shows the smallest TV values. This decrease in actuator usage results in a degraded performance when it comes to following the level setpoint.

The results show that the BO automatic tuner is able to tune multiple PI elements to a satisfactory level. The most parameters tuned at once is for cell 1 where 5 PI controllers are tuned (c_{11} is fixed) consisting of 2 parameters each, thus 10 parameters in total. The objective function in (6.5) is shown to be effective as the minimization of the objective function leads to an improvement over the FF controller for the entire bank in the case of Strategy 2.

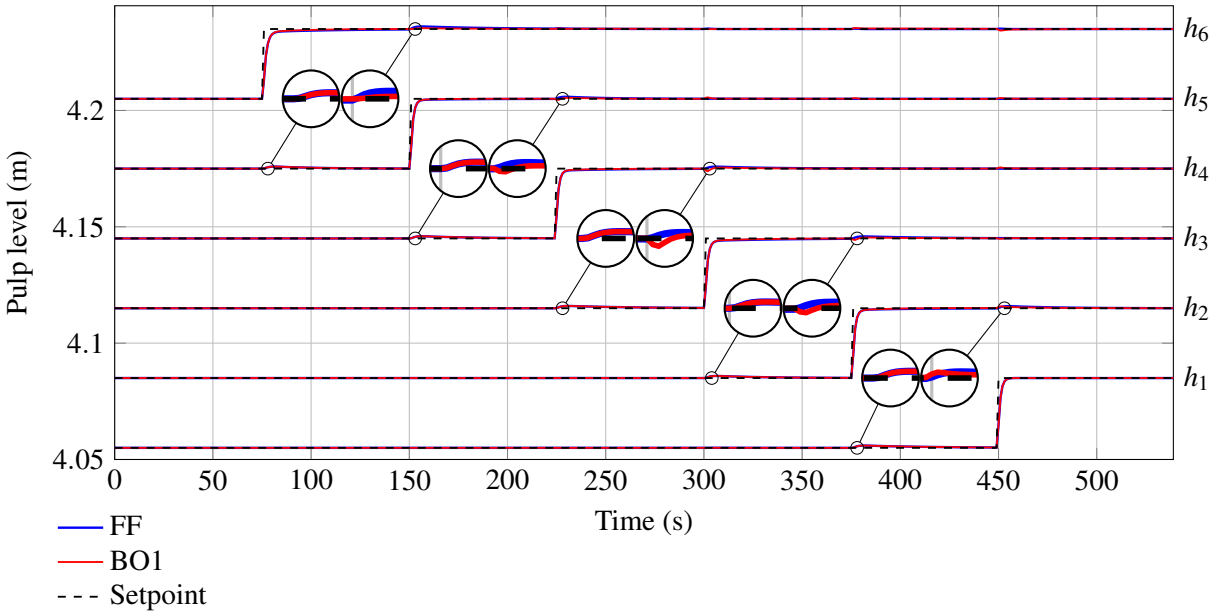
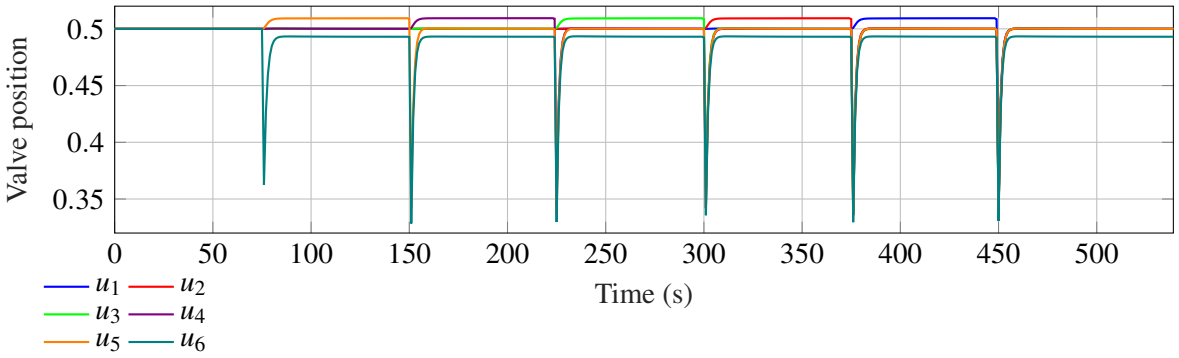
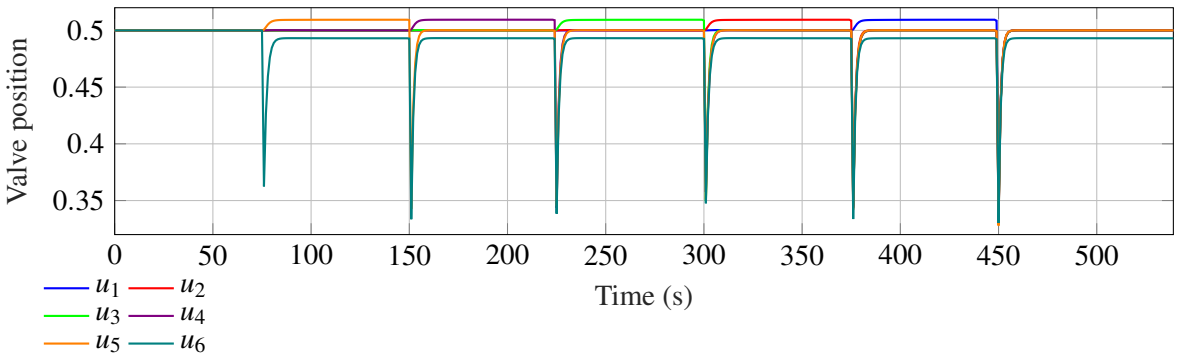


Figure 6.3. Setpoint changes for Strategy 1.



(a) FF manipulated variable action.



(b) BO manipulated variable action.

Figure 6.4. Manipulated variable action for setpoint changes with Strategy 1.

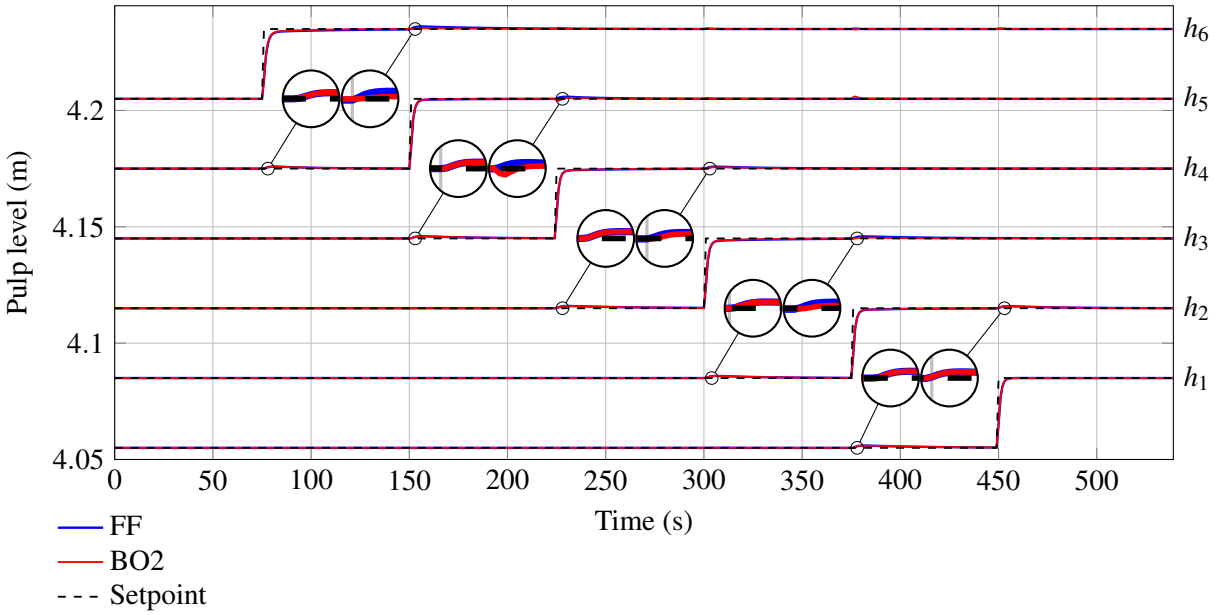
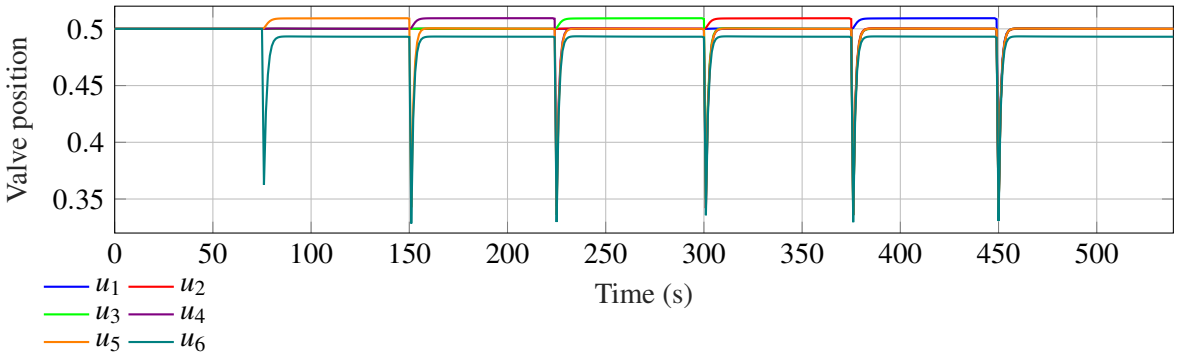
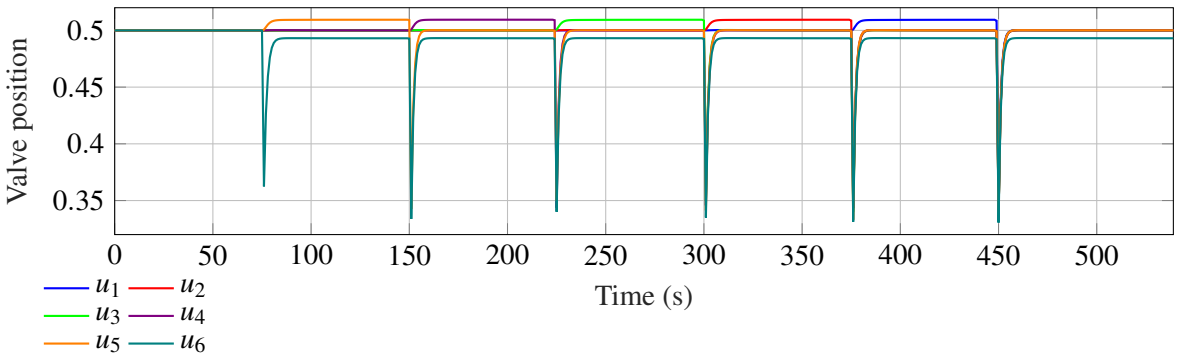


Figure 6.5. Setpoint changes for Strategy 2.



(a) FF manipulated variable action.



(b) BO manipulated variable action.

Figure 6.6. Manipulated variable action for setpoint changes with Strategy 2.

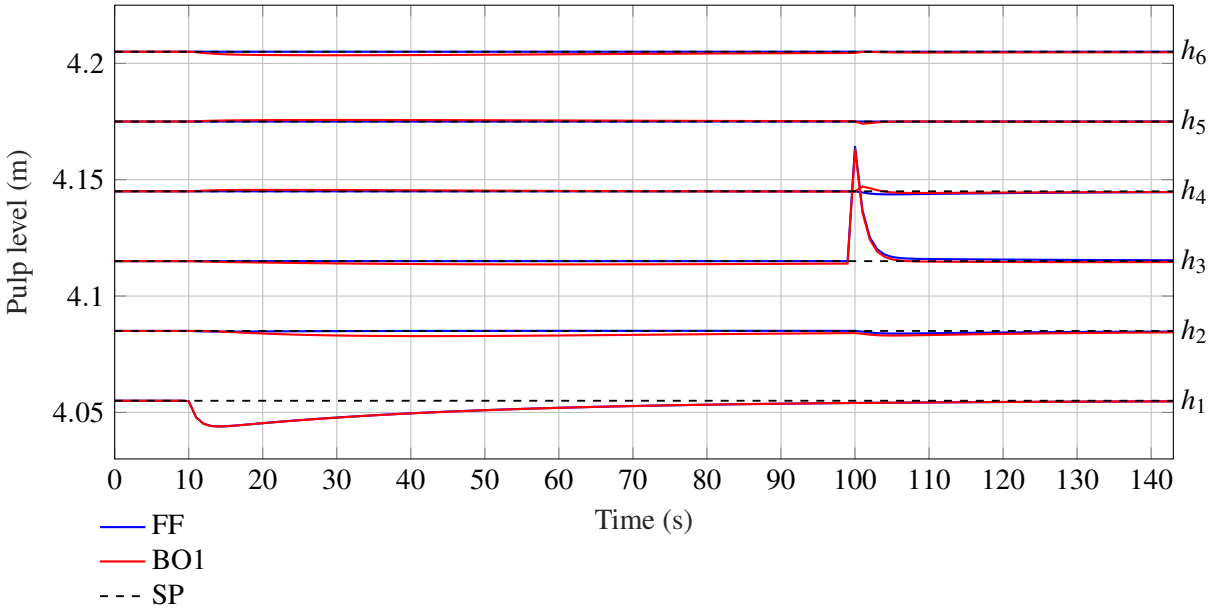
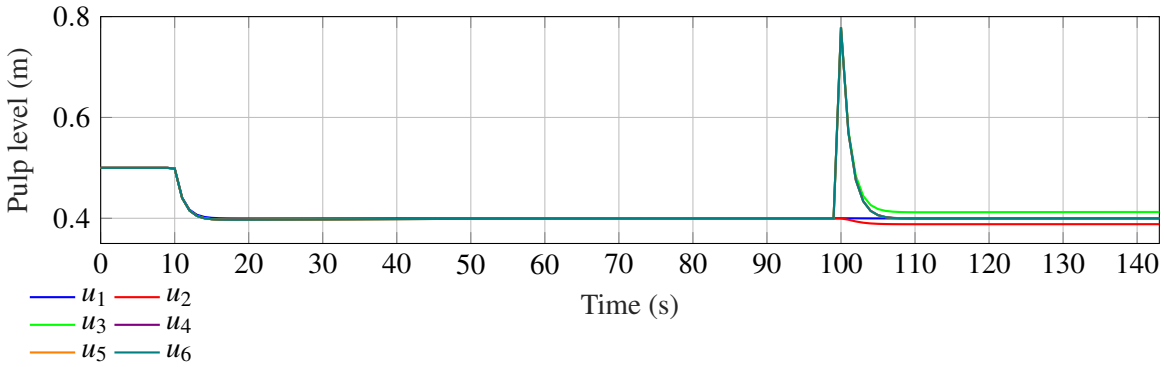
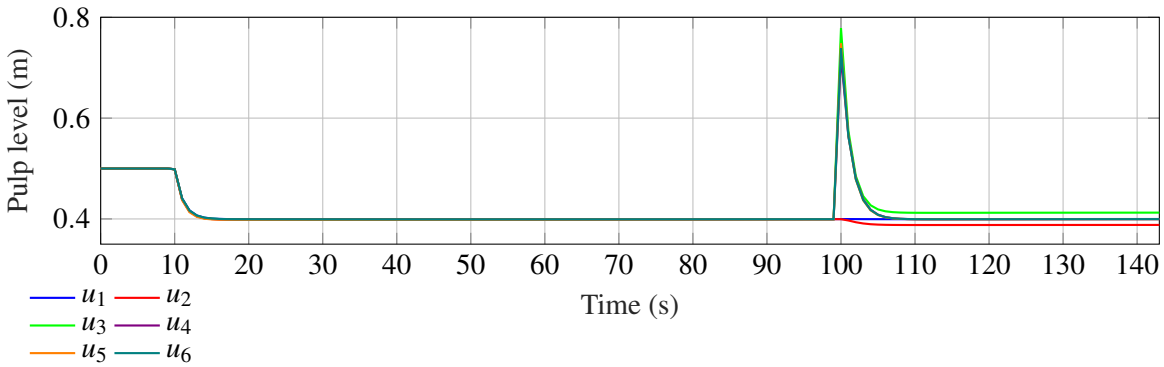


Figure 6.7. Disturbance rejection for Strategy 1.



(a) FF manipulated variable action.



(b) BO manipulated variable action.

Figure 6.8. Manipulated variable action for disturbance rejection with Strategy 1.

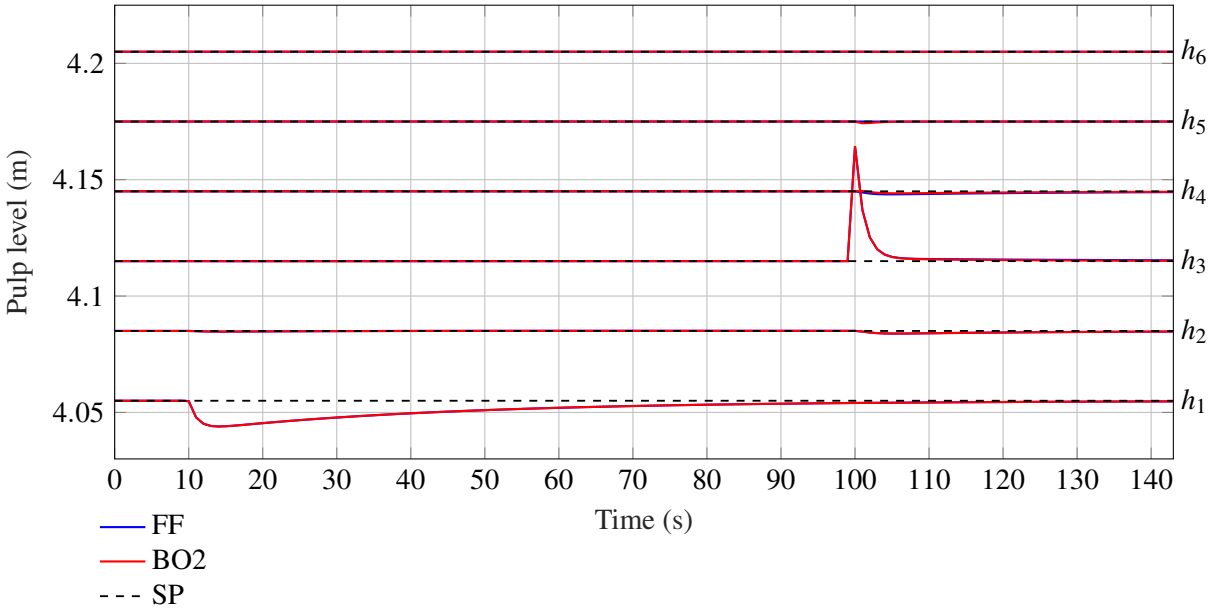
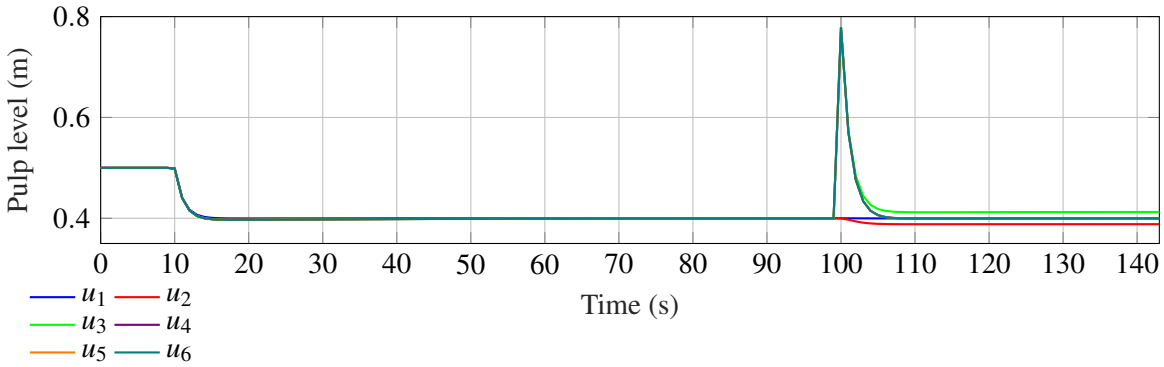
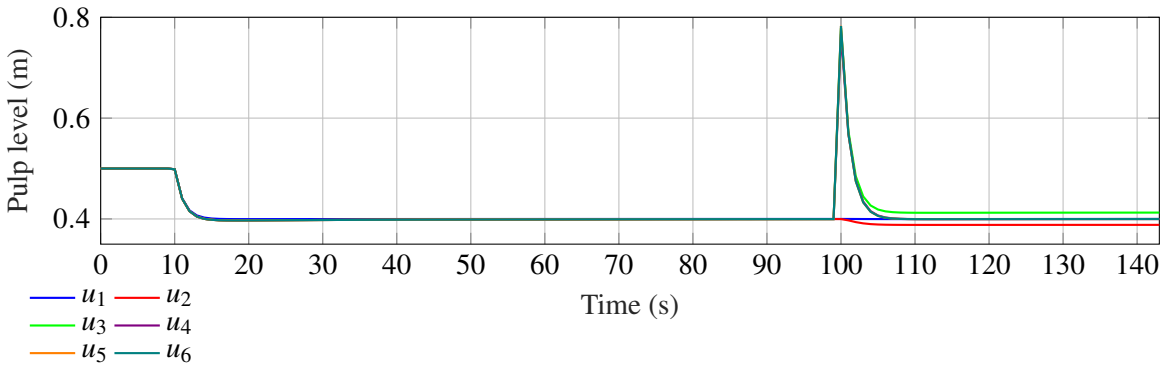


Figure 6.9. Disturbance rejection for Strategy 2.



(a) FF manipulated variable action.



(b) BO manipulated variable action.

Figure 6.10. Manipulated variable action for disturbance rejection with Strategy 2.

Table 6.3. Reference tracking performance.

Metric	Controller			Improvement	
	FF	BO1	BO2	BO1	BO2
IAE	65.4	57.3	57.7	12.39%	11.77%
ISE	34.2	33.4	33.5	2.34%	2.05%
ITAE	17645	16120	16274	8.64%	7.77%
TV	3.65926	3.38904	3.65115	7.38%	0.22%
Q	14123	12907	13031	8.61%	7.73%

Table 6.4. Disturbance rejection performance.

Metric	Controller			Improvement	
	FF	BO1	BO2	BO1	BO2
IAE	60.3	101.4	59.3	-68.2%	1.66%
ISE	57.1	61.0	56.9	-6.83%	0.35%
ITAE	3759	6057	3603	-61.1%	4.15%
TV	6.80198	6.65474	6.71355	2.16%	1.30%
Q	3019	4857	2894	-60.88%	4.14%

The improvements in Tables 6.3 and 6.4 are quite small. Because the cells in the simulation have similar dynamics throughout the flotation bank, it is difficult to improve upon the FF controller. However, if the cell dynamics differ and change, as can be expected on an industrial plant, the FF controller settings are not necessarily the best to ensure good performance, especially in terms of interactions between cells. However, the BO automatic tuner will be able to compensate for these differences and find the best controller settings according to the cost function.

6.4.2 Tuning with measurement noise

The process above was repeated using Strategy 2 with noise added to the cell level measurements. The magnitude of the reference steps were kept at 3.0 cm. Band limited white noise was selected to simulate level sensor noise of up to ± 10 mm in magnitude. This equates to 11% of the operating range of the levels, which is in line with the average resolution of common radar levels sensors such as the VegaPuls 31 (Vega, 2024). Therefore, the noise is similar to what is expected on an industrial plant. The automatic tuning procedure was conducted with the noise present providing new controller

parameters. The final setpoint following and disturbance rejection performance is shown in Figs. 6.11 and 6.13 for the controller parameters in (6.9).

$$\mathbf{K}_{\mathbf{ij}}^{(\mathbf{noise})} = \begin{Bmatrix} -8.08 & 0 & 0 & 0 & 0 & 0 \\ -7.53 & -8.08 & 0 & 0 & 0 & 0 \\ -3.02 & -4.55 & -7.66 & 0 & 0 & 0 \\ -5.16 & -3.94 & -5.00 & -8.08 & 0 & 0 \\ -7.68 & -6.32 & -6.00 & -8.30 & -8.08 & 0 \\ -6.22 & -6.88 & -8.48 & -8.09 & -7.61 & -6.20 \end{Bmatrix} \tag{6.9a}$$

$$\boldsymbol{\tau}_{Iij}^{(noise)} = \begin{Bmatrix} 0.579 & 0 & 0 & 0 & 0 & 0 \\ 0.438 & 0.409 & 0 & 0 & 0 & 0 \\ 0.213 & 0.197 & 0.538 & 0 & 0 & 0 \\ 0.435 & 0.304 & 0.394 & 0.444 & 0 & 0 \\ 0.837 & 0.266 & 1.401 & 0.874 & 0.338 & 0 \\ 0.805 & 0.418 & 0.612 & 0.392 & 0.388 & 0.565 \end{Bmatrix} \tag{6.9b}$$

It can be seen from both the controller gains and time constants that the BO automatic tuner favors detuning the controllers in the presence of noise. This makes sense as the original controller settings are quite aggressive and in the presence of noise it might be beneficial to detune the controllers. The newly tuned controller performance is compared to the FF controller. As seen in Tables 6.5 and 6.6 the controller given by the BO automatic tuner performs almost on par with the FF controller.

The impact of the measurement noise can be seen in the change in the controller parameters and the error metrics from Tables 6.3 and 6.4 to Tables 6.5 and 6.6. This result is obtained without filtering the

Table 6.5. Reference tracking performance with noise.

Metric	**Controller**		
	FF	BO	Change
IAE	1120	1155	-3.13%
ISE	625	659	-5.44%
ITAE	301077	314759	-4.54%
TV	91.9612	92.3630	-0.44%
Q	240992	251939	-4.54%

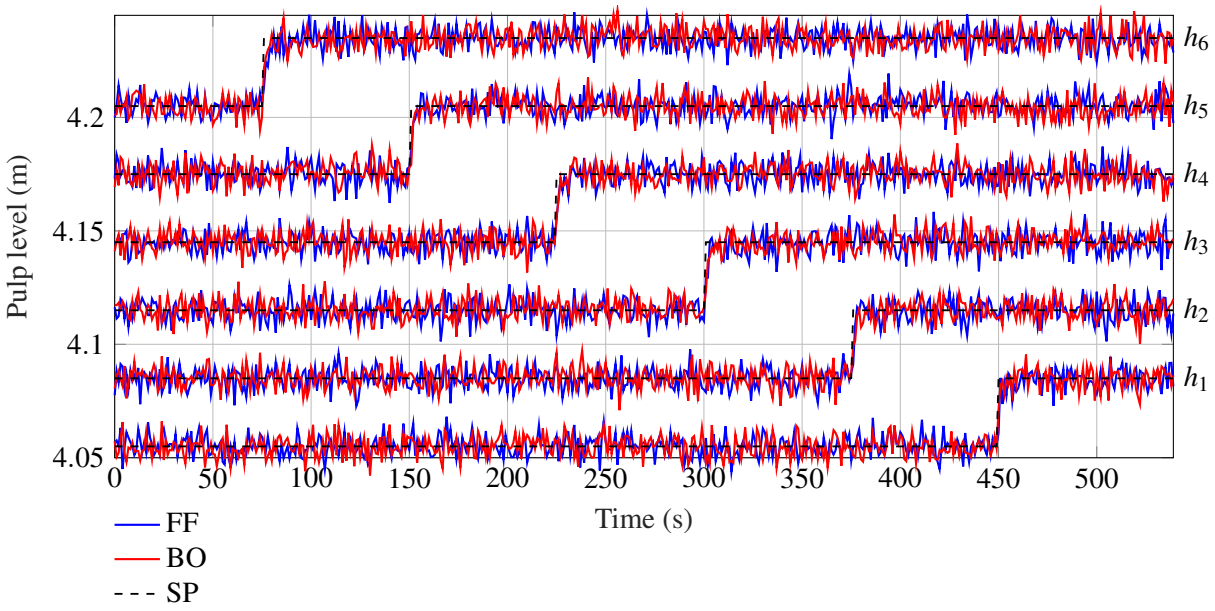
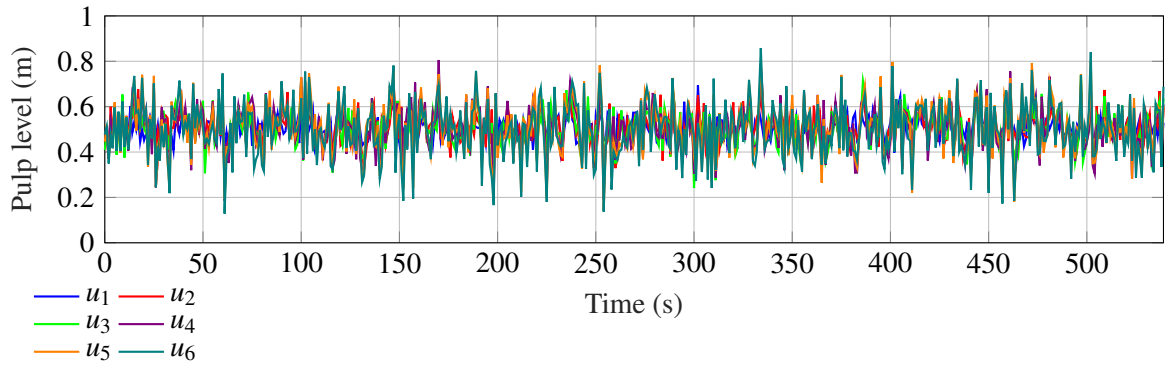
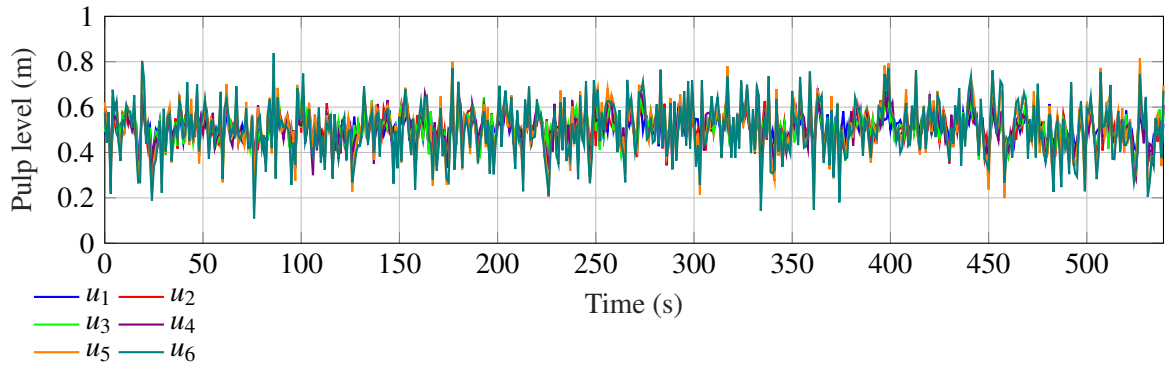


Figure 6.11. Setpoint tracking performance with measurement noise.



(a) FF manipulated variable action.



(b) BO manipulated variable action.

Figure 6.12. Manipulated variable action for setpoint tracking performance with measurement noise.

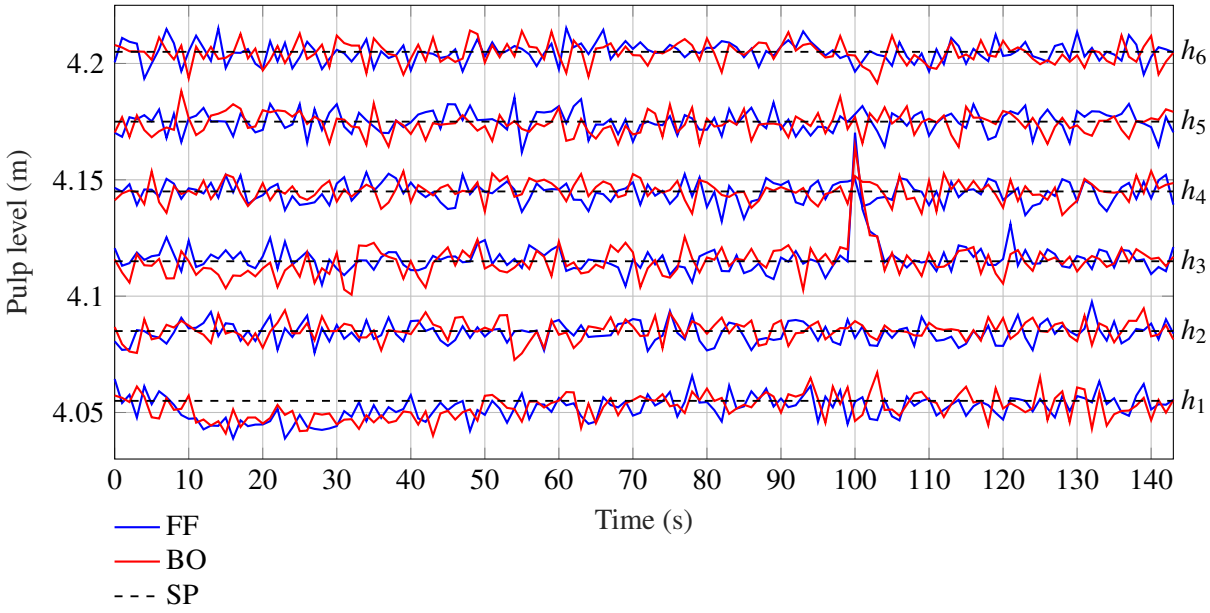
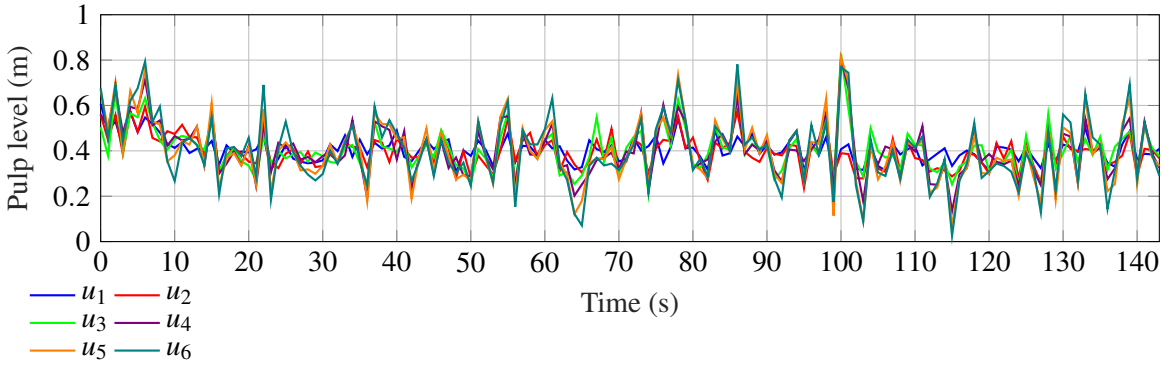
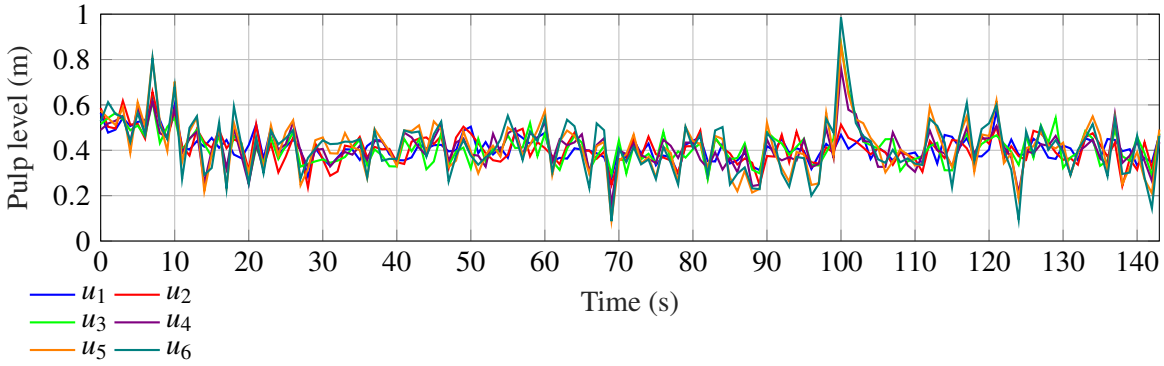


Figure 6.13. Disturbance rejection with measurement noise.



(a) FF manipulated variable action.



(b) BO manipulated variable action.

Figure 6.14. Manipulated variable action for disturbance rejection with measurement noise.

Table 6.6. Disturbance rejection performance with noise.

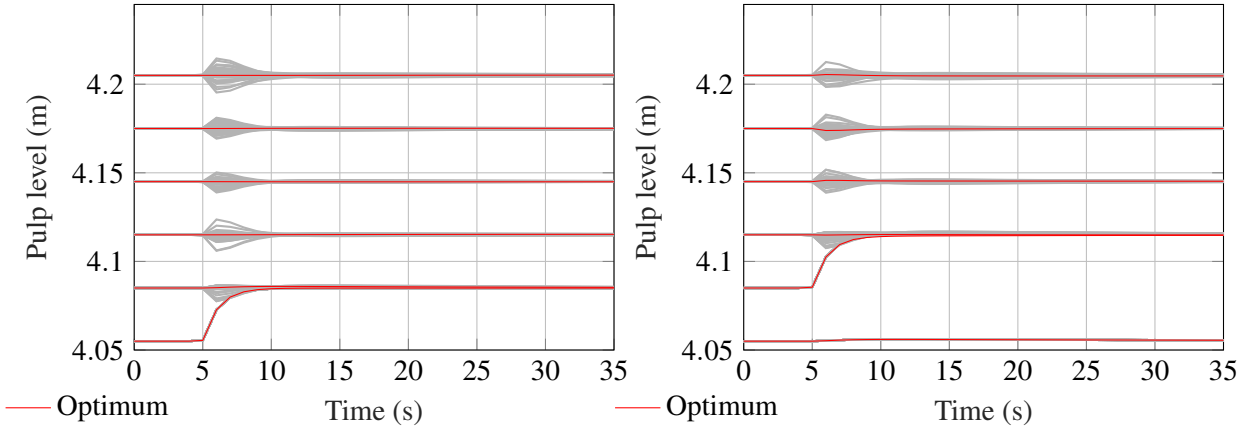
Metric	Controller		
	FF	BO	Change
IAE	313	324	-3.51%
ISE	197	225	-14.0%
ITAE	22506	22776	-1.20%
TV	316.0	366.2	-15.88%
Q	18044	18265	-1.22%

levels and shows the BO automatic tuner is capable of handling noisy environments while keeping the process stable. It does not provide an improvement as in the case with no noise, but does come close to equalling the performance of the FF controller. The impact of the noise can also be seen in the control action, with a much higher TV in the valve position for both controllers compared to the case where there is no measurement noise. The FF controller does slightly better with regards to the TV metric.

The use of filters on the level measurements might help in improving the controller settings found by the BO automatic tuner. However, in this work the goal is to test if the BO automatic tuner is able to function in a noisy environment, as filtering the levels might not get rid of all the noise and might come at the cost of information loss. Thus, the levels were not filtered in this work.

6.4.3 Iteration convergence and efficiency

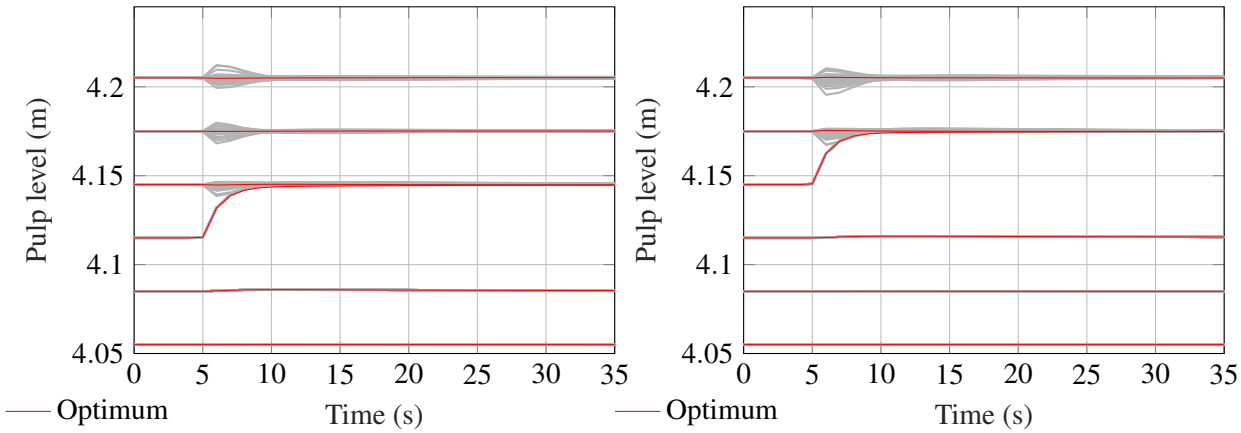
The matters of iteration convergence and efficiency still remain. Convergence in this case implies that after a step change, the levels either return to their setpoint or settle to a new steady-state. As expected from the robust stability analysis in Section 6.3.2, the levels do not diverge to an unstable region after a step change. To show the convergence of the tuning process, the level responses of all 80 iterations for Strategy 2 for the all the columns of the noise free case are shown in Figs. 6.15 to 6.17(a). Here the iterations are plotted in gray, with the optimum, corresponding to the controller settings given in (6.7), plotted in red. As seen from the level responses all iterations converge, and thus the bounds found in Table 6.2 are valid and adhered to. (The BO iterations while tuning with measurement noise is not displayed as the added noise makes it very difficult to distinguish the different iterations from each other.) Fig. 6.18 shows the objective function value of all 80 iterations for all columns. It is evident



(a) Cell 1. **(b)** Cell 2.

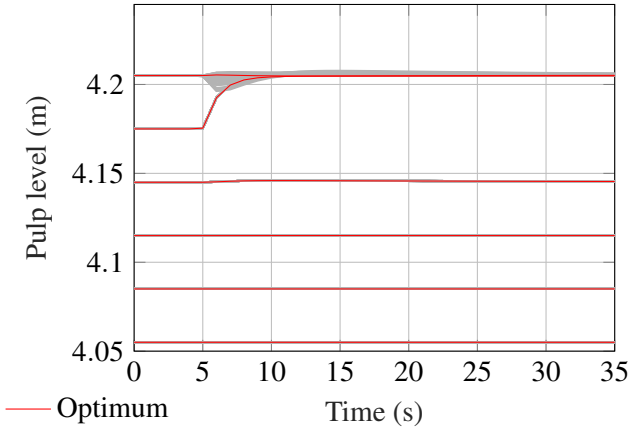
Figure 6.15. MIMO BO iterations cell 1 and 2.

that much of the search space produce similar responses. The first column, where the most parameters (10) are being tuned, has the most variability. The minimum objective function value at each iteration is shown in Fig. 6.19. A value of 1 represents a performance equal to that of the FF controller. All the columns, except for column 1, show an improvement in performance. (Note, column 1, which depends only on c_{11}, does not show an improvement. As discussed in Section 6.4, this is expected since the diagonal elements are first tuned using BO, then fixed, and afterwards the off-diagonal elements are tuned.) Fig. 6.19 shows that the iterations required to find the minimum of the objective function decreases as fewer parameters need to be tuned, i.e. as the automatic tuner moves further down the



(a) Cell 3. **(b)** Cell 4.

Figure 6.16. MIMO BO iterations cell 3 and 4.



(a) Cell 5.

Figure 6.17. Cell 5 BO iterations.

bank. Each of the 80 step tests is given 35 seconds to complete, which makes the total tuning time per column about 47 minutes and the total for the whole bank 3 hours and 54 minutes.

The performance of the controllers obtained by the BO automatic tuner are compared for the cases where 20, 40 and 80 iterations are used respectively. The performance for each case is given in Tables

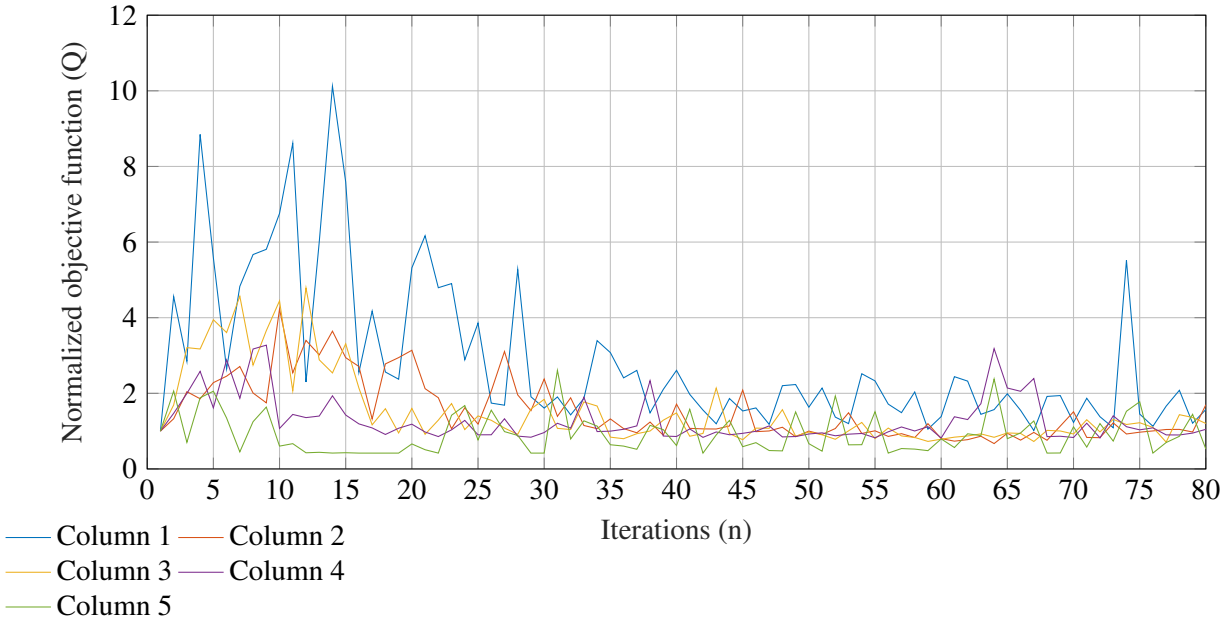


Figure 6.18. Objective function value for each BO iteration normalized to the performance of the FF controller.

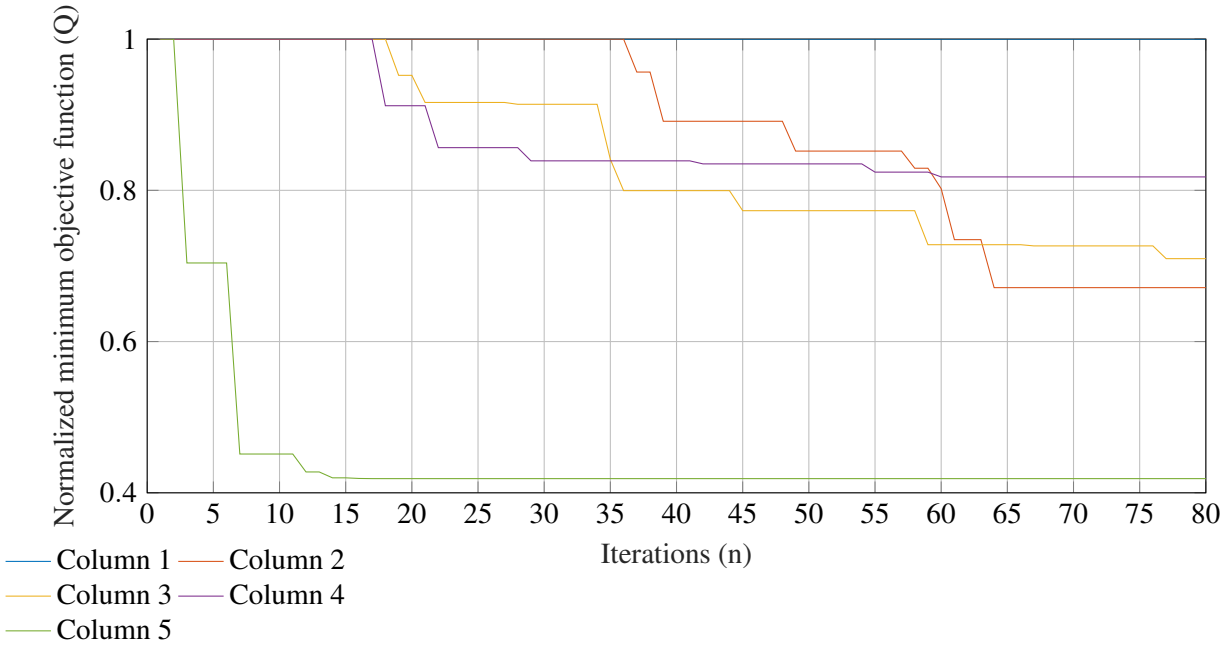


Figure 6.19. Minimum of objective function value traces normalized to the performance of the FF controller.

Table 6.7. Reference tracking performance for BO with different number of iterations.

Metric	**No. of Iterations**		
	20	40	80
IAE	62.4	60.3	57.7
ISE	33.8	33.6	33.5
ITAE	17629	17312	16274
Q	14110	13856	13031
Change in Q from FF	0.10%	1.89%	7.73%

6.7 and 6.8. The BO tuner is able to obtain a larger improvement over the FF controller with 80 iterations than with 20 or 40 iterations. In most the cases the minimum for each column is found well before the 80th iteration. The BO automatic tuner is able to tune multiple PI elements to a satisfactory level to improve the overall performance in the level control of the flotation bank. A performance improvement is seen in the simulated level control for both the reference tracking and disturbance rejection. The BO automatic tuner has to optimize at most 10 parameters. This is done within the 80 iterations it is given, however, the optimum is generally found earlier than the 80th iteration. The BO

Table 6.8. Disturbance rejection performance for BO with different number of iterations.

Metric	No. of Iterations		
	20	40	80
IAE	60.9	60.7	59.3
ISE	57.0	57.0	56.9
ITAE	3828	3775	3603
Q	3074	3031	2894
Change in Q from FF	-1.82%	-0.41%	4.14%

automatic tuner successfully finds parameters that are on par or better than the FF controller, even in the presence of measurement noise. The advantage in an industrial setting is that any flotation bank can be tuned automatically with relative ease.

6.5 CHAPTER SUMMARY

The BO automatic tuner is extended to a MIMO controller. The MIMO controller structure is defined and analyzed for robust stability using a linearized version of the plant model. The robust stability analysis is used to find tuning bounds for the MIMO automatic tuner. The tuner tunes column for column, thus tuning, at most, 10 parameters at a time. The BO automatic tuner is implemented using two different strategies. The best performing strategy is implemented in simulation on a system with measurement noise. The BO automatic tuner is able to find a performance improvement over the FF controller in the case where there is no measurement noise. In the case where measurement noise is added the BO automatic does slightly worse than the FF controller, however comes very close to matching the performance. All iterations conducted by the BO automatic tuner resulted in stable closed-loop responses. As the number of iterations conducted the performance obtained by the BO settings gradually increased.

CHAPTER 7 CONCLUSION

PID controller tuning is a widely researched field within control systems. Just as wide as the research field are the opinions on what the best controlling settings are produced by the tuning rules developed by numerous researchers. All these different settings have their advantages, drawbacks and limitations. The BO automatic tuner is no different. One has to select the appropriate method for each case.

The advantages of the BO automatic tuner is the flexibility in response the method is able to produce by creating a cost function that produces the response that is required. BO is also sample efficient and model free which gives it an advantage in the learning domain. BO is globally optimal and can be implemented on a very wide range of processes. It can be implemented on an already existing controller infrastructure and thus requires no changes to the controller and system. The drawbacks are that although it is an efficient learning method, one has to ask if the increase in tests conducted justify the performance improvement. The limitations creep in when one is dealing with a highly dimensional controller where a large number of parameters are tuned. The BO automatic tuner is required to tune at max 10 parameters in this work, and is able to do so. The number of iterations required to find the optimal settings increases as the number of tuning parameters increase.

The BO automatic tuner is used to tune PI controllers as if tuning for initial controller settings, but to do so safely one requires some other initial tuning as well as a rough model of the system to ensure stability. The system uncertainty can also be included in the robust stability analysis if there is large uncertainties in the model used for the analysis. The BO automatic tuner is best used to optimize controller settings. Tuning rules such as SIMC rules provide good performance with much less perturbation. However, for mineral processing plants the BO automatic tuner might be worth it where more than a 1% performance improvement is substantial. Even more so if the perturbations can be done without disturbing the normal operation too much.

The BO automatic tuner is able to tune both SISO and MIMO controllers that are implemented on flotation level controllers. The settings found by the BO automatic tuner obtains a performance improvement over SIMC settings and FF controller. The different cost functions perform well in obtaining the required response. The SISO cost function is easily set up by using the settling time. The MIMO cost function is a bit more intricate, where one has to find the balance between the different error metrics to create a good response trade-off between reference tracking and disturbance rejection. The best strategy when tuning controllers with the BO automatic tuner is to have the full controller in place when tuning. This also creates the need for initial tuning values as discussed above.

Based on the above discussion future work can include:

- Implementing perturbation methods that have less of an impact on the normal operation of the plant. For example small signal injection can be used to perturb the plant and generate performance data.
- Comparing BO with other iterative optimization methods on the same process.
- Using BO to tune highly dimensional controllers without the need to break the tuning up into smaller parts, i.e. tuning all the elements of a MIMO controller at once. This will also require a different perturbation method to generate performance data for all the elements of the MIMO controller.

The BO automatic tuner shows promise in optimizing controller settings where it is hard to find improvements. This comes at the cost of an increased number of tests conducted. However, BO might be best suited as a learning agent that continually learns and makes tuning suggestions as the processing is running.

REFERENCES

Ahn, K. and Truong, D. (2009). Online tuning fuzzy PID controller using robust extended Kalman filter, *Journal of Process Control* **19**(6): pp. 1011–1023.

Aske, E. M. B. and Skogestad, S. (2009). Consistent inventory control, *Industrial & Engineering Chemistry Research* **48**(24): pp. 10892–10902.

Åström, K. and Hägglund, T. (1984). Automatic tuning of simple regulators with specifications on phase and amplitude margins, *Automatica* **20**(5): pp. 645–651.

Åström, K., Hägglund, T., Hang, C. and Ho, W. (1993). Automatic tuning and adaptation for PID controllers - a survey, *Control Engineering Practice* **1**(4): pp. 699–714.

Behera, S., Jyotiranjan, M. and Pati, B. (2017). Optimal pole placement for a self tuning PID controller, *Proceedings of the 6th International Conference on Computer Applications In Electrical Engineering-Recent Advances (CERA), 5-7 October 2017, Roorkee, India*, pp. 456–461.

Berkenkamp, F., Krause, A. and Schoellig, A. P. (2021). Bayesian optimization with safety constraints: safe and automatic parameter tuning in robotics, *Machine Learning* **112**(10): pp. 3713–3747.

Besharati Rad, A., Lo, W. L. and Tsang, K. (1997). Self-tuning PID controller using Newton-Raphson search method, *IEEE Transactions on Industrial Electronics* **44**(5): pp. 717–725.

Boubertakh, H., Tadjine, M., Glorennec, P.-Y. and Labiod, S. (2010). Tuning fuzzy PD and PI controllers using reinforcement learning, *ISA Transactions* **49**(4): pp. 543–551.

Boyd, S., Hast, M. and Åström, K. J. (2015). MIMO PID tuning via iterated LMI restriction, *International Journal of Robust and Nonlinear Control* **26**(8): pp. 1718–1731.

Campestrini, L., Barros, P. R. and Bazanella, A. S. (2006). Auto-tuning of PID controllers for MIMO processes by relay feedback, *IFAC Proceedings Volumes* **39**(2): pp. 451–456.

Cohen, G. H. and Coon, G. A. (1953). Theoretical consideration of retarded control, *Transactions ASME* **75**: pp. 827–834.

Coutinho, J. P., Santos, L. O. and Reis, M. S. (2023). Bayesian optimization for automatic tuning of digital multi-loop PID controllers, *Computers & Chemical Engineering* **173**: pp. 108211.

Euzebio, T. A., Silva, M. T. and Yamashita, A. S. (2021). Decentralized PID controller tuning based on nonlinear optimization to minimize the disturbance effects in coupled loops, *IEEE Access* **9**: pp. 156857–156867.

Fiducioso, M., Curi, S., Schumacher, B., Gwerder, M. and Krause, A. (2019). Safe contextual Bayesian optimization for sustainable room temperature PID control tuning, *Proceedings of the 28th International Joint Conference on Artificial Intelligence, 10-16 August 2019, Macao, China*, pp. 5850–5856.

Frazier, P. I. (2018). A tutorial on Bayesian optimization, *arXiv preprint arXiv:1807.02811* .

Gawthrop, P. and Nomikos, P. (1990). Automatic tuning of commercial PID controllers for single-loop and multiloop applications, *IEEE Control Systems Magazine* **10**(1): pp. 34–42.

Gous, G. Z., Wiid, A. J., le Roux, J. D. and Craig, I. K. (2023). Advanced regulatory control techniques for improved averaging level control performance, *Industrial & Engineering Chemistry Research* **62**(38): pp. 15578–15587.

Guardeño, R., López, M. J. and Sánchez, V. M. (2019). MIMO PID controller tuning method for quadrotor based on LQR/LQG theory, *Robotics* **8**(2): pp. 36.

Gyöngy, I. and Clarke, D. (2006). On the automatic tuning and adaptation of PID controllers, *Control Engineering Practice* **14**(2): pp. 149–163.

Hang, C. and Sin, K. (1991). On-line auto tuning of PID controllers based on the cross-correlation technique, *IEEE Transactions on Industrial Electronics* **38**(6): pp. 428–437.

Ho, W., Hong, Y., Hansson, A., Hjalmarsson, H. and Deng, J. (2003). Relay auto-tuning of PID controllers using iterative feedback tuning, *Automatica* **39**(1): pp. 149–157.

Howell, M. and Best, M. (2000). On-line PID tuning for engine idle-speed control using continuous action reinforcement learning automata, *Control Engineering Practice* **8**(2): pp. 147–154.

Jones, A. and De Moura Oliveira, P. (1995). Genetic auto-tuning of PID controllers, *Proceedings on the 1st International Conference on Genetic Algorithms in Engineering Systems: Innovations and Applications (GALESIA), 12-14 September 1995, Sheffield, England*, pp. 141–145.

Jones, D. R. (2001). A taxonomy of global optimization methods based on response surfaces, *Journal of Global Optimization* **21**(4): pp. 345–383.

Jones, D. R., Schonlau, M. and Welch, W. J. (1998). Efficient global optimization of expensive black-box functions, *Journal of Global Optimization* **13**(4): pp. 455–492.

Jovanović, I. and Miljanović, I. (2015). Contemporary advanced control techniques for flotation plants with mechanical flotation cells – a review, *Minerals Engineering* **70**: pp. 228–249.

Jämsä-Jounela, S.-L., Dietrich, M., Halmevaara, K. and Tiili, O. (2003). Control of pulp levels in flotation cells, *Control Engineering Practice* **11**(1): pp. 73–81.

Kämpjärvi, P. and Jämsä-Jounela, S.-L. (2003). Level control strategies for flotation cells, *Minerals Engineering* **16**(11): pp. 1061–1068.

le Roux, J. D. and Craig, I. K. (2019). Plant-wide control framework for a grinding mill circuit, *Industrial & Engineering Chemistry Research* **58**(26): 11585–11600.

MathWorks (2022). Matlab version: 9.13.0 (r2022b). https://www.mathworks.com.

Minasidis, V., Skogestad, S. and Kaistha, N. (2015). Simple rules for economic plantwide control, *Computer Aided Chemical Engineering* **37**: pp. 101–108.

Močkus, J. (1975). On Bayesian methods for seeking the extremum, *Proceedings of the Optimization Techniques IFIP Technical Conference, 1-7 July 1974, Novosibirsk, Russia*, pp. 400–404.

Neumann-Brosig, M., Marco, A., Schwarzmann, D. and Trimpe, S. (2020). Data-efficient autotuning with Bayesian optimization: An industrial control study, *IEEE Transactions on Control Systems Technology* **28**(3): pp. 730–740.

Ono, H., Sonoda, T. and Maekawa, A. (2000). Automatic tuning of PID controllers for MIMO processes, *IFAC Proceedings Volumes* **33**(4): pp. 79–84.

Pandey, S. K., Veeranna, K., Kumar, B. and Deshmukh, K. U. (2020). A robust auto-tuning scheme for PID controllers, *Proceedings of the 46th Annual Conference of the IEEE Industrial Electronics Society, 18-21 October 2020, Singapore, Singapore*, pp. 47–52.

Pavković, D., Polak, S. and Zorc, D. (2014). PID controller auto-tuning based on process step response and damping optimum criterion, *ISA Transactions* **53**(1): pp. 85–96.

Rasmussen, C. E. and Williams, C. K. I. (2006). *Gaussian processes for machine learning*, MIT Press, London, England.

Richter, A. V., Le Roux, J. D. and Craig, I. K. (2024a). Automatic tuning of level controllers in a flotation bank using Bayesian optimization, *IFAC-PapersOnLine* **58**(25): 13–18.

Richter, A. V., Le Roux, J. D. and Craig, I. K. (2024b). Bayesian optimization for automatic tuning of a MIMO level controller of a flotation bank, *Submitted to Journal of Process Control* .

Schei, T. S. (1994). Automatic tuning of PID controllers based on transfer function estimation, *Automatica* **30**(12): pp. 1983–1989.

Schubert, J.H., Henning, R.G.D., Hulbert, D.G. and Craig, I.K. (1995). Flotation control - a multivariable stabilizer, *Proceedings of the XIX International Mineral Processing Congress, 1 April 1995, San Francisco, USA*, pp. 237–244.

Seborg, D. E., Mellichamp, D. A., Edgar, T. F. and Doyle, F. J. (2019). *Process Dynamics and Control*, 3rd edn, John Wiley Sons, Hoboken, NJ, USA.

Shahriari, B., Swersky, K., Wang, Z., Adams, R. P. and de Freitas, N. (2016). Taking the human out of the loop: A review of Bayesian optimization, *Proceedings of the IEEE* **104**(1): pp. 148–175.

Shamsuzzoha, M., Skogestad, S. and Halvorsen, I. J. (2010). On-line PI controller tuning using closed-loop setpoint response, *IFAC Proceedings Volumes* **43**(5): pp. 511–516.

Shean, B. and Cilliers, J. (2011). A review of froth flotation control, *International Journal of Mineral Processing* **100**(3-4): pp. 57–71.

Skogestad, S. (2003). Simple analytic rules for model reduction and PID controller tuning, *Journal of Process Control* **13**(4): pp. 291–309.

Skogestad, S. (2023). Advanced control using decomposition and simple elements, *Annual Reviews in Control* **56**: pp. 100903.

Skogestad, S. and Postlethwaite, I. (2007). *Multivariable Feedback Control : Analysis and Design*, 2nd edn, John Wiley, Chichester ; Hoboken, NJ, USA.

Snoek, J., Larochelle, H. and Adams, R. P. (2012). Practical Bayesian optimization of machine learning algorithms, *arXiv preprint arXiv:1206.2944* .

Somefun, O. A., Akingbade, K. and Dahunsi, F. (2021). The dilemma of PID tuning, *Annual Reviews in Control* **52**: pp. 65–74.

Streltsov, S. and Vakili, P. (1999). A non-myopic utility function for statistical global optimization algorithms, *Journal of Global Optimization* **14**(3): pp. 283–298.

Sung, S. W., Lee, I.-B. and Lee, B.-K. (1998). On-line process identification and automatic tuning method for PID controllers, *Chemical Engineering Science* **53**(10): pp. 1847–1859.

Tamura, K. and Ohmori, H. (2007). Auto-tuning method of expanded PID control for MIMO systems, *IFAC Proceedings Volumes* **40**(13): pp. 98–103.

Tan, K., Lee, T. and Jiang, X. (2001). On-line relay identification, assessment and tuning of PID controller, *Journal of Process Control* **11**(5): pp. 483–496.

van Niekerk, J.A., le Roux, J.D. and Craig, I.K. (2022). On-line automatic controller tuning using Bayesian optimisation - a bulk tailings treatment plant case study, *IFAC-PapersOnLine* **55**(21): pp. 126–131.

van Niekerk, J.A., le Roux, J.D. and Craig, I.K. (2023). On-line automatic controller tuning of a multivariable grinding mill circuit using Bayesian optimisation, *Journal of Process Control* **128**: pp. 103008.

Vega (2024). Vegapuls 31 datasheet [Online]. Available at: https://www.vega.com/en-za/products/product-catalog/level/radar/vegapuls-31 (Accessed: 17 August 2024).

Veronesi, M. and Visioli, A. (2011). An automatic tuning method for multiloop PID controllers, *IFAC Proceedings Volumes* **44**(1): pp. 7517–7522.

Voda, A. and Landau, I. (1995). A method for the auto-calibration of PID controllers, *Automatica* **31**(1): pp. 41–53.

Wakitani, S., Nishida, K., Nakamoto, M. and Yamamoto, T. (2013). Design of a data-driven PID controller using operating data, *IFAC Proceedings Volumes* **46**(11): pp. 587–592.

Wang, L. (2017). Automatic tuning of PID controllers using frequency sampling filters, *IET Control Theory & Applications* **11**(7): pp. 985–995.

Wang, X. S., Cheng, Y. H. and Sun, W. (2007). A proposal of adaptive PID controller based on reinforcement learning, *Journal of China University of Mining and Technology* **17**(1): pp. 40–44.

Wilson, J. T., Hutter, F. and Deisenroth, M. P. (2018). Maximizing acquisition functions for Bayesian optimization, *Proceedings of the 32nd Conference on Neural Information Processing Systems, 3-8 December 2018, Montreal, Canada*, pp. 9906–9917.

Woodyatt, A. R. and Middleton, R. H. (1997). Auto-tuning PID controller design using frequency domain approximation, *Proceedings of the European Control Conference (ECC), 1-7 July 1997, Brussels, Belgium*, pp. 1049–1054.

Yang, Q., Xue, Y., Yang, S. X. and Yang, W. (2012). An auto-tuning method for dominant-pole placement using implicit model reference adaptive control technique, *Journal of Process Control* **22**(3): pp. 519–526.

Yu, D., Chang, T. and Yu, D. (2007). A stable self-learning PID control for multivariable time varying systems, *Control Engineering Practice* **15**(12): pp. 1577–1587.

Yu, D.-L., Chang, T. and Yu, D.-W. (2005). Fault tolerant control of multivariable processes using auto-tuning PID controller, *IEEE Transactions on Systems, Man and Cybernetics, Part B (Cybernetics)* **35**(1): pp. 32–43.

Ziegler, J. G. and Nichols, N. B. (1942). Optimum settings for automatic controllers, *Journal of Fluids Engineering* **64**(8): pp. 759–768.