Developing Control Strategies to Regulate Dissolved Oxygen Levels in a Biological Fermenter

by

Omar Khan

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Master of Applied Science
in
Chemical Engineering

Waterloo, Ontario, Canada, 2017

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Abstract

Vaccine development comprises multiple stages, the first of which typically involves cultivating an organism in a microbial fermenter to produce a vaccine product. In order to ensure the optimal synthesis of the vaccine product, it is necessary to maintain adequate control of the dissolved oxygen (DO), which is required for the organism to grow and survive. Our work is concerned with controlling the dissolved oxygen in a biological fermenter using a PID (Proportional-Integral-Derivative) Controller.

The product from this fermenter is used to create the immunization for a medical illness. However, the present configuration of the PID Controller is inadequate for maintaining the dissolved oxygen at the desired level of 35% relative to saturation. This inadequacy results in violent DO oscillations which compromise the quality of the product. To solve this issue, we use open-loop experimental data to develop empirical transfer-function models of the control process for dissolved oxygen. Then, we create an optimization algorithm for the PID Controller and apply it to obtain the proportional, integral, and derivative gains that would best regulate the dissolved oxygen in the fermenter.

The parameters obtained from this algorithm are applied experimentally to the biological fermenter set-up and the results are used to demonstrate that the PID optimization algorithm provides controller settings which successfully regulate the dissolved oxygen. In addition, we employ our transfer-function models of the DO control process to design and configure a set of Internal Model Controllers and Model Predictive Controllers.

The Internal Model and Model Predictive Controllers are subsequently optimized to handle external disturbances and robustly regulate the dissolved oxygen levels. This optimization is performed by varying the tuning parameters of the controllers and selecting the parameters which best maintain the dissolved oxygen levels at their desired values, minimize the effect of external disturbances, and minimize the effect of errors in process modelling. Finally, a non-linear model of the DO control process is developed and utilized to successfully obtain a set of gain-scheduled PID tuning parameters.

Acknowledgements

I would like to thank my supervisors Dr. Chandra Madhuranthakam as well as Dr. Peter Douglas for their continued support throughout the project. I would like to thank our industrial partners for undertaking the fermentation experiments, verifying our control strategies, and providing helpful insight into the control system. I would like to also thank my examiners for taking out the time from their schedules to revise my thesis. Finally, I would like to thank NSERC for providing us with generous funding through their ENGAGE grant throughout the project, as well as the University of Waterloo for their support.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

One of the greatest breakthroughs in medicine has been the discovery of vaccines [1], which are biologically-derived mixtures that prevent the spread of infectious diseases [2]. Vaccines usually function by providing inactivated micro-organisms, or inactivated portions of micro-organisms to cells of the immune system. These inactivated biological preparations present antigens, or immunogenic markers, to immune cells which then initiate a process of growth and proliferation.

The mechanism of vaccine function relies on the adaptive immune response and its ability to generate memory cells [3, 4] which 'remember' the structure of the antigen presented by the vaccine corresponding to a particular disease [5]. When the actual bacteria or viruses of the disease enter the body, the memory cells can multiply much more quickly and generate a much more effective response than the immune response corresponding to primary exposure [6, 7]. The response by the memory cells, called the secondary response, is far more potent and highly capable of quickly eliminating the disease vaccinated against without causing many health-related issues.

Given the usefulness of vaccines in helping eliminate and significantly reduce the spread of otherwise intractable illnesses (e.g. smallpox) [8], vaccine manufacturing is a highly well-known and competitive industry. Generally, vaccine manufacturing is performed using a biological fermenter, or bioreactor [9]. The fermenter grows the micro-organism which produces the vaccine product, usually an inactivated or attenuated antigen to be injected and presented to the patient's immune cells. Because the quality of the fermentation product is highly dependent on the effective growth of the micro-organism, it is necessary to design efficient and optimized processes in order to manufacture the vaccine [10].

The design of efficient and optimized vaccine manufacturing processes is often facilitated by using sophisticated electronic controllers to very closely regulate the growth of the organism during the fermentation process. This regulation is performed by monitoring the physical variables relevant to microbial growth (e.g. temperature, pH, dissolved oxygen) and using the electronic controllers to maintain those physical variables at constant levels.

One of the most effective and widely used control techniques for this purpose is the feedback control strategy. Feedback control has several applications, such as in electronics, machine design, engineering, and pharmaceuticals [11]. The ubiquity of feedback control is due to its ability to effectively monitor a wide range of processes regardless of any external disturbances that may affect the system.

Feedback control normally occurs in the form of a negative feedback loop [12]. It is implemented by measuring the quantity to be controlled, or the controlled variable, and using that measured quantity to make modifications to other process variables (specifically, the manipulated variables) which affect the value of the controlled variable [13]. These modifications are continued until the controlled variable reaches its desired set point [14].

The key component which ensures that proper feedback control is achieved is called the controller [15]. The controller compares the measured value of the controlled variable to its desired set point and generates a signal which is proportionate to the error, or the discrepancy between the desired set point and the measured value of the controlled variable [16]. The signal generated by the controller is used to change the manipulated variable which then alters the value of the controlled variable in order to bring it closer to the desired set point [13, 16, 17].

In the initial stage of vaccine development, during which the biological products are synthesized by the microbes grown in the fermenter, feedback control is used by our industrial partner to regulate multiple physical variables that affect organism growth. One such physical variable at play here is the dissolved oxygen level, which is measured as a percentage relative to saturation.

Because the cells being grown are aerobic, they rely on oxygen to assist with their metabolic processes for growth and energy generation. As a result, it is essential to regulate the levels of dissolved oxygen in the growth medium via feedback controllers. For the

purposes of feedback control, the controlled variable is the dissolved oxygen (henceforth denoted by dO_2), the controller is a Proportional-Integral-Derivative (PID) Controller [18], and the manipulated variable is the stirrer rate (denoted by u).

The PID Controller functions as follows: when the level of dO_2 drops below the desired level (set-point), the controller responds by increasing the stirrer rate, which then ensures that air bubbles are broken up more quickly by the impellers within the fermenter. More broken-up air bubbles result in a larger available surface area for the diffusion of oxygen into the medium. As a result, the level of dO_2 increases [19, 20] towards the set-point. In contrast, when the level of dO_2 rises, the controller responds by decreasing the stirrer rate, which results in a drop in the level of dO_2 towards the set-point.

However, despite the presence of a structured feedback loop, there are problems within the system which prevent the fermenter from being optimally productive. The prevailing issues being experienced in the fermentation process primarily arise from the difficulty in controlling the amount of dO_2 within the fermenter. In a typical experiment, during the first few hours of the cell growth process, there are large, violent oscillations in the levels of dO_2, even though the dissolved oxygen levels are supposed to be kept constant at 35% relative to saturation.

These large oscillations hinder the ability of the culture to grow, and result in numerous difficulties in the productivity and efficiency of the fermentation process. It is hypothesized that these oscillations in the (dO_2) are due to a sub-optimal PID configuration. In other words, the PID controller settings (i.e. the proportional, integral, and derivative gains) are far from the values required to achieve good control.

In order to overcome this issue, we use open-loop experimental data to obtain transfer-function models of the feedback control process for the dO_2 in the microbial fermenter. Using these models, we then propose and design an algorithm to optimize the PID control parameters using the framework developed originally by Madhuranthakam et al [21]. We show that our PID optimization algorithm provides settings which, when applied to the fermenter experimentally, result in successful control of the dO_2 for organism growth.

While the PID optimization algorithm is sufficient to serve as a control strategy for the industrial control of dO_2, it is also insightful to develop multiple control strategies for dO_2 regulation for theoretical use. To this end, we design a set of optimal Internal Model Controllers (IMCs) and Model Predictive Controllers (MPCs) [22, 23]. These controllers are tuned specifically to ensure robust and tight regulation of dO_2 which is relatively un-

hindered by external disturbances. Finally, we use open-loop experimental data collected over a large range of stirrer speeds to devise a non-linear model of the controlled process. Employing a combination of the non-linear model and a variant of the PID optimization algorithm, we obtain a set of gain-scheduled PID parameters which apply for a wider range of stirrer speeds. These controllers are all tested and optimized using control system tools provided in MATLAB (Mathworks, Inc.).

Chapter 2

The Control System

2.1 Overview of the Problem

To produce the components of the vaccine, the organism is grown and incubated in a scaled-down fermenter. As the microbes grow in the medium, they release chemical factors which are collected from the broth and used to synthesize the vaccine. To facilitate organism growth, the fermenter contains baffles which promote mixing, a stirring system comprised of a shaft attached to two impellers, and a sparger which bubbles air into the reaction vessel.

The impellers are present for the purposes of ensuring good mixing, heat transfer, and for breaking up the air bubbles which are injected through the sparger [24]. By breaking up the air bubbles, the impeller ensures a high surface area to volume ratio which promotes the exchange of gases between the bubbles and the liquid medium. Additionally, the blades of the impeller are arranged symmetrically, such that a 60° angle exists between each individual blade.

The stirring system is further attached to a PID controller responsible for regulating the levels of dissolved oxygen. This regulation is performed by increasing the stirrer speed in response to a fall in dO_2 below the set-point. An increase in stirrer speed results in more air bubbles being broken down, which increases the surface area of exchange between the air-water interface and causes more oxygen to dissolve in the growth medium, returning the dO_2 to set-point levels. A rise in dO_2 above the set-point results in a fall in stirrer speed, which eventually causes less oxygen to dissolve in the growth medium, returning the dO_2 to set-point levels.

The PID controller is presently being operated in the PI (proportional-integral) configuration by our industrial partner, with the proportional gain set to $K_c = 0.33$ and the integral time set to $T_i = 150$ seconds. It is also worth noting that the biological fermenter possesses a set of mechanical constraints. Specifically, the stirrer rate is confined to lie between 100 and 675 rpm, while the flow rate of air through the sparger is either 0.5 slpm (standard litres per minute) or 1.8 slpm.

The sparger bubbles gas into the reaction vessel, and contains several holes through which the gas (air) is bubbled. Each hole is 0.5 mm in diameter. The purpose of the sparger is to facilitate gas exchange by bubbling gas into the growing medium. These bubbles provide oxygen to the medium and take away waste gases such as carbon dioxide. The sparger is operated throughout the fermentation process and its air flow rate is regulated by an auxiliary controller.

In a usual fermentation process, the auxiliary controller for the airflow maintains the flow rate of air through the sparger at 0.5 slpm throughout most of the fermentation. It is only when the stirrer rate exceeds 675 rpm (the mechanical upper limit) that the airflow setpoint is set to 1.8 slpm from 0.5 slpm in an effort to increase the dissolved oxygen. Thus, the control of the airflow acts to supplement the main stirrer rate/DO-control loop, by acting as an emergency mechanism to increase the level of dO_2 when the stirrer rate alone is unable to do so. However, according to the data observed, it is very rare for this secondary control mechanism to intervene. Thus, we will ignore it in the development of the process model.

When the fermentation process begins, the broth, which contains the microbes to be grown, is placed in batch mode, where it consumes a carbon source present in the medium. Probes which measure the pH, dissolved oxygen, and temperature are placed into the reactor to record the pH, dO_2, and temperature respectively. The measurements are performed throughout the fermentation process to ensure that these 3 physical variables are tightly regulated at their desired values (36°C for temperature, 7.2 for pH, 35% for dissolved oxygen concentration with respect to its saturation levels).

Next, between 20-35 hours after fermentation begins, a spike in the levels of dO_2 triggers fed-batch mode. The spike in the dO_2 is indicative of a fall in the levels of the carbon source (i.e. existing carbon sources used during the batch more are nearing depletion). The feed to the vessel fed by a feed pump.

Generally, as soon as fermentation commences, the pH and temperature quickly reach their desired levels and rarely deviate from there. However, the dissolved oxygen levels exhibit entirely different behaviour, oscillating wildly for the initial stages of the incubation. This results in a host of problems which negatively affect the growth of the microorganism and, by extension, the quality of the vaccine product. A representative plot of the uncontrolled dissolved oxygen in an experimental run is shown in Figure 2.1.

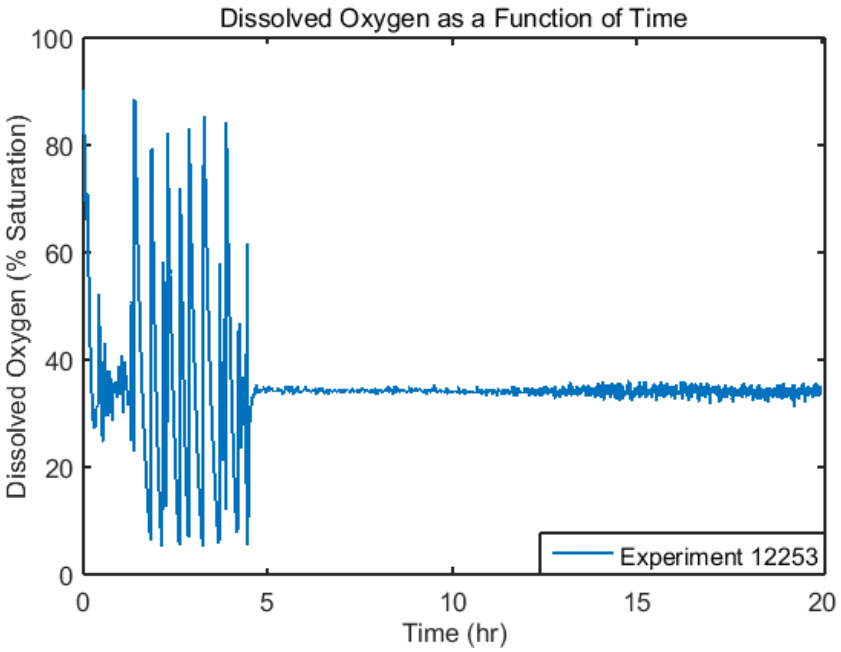

Figure 2.1: Time-series of the dO_2 for the initial stages of the microbial fermentation process. The batch corresponding to this recording is identified as 12253. For the remainder of this work, we will identify experimental recordings by their ID numbers.

In Figure 2.1, it is evident that during the first 5 hours of the cell growth process, there are large, violent oscillations in the levels of dO_2. In usual experimental runs in the scaled-down fermenter, similar oscillations are observed for between 5-10 hours following the start of fermentation. Only after 5-10 hours do these oscillations come under control, and in many cases, control is achieved through an inconvenient manual intervention.

Because broth conditions such as pH and temperature are so easily controlled in a typical operation, we believe that they are not major targets when it comes to identifying factors which bring about violent oscillations in dissolved oxygen, even though the pH and

temperature may influence the efficacy of the growth process. Additionally, because the feed is usually administered long after the dO_2 oscillations occur, we will also ignore the effects of the feed during modelling and controller design. It should also be noted that while noise and measurement artifacts can play a significant role in reducing measurement accuracy, in this case, they do not pose a significant problem. From our current estimates and from the information provided to us, noise and probe-to-probe variation make up only a 5% error.

Given the information discussed above, it is possible that the controller is inherently incapable of maintaining dO_2 at 35% or that there are appreciable changes in the metabolic activity of the organism which cannot be properly dealt with by the current control configuration. Regardless, we hypothesize that it is the inadequacy of the PID controller for dissolved oxygen (currently set to $K_c = 0.33$ and $T_i = 150$s) which results in the dO_2's erratic behaviour.

This hypothesis shall form the basis for the remainder of this study, in which we develop and test multiple control strategies for regulating the dissolved oxygen concentration in the microbial fermenter. This development and testing shall be performed via a two-pronged approach. In the first part of the approach, we shall model the process and create an algorithm to optimize the PID control parameters for the purpose of industrial application. In the second part of the approach, we will use more theoretical control tools (e.g. IMC, MPC Control) and employ a systematic process to arrive at the optimal settings corresponding to these control tools.

In addition to the violent oscillations in dO_2, the industrial fermentation process presents multiple obstacles when it comes to finding the optimal control settings. First, the behaviour of the fermenter changes from batch to batch. As a result, the mathematical model used to describe the system changes from one batch to another. Moreover, the system is known to exhibit nonlinear behaviour: changes in the stirrer rate have different effects on the dO_2, depending on the range within which the stirrer rate is being changed.

Next, for a stirrer rate at around 200-250 rpm, the system exhibits open-loop instability. In other words, even when the feedback is turned off, a constant stirrer rate of 200-250 rpm results in violent dO_2 oscillations. Finally, the controller which regulates the dO_2 by manipulating the stirrer rate is a simple PID controller without a derivative filter. The absence of the derivative filter restricts our ability to use the derivative term, since abrupt changes in the dO_2 level can result in a disastrously large controller output. Due to the lack of controller flexibility, and given the existing complications, modifying a simple

PID controller to achieve the control objectives will require extensive work and accurate modelling, described in the following sections of the thesis.

2.2 A Sketch of the Model

In order to successfully develop an accurate mathematical model for the bioreactor system, it is necessary to develop a foundation on which the model will be based. For a control system, this foundation is often a simple block diagram sketch of the system and its underlying processes. This sketch will serve as the framework for describing the feedback control process. In this section, we shall outline and apply important process information to develop a block diagram model of the control system.

To start, we will assume that the process model has only one major control loop, in which the stirrer rate (constrained to lie between 100 and 675 rpm), is altered by the PID controller in order to regulate the dO_2. In reality, the fermentation process has several control loops, which are all capable of controlling dissolved oxygen one way or the other. These control loops involve temperature, pH, and airflow rate, which all have their own effects on the concentration of dissolved oxygen in the medium. However, as mentioned earlier, these secondary control loops tend not to play a role in a typical operation and so will be justifiably ignored in our model formulation.

To create a preliminary sketch of the model, we will refer to a single-control loop block diagram in Figure 2.2. Each block represents a system which acts on an input in a particular way to provide an output. Generally, the blocks are described by mathematical transfer functions [25]. Transfer functions relate the input to the block with the output from the block, such that the transfer function equals the ratio of the output to the input in the Laplace domain [25, 26]. Note that the block diagram is meant to represent the flow of information in the bioreactor system; it is not necessarily reminiscent of physical connections between different components of the system.

Using our knowledge of the process and the control loop of interest, we can begin to describe what each variable and transfer function in the block diagram represents in terms of the microbial fermentation process:

1. Y_{sp}: The desired set-point of the controlled variable, or the dissolved oxygen concentration Y. In this case, the desired level of dO_2 is 35% of oxygen saturation.

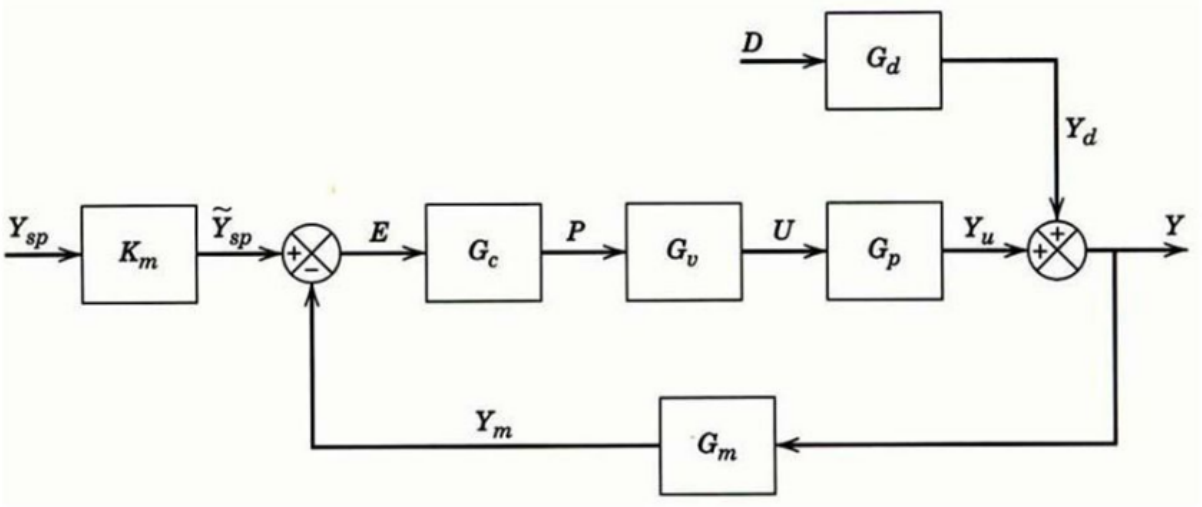

Figure 2.2: Block Diagram of a typical feedback control system. Image taken from [13].

2. $\tilde{Y}_{sp}$: The set-point used by the controller G_c, which is an electric current that represents the value of the dO_2 set-point. We will assume that the transfer function between the actual set-point and its corresponding electric signal is a simple proportional gain K_m (i.e. there are no delays).

3. E: The error signal, or the difference between the measured value of the dO_2 and its set point. This signal is fed to our controller.

4. G_c: The controller of the system. It processes the error signal and provides an output signal depending on the magnitude of the error signal. The default industrial setup uses a PID controller in which the derivative term is set to zero. The transfer function is given by:

$$G_c(s) = K_c\left(1 + \frac{1}{T_i s} + T_d s\right) \tag{2.1}$$

where K_c represents the proportional gain (default setting: 0.333), T_i represents the integral time (default setting: 150 s), and T_d represents the derivative time (default setting: 0). In a typical PID controller, there is also a low-pass filter in the derivative term, which acts to limit the effect of sudden variations in the error signal due to external noise. Here, we have omitted the low-pass filter because the industrial setup does not contain a derivative filter. In addition to using a simple PID controller for this block, we will also substitute IMC, MPC, and a gain-scheduled PID controller for G_c later in this study.

5. G_v: The final control element. This device converts an electrical output signal from the controller into a physical signal which changes the stirrer speed, denote by the process variable U. From experimental data, we have determined that $G_v = 20$, which means that the signal from the controller is multiplied by 20 to give the stirrer rate. In addition, the controller output is directly conveyed to the stirrer without any significant delays. Furthermore, because the stirrer rate is constrained to lie between 100 and 675 rpm, we will also incorporate a saturation block into the final control element to ensure that these constraints are obeyed.

6. U: The manipulated variable, which is adjusted by the final control element according to the controller output signal P. In this fermenter, the manipulated variable is the stirrer rate.

7. G_p: The plant or process, which receives an input from the manipulated variable U and outputs part of the controlled variable Y_u (the dO_2). To determine G_p, differential equation models for organism growth may be developed. Alternatively, step-response experiments may be carried out to obtain an empirical estimate of this transfer function (refer to chapter 3 for details).

8. D: The disturbance variable. For now, we will neglect the effects of any disturbance variables on the system, since according to our current information, noise does not play a huge role in altering our measurements or our system. However, optimizing our controller settings with respect to disturbance rejection will serve as a potential control strategy later in our work.

9. G_d: The disturbance transfer function, which receives a disturbance input and uses that input to bring about a change in the controlled variable Y. Since the disturbance variable is initially neglected, we will ignore this transfer function as well for the time-being, and will take it into account later on in the report.

10. Y: The overall controlled variable, equivalent to the dissolved oxygen concentration. Note that there are two contributions to the controlled variable: the contribution Y_d due to the disturbance and the contribution Y_u due to the change in the stirrer rate.

11. G_m: Sensor and transmitter. This device measures Y and converts that measurement into an electrical signal which may be processed by the controller. We will assume that the sensor instantaneously measures the dissolved oxygen concentration and that there are no significant delays in the measurement process. Any actual measurement delays will be incorporated into the plant block G_p. As a result, we will set $G_m = 1$.

Chapter 3

Determining Plant Model

3.1 Step-Response Open-Loop Experiments

The crucial element missing from our mathematical model described in section 2.2 is the plant model, or the process transfer function G_p. As mentioned earlier, it is possible to obtain G_p by using a mechanistic model - that is, developing differential equations to describe microbial growth and using those differential equations to find a transfer function between the stirrer speed and dO_2 [27]. Unfortunately, due to the lack of data for the growth rate of the cells and the concentration of the carbon source throughout the fermentation process, it is difficult to come up with a pure mechanistic model for the processes occurring within the bioreactor.

Nevertheless, it is possible to develop an empirical model using carefully designed step-response experiments conducted by our industrial partner. In these experiments, the controller is turned off and the measurement probe for dissolved oxygen is removed from the medium. This results in the experiment being conducted in 'open-loop' mode, because the feedback to the controller in the control loop (Figure 2.2) is now cut off [13, 14]. Under these open-loop conditions, the stirrer rate becomes constant and the dO_2 of the system eventually settles. Once the system settles to a steady state, the stirrer speed is changed in a stepwise fashion and the resulting changes in dissolved oxygen are recorded.

The rationale behind performing step-response experiments between stirrer rate and dissolved oxygen is that the data obtained from these experiments can be used to empiri-

cally determine the process transfer function G_p. Because G_p is defined as:

$$G_p(s) = \frac{Y(s)}{U(s)} \tag{3.1}$$

where $Y(s)$ is the dissolved oxygen in the Laplace domain while $U(s)$ is the stirrer speed in the Laplace domain, a known step change in the stirrer rate $U(s)$, as well as a known trajectory followed by the dissolved oxygen $Y(s)$, is enough information to estimate G_p. More specifically, the estimation of G_p is carried out by fitting known, pre-defined transfer functions with unknown parameters to experimental step-response results.

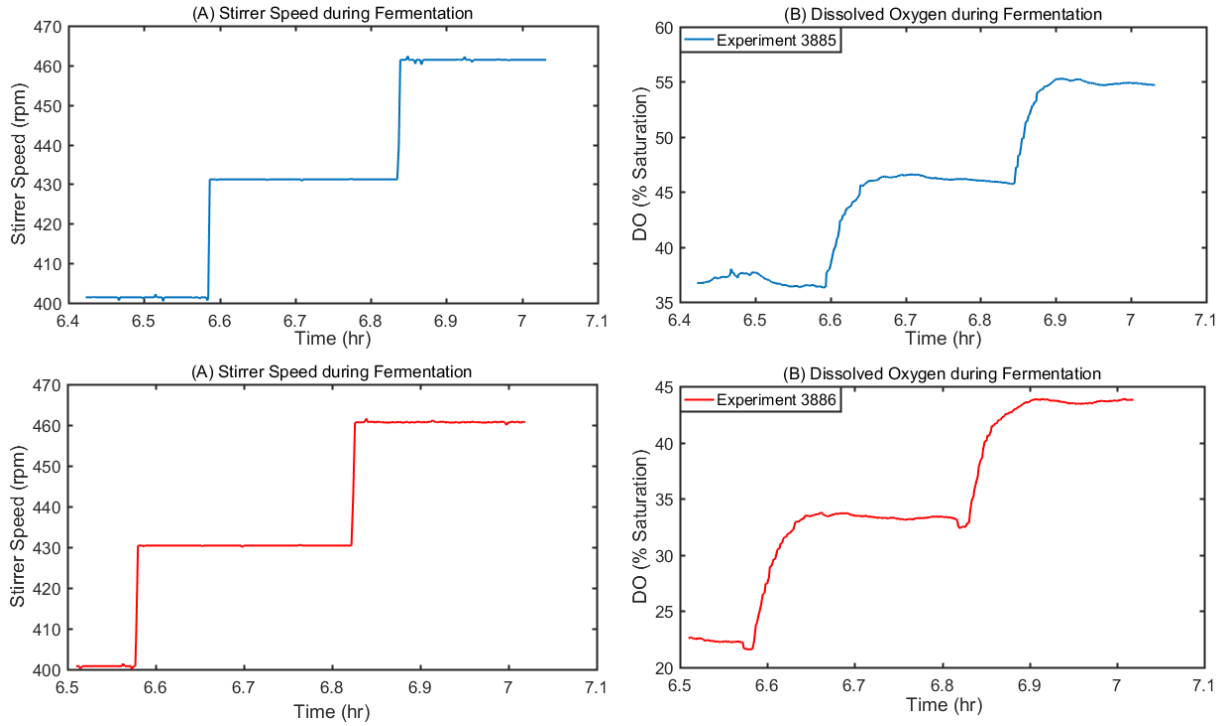

Figure 3.1: Stirrer Rate and dissolved oxygen data during the open-loop phase in the fermentation experiments 3885 and 3886. Notice how the step changes in the stirrer rate bring about exponential-like responses in the dissolved oxygen.

Four experiments were performed in which the stirrer rate was varied from steady state in a stepwise fashion, and the resulting dynamics of the dissolved oxygen were recorded.

Figure 3.1 shows the results of batches 3885 and 3886, while figure 3.2 shows the results of batches 12429 and 12430. In experiments 3885 and 3886, the stirrer speed started at 400 rpm, and was first stepped up by 30 rpm to 430 rpm. Once the system was allowed sufficient time to reach a new steady state, another step change of 30 rpm was performed, and the resulting changes in dissolved oxygen recorded.

In experiments 12429 and 12430, the stirrer rate was increased from a starting value of 450 rpm to 500 rpm. Once the system reached steady state, the stirrer rate was decreased to 475 rpm, and the resulting changes in dissolved oxygen were recorded throughout these step changes. Because the observed behaviour of the dissolved oxygen changes from one experiment to another, it is important to both propose and use multiple transfer functions G_p to describe the process. The proposal, estimation, and isolation of the best transfer function models for each experimental batch shall constitute the main goal of the remainder of this chapter.

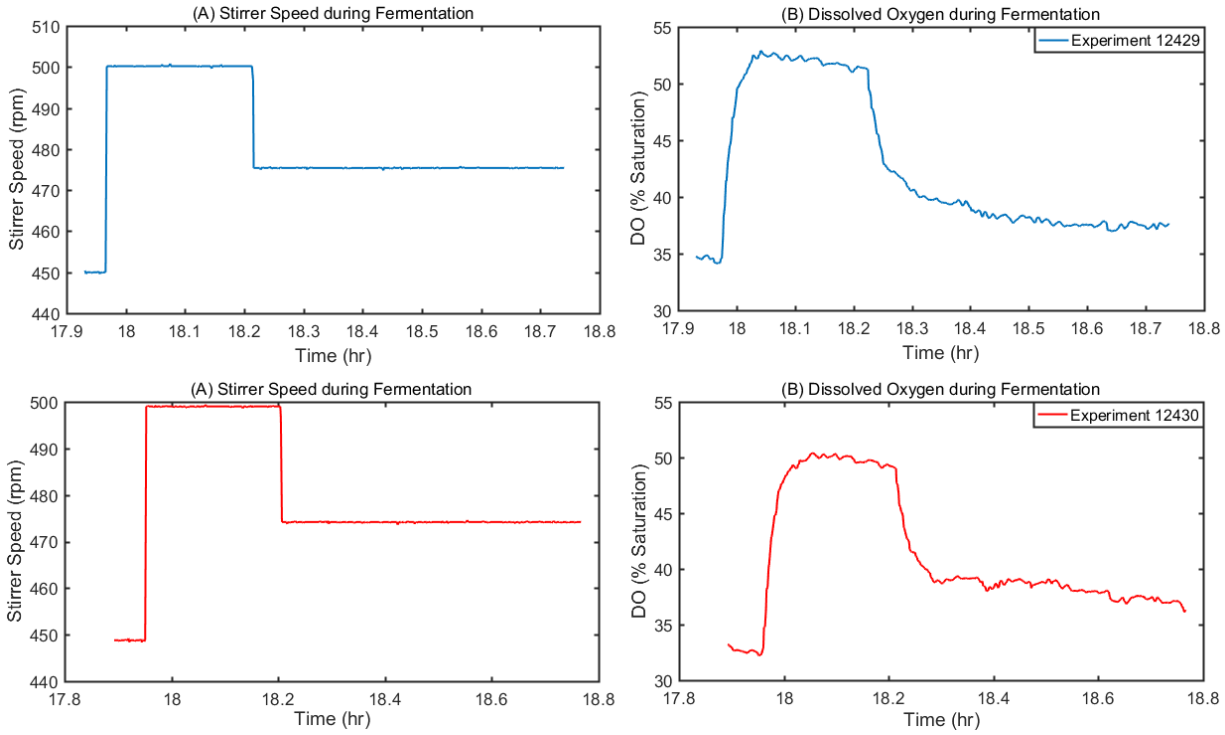

Figure 3.2: Stirrer Rate and dissolved oxygen data during the open-loop phase in the fermentation experiments 12429 and 12430. Notice how the step changes in the stirrer rate bring about exponential-like responses in the dissolved oxygen.

3.2 Proposal and Selection of Transfer Function Models

The first step in finding the best transfer function models for each experimental batch is proposing multiple candidate models. Here, we proposed 5 potential transfer functions that described the relationship G_p between the dissolved oxygen $Y(s)$ and the stirrer speed $U(s)$. These transfer functions are listed and described below:

1. **First-Order Process (FOP):** The transfer function for a first-order process is given by:
$$\frac{Y(s)}{U(s)} = \frac{K}{(\tau_1 s + 1)} \tag{3.2}$$
This relationship represents a first-order process due to the presence of only one pole (i.e. root of the denominator). Physically, it represents a system in which a step change in the manipulated variable will result in the controlled variable approaching a new steady state in an exponential fashion with a single time constant τ_1.

2. **Second-Order Process (SOP):** The transfer function for a second-order process is given by:
$$\frac{Y(s)}{U(s)} = \frac{K}{(\tau_1 s + 1)(\tau_2 s + 1)} \tag{3.3}$$
This expression represents a second-order process, because it contains two poles. Physically, it represents a system in which a step change in U is met by a biexponential response by Y, in which there are two time constants - τ_1 and τ_2.

3. **Second-Order Process with Lead (SOPLD):** The transfer function for a second-order process with lead is given by:
$$\frac{Y(s)}{U(s)} = \frac{K(\tau_3 s + 1)}{(\tau_1 s + 1)(\tau_2 s + 1)} \tag{3.4}$$
This equation represents a variation on the second-order process (Equation 3.3), and is achieved by adding a lead or a zero. Physically, a zero represents a component of the transfer function which causes the output to 'lead' the input in phase, when the input and outputs are sinusoidal. In the case of a step change in U, a zero simply acts to modulate the coefficients of the exponentials (whose time constants are obtained from the denominator of the transfer function) in the response of the controlled variable Y. In particular, a zero in the transfer function can bring about

an overshoot in the controlled variable Y when there is a step change in U. This overshoot has been observed in the experimental recordings, and there is reason to believe that a zero is necessary to describe the dynamics of the plant.

4. **Second-Order Process with Time Delay and Derivative (SOPTDD):** By neglecting the constant term in the numerator of Equation 3.4, it is possible to obtain a model which serves as a more 'extreme' version of a zero. We will refer to this model as a second-order process with time delay and derivative (SOPTDD), whose transfer function is described below:

$$\frac{Y(s)}{U(s)} = \frac{Ke^{-\theta_d s}s}{(\tau_1 s + 1)(\tau_2 s + 1)} \tag{3.5}$$

   The term derivative is used to refer to the numerator of equation 3.5 which contains a lone s, much like the derivative term of a PID controller, that only has s in its numerator. Physically, the derivative represents zero gain at very low input frequency (i.e. $s = 0$). Specifically, a step change in U will cause a large overshoot in Y, but after some time, that overshoot will decay to zero according to the Final Value Theorem. This means that the cells in the bioreactor will eventually completely consume the dO_2 if the stirrer rate is not increased to match the cell's consumption. In addition, the SOPTDD model also exhibits a time delay term, represented by the exponential term in the numerator. Physically, this means that Y will be delayed by time θ_d whenever there is a step change in U. For our bioreactor, the presence of a non-zero θ_d will be used to indicate delays in the measurement of dO_2, such that the current dO_2 reading actually corresponds to the level of dissolved oxygen θ_d seconds back in time.

5. **Second-Order Process with Time Delay and Lead (SOPTDLD):** Finally, the last transfer function model used to describe our plant is a second-order process with time delay and lead (Equation 3.6). Its transfer function is shown below:

$$\frac{Y(s)}{U(s)} = \frac{Ke^{-\theta_d s}(\tau_3 s + 1)}{(\tau_1 s + 1)(\tau_2 s + 1)} \tag{3.6}$$

   Like the SOPLD and SOPTDD, this process is an extension to the simple second order process in equation 3.3, but now contains both a lead and time delay.

Open-loop data from each of the four experiments shown in Figures 3.1 and 3.2 were fit to each of the five transfer functions proposed. This resulted in a total of 20 fits, with 5 different transfer function fits for each experimental batch.

For a particular experimental batch, the transfer function fits were accomplished using the 'tfest' function in MATLAB. The fits were also refined by running the 'tfest' function 20 more times, using the final guess from the previous iterate as the initial guess for the current iterate. Then, each fit was plotted against the experimental data for comparison. For every experimental batch we computed the residuals ($e_{xi} = y_i - y_x$, where y_i is the ith experimental data point and y_x is the corresponding value from model x) for dissolved oxygen corresponding to a particular model fit.

We then analysed the correlation, distribution, and frequency characteristics of these residuals to check the quality of our fit. In addition, we computed the corresponding sum of square error (SSE) for each fit, which we denote as $\mathrm{SSE_x} = \sum e_{xi}^2$. The SSE for each fit was used as a key metric to assess the quality of the fit to a particular transfer function in relation to the fits from the other transfer functions. We also computed the degrees of freedom corresponding to each fit using the following relation:

$$d_x = N - p_x \tag{3.7}$$

where d_x is the number of degrees of freedom for model x, N is the number of data points used for fitting the experiment, and p_x is the number of parameters in model x. In this case, the number of parameters for the five models are 2, 3, 4, 4, and 5 for the FOP, SOP, SOPLD, SOPTDD, and SOPTDLD transfer functions respectively.

For each batch, we used the quality of the fits of each transfer function, which were indicated by the sum of squared errors and the degrees of freedom for each fit [28], to determine the best possible transfer function model describing that batch. The relative quality of the fits was judged by using the F-test [44, 30] to perform comparisons between the five models and find the most parsimonious transfer function model for a particular experimental batch.

Each F-test comparison executed for a particular experimental batch had a purpose in deciphering the physical behaviour of the bioreactor system. Because of the computational load and lack of usefulness of a comprehensive series of F-tests, in which every model was compared to every other model, we used only five F-tests.

The selection of these F-tests was based on choosing comparisons which provide the most information. For instance, we only compared the Equation 3.2 model to the Equa-

tion 3.3 model to check whether two poles were necessary to describe the plant's dynamics. It was not necessary to compare Equation 3.2 to Equation 3.6 if we already found that the second order process was a better descriptor of the system's response. Overall, the following list defines and describes the F-test comparisons performed:

1. Between equations 3.2 and 3.3:

$$F_{FS} = \frac{\frac{\mathrm{SSE_{FOP}} - \mathrm{SSE_{SOP}}}{\mathrm{d_{FOP}} - \mathrm{d_{SOP}}}}{\mathrm{SSE_{SOP}}/\mathrm{d_{SOP}}} \tag{3.8}$$

   The purpose of this F-test is to determine whether or not an additional pole was necessary to describe the behaviour of a particular experimental dataset.

2. Between equations 3.3 and 3.4:

$$F_{SSLD} = \frac{\frac{\mathrm{SSE_{SOP}} - \mathrm{SSE_{SOPLD}}}{\mathrm{d_{SOP}} - \mathrm{d_{SOPLD}}}}{\mathrm{SSE_{SOPLD}}/\mathrm{d_{SOPLD}}} \tag{3.9}$$

   The goal of this F-test is to check whether or not a lead or zero is required when a second-order model is already present.

3. Between equations 3.4 and 3.6:

$$F_{SLDSLDTD} = \frac{\frac{\mathrm{SSE_{SOPLD}} - \mathrm{SSE_{SOPTDLD}}}{\mathrm{d_{SOPLD}} - \mathrm{d_{SOPTDLD}}}}{\mathrm{SSE_{SOPTDLD}}/\mathrm{d_{SOPTDLD}}} \tag{3.10}$$

   This F-test is meant to determine whether a time delay is needed for a model which already has two poles and a zero.

4. Between equations 3.4 and 3.5:

$$F_{SLDSTDD} = \frac{\mathrm{SSE_{SOPLD}}}{\mathrm{SSE_{SOPTDD}}} \tag{3.11}$$

   This F-statistic compares directly between two second order models, one of which has only a lead, while the other has a time delay and a 'derivative' term.

5. Between equations 3.5 and 3.6:

$$F_{STDDSLDTD} = \frac{\frac{\mathrm{SSE_{SOPTDD}} - \mathrm{SSE_{SOPTDLD}}}{\mathrm{d_{SOPTDD}} - \mathrm{d_{SOPTDLD}}}}{\mathrm{SSE_{SOPTDLD}}/\mathrm{d_{SOPTDLD}}} \tag{3.12}$$

   The F-statistic above weighs the quality of a model with a trivial zero (SOPTDD) with the quality of a model with a non-trivial zero (SOPTDLD).

Like many other statistical measures, F-values lie on a parent distribution - the F-distribution. The shape of the F-distribution depends on the two degree-of-freedom parameters associated with the F-statistic. When an F-value is calculated according to one of the equations above, its relative location on the distribution may be determined. This location gives rise to a p-value, which denotes the likelihood of finding an F-value greater than that calculated by the F-statistic.

If this likelihood determined from the p-value is sufficiently small (below 0.05 for our work), then that means the F-statistic calculated is statistically significant. In simpler terms, this means that there is a statistically significant difference between the two models compared via the F-test. For example, if F_{FS} is sufficiently large that its p-value is less than 0.05, then the SOP model gives a significantly better fit than the FOP model because the SOP model has a far lower SSE for a marginally higher number of parameters.

However, if F_{FS} was small enough that its p-value was greater than 0.05, then the SOP model may or may not give a more accurate fit, but the improved accuracy is not enough to make up for the extra parameter in the SOP model. Thus, in the latter case, the FOP model would be chosen as the superior option. An alternative technique to assess the most parsimonious model for a particular data set is the Akaike Information Criterion (AIC) [31]. For a transfer function model x with p_x parameters, the AIC is defined by:

$$\mathrm{AIC_x} = 2p_x - 2\ln L \tag{3.13}$$

where L denotes the likelihood function evaluated by comparing the experimental data to the results from fitting the transfer function model. This likelihood function denotes the probability that a model with a particular set of estimated parameters describes the underlying experimental data. It is common practice to express the likelihood function as the inverse of an error function. In MATLAB, for example, the expression for the AIC is written as follows:

$$\mathrm{AIC_x} = N\left[2\ln(2\pi) + 1\right] + 2p_x + \ln\left[\det\left(\frac{1}{N}\sum_{i=1}^{N} e_{xi}^2\right)\right] \tag{3.14}$$

where N is the number of data points in the batch and e_{xi} is the residual for the ith data point in model x. Equation 3.14 penalizes models with a large number of parameters, and also penalizes models that are less accurate (i.e. high residuals). Therefore, the AIC is very similar to the F-test, in that it seeks to determine the model which achieves the

optimal balance between accuracy and simplicity. Generally, the smaller the AIC value for a model, the better that model is at achieving this balance.

The AIC and F-test will both be used to determine the best-fit model equation for each experimental batch. Once the best-fit model is determined, it will be examined further to ascertain the quality of the fit and the nature of its residuals. If the fit and residuals are sufficiently good, then the model will be used as the G_p for its corresponding experimental batch when the optimal control strategies are determined starting from Chapter 4.

3.3 Results of Transfer Function Model Fits

Five fits to each of the four experimental batches were performed, starting with batch 3885. Table 3.1 shows the parameter values resulting from fitting each model to experiment 3885, while Table 3.2 shows the results of the AIC and F-tests used to compare the models. Interestingly, the pure second-order model and the second-order model with the lead does not exhibit real poles, since the discriminant of the quadratic expression in their denominators (see table 3.1, rows 2 and 3) is negative. Thus, the fits corresponding to these models show weak oscillatory behaviour. It should also be noted that the zeros in the SOPTDLD and the SOPLD models are both negative.

In table 3.2, the second column shows the values of the AIC for each model fit, and indicates that the 2nd order model with time delay and derivative is the best fit for the experiment. This finding is corroborated by the F-tests and their corresponding p-values, shown in the fifth and sixth columns respectively. The F-tests demonstrate that the second-order model is superior to the first-order model (negligible p-value in 1st entry), and that the lead and time delay with lead are superior alternatives to a simple second-order model (small p-values in the 2nd and 3rd entry).

Additionally, the results show that while the derivative model is preferable to the second-order model with lead only (refer to the F-test in the 4th row), the derivative model is not significantly worse than the SOPTDLD fit, since the p-value for that comparison is greater than our significance bound of 0.05. This suggests that the SOPTDD model is a superior alternative to the SOPTDLD model because it has fewer parameters. Furthermore, the AIC for the SOPTDD model is slightly lower than that for the SOPTDLD model. Therefore, from both the F-tests and AIC, we conclude that equation 3.5

represents the best descriptor of the process transfer function in experiment 3885.

For a stronger visual comparison, the fits to experiment 3885 of all five models have been plotted. These plots are shown in Figures 3.3 and 3.4. Note that the SOP and SOPLD fits show very slight oscillatory behaviour, which is consistent with the obtained forms of their transfer functions. Note also that the SOPTDD and SOPTDLD fits in Figure 3.4 essentially overlap with each other, indicating that these two models exhibit roughly the same accuracy, even though the SOPTDD model is the simpler of the two.

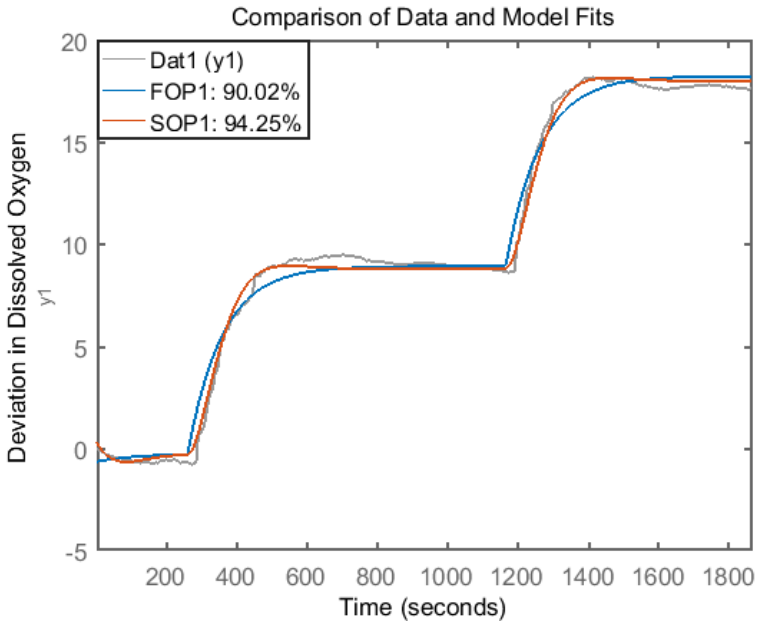

Figure 3.3: Comparison of Experimental Data (grey) to the fits from the FOP and SOP models. Legend shows the quality of the fit in percentage for each model (100% denotes a completely accurate fit).

Model Type	Parameter Values
1st Order (3.2)	$K = 0.309$, $\tau_1 = 98.2$ s
2nd Order (3.3)	$K = 0.305$, Denominator: $1 + 1.61(54.8s) + (54.8s)^2$
2nd Order, Lead (3.4)	$K = 0.306$, $\tau_3 = -9.50$, Denominator: $1 + 1.80(45.5s) + (45.5s)^2$
2nd Order, Derivative, Time Delay (3.5)	$K = 2033$, $\tau_1 = 86.0$ s, $\tau_2 = 5656$ s, $\theta_d = 24$ s
2nd Order, Lead, Time Delay (3.6)	$K = -2.55$, $\tau_1 = 85.8$ s, $\tau_2 = 52310$ s, $\tau_3 = -7356$ s, $\theta_d = 24$ s

Table 3.1: Parameter values obtained from fitting the 5 models to the dataset for experiment 3885. When possible, parameters are listed in a manner consistent with the corresponding model equations.

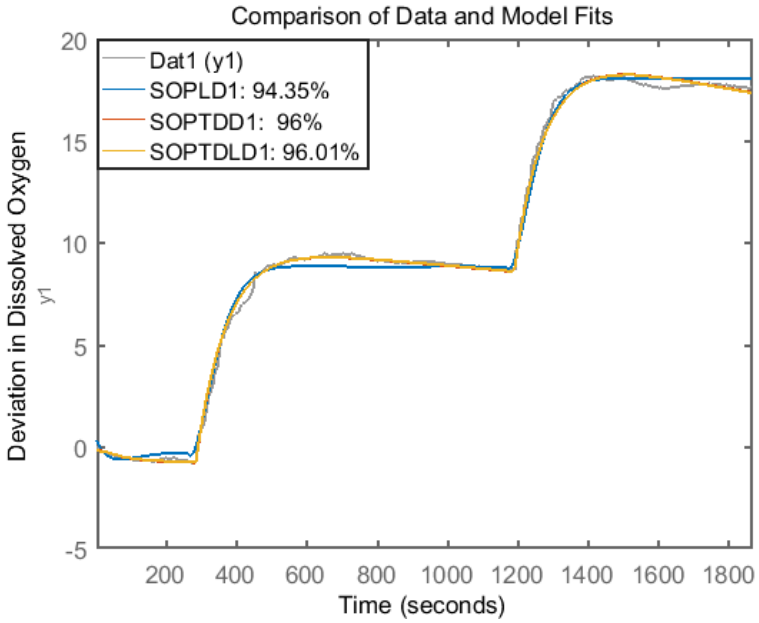

Figure 3.4: Comparison of Experimental Data (grey) to the fits from the SOPLD, SOPTDD, and SOPLDTD models. Legend shows the quality of the fit in percentage for each model (100% denotes a completely accurate fit).

Model Type	AIC Value	SSE	Degrees of Freedom	F-Statistics	P-Value
1st Order (3.2)	600.9	123.3	309	$F_{FS} =$ 620.5	0
2nd Order (3.3)	261.7	40.9	308	$F_{SSLD} =$ 10.25	0.0015
2nd Order, Lead (3.4)	253.5	39.6	307	$F_{SLDSLDTD} =$ 307.6	0
2nd Order, Derivative, Time Delay (3.5)	37.0	19.9	307	$F_{SLDSTDD} =$ 1.99	1.12×10^{-9}
2nd Order, Lead, Time Delay (3.6)	37.2	19.7	306	$F_{STDDSLDTD} =$ $1.85s$	0.175

Table 3.2: Comparing the 5 transfer function models fits to the dataset for experiment 3885. From both the AIC and the p-test, we can conclude that the SOPTDD model (equation 3.5) is the best description for the process in experiment 3885.

Each transfer function fit performed has a set of corresponding residuals e_{ix}, which denotes the difference between the experimental data and the corresponding points obtained from the empirical model fit. In order to fully verify that the SOPTDD model represents a good fit for batch 3885, we have plotted the power spectrum [33], probability distribution [32], autocorrelation, and cross-correlation [30] for the residuals of the SOPTDD fit. The results of this plot are shown in Figure 3.5.

In an ideal scenario, the residuals of a fit should exhibit both whiteness and independence. How 'white' the residuals are can be determined using an autocorrelation plot of the residuals, in addition to a power spectrum plot. If the residuals are purely white (i.e. exhibit nearly constant power at all frequencies and do not appear to exhibit a certain frequency), then the autocorrelation plot will be a Kronecker-Delta function, with a peak at zero lag and insignificant correlations at all other lags. Moreover, the power spectrum plot will show relatively constant power for all frequencies.

Similarly, residuals that are independent show insignificant cross-correlation at all lags. If residuals are independent, then that means the model adequately describes the relation between the given input (stirrer rate) and the output (dissolved oxygen), and there are no missing terms in the input-output relation used. Another diagnostic tool used to assess the quality of the residuals is their distribution. Residuals that are relatively normally distributed with a mean around zero indicate that the model is reasonably good.

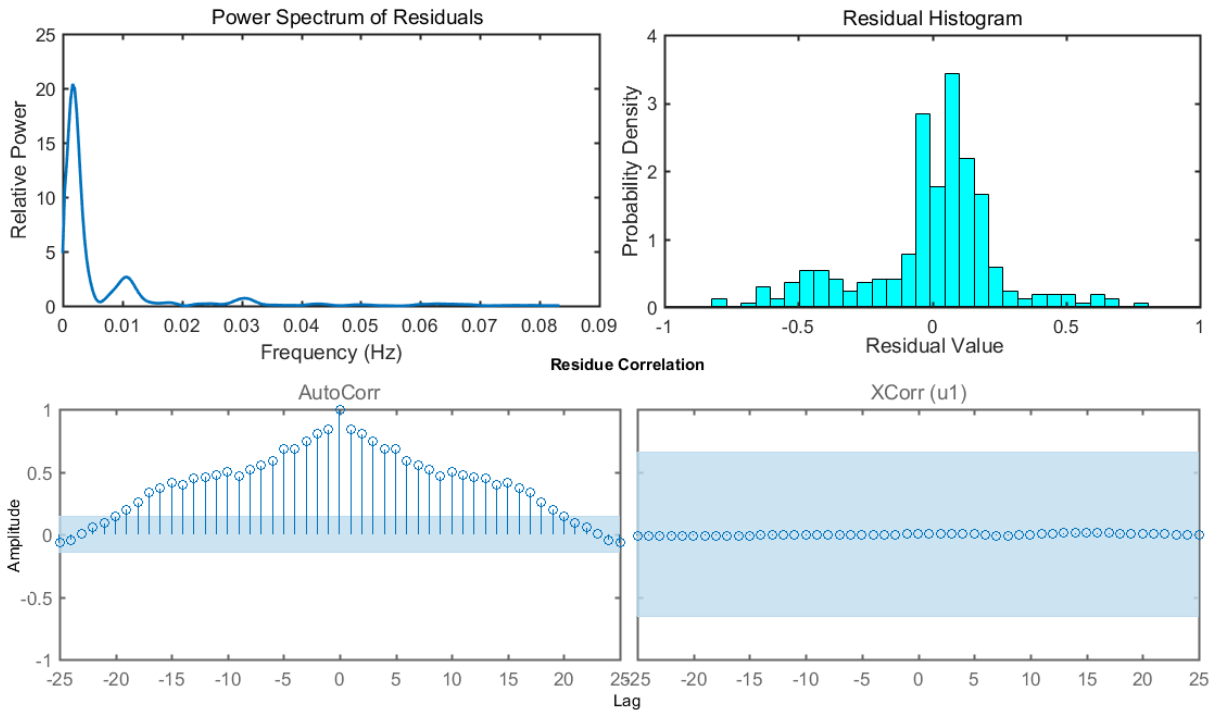

Figure 3.5: **Top Left:** Power Spectrum of Residuals. **Top Right:** Amplitude Distribution of the Residuals. **Bottom:** Correlation Plots for SOPTDD residuals when fit to Experiment 3885. Left panel shows autocorrelation between residuals and right panel shows cross-correlation between residuals and the input (i.e. the stirrer rate). 99% confidence band is shown by the blue region in both panels.

Figure 3.5 shows that the residuals resulting from the SOPTDD fit to batch 3885 exhibit high power at low frequency (see top left of figure for power spectrum), and are relatively normally distributed with a mean of roughly zero. Moreover, the residuals for this fit exhibit independence because the cross-correlations are relatively insignificant with respect to the 99% confidence band. Because of the insignificance of the cross-correlations to the input and the relatively normal distribution of the residuals, we conclude that the SOPTDD transfer function fit is sufficient to describe the variations in dissolved oxygen when there are changes in the stirrer speed.

However, the residual characteristics also suggest the weaknesses of the SOPTDD fit. High power at low frequency and significant autocorrelations at non-zero lags imply that the residuals lack 'whiteness'. A potential reason for the lack of 'whiteness' is the absence

Model Type	Parameter Values
1st Order (3.2)	$K = 0.356$, $\tau_1 = 92.3$ s
2nd Order (3.3)	$K = 0.353$, Denominator: $1 + 1.81(47.3s) + (47.3s)^2$
2nd Order, Lead (3.4)	$K = 0.354$, $\tau_1 = 15.9$ s, $\tau_2 = 60.4$ s, $\tau_3 = -13.8$ s
2nd Order, Derivative, Time Delay (3.5)	$K = 3167$, $\tau_1 = 80.8$ s, $\tau_2 = 8009$ s, $\theta_d = 20$ s
2nd Order, Lead, Time Delay (3.6)	$K = 0.326$, $\tau_1 = 89.0$ s, $\tau_2 = 564.7$ s, $\tau_3 = 719.1$ s, $\theta_d = 18$ s

Table 3.3: Parameter values obtained from fitting the 5 models to the dataset for experiment 3886. When possible, parameters are listed in a manner consistent with the corresponding model equations.

of a disturbance term from our transfer function model. In other words, the model fails to account for the disturbance which gives rise to noise and variations in the dissolved oxygen that creates the 'non-whiteness' and suboptimal frequency characteristics.

The data from experiment 3886 are also fit to the 5 transfer function models, and the models are compared using F-tests and the AIC test. The parameter values from the 5 fits are shown in Table 3.3 while the results of the model comparisons are shown in Table 3.4. From table 3.3, it is observed that while the fit to the second-order process gives complex poles, and hence some slight oscillatory behaviour, all other model fits give real poles, unlike experiment 3885.

In Table 3.4, the F-tests indicate that models with more poles, an additional lead, and an additional time delay give the best fit results while keeping the number of parameters low. This finding is corroborated by the AIC tests, in which added levels of complexity lower the AIC. The end result is that the second-order process with lead and time delay (equation 3.6) is the best fit for the data in batch 3886. This is in contrast to batch 3885, where the best-fit transfer function was the SOPTDD process model.

Model Type	AIC Value	SSE	Degrees of Freedom	F-Statistics	P-Value
1st Order (3.2)	564.8	110.9	305	$F_{FS} =$ 562.6	0
2nd Order (3.3)	247.2	38.9	304	$F_{SSLD} =$ 33.0	2.20×10^{-8}
2nd Order, Lead (3.4)	217.4	35.1	303	$F_{SLDSLDTD} =$ 221.9	0
2nd Order, Derivative, Time Delay (3.5)	61.2	21.2	303	$F_{SLDSTDD} =$ 1.65	6.92×10^{-6}
2nd Order, Lead, Time Delay (3.6)	48.3	20.2	302	$F_{STDDSLDTD} =$ 15.0	1.31×10^{-4}

Table 3.4: Comparing the 5 transfer function models fits to the dataset for experiment 3886. From both the AIC and the p-test, we can conclude that the SOPTDLD model (equation 3.6) is the best description for the process in experiment 3886.

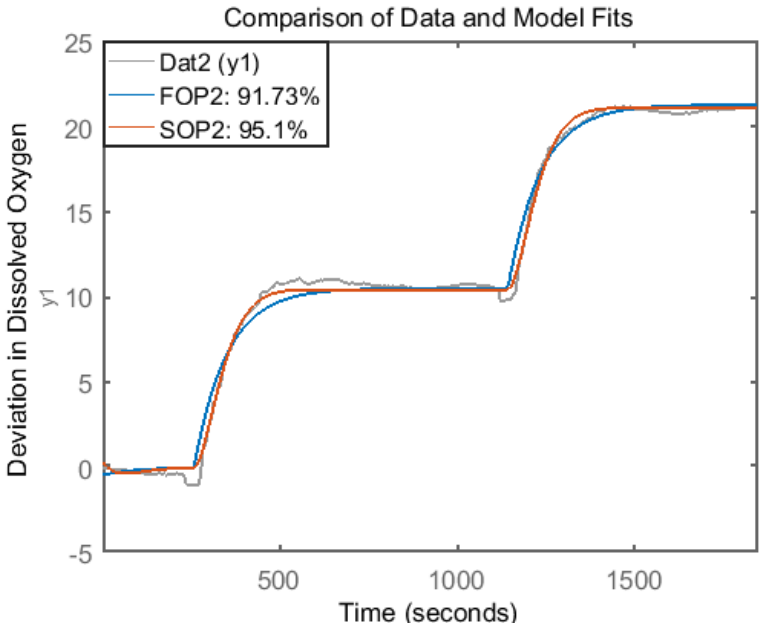

Figure 3.6: Comparison of Experimental Data (grey) to the fits from the FOP and SOP models. Legend shows the quality of the fit in percentage for each model (100% denotes a completely accurate fit).

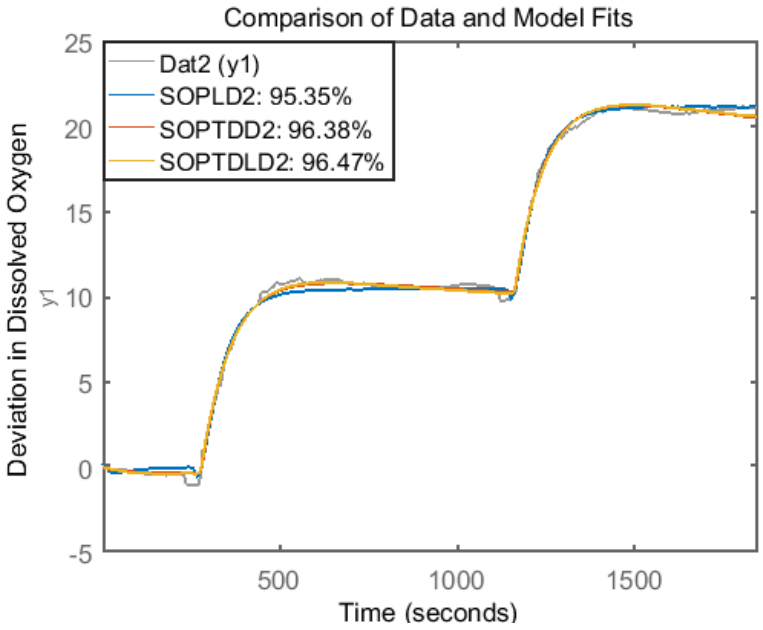

Figure 3.7: Comparison of Experimental Data (grey) to the fits from the SOPLD, SOPTDD, and SOPLDTD models. Legend shows the quality of the fit in percentage for each model (100% denotes a completely accurate fit).

For visual comparison, the fits corresponding to each model describing experiment 3886's data have been plotted in Figures 3.6 and 3.7. From both plots, we find that the SOPTDLD model is the most accurate when describing the behaviour in the observed experimental data, showing a 96.47% fit quality. Since the SOPTDLD model was the best descriptor of the step response data from experiment 3886 according to the F-tests, AIC, and response plots, our next step in the model verification was to fully determine the quality of its fit using residual analysis.

The residual analysis for batch 3886 followed exactly the same steps as that for batch 3885. It consisted of computing and plotting the power spectrum, the amplitude distribution, the autocorrelation and the cross-correlation with the input (i.e. stirrer rate) of the residuals from the SOPTDLD fit (Figure 3.8). The cross-correlation was found to be insignificant for all calculated lags, while the distribution of the residuals was found to have a mean of about zero and a somewhat normal shape with the exception of a large peak near zero. This indicated that the transfer function model adequately described the relation between the dissolved oxygen and the stirrer speed.

On the other hand, the autocorrelation was found to be significant for smaller lags, while the power spectrum was demonstrated to contain more power at the lower frequencies, just as for batch 3885. Again, this lack of whiteness is attributed to our ignoring of the disturbances in the formulation of our transfer function models. The low-frequency behaviour of the residuals also suggests that the external disturbance to the microbial system is low frequency noise.

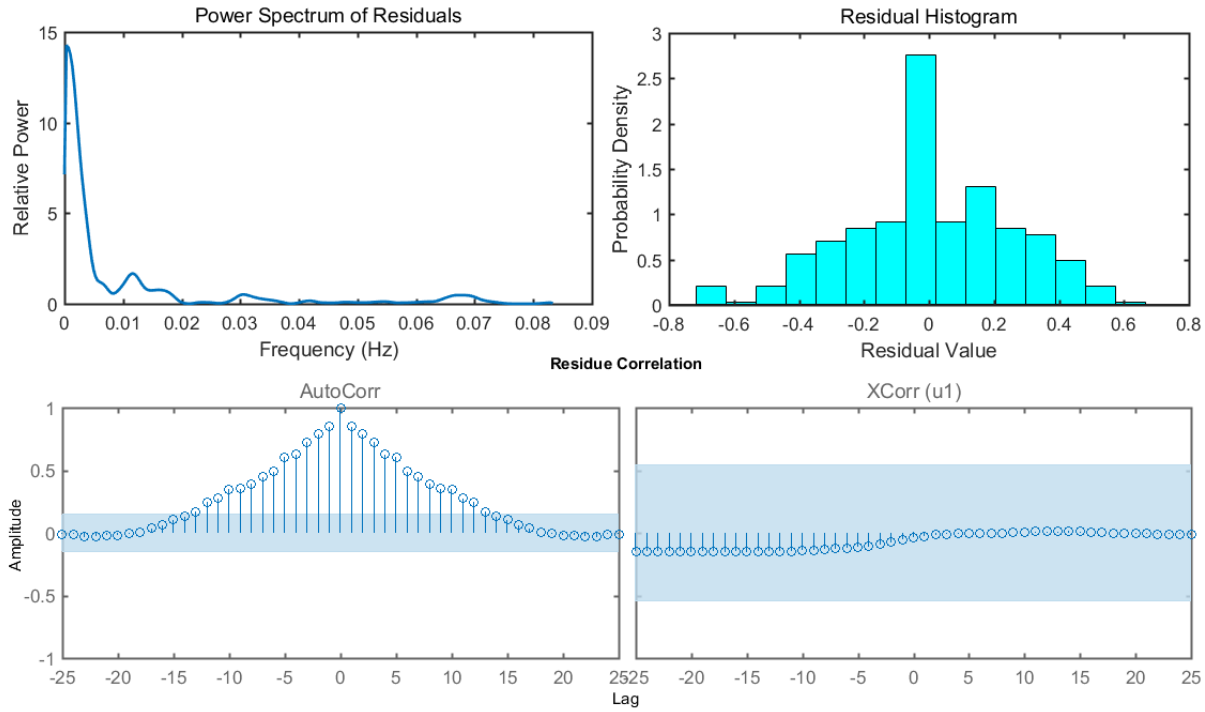

Figure 3.8: **Top Left:** Power Spectrum of Residuals. **Top Right:** Amplitude Distribution of the Residuals. **Bottom:** Correlation Plots for SOPTDLD residuals when fit to Experiment 3886. Left panel shows autocorrelation between residuals and right panel shows cross-correlation between residuals and the input (i.e. the stirrer rate). 99% confidence band is shown by the blue region in both panels.

The model determination for experiments 12429 and 12430 was done in a manner very similar to the model determination for 3885 and 3886. Conveniently, the results of the

Model Type	Parameter Values
1st Order (3.2)	$K = 0.278$, $\tau_1 = 59.6$ s
2nd Order (3.3)	$K = 0.275$, Denominator: $1 + 1.01(57.3s) + (57.3s)^2$
2nd Order, Lead (3.4)	$K = -0.523$, $\tau_1 = 107.9$ s, $\tau_2 = 9236$ s, $\tau_3 = -7522$ s
2nd Order, Derivative, Time Delay (3.5)	$K = 1156$, $\tau_1 = 110.7$ s, $\tau_2 = 2767$ s, $\theta_d = 10$ s
2nd Order, Lead, Time Delay (3.6)	$K = -0.829$, $\tau_1 = 80.7$ s, $\tau_2 = 11960$ s, $\tau_3 = -5721$ s, $\theta_d = 24$ s

Table 3.5: Parameter values obtained from fitting the 5 models to the dataset for experiment 12429. When possible, parameters are listed in a manner consistent with the corresponding model equations.

model fitting were also similar to those found earlier. For example, in tables 3.5 and 3.6, which represent the results of the fitting to the data from experiment 12429, we found that the second-order transfer function with lead and time delay was the best fit model. This result is very similar to that found for batch 3886, and is supported by both the F-tests and the AIC tests.

However, it should be noted that for this best-fit model, the lead (zero) exists on the right half of the complex plane, a finding which will be made use of in a later chapter when we develop internal model controllers for each experimental dataset. The right-half plane zero also has special consequences when determining the model predictive controller for the system. Figures 3.9 and 3.10 show the plots of the model fits against the experimental data. While the simple first-order and second-order transfer functions fail to capture the behaviour of much of the data, the more complex models, which include leads and time delays show significant improvement in the fit quality.

Because the SOPTDLD model (equation 3.6) best describes experiment 12429, we decided to further examine the quality of the model fit via residual analysis. The elements of residual analysis are exactly the same as those of batches 3885 and 3886, and the complete plots are shown in Figure 3.11. The figures shows little cross-correlation between the residuals and the stirrer speed input, in addition to a relatively normal residual distribution centred slightly left of zero. This suggests that the residuals are independent of the

Model Type	AIC Value	SSE	Degrees of Freedom	F-Statistics	P-Value
1st Order (3.2)	2548	5359	483	$F_{FS} = 37.0$	2.44×10^{-9}
2nd Order (3.3)	2516	4978	482	$F_{SSLD} = 14163$	0
2nd Order, Lead (3.4)	861.0	163.5	481	$F_{SLDSLDTD} = 278.6$	0
2nd Order, Derivative, Time Delay (3.5)	811.7	148.3	481	$F_{SLDSTDD} = 1.10$	0.143
2nd Order, Lead, Time Delay (3.6)	639.0	103.5	480	$F_{STDDSLDTD} = 208.1$	0

Table 3.6: Comparing the 5 transfer function models fits to the dataset for experiment 12429. From both the AIC and the p-test, we can conclude that the SOPTDLD model (equation 3.6) is the best description for the process in experiment 12429.

model input and that the SOPTDLD transfer function is sufficient to capture the dissolved oxygen-stirrer rate dependence.

In contrast, the autocorrelation plot showed statistically significant correlation at non-zero lags, while the power spectrum of the residuals showed greater power at low frequencies. From these results, it can be concluded that the residuals fail to demonstrate 'whiteness', the even distribution of power at all possible frequencies. Again, we posit that this is likely due to the disturbance which has not been accounted for in the transfer function models.

Next, the step-response data from experiment 12430 were fit to our 5 models and the results tabulated (Tables 3.7 and 3.8). Here, it was found that the second-order transfer function with time delay and derivative served as the best fit for the experiment, similar to the results for batch 3885. Plotting the outputs of the fits against the experimental data (Figures 3.12 and 3.13) confirmed this finding.

Residual analysis of the best-fit model (i.e. equation 3.5) showed no significant cross-correlation between the input and the residuals, but showed significant autocorrelation within the residuals, likely due to the lack of inclusion of a term which accounted for the disturbances. In addition, the distribution of the residuals was found to be somewhat normal and centred at zero, indicating that the model was capable of adequately describing the relationship between the dissolved oxygen and stirrer rate. Finally, the power spectrum once again showed the presence of low frequency noise, which, as we proposed for the other 3 batches, is likely due to the lack of a disturbance term in the transfer function model.

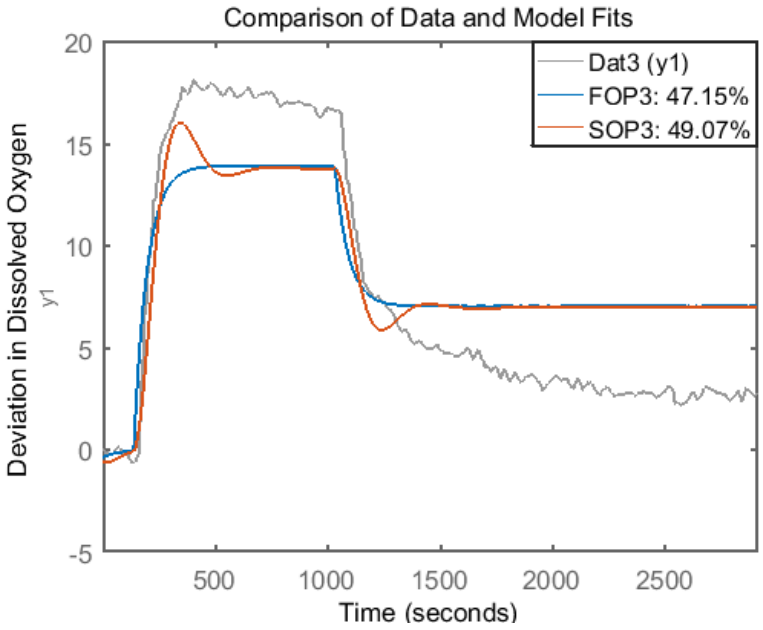

Figure 3.9: Comparison of Experimental Data (grey) to the fits from the FOP and SOP models. Legend shows the quality of the fit in percentage for each model (100% denotes a completely accurate fit).

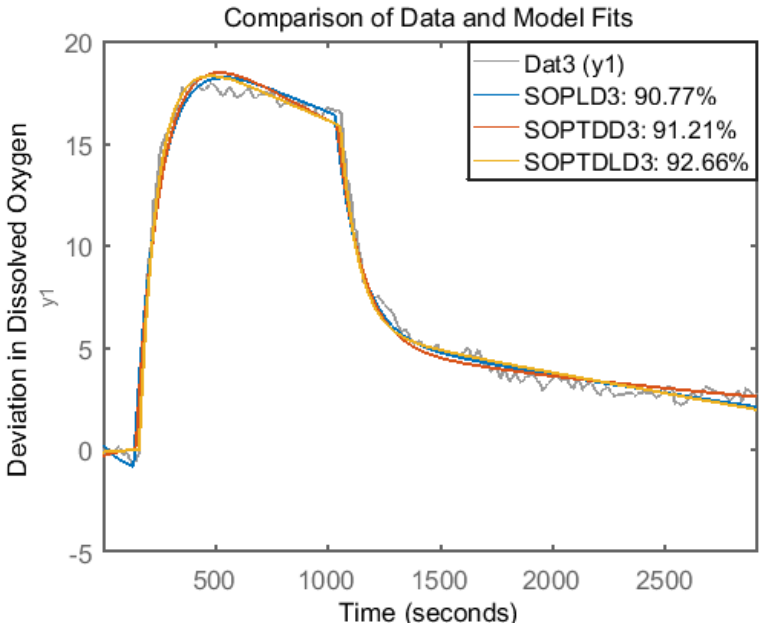

Figure 3.10: Comparison of Experimental Data (grey) to the fits from the SOPLD, SOPTDD, and SOPLDTD models. Legend shows the quality of the fit in percentage for each model (100% denotes a completely accurate fit).

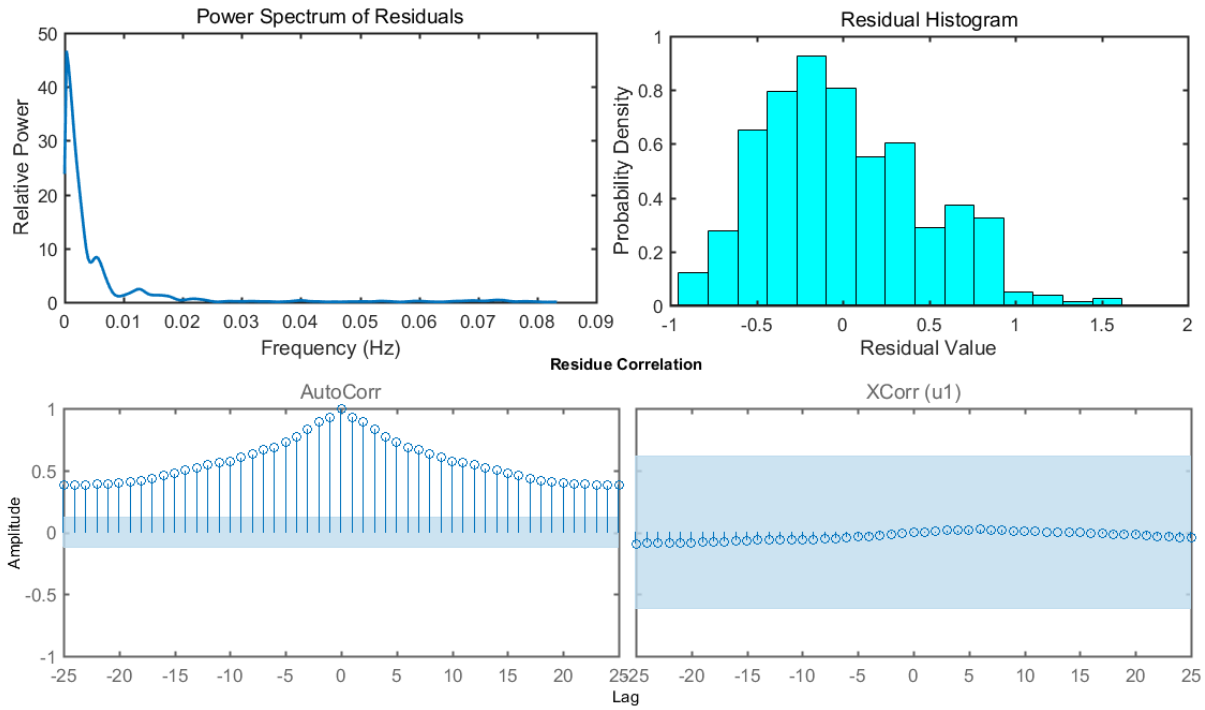

Figure 3.11: **Top Left:** Power Spectrum of Residuals. **Top Right:** Amplitude Distribution of the Residuals. **Bottom:** Correlation Plots for SOPTDLD residuals when fit to Experiment 12429. Left panel shows autocorrelation between residuals and right panel shows cross-correlation between residuals and the input (i.e. the stirrer rate). 99% confidence band is shown by the blue region in both panels.

Model Type	Parameter Values
1st Order (3.2)	$K = 0.282$, $\tau_1 = 69.6$ s
2nd Order (3.3)	$K = 0.279$, Denominator: $1 + 1.17(57.5s) + (57.5s)^2$
2nd Order, Lead (3.4)	$K = -0.537$, $\tau_1 = 103.0$ s, $\tau_2 = 15280$ s, $\tau_3 = -10870$ s
2nd Order, Derivative, Time Delay (3.5)	$K = 2130$, $\tau_1 = 97.6$ s, $\tau_2 = 5461$ s, $\theta_d = 7$ s
2nd Order, Lead, Time Delay (3.6)	$K = -0.827$, $\tau_1 = 80.7$ s, $\tau_2 = 11940$ s, $\tau_3 = -5722$ s, $\theta_d = 24$ s

Table 3.7: Parameter values obtained from fitting the 5 models to the dataset for experiment 12430. When possible, parameters are listed in a manner consistent with the corresponding model equations.

Model Type	AIC Value	SSE	Degrees of Freedom	F-Statistics	P-Value
1st Order (3.2)	2330	2570	523	$F_{FS} = 66.89$	2.22×10^{-15}
2nd Order (3.3)	2270	2278	522	$F_{SSLD} = 8817$	0
2nd Order, Lead (3.4)	757.2	127.1	521	$F_{SLDSLDTD} = -438.3$	1
2nd Order, Derivative, Time Delay (3.5)	622.5	21.23	521	$F_{SLDSTDD} = 1.29$	0.002
2nd Order, Lead, Time Delay (3.6)	639.0	809.0	520	$F_{STDDSLDTD} = -456.5$	1

Table 3.8: Comparing the 5 transfer function models fits to the dataset for experiment 12430. From both the AIC and the p-test, we can conclude that the SOPTDD model (equation 3.5) is the best description for the process in experiment 12430.

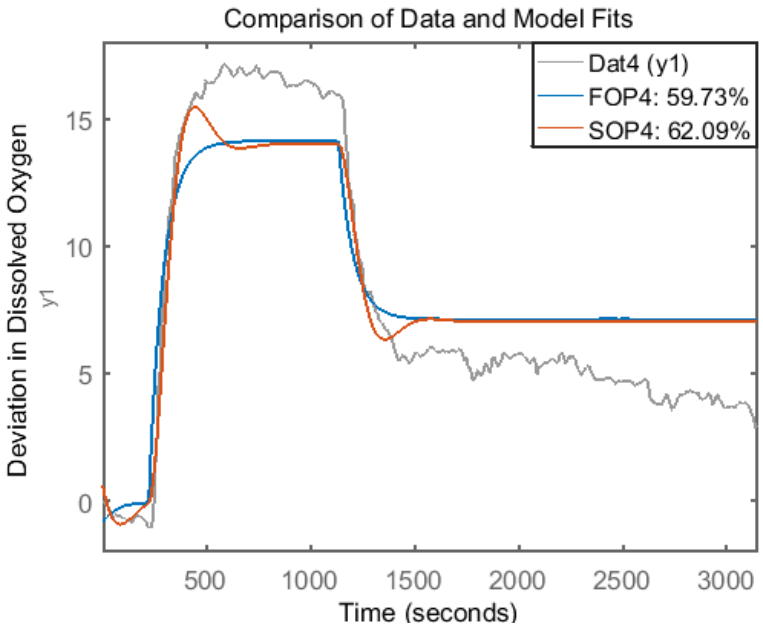

Figure 3.12: Comparison of Experimental Data (grey) to the fits from the FOP and SOP models. Legend shows the quality of the fit in percentage for each model (100% denotes a completely accurate fit).

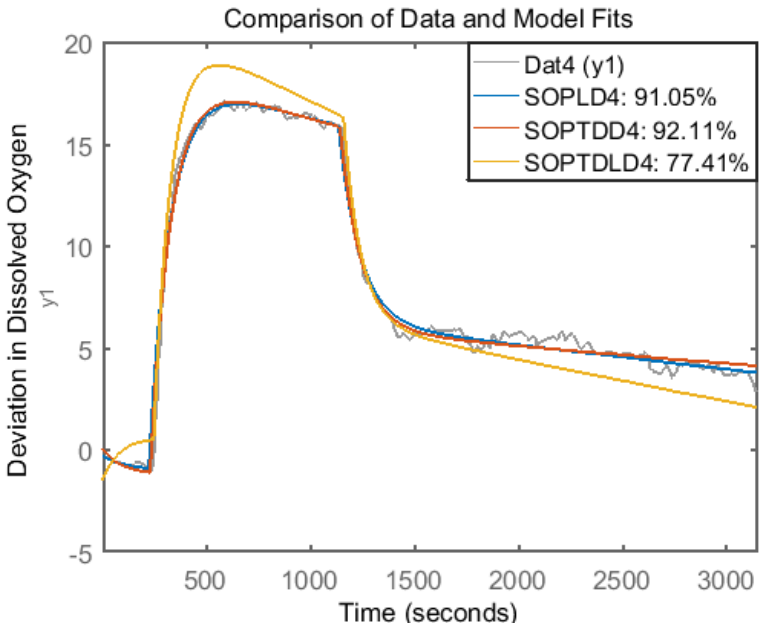

Figure 3.13: Comparison of Experimental Data (grey) to the fits from the SOPLD, SOPTDD, and SOPLDTD models. Legend shows the quality of the fit in percentage for each model (100% denotes a completely accurate fit).

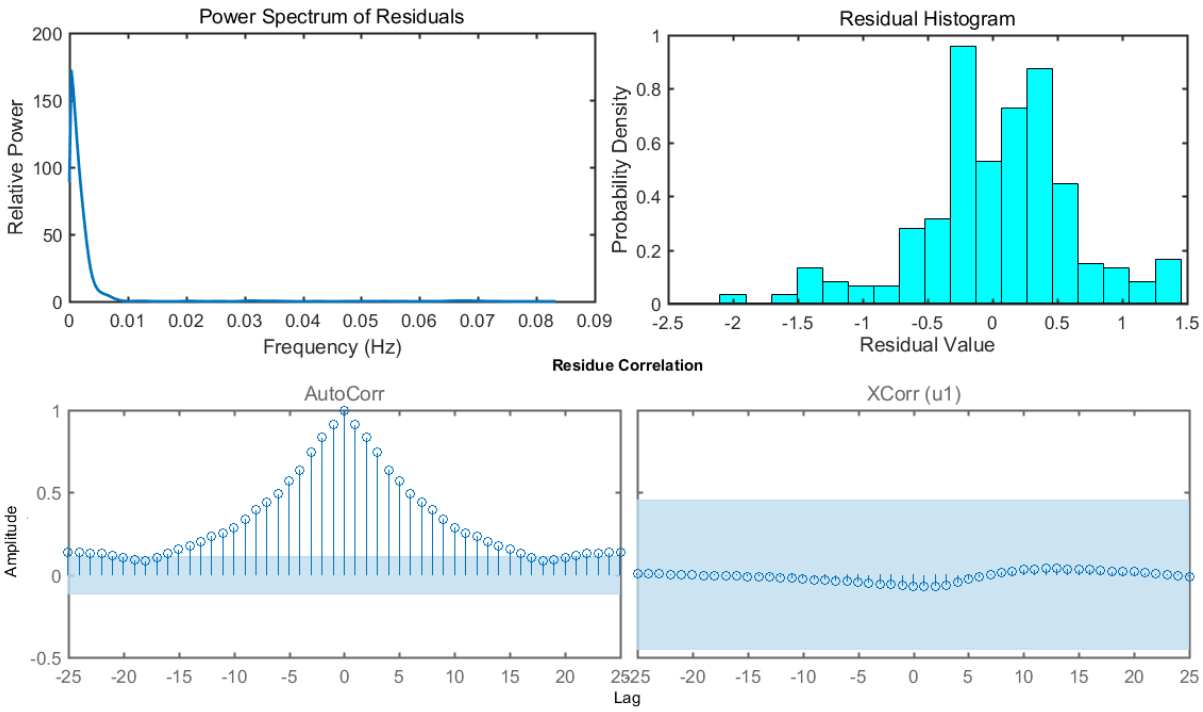

Figure 3.14: **Top Left:** Power Spectrum of Residuals. **Top Right:** Amplitude Distribution of the Residuals. **Bottom:** Correlation Plots for SOPTDD residuals when fit to Experiment 12430. Left panel shows autocorrelation between residuals and right panel shows cross-correlation between residuals and the input (i.e. the stirrer rate). 99% confidence band is shown by the blue region in both panels.

Chapter 4

Optimal PID Control

4.1 Optimization of set-point tracking

PID controllers are among the most popular and frequently used controllers in industry. Their simplicity and ease of implementation make them ideal candidates for serving as control systems for most industrial processes, including the microbial fermentation process examined here. A host of tuning rules associated with PID controllers exist. These include the Ziegler-Nichols Method and the Cohen-Coon Method [36, 37]. However, due to the complexity of the best-fit models found in the previous chapter as well as their batch-to-batch variability, it is necessary to develop a more systematic tuning method which can easily be implemented using computational software such as MATLAB.

Here, we will develop a PID tuning method based on a nonlinear least-squares optimization technique derived from Madhuranthakam et al's PID tuning method [21]. Four block diagrams are constructed in Simulink - one for each batch. Every block diagram contains the same PID controller followed by a twenty-fold gain block and a saturation block (the latter two blocks represent the final control element G_v mentioned in chapter 2). The saturation block contains the upper and lower limits of the stirrer rate at 100 and 675 rpm, but these limits are re-centred according to the initial stirrer speed at which the step response experiments were conducted.

This re-centring is necessary because the process transfer functions G_p were estimated by subtracting the initial values of the stirrer rate and dissolved oxygen so that the system starts at (0,0). As a result, the limits on the stirrer rate have to be made relative to the

initial stirrer rate for the particular experimental batch. For example, the initial stirrer rates for 3885 and 3886 are around 400 rpm, while the initial stirrer rates for 12429 and 12430 are around 450 rpm, meaning the limits for these two sets of batches are [-300,275] and [-350,225] respectively.

Following the specification of the final control element, G_p is then plugged into the Simulink block diagram for its corresponding batch. In addition, a step change in the set-point y_{sp} of dissolved oxygen is specified. The set-point change includes a step in the relative dissolved oxygen from 0 to 10% at $t = 0$, followed by a step down from 10 to -5% at $t = 1000$, where the dissolved oxygen is held until $t = 2000$ seconds.

After these initial arrangements are made, we end up with 4 Simulink block diagrams representing each of the 4 experimental batches sharing the exact same PID controller, with each diagram containing the same step stimulus to which the control system must respond. The algorithm is then run using an initial guess for the PID settings (i.e. K_c, T_i and T_d) obtained from approximate manual tuning done using the PID tuner app in MATLAB. PID settings for future iterates are then estimated by a nonlinear least-squares strategy which seeks to find the optimal values for K_c, T_i and T_d in equation 2.1 such that the following cost function is minimized:

$$E_{IA} = \int_0^{T_s} |e| dt \tag{4.1}$$

where E_{IA} is the integral absolute error, e is the error $(y_{sp} - y)$ between the set-point and the actual dissolved oxygen value, and T_s is the simulation time (2000 seconds). For the fermentation models determined in chapter 3, the initial guess used was $K_c = 0.3625$, $T_i = 92.95$s, and $T_d = 3.501$ s. This initial guess was obtained by using the PID Tuner application in MATLAB to arrive at rough PID settings that would optimize the control of dissolved oxygen for the process model of batch 3885.

When the responses of all 4 control systems was computed for the initial PID parameters, the E_{IA}'s for the individual systems were tallied and added. The total E_{IA} for all 4 block diagrams/experimental batches was then minimized by a nonlinear least-squares optimization technique by varying the PID parameters and running the control systems again with the new PID parameters. If the new PID parameters provided a smaller total E_{IA}, then those settings were accepted and the process was continued. After the maximum number of iterations had been reached, or after the tolerance criteria for E_{IA} were met, the simulation stopped and the PID parameters used in the last run were outputted and

the results plotted.

Running our optimization algorithm multiple times, with the final value from one run being used as the initial guess for the other run gave us a set of optimal PID parameters: $K_p = 0.3226$, $T_i = 91.0$s, and $T_d = 4.3$ s. Figure 4.1 shows the response of the dissolved oxygen to the step change stimulus described above for the optimized PID parameters for all 4 experimental batch models. As seen from the figure, the PID parameters do quite well in allowing the dissolving oxygen to reach the set-point quickly. The overshoot (i.e. the rise in the dO_2 beyond the set point) is kept relatively low and the dO_2 stays relatively close to the desired set-point.

Compared to the previously used settings of $K_p = 0.3333$, $T_i = 150$s, and $T_d = 0$ s (see Figure 4.2), the new settings do a far better job of allowing the system to reach the desired level of dissolved oxygen. The settings found allow the system to settle much more quickly and stay relatively close to the set-point throughout the run. We expect these new settings to solve the problem of the DO oscillations observed experimentally, since a potential reason for the occurrence of the oscillations is that the controller is incapable of adjusting to changes in DO, which causes it to respond insufficiently and furthers the degree and extent of the oscillations.

After testing this hypothesis experimentally, we found that indeed, the new PID settings solve the problem of uncontrollable oscillations in dissolved oxygen. Figures 4.3,4.4, and 4.5 show an experimental run in which the old PID settings of $K_p = 0.3333$, $T_i = 150$s, and $T_d = 0$ s were initially used. As seen from the figures, the original settings were incapable of adequately regulating the dissolved oxygen, resulting in large, uncontrolled oscillations.

However, as soon as the new settings calculated from our PID optimization algorithm were implemented, the dissolved oxygen settled to its desired set point of 35%. This is likely due to the introduction of a (albeit small) derivative term and the reduction of the integral time to 91 seconds. Because of these changes, the system now responds much more quickly than before, and is able to achieve control much more easily. From these experimental runs, we have demonstrated that our PID optimization algorithm is capable of arriving at settings which allow the system to better achieve control.

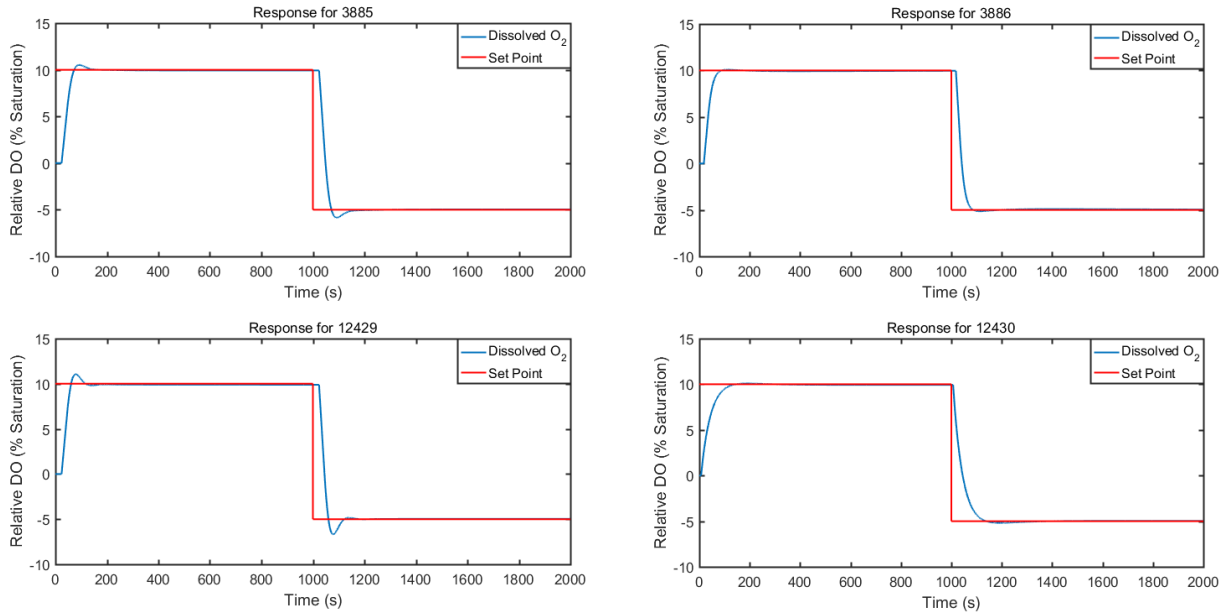

Figure 4.1: Response of all 4 experimental batch models to the step change stimulus shown in red. Optimized PID settings are used for control.

4.2 Optimization of set-point tracking and disturbance rejection

The optimization algorithm mentioned and tested in the previous section is based primarily on ensuring that the dO_2 closely tracks the set-point. However, of equal, or perhaps even greater importance, is the ability of the control system to reduce the effect of a disturbance to the system. Disturbances may occur due to external noise or an interruption in the process, and a well-adjusted control system ought to have countermeasures in place to ensure that disturbances have minimal effect on the controlled variable (dO_2).

Because of the importance of disturbance rejection in ensuring the maintenance of proper control, a potential variation on the optimization algorithm is to now include disturbance rejection as a criterion with respect to which the PID settings ought to be optimized. The only difference in the block diagrams between the disturbance rejection criterion and the set-point tracking criterion is that in the former, the set-point is now held constant at zero but there is a step in the stirrer rate of 20 rpm just before the output at $t = 0$. The added disturbance to the stirrer rate is held at the +20 rpm elevated value until $t = 2000$.

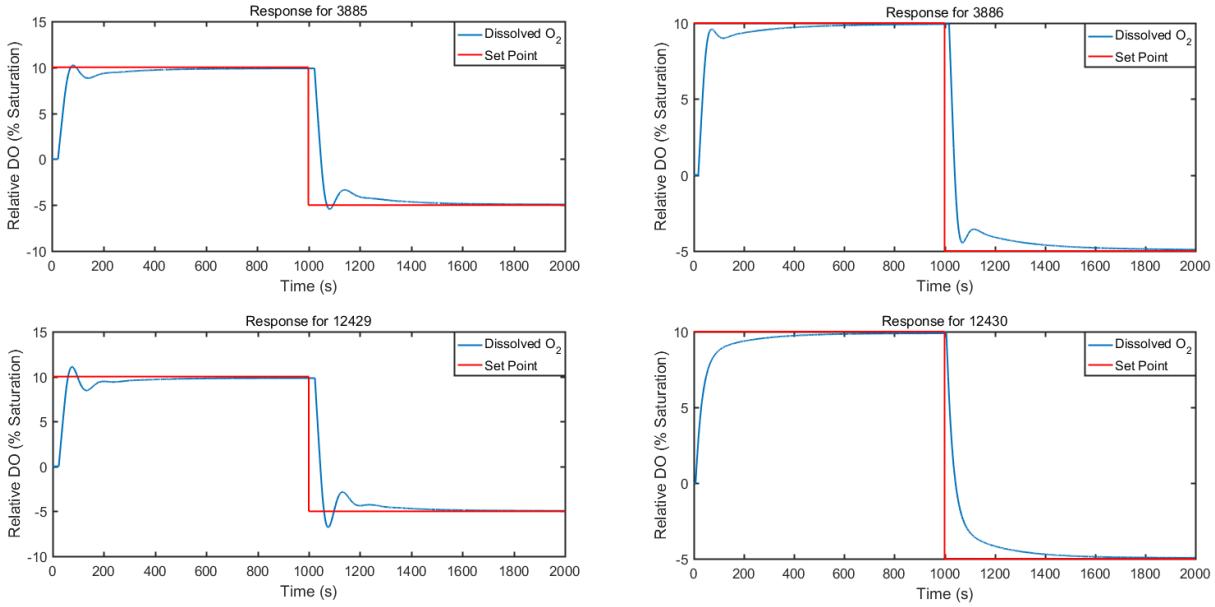

Figure 4.2: Response of all 4 experimental batch models to the step change stimulus shown in red, but now, the industrial settings are used.

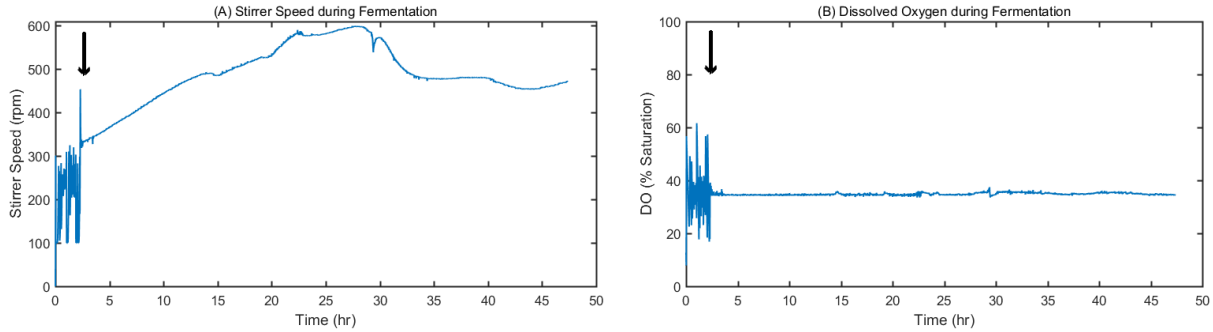

Figure 4.3: First experimental run of the biological fermenter, starting with the old settings of $K_p = 0.3333$, $T_i = 150$s, and $T_d = 0$ s. The black arrows indicate the point at which the settings were switched to $K_p = 0.3226$, $T_i = 91.0$s, and $T_d = 4.3$ s.

When disturbance rejection is added to set-point tracking as another optimization criterion, then the resulting settings of $K_p = 0.3228$, $T_i = 83.0$s, and $T_d = 4.3$ s are very similar to those found from using only set-point tracking as the optimization criterion. Note that here, the initial guess used was $K_p = 0.3226$, $T_i = 91.0$s, and $T_d = 4.3$ s - the

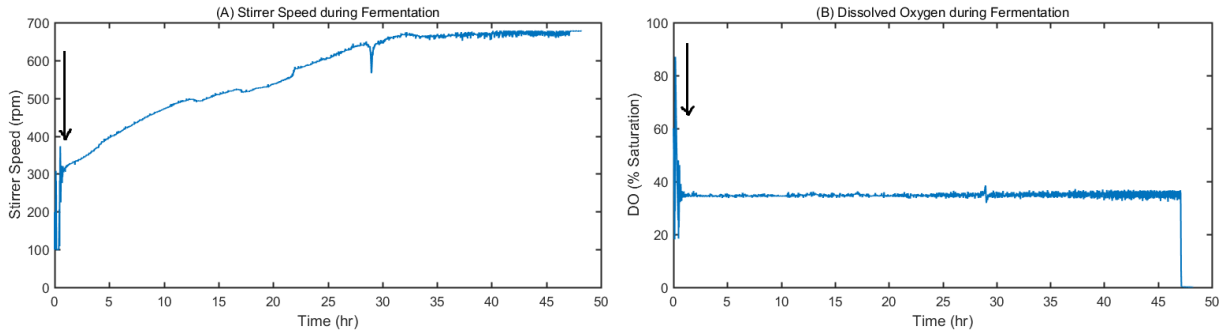

Figure 4.4: Second experimental run of the biological fermenter, starting with the old settings of $K_p = 0.3333$, $T_i = 150$s, and $T_d = 0$ s. The black arrows indicate the point at which the settings were switched to $K_p = 0.3226$, $T_i = 91.0$s, and $T_d = 4.3$ s.

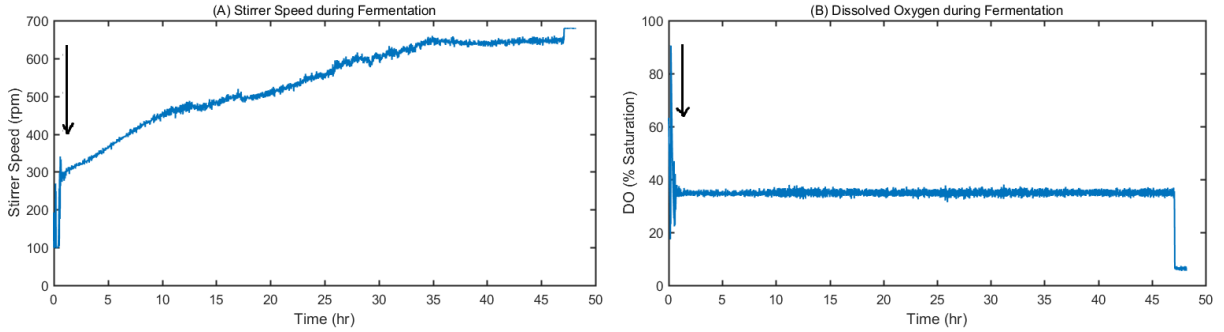

Figure 4.5: Third experimental run of the biological fermenter, starting with the old settings of $K_p = 0.3333$, $T_i = 150$s, and $T_d = 0$ s. The black arrows indicate the point at which the settings were switched to $K_p = 0.3226$, $T_i = 91.0$s, and $T_d = 4.3$ s.

settings found from the previous optimization. Because the settings between the two sets of optimization are very similar, we hypothesise that the PID settings we have are equal, or very close, to the true optimal settings for the microbial fermenter. This hypothesis is further corroborated by the success of the experiments shown in figures 4.3, 4.4, and 4.5.

Figure 4.6 shows the results of the set-point tracking simulation using the settings of $K_p = 0.3228$, $T_i = 83.0$s, and $T_d = 4.3$ s found from optimizing with respect to both disturbance rejection and set-point tracking. Comparing the optimized settings to the original industrial settings shows that the former do a far more superior job at tracking the set-point (even though the overshoot is slightly larger) than the industrial settings. In addition, figure 4.7 compares the optimized settings to the original industrial settings using

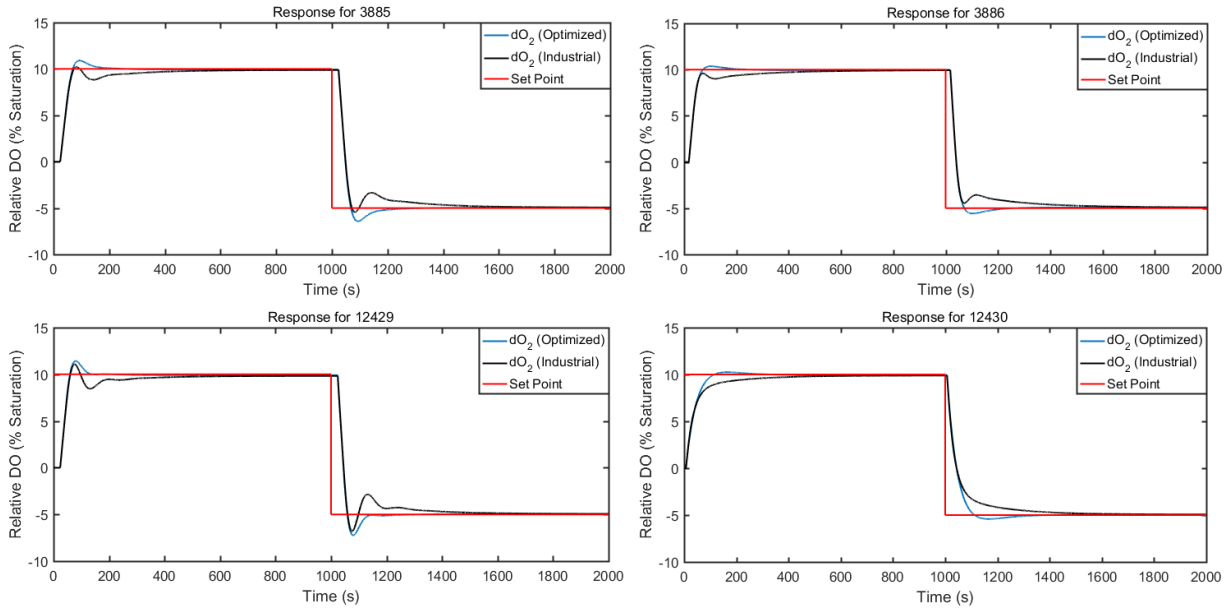

Figure 4.6: Response of all 4 experimental batch models to the step change stimulus shown in red. Both optimized PID settings (blue) and the original industrial settings (black) are used for control.

the input disturbance simulation. The figure demonstrates that not only do the optimized settings allow the system to settle nearly twice as quickly as the industrial settings, but also that the maximum effect of the input disturbance on the dissolved oxygen is also significantly less for the optimized settings.

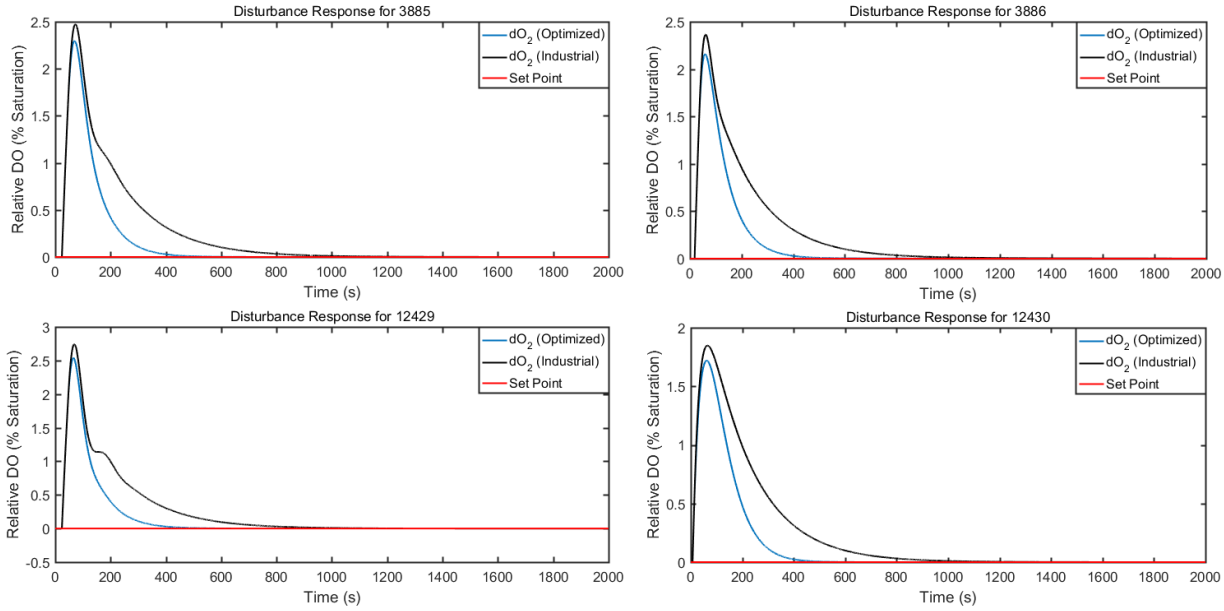

Figure 4.7: Response of all 4 experimental batch models to an input step disturbance of 20 rpm. Both optimized PID settings (blue) and the original industrial settings (black) are used for control.

Chapter 5

Internal Model Controllers

5.1 Deriving Internal Model Controllers

Internal Model Control (IMC) is a control strategy which involves determining an analytical expression for the controller transfer function in terms of the process model [38]. It is closely related to direct synthesis, which provides a controller such that the closed loop transfer function of the feedback loop (i.e. Y/Y_{sp}) obeys a user-specified relationship.

In fact, IMC provides the exact same controller that the direct synthesis method provides, as long as the user-specified relationships between Y and Y_{sp} and the specification of design parameters are consistent. However, IMC is based on a slightly re-formulated version of the typical feedback control loop, and is shown in Figure 5.1. With internal model control, the controller output P is applied to both the real process G_p (including disturbances) and the model of the process $\tilde{G}_p$. The outputs of both are subtracted, yielding $Y - \tilde{Y}$, which is then used to form the input signal for the controller G_c^*.

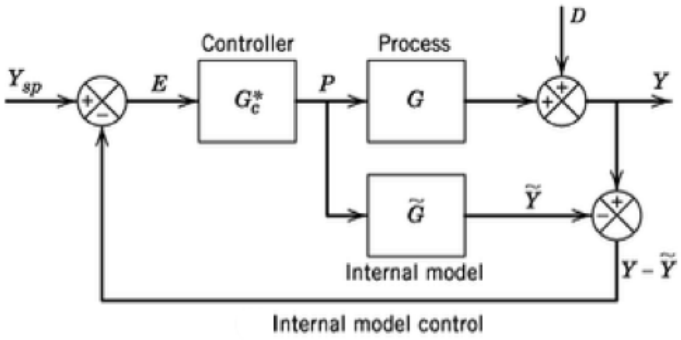

Figure 5.1: Block Diagram of an internal model control structure. Figure used from [13].

When proposing a new controller or control structure, three important criteria necessary for physical realisability must be taken into consideration:

1. **Stability:** For an IMC setup, it can be shown that if the process model $\tilde{G}_p$ is stable and sufficiently close to the actual process G_p, then the feedback control system is stable if and only if the IMC controller G_c^* is stable. Given that the process models $\tilde{G}_p$ found in chapter 3 are all stable, we only require the IMC controller formulated to be stable for overall process stability [13].

2. **Proper:** A physically realisable IMC controller must be proper. If the IMC controller $G_c^*(s)$ is described by a rational expression, this means that the order of its numerator must be less than or equal to the order of its denominator. This ensures that the IMC does not perform strict differentiation, which may be problematic because fast step-changes in the controlled variable or set-point can bring about large impulse changes in the manipulated variable u. Mathematically, a proper IMC controller is such that:
$$\lim_{s\to\infty} |G_c^*(s)| \tag{5.1}$$
is finite.

3. **Causality:** A physically realisable IMC controller must be causal. That is, it should respond to changes in the controlled variable/set-point from the past, and not anticipate changes in the future. A 'future-predicting' IMC controller is impossible to achieve [39].

Designing an IMC controller involves a sequence of algebraic operations. First, the process model $\tilde{G}$ is decomposed into two components, $\tilde{G}_{p+}$ and $\tilde{G}_{p-}$, such that $\tilde{G} = \tilde{G}_{p+}\tilde{G}_{p-}$.

$\tilde{G}_{p+}$ contains terms which can bring about instability and physically unrealisable terms in the controller G_c^*. These terms are also known as the non-minimum phase elements, and include right-half plane zeros and time delays.

On the other hand, $\tilde{G}_{p-}$ contains everything else from the process, and is therefore a minimum phase component and invertible [40]. The separation of the process model into minimum phase and non-minimum phase components is required to ensure that the resulting controller is stable and proper. It should be noted that $\tilde{G}_{p+}$ is formulated such that its steady-state gain is 1 (i.e. $\tilde{G}_{p+}(s=0)=1$ from the final value theorem), while any non-unitary gains are placed into the 'stable', minimum phase component $\tilde{G}_{p-}$. Once $\tilde{G}_{p+}$ and $\tilde{G}_{p-}$ are specified, the IMC controller can be analytically formulated using the following relation:

$$G_c^* = \frac{1}{\tilde{G}_{p-}} f \tag{5.2}$$

where f is the IMC filter. It is generally specified as a low pass filter of the form $f = 1/(\tau_c s + 1)^r$, where τ_c is the desired closed-loop time constant and r is the filter coefficient. The specification of only $\tilde{G}_{p-}$ ensures stability and causality, since $\tilde{G}_{p-}$ does not include right-half plane zeros and time delays, which would appear as unstable poles (a stability problem) and anticipatory steps (a causality problem) in G_c^*.

Moreover, the addition of an IMC filter ensures that the controller remains proper. For the remainder of our discussion, we will set r to unity, making one notable exception in the process model for batch 12429, which contains a right-half plane zero. It is possible to convert directly from an internal model controller to a feedback controller G_c shown in Figure 2.2 using the expression below:

$$G_c = \frac{G_c^*}{1 - G_c^* \tilde{G}} \tag{5.3}$$

When G_c^* or G_c are specified according to the IMC formulation, we can show that the overall closed-loop transfer function Y/Y_{sp} becomes:

$$\frac{Y}{Y_{sp}} = \tilde{G}_+ f \tag{5.4}$$

Therefore, if we use the direct synthesis method and specify our desired closed-loop transfer function between Y and Y_{sp} to be given by equation 5.4, we will obtain the exact same

expression for our feedback controller as that shown in equation 5.3. This is why direct synthesis and internal model control are, at some level, equivalent.

From our analysis in the previous section, we determined 3 distinct models that were capable of describing the behaviour of the experimental batches. For batches 3885 and 12430, the SOPTDD model (equation 3.5) was found to be the best option according to both the F-test and the Akaike Information Criterion (AIC). Using this model, only the time delay was excluded from $\tilde{G}_{p-}$. As a result, the expression for the internal model controller was found to be:

$$G_c^* = \frac{(\tau_1 s + 1)(\tau_2 s + 1)}{K s(\tau_c s + 1)} \tag{5.5}$$

where τ_c is our filter time constant and $r = 1$ is the filter coefficient. Using equation 5.3, it is possible to determine the expression for the feedback controller using some simple algebra. It can be shown that:

$$G_c = \frac{(\tau_1 s + 1)(\tau_2 s + 1)}{K s\left[(\tau_c s + 1) - e^{-\theta_d s}\right]} \tag{5.6}$$

Specifying a controller transfer function in terms of a non-polynomial term (i.e. the exponential in the denominator above) creates computational difficulty. Thus, to simplify matters, we use an approximation to the exponential known as the Pade approximation [42]:

$$e^{-\theta_d s} \approx \frac{1 - 0.5\theta_d s}{1 + 0.5\theta_d s} \tag{5.7}$$

Substituting the Pade approximation into 5.6 and simplifying, we get:

$$G_c = \frac{(\tau_1 s + 1)(\tau_2 s + 1)(0.5\theta_d s + 1)}{K(\tau_c + \theta_d)s^2\left[\frac{\theta_d \tau_c}{2(\tau_c + \theta_d)} s + 1\right]} \tag{5.8}$$

as the final expression for our feedback controller for the SOPTDD models that describe experiments 3885 and 12430. The second type of process model we found was from experiment 3886, where we determined that a second-order model with time delay and lead (equation 3.6) was the best descriptor for the experimental results of the system. With

this model, we found that our IMC controller was given by:

$$G_c^* = \frac{(\tau_1 s + 1)(\tau_2 s + 1)}{K(\tau_3 s + 1)(\tau_c s + 1)} \tag{5.9}$$

where we have once again set $r = 1$. From this, the expression for the corresponding feedback controller is:

$$G_c = \frac{(\tau_1 s + 1)(\tau_2 s + 1)}{K(\tau_3 s + 1)\left[(\tau_c s + 1) - e^{-\theta_d s}\right]} \tag{5.10}$$

Applying the Pade approximation, the equation for the feedback controller becomes:

$$G_c = \frac{(\tau_1 s + 1)(\tau_2 s + 1)(0.5\theta_d s + 1)}{K(\tau_c + \theta_d)s(\tau_3 s + 1)\left[\frac{\theta_d \tau_c}{2(\tau_c + \theta_d)} s + 1\right]} \tag{5.11}$$

The third type of process model, used to describe the results of experiment 12429, is also a second-order model with time delay and lead (equation 3.6) except now, the lead is negative (i.e. the zero occupies the right-half plane). In this instance, the $\tilde{G}_+$ factor in the process model becomes:

$$\tilde{G}_{p+} = (\tau_3 s - 1)e^{-\theta_d s} \tag{5.12}$$

As a result, the expression for the internal model controller is:

$$G_c^* = \frac{(\tau_1 s + 1)(\tau_2 s + 1)}{K(\tau_c s + 1)^2} \tag{5.13}$$

where we have now used $r = 2$ as the filter coefficient. The use of $r = 2$ is made so that the controller remains proper and does not have a pure derivative term. This means that the expression for the equivalent feedback controller is:

$$G_c = \frac{(\tau_1 s + 1)(\tau_2 s + 1)}{K\left[(\tau_c s + 1)^2 - (\tau_3 s - 1)e^{-\theta_d s}\right]} \tag{5.14}$$

Finally, after employing the Pade approximation and simplifying, the feedback controller

using the IMC formulation for experiment 12429 is given by the following transfer function:

$$G_c = \frac{(\tau_1 s + 1)(\tau_2 s + 1)(0.5\theta_d s + 1)}{K\left[\frac{1}{2}\tau_c^2\theta_d s^3 + (\tau_c^2 + \tau_c\theta_d + \frac{1}{2}\tau_3\theta_d)s^2 + (2\tau_c - \tau_3)s + 2\right]} \tag{5.15}$$

Alternatively, we may also employ an improper IMC for experiment 12429, and in fact, we will find later that the results from the improper IMC are qualitatively similar to those of the proper IMC. In this case, a smaller filter coefficient of $r = 1$ may be used, resulting in an internal model feedback controller given by:

$$G_c = \frac{(\tau_1 s + 1)(\tau_2 s + 1)(0.5\theta_d s + 1)}{K\left[\frac{\theta_d}{2}(\tau_c + \tau_3)s^2 + (\tau_c - \tau_3)s + 2\right]} \tag{5.16}$$

after applying the Pade approximation.

5.2 Testing the IMC Controllers for a select few τ_c

The derived expressions for the IMC Controllers found in the previous section and the model fitting results from Tables 3.1, 3.3, 3.5, and 3.7, allow us to substitute for the numerical parameters contained in the controller transfer functions for each batch. After substitution, we obtain the equations below, which show the transfer functions for all the IMC controllers. Note that the subscript 1 denotes the controller corresponding to batch 3885, 2 denotes 3886, 3 denotes 12429, 3-Im denotes the improper form of the 12429 controller, and 4 denotes 12430:

$$G_{c1} = \frac{5838701s^3 + 555457.7s^2 + 5754s + 1}{24394\tau_c s^3 + (48788 + 2033\tau_c)s^2} \tag{5.17}$$

$$G_{c2} = \frac{452506s^3 + 56162s^2 + 662.8s + 1}{2107\tau_c s^3 + (4214 + 237\tau_c)s^2 + (5.86 + 0.326\tau_c)s} \tag{5.18}$$

$$G_{c3} = \frac{11582715s^3 + 1109719s^2 + 12053s + 1}{9.95\tau_c^2 s^3 + (56892 + 19.9\tau_c + 0.829\tau_c^2)s^2 + (1.66\tau_c - 4741)s + 1.66} \tag{5.19}$$

$$G_{c3-Im} = \frac{11582715s^3 + 1109719s^2 + 12053s + 1}{(56892 + 9.95\tau_c)s^2 + (0.829\tau_c - 4741)s + 1.66} \tag{5.20}$$

$$G_{c4} = \frac{1866563s^3 + 552611s^2 + 5562s + 1}{7458\tau_c s^3 + (14916 + 2130\tau_c)s^2} \tag{5.21}$$

Fortunately, the quantitative forms of the IMC Controllers chiefly depend on the parameters estimated from process model fitting. Because these process model parameters are known *a priori*, they can be treated as fixed for an IMC Controller corresponding to a particular batch. The only variable parameter for the purpose of IMC Tuning is the closed-loop time constant τ_c. Therefore, for much of the remaining chapter, we will focus on simulating the IMC controllers for the 4 batches under multiple process conditions with different values of τ_c to assess the quality of the control and eventually determine the optimal τ_c for the multiple process conditions.

Before undertaking extensive analysis to find optimal values of τ_c, we performed a few runs using select values of τ_c to gain a better understanding of the five controllers in equations 5.17 to 5.21. For G_{c1}, G_{c2}, and G_{c4}, four values of τ_c were initially used: 10, 50, 100, and 500. The responses of all the controllers to the same step change in the dissolved oxygen set point (i.e. a step change from 0 to 10 at $t = 0$ and then a step change down to -5 at $t = 1000$ s until $t = 2000$ s) were computed via Simulink, using the same process models that the controllers were originally derived from (e.g. 3885 process model used for G_{c1} etc).

Figure 5.2 shows the response of G_{c1} to the set-point changes in dissolved oxygen for all 4 filter time constants. Here, small values of τ_c result in stable control but with an overshoot, which is largest when $\tau_c = 10$. In addition to the overshoot, the stirrer rate rapidly hits its extreme values corresponding to the specified bounds of 100 and 675 rpm at low τ_c. Practically, fast changes in the stirrer rate due to a small τ_c can potentially strain the mechanical apparatus and are undesirable.

For larger values of τ_c, there is no noticeable overshoot and the DO converges neatly to the set point. The speed of this convergence decreases for larger τ_c. For instance, $\tau_c = 500$ gives a slow controller response that does not even reach the set point within the time of the simulation; however, $\tau_c = 50$ gives a reasonably quick response that minimizes the overshoot and mechanical strain at the same time.

This aforementioned pattern is repeated in Figure 5.3, which shows the response of the IMC for batch 3886 to step changes in the DO set point. Once again, $\tau_c = 50$ was capable of quickly responding to set-point changes without exhibiting any significant overshoot. Figure 5.6 (corresponding to batch 12430), however, was slightly different. Here, even $\tau_c = 10$ did not exhibit an overshoot.

While G_{c1}, G_{c2}, and G_{c4} generally exhibit stable, well-behaved dynamics as internal model controllers for positive τ_c, the G_{c3} and G_{c3-Im} are largely incapable of catching up to the set-point in a manner that would be expected by a typical IMC Controller. This is due to the denominators of the transfer functions from equation 5.19 and 5.20, in which it is possible for the coefficients of s to be negative provided that τ_c is sufficiently small.

When this is the case, the controller exhibits unstable poles, which can lead to poor quality control, because even though the poles of the controller ought to cancel with the zeros of the process model, numerical approximations can lead to imprecision. Thus, even if the control system does not fully exhibit instability, its performance is significantly hindered, effectively rendering the controller useless. Due to the potential instability inherent in the IMCs for 12429, an instability whose effect is augmented for low τ_c and low r, it was necessary to compare between very large values of τ_c for the initial analysis.

For example, when simulating the IMC given by equation 5.19, the values of τ_c used in the simulation were 1000, 3000, 5000, and 7000 seconds. In all four cases, the system fails to reach the set-point and responds very slowly to changes in y_{sp} (Figure 5.4). There is even a massive overshoot for $\tau_c = 1000$ s of around 90% relative to the dO_2 set-point of 10. G_{c3-Im} also exhibited very similar behaviour. For G_{c3-Im}, we selected $\tau_c = 10000$, 30000, 50000, and 70000 as our 4 values for comparison. Generally, all 4 values of τ_c exhibit poor, slow control, in which the controller is incapable of even reaching the desired set points. While $\tau_c = 10000$ was the most effective filter time, it was not adequate enough to respond to changes in the set point, as seen in Figure 5.5.

The results from the performance of G_{c3} and G_{c3-Im} show that in this case, using a proper transfer function helps the control perform better to a limited extent - by curbing the rapid variations in stirrer rate and allowing smaller τ_c to be used. However, it does not drastically improve the set-point tracking of the controller, since that may inherently be limited by the unfavourable dynamics of the process. In conclusion, while it was possible to find good filter time constants for G_{c1}, G_{c2}, and G_{c4}, G_{c3} and G_{c3-Im} presented numerous challenges due to the presence of a potentially negative coefficients which resulted in numerical issues leading to poor control. This necessitated the use of large filter time constants which were simply incapable of providing acceptable control performance.

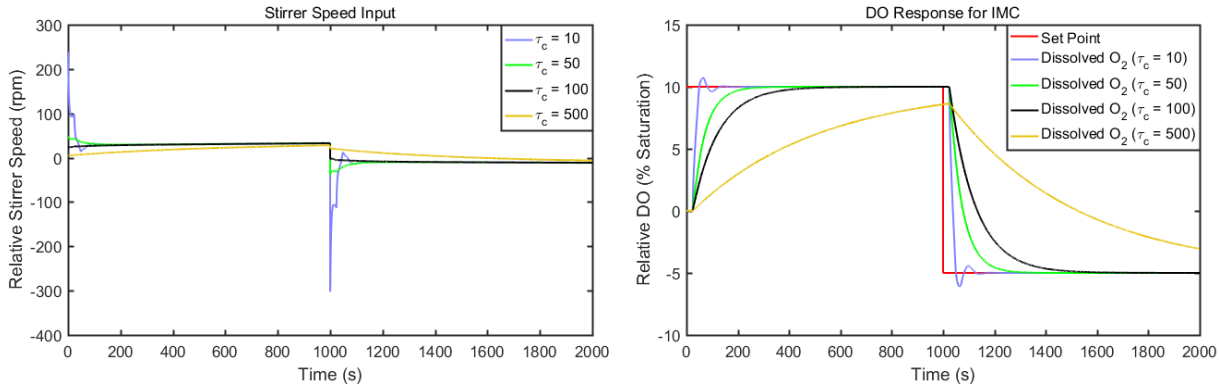

Figure 5.2: Response of the IMC controller to step changes in the dissolved oxygen set-point for Batch 3885.

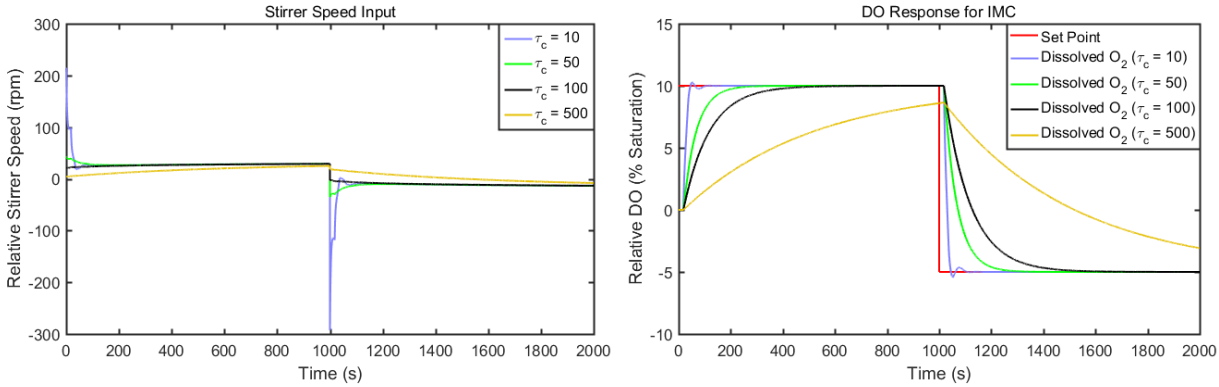

Figure 5.3: Response of the IMC controller to step changes in the dissolved oxygen set-point for Batch 3886.

5.3 Testing IMC Performance in response to variations in τ_c

The simplicity of the IMC is due to the presence of a single variable tuning parameter τ_c, whereas every other parameter in the expression for the controller is fixed and depends only on the estimated plant parameters. This means that it is possible to tune the IMC controllers for each of the four batches by varying the lone tuning parameter τ_c. In this section, we will use 2 well-known controller performance indices - the integral absolute error (IAE) and the percentage overshoot - to evaluate the performance of each of the five

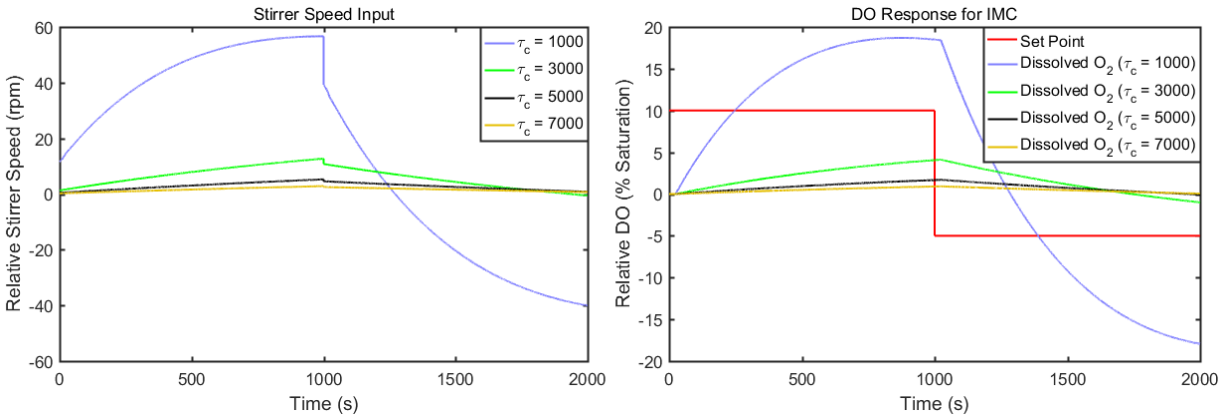

Figure 5.4: Response of the IMC controller G_{c3} to step changes in the dissolved oxygen set-point for Batch 12429.

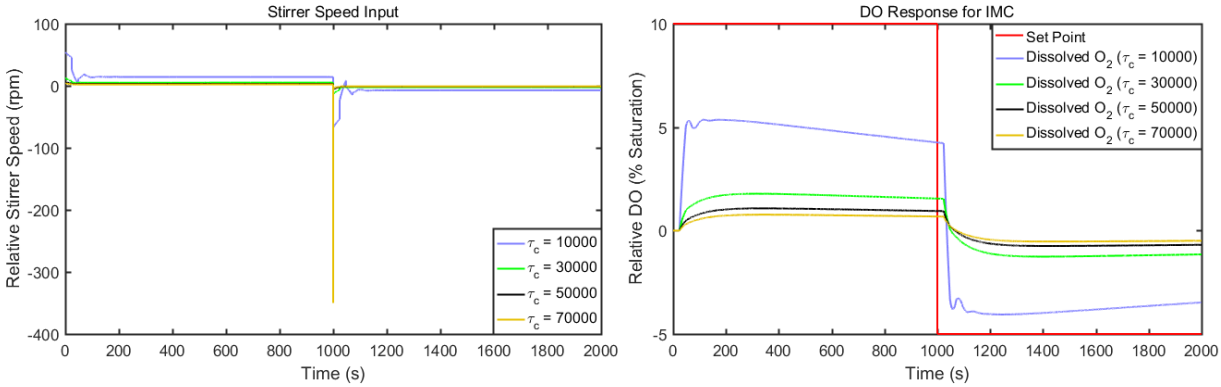

Figure 5.5: Response of the IMC controller G_{c3-Im} to step changes in the dissolved oxygen set-point for Batch 12429.

controllers derived in section 5.1.

For each experimental batch/plant model, τ_c will be varied over a large range and the values of the two performance indices will be computed as functions of τ_c. We will use the same type of simulation as the previous section, in which the set-point is stepped up from 0 to 10 at $t = 0$ and then stepped down to -5 at $t = 1000$ s, where it stays until the end of the simulation at $t = 2000$ s. The IAE will be calculated according to equation 4.1, while the percentage overshoot (ϵ) will be defined as:

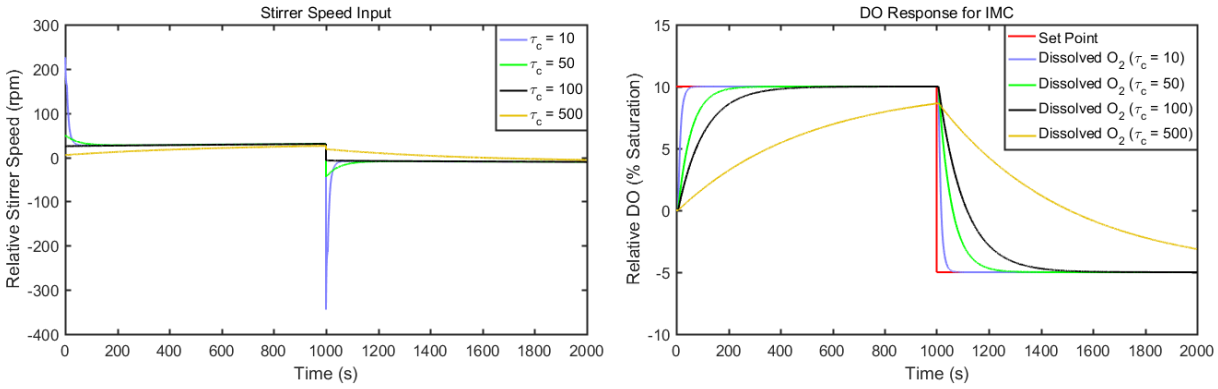

Figure 5.6: Response of the IMC controller to step changes in the dissolved oxygen set-point for Batch 12430.

$$\epsilon = \frac{y_{max} - y_{sp\,max}}{y_{sp\,max}} \times 100\% \tag{5.22}$$

where y_{max} is the maximum value of the dissolved oxygen, which usually results when the set-point is initially stepped up to 10 at time zero, and $y_{sp\,max}$ is the maximum value of the set-point (i.e. 10). For batches 3885, 3886, and 12430 - which correspond respectively to the IMC controllers G_{c1}, G_{c2}, and G_{c4} - τ_c was varied from 1 to 1000 seconds. The IAE and percent overshoot were computed for each τ_c in this range and the results plotted using MATLAB.

Figure 5.7 shows the variation in IAE and percent overshoot in response to changes in the filter time τ_c for batches 3885, 3886, and 12430. Note that the larger the filter time, the larger the integral absolute error and the smaller the overshoot. This is because a larger filter time causes the controller to respond more slowly and gradually, which results in slower variations in the manipulated variable/stirrer rate. The advantage of a large τ_c is that the controller causes less physical strain on the stirrer, while the disadvantage is that the controller is slower in tracking the set-point.

For good performance, an ideal controller should be able to closely track the set-point (i.e. low IAE) and also exhibit a small overshoot, since large overshoots can be detrimental to the organisms growing in the fermenter being controlled. Since optimal overshoot and optimal set-point tracking exert opposing influences on τ_c, it is possible to find an optimal τ_c in which the dissolved oxygen can track its set-point without a prohibitively large over-

shoot. From Figure 5.7, a filter time of around 15 seconds is best for batch 3885, because of the relatively small IAE (almost equal to the minimum IAE which occurs at $\tau_c = 10$) and overshoot. Likewise, for 3886 and 12430, the optimal τ_c's are at approximately $\tau_c = 11$ and $\tau_c = 9$ respectively.

The τ_c for batch 12429 was also varied and its performance criteria computed, using both G_{c3} and G_{c3-Im}. Here, however, the range of values that τ_c took on was different - from 10 to 10000 seconds for G_{c3} and from 10000 to 100000 seconds for G_{c3-Im}. Figure 5.8 shows the results corresponding to G_{c3}. Here, we notice that small filter time values result both excessively large overshoots and IAE, while larger τ_c tend to reduce the values of both performance indices. This relationship is not monotonic, however, since there is an optimal value of around 1500 seconds at which the IAE is minimum and the overshoot is nearly zero. Nevertheless, the set-point tracking even for $\tau_c = 1500$ is still too slow for G_{c3} to be considered a good controller.

On the other hand, 5.9 shows the results of the computation corresponding to G_{c3-Im}. It indicates that a lower τ_c of 10000 is the best option, since the 'overshoot' (more of an undershoot here) is closest to zero and the IAE is smallest at this value. It is possible to opt for even smaller τ_c but then this causes issues with the stability of the controller because the plant model for 12429 has a right-half plane zero. Because the performance of the G_{c3-Im} controller is slightly better than that of the G_{c3} controller, we will choose the improper controller for our simulations for 12429 in the next section.

One of the problems frequently experienced by the G_{c3} and G_{c3-Im} controllers is that if the time of the simulation is extended far enough, to say, around 20000 seconds, the system strangely begins to settle to a value that is the negative of the current set-point. For example, if the current set-point of dissolved oxygen is -5, then after some time following the initial dive towards -5, the control system settles back up to a dO_2 of +5. This phenomenon is far more egregious for the G_{c3} controller, which is another reason we have set it aside for the remainder of this chapter.

5.4 IMC Robustness Testing

Robustness is an important feature of controllers which pertains to their ability to deal with uncertainties in the process model parameters and with uncertain disturbances that may affect the system [43]. Broadly speaking, a robust controller is one whose response

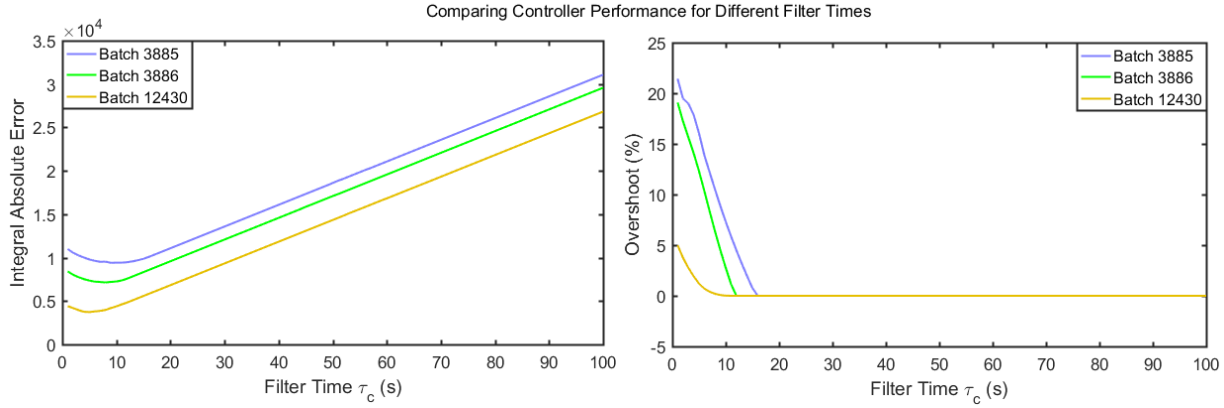

Figure 5.7: Integral Absolute Error (left panel) and Percent Overshoot (right panel) plotted as functions of the tuning parameter τ_c for the IMC controllers corresponding to batches 3885, 3886, and 12430.

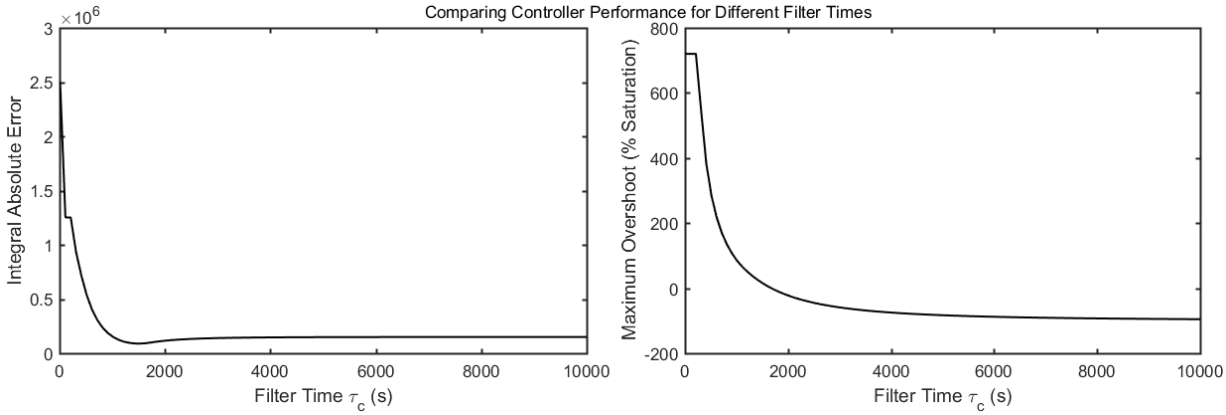

Figure 5.8: Integral Absolute Error (left panel) and Percent Overshoot (right panel) plotted as functions of the tuning parameter τ_c for the IMC controller corresponding to batch 12429, using G_{c3} as the controller.

remains effective even in the presence of uncertain or changing G_p parameters and/or external disturbances. Robustness is one of the key qualities needed in our controllers, because the make-up and behaviour of the microbial samples change from one batch to another. Consequently, it is necessary to have a controller which is capable of dealing with the ensuing uncertainty.

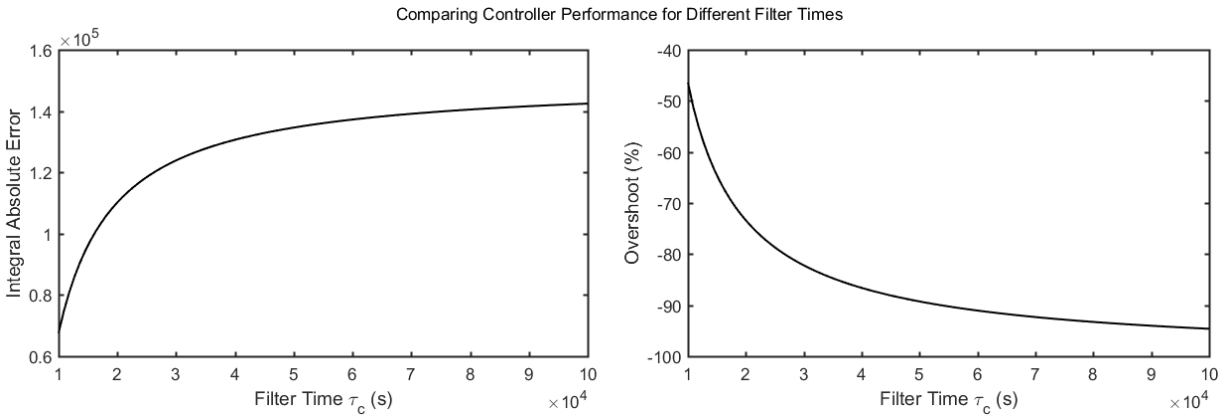

Figure 5.9: Integral Absolute Error (left panel) and Percent Overshoot (right panel) plotted as functions of the tuning parameter τ_c for the IMC controller corresponding to batch 12429, using G_{c3-Im} as the controller.

In this section, we will assess the robustness of the IMC controllers determined in section 5.1. Since the only tuning parameter is τ_c, our task is to vary τ_c for each of the four controllers and check whether the controller of interest maintains a robust response. We will use two ways to assess robustness. In the first method, we will evaluate the performance metrics of the controllers (i.e. IAE, Overshoot) for changing τ_c with respect to disturbance rejection at the input. In the second method, we will randomly generate 500 process models similar to the ones found for 3885, 3886, 12429, and 12430 and check how the performance metrics of the controllers vary across the generated models.

To test for robustness using the disturbance rejection method, we held the set-point at zero and stepped up the input disturbance to +20 rpm at $t = 0$ for a given τ_c. This added disturbance was held at the elevated level of +20 rpm until the end of the simulation at $t = 2000$. When a given simulation ended, the response of the dO_2 was compared to the set-point (zero), and the resulting error was used to compute the IAE. In addition, the overshoot ϵ_d was calculated according to the following expression:

$$\epsilon_d = y_{max} - y_{sp\,max} \tag{5.23}$$

where ϵ_d denotes the overshoot for a disturbance rejection control test. The procedure of calculating the IAE and overshoot for a given τ_c was repeated for multiple τ_c ([1,100] for 3885, 3886, 12429, and [10000,100000] for 12429) until full plots of the IAE and ϵ_d were

obtained as functions of τ_c for all four IMC systems. Figure 5.10 shows the performance metrics of batches 3885, 3886, and 12430 as functions of τ_c with respect to input disturbance rejection, while Figure 5.11 shows the variation in the controller performance metrics for batch 12429.

With respect to input disturbance rejection, both figures demonstrate that the lower the closed-loop time constant τ_c, the better the abilities of the controllers to respond to a disturbance at the input. Specifically, a lower τ_c minimizes both the IAE and the overshoot. That a small filter time constant minimizes the effect of input disturbances does not come as a surprise, mainly because a fast-acting controller can more easily adjust its output to match the new disturbance affecting it.

A comprehensive technique to check robustness for an IMC Controller involves generating a series of uncertain models and evaluating how well a given IMC Controller responds to those uncertain models. Specifically, our second robustness-testing method will require using the uncertain transfer function capability, where for each batch, we generate a base (nominal) process transfer function of the form:

$$G_p(s) = e^{-\theta_d s}\frac{p_1 s + p_2}{s^2 + p_3 s + p_4} \tag{5.24}$$

The nominal transfer function shall correspond to the actual process model for the particular batch being simulated (e.g. for batch 3885, the nominal transfer function is the SOPTDD model in Table 3.1). The nominal transfer function for each batch shall be the basis for generating 500 more models using 500 combinations of the parameters [p_1, p_2, p_3, p_4, θ_d].

The combinations are generated by supposing that each parameter in [p_1, p_2, p_3, p_4, θ_d] is uncertain and uniformly distributed on an interval $[p_{min}, p_{max}]$, where p_{min} is the minimum value of the parameter p found from using the minimum corresponding parameter value out of all four process models for batches 3885, 3886, 12429, and 12430. Similarly, p_{max} is the maximum value of the parameter p found from using the maximum corresponding parameter value out of all four process models for batches 3885, 3886, 12429, and 12430. For example, θ_d will be uniformly distributed on the interval [7,24], since the minimum value of the time delay is 7, and occurs in the model for batch 12430, while the maximum value of the time delay in all four batches is 24, and occurs in the model for batch 3885.

Once the intervals for the five uncertain parameters are generated, 500 values are ran-

domly generated for each parameter from within their interval. The 500 values for each parameter are then paired up, resulting in 500 parameter quintuplets and, by extension, 500 process models. The 500 process models created are kept the same for all four IMC controllers for the sake of consistency. Meanwhile, the 501st process model will be the nominal transfer function for each IMC system. Once all 501 models are generated for a given IMC Controller, simulations are performed to obtain the IAE and Overshoot performance metrics (with respect to set-point tracking) as functions of the model number. This process is repeated for different τ_c, ultimately resulting in a surface plot of the IAE or the overshoot ϵ as a function of both the model number (1-501) and τ_c.

The first of these surface plots is shown in Figure 5.12, which shows the IAE and Overshoot as functions of model number and τ_c for the 3885 IMC Controller. The figure indicates that low τ_c tend to result in less error, but also bring about a greater degree of overshoot. The same principle applies when the surface plots for the 3886 and 12429 IMC controllers (Figures 5.13 and 5.14) are examined. Figure 5.15, on the other hand, shows slightly different behaviour for the 12430 IMC controller. Here, both the IAE and overshoot are very large for small τ_c in many of the randomly generated model. Only at a slightly higher τ_c does the IAE sink back down again.

To better analyse the results of the aforementioned surface plots, we have plotted the average IAE/overshoot and the standard deviations of the IAE/overshoot for each τ_c for a particular controller. Ideally, to determine a robust controller, we wish to select a τ_c which yields a small average IAE and overshoot (across all 501 models) and a small standard deviation for the IAE and overshoot. A small mean would suggest that the controller can respond effectively to a large range of models (on average), while a small standard deviation would suggest that the controller's response stays relatively constant when the model parameters are varied.

Figure 5.16 shows the mean/standard deviation plots for the IAE and overshoot of batches 3885, 3886, and 12430 in response to changes in τ_c. The controllers from batches 3885 and 3886 appear to be most effective at around $\tau_c = 15$, where the mean and standard deviations of both the IAE and overshoot are relatively low. In contrast, batch 12430 tends to favor larger τ_c of around 40 s, primarily because of the surface plot showing large IAEs for very small τ_c in most of the 501 randomly generated models.

Finally, figure 5.17 shows the mean/standard deviation plots of IAE and overshoot for batch 12429. Here it is slightly more difficult to come up with an optimal value for τ_c. This is because while the mean IAE and overshoot favour a lower closed-loop time con-

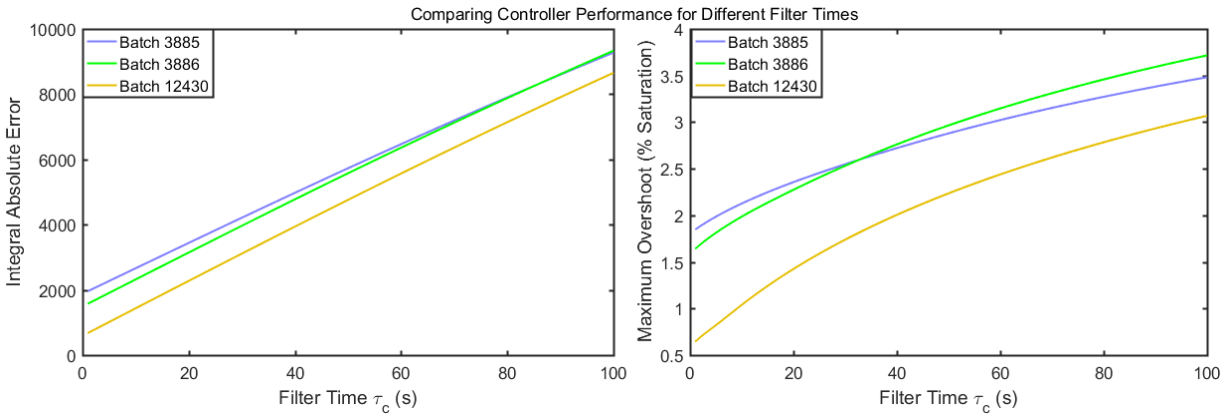

Figure 5.10: Integral Absolute Error (left panel) and Overshoot ϵ_d (right panel) plotted as functions of the tuning parameter τ_c for the IMC controllers corresponding to batches 3885, 3886, and 12430 in the presence of an input disturbance.

stant, the standard deviations favour a higher τ_c. However, since the standard deviations do not increase as much for low τ_c as the means decrease, we assert that a τ_c of 10000 achieves optimal robustness for batch 12429.

To conclude, our analysis of IMC controllers has indicated that lower closed-loop time constants tend to be favoured, in terms of not only the ability to engage in accurate set-point tracking but also the ability to reject disturbances and act robustly. For the IMC controllers for 3885, 3886, and 12430, optimal values of τ_c were found to be 15, 15, and 40 seconds respectively. On the other hand, for 12429, the optimal value of τ_c was 10000 seconds, simply because that was the lowest limit possible that would not give rise to poor control.

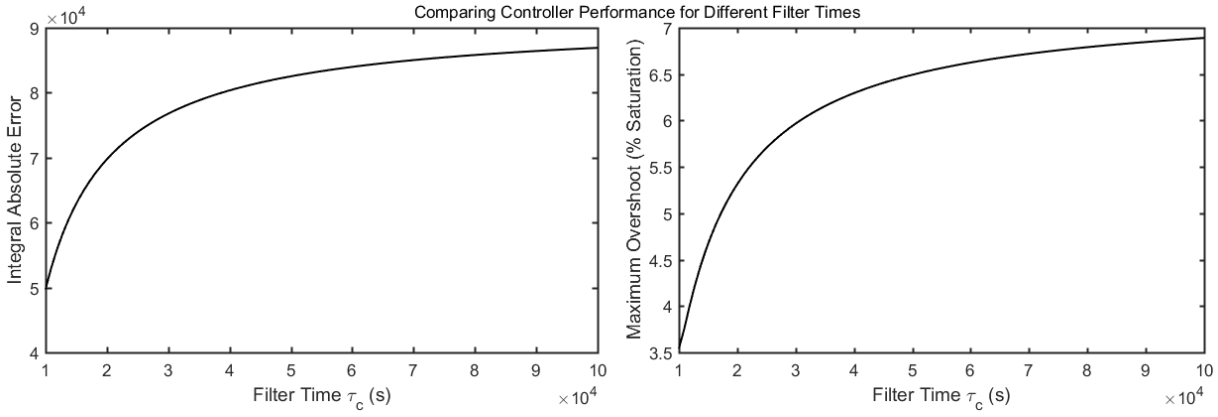

Figure 5.11: Integral Absolute Error (left panel) and Overshoot ϵ_d (right panel) plotted as functions of the tuning parameter τ_c for the IMC controller corresponding to batch 12429 in the presence of an input disturbance.

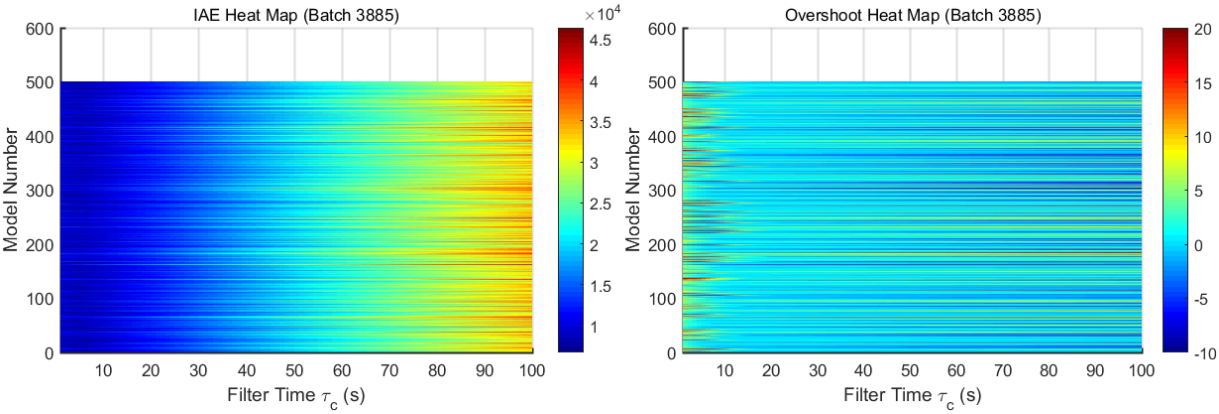

Figure 5.12: Integral Absolute Error (left panel) and Percent Overshoot (right panel) surfaces plotted as functions of the tuning parameter τ_c and the model number for the IMC controller corresponding to batch 3885.

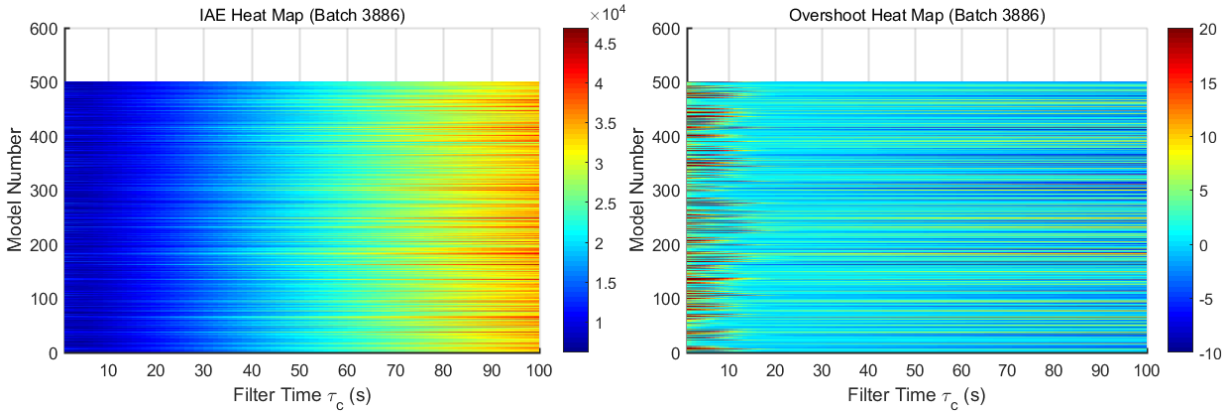

Figure 5.13: Integral Absolute Error (left panel) and Percent Overshoot (right panel) surfaces plotted as functions of the tuning parameter τ_c and the model number for the IMC controller corresponding to batch 3886.

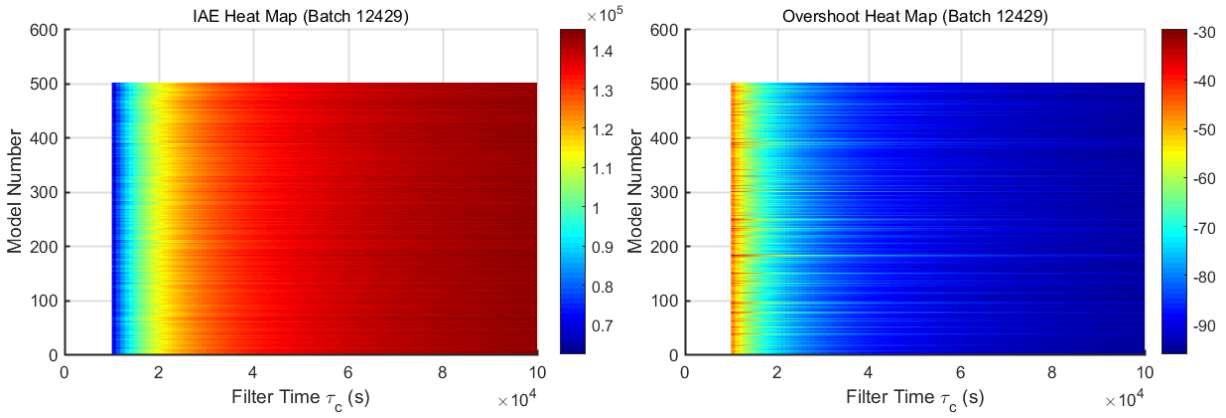

Figure 5.14: Integral Absolute Error (left panel) and Percent Overshoot (right panel) surfaces plotted as functions of the tuning parameter τ_c and the model number for the IMC controller corresponding to batch 12429.

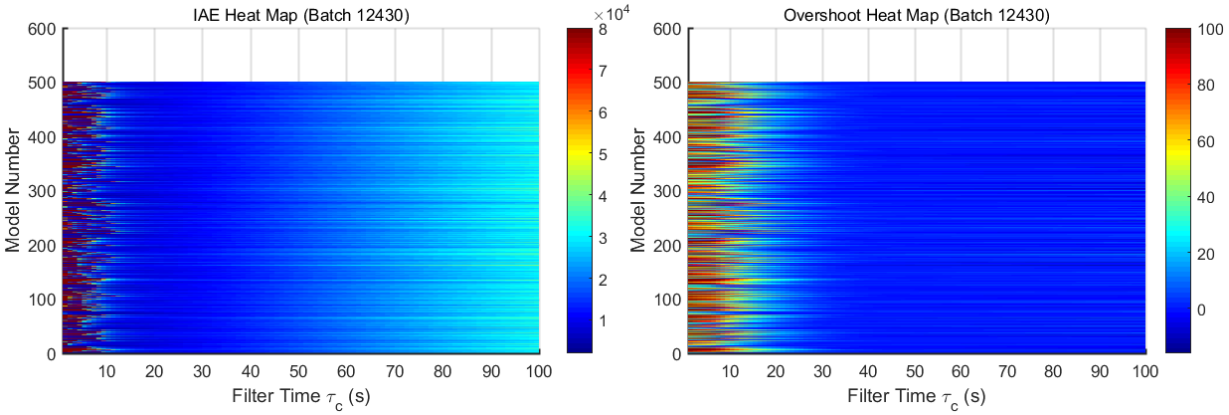

Figure 5.15: Integral Absolute Error (left panel) and Percent Overshoot (right panel) surfaces plotted as functions of the tuning parameter τ_c and the model number for the IMC controller corresponding to batch 12430.

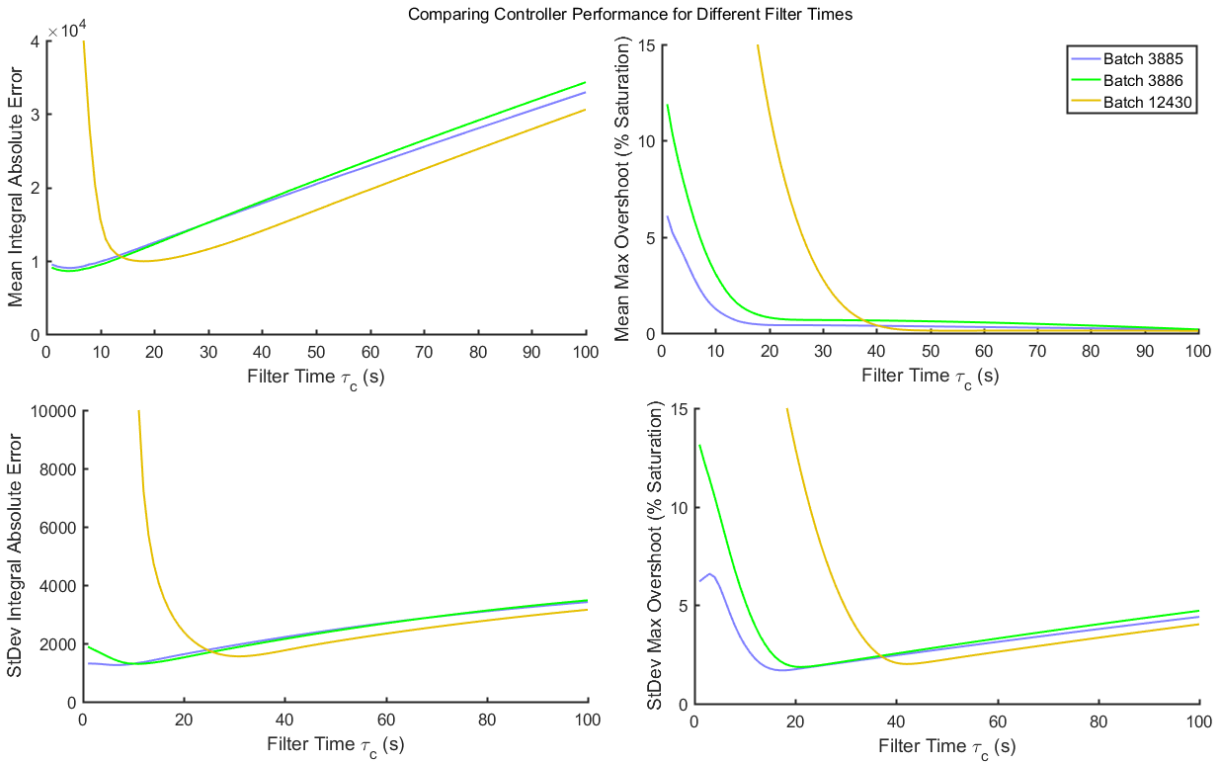

Figure 5.16: **Top Left:** Model-averaged IAE plotted against τ_c. **Top Right:** Model-averaged overshoot plotted against τ_c. **Bottom Left:** Standard deviation of IAE plotted against τ_c. **Bottom Right:** Standard deviation of overshoot plotted against τ_c. Figures all correspond to the IMC controllers for batches 3885, 3886, and 12430.

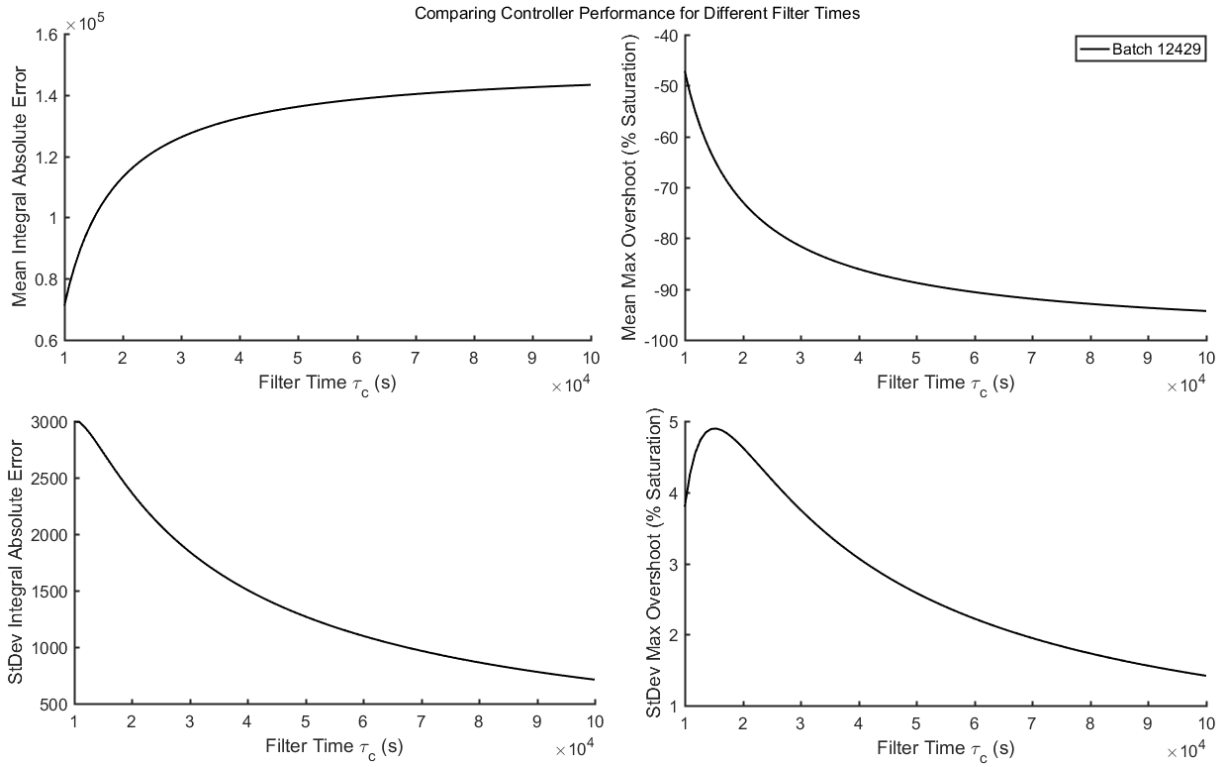

Figure 5.17: **Top Left:** Model-averaged IAE plotted against τ_c. **Top Right:** Model-averaged overshoot plotted against τ_c. **Bottom Left:** Standard deviation of IAE plotted against τ_c. **Bottom Right:** Standard deviation of overshoot plotted against τ_c. Figures all correspond to the IMC controller for batch 12429.

Chapter 6

Model Predictive Control and Nonlinear PID Control

In this chapter, we will implement two separate control techniques. The first technique, known as Model Predictive Control (MPC) is based on a discrete model of the control system. The second technique, dubbed Nonlinear PID Control, is based on obtaining the optimal PID settings for different ranges of the scheduling variable - in this case, the stirrer rate.

6.1 Theory of Model Predictive Control

With the emergence of digital control strategies and sophisticated techniques for feedback regulation, we believe it is important to examine in depth one such technique: Model Predictive Control (MPC) [13, 23]. Model Predictive Control relies on using a process model (i.e. G_p in Figure 2.2) to make predictions of future values of the output, or controlled variables (i.e. Dissolved Oxygen). These predictions are used to inform subsequent alterations to the manipulated variables, or inputs (i.e. Stirrer Rate).

While MPC does not typically propose a closed-form transfer function for the controller, it proposes a series of control moves that optimize the control response such that the discrepancy between the output variable and its corresponding set-point is minimized. Generally, MPC is used for multiple-input, multiple-output (MIMO) control systems, but here, we apply it to a single-input, single-output (SISO) control system.

For our work, the MPC controller designed uses a discrete formulation of the process model G_p, expressing it in terms of a step-response model. This model relates the value of the controlled variable y at a single sampling instant $k+1$ to changes in the manipulated variable u occurring at previous instants, and is given by the following relationship. Note here that as i increases, we move forward with the step response coefficients but backward with the change in the manipulated variable:

$$y(k+1) = y(0) + \sum_{i=1}^{N-1} S_i \Delta u(k-i+1) + S_N u(k-N+1) \tag{6.1}$$

where $y(0)$ is the initial value of y, $\Delta u(k-i+1) = u(k-i+1) - u(k-i)$, and $S_1, ..., S_N$ are the step-response coefficients, which represent the value of y at any sampling instant as a result of a step change in u from the sampling instant -1 to the sampling instant 0. For such a step change, since $\Delta u(0) = 1$ and all other Δu are zero, only the $i = k+1$ term is non-zero. Hence:

$$y(k+1) = y(0) + S_{k+1} \tag{6.2}$$

so the value of $y(k+1)$ is simply the initial value summed with the step-response coefficient corresponding to that time instant. The step-response coefficients can easily be derived from the process transfer function G_p by taking the inverse Laplace transform of $Y(s) = G_p U(s)$ for a unit-step input and applying the definition of the step-response coefficients by equating them to the value of the transformed function y at a particular sampling instant i. Note that here, we assume that the step-change is applied when $y(0) = 0$.

Model predictive control is a control strategy which uses step-response models, and is based on a receding horizon approach [13]. This approach makes use of 3 horizon quantities, described below:

1. Model Horizon (N): This describes the number of step-response coefficients used to calculate the process model. In other words, a model horizon of N means that the manipulated variable from N steps back is being used to compute the value of y at a sampling instant k.

2. Control Horizon (M): This parameter forms the basis for the MPC approach. If the current sampling instant is k, then the MPC approach computes a sequence of future

input changes (control moves) from k to $k+M-1$ (i.e. M control moves, including the present control move at k). Beyond $k+M-1$, the input u is made constant.

3. Prediction Horizon (P): The prediction horizon denotes the extent to which the model is predicted. In particular, the MPC calculation computes a series of control moves from k to $k+M-1$ such that the output predicted from k to $k+P$ reaches the set point in the most optimal fashion.

The fundamentals of MPCs and step-response models discussed above allow the derivation of the MPC approach for our controllers. Suppose that $y(0)=0$ and that k is the current sampling instant. The value of y at a future sampling instant $k+j$ (denoted by $\hat{y}(k+j)$), where j is a number from 1 to the prediction horizon P, can be deduced from equation 6.1:

$$\hat{y}(k+j) = \sum_{i=1}^{N-1} S_i \Delta u(k-i+j) + S_N u(k-N+j) \tag{6.3}$$

We can split up the summation term into two parts, with one going from $i=1$ to $i=j$ and the other going from $i=j+1$ to $i=N-1$:

$$\hat{y}(k+j) = \sum_{i=1}^{j} S_i \Delta u(k-i+j) + \sum_{i=j+1}^{N-1} S_i \Delta u(k-i+j) + S_N u(k-N+j)$$

As mentioned earlier, the larger the value of i in the summation, the further back we are in time with respect to the change in the manipulated variable. If k is our current sampling instant, that means we know the value of y and u until $y(k)$ and $u(k)$ respectively, but anything after that represents future values of y and u.

Thus, the first summation (from 1 to j) denotes the impact of future (corresponding to $i=1$ to $j-1$) and current (corresponding to $i=j$) control actions. On the other hand, the second summation (from $j+1$ to $N-1$) and the term outside involving S_N denote the impact of past control actions on the output y, where 'past' and 'future' are defined with respect to the current sampling instant k. Because the past control actions are fully known and the current and future control actions need to be computed, it is useful

to condense our writing by defining:

$$\hat{y}^o(k+j) \equiv \sum_{i=j+1}^{N-1} S_i \Delta u(k-i+j) + S_N u(k-N+j) \tag{6.4}$$

An interpretation of $\hat{y}^o$ is that is represents the value of y that would result if all future and current control moves were zero. In other words, if the future and present input values were held constant, the value of y would simply be $\hat{y}^o$. Using this definition of $\hat{y}^o$, our expression for $\hat{y}(k+j)$ becomes:

$$\hat{y}(k+j) = \hat{y}^o(k+j) + \sum_{i=1}^{j} S_i \Delta u(k-i+j) \tag{6.5}$$

For the purpose of computing our MPC, we will now define a vector of future predicted responses, denoted by $\hat{\boldsymbol{Y}}_{\boldsymbol{f}}$, given by:

$$\hat{\boldsymbol{Y}}_{\boldsymbol{f}} = \begin{bmatrix} \hat{y}(k+1) \\ \hat{y}(k+2) \\ \vdots \\ \hat{y}(k+P) \end{bmatrix}$$

The vector describing the contribution of past control actions to the future responses is denoted by $\hat{\boldsymbol{Y}}_{\boldsymbol{f}}^{\boldsymbol{o}}$:

$$\hat{\boldsymbol{Y}}_{\boldsymbol{f}}^{\boldsymbol{o}} = \begin{bmatrix} \hat{y}^o(k+1) \\ \hat{y}^o(k+2) \\ \vdots \\ \hat{y}^o(k+P) \end{bmatrix}$$

In order to write a vector equation for $\hat{\boldsymbol{Y}}_{\boldsymbol{f}}$ in terms of the manipulated variable and the contribution of the past control actions, it is necessary to define a $P \times M$ matrix of step response coefficients which will multiply the manipulated variables in the vector equation:

$$\boldsymbol{S} = \begin{bmatrix} S_1 & 0 & \cdots & 0 \\ S_2 & S_1 & 0 & \vdots \\ \vdots & \vdots & \ddots & 0 \\ S_M & S_{M-1} & \cdots & 0 \\ S_{M+1} & S_M & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ S_P & S_{P-1} & \cdots & S_{P-M+1} \end{bmatrix}$$

If the vector denoting the future changes in the manipulated variables is $\boldsymbol{\Delta U}_f$:

$$\boldsymbol{\Delta U}_f = \begin{bmatrix} \Delta u(k) \\ \Delta u(k+1) \\ \vdots \\ \Delta u(k+M-1) \end{bmatrix}$$

then we can write the expression for the future output predictions in vector form as:

$$\hat{\boldsymbol{Y}}_{\boldsymbol{f}} = \hat{\boldsymbol{Y}}_{\boldsymbol{f}}^{\boldsymbol{o}} + \boldsymbol{S}\boldsymbol{\Delta U}_f \tag{6.6}$$

There are two unknowns in this equation, the future control action $\boldsymbol{\Delta U}_f$, and the resulting behaviour of the predicted future output $\hat{\boldsymbol{Y}}_f$. The goal of MPC is to compute an optimal $\boldsymbol{\Delta U}_f$ which satisfies any lower and upper bounds placed upon the system. Generally, this is done with an optimization process which seeks to minimize a cost function (or performance index) containing the predicted error vector $\hat{\boldsymbol{E}}_{\boldsymbol{f}} = \boldsymbol{Y}_{\boldsymbol{r}} - \hat{\boldsymbol{Y}}_{\boldsymbol{f}}$, where $\boldsymbol{Y}_{\boldsymbol{r}}$ is the reference trajectory. Usually, the reference trajectory represents the desired set point of the controlled variable y, or y_{sp}. For an MPC in which the input and output are not constrained, the performance index is given by:

$$\boldsymbol{J} = \hat{\boldsymbol{E}}_{\boldsymbol{f}}^{\boldsymbol{T}}\boldsymbol{Q}\hat{\boldsymbol{E}}_{\boldsymbol{f}} + \boldsymbol{\Delta U}_f^{\boldsymbol{T}}\boldsymbol{R}\boldsymbol{\Delta U}_f \tag{6.7}$$

where $\boldsymbol{Q}$ and $\boldsymbol{R}$ are weighting matrices. $\boldsymbol{Q}$ is chosen as a positive-definite (i.e. positive eigenvalues) $P \times P$ matrix and $\boldsymbol{R}$ is chosen as a positive semi-definite $M \times M$ matrix (i.e. non-negative eigenvalues). $\boldsymbol{Q}$ and $\boldsymbol{R}$ are selected in order to balance between the need for ensuring good set-point tracking (i.e. a low predicted error) and the need for smoother, less aggressive control (i.e. low magnitude of the control moves $\boldsymbol{\Delta U}_f$). The MPC control

scheme is obtained by minimizing the performance index $\boldsymbol{J}$ with respect to $\boldsymbol{\Delta U}_f$. Using analytical techniques, it can be shown that the series of control moves which minimizes the performance index is given by:

$$\boldsymbol{\Delta U}_{f\ opt} = (\boldsymbol{S^T Q S} + \boldsymbol{R})^{-1} \boldsymbol{S^T Q} \hat{\boldsymbol{E}}_f^{\boldsymbol{o}} \tag{6.8}$$

where $\hat{\boldsymbol{E}}_f^{\boldsymbol{o}}$ is the error vector which results from when no further control action is taken at and beyond the sampling instant k. In other words, $\hat{\boldsymbol{E}}_f^{\boldsymbol{o}} = \boldsymbol{Y_r} - \hat{\boldsymbol{Y}}_f^{\boldsymbol{o}}$. It is possible to re-express the vector of future control moves in terms of a gain matrix $\boldsymbol{K}_c$ multiplying $\hat{\boldsymbol{E}}_f^{\boldsymbol{o}}$, if we define $\boldsymbol{K}_c \equiv (\boldsymbol{S^T Q S} + \boldsymbol{R})^{-1} \boldsymbol{S^T Q}$. It should be noted that $\boldsymbol{K}_c$ depends only on static quantities and tuning parameters, provided that the step response matrix $\boldsymbol{S}$ is independent of time, a rule which applies in our case.

While the optimal control moves $\boldsymbol{\Delta U}_f$ apply for the M sampling instants, according to the definition of the control horizon, only the first (current) move is used for the actual control. When this control move is carried out at an instant k, the next M control moves are then computed using equation 6.8, starting at the time instant $k+1$. Once computed, only the move at $k+1$ is implemented and the process continues. Because only the first control move is implemented out of the M computed, the 'end' of the control horizon can be seen as moving forward with the sampling instant, giving rise to the receding horizon approach inherent to Model Predictive Control.

One of the advantages of MPC is that it is capable of using hard inequality constraints for the input variable u. In our case, this is especially useful because of the mechanical constraints of 100 (lower limit) and 675 (upper limit) rpm on the stirrer rate for the microbial fermenter. The main difference between a constrained MPC and its unconstrained counterpart would be the additional terms assigning a steep penalty to control moves that venture into the 'out-of-bounds' region in the expression for the performance index (equation 6.7). In addition to these inequality constraints, the design of an MPC incorporates multiple tuning parameters, which include the model horizon N, the control horizon M, the prediction horizon P, as well as the weighting matrices $\boldsymbol{Q}$ and $\boldsymbol{R}$.

Though there are no hard and fast rules for MPC tuning and ensuring the closed-loop stability of MPC controllers, some guidelines exist which may inform the selection of the aforementioned tuning parameters. For example, the model horizon is usually selected so that the product between the model horizon and the sampling time T_s equals the settling time τ of the process (i.e. $N = \tau/T_s$). Furthermore, the control horizon typically

lies between one-third and one-half of the model horizon, while the prediction horizon is typically selected as the sum of the model and control horizons.

For a SISO model, the elements of $\boldsymbol{Q}$ and $\boldsymbol{R}$ are selected according to performance criteria for the controller. For instance, the matrix $\boldsymbol{Q}$ contributes to the error term in the performance index in equation 6.7. Thus, larger entries in $\boldsymbol{Q}$ tend to make the MPC more aggressive in reaching the set point. In contrast, the matrix $\boldsymbol{R}$, oft dubbed the 'move suppression matrix', ensures that the control moves are kept at low levels, leading to less aggressive control.

For our MPC simulations in section 6.2, we will develop MPC controllers for each experimental batch, and use multiple values of the tuning parameters (i.e. the horizons and the weighting matrices) to compare the quality of the control with respect to the IAE and overshoot. Then, in section 6.3, we will perform similar testing to determine the combination of tuning parameters which best maximize the robustness of control, using a combination of disturbance rejection and plant-model mismatch simulations.

6.2 Implementing MPC

To begin, we will set our model horizon equal to the larger time constant of the G_p corresponding to a given batch rounded up to the nearest 100. For example, in batch 3885's SOPTDD model, the larger time constant is $\tau_2 = 5656$ seconds, while the sample time for the experimental data is 6 seconds. Therefore, the model horizon for this example will be $N = 5656/6 = 942.67$ rounded up to the nearest 100, meaning that the model horizon is $N = 1000$ for this experimental batch.

For all our MPC simulations, we will use the same procedure to compute the model horizon. By default, the control horizon equals 40% of the model horizon, and the prediction horizon equals the sum of the model horizon and the control horizon. In addition, the weighting matrices $\boldsymbol{Q}$ and $\boldsymbol{R}$ are specified as diagonal matrices with the same entry throughout the diagonal (henceforth referred to as the weight).

By default, the weight on $\boldsymbol{Q}$ (W_Q) is set to 1 while the weight on $\boldsymbol{R}$ (W_R) is set to 0.1. Since the relative ratio between the weight on $\boldsymbol{Q}$ to the weight on $\boldsymbol{R}$ is more indicative of the emphasis placed on set point tracking vs less aggressive control, we will vary this ratio because that is the main quantity of interest. To vary the weight ratio, W_R

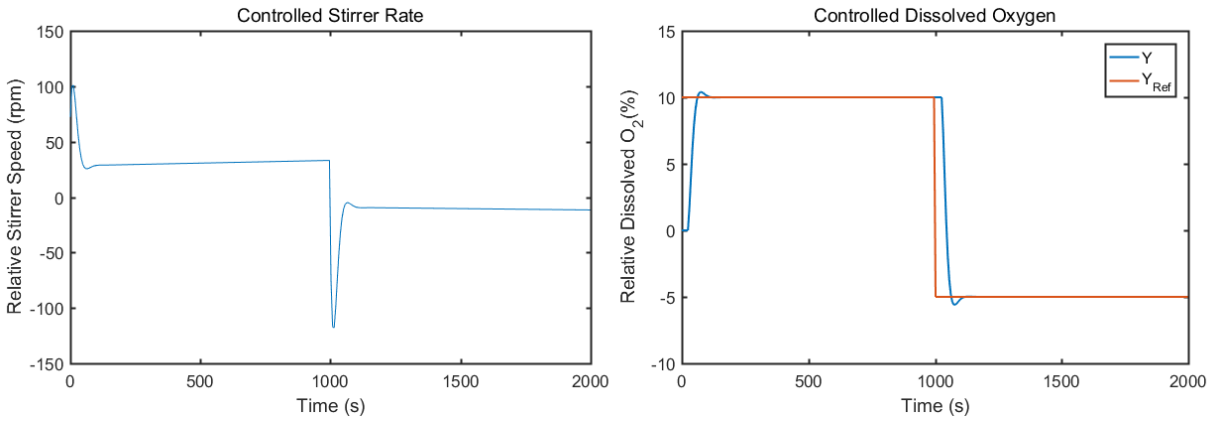

Figure 6.1: Response of the default MPC controller to step changes in the dissolved oxygen setpoint for Batch 3885. MPC Parameters:
$N = 1000,\ P = 1400,\ M = 400,\ W_Q = 1,\ W_R = 0.1.$

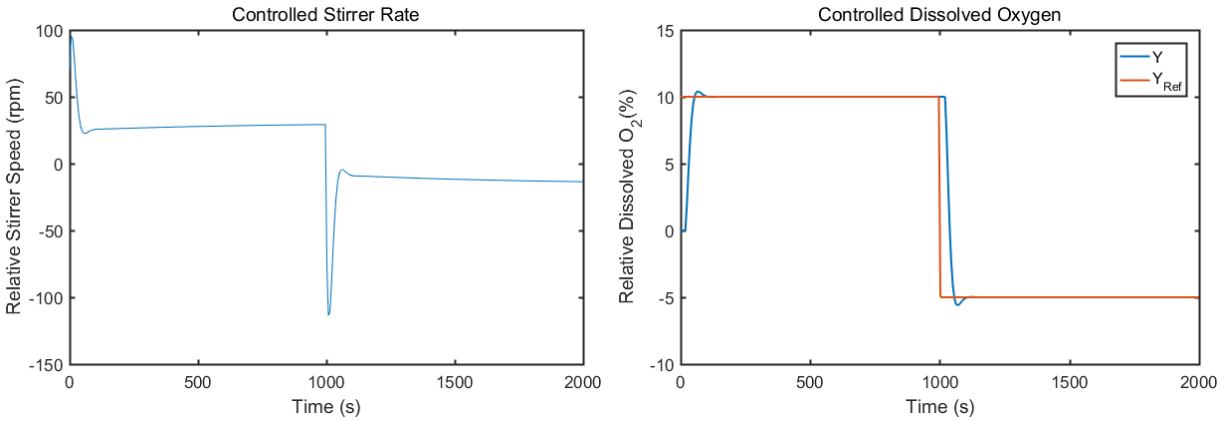

Figure 6.2: Response of the default MPC controller to step changes in the dissolved oxygen setpoint for Batch 3886. MPC Parameters:
$N = 100,\ P = 140,\ M = 40,\ W_Q = 1,\ W_R = 0.1.$

will be kept constant while W_Q will be varied. Figures 6.1, 6.2, and 6.3 show the responses of the default MPC controllers for the process models in batches 3885, 3886, and 12430 respectively. Once again, the stimulus being tested is a step change in the set point of dissolved oxygen by 10 units upwards followed by a downward step change to a relative dissolved oxygen of -5 at 1000 seconds.

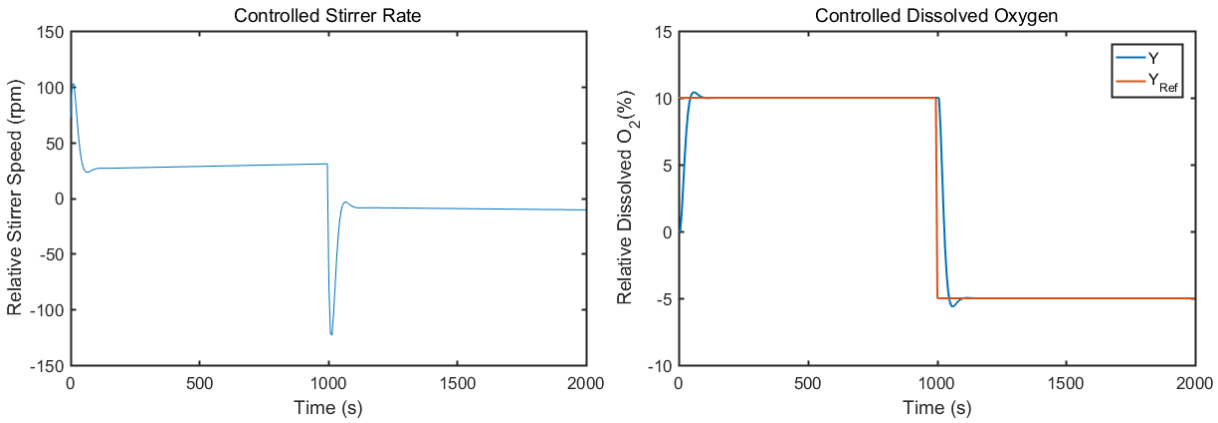

Figure 6.3: Response of the default MPC controller to step changes in the dissolved oxygen setpoint for Batch 12430. MPC Parameters:
$N = 1000,\ P = 1400,\ M = 400,\ W_Q = 1,\ W_R = 0.1.$

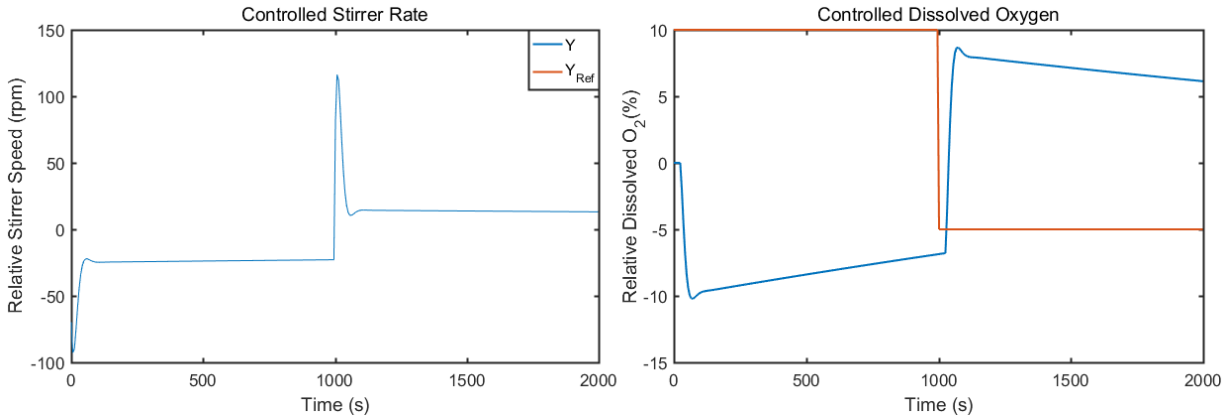

Figure 6.4: Response of the default MPC controller to step changes in the dissolved oxygen setpoint for Batch 12429. MPC Parameters:
$N = 2000,\ P = 2800,\ M = 800,\ W_Q = 1,\ W_R = 0.1.$

From these figures, the MPCs appear to respond very quickly to step changes in the dissolved oxygen set points and exhibit minimal overshoot. Generally, the model predictive controllers are superior to their internal model control counterparts, exhibiting smaller IAE and smaller overshoot. One notable exception to this rule is figure 6.4, which shows the response of the MPC for batch 12429 to step changes in the set-point for batch 12429. Because the MPC response is so poor - since it is in the opposite direction of the set-point

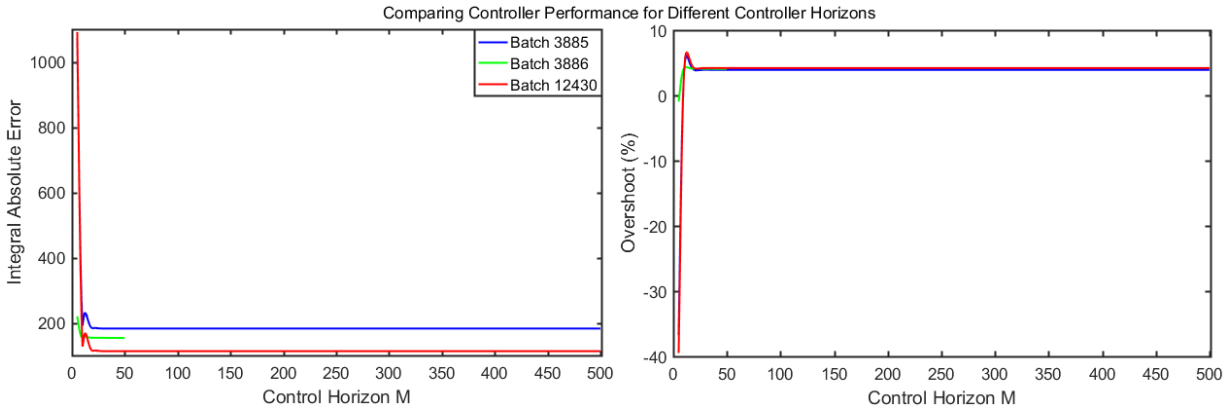

Figure 6.5: Integral Absolute Error (left panel) and Percent Overshoot (right panel) plotted as functions of the tuning parameter M for the MPC controllers corresponding to batches 3885, 3886, and 12430.

changes - we will henceforth ignore the MPC for 12429. A possible reason for this poor performance is that like the IMC Controller for 12429, a specifically designed MPC Controller is incapable of dealing with the right-half plane zero in the G_p for 12429.

While default parameter configurations offer us sound model predictive controller objects, it is necessary to go one step further and modify the tuning parameters to examine the variations in the performance of the MPC with respect to different tuning parameter values. Here, the tuning parameters varied include the control horizon M, the prediction horizon P, and the ratio between the weights W_Q and W_R (denoted by γ).

First, the control horizon was varied from 5 sampling intervals (i.e. a time of 30 seconds given the sampling time of 6 seconds) to $0.5N$ for batches 3885, 3886, and 12430. The IAE and percentage overshoot were calculated for each control horizon value between 5 and $0.5N$, keeping the other tuning parameters (P and γ) constant at their default values. The results for all batches are plotted in Figure 6.5.

From the figure, larger control horizons tend to result in smaller error across all three process models. The reason for this is that large control horizons allow the controller to plan the control moves several intervals in advance. Earlier planning results in more precise control which aids the ability of the MPC controller to perform set-point tracking.

For overshoot, larger control horizons give rise to a non-monotonic response. At very

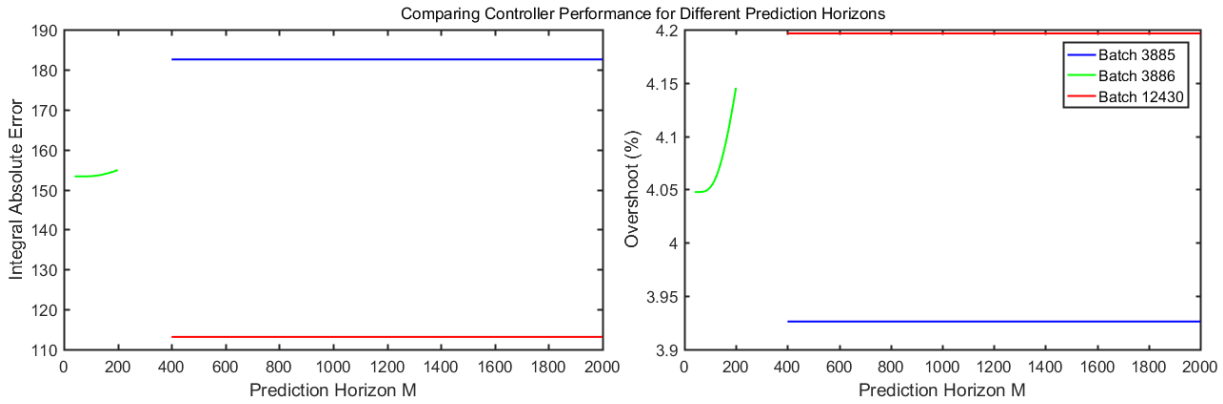

Figure 6.6: Integral Absolute Error (left panel) and Percent Overshoot (right panel) plotted as functions of the tuning parameter P for the MPC controllers corresponding to batches 3885, 3886, and 12430.

low control horizons, the overshoot increases drastically as the control horizon is raised (to 7-8%), but for slightly higher control horizons, the overshoot tends to decrease as M rises to about 4-5%. A possible reason for the decreasing phase in the overshoot plot is that larger control horizons make the controller less hasty and more precise when it comes to approaching the set-point. As a result, the system approaches the set-point more gradually instead of drastically shooting up and then being forced to come back down.

The second parameter varied was the prediction horizon, which was changed from the default control horizon M to $2N$, with the IAE and overshoot corresponding to the prediction horizons within this range plotted in Figure 6.6, for batches 3885, 3886, and 12430. Given default values of M and γ, the prediction horizon has very little effect on the performance of the MPC controller.

This is especially true for batches 3885 and 12430, where the IAE and overshoot stay relatively constant for the entire range over which P is varied. However, batch 3886 shows slightly different behaviour, with both the IAE and overshoot rising slightly as P is increased. A possible reason for the lack of a non-constant response is that the prediction horizon for 3885 and 12430 is already large enough at 400 that increasing it further does not add much to the ability of the MPC controller to track the set-point.

The next parameter varied for the MPC was the weight ratio $\gamma = W_Q/W_R$. This ratio was varied from 0.1 to 100 with W_R held constant at 0.1. In other words, only the

weight on the error W_Q was varied from 0.01 to 10. Nonetheless, the absolute values of W_Q and W_R are largely irrelevant, since as described earlier, the ratio between these weights matters most. Figure 6.7 shows the variations in IAE and overshoot when γ is varied from 0.1 to 100, with M and P held constant at their default values.

As the weight ratio increases, the IAE for all 3 batches drops, while the overshoot undergoes 'oscillatory' changes - rising, falling, and rising again. The reason for this behaviour is that a greater ratio results in more emphasis being placed on the error term of the performance index $\boldsymbol{J}$. This causes the controller to be more aggressive and match the set point more closely. A more aggressive controller results in a smaller IAE.

However, because the manipulated variable U changes more rapidly as a result of aggressive control, there is a greater degree of overshoot for larger γ. Nonetheless, a greater overshoot also contributes to a larger IAE, which is why the overshoot exhibits a rising and falling pattern - it rises because of more aggressive control changing the manipulated variable very quickly, but it also falls because more aggressive control makes the measured value of Y track its set-point more effectively.

While Figure 6.7 generally points to a larger weight ratio for better control, just as Figures 6.5 and 6.6 point to larger control and prediction horizons, it should be noted that a larger weight ratio will result in more rapid changes to the manipulated variable (stirrer rate). These rapid changes may not be ideal for the mechanical well-being of the system, but are acceptable here because the mechanical well-being cannot be fully ascertained from *in silico* simulations.

A wider examination of the effects of the control and prediction horizons can be performed using IAE and overshoot surface plots (or heat maps). To generate these surface plots, a vector of control horizons is generated in which M is varied from 2 to $0.4N$. Additionally, a vector of prediction horizons is generated, where P is varied from 3 to $1.4N$ (i.e. $N + 0.4N$) for each batch model. Then, MPCs are generated using the prediction and control horizon vectors, such that every control horizon is paired with every prediction horizon greater than or equal to it.

Each generated MPC is used in the set-point tracking simulation, where the IAE and overshoot are computed for specific values of the control and prediction horizons. Then, the IAE and overshoot of the MPCs for that particular batch are plotted against the prediction and control horizons, generating a 3-D surface plot. The first of these surface plots is shown in Figure 6.8, which corresponds to the IAE and overshoot for batch 3885. Most

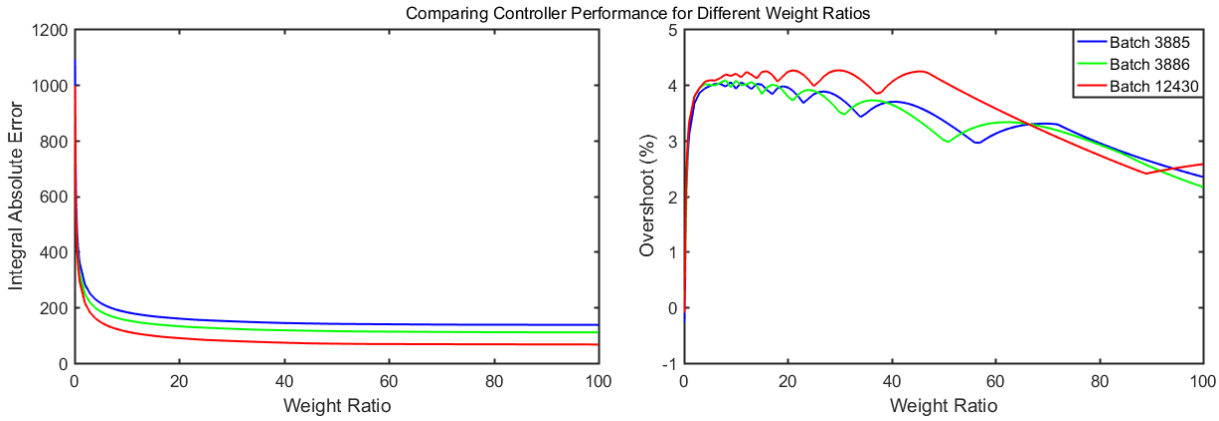

Figure 6.7: Integral Absolute Error (left panel) and Percent Overshoot (right panel) plotted as functions of the tuning parameter γ for the MPC controllers corresponding to batches 3885, 3886, and 12430.

of the error and overshoot difficulties occur only when the control horizon is very small and the prediction horizon is relatively large. In contrast, for large control and prediction horizons, the IAE and overshoot are generally well-behaved because the controller can 'see further ahead', allowing it to make control moves more precisely.

A similar pattern is repeated with batches 3886 (Figure 6.9) and 12430 (Figure 6.10), where large control and prediction horizons result in better IAEs and overshoots for set-point tracking simulations. Because the IAE and overshoot stay relatively constant over such a large range of M and P, there is a considerable degree of freedom in selecting a combination of M and P control and prediction horizons. For instance, we can choose to select relatively small M and P, such as $(M, P) = (50, 100)$ for batch 3885 in order to save time and memory during computation.

6.3 Robust Model Predictive Control

Robustness is an essential feature of controllers which pertains to their ability to respond to external disturbances and uncertain process models. Determining the robustness of an MPC controller for a particular process model will involve varying the control horizon, prediction horizon, and weight ratio. That three tuning parameters need to be varied to determine the most robust combination is rather cumbersome, since having to change three

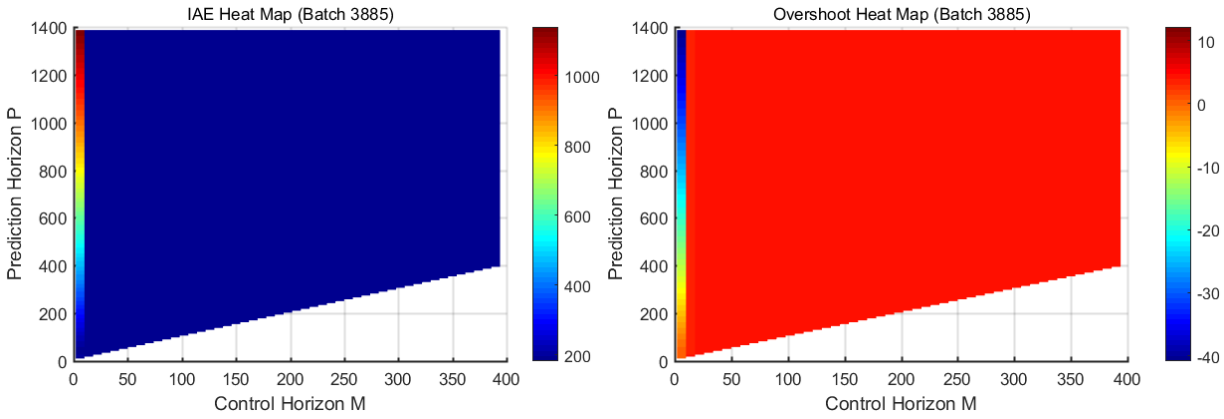

Figure 6.8: Integral Absolute Error (left panel) and Percent Overshoot (right panel) surfaces plotted as functions of the tuning parameters M and P for the MPC controller corresponding to batch 3885.

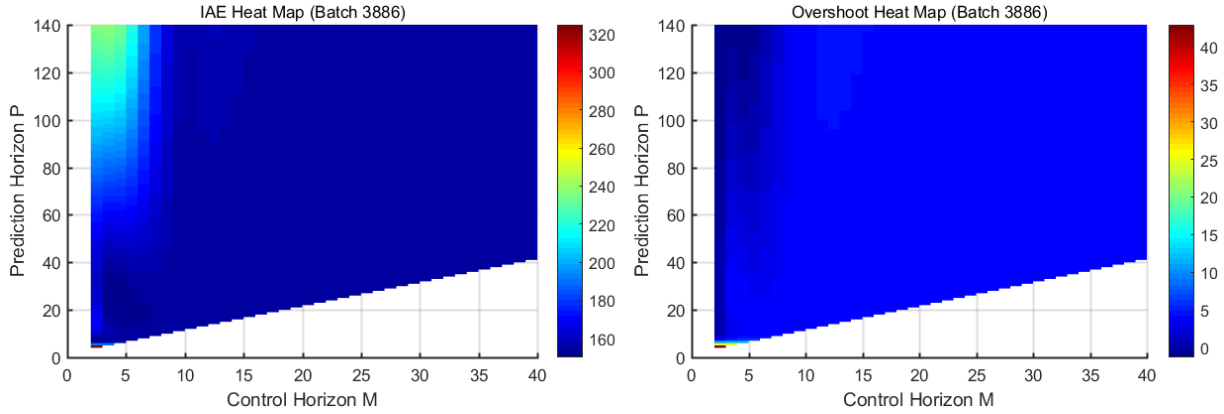

Figure 6.9: Integral Absolute Error (left panel) and Percent Overshoot (right panel) surfaces plotted as functions of the tuning parameters M and P for the MPC controller corresponding to batch 3886.

parameters across multiple models and evaluating the IAE/overshoot will be extremely expensive computationally.

Fortunately, the results from the previous section pave the way for a much simpler analysis of MPC robustness which will achieve a similar objective of finding a robust parameter combination without the computational load. Using the principles derived from

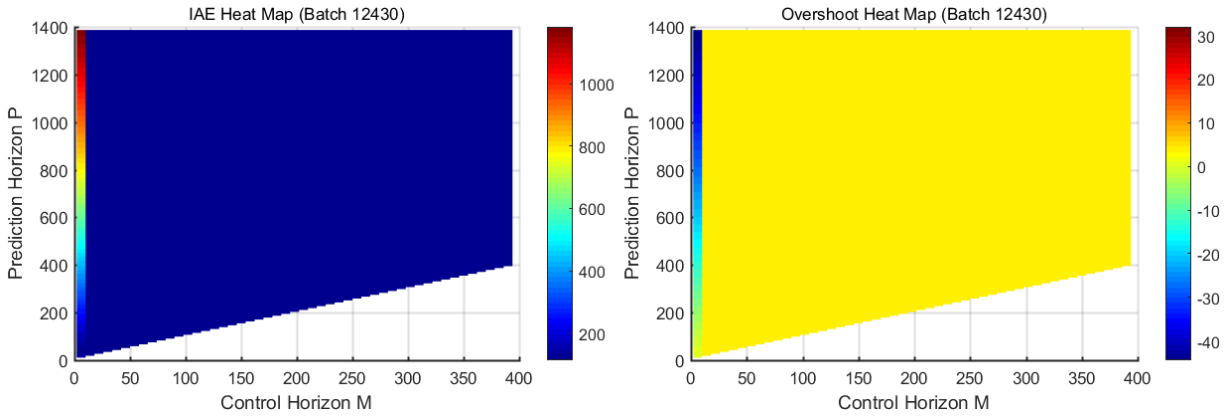

Figure 6.10: Integral Absolute Error (left panel) and Percent Overshoot (right panel) surfaces plotted as functions of the tuning parameters M and P for the MPC controller corresponding to batch 12430.

Design of Experiments [44], we will select four points on the (M,P) surface plots for each batch where the IAE and Overshoot are minimized (i.e the uniformly coloured region).

These four points will be situated near the corners (extremities) of the surface where the IAE is relatively small and uniform. Then, for each of these four pairs of (M,P) values, 500 randomly generated process transfer functions will be simulated and for the same set-point change, the mean and standard deviation of the IAE and overshoot of the MPC controllers will be evaluated. For consistency, these 500 generated transfer functions are the same as those used for the IMC Controllers.

After this process is carried out for all three batches, the best (M,P) pair is selected and used for the remainder of the robustness analysis. Using the optimal (M,P) pair for each batch, the weight ratio is varied from 0.1 to 40 for 500 randomly generated process transfer functions. Then, the resulting mean and standard deviations of the IAE and overshoot are plotted together for each batch to get a stronger idea of which weight ratio is most optimal for a well-designed, robust MPC controller.

The four points selected for the initial (M,P) robustness tests for each of the three batches are shown in Table 6.1. Each of these four points was selected from the extreme ends of the IAE surface plots. In addition, the differences between the IAEs of all four points for every batch is less than 1%, making these points roughly equivalent in terms of control accuracy for their particular process models.

Batch ID	Point 1	Point 2	Point 3	Point 4
3885	(18,19)	(18,1387)	(394,395)	(394,1387)
3886	(9,10)	(9,140)	(40,41)	(40,140)
12430	(18,19)	(18,1387)	(394,395)	(394,1387)

Table 6.1: Four pairs of horizons (M,P) selected for each batch in order to perform robustness analysis.

A bar graph of the mean and standard deviation of the IAE and overshoot for batch 3885 in response to multiple randomly generated G_p's is shown in Figure 6.11. Here, the mean and standard deviations of the IAE are almost equal regardless of the control/prediction horizon pair chosen. The only differences appear in the overshoot bar graph, where points 1, 3, and 4 seem to have the lowest mean overshoots. Because point 3 represents the pair of lowest IAE from the surface plot (Figure 6.8), we will select point 3 as the control-prediction horizon pair when varying γ across the randomly generated models to obtain an optimally robust controller.

The corresponding bar graphs for batches 3886 and 12430 are shown in Figures 6.12 and 6.13 respectively. Repeating the pattern from batch 3885, the 4 points do not appear to have a significant effect on changing the IAE and overshoot. Because we cannot differentiate between the 4 horizon pairs using this technique, we will once again choose point 3 for both batches 3886 and 12430 as it gives the lowest IAE according to the error surfaces plotted earlier (Figures 6.9 and 6.10).

With point 3 serving as the chosen point for batch 3885, we then plotted the averaged IAE and overshoot as functions of the weight ratio in Figure 6.14. Here, the averaged IAE is very high for small weight ratios, but then sinks back down and rises back up again as the weight ratio increases. The mechanism underlying this is as follows: when the weight ratio is very small, the system is simply unable to keep up with changes in the set-point, a problem which persists through multiple plants.

In consequence, the IAE is large, even though the overshoot is small because the controller is fairly conservative in calculating its moves. Moreover, as the weight ratio increases, the controller becomes more aggressive for all randomly generated transfer function processes, and so results in a smaller IAE, even though the overshoot increases very slightly. However, for very large weight ratios, the controller becomes so aggressive that it

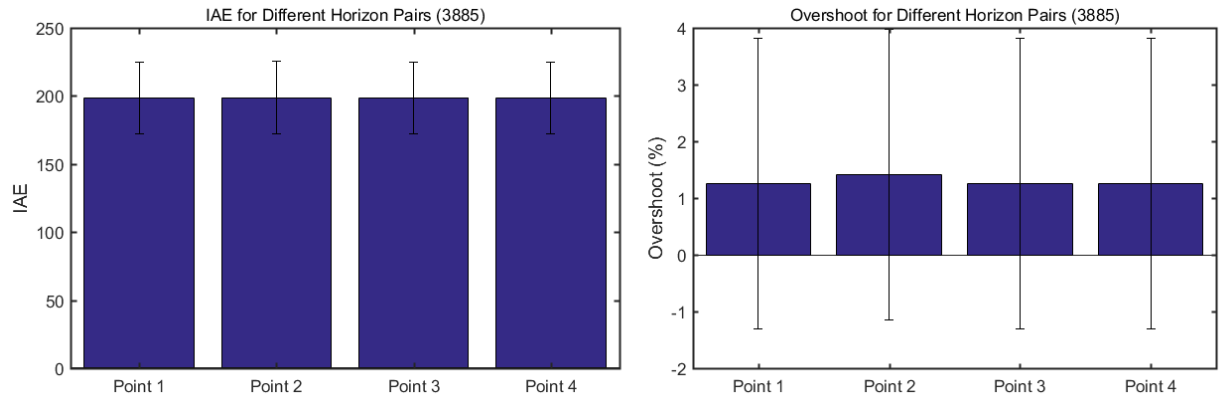

Figure 6.11: Integral Absolute Error (left panel) and Percent Overshoot (right panel) bars plotted for each Control/Prediction Horizon Point, for batch 3885. Error bars denote the standard deviation.

begins to lose its robustness, which often requires more conservative control action in the face of uncertainty.

As a result, both the IAE and overshoot undergo a dramatic increase for larger weight ratios, a pattern which is repeated by batches 3886 and 12430 as well (Figures 6.15 and 6.16). Thus, analysis of MPC controllers has demonstrated that large control horizons, along with large prediction horizons (but not too much larger than the control horizons) are ideal for tight, robust control. As far as the weight ratio of the MPC is concerned, a balance needs to be kept at around a weight ratio of approximately 10-15 across all 3 batches to ensure an acceptable combination of robustness and effective set-point tracking.

6.4 Nonlinear Model and PID Control

In this section, we will shift away from Model Predictive Control and focus on a gain-scheduling control strategy which takes into account the nonlinearities in the system. In Chapter 4, we showed that process models estimated over a range of between 400 and 500 rpm for 4 different batches provided us with a set of optimized settings that worked well in controlling the dissolved oxygen of the biological fermenter. However, in reality, it has been observed that while the system behaves linearly over a small range of stirrer rates (e.g. from 400-500 rpm), there is a possibility that the system is nonlinear over the entire

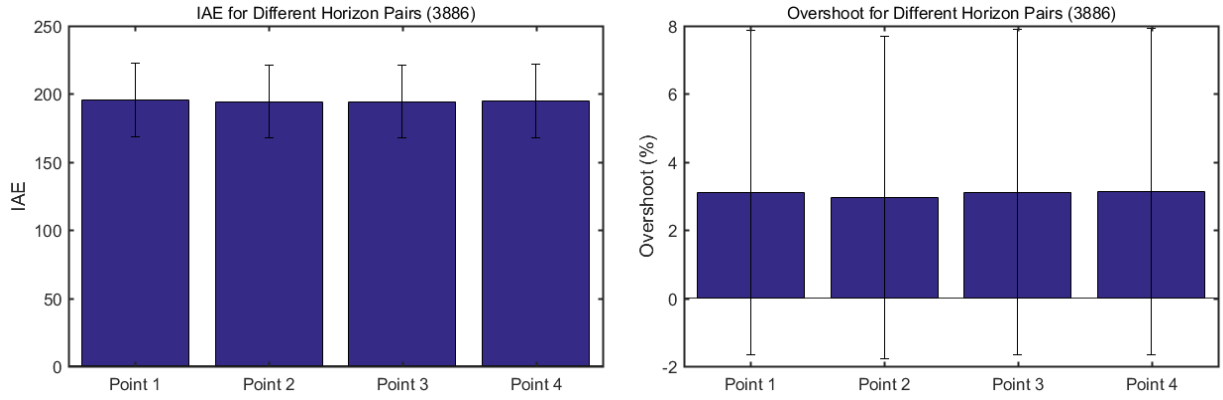

Figure 6.12: Integral Absolute Error (left panel) and Percent Overshoot (right panel) bars plotted for each Control/Prediction Horizon Point, for batch 3886. Error bars denote the standard deviation.

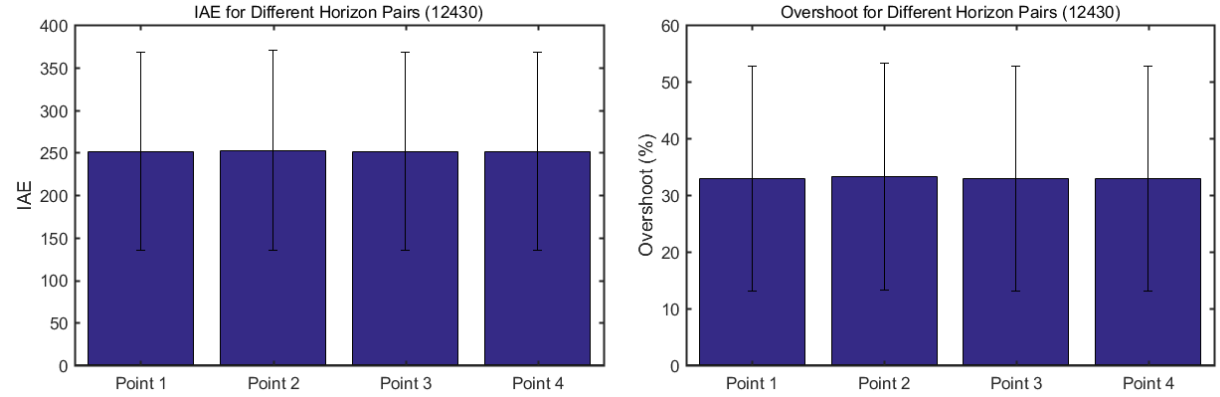

Figure 6.13: Integral Absolute Error (left panel) and Percent Overshoot (right panel) bars plotted for each Control/Prediction Horizon Point, for batch 12430. Error bars denote the standard deviation.

range of mixing speeds, from 100 to 675 rpm.

To check for the presence of an appreciable nonlinearity, step response experiments are performed in which the stirrer rate is varied in increments of 50 rpm from 100 to 650 rpm with the controller switched off while the dissolved oxygen is recorded. The results from two such experiments are shown in Figures 6.17 and 6.18.

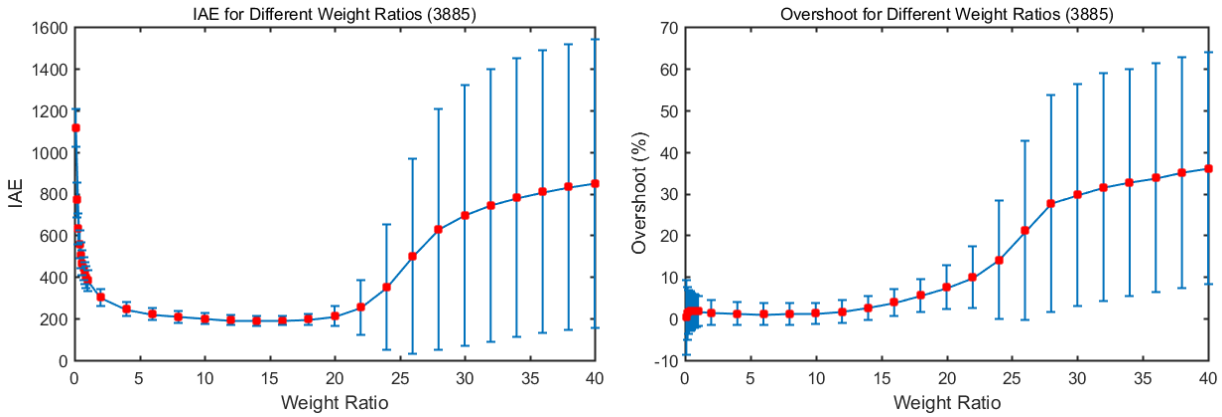

Figure 6.14: Averaged Integral Absolute Error (left panel) and Percent Overshoot (right panel) plotted against weight ratio for batch 3885. Errorbars represent standard deviation.

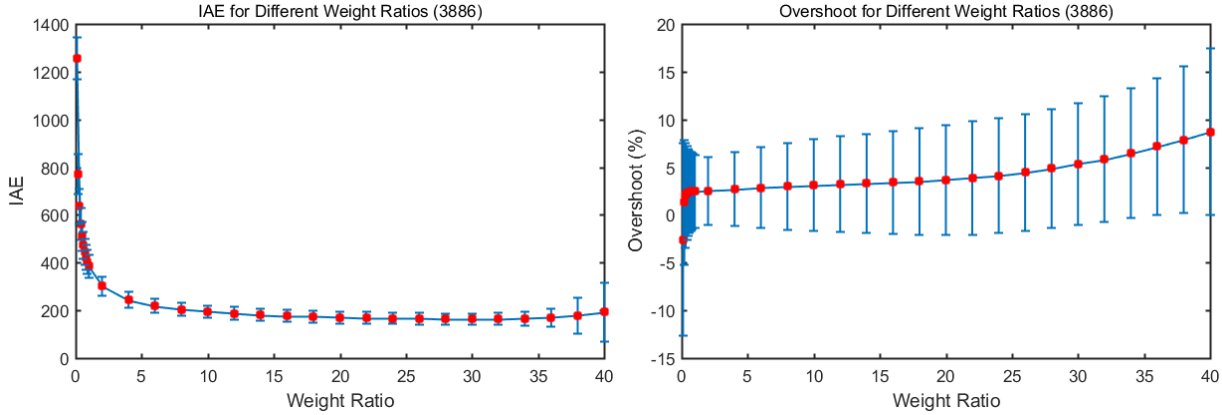

Figure 6.15: Averaged Integral Absolute Error (left panel) and Percent Overshoot (right panel) plotted against weight ratio for batch 3886. Errorbars represent standard deviation.

Both experimental recordings demonstrate extensive nonlinearity. Step changes in the stirrer rate from 300 to 600 rpm clearly show different gains in the dissolved oxygen level, even though all step changes are of the same magnitude (50 rpm). Moreover, for stirrer rates below 200 rpm, the response of the dissolved oxygen to step changes in the input is small and difficult to characterize. Finally, a step change in u from 200 to 250

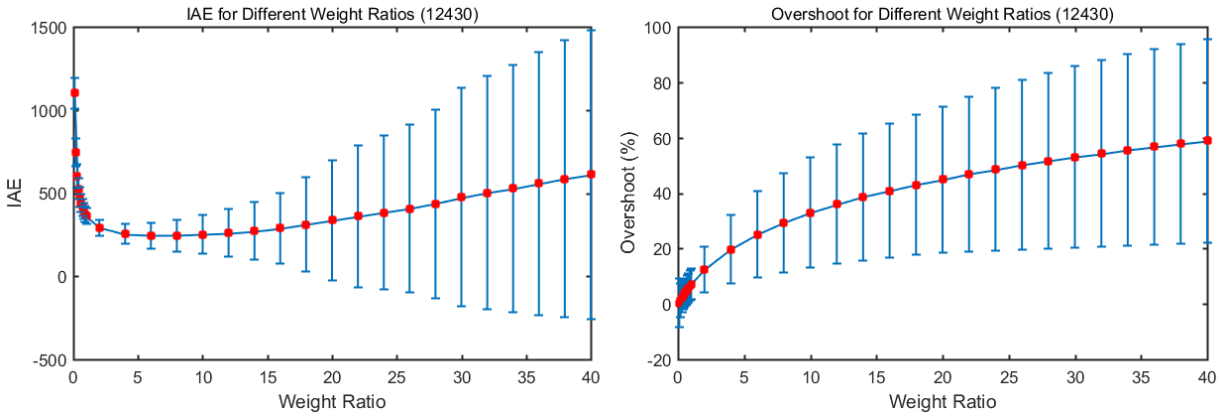

Figure 6.16: Averaged Integral Absolute Error (left panel) and Percent Overshoot (right panel) plotted against weight ratio for batch 12430. Errorbars represent standard deviation.

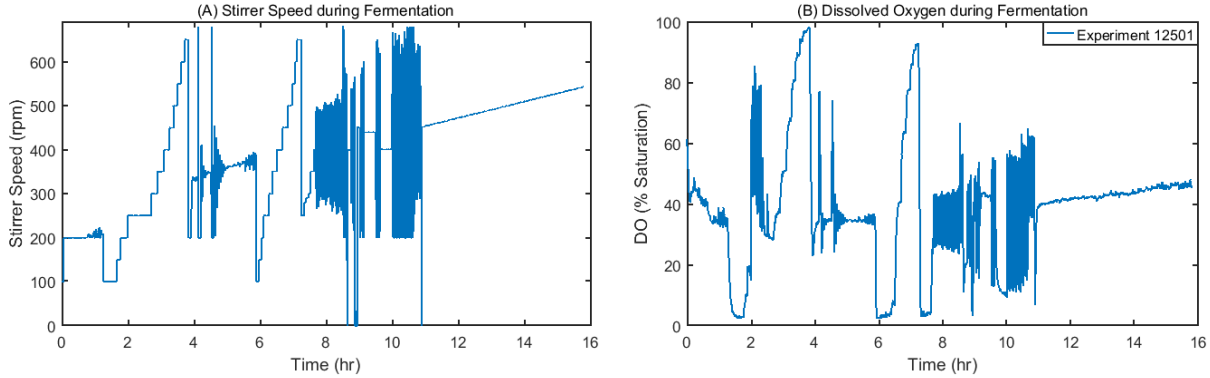

Figure 6.17: First step response experiment conducted over stirrer rates from 100 to 650 rpm, with stirrer rate changed in increments of 50 rpm.

rpm results in unstable oscillations in the dissolved oxygen content, a characteristic not observed for other stirrer speeds.

In order to account for the nonlinear behaviour inherent to the system, it is helpful to refine our tuning methods. This is done by implementing a gain scheduling method [45], in which we determine the 3 PID parameters (K_p, T_i, T_d) as functions of the stirrer rate u. For each range of 50 rpm, a best-fit process model G_p is determined (using the AIC and F-test tools detailed in chapter 3) and used to obtain the PID settings using an

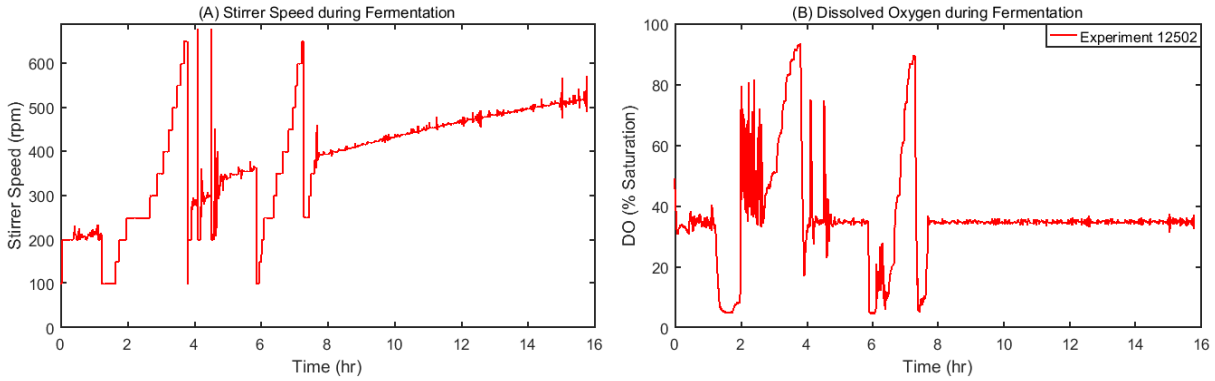

Figure 6.18: Second step response experiment conducted over stirrer rates from 100 to 650 rpm, with stirrer rate changed in increments of 50 rpm.

optimization algorithm similar to the one described in chapter 4. We will ignore stirrer speeds below 200 rpm in the determination of the PID settings, due to the lack of a DO response which can properly be described by our transfer function estimation techniques.

To determine a table of PID settings for each 50 rpm range in the stirrer rate, we isolated the open-loop input-output data from experiments 12501 and 12502 for the stirrer rate and dissolved oxygen from 300 to 600 rpm. In addition, we also isolated the open-loop input-output data from experiment 12501 for the first step response from 250 to 300 rpm. This resulted in 4 sets of input-output data for each 50 rpm input range (i.e. $[300, 350], [350, 400], [400, 450], [450, 500], [500, 550], [550, 600]$), and 1 set of input-output data corresponding to the range $[250, 300]$ rpm.

Overall, we obtained 25 sets of input-output data and estimated the best transfer functions G_p corresponding to each data set, using the techniques described in chapter 3. Specifically, we found the best-fit G_p for each input-output data set, calculated the AIC value for 5 possible transfer function fits (FOP, SOP, SOPLD, SOPTDD, SOPTDLD), and selected the model with the lowest AIC as the best-fit model.

When the 25 best-fit models were fully determined, the models were grouped according to the stirrer rate range they corresponded to. For each range, the group of models was then plugged into 1 or 4 Simulink block diagrams and the PID parameters were estimated by the 'fmincon' algorithm in MATLAB, which outputs PID settings that minimize the integral absolute error. This algorithm is slightly different from the nonlinear least-squares method used in the previous section, and was selected due to its ability to converge faster

Stirrer Rate Range (rpm)	Proportional Gain (K_p)	Integral Gain ($K_i = K_p/T_i$)	Derivative Gain ($K_d = K_pT_d$)
$[250, 300]$	0.45	0.00328	1.0
$[300, 350]$	1.49	0.00705	10
$[350, 400]$	0.67	0.00415	6.9
$[400, 450]$	0.67	0.00583	5.7
$[450, 500]$	1.09	0.0108	8.2
$[500, 550]$	0.56	0.0114	1.3
$[550, 600]$	0.62	0.0140	0.14

Table 6.2: Determining the PID settings by accounting for the nonlinearities in the process model.

and perform a global search of the PID parameters. Unlike the least-squares method, a global search allows a wider range of PID parameters to be sampled so that the true minimum is more likely to be reached. Using this technique, we determined the PID settings for all stirrer rate ranges from 250-600 rpm. The results are shown in Table 6.2.

From the table, we observe no significant, monotonic trend in the PID settings, but we have determined a set of values which could be used for each range of stirrer speeds. Thus, this table can serve as a basis should the need arise to take into account the nonlinearity of the system.

Chapter 7

Conclusions and Recommendations

At a fundamental level, the micro-organism fermenter system presents a classical process control problem: the variable to be controlled (dO_2) is unable to be properly regulated by the stirrer speed so alternative control strategies need to be devised to ensure adequate regulation. The purpose of this work was to test multiple control techniques, both practical and theoretical, in order to ensure that the dissolved oxygen in the fermenter was effectively maintained at desired levels. To achieve this goal, our industrial partners conducted step-response experiments which allowed the estimation of the process transfer function - the relation between the manipulated variable u and the controlled variable of interest dO_2.

Following the estimation of the process transfer functions using the results of 4 step-response experiments (identified as 3885, 3886, 12429, 12430), the first step was to determine a practical control strategy that would allow our industrial partners to meet their goals of removing the DO oscillations. The control strategy needed to be reproducible, applicable to a PID controller, and systematic, so that it could be employed for multiple systems that may exhibit different behaviour from one batch to another. To satisfy this objective, we used the foundation developed by Madhuranthakam et al [21] to create an algorithm which utilized the process transfer functions to calculate optimized PID settings based on least-squares minimization.

Our PID optimization algorithm provided settings that, when applied to the microbial experimental setup, were capable of controlling the dissolved oxygen levels adequately and ensuring that the DO levels do not oscillate violently. Given the success of our methodology in achieving the control objectives of our industrial partners, we set up MATLAB programs in order to implement a more automated strategy of determining the PID control

settings for any biological fermenter. With this automated strategy, the ideal PID settings for a given batch can be estimated during experimental operation automatically using the following steps:

1. Start the fermentation with the controller turned off and the stirrer rate fixed at 400 rpm. Allow the system to reach steady state (should take approximately 30 minutes).
2. Once at steady state, step up the stirrer rate automatically to 450 rpm and allow the system to reach steady state. Record and write the data of the step experiment to a csv file.
3. With MATLAB, estimate the best plant model G_p among the five models proposed in section 3, using a combination of the AIC and/or F-tests.
4. With the plant model determined, use the PID optimization algorithm to determine the best possible settings for that particular experimental batch. Write the PID settings to the csv file which is then read by the controller.
5. Implement the settings determined and turn on the controller. The system should reach the desired DO steady state without undergoing erratic behaviour.

So far, we have created MATLAB code that will automatically detect the presence of a csv file containing open-loop stirrer rate/dO_2 data, use that to estimate G_p, and use the G_p to find the optimal PID settings. Overall, the process is generic and easily implemented with the framework we have created, as long as the software for performing these actions automatically can be integrated with the fermenter system.

While the practical approaches to controlling dO_2 were demonstrated to be sound and effective, it was necessary to develop and test theoretical approaches to dO_2 control. To this end, we created internal model controllers for each of the four process models. These IMC controllers underwent extensive testing to find the optimal value of the closed-loop time constant τ_c - the lone tuning parameter - which ensured robust and precise control. The extensive testing included varying τ_c and determining the two main controller performance metrics - the integral absolute error and overshoot - as functions of τ_c for a particular IMC controller. Ultimately, we found that lower τ_c typically yielded the best results, and proposed optimal τ_c values for each of the four IMC controllers.

Another theoretical strategy that we developed for controlling the microbial fermentation was Model Predictive Control (MPC). This strategy is based on employing a discretized version of the process model G_p to generate a set of future control moves which

ensure that a cost function - based partly on the error between the desired set-point and the actual value of the controlled variable - was minimized. MPC Controllers for each of the four batches (3885, 3886, 12429, 12430) were generated and the tuning parameters (control horizon, prediction horizon, weight ratio) were varied for all but 12429 to find the optimal combination that would ensure robustness and good set-point tracking. Ultimately, we determined that a large weight ratio and large prediction/control horizons provided optimal set-point tracking, control, and robustness.

The last control technique used to regulated the dO_2 was based on a non-linear model of the fermentation process, in which the transfer function relationship between dO_2 and stirrer speed varied according to the range of stirrer speed which the system occupied. The nonlinearity was verified via experimental analysis and modelled in MATLAB, so that optimal PID settings for each stirrer rate range could be determined.

Overall, this thesis lays out a framework for a generic and satisfactory control strategy which has been demonstrated to work for the biological fermentation process. In addition, it proposes and implements multiple theoretical control strategies and maximizes the effectiveness of those strategies with respect to various controller performance metrics, including robustness, set-point tracking, and overshoot minimization. More analysis could be undertaken by examining and optimizing with respect to other control system metrics, such as frequency response and stability margins; however, the foundation laid by our work is sufficient enough to ensure that both our control objectives and our industrial partner's control objectives are met and will continue to be met in the future.

References

[1] S. Plotkin, "History of vaccination," Proc Natl Acad Sci U S A, vol. 111, pp. 12283-7, Aug 26 2014.

[2] B. Pulendran and R. Ahmed, "Immunological mechanisms of vaccination," Nat Immunol, vol. 12, pp. 509-17, Jun 2011.

[3] R. Ahmed and D. Gray, "Immunological memory and protective immunity: understanding their relation," Science, vol. 272, pp. 54-60, Apr 05 1996.

[4] S. Crotty and R. Ahmed, "Immunological memory in humans," Semin Immunol, vol. 16, pp. 197-203, Jun 2004.

[5] F. Sallusto, A. Lanzavecchia, K. Araki, and R. Ahmed, "From vaccines to memory and back," Immunity, vol. 33, pp. 451-63, Oct 29 2010.

[6] K. Murphy, P. Travers, M. Walport, and C. Janeway, Janeway's immunobiology, 8th ed. ed. London: Garland Science ; London : Taylor & Francis [distributor], 2012.

[7] L. Sompayrac, How the immune system works, 4th ed. ed. Oxford: Wiley-Blackwell, 2012.

[8] E. A. Belongia and A. L. Naleway, "Smallpox vaccine: the good, the bad, and the ugly," Clin Med Res, vol. 1, pp. 87-92, Apr 2003.

[9] L. E. Gallo-Ramirez, A. Nikolay, Y. Genzel, and U. Reichl, "Bioreactor concepts for cell culture-based viral vaccine production," Expert Rev Vaccines, vol. 14, pp. 1181-95, 2015.

[10] C. Sheridan, "The business of making vaccines," Nat Biotechnol, vol. 23, pp. 1359-66, Nov 2005.

[11] Antunes, R.; Gonzalez, V. A Production Model for Construction: A Theoretical Framework. Buildings 2015, 5, 209-228.

[12] A. I. Mees, Dynamics of feedback systems. Chichester: John Wiley, 1981.

[13] D. E. Seborg, Process dynamics and control, 3rd ed. Hoboken, N.J.: John Wiley & Sons, 2011.

[14] B. Roffel and B. H. Betlem, Process dynamics and control : modeling for control and prediction. Chichester: John Wiley, 2006.

[15] G. C. Goodwin, S. F. Graebe, and M. E. Salgado, Control system design. Upper Saddle River, NJ ; London: Prentice Hall, 2001.

[16] B. A. Ogunnaike and W. H. Ray, Process dynamics, modeling, and control. New York ; Oxford: Oxford University Press, 1994.

[17] P. C. Chau, Process control : a first course with MATLAB. Cambridge: Cambridge University Press, 2002.

[18] B. Vanavil, A. V. N. L. Anusha, M. Perumalsamy, and A. S. Rao, "Enhanced IMC-PID Controller Design with Lead-Lag Filter for Unstable and Integrating Processes with Time Delay," Chemical Engineering Communications, vol. 201, pp. 1468-1496, 2014.

[19] F. Garcia-Ochoa and E. Gomez, "Bioreactor scale-up and oxygen transfer rate in microbial processes: An overview," Biotechnology Advances, vol. 27, pp. 153-176, Mar-Apr 2009.

[20] H. Yagi and F. Yoshida, "Gas Absorption by Newtonian and Non-Newtonian Fluids in Sparged Agitated Vessels," Industrial & Engineering Chemistry Process Design and Development, vol. 14, pp. 488-493, 1975.

[21] C. R. Madhuranthakam, A. Elkamel, and H. Budman, "Optimal tuning of PID controllers for FOPTD, SOPTD and SOPTD with lead processes," Chemical Engineering and Processing, vol. 47, pp. 251-264, Feb 2008.

[22] C. E. Garcia and M. Morari, "Internal Model Control .1. A Unifying Review and Some New Results," Industrial & Engineering Chemistry Process Design and Development, vol. 21, pp. 308-323, 1982.

[23] E. F. Camacho and C. Bordons, Model predictive control, 2nd ed. ed. London: Springer, 2004.

[24] M. Fujasova, V. Linek, and T. Moucha, "Mass transfer correlations for multiple-impeller gas-liquid contactors. Analysis of the effect of axial dispersion in gas and liquid phases on "local" k(L)a values measured by the dynamic pressure method in individual stages of the vessel," Chemical Engineering Science, vol. 62, pp. 1650-1669, Mar 2007.

[25] W. E. Boyce and R. C. DiPrima, Elementary differential equations and boundary value problems, 9th ed. ed. Hoboken, N.J.: Wiley, 2010.

[26] J. R. Brannan and W. E. Boyce, Differential equations with boundary value problems : modern methods and applications, 2nd ed., International student ed. Hoboken, N.J.: Wiley, 2011.

[27] M. L. Shuler and F. Karg, Bioprocess engineering : basic concepts, 2nd ed. ed. Upper Saddle River, NJ: Prentice Hall, 2002.

[28] R. a. Shanmugam and R. a. Chattamvelli, Statistics for scientists and engineers: John Wiley & Sons, 2015.

[29] D. C. Montgomery, Design and analysis of experiments, 7th ed., international ed. ed. Hoboken: John Wiley & Sons, 2009.

[30] A. P. Field, J. Miles, and Z. Field, Discovering statistics using R. London: SAGE, 2012.

[31] H. Akaike, "Likelihood of a Model and Information Criteria," Journal of Econometrics, vol. 16, pp. 3-14, 1981.

[32] A. Papoulis, "Citation Classic - Probability, Random-Variables, and Stochastic-Processes," Current Contents/Engineering Technology & Applied Sciences, pp. 14-14, 1980.

[33] M. P. Norton and D. G. Karczub, Fundamentals of noise and vibration analysis for engineers, 2nd ed. ed. Cambridge: Cambridge University Press, 2003.

[34] G. F. Franklin, J. D. Powell, and A. Emami-Naeini, Feedback control of dynamic systems, 6th ed. Upper Saddle River, N.J. ; London: Pearson, 2010.

[35] S. T. Karris, Introduction to Simulink with Engineering Applications, 1st ed. Orchard Publications, 2006.

[36] J. G. Ziegler and N. B. Nichols, "Optimum Settings for Automatic Controllers," Journal of Dynamic Systems Measurement and Control-Transactions of the Asme, vol. 115, pp. 220-222, Jun 1993.

[37] G.H. Cohen, G.A. Coon, Theoretical consideration of related control, Trans. ASME 75 (1953) 827834.

[38] D. E. Rivera, M. Morari, and S. Skogestad, "Internal Model Control .4. Pid Controller-Design," Industrial & Engineering Chemistry Process Design and Development, vol. 25, pp. 252-265, Jan 1986.

[39] J. H. McClellan, R. W. Schafer, and M. A. Yoder, DSP first : a multimedia approach. Upper Saddle River, N.J.: Prentice Hall ; London : Prentice-Hall International (UK), 1998.

[40] H. T. Nguyen, O. Kaneko, and S. Yamamoto, "Data-driven IMC for Non-Minimum Phase Systems - Laguerre Expansion Approach," 2011 50th Ieee Conference on Decision and Control and European Control Conference (Cdc-Ecc), pp. 476-481, 2011.

[41] M. Morari, "Robust Process-Control," Abstracts of Papers of the American Chemical Society, vol. 197, pp. 57-Iaec, Apr 9 1989.

[42] G. A. Baker and J. L. Gammel, The Pade Approximant in Theoretical Physics: Academic Press, 1970.

[43] K. Zhou and J. C. Doyle, Essentials of robust control. London: Prentice-Hall International, 1998.

[44] D. C. Montgomery, Design and analysis of experiments, 7th ed., international ed. ed. Hoboken: John Wiley & Sons, 2009.

[45] W. J. Rugh and J. S. Shamma, "Research on gain scheduling," Automatica, vol. 36, pp. 1401-1425, Oct 2000.