The Dissertation Committee for Parham Pournazari
certifies that this is the approved version of the following dissertation:

Self-Learning Control of Automated Drilling Operations

Committee:

Eric van Oort, Supervisor

Benito R. Fernández, Co-Supervisor

Dongmei Chen

Ronald E. Barr

Scott Niekum

Self-Learning Control of Automated Drilling Operations

by

Parham Pournazari

DISSERTATION

Presented to the Faculty of the Graduate School of

The University of Texas at Austin

in Partial Fulfillment

of the Requirements

for the Degree of

DOCTOR OF PHILOSOPHY

THE UNIVERSITY OF TEXAS AT AUSTIN

May 2018

Dedicated to my parents, for all of their sacrifices, and my beautiful wife for her unwavering support.

Acknowledgments

To my advisor Prof. Eric van Oort for his continuous guidance and encouragement, and providing me with the freedom to explore new ideas in my Ph.D research. I am grateful for your mentorship, and your ability to inspire in times of uncertainty.

I would like to thank Dr. Pradeep Ashok for his valuable feedback and support, and providing me with the exposure to many exciting projects throughout my time as a graduate student. I would like to also mention Dr. Benito Fernández for his meaningful contribution to my research, through his teachings in the areas of nonlinear control and intelligent systems.

A special thank you to Dr. Mitch Pryor and Dr. Dongmei Chen, as well as the rest of the Drilling and Rig Automation Group for their valuable insights throughout my studies.

This research would not have been possible without the generous funding provided by the sponsors of the RAPID consortium. Thank you for supporting our efforts in conducting impactful drilling engineering research.

To my friends at the Drilling and Rig Automation Group, for all the fun times we had working on the nanoRig, visiting drillings rigs, discussing politics, and traveling abroad. These experiences shaped my time as graduate student at UT, and made the tough times a little more bearable.

To my family, for instilling in me the value of higher education, and their unconditional support throughout the many years of my schooling. I am forever grateful.

To Leylah, for never stopping to believe in me when I did not believe in myself. She even gave me her computer when mine died of overuse while typing this dissertation.

Self-Learning Control of Automated Drilling Operations

Publication No. _____

Parham Pournazari, Ph.D.
The University of Texas at Austin, 2018

Supervisor: Eric van Oort
Co-Supervisor: Benito R. Fernández

In recent years, drilling automation has sparked significant interest in both the upstream oil and gas industry and the drilling research community. Automation of various drilling tasks can potentially allow for higher operational efficiency, increased consistency, and reduced risk of trouble events. However, wide adoption of drilling automation has been slow. This can be primarily attributed to the complex nature of drilling, and the high variability in well types and rig specifications that prevent the deployment of off-the-shelf automation solutions. Such complexities justify the need for an automation system that can self-learn by interacting with the drilling environment to reduce uncertainty.

The aim of this dissertation is to determine how a drilling automation system can learn from the environment and utilize this learning to control drilling tasks optimally. To provide an answer, the importance of learning, as well as its limitations in dealing with challenges such as insufficient training data, are explored.

A self-learning control system is presented that addresses the aforementioned research question in the context of optimization, control, and event detection. By adopting an action-driven learning approach, the control system can learn the parameters that describe system dynamics. An action-driven approach is shown to also enable the learning of the relationship between control actions and user-defined performance metrics. The resulting knowledge of this learning process enables the system to make and execute optimal decisions without relying on simplifying assumptions that are often made in the drilling literature. Detection of trouble drilling events is explored, and methods for reduction of false/missed alarms are presented to minimize false interruptions of the drilling control system. The subcomponents of the self-learning control system are validated using simulated and actual field data from drilling operations to ascertain the effectiveness of the proposed methods.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

Drilling of onshore and offshore oil and gas wells is practiced with the aim of exploring for and extracting hydrocarbons to meet the world's growing energy demands. During recent years, the exploration of more challenging reservoirs such as high-pressure high-temperature, deep water, and those only accessible through horizontal drilling techniques, has initiated a growing interest in automated drilling solutions. Drilling efficiency and quality have been major drivers of drilling automation. With the recent decline in oil prices, it has become more desirable for oil and gas operators and drilling contractors to explore drilling automation to keep well construction costs as low as possible, and remain profitable. Furthermore, operational safety, just like in other industries such as mining and manufacturing, has been a key driver of automation in oil and gas, particularly drilling, due to inherently hazardous work conditions.

Despite promising value propositions, and the availability of rig site computational power and sensing technologies, wide adoption and particularly, successful implementation of holistic automation technologies has remained slow. This can be attributed to several factors. First, development and com-

missioning of an automation system on a drilling rig requires the integration of several off-the-shelf technologies, supplied by various vendors. Inevitably throughout this process, aggregation of data becomes a challenge, and the automation system has to often function with missing contextual information. Second, each process of well construction is unique on its own in terms of well design and geology, and therefore a commissioned rig automation system might not work properly on different wells. Finally, from a theoretical standpoint, the automation of a well construction process is challenging due to the high level of uncertainty involved in the estimation of states that are difficult if not impossible to measure, as well as slowly varying plant parameters within the environment. In essence, a lot of the contextual and real-time information required for successful implementation of a drilling automation system is often unavailable and/or inaccurate. This necessitates the automation system to have its own learning capability.

1.1 Background

Drilling of an oil and gas well is performed with the purpose of creating a wellbore, straight or directional, that is often several miles long to provide access to hydrocarbon information and reservoirs (note that some wells are drilled only for information purposes, e.g. most exploration wells). A drilling rig, as portrayed in Figure 1.1 consists of many sub-systems that together facilitate the drilling process. Drilling a hole consists of transferring rotary power to a drill-bit through a drillstring, which consists of several stands of

drill-pipes, drill-collars, and the bottom-hole-assembly (BHA). The drillstring is supported by the rig structure on surface by a hoisting device (the drawworks). To drill the wellbore, the driller controls two primary parameters from the surface. One is the rotary speed of the drill-bit (RPM) which is transmitted from a large motor (topdrive), also hoisted by the drawworks. The other parameter is the axial downward force, the weight-on-bit (WOB), which is exerted on the drill-bit to enable the normal force required for the cutting action that fails the rock formation. Weight-on-bit is simply achieved by the weight of the drill-pipes and the top-drive resting on the drill-bit, counterbalanced by an upward force that is provided by the drawworks. The hole is then drilled as the drill-bit cuts through the formation and generates small cuttings, all of which are transmitted back to the surface through circulation of drilling fluid in the drillstring and back up the wellbore through the annular space. The rate of penetration (ROP), is usually an unknown and varying function of the two drilling parameters WOB and RPM, and also depends on the type of drill-bit used, as well as the particular formation (sandstone, shale, limestone, granite, etc.) being drilled.

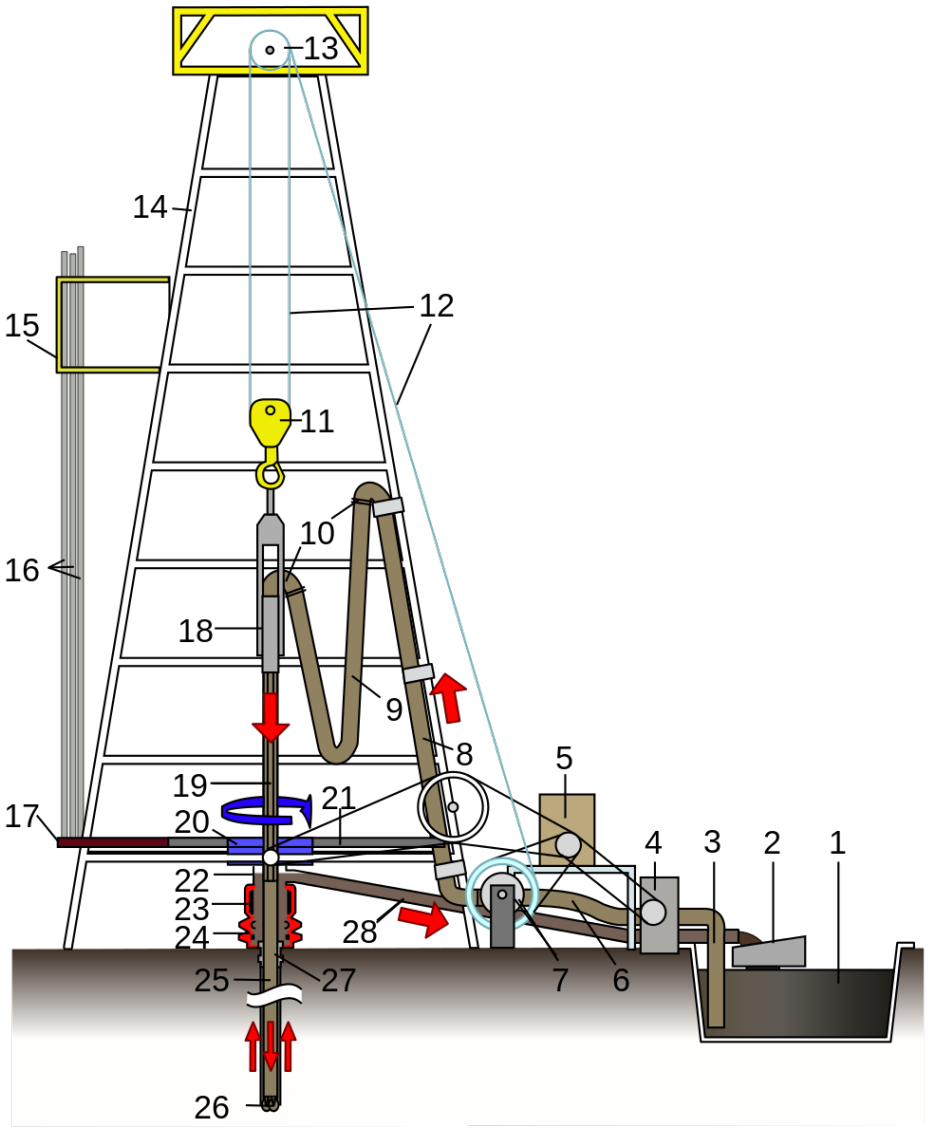

Figure 1.1: Diagram of an onshore drilling rig. The drillstring (25) hangs off the crown block (13), and the drawwork (7). While in this diagram the rotary power is provided by a rotary drive (19), most modern drilling rigs use a top drive, which hangs off the crown block as well. The swivel (18) enables circulation of drilling fluid through the drill-pipe as it rotates. The mud pumps (4) continuously circulate drilling mud from the mud tanks (1) through the standpipe (8) down the drillstring. The cuttings are transferred to the surface and separated from the drilling mud using the shale shakers (2).

The drilling fluid plays an important role in the entire drilling process. In addition to controlling the formation pressure, the drilling fluid helps to keep the drill-bit cool, and lift the rock cuttings back to the surface. The fluid is pumped from the mud tanks using several mud pumps, through the stand-pipe and down the drill-string. It then travels up the annulus and through the return line to the shale-shakers, where it is separated from the rock cuttings. The circulating pressure of the mud (ECD) should always remain in an allowable pressure window to avoid kick (when formation fluids flow into the wellbore) and lost circulation (when drilling fluid is lost to the formation) events, where the maximum of this window is dictated by the fracture gradient pressure, and the minimum by the formation pore pressure or the mud weight required for wellbore stability, whichever quantity is higher.

After the drilling process has progressed to a certain depth, the drill-string is pulled (tripped) back to the surface, in order to change the drill-bit or the BHA, or to run casing. An interesting phenomena is the interaction between axial drillstring dynamics and the drilling fluid when the pipe is being tripped in/out of the hole. This interaction, also referred to as the swab/surge effect, occurs when the movement of the pipe causes a so called "plunger" effect, where the frictional interaction between the drillstring and the drilling fluid causes transient pressure changes that can drastically affect the annular pressure.

While there are several automation areas that can contribute to the improvement of rig-site efficiency and safety, automation of the drilling hole

process and pipe tripping are desirable in the industry. Automated drilling involves surface inputs such as the top-drive rotary speed (RPM) and the WOB, to ensure the well is drilled as fast as possible without jeopardizing wellbore quality through mitigation of drilling dysfunctions. The goal for tripping is to trip the drillstring as fast as possible without going outside the allowable pressure constraints.

1.2 Research Question

This dissertation is devoted to answering the following question: *How can a drilling automation system learn about its environment, and control it optimally?* In developing a solution, we shall explain several essential aspects of this question.

- The *environment* refers to the physical aspects of the drilling system and more specifically, the drillstring dynamics and hydraulics of drilling fluid from a swab/surge perspective. The drillstring interacts with the wellbore through contact along the wellbore walls, as well as the rock-bit interaction. The hydraulics of the drilling fluid are affected by the dynamics of the drillstring through frictional contact along the drillstring's outer surface.

- A *drilling automation system* receives a set of performance indices from the user, most often regarding efficiency, quality and safety, and translates these into control inputs to the environment which are executed

through sensory feedback. In doing so, the automation system has to robustly detect trouble events with a low rate of missed and false alarms, to prevent reduced efficiency, as well as potentially severe and costly consequences.

- Given the definition and scope of the *environment*, learning then implies:
  - Uncovering the relationship between performance indices, including operational constraints, and control inputs. An example is the relationship between the rate of penetration (ROP) and weight-on-bit (WOB).
  - Uncovering the relationship between control inputs and the environment. An example is the relationship between drawworks feed-rate and the maximum downhole pressure change of the drilling fluid due to swab/surge effects.
- *Control* is concerned with ensuring set-points derived from performance indices are executed effectively. A major aspect of achieving acceptable control performance is related to the controller's understanding of the environment and its dynamics, which we aim to solve through learning. However, several other challenges remain. Learning in a real-world setting inevitably comes with uncertainty, and the controller has to therefore be robust to both process and measurement uncertainty. In addition, controlling downhole parameters such as WOB through actuation from several thousand meters away presents challenges regarding delay that

are difficult to address with simple control algorithms. Nonlinearity in system dynamics is another complication, which requires careful analysis and design of special control techniques that take this into account.

- *Optimality* in automation can refer to two control aspects:
  - Optimization of control inputs in a tracking control problem,
  - Optimization of control set-points in a performance maximization problem.

  Although we generally pay attention to the first idea when studying the control problem, in the context of drilling automation, we are primarily concerned with the latter. When the relationship between a cost function and the manipulated variables is available, either linear or nonlinear optimization techniques can be used to arrive at the optimal combination of the manipulated set. Depending on the complexity of this relationship, heuristic techniques can often be faster but at the expense of obtaining a sub-optimal solution. For a drilling automation system, the aforementioned relationship is often unknown and changing, hence classical techniques are not readily applicable.

1.3 Proposed Solution

The solution developed in this dissertation emphasizes the idea of *action-driven learning*, which in broad terms is its main contribution to the existing drilling automation literature. A graphical representation of this solution is

portrayed in Figure 1.2. In the depicted architecture, all aspects of the automation system benefit from various forms of learning to enhance their performance. Although some of this learning can come from historical data-sets from offset wells, a special emphasis is made on the knowledge that can be gained from the ongoing drilling process itself, through the actions taken by the automation system.

The main idea behind action-driven learning is that a drilling automation system, through calculated actions, actively selects the right conditions and learning actions to improve its understanding of the environment in various ways, rather than solely relying on the knowledge supplied by its designer. While most classical definitions of an automation system would consider the optimizer, the controller and the event detector as primary aspects of the design, the architecture shown in Figure 1.2 introduces a virtual learning agent, which becomes an integral part of the other three components. With the system focused on action-driven learning, the actions taken by the optimizer and the controller no longer strictly aim to maximize a performance index, but also serve to maximize learning, which in turn would improve the overall performance of the system.

The architecture in Figure 1.2 consists of several important subsystems. The *Learning Agent* is an integrated physics-based model along with the appropriate learning algorithms, that enable the system to use sensor measurements and capture important knowledge about the dynamics of the environment. The *Optimizer* determines the set-points that are assigned to process

level controllers, including the drawworks feed-velocity, the WOB, and the top-drive rotary speed. Instead of assuming full knowledge of its environment, the *Optimizer* constantly has to decide whether to exploit its incomplete knowledge to maximize user-defined performance indices, or whether to explore the environment to enhance its understanding, and reach a globally optimal combination of set-points. The *Controller* executes the set-points assigned by the *Optimizer*. With sufficient *a priori* information, as well as the learning enabled through the *Learning Agent*, the *Controller* uses surface measurements to control downhole drilling parameters. The *Event Detector* addresses the safety concerns for a drilling automation system and informs the drilling automation system about potential trouble events that require immediate attention.

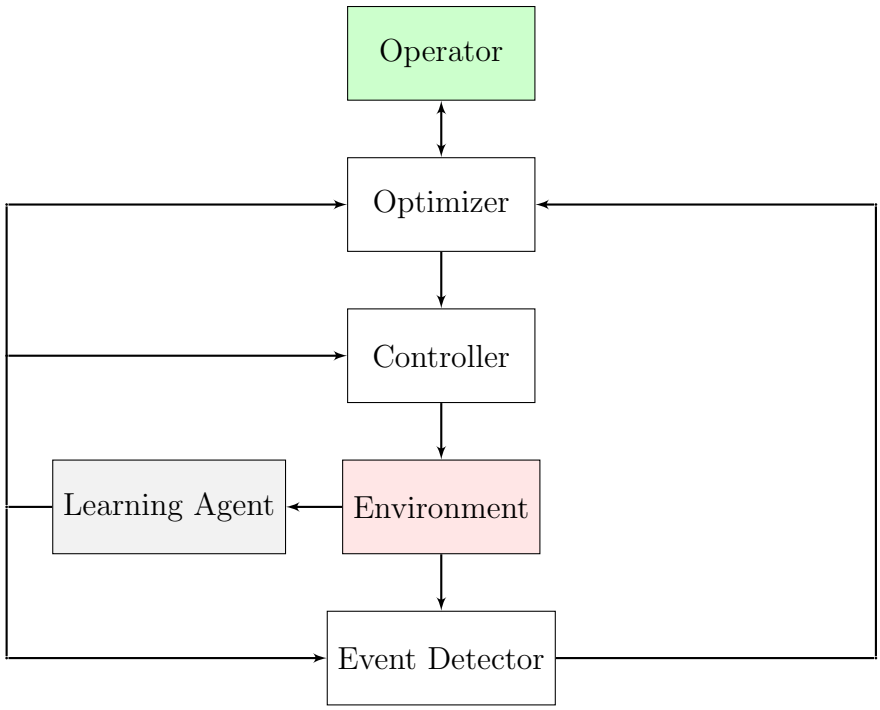

Figure 1.2: The architecture of the proposed self-learning control system. The connection between the *operator* and the *critic* represents the data exchange for performance index specification and operational status reporting.

1.4 Scope

The scope of this dissertation revolves around demonstrating the proposed solution on the processes of drilling the hole and pipe tripping. The tracking control problem focuses on tracking weight-on-bit (WOB) set-points assigned by the *Optimizer*, since control of bit RPM is primarily a vibration mitigation problem, which is passively dealt with by the *Optimizer* through avoidance of vibration inducing set-points.

The automation system's event detection and its ability to minimize

missed/false alarms focuses on kick and lost circulation events, which amongst others, are difficult to detect, can be costly, and have hazardous consequences.

1.5 Dissertation Outline

This dissertation is structured as follows:

- Chapter 2 focuses on the *Learning Agent*, and addresses the problem of learning a physics-based model from real-time data. The problem is targeted in a probabilistic fashion, and various learning actions and conditions are considered. The learning enabled by the approaches developed in this chapter aids: i) the *Optimizer* in Chapter 3 to predict the system's response to various set-points, and ensure constraint satisfaction ii) the *Controller* in Chapter 4 to perform model-based self-tuning, and estimate the unmeasurable states and output of the plant.

- Chapter 3 is focused on the *Optimizer*, and studies the problem of optimization without a complete *a priori* understanding of the environment. An iterative learning approach is proposed through comparison of two different formulations. We then explore various strategies that help the *Optimizer* deal with a changing environment. The chapter also includes a discussion on how the *Optimizer* can address risk while determining optimal set-points.

- Chapter 4 is devoted to the *Controller*, and explores techniques to best execute the decisions of the *Optimizer*. The chapter addresses the chal-

lenges of nonlinearity, uncertainty and latency that can affect the performance of the *Controller*. The *Controller* is implemented on a high-fidelity model to assess its handling of the aforementioned challenges.

- Chapter 5 presents the *Event Detector*, and argues the case that the optimality of the drilling automation system is largely dependent on its ability to avoid reacting to false alarms, while appropriately detecting costly and hazardous events that require the attention of the operator. Such a system is therefore required to have strong baseline performance, and also leverage real-time learning to improve itself. Tools for robust change identification, pattern recognition, and event classification are presented. Data from various field events are used to validate the presented approach.

Chapter 2

Learning Agent: Action-Driven Learning of Physics-Based Drilling Models

In this chapter, we formulate the *Learning Agent*, and explore the problem of learning the parameters of physics-based drilling models in real-time using appropriate actions. The goal of this chapter is to show that it is possible to sufficiently learn a physics-based model of the drilling process which can later aid the *Optimizer* in making decisions, and the *Controller* in executing these decisions effectively. To remain consistent with the scope of this dissertation, physics-based modeling will focus on an integrated model of the drilling and tripping processes through their coupling in the axial domain (along the direction of the drillstring). Suitable probabilistic learning algorithms are discussed and applied to the developed physics-based model. The learning of certain model parameters is discussed in the context of optimal conditions and actions that enhance the sequential learning task.

2.1 Introduction

A physics-based model of the drilling process is concerned with predicting the behavior of the drill-bit and the drillstring in response to surface inputs,

namely the drawworks axial speed and the top-drive rotary speed. The axial and torsional domains are coupled through rock-bit interaction, when the axial force applied on the bit (WOB) produces a torque on the bit (TOB). During a tripping process, axial drillstring dynamics are coupled with drilling fluid hydraulics, through frictional interaction between the annulus walls and the drilling fluid, as well as the dynamics of changing volumes. Section 2.2 of this chapter is devoted to establishing this model.

Based on the scope of this dissertation, we are interested in using the physics-based model for two purposes: i) estimation of the velocities and forces within the drillstring components to determine downhole WOB when drilling, ii) downhole fluid pressure changes induced by an axially moving drillstring when tripping. After the functional form of the physics-based model is established, the problem of interest is learning the parameters of this model. While there are several approaches that can be taken towards the parameter learning problem, a probabilistic approach is desirable as it enables uncertainty quantification for the learned parameters. Probabilistic parameter estimation is handled well by a dynamic Bayesian approach, which will be discussed in detail later in this chapter.

Learning the parameters of a physics-based model has been a problem of interest in the drilling automation community and the subject of active research. However, the sequential learning of several parameters of a model is a challenging task, and only possible when the parameter learning task is conducted with the right conditions and appropriate system actions. This is

the focus of the latter part of this chapter, where the appropriate learning conditions are discussed.

2.1.1 Literature Review

2.1.1.1 Modeling of Drillstring Dynamics

From a drillstring dynamics perspective, earlier studies focused on drillstring axial and torsional vibrations using a continuous beam model and solving the wave equation analytically (Bailey, 1960). These earlier studies ignored the effects of lateral vibrations as well as the interaction between wellbore and drillstring in inclined wells (Shor et al., 2014). Finite Element Models (FEM) began to appear in the vibrations domain as deviated, horizontal, and extended reach wells became widespread. A paper by Heisig studied the natural frequencies of the drillstring in lateral sections using a FEM model (Heisig et al., 2000).

Dunayevsky et al. (1984) and Skaugen et al. (1987) proposed a model that took into account the coupling between axial and lateral vibrations, and Skaugen et al. (1987) analyzed the significance of rock-bit interaction in understanding frequency domain torsional vibrations. Lee presented a comprehensive investigation of axial vibrations while considering effects of hydraulic damping as well as friction forces in inclined wells (Skaugen et al., 1987). A transfer matrix approach was used to solve the frequency domain response of the drillstring. More advanced FEM models such as the work by Hu et al. (2012) enabled a more thorough analysis of the drillstring dynamics by con-

sidering the special curvature of the drillstring for an inclined well as well as the strain and bending within each drillstring element.

With the increased interest in automatic control of drilling operations as well as real-time analysis and event detection, research was undertaken into the development of control-oriented models of drilling processes by simplifying the previously developed models through certain assumptions. In particular, decoupling and discretizing mathematical models was shown to enable faster computation time and analysis for controller design. In the vibrations domain for instance, Navarro-López and Cortés (2007) demonstrated that an n^{th} order discretization of the wave equation, which was previously used for frequency domain analysis can be used to design a sliding mode controller for active stick-slip mitigation. Lumped-parameter models therefore, became popular for real-time analysis and control of drilling dynamics, and became the basis for development of several control methodologies such as fuzzy, neural network, H-infinity and LMI controllers (Dashevskiy et al., 1999; Yilmaz et al., 2013; Harris et al., 2014).

2.1.1.2 Modeling of Drilling Hydraulics

In the hydraulics domain, researchers began studying the problem of predicting downhole pressure variations in response to a moving drillstring during pipe tripping operations in 1961. Burkhardt et al. (1961) set the industry standard at the time by modeling the steady-state pressure changes for both open-ended and closed-ended pipes in non-Newtonian drilling fluids.

A dynamic extension of the work was also presented that took into account the effects of pipe acceleration on the downhole pressure drop (Burkhardt et al., 1961). A later study focused on analyzing frictional pressure drops as a function of pipe movement for yield-power-law drilling fluids (Crespo et al., 2012). Mitchell proposed a more elaborate hydraulics model of the tripping process by exploring pressure dynamics and the effects of fluid compressibility and wellbore elasticity (Mitchell et al., 1988). The one dimensional, partial-differential equations were solved using a finite difference scheme, and it was shown that pressure dynamics are in fact critical in understanding and predicting the transient downhole pressures during pipe tripping. To gain an understanding of the effects of various drillstring-wellbore geometries, more elaborate 2-dimensional models such as the one by Chin et al. (2011) were developed. These models considered the effects of drillstring eccentricity as well as pipe rotation on downhole pressure fluctuations. Several studies were published that tested the validity of such models through laboratory and field experimentation (Samuel et al., 2003).

With the emergence of managed pressure drilling (MPD) technology, hydraulics modeling was extended into the multi-phase domain to accurately capture the dynamics of both liquid and gas within the annulus. These models ranged from general two-phase CFD models to a more simplified drift-flux model to describe the interaction between and liquid and gas particles (Lage et al., 2000; Aarsnes et al., 2014). Active control of downhole pressure during MPD subject to heave disturbances has been of particular interest. An

early study in this area was presented by Kaasa et al. (2012) which showed that pressure dynamics can be captured by writing the mass and momentum balance equations for two control volumes and a controller can be designed accordingly. Papers by Landet (2011) and Gjerstad et al. (2013) extended this methodology by demonstrating that several control volumes are required to sufficiently capture higher frequency pressure oscillations. Gjerstad et al. (2013) also developed simplified explicit equations for the frictional forces in laminar flow of Herschel Bulkley fluids, which were used along with the lumped-parameter model for controller development. Exploration of lumped-parameter models with higher number of control volumes led to the investigation of hydraulic transmission-line models. These models were essentially an infinite-dimensional representation of the lumped-parameter models, which allowed for the modal analysis of the pressure dynamics in the frequency domain. In a study by Aarsnes et al. (2012), it was shown that the accuracy of the lumped-parameter models drops as the number of control volumes used for discretization is decreased. However, it was also demonstrated that depending on the length of the wellbore, a lumped-parameter model with 20 control volumes might be sufficiently accurate.

2.1.1.3 Model Estimation

The problem of state and parameter estimation of dynamical systems has been an active area of research in the engineering community for several decades. With the advent of modern control theory, researchers have focused

on deriving optimal estimation, and filtering techniques that allow the estimation of unknown states of a dynamical system when the full state is unmeasurable. The simplest form of an estimation problem can be formulated as a least square optimization problem, which aims to minimize the estimated process and measurement disturbances in a least square sense. The optimal solution to this least squares problem is used to compute the state estimate, given the output measurements. The approach, however, is a non-sequential or batch estimation technique, in which it requires the solution of a least squares problem using all previous output measurements. The least square recursive estimator (LSRE) can also be formulated by the minimization of the mean square reconstruction error, which is the difference between the actual and estimated state (Muske and Edgar, 1997). The LSRE gives the minimum-variance estimate of the state, if the process and sensor noise are independent, normally distributed random variables with covariance Q and R. For linear Gaussian systems, this technique has also been shown to produce the most probable or maximum likelihood estimate (Muske and Edgar, 1997). This recursive estimator is referred to as the discrete Kalman filter, which provides an optimal estimation of the states in the presence of sensor and process noise. The filter gain is computed at each sampling time using the covariance of the state estimate. Another possible approach is to use the recursive form of the batch state estimation problem by formulating a moving horizon approach, where the state estimate at time k is determined recursively from the solution of the least squares problem. This is done by using the predicted estimate at time $k - n - 1$ and the

most recent $n+1$ output measurements (Muske and Edgar, 1997). However, since the moving horizon produces the same estimates as the Kalman filter, there is no incentive to use it due to additional required computational effort. One benefit of the moving horizon estimator is that it allows the addition of constraints on the estimated states and the disturbances. These constraints can be applied to prevent physically unrealistic state estimates.

For nonlinear systems, additional complexity is introduced to the state estimation problem, as the same optimal estimation framework is generally not available for nonlinear systems. The optimal linear filter produces the minimum variance and maximum likelihood estimate of the state of the linear system. To develop a nonlinear analogy, one must first specify which estimate is desired, since they may not be the same for a nonlinear stochastic system. The minimum variance estimate is the conditional mean of the state. The maximum likelihood or Bayesian estimate is the conditional mode or most probable estimate. These estimates are determined from the conditional probability density of the state given the measurement. The conditional probability density of a linear system with Gaussian noise is Gaussian, whereas for a nonlinear system it is not Gaussian even when the measurement and disturbances are.

A simple approximation to the optimal nonlinear state estimation problem is to linearize the nonlinear model about a given operating point and apply optimal linear state estimation to the linearized system. The extended Kalman filter computes a state estimate at each sampling time by the use of

the Kalman filtering on a linearized model of the nonlinear system. This technique is justified if there exists a large region around the operating point in which the linearized model is a good representation of the nonlinear system. In addition, if the disturbances are well represented by a zero mean Gaussian distribution, then the optimal estimate for the linearized system should also be a reasonable approximation for the nonlinear system. In that scenario, the extended Kalman filter is expected to produce an accurate estimate of the state. A linearized model of the continuous system can be developed from the Taylor series expansion of the system model and ignoring the higher order terms. However, this covariance matrix can become a poor estimate, because of the approximations made in the propagating the covariance matrix in the EKF (Muske and Edgar, 1997).

To overcome the limitations of the EKF, the unscented Kalman filter (UKF) was proposed using the concept of sample statistics. The UKF uses a deterministic sampling technique to select a minimal set of sample points around the current estimate (Wan and Van Der Merwe, 2000). These sample points (called the sigma points), form a set of points that lie on the covariance contour. These points are then propagated through the nonlinear system dynamics to compute a cloud of transformed points. The main idea behind this approach is that it is easier to approximate a probability distribution than an arbitrary nonlinear function. It has been demonstrated that the UKF results in approximations that are accurate to the third order for all types of nonlinearities. For non-Gaussian uncertainties, approximations are accurate to at least

the second moment. A major advantage of UKF is that it avoids the explicit computation of the Jacobian matrices (Wan and Van Der Merwe, 2000). The cubature Kalman filter (CKF) was introduced by Arasaratnam and Haykin (2009) as an improvement to the UKF, which is claimed to suffer from curse of dimensionality and/or estimate divergence for high dimensional systems. The cubature Kalman filter uses a spherical-radial cubature rule, which allows the numerical computation of multivariate moment integrals encountered in the nonlinear Bayesian filter.

To avoid the approximations made with non-Gaussian uncertainties, particle filtering can be used to deal with state estimation problems arising from multi-modal, and non-Gaussian distributions (Johannes and Polson, 2007). A particle filter approximates multi-dimensional integration involved in propagation and update steps of the estimation problem using Monte Carlo sampling. The Ensemble Kalman filter, proposed by Evensen (2003), is an example of a particle filtering technique. The filter is initialized by drawing N particles from a suitable distribution. At each time step, N samples for sensor and process noise are drawn using the distributions of measurement noise, and state estimates. Similar to the UKF, this approach uses only the 1^{st} and 2^{nd} order moments generated using ensemble integration, to estimate the Kalman gain. The accuracy of the estimates therefore depends on the number of data points. While particle filtering can, in theory, deal with all probability distributions of state, Daum and Huang (2003) points two major pitfalls, with particle filters: 1) the algorithm suffers from curse of dimensionality like most

other nonlinear filters developed under Bayesian framework, 2) if we end up having a poor proposal density, then not all samples will be useful.

A limitation of the filtering algorithms discussed above is that they can't handle constraints systematically (Prakash et al., 2011). Nonlinear dynamic data reconciliation, and moving horizon estimation formulations, provide systematic approaches to handle bounds on states/parameters, or any other algebraic constraints. The MHE problem is formulated as a constrained nonlinear optimization problem defined over a moving time. This is, however, difficult to perform in real-time since it requires large dimensional nonlinear optimization to be solved. Zavala et al. (2008) proposed advanced step MHE formulation, which performs the nonlinear dynamic optimization in the background and requires little online computation to update state and parameter estimates. Vachhani et al. (2006) proposed a recursive constrained formulation called recursive nonlinear dynamic data reconciliation. This approach combines the advantages of recursive estimation while handling the constraints on the variables. Vachhani et al. (2005) proposed a constrained version of the UKF for state and parameter estimation in nonlinear systems. For constrained particle filtering, Lang et al. (2007) proposed a version of sequential importance sampling particle filter that can handle constraints. This approach is similar to using a truncated distribution to satisfy the constraints, which ensures that the posterior also satisfies the constraints. Prakash et al. (2010) proposed using the constrained UKF to generate the distribution required on particle filtering, and to deal with the requirement of generating particles con-

sistent with the bounds.

Parameter estimation, also referred to as system identification, is the problem of determining a nonlinear mapping from the input to the output. For dynamical state space systems, a problem of interest is estimating both the unobserved state as well as the model parameters which is referred to as the dual estimation problem. A common approach to the dual estimation problem is to have a separate state space representation for the states and the parameters. Two Kalman filters are then run simultaneously for state and parameter estimation (Wan et al., 1999). At every time step, the current estimate of the parameters is used in the state filter as a given input. In a similar manner, the current estimate of the state is used in the parameter filter. Another approach is referred to as the joint filtering problem, where the unknown system state and parameters are concentrated into a single higher dimensional state vector. A single filter is then run on the joint vector to produce simultaneous estimates of the states and the parameters (Wan et al., 1999). In the joint and dual Bayesian approaches to solving parameter estimation problems, it had been shown that the non-dynamics in the unknown parameters often cause the degeneracy of the algorithm as the parameter space is only explored at the initialization of the algorithm. Therefore, these algorithms are sometimes inefficient and after a few iterations the marginal posterior distribution of the parameter is approximated by a single delta Dirac function (Andrieu and Doucet, 2003).

From a non-Bayesian perspective, online expectation-maximization type

algorithms have also been effective for the parameter estimation problem. Such algorithms are primarily different from the Bayesian approach in that they do not attempt to estimate the model covariance. In addition, EM algorithms do not require a priori knowledge about the system (Ghahramani and Roweis, 1999). In the EM algorithm, the conditional expectation of the signal is computed, given the data and the current estimate of the model (E-step). Then a model is found that maximizes a function of this conditional mean (M-step). For linear models, the M step can be solved in closed form and the E step is computed with a Kalman smoother. However, for a nonlinear model the M step can no longer be computed in closed-from and therefore a gradient-based approach is used instead. The E step is usually approximated by an EKF where a linearization of the model is used for backward propagation of the state estimates (Ghahramani and Roweis, 1999). Wan et al. (1999) improved the E step of the EM algorithm for nonlinear models by using an unscented Kalman filter instead of an EKF to compute the forward and backward passes in the Kalman smoother.

Within the oil and gas drilling space, state and parameter estimation of drilling models have been approached with different techniques. To estimate the bulk modulus in a hydraulics model, Kaasa et al. (2012) used a recursive least squares technique. Gravdal et al. (2005) used the unscented Kalman filter for calibration of friction factors in a hydraulics model. Pixton et al. (2014) employed the moving horizon estimator for calibrating friction factor and density in the annulus. Nonlinear adaptive observers were designed to estimate

the friction factor and density (Stamnes et al., 2011). Aamo (2013) used a back-stepping transformation observer design for estimating the states of a transmission line model. Nikoofard et al. (2015) used an unscented Kalman filter to tune the parameters of a drift flux model in under-balanced drilling. Gjerstad et al. (2013) employed an ensemble Kalman filter to calibrate the friction parameters of a lumped parameter model for drilling, using measurement data during pipe tripping with circulation.

2.1.1.4 Summary of Literature Review

Most of the physics-based models in the literature used for automated drilling systems are deterministic, and fail to capture the uncertainties involved in the drilling process and sensor measurements. Most of the parameters of a deterministic drilling model are either unknown to begin with, or slowly vary throughout the drilling operation. Although tuning of model parameters has been practiced by several researches, a holistic approach to learning real-time dynamical models is missing in the drilling literature. This becomes especially important when such models are used for real-time control and optimization, where inaccurate estimations can result in suboptimal performance and non-productive time. Therefore, a learning approach is needed that has the capability to re-evaluate its environment, and make probabilistic estimations of the environment's operating conditions through the use of special actions in specific conditions.

2.2 An Integrated Mathematical Model of the Drilling and Tripping Processes

This section presents a dynamic mathematical model of the drillstring and its behavior when subject to a velocity input from the surface. In the context of automated tripping, the pressure fluctuations of the drilling fluid are also of importance, and the transient model is extended to capture this pressure behavior. From an estimation problem perspective, this model forms the transition function used for the prediction step of the filtering problem. The estimation of this model's parameters are thus the purpose of the learning task.

2.2.1 Axial Drillstring Dynamics

Figure 2.1 portrays a mechanical diagram of the axial drillstring dynamics. The model is driven by a velocity input on the surface, v_{in} using a drawworks system. It is assumed that dynamics of the traveling block are considered and handled by a stiff controller within the drawworks' variable frequency drive (VFD). Drill-pipes are represented using a beam element with stiffness k_{ds}, and inertia m_{ds}. Similarly, drill-collars and the bottom hole assembly are represented by stiffness k_c, and inertia m_c. The entire drillstring is subject to nonlinear damping Ψ, which is caused by the friction between the drillstring and the wellbore, as well as linear damping, b due to fluid interaction and material hysteresis. Drill-bit and its interaction with the rock formation is captured by another spring element, k_{fr}, and v_{ROP} represents the velocity

boundary condition which is a function of the applied weight-on-bit, angular velocity Ω, and various bit and formation parameters. Although several existing models in the literature attempt to empirically capture this relationship, in the context of feedback-control we treat this term as a disturbance, and estimate it in real-time from measurements.

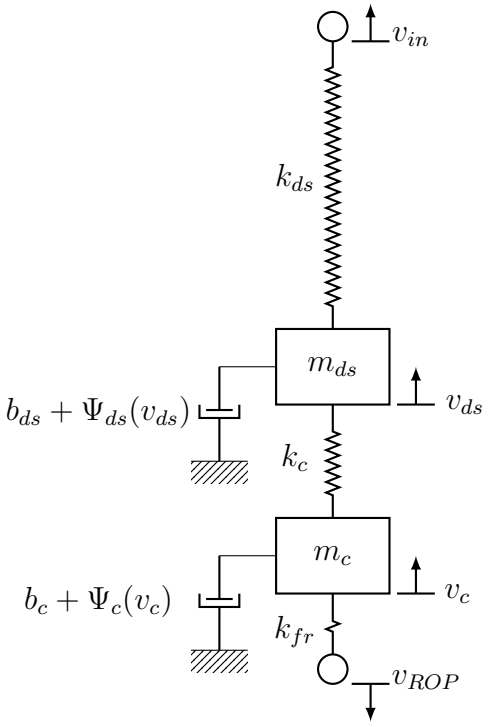

Figure 2.1: A representation of axial drillstring dynamics using an equivalent mass-spring-damper system. The sources of damping have linear and nonlinear terms, which arise from the contact with the wellbore and the drilling fluid.

The simplest model representing drill-pipe dynamics can be written for two-lumped elements:

$$\dot{F}_{ds} = k_{ds}(v_{in} - v_{ds}) \tag{2.1}$$

$$\dot{v}_{ds} = \frac{1}{m_{ds}}\big(F_{ds} - F_c - b_{ds}v_{ds} - \Psi_{ds}(v_{ds})\big) \tag{2.2}$$

For improved model accuracy, higher order dynamics can be captured by dividing the drill-pipe section into higher number of elements, which in matrix form can be written as:

$$\dot{\mathbf{x}} = A_n\mathbf{x} + B_n\mathbf{u} \tag{2.3}$$

$$\mathbf{x} = \begin{bmatrix} F_1 \\ v_1 \\ \vdots \\ F_n \\ v_n \end{bmatrix}, \mathbf{u} = \begin{bmatrix} v_{in} & \Psi_{ds} & F_c \end{bmatrix}^T, \tag{2.4}$$

$$A_n = \begin{bmatrix} 0 & -k_n & 0 & \dots & 0 \\ \frac{1}{m_n} & -\frac{b_n}{m_n} & -\frac{1}{m_n} & \dots & 0 \\ 0 & k_n & 0 & -k_n & \dots \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ \dots & \dots & \dots & \dots \frac{1}{m_n} & -\frac{b_n}{m_n} \end{bmatrix}, B_n = \begin{bmatrix} k_n & 0 & 0 \\ \vdots & \vdots & \vdots \\ 0 & -\frac{1}{m_n} & -\frac{1}{m_n} \end{bmatrix}, \tag{2.5}$$

where the nonlinear damping term, Ψ_{ds} is approximated as a lumped source term, and is given by:

$$\Psi_{ds} = \frac{|v_{ds}|}{v_{ds}} \sum_{i=1}^{N} \mu\beta_B w L_i \mathbf{v}_{n,i} \tag{2.6}$$

where μ, β_B, w correspond to the coefficient of friction, coefficient of buoyancy and unit drill-pipe weight respectively, and $\mathbf{v}_{n,i}$ is a vector in the null space of the unit vector $\hat{\mathbf{v}}_{d,i}$ defined by:

$$\hat{\mathbf{v}}_{d,i} = \frac{(h_{i+1} - h_i), (z_{i+1} - z_i)}{\sqrt{(h_{i+1} - h_i)^2 + (z_{i+1} - z_i)^2}} \tag{2.7}$$

With the drillstring usually distributed over several thousand meters it is important to assess the accuracy of using a lumped-parameter approach for prediction, especially the drill-pipe section. For this assessment, a transmission line approach can be used, which provides an exact solution for fully-distributed input-output dynamics. The force and velocity distribution along the drill-pipe is given by:

$$\frac{\partial v}{\partial t} = \frac{1}{\bar{m}}\frac{\partial F}{\partial z}, \qquad \frac{\partial v}{\partial z} = \frac{1}{\bar{k}}\frac{\partial F}{\partial t}, \tag{2.8}$$

where $\bar{m}$ represents the mass of the drill-pipe per unit length, and $\bar{k}$ is stiffness of the drill-pipe times length. For simplicity, the damping coefficient is neglected when solving for the solution of Equation 2.8, and will be treated as a boundary condition in this model (Ma and Chen, 2015). The solution of the coupled partial differential equations in 2.8 can then be found in the frequency domain. For the force at the top of the drillstring, we have:

$$F(s,0) = sech(Ts)F(s,L_{ds}) + Z_{ds}\ tanh(Ts)v(s,0) \tag{2.9}$$

where $T = \sqrt{\frac{\bar{m}}{\bar{k}}}L$, $Z_{ds} = \sqrt{\bar{m}\bar{k}}$ and $F(s,L_{ds})$ represents the reaction force at the end of the drill-pipe section. The solution for the velocity at the end of the drill-pipe section is given by:

$$v(s,L_{ds}) = -\frac{1}{Z_{ds}}tanh(Ts)F(s,L_{ds}) + sech(Ts)v(s,0) \tag{2.10}$$

With $sech(s)$ defined as $\frac{2}{e^s+e^{-s}}$, $tanh(s) = \frac{e^{2s}-1}{e^{2s}+1}$, and noting the Laplace relationship $\mathscr{L}^{-1}\{e^{-as}F(s)\} = u_a(t)F(t-a)$, the time-domain solution of Equations 2.9 and 2.10 can be written as time-delay equations such that:

$$F(t,0) = 2F(t-T,L_{ds}) + Z_{ds}\big(v(t,0) - v(t-2T,0)\big) - F(t-2T,0) \tag{2.11}$$

$$v(t, L_{ds}) = 2v(t-T, 0) - \frac{1}{Z_{ds}}\big(F(t, L_{ds}) - F(t-2T, L_{ds})\big) - v(t-2T, L_{ds}) \quad (2.12)$$

To account for the linear damping, b_{ds} is divided in half, $r_{ds} = b_{ds}/2$ and added to the boundaries of Equations 2.11 and 2.12 so that (Ma and Chen, 2015):

$$F_{ds}(t) = 2F_c(t-T) - F_{ds}(t-2T) + (Z_{ds}+r_{ds})v_{in}(t) + (r_{ds}-Z_{ds})v_{in}(t-2T) + 2r_{ds}v_{ds}(t-T) \quad (2.13)$$

$$v_{ds}(t) = \frac{r_{ds}}{Z_{ds}+1}\left(2v_{in}(t-T) + \frac{r_{ds}v_{ds}(t-2T)}{Z_{ds}-1} - \frac{F_c(t) - F_c(t-2T)}{Z_{ds}}\right) \quad (2.14)$$

In Figure 2.2, we demonstrate a comparison of drill-pipe step response prediction of different order models. As shown, lower order representations of the drill-string do not accurately capture the transient behavior well, while also ignoring the initial delay associated with v_{ds}. In the comparison shown, the 18^{th} order model provides the closest approximation of the initial step delay.

Due to the much shorter length of the drill-collars and the bottom-hole assembly, v_c and F_c can be written using a lumped model approximation such that:

$$\dot{F}_c = k_c(v_{ds} - v_c) \quad (2.15)$$

$$\dot{v}_c = \frac{1}{m_c}\big(F_c - F_{fr} - b_c v_c - \Psi_c(v_c)\big) \quad (2.16)$$

The contact force with the formation can finally be written as:

$$\dot{F}_{fr} = k_{fr}(v_c - v_{ROP}) \quad (2.17)$$

In another simulation, the ability of various order models in approximating drill-collar velocity is highlighted. As shown in Figure 2.3, all lumped approximations of v_c are over-estimates of the true transient velocity, due to reaction force F_c being non-zero. The higher order approximations, however, can capture the initial delay adequately. While higher order approximations of the the drill-pipe dynamics can be used in the construction of the state estimator, it is important to use the exact model for testing the ability of the learning algorithm to estimate model parameters, as well as in controller performance assessment.

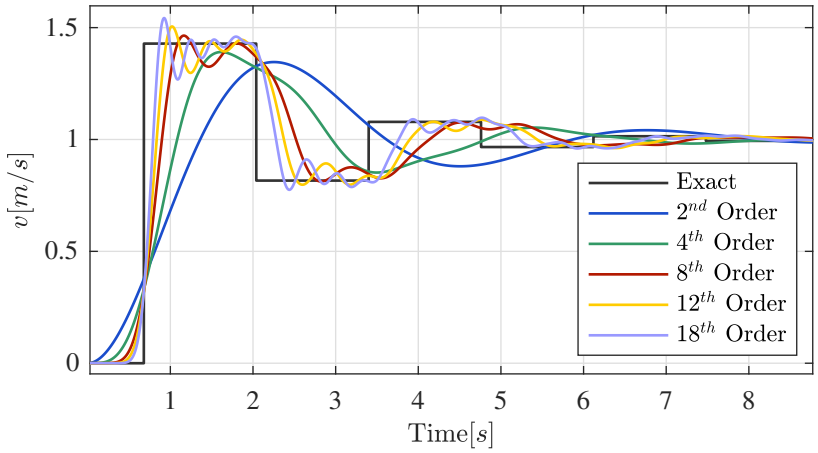

Figure 2.2: v_{ds}-step response comparison of the exact and lumped-parameter models. The reaction force, F_c, is zero.

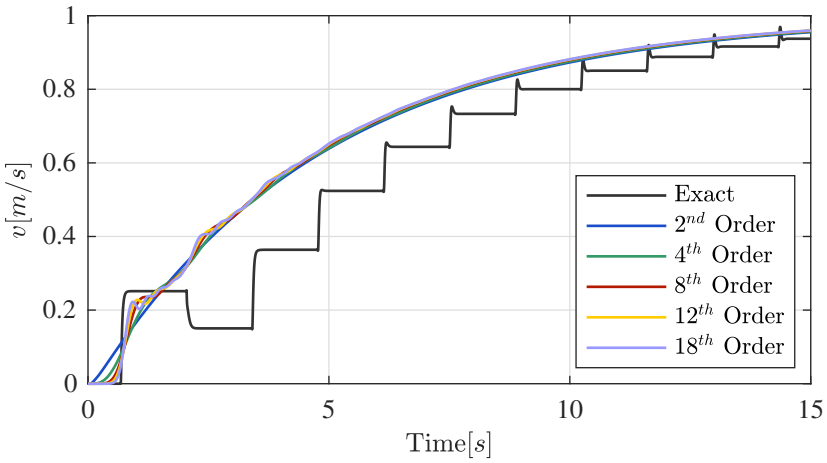

Figure 2.3: v_c-step response comparison of the exact and lumped-parameter models. The reaction force, F_c, is non-zero.

2.2.2 Drilling Fluid Hydraulics

In the following section, the previously developed model of the axial drillstring dynamics is extended to include the pressure and flow dynamics of the drilling mud when the drill-string moves in and out of the wellbore. This extension will provide the ability to predict allowable drill-string speeds while tripping, to avoid unsafe swab and surge pressures. We establish a lumped-parameter model that can be used for real-time learning, similar to the one presented in (Gjerstad et al., 2013). The performance of this model is compared to the results of a well-known infinite-dimensional model by Mitchell, as well as published field test data (Mitchell et al., 1988). Finally, we demonstrate that the integrated model presented in this dissertation can provide more realistic estimations of downhole pressure in tripping operations, due its

consideration of drillstring elasticity and damping.

To establish a hydraulic model of fluid flow, the drill-string and the annulus are broken down into 5 control volumes as shown in Figure 2.4. These control volumes enable the capturing of important cross-sectional changes in the annulus. Transient pressure changes are captured by considering the compressibility of the drilling fluid K. The pressure dynamics for the five control volumes can be written as:

$$\dot{P}_1 = \frac{K}{V_1}\big(Q_{bit} - Q_1 - v_c A_c\big) \tag{2.18}$$

$$\dot{P}_2 = \frac{K}{V_2}\big(Q_1 - Q_2 - v_{ds} A_{ds} + v_c A_c\big) \tag{2.19}$$

$$\dot{P}_{3,4} = \frac{K}{V_{3,4}}\big(Q_{2,3} - Q_{3,4}\big) \tag{2.20}$$

$$\dot{P}_{ds} = \frac{K}{V_{ds}}\big(Q_{in} - Q_{bit}\big) \tag{2.21}$$

The flow-rate through the drill-bit can be written using the quasi-steady relationship:

$$Q_{bit} = C_z A_z \sqrt{2(P_{ds} - P_1)/\rho} \tag{2.22}$$

For the case of mud-motors, the flow-rate can be calculated using steady, empirical relationships given by the manufacturer. The flow-rate through each control volume can be written as a function of the total hydraulic force at the ends of the control volumes:

$$\dot{Q}_n = \frac{1}{H_n}\big(F_n - F_{n+1} - F_f\big) \tag{2.23}$$

where $H_n = \frac{m_{f,n}}{A_n}$, $F_f = P_f A_n$, and P_f is the frictional pressure drop in each control volume. The equations for frictional pressure drop inside the drillstring and the annulus for Bingham Plastic fluids are summarized in Appendix B. Gjerstad et al. (2013) also recommends modifying F_n by a correction term as a function of P_f, in order to account for the effect of the pressure difference created due to the moving cross-sectional area.

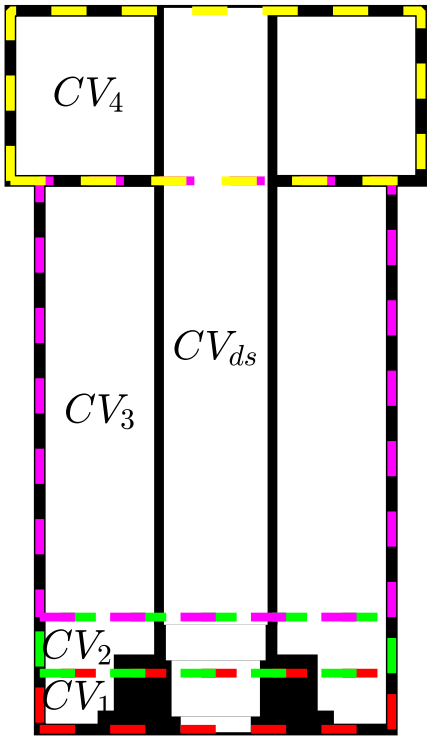

Figure 2.4: A schematic of the control volumes used for capturing various cross-sectional area changes in the annulus. Since the pressure dynamics inside the drillstring are not as important as the annulus, only one control volume is used in the calculations.

Figure 2.6 shows a comparison of simulation and field test results of a pressure surge scenario with a 7-inch moving pipe inside a 9.625-inch casing, with water-based mud of plastic viscosity $12cp$, and yield point 7 $lbf/100ft^2$. The velocity profile of the drillstring during the tripping process is shown in Figure 2.5. The lumped model results, which represent the model discussed above, demonstrate an excellent match with both the infinite-dimensional model of Mitchell, and the field test data. Despite the use of a finite control volume approximation, the lumped parameter approach can sufficiently capture the transient pressure surges captured in the field data. An important advantage of this model compared to that of Mitchell's is that it is based on ordinary-differential equations, and can therefore be used in the context of our filtering problem.

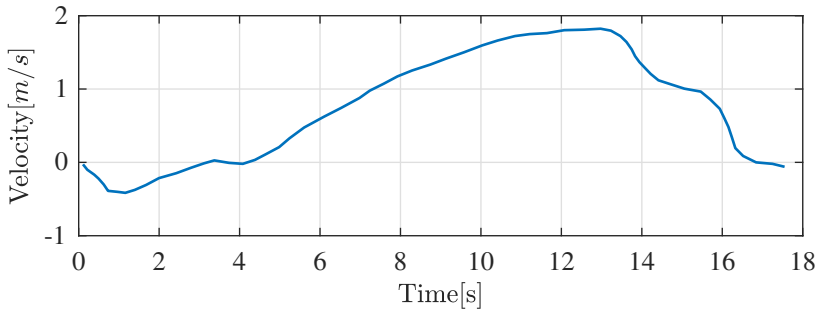

Figure 2.5: Velocity profile of the tripping process as published in Mitchell et al. 1988.

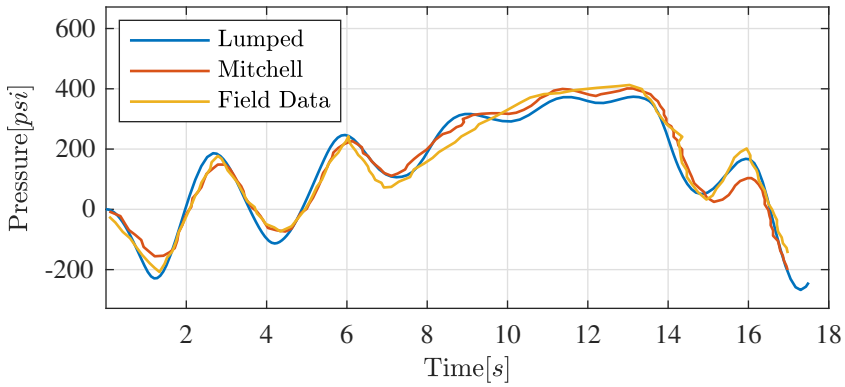

Figure 2.6: Comparison of downhole pressure change estimates by the lumped and Mitchell's PDE model, with field data published in Mitchell et al. 1988.

Another simulation is shown in Figure 2.7, where the effect of considering drillstring dynamics on pressure surges is demonstrated. As shown, the elasticity of the string, as well as the frictional damping from the wellbore, can alter the transient pressure surge behavior significantly. While in the demonstrated scenario the impact of drillstring dynamics reduces the pressure surge as compared to a rigid model, with an underdamped drillstring, the elasticity effect could result in higher transient pressure surges. This highlights the importance of an integrated dynamics-hydraulics model as presented in this section, and suggests that prediction of optimal string speeds in tripping operations require the use of an integrated model to avoid over/under estimating dynamic pressure changes and achieve true optimality.

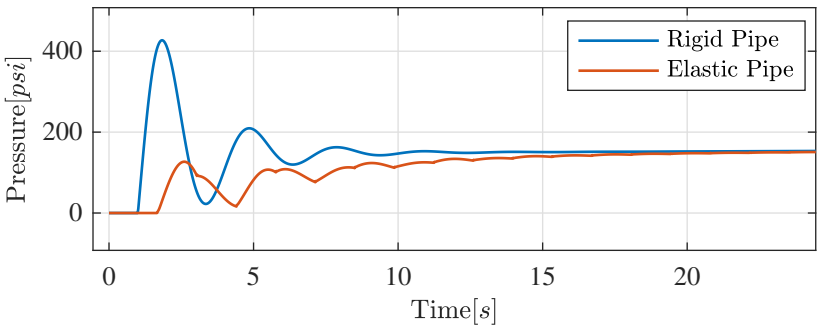

Figure 2.7: Comparison of downhole pressure dynamics while tripping with and without the effects of drillstring dynamics.

2.3 Probabilistic State and Parameter Estimation

In the previous section, a physics-based model of the drillstring behavior in drilling and tripping operations was established. The presented model based on ordinary differential equations will be the subject of parameter and state estimation in the following section. In the first part of this section, we introduce the Bayesian filtering problem. Various Kalman filtering techniques are explored afterwards as a solution to the filtering problem, and the nonlinear variations of the Kalman filter are shown. A Monte Carlo solution of the filtering problem, known as the particle filter, is also studied and compared to Kalman filtering techniques in terms of its applicability to the model of interest. We then shift our focus to the idea of action driven learning, and study the optimal actions/conditions that enable the learning task.

2.3.1 Recursive Filtering Algorithms

From a mathematical perspective, the purpose of recursive filtering is to estimate the hidden states and parameters of system, at each step, given a likelihood using real-time measurements, and a prior using model estimates. More precisely, the purpose is to compute the marginal posterior distribution of the state $\mathbf{x}_t$ at each time step t given the measurement $\mathbf{z}_t$. Two assumptions are made here: (i) the states follow a first-order Markov process $p(\mathbf{x}_t|\mathbf{x}_{0:t-1}) = p(\mathbf{x}_t|\mathbf{x}_{t-1})$ (ii) the measurements are independent of the states. Let Z_t denote the collection of measurements $\mathbf{y}_{0:t} := \{\mathbf{y}_0, ...\mathbf{y}_t\}$, and $p(\mathbf{x}_t|Z_t)$ the condition probability distribution function of $\mathbf{x}_t$. From Baye's rule it can be obtained (Chen et al., 2003):

$$\begin{aligned} p(\mathbf{x}_t|Z_t) &= \frac{p(Z_t|\mathbf{x}_t)p(\mathbf{x}_t)}{p(Z_t)} \\ &= \frac{p(\mathbf{z}_t, Z_{t-1}|\mathbf{x}_t)p(\mathbf{x}_t)}{p(\mathbf{z}_t, Z_{t-1})} \\ &= \frac{p(\mathbf{z}_t|Z_{t-1}, \mathbf{x}_t)p(Z_{t-1}|\mathbf{x}_t)p(\mathbf{x}_t)}{p(\mathbf{z}_t|Z_{t-1})p(Z_{t-1})} \\ &= \frac{p(\mathbf{z}_t|Z_{t-1}, \mathbf{x}_t)p(\mathbf{x}_t|Z_{t-1})p(Z_{t-1})p(\mathbf{x}_t)}{p(\mathbf{z}_t|Z_{t-1})p(Z_{t-1})p(\mathbf{x}_t)} \\ &= \frac{p(\mathbf{z}_t|\mathbf{x}_t)p(\mathbf{x}_t|Z_{t-1})}{p(\mathbf{z}_t|Z_{t-1})} \end{aligned} \tag{2.24}$$

As evident in Equation 2.24, the posterior density $p(\mathbf{x}_n|Z_n)$ is described by three important terms:

- The prior $p(\mathbf{x}_t|Z_{t-1})$ which is the knowledge from our model obtained

by:

$$p(\mathbf{x}_t|Z_{t-1}) = \int p(\mathbf{x}_t|\mathbf{x}_{t-1})p(\mathbf{x}_{t-1}|Z_{t-1})d\mathbf{x}_{t-1} \tag{2.25}$$

where $p(\mathbf{x}_t|\mathbf{x}_{t-1})$ is the transition density of the state,

- The likelihood $p(\mathbf{z}_t|\mathbf{x}_t)$ which essentially describes the measurement noise model,

- The evidence which involves the integral:

$$p(\mathbf{z}_t|Z_{t-1}) = \int p(\mathbf{z}_t|\mathbf{x}_t)p(\mathbf{x}_t|Z_{t-1})d\mathbf{x}_t. \tag{2.26}$$

The purpose of recursive Bayesian estimation in dynamical systems is to calculate/approximate the terms shown above. From an optimality perspective, the optimal Bayesian filter in the context of this dissertation is one with minimum mean-squared error (MMSE) which can be written as:

$$\mathbb{E}[||\mathbf{x}_t - \hat{\mathbf{x}}_t||^2|\mathbf{z}_t] = \int ||\mathbf{x}_t - \hat{\mathbf{x}}_t||^2 p(\mathbf{x}_t|\mathbf{y}_{0:t})d\mathbf{x}_t \tag{2.27}$$

2.3.1.1 Kalman Filter Based Algorithms

The closed-form solution of the filtering problem outlined in Equation 2.24 can be obtained in the special case of a linear transition function, and where the process and measurement are affected by random Gaussian processes. Specifically, this closed-form solution is the well-known Kalman filter. The Kalman filter is an iterative prediction-correction process, where an estimate of the next time step state is obtained in the prediction step, and an update is made to this estimate using the arriving measurement in the correction

step. For the case of linear systems with Gaussian process and measurement noise, the Kalman filter is the maximum a posteriori solution to the Bayesian filter problem. For a closer look, in the prediction step of the algorithm, the *a priori* state and covariance estimates are obtained by:

$$\begin{aligned} \hat{\mathbf{x}}_{t|t-1} &= \mathbf{F}_t \mathbf{x}_{t-1|t-1} + \mathbf{B}_t \mathbf{u}_t \\ \mathbf{P}_{t|t-1} &= \mathbf{F}_t \mathbf{P}_{t-1|t-1} \mathbf{F}^T + \mathbf{Q}_t \end{aligned} \tag{2.28}$$

where $\mathbf{F}$ is the state transition matrix, and $\mathbf{B}$ is the input matrix. After a measurement arrives, the algorithm performs a correction to $\hat{\mathbf{x}}_t$ and $\mathbf{P}_t$. This is done by calculating the innovation residual $\tilde{\mathbf{y}}_t$, and the innovation covariance, $\mathbf{S}_t$, and calculating the optimal Kalman gain as a function of the innovation covariance and the *a priori* state covariance.

$$\begin{aligned} \tilde{\mathbf{y}}_t &= \mathbf{z}_t - \mathbf{H}_t \hat{\mathbf{x}}_{t|t-1} \\ \mathbf{S}_t &= \mathbf{R}_t + \mathbf{H}_t \mathbf{P}_{t|t-1} \mathbf{H}_t^T \\ \mathbf{K}_t &= \mathbf{P}_{t|t-1} \mathbf{H}_t^T \mathbf{S}_t^{-1} \end{aligned} \tag{2.29}$$

The correction to the state and the covariance is then made such that:

$$\begin{aligned} \hat{\mathbf{x}}_{t|t} &= \hat{\mathbf{x}}_{t|t-1} + \mathbf{K}_t \tilde{\mathbf{y}}_t \\ \mathbf{P}_{tt} &= \mathbf{P}_{t|t-1} - \mathbf{K}_t \mathbf{H}_t \mathbf{P}_{t|t-1} \end{aligned} \tag{2.30}$$

The Extended Kalman Filter For the purpose of our filtering problem however, the use of the Kalman filter is limited by the nonlinearity of the system. To remedy this problem, one can approximate time-propagation of the covariance using a Taylor series expansion of the state transition and measurement functions. This is the core idea behind the Extended Kalman filter.

In the prediction step of the Extended Kalman filter, the mean of the state vector is computed using the non-linear transition function such that (Chen et al., 2003):

$$\begin{aligned} \hat{\mathbf{x}}_t &= \mathbf{f}(\mathbf{x}_{t-1}) + \mathbf{g}(\mathbf{u}_t) \\ \mathbf{P}_t &= \hat{\mathbf{F}}_t \mathbf{P}_{t-1} \hat{\mathbf{F}}_t^T + \mathbf{Q}_t \end{aligned} \tag{2.31}$$

where

$$\hat{\mathbf{F}}_t = \frac{d\mathbf{f}(\mathbf{x})}{d\mathbf{x}}\Big|_{\mathbf{x}=\hat{\mathbf{x}}_t} \tag{2.32}$$

In the correction step, the innovation and the optimal gain are computed similar to the regular case, such that:

$$\begin{aligned} \tilde{\mathbf{y}}_t &= \mathbf{z}_t - \hat{\mathbf{H}}_t \hat{\mathbf{x}}_t \\ \mathbf{S}_t &= \mathbf{R}_t + \hat{\mathbf{H}}_t \mathbf{P}_t \hat{\mathbf{H}}_t^T \\ \mathbf{K}_t &= \mathbf{P}_t \hat{\mathbf{H}}_t^T \mathbf{S}_t^{-1} \end{aligned} \tag{2.33}$$

where,

$$\hat{\mathbf{H}}_t = \frac{d\mathbf{h}(\mathbf{x})}{d\mathbf{x}}\Big|_{\mathbf{x}=\hat{\mathbf{x}}_t} \tag{2.34}$$

While the Extended Kalman filter allows one to apply the Kalman filter approach to non-linear systems, it still poses many limitations. The main limitation concerning the filtering problem in this dissertation is that it requires the transition and measurement functions to be differentiable. Secondly, computation and programming of the Jacobian case, where the transition function involves regression-based models, is error prone and impractical.

The Unscented Kalman Filter Another relatively recent approach to applying Kalman filtering to nonlinear problems is based on the idea of un-

scented transformation (UT) (Wan and Van Der Merwe, 2000). The Unscented Kalman filter shies away from linearization of the state and measurement functions, by directly trying to approximate the mean and covariance of the target distribution. The philosophy behind UT is to deterministically select a fixed number of sigma points that capture the mean and covariance of the original distribution of $\mathbf{x}$, such that:

$$\begin{aligned}
\chi_{t-1}^{(0)} &= \mathbf{x}_{t-1} \\
\chi_{t-1}^{(i)} &= \mathbf{x}_{t-1} + \sqrt{L+\lambda}\left[\sqrt{\mathbf{P}_{t-1}}\right]_i \\
\chi_{t-1}^{(i+L)} &= \mathbf{x}_{t-1} - \sqrt{L+\lambda}\left[\sqrt{\mathbf{P}_{t-1}}\right]_i, i = 1...L
\end{aligned} \tag{2.35}$$

In the prediction step of the Unscented Kalman filter, the sigma points are propagated through the nonlinear transition function:

$$\chi_t^{(i)} = \mathbf{f}(\chi_{t-1}^{(i)}), i = 1...2L \tag{2.36}$$

After propagation of the sigma points, the mean of the resulting distribution, and its covariance are computed by:

$$\begin{aligned}
\bar{\mathbf{x}}_t &= \sum_{i=0}^{2L} W_i^{(m)} \hat{\chi}_t^i \\
\bar{\mathbf{P}}_t &= \sum_{i=0}^{2L} W_i^{(c)} (\hat{\chi}_t^{(i)} - \bar{\mathbf{x}}_t)(\hat{\chi}_t^{(i)} - \bar{\mathbf{x}}_t)^T + \mathbf{Q}_{t-1}
\end{aligned} \tag{2.37}$$

where $W_i^{(m)}$ and $W_i^{(c)}$ are constant weights given by:

$$\begin{aligned}
W_0^{(m)} &= \frac{\lambda}{L+\lambda} \\
W_0^{(c)} &= \frac{\lambda}{L+\lambda} + (1 - \alpha^2 + \beta) \\
W_i^{(m)} &= W_i^{(c)} = \frac{1}{2(L+\lambda)}
\end{aligned} \tag{2.38}$$

where β is an additional parameter that can be used to incorporate prior information about non-Gaussian distribution of $\mathbf{x}$. In the update step of the kalman filter, a new set of sigma points are formed by:

$$\begin{aligned}\bar{\chi}_t^{(0)} &= \bar{\mathbf{x}}_t \\ \bar{\chi}_t^{(i)} &= \bar{\mathbf{x}}_t + \sqrt{L+\lambda}\left[\sqrt{\bar{\mathbf{P}}_t}\right]_i \\ \bar{\chi}_t^{(i+L)} &= \bar{\mathbf{x}}_t - \sqrt{L+\lambda}\left[\sqrt{\bar{\mathbf{P}}_t}\right]_i, i = 1...L\end{aligned} \tag{2.39}$$

and propagated through the measurement model such that:

$$\hat{\xi}_t^{(i)} = \mathbf{h}(\bar{\chi}_t^{(i)}), i = 1...2L \tag{2.40}$$

The mean and covariance of the innovation are then computed using a similar approach to the state vector:

$$\begin{aligned}\boldsymbol{\mu}_t &= \sum_{i=0}^{2L} W_i^{(m)} \hat{\xi}_t^{(i)} \\ \mathbf{S}_t &= \sum_{i=0}^{2L} W_i^{(c)} (\hat{\xi}_t^{(i)} - \boldsymbol{\mu}_t)(\hat{\xi}_t^{(i)} - \boldsymbol{\mu}_t)^T + \mathbf{R}_k \\ \mathbf{C}_t &= \sum_{i=0}^{2L} W_i^{(c)} (\bar{\chi}_t^{(i)} - \bar{\mathbf{x}}_t)(\bar{\xi}_t^{(i)} - \boldsymbol{\mu}_t)^T\end{aligned} \tag{2.41}$$

which can be finally used to perform the state and covariance correction through calculation of the optimal Kalman gain:

$$\begin{aligned}\mathbf{K}_t &= \mathbf{C}_t \mathbf{S}_t^{-1} \\ \mathbf{x}_t &= \bar{\mathbf{x}}_t + \mathbf{K}_t[\mathbf{z}_t - \boldsymbol{\mu}_t] \\ \mathbf{P}_t &= \bar{\mathbf{P}}_t - \mathbf{K}_t \mathbf{S}_t \mathbf{K}_t^T\end{aligned} \tag{2.42}$$

The advantage of the Unscented Kalman filter over the Extended Kalman filter is that it can capture higher order moments caused by the non-linear transform

better than Taylor series (Wan and Van Der Merwe, 2000). However, the covariance computation is only exact for first order polynomials. Also, despite slightly higher computation cost, overall implementation is easier as it only requires the knowledge of the exact transition and measurement functions.

2.3.1.2 Particle Filtering

The broadest class of Bayesian filtering algorithms are known as particle filters, where no assumptions regarding linearity or Gaussianity of the problem are made. The main idea behind these algorithms is to use Monte Carlo sampling to approximate the posterior distribution of state. The most basic example is the sequential importance sampling technique (SIS), where the posterior distribution at time $t-1$, $p(\mathbf{x}_{0:t-1}|\mathbf{z}_{1:t-1})$ is approximated with a weighted set of samples $\{\mathbf{x}^i_{0:t-1}, w^i_{t-1}\}^N_{i=1}$, and these samples are recursively updated to get an approximation of the posterior distribution of $p(\mathbf{x}_{0:t}|\mathbf{z}_{1:t})$ (Chen et al., 2003). Although the idea is similar to the case of the UT, the number of required samples, or particles, are much greater in the case of particle filters. SIS is further based on importance sampling, where the target distribution $p(x)$ is approximated using samples drawn from a proposal distribution $q(x)$. To account for the difference between the target and proposal distributions, one has to weight every sample by x^i by $w_i \propto \pi(x^i)/q(x^i)$, where $\pi(x)$ is a function that is proportional to $p(x)$. Applying this concept to the filtering problem, we obtain:

$$p(\mathbf{x}_{0:t-1}|\mathbf{z}_{1:t-1}) \approx \sum_{i=1}^{N} w^i_{t-1}\delta_{\mathbf{x}^i_{0:t-1}} \tag{2.43}$$

where $\delta_{\mathbf{x}^i_{0:t-1}}$ is a delta function centered at $\mathbf{x}^i_{0:t-1}$. The most important task in particle filtering is updating the particles and particle weights, such that they approximate the posterior distribution at the next time step (Chen et al., 2003). To do so, a proposal distribution can be written as:

$$q(\mathbf{x}_{0:t}|\mathbf{z}_{1:t}) = q(\mathbf{x}_t|\mathbf{x}_{0:t-1}, \mathbf{z}_{1:t})q(\mathbf{x}_{0:t-1}|\mathbf{z}_{1:t-1}) \tag{2.44}$$

so that each particle can be augmented with a new step at time step t sampled from $q(\mathbf{x}_t|\mathbf{x}_{0:t-1}, \mathbf{z}_{1:t})$. As noted, the weights are updated such that:

$$w^i_t \propto \frac{p(\mathbf{x}^i_{0:t}|\mathbf{z}_{1:t})}{q(\mathbf{x}^i_{0:t}|\mathbf{z}_{1:t})} \tag{2.45}$$

It can then be shown that the weight update can recursively be done in terms of w^i_{t-1} such that (Chen et al., 2003):

$$w^i_t \propto w^i_{t-1}\frac{p(\mathbf{z}_t|\mathbf{x}^i_t)p(\mathbf{x}^i_t|\mathbf{x}_{t-1})}{q(\mathbf{x}^i_t|\mathbf{x}^i_{0:t-1}, \mathbf{z}_{1:t})} \tag{2.46}$$

With the assumption of a Markov process, the update equations simplify to:

$$\begin{aligned} \mathbf{x}^i_t &\simeq q(\mathbf{x}_t|\mathbf{x}^i_{t-1}, \mathbf{z}_t) \\ w^i_t &\propto w^i_{t-1}\frac{p(\mathbf{z}_t|\mathbf{x}^i_t)p(\mathbf{x}^i_t|\mathbf{x}_{t-1})}{q(\mathbf{x}^i_t|\mathbf{x}^i_{0:t-1}, \mathbf{z}_t)} \end{aligned} \tag{2.47}$$

In practical application of particle filters, iteration of updates in Equations 2.47 can result in degeneracy, where only some of the particles have a survival weight, and all the other ones have very small weights. One way to deal with this problem is through re-sampling whenever the effective sample size goes below a certain limit. While most particle filtering algorithms fall under the family of the SIS, several other variations have been proposed. The SIS algorithm, for instance, uses the proposal distribution $q(\mathbf{x}_n|\mathbf{x}^i_{t-1}, \mathbf{z}_n)$, which is

taken to be the state transition distribution $p(\mathbf{x}_n|\mathbf{x}_{t-1}^i)$ and resampling is done at every iteration. Van Der Merwe et al. proposed the use of the Extended or the Unscented Kalman filters as the proposal distribution instead of the transition prior (Van Der Merwe et al., 2001). This enables more efficient sampling by moving the particles towards regions of high likelihood. Consequently, the problem of only a few particles surviving will be avoided. The use of the EKF/UKF as the proposal requires the propagation of the sufficient statistics of the EKF/UKF for each particle (Van Der Merwe et al., 2001).

2.3.2 Estimator Design for Drilling Models

The particle filtering methods discussed previously seem to enable us to avoid any assumptions regarding linearity and Gaussianity of the model. However, this comes at the added cost of design , computation and implementation complexity on field-level hardware. To put in perspective the performance of the aforementioned recursive Bayesian filtering techniques in a non-linear, non Gaussian setting, a comparison study is shown in Figure 2.8 (Van Der Merwe et al., 2001). The process model is described by:

$$x_{t+1} = 1 + sin(\pi\omega t) + \phi x_t + v_t \tag{2.48}$$

where ω and ϕ are scalar parameters, and v_n is the process noise modeled by a Gamma random process. The measurement equation is given by:

$$\begin{aligned} y_t &= \phi x_t^2 + r_t \qquad t \leq 30 \\ y_t &= \phi x_t - 2 + r_t \quad t > 30 \end{aligned} \tag{2.49}$$

where r_n is Gaussian measurement noise. In the first $30s$ of this comparison, the particle filter based on the UKF proposal is the obvious winner, with the EKF proposal performing similarly well. The interesting observation, however, is that when the measurement equation behaves linearly with respect to x at $t = 30$, the performance margin of particle filters is significantly reduced, with the generic particle filter performing worse than the EKF and UKF. We can therefore note that while the process model has remained nonlinear, the nonlinearity of the measurement equation can motivate the use of particle filters for greater accuracy. However, if the measurement model is somewhat linear with respect to the state, and the process model is not severely nonlinear, the use of the more involved particle filters is not practically justified.

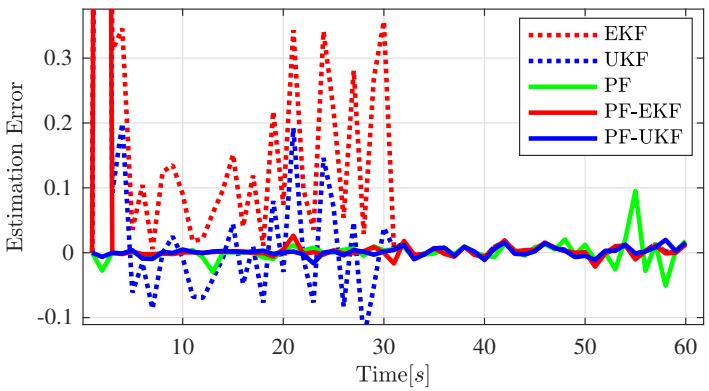

Figure 2.8: Comparison of several Bayesian filtering techniques on a system with an unstationary measurement equation. At $t = 30$, the measurement equation becomes linear (Van Der Merwe et al., 2001).

In designing real-time parameter learners of the drilling models, the UKF is therefore a fair choice for two reasons: (i) given that the observation

model is linear, the algorithm enables one to sufficiently deal with the non-linearity that arises from the joint parameter estimation formulation without resorting to the more involved particle filters, and (ii) as opposed to the EKF, it avoids the Taylor series approximation involved in approximating the posterior distribution and the use of a Jacobian. The UKF-based joint parameter estimation can then be set up as follows:

$$d\mathbf{x} = F(\mathbf{x}, u, t)dt + d\beta \tag{2.50a}$$

$$\mathbf{z}_k = H\mathbf{x} + \mathbf{v} \tag{2.50b}$$

where $\mathbf{z}_k$ is the measurement vector, H is the measurement matrix, β is a zero-mean random variable of variance q that represents process uncertainty, and v is the measurement uncertainty which can be written as:

$$E[\mathbf{v}] = 0, E[\mathbf{v}\mathbf{v}^T] = R = \sigma_{hkl}^2 \tag{2.51}$$

where σ_{hkl}^2 is the variance of measurement error in the surface tension measurement and,

$$H = \begin{bmatrix} 1 & \dots & 0 \end{bmatrix} \tag{2.52}$$

The state vector $\mathbf{x}$ is given by:

$$\mathbf{x} = \begin{bmatrix} F_1 \\ v_1 \\ \vdots \\ F_{n-3} \\ v_{n-3} \\ F_c \\ v_c \\ F_{fr} \\ \theta_1 \\ \vdots \\ \theta_m \end{bmatrix}, \tag{2.53}$$

and the transition function, $\mathbf{f}(\mathbf{x})$ by:

$$f(\mathbf{x}) = \begin{bmatrix} k_q\big(v_{input} - v_1\big) \\ \frac{1}{m_q}\big(F_1 - F_2 - b_q v_1\big) \\ \vdots \\ k_q\big(v_{n-4} - v_{n-3}\big) \\ \frac{1}{m_q}\big(F_{n-3} - F_c - b_q v_{n-3} - \Psi_{ds}(v_{n-3})\big) \\ k_c\big(v_n - v_c\big) \\ \frac{1}{m_c}\big(F_c - F_{fr} - b_c v_c - \Psi_c(v_c)\big) \\ k_{fr}\big(v_c - v_{disturbance}\big) \\ \epsilon_1 \\ \vdots \\ \epsilon_m \end{bmatrix} \tag{2.54}$$

$$v_{disturbance} = v_{ROP} = \Gamma(WOB_{applied}, \Omega) \tag{2.55}$$

The formulation above represents a generic setup for the parameter learner. While n represents the number of lumped elements used to describe the drill-pipe dynamics, m is the number of model parameters being simultaneously estimated. In the case where the drill-pipe parameters are sufficiently

known, a more accurate formulation can help with improved estimation of parameters that are not related to the drill-pipe, including the formation stiffness k_{fr}. In this case, $f(\mathbf{x})$ can be written as:

$$f(\mathbf{x}) = \begin{bmatrix} k_c\big(v_{ds} - v_c\big) \\ \frac{1}{m_c}\big(F_c - F_{fr} - b_c v_c - \Psi_c(v_c)\big) \\ k_{fr}\big(v_c - v_{disturbance}\big) \\ \epsilon_1 \\ \vdots \\ \epsilon_m \end{bmatrix} \tag{2.56}$$

where v_{ds} is calculated using delay differential equations such that:

$$v_{ds}(t) = \frac{r_{ds}}{Z_{ds}+1}\left(2v_{in}(t-T) + \frac{r_{ds}v_{ds}(t-2T)}{Z_{ds}-1} - \frac{F_c(t) - F_c(t-2T)}{Z_{ds}}\right) \tag{2.57}$$

The measurement equation is now given by:

$$z = 2F_c(t-T) - F_{ds}(t-2T) + (Z_{ds}+r_{ds})v_{in}(t) + (r_{ds}-Z_{ds})v_{in}(t-2T) + 2r_{ds}v_{ds}(t-T) \tag{2.58}$$

However, by inspection of Equation 2.60, it can be noted that observation is no longer a function of the state, but rather a delayed value of state given by $2F_c(t-T)$. To overcome this issue, we introduce an approximation, in order to predict the history of F_c such that:

$$F_{c,delayed} = F_c - Tk_c\big(v_{ds} - v_c\big) \tag{2.59}$$

The new observation equation becomes:

$$z = 2F_{c,delayed} - F_{ds}(t-2T) + (Z_{ds}+r_{ds})v_{in}(t) + (r_{ds}-Z_{ds})v_{in}(t-2T) + 2r_{ds}v_{ds}(t-T) \tag{2.60}$$

In Figure 2.9, a comparison of the delay differential equation (DDE) based estimator and the ordinary differential equation (ODE) based estimators are shown. The estimators are estimating F_c, with a 10% uncertainty in r_{ds} and 5% uncertainty in Z_{ds}. As expected, the DDE estimator performs significantly better than the ODE estimator, when the drill-string is subject to high frequency inputs. Note that as mentioned previously, drill-pipe parameters need to be sufficiently known before the DDE estimator can be used, as assumed above with both r_{ds} and Z_{ds}.

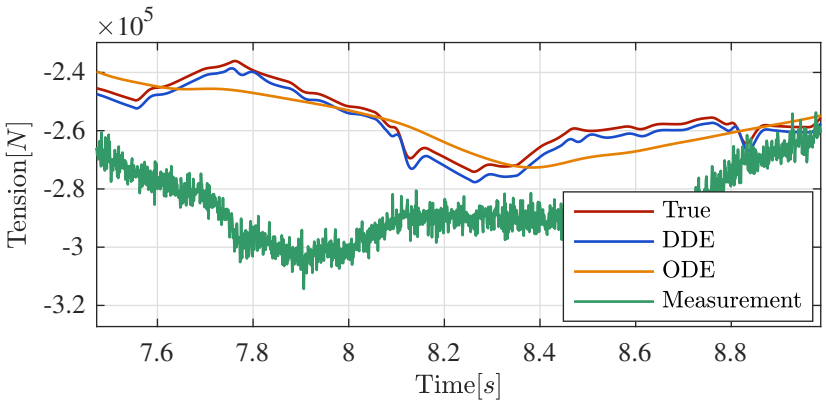

Figure 2.9: Comparison of state estimators based on ODE and DDE transition functions. The discrepancy between the measurement and state estimates in due to the measurement being based on surface value of F_{ds}.

For estimation of hydraulic parameters and states. the transition function is given by:

$$f(\mathbf{x}) = \begin{bmatrix} K_{ds}(Q_{in} - Q_{bit}) \\ K_{cv_1}(Q_{bit} - Q_1) \\ \frac{1}{M_1}(P_1A_1 - P_2A_1 - P_{f1}A_1) \\ K_{cv_2}(Q_1 - Q_2) \\ \frac{1}{M_2}(P_2A_2 - P_3A_2 - P_{f2}A_2) \\ K_{cv_3}(Q_2 - Q_3) \\ \frac{1}{M_3}(P_3A_3 - P_4A_3 - P_{f3}A_3) \\ K_{cv_4}(Q_3 - Q_4) \\ \frac{1}{M_4}(P_4A_4 - P_{f4}A_4) \end{bmatrix} \tag{2.61}$$

$$\mathbf{z}_k = P_{f_{ds}} + \mathbf{x}(1) + \mathbf{v} \tag{2.62}$$

where $\mathbf{z}_k$ is the measurement vector, H is the measurement matrix, β is a zero-mean random variable of variance q that represents process uncertainty, and v is the measurement uncertainty which can be written as:

$$E[\mathbf{v}] = 0, E[\mathbf{v}\mathbf{v}^T] = R = \sigma_{spp}^2 \tag{2.63}$$

where σ_{spp}^2 is the variance of error in the standpipe pressure measurement, and

the state vector $\mathbf{x}$ is given by:

$$\mathbf{x} = \begin{bmatrix} P_{ds} \\ P_1 \\ Q_1 \\ P_2 \\ Q_2 \\ P_3 \\ Q_3 \\ P_4 \\ Q_4 \\ \theta_1 \\ \vdots \\ \theta_m \end{bmatrix}, \tag{2.64}$$

Since the initial learning task has to be undertaken by the ODE estimator, the right choice of n is obviously of high importance, as already discussed previously. To highlight this, let us refer to Figure 2.10, where different order ODE learners are used to estimate the PDF of the drillstring mass parameter. To perform this simulation, the UKF-based learner was deployed on a simulator of the drillstring, based on the exact representation described in Equations 2.14 and 2.13. As shown, the lowest order formulation, with $n + m = 7$, is a clear biased estimator of the mean, with both the initial guess, and the true value of m_{ds} almost equally probable. Higher order estimators begin to do a better job, with $n + m = 21$ being the least biased estimator of the true parameter, and a smaller variance.

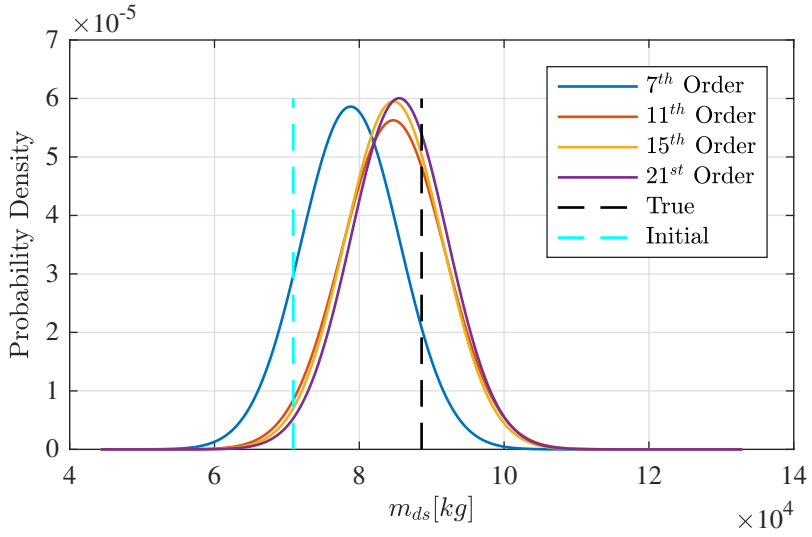

Figure 2.10: Estimation of the drillstring mass parameter, using various order UKF learners. Data for learning was generated using the infinite dimensional model.

A similar result is obtained while estimating the mean value of k_{ds} as shown in Figure 2.11. While the PDFs of all the estimators are closer in this scenario, the higher order estimators perform better, with the exception of $n + m = 21$ which underestimates the mean. While intuitively these results motivate increasingly higher values of n, we found that numerical instabilities can occur at values higher than $n = 20$. Implementation of higher order learners can also become increasingly difficult due to higher costs of matrix manipulation.

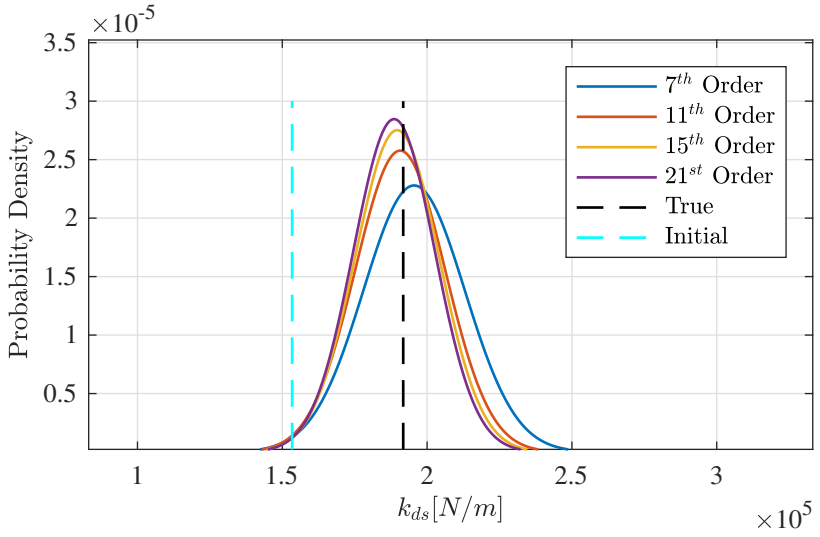

Figure 2.11: Estimation of the drillstring stiffness parameter, using various order UKF learners. Data for learning was generated using the infinite dimensional model.

2.3.3 Ensemble Unscented Learner (EUL)

In Figures 2.11 and 2.10, it was shown that the UKF learner based on an ODE transition function can effectively learn the parameters of the realistic model, with the parameter mean moving from the initial guess towards the true value of the parameter. While increasing the order of the transition function allows a more accurate estimation, the resulted performance gain can be minimal (as shown when $n > 15$). In this section, we explore the idea of ensemble learning applied to the UKF learner, with the aim of reducing estimation bias.

In the examples of Figures 2.10 and 2.11, the learner was initialized using an initial guess value of m_{ds} and k_{ds}. In practice, however, it would

make sense that the learner has knowledge regarding an initial distribution, essentially a description of both the mean and variance associated with the uncertain parameter. The idea behind EUL is that instead of using one UKF learner initialized using the mean of the initial guess, one would employ many UKF learners, each of them initialized with values sampled from the *a priori* parameter distribution. Amongst these learners, inherently some will outperform the rest, with the best-performing being the one initialized with the closest value to the true mean. When estimating two parameters simultaneously, the initial guess is essentially a two dimensional Gaussian distribution, and the initialization of ensemble learners will be sampled from this distribution. With an ergodicity assumption, the best performer can be selected using various criteria, including the time-series variance of each estimate, as well as the total final uncertainty of the learner, by taking the trace of the covariance matrix, $\mathbf{P}$. The EUL can be summarized as follows:

1. Obtain an initial guess distribution of $\theta_1 : \theta_2$.
2. From the distribution in (1), randomly sample J different UKF learners.
3. Use time-series variance of each learner's estimates, as well as $trace(\mathbf{P})$, to select best performing learner.

Of course the computational cost of EUL is proportional to the number of random learners used, so in practice the use of EUL with many learners can become computationally expensive. Each member of the ensemble can, however, work in lieu of the rest, so the algorithm can be implemented in a parallel

fashion. In Figure 2.12, the performance results of the EUL are shown. In this task, two parameters, k_{ds} and b_{ds} were estimated at the same time. Clearly, the performance gain obtained by using up to 5 ensemble filters can be quite significant. At values above 5, however, the use of more filters would not be justified, as the reduction in estimation error becomes minimal.

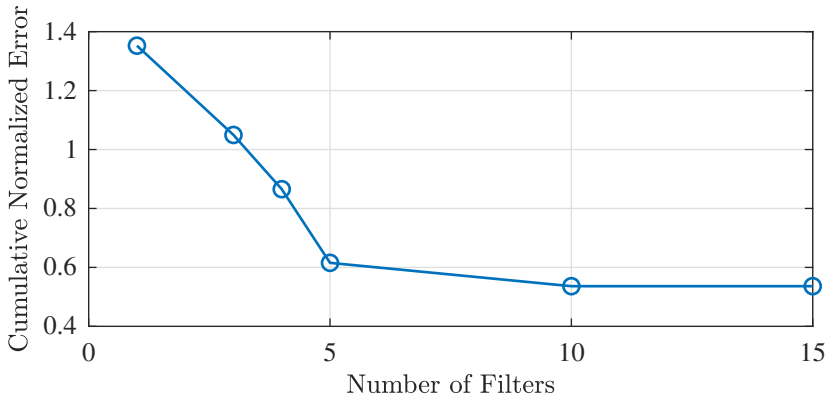

Figure 2.12: Performance of EUL with different number ensemble filters. Performance was measured by taking an l^2 norm of the normalized estimation errors of k_{ds} and b_{ds}.

2.4 Considerations for Observability and Action Choices

In this section, the learning of particular parameters of the physics-based models are explored and the following idea is repeatedly revisited: *for practical learning of model parameters, the learning agent must only estimate parameters during certain time frames of the drilling process, using specific actions designed to make the relevant parameters observable.* Initially, the simultaneous learning of drillstring parameters are discussed, and results are shown that verify the estimation ability of the learner. The learning of drilling

fluid parameters are also discussed, and it is shown that the learner shall take into account the flow regime of the drilling fluid to perform reliable estimation. Based on these discussions, we summarize the set of actions, and drilling scenarios that the learning agent takes into account to systematically learn and reduce the uncertainty in model parameters. In Table 2.1, model parameters used for the studies in this section are presented.

Parameter	**Initial Estimate**	**True Value**
$m_{ds}[kg]$	8.86×10^4	8.86×10^4
$k_{ds}[N/m]$	1.53×10^5	1.92×10^5
$b_{ds}[Ns/m]$	8.34×10^4	1.04×10^5
$m_{c[kg]}$	2.72×10^4	2.72×10^4
$k_c[N/m]$	3.36×10^7	4.19×10^7
$b_c[Ns/m]$	6.83×10^5	8.54×10^5
$\kappa[N]$	0	2.40×10^5
$k_{fr}[N/m]$	0	1.00×10^7
PV$[cp]$	25.6	32.0
YP$[lb/100ft^2]$	8.80	11.0
$K[GPa]$	4.00	5.00

Table 2.1: Model parameters used for estimator initialization and data generation. Note that for the mass parameters, actual measurement is trivial when the drillstring is not in slips. Hence initial estimates and true values for these parameters are the same.

In Sections 2.3.2 and 2.3.3, an important assumption was that the measurements contains relevant information about the parameters of interest. In control theory notation, this is referred to as parameter observability, which is a measure of how well the states and parameters of a model can be inferred, by looking at the input and output signals of data. In the following section,

we demonstrate that such an assumption does not always hold, and it is shown that learning while the parameter of interest is unobservable can lead to estimate degeneration. Let us consider the familiar state and parameter vector $\mathbf{x}$ defined by:

$$\mathbf{x} = \begin{bmatrix} x_1 \\ \vdots \\ x_n \\ \theta_1 \\ \vdots \\ \theta_m \end{bmatrix} \tag{2.65}$$

The observation function is defined as:

$$z = h(\mathbf{x}) \tag{2.66}$$

and its higher order derivatives by:

$$z = h = L_f^0(h)$$

$$\dot{z} = \dot{h} = L_f^1(h)$$

$$\cdots$$

$$z^{n+m-1} = L_f^{n+m-1}(h),$$

where $L_f(h)$ is the Lie derivative of h along the $f(\mathbf{x})$ vector field. An observability matrix can then be formulated as:

$$O = \begin{bmatrix} \frac{\partial L_f^0(h)}{\partial x_1} & \cdots & \frac{\partial L_f^0(h)}{\partial \theta_m} \\ \vdots & \ddots & \vdots \\ \frac{\partial L_f^{n+m-1}(h)}{\partial x_1} & \cdots & \frac{\partial L_f^{n+m-1}(h)}{\partial \theta_m} \end{bmatrix} \tag{2.68}$$

The observability matrix constructed in Equation 2.68 has to be full rank for observability to hold. Therefore, if parameter learning of a certain parameter

is performed when that particular parameter is unobservable, then the estimate will degenerate over time. In addition, the observability matrix O might sometimes be full rank, but still contain little information about the parameter space. It is therefore important to have a real-time measure of such conditions, in order to pause the estimation of the almost unobservable parameters, and avoid estimation degeneracy. Such an estimate can be obtained by deriving an observability index, which can be calculated by evaluating the condition number of the observability matrix (Nakhaeinejad, 2010). The condition number of a matrix is obtained by performing a singular value decomposition, and producing a basis for the row and column space of the matrix such that:

$$O = USV^T \tag{2.69}$$

where U is a unitary matrix of size $m \times m$, S is a positive semi-definite diagonal matrix, and V is an $n \times n$ unitary matrix. The condition number can then be obtained by evaluating the maximum and minimum diagonal elements of S, such that:

$$C = \frac{\sigma_{max}}{\sigma_{min}} \tag{2.70}$$

where a large value of C would correspond to a less observable system. In addition, each state/parameter corresponding to each singular value can be obtained from the companion matrix V^T. The column of V that corresponds to the largest singular value σ_{max} provides the most observable combination of states/parameters. This procedure will therefore aid in the selection of the right parameters/states to estimate at a single instance in time (Nakhaeinejad, 2010).

2.4.1 Learning Drillstring Parameters

To explore the concept mentioned above, consider again the simultaneous learning of parameters b_{ds} and k_{ds}. In Figure 2.13, the agent executes a trapezoidal sequence of actions with zero bias, as shown in the upper plot. Since the input velocity comes back to zero within ten seconds, the learner fails to estimate the parameter value. As indicated by the high condition number prior to $t = 30$, the learner completely loses observability of the parameter when the string velocity comes to zero. The new step change induces a short period of correct estimation, but the estimation diverges again as soon as the step input drops.

Better estimation results are obtained in the case of Figure 2.14, where a similar sequence of actions are taken, but with a non-zero bias. At $t > 20$, the learner begins to estimate the correct value of b_{ds} and its estimate does not diverge afterwards. The stable condition number confirms these results, and proves to be a useful measure that the agent can use to prevent estimation degeneracy as in the case of Figure 2.13.

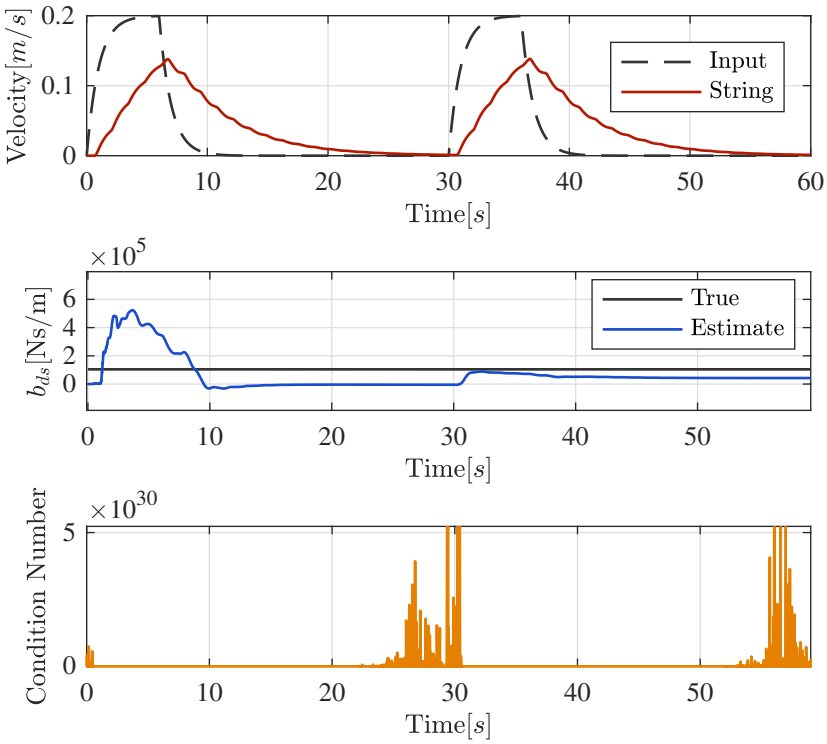

Figure 2.13: Simultaneous estimation of k_{ds} and b_{ds} using a trapezoidal action sequence with zero bias. Estimation results shown for b_{ds}.

A similar observation can be made for the k_{ds} parameter. By inspection of the transition function in Equation 2.3.2, one can note the condition for observability of k_{ds}, which is strictly when $v_{in} - v_{ds}$ is non-zero. As revealed in Figure 2.15, a step-response type action fails to allow the learner to estimate the stiffness parameter, due to the value of $v_{in} - v_{ds}$ quickly becoming zero.

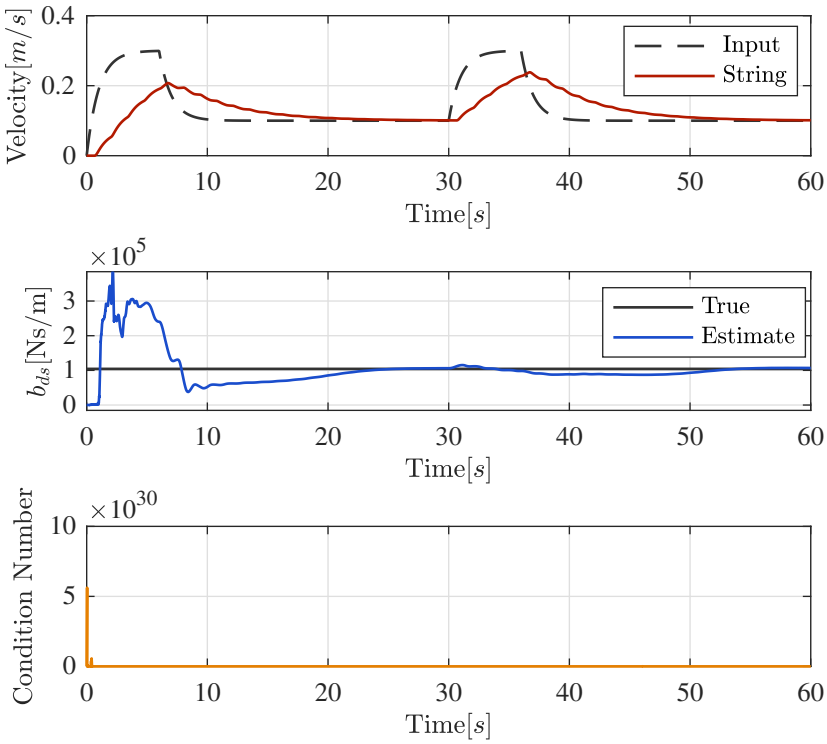

Figure 2.14: Simultaneous estimation of k_{ds} and b_{ds} using a trapezoidal action sequence with non-zero bias. Estimation results shown for b_{ds}.

.

In Figure 2.16 however, a much better estimation performance is seen, as the value of $v_{in} - v_{ds}$ is enforced to remain non-zero at all times. In fact, for the short period where $v_{in} - v_{ds}$ approaches zeros, the estimate of k_{ds} slightly diverges.

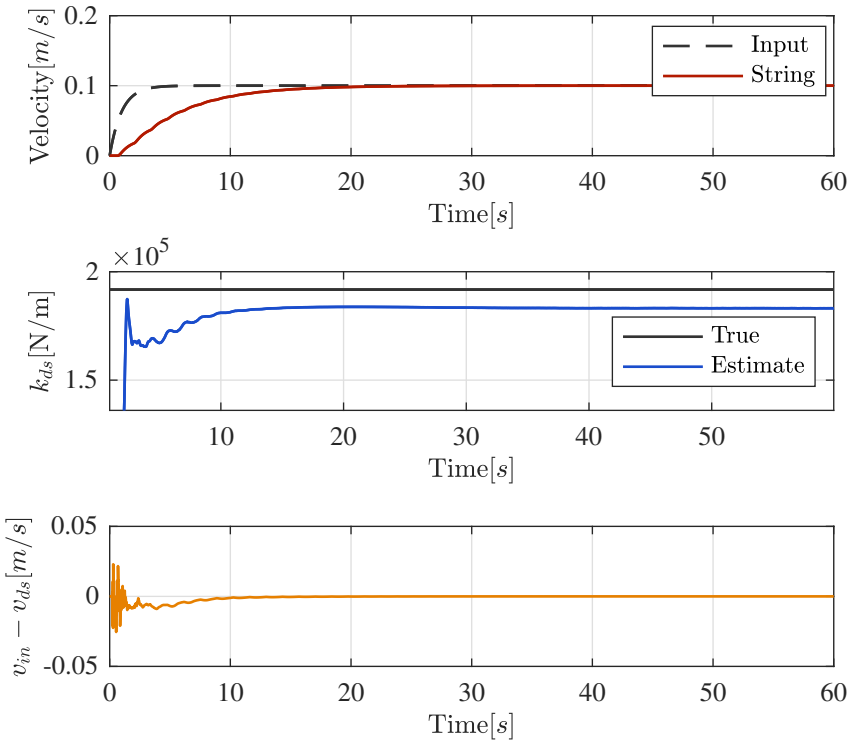

Figure 2.15: Simultaneous estimation of k_{ds} and b_{ds} using a step action sequence. Estimation results shown for k_{ds}.

Learning κ, which is the nonlinear friction force, can be performed in a similar manner from an actions perspective. However, observability of this parameter is reduced, especially in shorter lateral sections, if the input velocity is high. This is due to linear damping from the b term accounting for the majority of the dissipative force. As shown in Figure 2.17, a more precise estimate is obtained at a very low input velocity, even though with both input velocities the learning task still converges. For the learning task shown in

Figure 2.17, the DDE learner of Figure 2.9 was used for improved, unbiased estimation. The reader should note, however, that the use of the DDE learner is only an option if the drill-pipe parameters b_{ds} and k_{ds} have already been estimated.

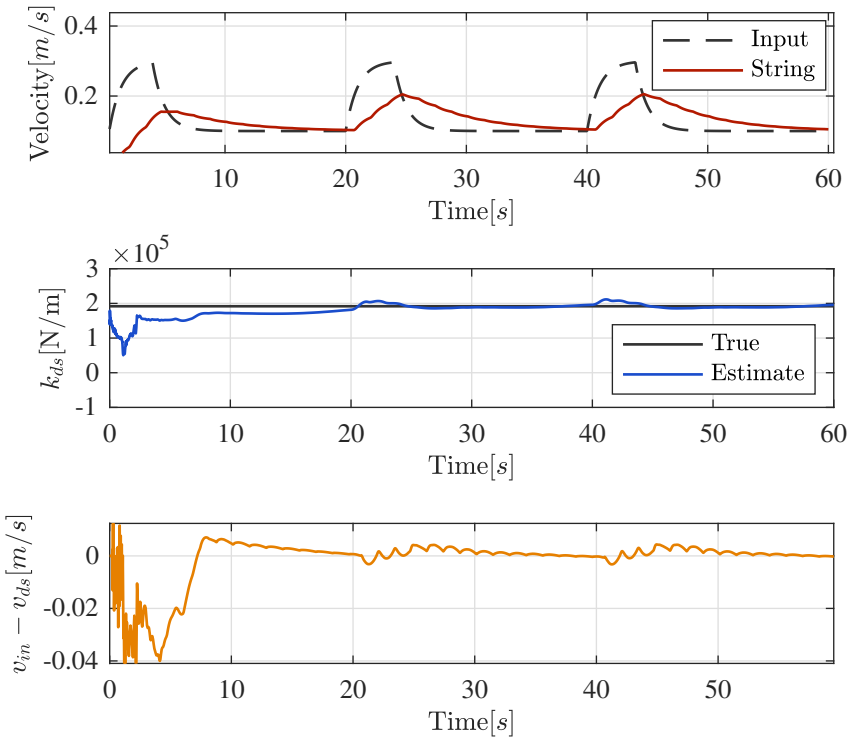

Figure 2.16: Simultaneous estimation of k_{ds} and b_{ds} using a trapezoidal action sequence with non-zero bias. Estimation results shown for k_{ds}.

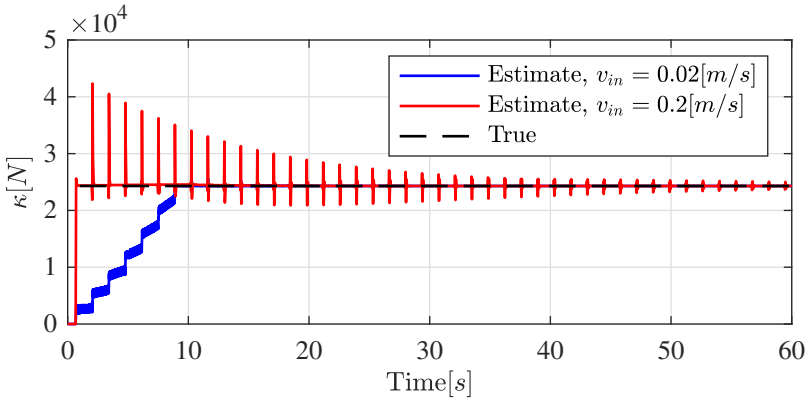

Figure 2.17: Estimation of the nonlinear friction parameter κ using two different step sequences. The learner used for this task was based on the DDE transition function, since the drillstring parameters have already been sufficiently identified.

2.4.2 Learning Drilling Fluid Parameters

The overall learning approach for hydraulic parameters is similar to the one for drillstring parameters discussed above. The most uncertain hydraulic parameters from an automation perspective are K the bulk modulus of elasticity of the fluid, and the rheological fluid parameters, which in the case of Bingham Plastic fluids are represented by the plastic viscosity (PV), and the yield point (YP). The uncertainty in these parameters arises from poor initial knowledge, as well as the influence of pressure and temperature.

The learning of the rheological parameters presents another interesting lesson about observability. Consider the learning results shown in Figure 2.18. While it is clear that a step-type input would provide the best action sequence for learning (we learned this from the case of learning b_{ds}), the size of the

step input seems to play an important role as well. As demonstrated, when the flow regime is laminar, the learner performs the best job of estimating these parameters. At higher Reynolds number with turbulent flow, the rheological parameters become significantly less observable, and hence estimation divergence occurs as shown in Figure 2.18 with a $650gpm$ input. The learning conditions for these parameters are dominated primarily by the flow regime in the drillstring and the annulus, and specifically by the highest Reynolds number amongst all control volumes.

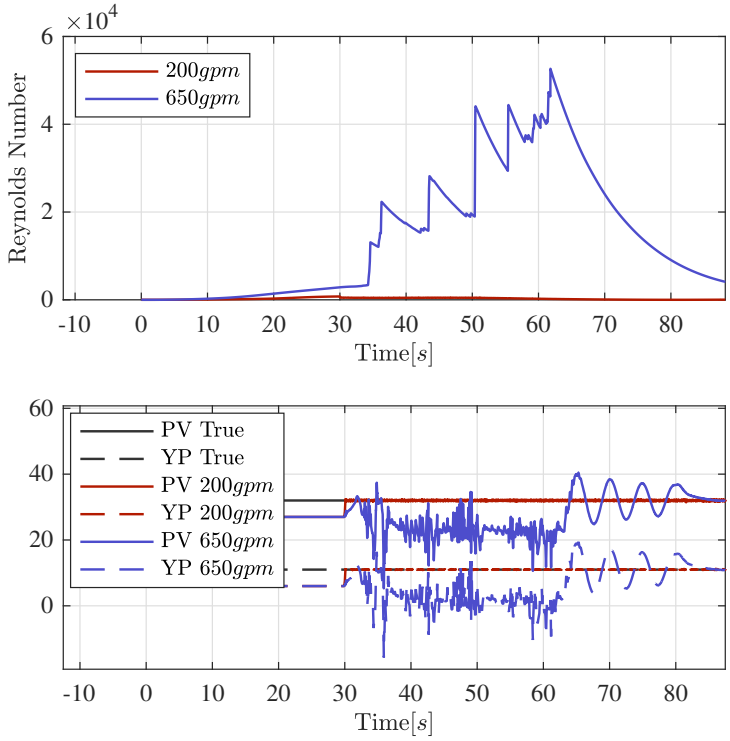

Figure 2.18: Simultaneous estimation of PV and YP using a trapezoidal action sequence with different steady-state flow-rates.

As evident in Figure 2.18, the learner has to actively use the highest Reynolds number as a measure of learning condition, and stop the update step of the algorithm as soon as the Reynolds number indicates turbulent flow. For the case of K, it seems that theoretically the value of $Q_{in} - Q_{bit}$ has to be sustained at non-zero values at all times for observability conditions to hold, and therefore a ramp-type input sequence would be optimal.

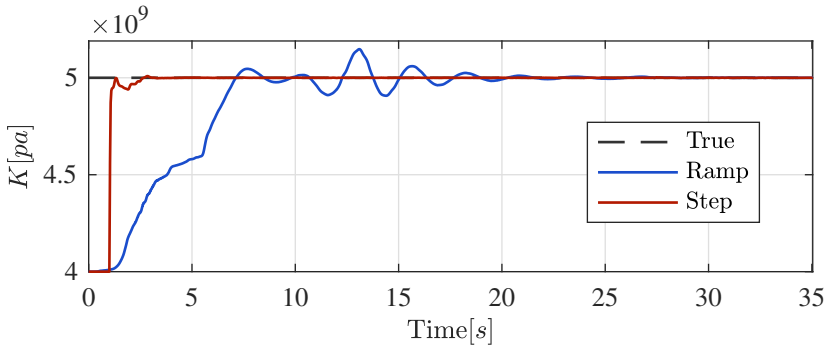

Figure 2.19: Estimation the bulk modulus of elasticity of the drilling fluid using ramp and step action sequences.

As shown in Figure 2.19, due to the slow settling of pressure dynamics, a step action sequence works just as well as a ramp input for the estimation task, and in fact it enables faster convergence to the true value. Therefore, the observability conditions for the K parameter seem to play a less important role when compared to other parameters.

2.4.3 Practical Learning Conditions

The simulation studies of Figures 2.13 to 2.17 reveal a common finding; that the optimal sequence of actions to learn spring-type and resistive

parameters are those that maintain prolonged periods of constant acceleration and constant velocity. Interestingly, noting the process of tripping pipe, this is exactly the sequence of actions pursued when a new string of pipe is tripped into the hole. Therefore, the learning agent, without having to dedicate additional time to learning, can systematically learn the drill-string parameters while tripping.

The idea that the learning agent can take advantage of a tripping process to learn about the model parameters before drilling begins opens up an interesting avenue for systematic uncertainty reduction. Recalling that the most uncertain model parameters are those listed in Table 2.1, the following process can be undertaken by the agent to sequentially learn all the uncertain parameters of interest. The process begins by tripping in the BHA and the drill-collars. During this stage, only the first two parameters k_c and b_c are subject of estimation, with the remaining parameters being non-existent. With a reliable estimate of k_c and b_c, the learner can subsequently focus on estimating k_{ds} and b_{ds} while the drillstring is still in the vertical section of the hole. After the initial estimation of drillstring parameters has been completed, future changes in these parameters can simply be extrapolated as a function of increasing hole depth. With all stiffness and damping parameters estimated, two uncertain parameters remain. κ can be estimated right before the drill-string tags bottom, and k_{fr} which is estimated while WOB is transferred to the drill-bit. During the drilling process itself, only κ and k_{fr} are expected to change time to time, which can be estimated after every connection. The

process described above is summarized in Table 2.2. This summary serves as the main guideline in which the *Learning Agent* follows to sequentially learn the parameters of the axial dynamics portion of the physics-based model. k_c and b_c are learned when $BD_{md} \leq L_c$ because the dynamics observed in the measurement data only represent the drill-collars and the BHA. Once these dynamics are captured by the agent, k_{ds} and b_{ds} are estimated while tripping in the vertical section of the hole, when $BD_{md} = BD_{tvd}$. By performing this estimation in the vertical section, the effects of linear damping are distinguishable from dry friction. At $BD_{md} \approx HD$, the nonlinear friction forces that represent κ have fully appeared in the dynamics of the drillstring, and therefore the only remaining unknown parameter to be estimated is κ. At the onset of drilling the first stand after the tripping process, k_{fr} can be estimated to assess the stiffness of the formation.

Parameter	**Learning Condition**	**Learning Action**
k_c	$BD_{md} \leq L_c$	Trapezoid
b_c	$BD_{md} \leq L_c$	Trapezoid
k_{ds}	$BD_{md} = BD_{tvd}$	Trapezoid
b_{ds}	$BD_{md} = BD_{tvd}$	Trapezoid
κ	$BD_{md} \approx HD$, pumps on	Step
k_{fr}	$BD_{md} = HD$	Ramp

Table 2.2: Learning conditions for drillstring parameters during a tripping-in process.

To highlight the importance of condition-based sequential learning, the case of learning the formation stiffness parameter k_{fr} is portrayed in Figure

2.20. In Case 1, the learner tries to simultaneously learn both the formation stiffness and the drillstring stiffness parameter upon tagging bottom, as it had failed to satisfy the k_{ds} learning condition in Table 2.2. As shown, not only does the learner fail to estimate the true value of k_{ds}, its estimate of the formation stiffness which is initially at 20% of its true value degrades further through the estimation process. In comparison, Case 2 shows the scenario where the learning condition of k_{ds} has been satisfied, and the learner has estimated the value of k_{ds} with a 5% error, prior to attempting to learn k_{fr}. Therefore, the learning process is significantly improved in Case 2. Note the estimation error present in k_{fr} in Case 2 is due to the prior 5% uncertainty in k_{ds}, as the estimation error propagates throughout the sequential learning process.

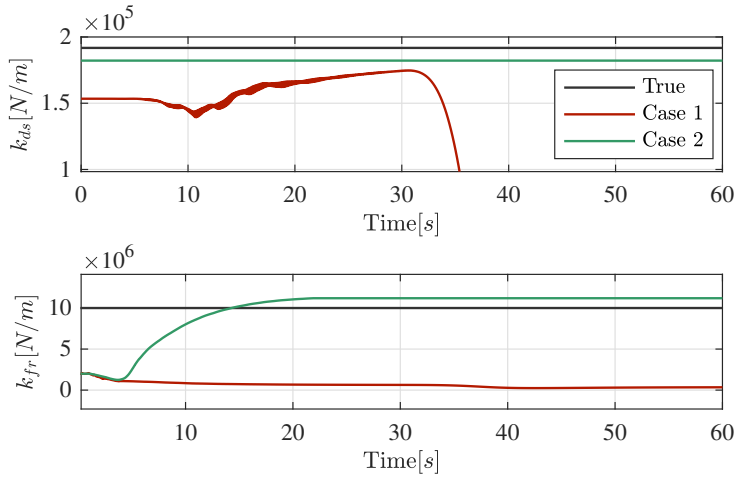

Figure 2.20: Estimation of formation stiffness after tripping in and tagging bottom. Case 1 demonstrates the simultaneous learning of two stiffness parameters. Case 2 demonstrates sequential learning, where k_{ds} had already been estimated earlier before the drillstring was fully tripped in.

Similar to the sequential learning process described in above for axial drillstring dynamics, the learning of drilling fluid parameters has to be performed with the conditions that satisfy the considerations discussed in Section 2.4.2. Opposite to the case of drillstring parameters, however, the learning task is completed at the end of a drilling cycle before a stand is tripped out of the hole, while starting and stopping the pumps for a connection. The learning sequence is summarized in Table 2.3

Parameter	**Learning Condition**	**Learning Action**
PV	$Re \leq Re_{turb}$	Step
YP	$Re \leq Re_{turb}$	Step
K	-	Step/Ramp

Table 2.3: Learning conditions for drilling fluid parameters during a sequenced pump-on process after a connection.

2.5 Conclusion

In this chapter, a detailed approach to parameter learning of physics-based drilling models is presented. This approach which formed the basis for the *Learning Agent* in architecture of Figure 1.2, uses Bayesian filtering to estimate the unknown/uncertain parameters of the model sequentially in different drilling scenarios. The parameters along with the model itself enable the rest of the self-learning control system to make and execute optimal decisions in real-time. The next chapter will discuss how the physics-based model can be used for optimal decision making.

Chapter 3

Optimizer: Action-Driven Drilling Optimization

Chapter 2 addressed the problem of self-learning a physics-based model for optimization and control. From an optimization perspective, however, this model is only useful for prediction of the environment's response to avoid process constraints. A major aspect of the learning problem for the purpose of optimization is yet to be addressed, which is concerned with the relationship between the automation system's commands and user-defined performance metrics. Since the necessary data for this learning task has to be generated by the automation system itself through interaction with the environment, learning cannot be performed in a supervised-learning manner. This chapter addresses this challenge, and proposes a method for iterative learning and optimization of the drilling environment.

3.1 Introduction

During a drilling process, the rate at which a hole is drilled, the rate of penetration (ROP) is usually an unknown and varying function of the two drilling parameters WOB, and RPM, and depends on the type of drill-bit used,

as well as the particular formation (sand, shale, limestone, etc.) being drilled. From an optimization perspective, there are several factors that make drilling optimization (to achieve optimal ROP for instance) a challenging, yet interesting problem. First, modeling the subsurface environment is difficult since the wellbore goes through different formations and the drill-bit encounters unknown obstacles. In addition, certain combinations of surface parameters induce undesired downhole vibrations. These result in accelerated damage to the drill-bit and other downhole equipment, and reduce the quality of the wellbore being drilled. The vibration constraints also change as a function of the formation, drill-bit wear, and top-drive dynamics.

In this chapter, we approach drilling optimization from a reinforcement learning (RL) perspective. In RL, an artificial agent learns how to behave through interacting with the environment, as opposed to learning from labeled training data. This interaction provides the agent with discrete rewards based on the set of actions that it takes at every time step. The idea of learning to drill by interacting occurs to us when we think of how a human naturally learns to drill. Rather than solely relying on information provided by a teacher, the driller learns about the consequences of its actions through direct feedback from the environment, and evolves into an expert through accumulated experience over time.

The chapter is structured as follows: after presenting a study of existing literature, we present two formulations for the reinforcement learning problem. First, the task is treated as a Markov Decision Process (MDP),

where the drilling optimization task is represented by a tuple of states and actions, $< s, a >$. The second formulation, which we refer to as qDrill, is based only on a continuous action space, and makes the optimization task more feasible by assuming that large action transitions are allowed, which turns out to be a fairly practical assumption. With the latter formulation, the results demonstrate superior performance in finding close-to-optimal actions, but there are no theoretical guarantees, as the formulation is based on heuristic search. The qDrill formulation is analyzed in regards to its learning and optimization parameters, and tested for various scenarios. At the end of this chapter, optimization of pipe tripping is explored, and a solution is presented which utilizes the IMDT model from Chapter 1 to avoid the optimization constraints.

3.1.1 Literature Review

Drilling automation with the purpose of ROP optimization involves two main challenges: (1) modeling, or online learning of ROP as a function of bit design, wellbore conditions, and the WOB and RPM parameters, (2) determining the operational constraints regarding torsional (stick-slip) and lateral (whirl) vibrations (Gidh et al., 2012). In the drilling literature, researchers have taken various approaches towards these challenges. Below we provide a summary of these studies and how they relate to the approach done in this dissertation.

Caicedo et al. (2005) developed a model for drill-bit performance through specific energy theory and calculation of the rock confined compressive strength based on rock mechanics principles. Their results were in good agreement with field data, but the required model parameters were often difficult to obtain in practice. Although their work provides valuable insights to prediction of ROP based on physical principles, it cannot be relied on by an automation system due to the amount of uncertainty involved in the model parameters. For prediction of bit instability due to torsional and lateral vibrations, Dunayevsky et al. (1998) investigated the problem by considering the mechanical systems that represent the string torsional movement, the bit lateral movement, and the coupled torsional-lateral vibrations of the bit assembly. For each system, the stable RPM and WOB regions were identified by performing a linear stability analysis using the eigenvalue method. The applicability of their approach is limited however for real-time automation, as the model parameters and boundary conditions are unknown in practice. Ambrus et al. (2015) built on the aforementioned works, by providing a model-based methodology using tabular methods to determine the safe operating drilling envelope and performing ROP optimization using the physics-based techniques in (Dunayevsky et al., 1998) and (Caicedo et al., 2005). Their approach is also limited for the same reasons as the two previous works, as no online learning techniques are employed.

Dunayevsky et al. (1998) also took a machine learning approach to ROP optimization. They used a field dataset that involved various drilling

scenarios in different lithologic units, and trained a dynamic neural network with Bayesian regularization, to predict the dynamics of the bottom hole assembly and drilling dysfunctions. Using the trained neural network, a model predictive controller was then used to maximize ROP subject to the identified constraints, over a finite horizon. The output of the controller was displayed to the driller as a recommendation to enhance drilling performance. Although their work overcomes the limitations of a physics-based approach with unknown boundary conditions, the use of historical data for training the neural network is limiting since that data might not necessarily represent the dynamics encountered in future instances. Dunlop et al. (2011) demonstrated that the drilling response of a PDC (polycrystalline diamond compact) bit can be modeled as three distinct operating phases, each with a linear relationship between weight-on-bit, bit torque, and depth-of-cut per revolution. Online learning of each phase of the model was facilitated by the use of a Bayesian change point detection, in order to determine which phase of the model to train. Identification of the unsafe operating regions was done through both surface and downhole measurements by a human driller. A follow-up paper proposed a closed-loop controller to implement the ROP optimization scheme automatically without the intervention of a driller (Chapman et al., 2012). Although they provided promising results with significant increases in ROP in field trials, their framework is not capable of avoiding vibration limit zones, and the human needs to bring the system back into a stable region if signs of undesired vibration are observed.

In a more recent paper by ExxonMobil researchers, drilling optimization was approached from a mechanical specific energy minimization perspective, to limit the impact of drilling performance limiters, including drilling vibrations (Payette et al., 2015). The researchers adopted a similar approach to those in (Dunlop et al., 2011), but they additionally attempt to learn the unsafe drilling envelope from real-time data by introducing a torsional severity index. Their optimization scheme encouraged the human driller to perform drill-off tests (a procedure to determine the ROP as function of RPM and WOB in steady-state) to learn the unexplored regions of the RPM-WOB envelope, while paying attention to the torsional severity index. Learning was achieved through the use of decision trees and gradient accent optimization was used to recommend the best set of operating parameters to the driller. The authors noted that the optimization process involved a fundamental compromise between exploration and exploitation, where it is necessary to manage the trade-offs associated with conducting drill-off tests and leveraging the drill-off test results. This work is the most complete in terms of considering both the ROP learning problem, as well as avoiding zones with high vibrations that result in high MSE values. However, their approach is limited as it does not leverage the availability of prior models to accelerate learning, and does not have any means of avoiding unobservable lateral vibrations. In addition, exploration of the WOB-RPM space is only encouraged by the optimization system, and performed by the human expert.

The model-based reinforcement learning technique employed in this

chapter is capable of leveraging the physics-based approach used in Ambrus et al. (2015) as a starting point for the drilling agent. The agent will then be capable of exploring the WOB-RPM space, and learning the action values, similar to the case of (Dunlop et al., 2011). Unlike their system however, the RL agent will not require the intervention of a human to observe undesired vibrations, and is capable of updating the value function based on a torsional severity index. Finally, our RL agent is capable of balancing exploration vs. exploitation, to learn about the regions of the state-space that can potentially enhance its performance.

3.2 Problem Formulation

In this section, we devote some time to formulate the reinforcement learning problem as both a MDP, and a k-arm Bandit problem, while noting the major differences in these approaches. At the end of this section, the reasons for the adopted formulation and the qDrill algorithm are explained.

3.2.1 Model of Environment

In order to implement the RL agent, a realistic representation of the environment needs to be developed. For this purpose, we use the physics-based approach used in (Dunayevsky et al., 1998) and (Caicedo et al., 2005), and the aggregation methodology presented in (Ambrus et al., 2015) to create a tabular representation of the environment. The tools presented in (Dunayevsky et al., 1998) et al. enable the generation of a surface as a function of various WOB,

RPM values that correspond to unstable torsional and lateral vibrations. Using this surface, along with the operational constraints of the top-drive, an allowable operating region is obtained. This region can be mapped to the ROP domain using the relationships proposed in Caicedo et al. (2005). Figure 3.1 demonstrates the various constraints that are combined to create a representation of the WOB-RPM space.

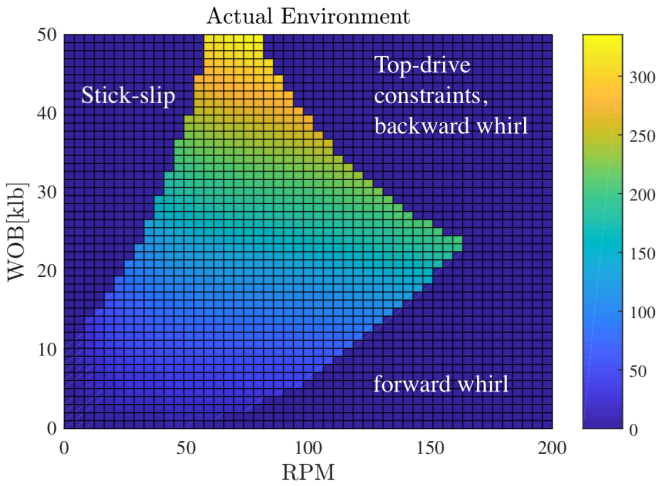

Figure 3.1: A tabular representation of the environment's ROP response in the WOB, RPM space. The blue regions correspond to unallowable drilling regions. The upper-left corner corresponds to the region of the state space that results in stick-slip vibrations. The upper-right corner corresponds to the top-drive performance constraints, and backward whirl vibrations. The lower-right corner corresponds to forward whirl vibrations (Ambrus et al., 2015).

In our stimulated environment, the agent receives reward depending on its location in the WOB-RPM space of Figure 3.1. In the actual environment, the agent calculates a torsional severity index by observing the torque and RPM sensor trends. Lateral vibrations are usually unobservable from surface

sensor data. Therefore, the agent would either need access to downhole accelerometer data, or to receive feedback from a modeled environment. ROP is calculated by averaging the time derivative of bit displacement in the hole.

3.2.2 Formulation as a Markov Decision Process

The reinforcement learning problem can be formulated as a Markov decision process (MDP), which can be written in terms of tuples $\langle \mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma \rangle$ consisting of $\mathcal{S}$, the set of all states; $\mathcal{A}$, the set of all actions; $\mathcal{P}_{sa}^{s'} = P_r(s'|s,a)$, the transition probability from state $s \in \mathcal{S}$ to state s' when action $a \in \mathcal{A}$ is taken; $\mathcal{R}_{sa} = E\{r|s,a\}$, the reward function giving the expected reward r when action a is taken in the state s; and γ, the discount factor corresponding to the weight of the future rewards versus that of the immediate reward. The actions are taken at discrete time steps according to a policy $\pi : \mathcal{S} \times \mathcal{A} \rightarrow [0,1]$, which defines for each action the selection probability conditioned on the state.

For the drilling problem, we define the set of states $\mathcal{S}$ as a discrete set of all possible combinations of WOB and RPM set points such that $\{wob, rpm\} \in \mathcal{S}$. The set of actions is defined as $\mathcal{A} = \{n, e, s, w\}$ and corresponds to the agent moving in the specified directions in the WOB-RPM space. At the beginning of an episode, the agent starts at $s = (0,0)$ and explores the state-space to collect as much reward as possible before the end of episode, defined as when a depth of $90ft$ (the length of a "stand" of 3 singles of $30ft$ Range II drill-pipe) has been drilled. The agent receives reward for every state, action pair. Upon hitting a vibration boundary, it receives a large negative reward. In this

formulation, we experiment with two different reward functions for the agent. One is defined as:

$$r_1 = \begin{cases} -1, \text{for each time step} \\ -10, \text{for hitting a vibration boundary} \end{cases}$$

and the other is:

$$r_2 = \begin{cases} ROP, \text{for each state} \\ -10 \times ROP, \text{for hitting a vibration boundary} \end{cases}$$

In the case of r_1, the agent's aim is simply to finish an episode as quickly as possible, to minimize negative returns. In the case or r_2, the agent directly receives the ROP associated with every state as reward, and its goal is to therefore maximize positive returns. In the context of MDPs, an optimal policy maximizes, for each state, the expected return, G, which is the discounted cumulative reward:

$$G = r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + ... = \sum_{k=1}^{\infty} \gamma^{k-1} r_{t+k}$$

where r_{t+1} is the reward received after taking action a_t in state s_t at time step t.

3.2.2.1 Model-based Reinforcement Learning: The Dyna Architecture

For simultaneous learning and interacting with the subsurface environment, we use the Dyna architecture as described in Sutton and Barto (1998). The Dyna architecture operates in a loop of interaction with the environment,

and unifies all the aspects of model learning and direct reinforcement learning. The Dyna loop involves using a model to create simulated experience, performing backups to improve policy, taking an action and reflecting the observed outcome on the model. In planning, a backup refers to updating the state-action and state value functions such that:

$$Q(s,a) \leftarrow \mathcal{R}_{sa} + \gamma \sum_{s'} \mathcal{P}_{sa}^{s'} V(s') \tag{3.1}$$

$$V(s) \leftarrow max(Q(s,a)) \tag{3.2}$$

In order to improve the efficiency at which simulated experience is generated, it is possible to use a heuristic that selects backups which are expected to produce a large value change. This efficient way of planning is referred to as prioritized sweeping (PS) (Sutton and Barto, 1998). PS uses all previous experiences to prioritize the sweeps and guide a more efficient exploration of the state-space.

The Dyna approach, along with the PS algorithm, were used to implement the presented MDP formulation of the drilling optimization problem on the environment in Figure 3.1 . In Figure 3.2, the learned state value function for the agent with the r_1 reward function after 3000 time steps is shown. With the agent receiving equal negative reward for every allowable state-action pair, it can only learn the value of each state after it has finished an entire episode. Therefore, it ends up exploring a wider area of the state-space during each episode. Note that the agent only learns a negative value for each state,

with more valued states being less negative. The value of impermissible states remains at 0, given that the agent never gets a chance to try them.

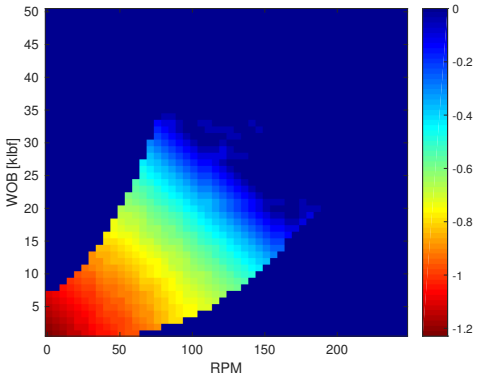

Figure 3.2: r_1 agent's learned state value function of the environment after 3000 timesteps.

The learned state value function for the agent with $R = r_2$ is shown in Figure 3.3. In this case, since the agent associates the value of each state directly with the obtained ROP, it can update the value of each state-action pair after every action. With that, the agent spends less time exploring low-reward actions, and begins to discover higher-reward actions quicker.

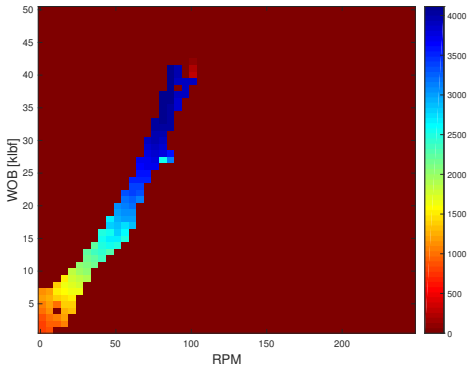

Figure 3.3: r_2 agent's learned state value function of the environment after 3000 timesteps.

In the case of both r_1 and r_2, the agent gets the opportunity to discover a significant portion of the state space. However, the learning is still quite slow (it took the agent 25 hours of drilling, with $30s$ spent in each state). In order to speed up the agent's learning, either the state-space has to be discretized with a coarser grid, or the agent has to spend less time in each state. However, none of these approaches seem desirable because a coarser grid results in a poor approximation of the environment's true operating region, and spending less time in each state would cause the agent to not observe the true rewards because of slow ROP dynamics.

3.2.3 Formulation as a k-arm Bandit: the qDrill algorithm

While the MDP formulation appeared to be a viable approach for solving the drilling optimization problem, its application can be limited because of the many time steps it spends exploring non-optimal actions. It turns out that

two main simplifying assumptions can be made that reduce the problem from a MDP problem to a simpler k-arm bandit problem, where we only care about actions, not states: i) at each time step, large action transitions can be made, so that navigating the WOB-RPM space is not limited to incremental actions, ii) the probability of transitioning to a particular (WOB, RPM) state is 1. These two assumptions essentially imply that the state space of (WOB, RPM) can simply become the action space, if the driller does not interact with the environment directly, but rather through a controller that ensures a smooth, successful transition from each action pair. This formulation is exactly what was we proposed in Chapter 1 where, through feedback and state estimation using IMDT, the controller ensures that critic's set-points are accurately executed.

In a k-arm bandit problem, the agent is repeatedly faced with the option of k different actions to choose from. After each action is taken, a numerical reward is received from a probability distribution. The objective is to maximize the total expected payout over an extended time-period. In the drilling problem, the action space is now a two dimensional continuous space of (WOB, RPM) values, and the agent tries to maximize its payout (total drilling time) by finding the optimal actions. The agent still faces the dilemma of exploration vs. exploitation while deciding to choose the optimal (greedy) action vs. learning about the reward associated with other actions (Sutton and Barto, 1998).

While the problem has been simplified with the states removed, we

still want to draw inspiration from the Dyna formulation in order to adopt the concept of model-based reinforcement learning. Therefore, instead of simply aiming to deploy an optimization routine on the actual environment, we develop a model-based approach, where the agent simultaneously learns a model of the environment, and then uses this model to find optimal actions. The benefit of this approach compared to direct interaction with the environment is obvious: through the use of a model it can benefit from *apriori* information about the environment, which is often available from offset wells. We call this approach the *qDrill* algorithm. The qDrill agent, uses model learning (function approximation) to generalize its understanding of the action space, rather than visiting every discrete action value. To find greedy actions in feasible time, the qDrill agent uses heuristic optimization to avoid exhaustive search. The qDrill algorithm is summarized as follows:

Do Forever:

1. **act**: the agent takes an action $A_{wob,rpm}$ and receives a numerical reward $\mathcal{R}_t$.

2. **learn**: the action-value model $Q(a)$ is retrained, when $Q(a)$ is augmented with $Q_{t+1} = Q_t + \alpha[\mathcal{R} - Q_t]$

3. **plan**: Using heuristic search, find the greedy action such that: $A_{gr} = \arg\max_a Q(a)$

In Figure 3.4, we provide a preliminary comparison of the qDrill and the MDP approach in terms of the time that it takes each approach to drill a

$90ft$ stand. Evidently, the qDrill formulation performs better than the MDP approach, which can be attributed to its freedom to explore the action space faster. In the following sections, we will therefore focus on the qDrill algorithm and specifically on analyzing the learning and optimization tools used in the qDrill algorithm, as well as various techniques to solve the exploration vs. exploitation dilemma.

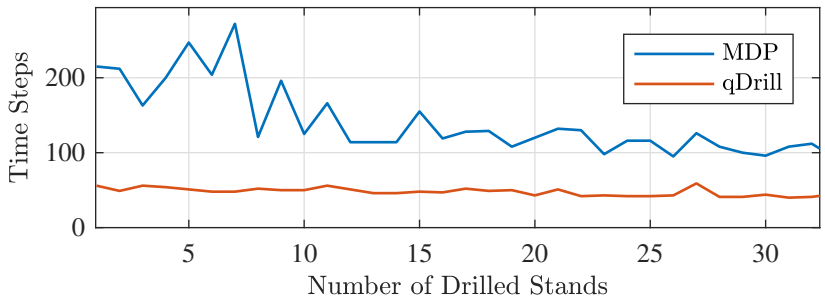

Figure 3.4: A comparison of the learning curve of the two RL formulations for drilling optimization.

3.3 Function Approximation of Action Values

When an action a is taken, the agent performs a re-run of the training process which is referred to as an epoch. At time step $t = 1$, the action $a_{t=1}$ is taken, and a reward $\mathcal{R}_{t=1}$ is received. In a non-stationary environment, the action value function $Q(a)$ is updated such that:

$$Q_{t+1} = Q_t + \alpha[\mathcal{R} - Q_t] \tag{3.3}$$

where α is the learning rate. From the simulated model of the environment, we have already learned that the relationship between an action $a(wob, rpm)$ and

its value is quite a nonlinear one, as portrayed by Figure 3.1. For the batch learning of this relationship, we resort to a few nonlinear regression techniques. We employ and compare the performance of three different techniques, the artificial neural network (ANN), the bagged tree regression (BTR), and support vector regression (SVR).

For the learning task using an ANN, we used a three-layer network, an input, one hidden layer with N_h nodes, and an output layer. The input layer was augmented with a bias node. For the hidden layer activation a $tanh()$ function was used, and for output a linear activation function.

In a support vector regression framework, the basic idea is to find a linear function:

$$f(x) = x'w + b \tag{3.4}$$

where x is the training data, to be flat as possible (Smola and Schölkopf, 2004). This can be written as a convex optimization problem

$$J(w) = \frac{1}{2}w^T w \tag{3.5}$$

subject to residuals having a margin less than ϵ such that:

$$|y_n - (x_n^T w + b)| \leq \epsilon \tag{3.6}$$

The linear ϵ-insensitive loss function ignores errors that are within the ϵ boundary . The described optimization problem is effectively solved in the Lagrange dual formulation, using the dual problem to obtain a lower bound to the solution of the primal (Smola and Schölkopf, 2004).

Nonlinear regression using the support vector method can be achieved using the kernel trick. such that the dot product $x_1^T x_2$ is replaced with a nonlinear kernel function $G(x_1, x_2) =< \phi(x), \phi(x_2) >$ where $\phi(x)$ is a transformation that maps x to a high-dimensional space (Huang et al., 2006). For this task, we used a gaussian kernel such that:

$$G(x_j, x_k) = e^{(-||x_j - x_k||^2)} \tag{3.7}$$

Tree based regression models are also popular nonlinear function approximation methods, due to their simplicity and efficiency. Regression trees are trained using a fast divide and conquer algorithm that recursively partitions the data into smaller subsets (Hastie and Tibshirani, 1990). A regression tree can be viewed as an additive model of the form:

$$m(\mathbf{x}) = \sum_{i=1}^{l} k_i \times (x \in D_i) \tag{3.8}$$

where k are the constants and $I()$ is an indicator function returning 1 if its argument is true, and 0 if it is false and D_i are disjoint partitions of the data. To predict a response, the algorithm starts from the root of the tree down to a leaf node. While individual regression trees tend to overfit, bootstrap-aggregated (bagged) regression trees (BTR) can improve performance by combining the results of many regression trees, which can improve generalization. Tree Bagging selects a random subset of predictors to use at each decision split as in the random forest algorithm (Breiman, 2001).

Prediction of the action values using a regression model is relatively straightforward, given the action space is only two dimensional. In addition,

the sufficiently long value of ΔT ($> 30s$) provides ample time for the agent to perform batch learning at each time-step online, using data available up until that time-step. An important consideration for the choice of the regression model, however, is its ability to generalize well early on, to only a few data-points, which will enable faster convergence to the optimal action set. In Figure 3.5, 100 actions were sampled from a 50×50 discretization of the environment's ROP space , and used to train three different regression models. With only a 100 actions, it is expected for much of the ROP space to remain unrepresented. As evident in Figure 3.5, the ANN model with 20 hidden layer neurons performs poorly relative to BTR and SVR.

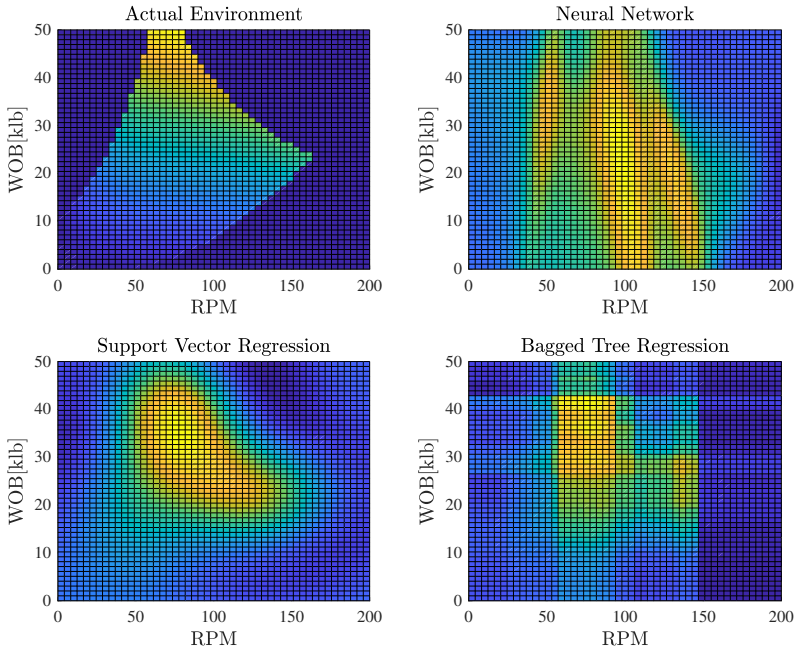

Figure 3.5: Agent's action-value function after 100 time steps with random actions.

To conduct further analysis, we focus on comparing the SVR and BTR methods and their performance with small and larger training data samples. To determine the optimal number of trees in the BTR model with a larger training data size, the study in Figure 3.6 was performed. As the prediction results demonstrate, 50 bagged trees yielded the best results regarding MSE, with > 50 number of trees slightly overfitting and providing sub-optimal performance.

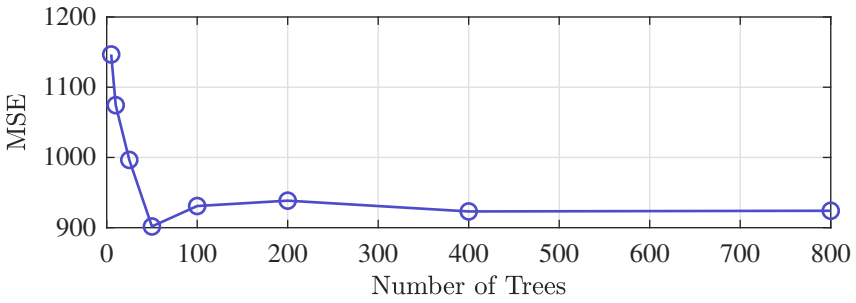

Figure 3.6: Mean squared prediction error results of the bagged tree model with various number of regression trees.

For a closer comparison of BTR and SVR, let us refer to the plot in Figure 3.7. As already evident from Figure 3.5, the SVR model performs better than BTR when the data consists of 100 actions. However, the performance of BTR with 1000 actions is improved drastically, while the SVR's prediction MSE decreases only by a small amount. This analysis tells us that the preferred method of regression learning for action values is the SVR method due its superior performance at the early stages of learning. However, the BTR method could also be used in the long-term, once the agent has obtained a

sufficiently large data-set from the environment.

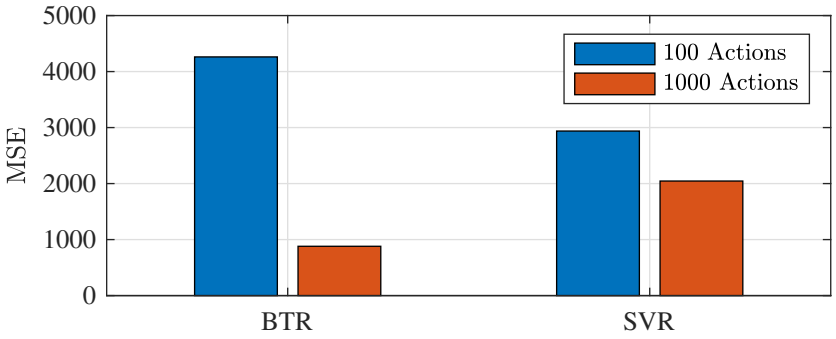

Figure 3.7: Mean squared prediction error results of the BTR (50 trees) and SVR models.

We have already noted that while training is performed online, the large ΔT between actions allows us to use batch learning in between the time-steps. With that said, the training and prediction times of each algorithm are still of considerable importance. In Figure 3.8, the better computational performance of the SVR algorithm is highlighted regarding both training and prediction.

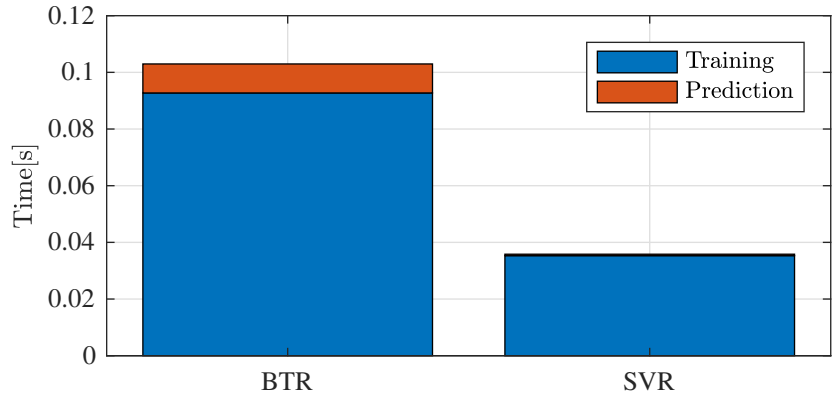

Figure 3.8: Computational time requirements to train and predict with the BTR and SVR models.

With sufficient analysis, SVR has shown to be a better choice because of its performance with small-size data, as well as its more favorable computational performance. Although the BTR method can be further explored for field implementation and longer experiments, for the rest of this chapter we will use the SVR to conduct the simulations.

3.4 Heuristic Search for Greedy Action Selection

In the planning stage of the qDrill algorithm, the agent has to find the greedy action set using the action value function model that was learned using the aforementioned regression techniques. For this purpose, we will resort to a heuristic search technique for a few reasons, i) the models presented in the previous technique are nonlinear, and therefore a closed-form solution for the optimal action set does not exist, and ii) the heuristic search methods do not require the knowledge of a gradient matrix (unlike gradient decent for instance), and therefore the global solution can be found in feasible time.

Particle swarm optimization (PSO) is one popular heuristic search technique that is inspired by social behavior of birds flying in search of food resources, and has been widely applied in engineering problems. While PSO is not the only heuristic search technique for gradient-free optimization, it is suited well to this particular problem. For instance, unlike the genetic algorithm (GA), it is not meant for combinatorial optimization problems, and therefore does not require the discretization of the solution space. In PSO, several particles are randomly dispatched over the search space, and each particle

is associated with a position vector $\mathbf{x}$ which represents a candidate solution. At every iteration of the algorithm, the position of each particle is updated based on its velocity, which is a function of the particle's own best performance, as well as the group's best global performance (Bryson et al., 2016). After many iterations, all the particles converge to the optimal solution which is the group's global best. Specifically, the velocity of the i^{th} particle at time $t+1$ is calculated as:

$$\mathbf{v}_{t+1}^{i} = w\mathbf{v}_{t}^{i} + C_1 n_{r_1}(\mathbf{P}_{t}^{i} - \mathbf{x}_{t}^{i}) + C_2 n_{r_2}(\mathbf{P}_{t}^{g} - \mathbf{x}_{t}^{i}) \tag{3.9}$$

where n_r is a random number drawn from a uniform distribution from 0 to 1, and w, C_1 and C_2 are the inertia, self-confidence, and swarm influence weighting terms respectively. To ensure convergence, the eigenvalues of the dynamic PSO system have to be stable. For stability, the weighting factors are bounded according to the following (Bryson et al., 2016):

$$0 < (C_1 + C_2) < 4 \tag{3.10}$$

$$\frac{C_1 + C_2}{2} - 1 < w < 1 \tag{3.11}$$

The self-confidence parameter affects how much each particle explores a local optimum before being influenced by the solutions found by other particles. A swarm influence parameter that is too high, however, may result in all particles converging too quickly to a sub-optimal solution. Figure 3.9 demonstrates graphically the nature of the position update of a particle at each iteration.

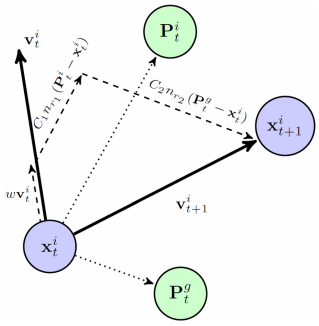

Figure 3.9: A visual representation of the PSO algorithm. At every iteration of the PSO, a particle with position x_t moves towards its position x_{t+1} with a velocity v_{t+1} which is vector sum of its social (towards the global best, P^g) and personal (towards the personal best P^i) velocities (Bryson et al., 2016).

Another important parameter β determines the population size, which also affects the convergence time of the optimization algorithm. In Figure 3.10, the results of PSO applied to finding the optimal action set using a SVR model trained with 1000 actions is shown. As already expected, the algorithm converges to the optimal solution faster at higher values of β, which however, comes at an added computational cost. Therefore, for this particular problem, $10 < \beta < 25$ is a suitable choice.

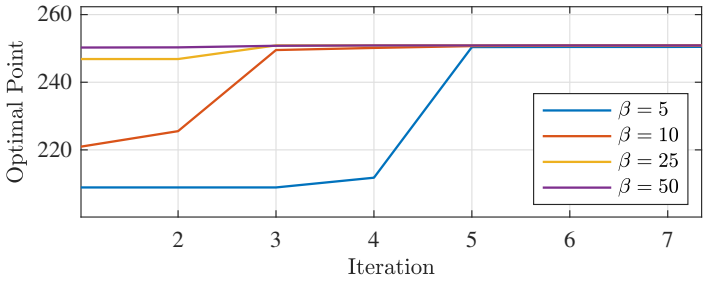

Figure 3.10: PSO convergence results with different values of population size β.

3.5 Exploration Strategies

In the previous sections of this chapter, we addressed how the agent uses regression learning to learn a model of the action values, and how it uses this model to find optimal action sets. An important aspect of the qDrill algorithm that remains to be discussed is the action selection procedure, where the agent has to deal with the dilemma of exploration vs. exploitation. At the start of every iteration, the agent has two choices: either to select the greedy action, or choose an action randomly. If at every iteration the greedy action A_{gr} is selected, the algorithm learns very little about its environment and gets stuck in local optima. On the other hand, repeated selection of random actions can affect short and long term performance of the agent, as the goal of the agent is performance maximization after all. A simple strategy to handle this dilemma, is an ϵ-greedy strategy, where (Sutton and Barto, 1998):

$$A \leftarrow \begin{cases} \arg\max_a Q(a) & \text{, with probability } 1-\epsilon \\ \text{random} & \text{, with probability } \epsilon \end{cases}$$

The effect of different ϵ values on the performance of the qDrill algorithm is explored in Figure 3.11. The top plot in Figure 3.11 corresponds to the total reward accumulated by the agent, which is proportional to the total distance drilled. In the lower plot of Figure 3.11, the derivative of the total reward is shown. A trivial observation from Figure 3.11 is that with $\epsilon = 0.5$, the agent performs poorly relative to the other cases, as the it spends too much time exploring low-value actions. With $\epsilon = 0.01$, the agent quickly begins to exploit its knowledge of ROP, and does not dedicate much effort to finding optimal

actions (this is shown by the fairly steady reward plot, as compared to the case of $\epsilon = 0.5$). With $\epsilon = 0.25$, the agent's performance is interestingly less predictable. Initially, it performs worse than both $\epsilon = 0.1$ and $\epsilon = 0.01$. Noting the reward plot, the agent with $\epsilon = 0.25$ begins to find more valuable actions over time, slowly over-takes $\epsilon = 0.1$, and at around $t = 250$, has eventually found more optimal rewards than the agent with $\epsilon = 0.01$.

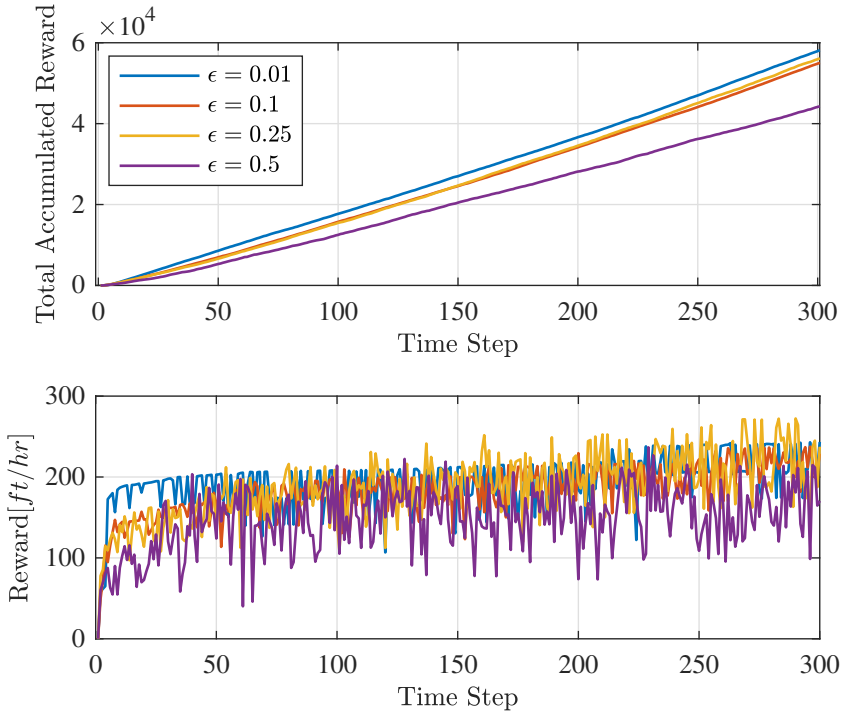

Figure 3.11: Performance results of 4 different qDrill agents with different ϵ-greedy exploration strategies. The results are averaged from 10 experiments for each agent.

The results shown in Figure 3.11 support the initial intuition we had about the performance of various ϵ-greedy strategies. We had suspected that

too much of exploration or exploitation would negatively impact the performance of the agent, and that a "sweet spot ϵ" would yield to the best performance.

Another way to improve the performance of an ϵ-greedy strategy is to vary ϵ dynamically based on some heuristic. A simple heuristic can be the number of time-steps since the agent began learning about its environment. With a dynamic ϵ, the agent explores more initially, but slowly decreases its exploration to focus on greedy actions. ϵ can also be changed dynamically as a function of a particular event, such as a change in formation hardness.

In addition to a dynamic epsilon, it would also make sense for the agent to explore regions of the search space where there is high potential for optimal actions, rather than choosing random actions. In a tabular formulation, this can be achieved through the use of the upper-confidence bound technique, N_A (the number of time-steps an action has been taken) is kept track of (Sutton and Barto, 1998). This provides the agent with a measure of the uncertainty in the value of a particular action. However, in the case of a continuous formulation, the upper confidence bound technique is not applicable. We can, however, analyze the record of past high performing actions through the use of unsupervised learning techniques. The idea is simple. Every time the agent decides to explore the environment, it first creates a two-dimensional cluster of its past actions, rather than blindly choosing a random action. Amongst these clusters, the best performing one (found by evaluating the value of the centroid of a cluster) inevitably corresponds to the greedy region of the action-

space, which has been exploited by the agent many times. The lower potential clusters in rank correspond to the cluster of actions that have been tried before, but were not as rewarding as the greedy action. The agent selects the top k clusters, and samples a random action from the distribution spanned by the members of these clusters.

The concept described above is visually described in Figure 3.12, where the agent takes on two different exploration strategies, while learning the ROP environment of Figure 3.5. The first strategy is ϵ-greedy with $\epsilon = 0.25$, and the second, called a Smart-greedy strategy, employs an initial ϵ of 0.5 for the first 50 time steps, and a reduced ϵ of 0.1 after that. In addition, the agent only samples its random actions from the highest performing of $k = 6$ past cluster of actions.

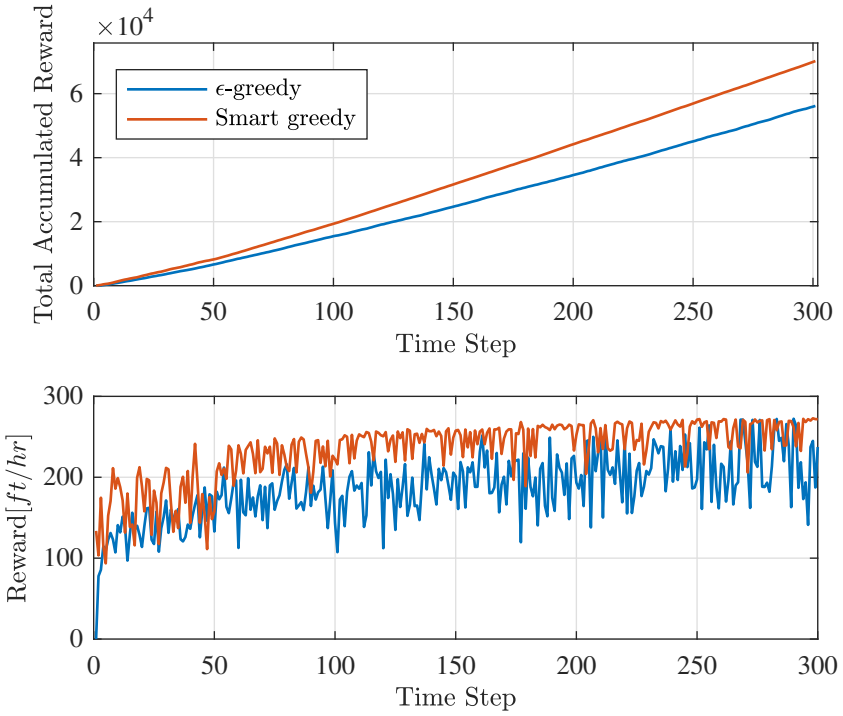

Figure 3.12: Performance results of two different qDrill agents with different exploration strategies. The Smart greedy agent uses k-means clustering to distinguish promising actions, and focuses its exploration efforts. The results are averaged from 10 experiments for each agent.

Evidently, the initial high rate of exploration gives the Smart-greedy agent the benefit of faster initial learning. With a reduced ϵ over time, as well as the strategic focus on high potential regions of the ROP space, the Smart-greedy algorithm easily outperforms the simpler ϵ-greedy strategy.

3.6 Adaptation to Environment Changes

The strategies discussed above appear to have a clear winner: the Smart-greedy strategy that picks the best performing cluster to explore within. It would, however, be interesting to evaluate the performance of each strategy in a non-stationary setting, where the environment goes through a major change. After all, the whole purpose of the iterative qDrill algorithm is to continuously adapt its behavior to address environment changes. In the following section, we will apply qDrill to two scenarios: one where the environment changes in a positive way (through formation change), and the other where the environment changes in a negative way through the tightening of the vibration constraints (through formation change, change in drill-bit cutter characteristics, etc.). Figure 3.13 demonstrates 3-dimensional operating region associated with each formation.

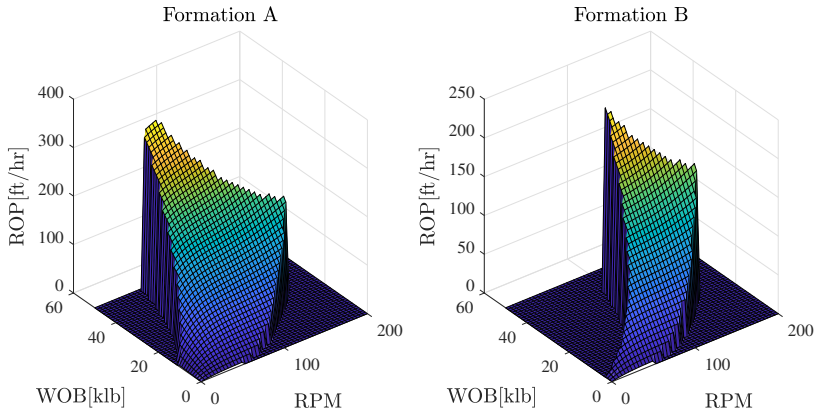

Figure 3.13: A 3D WOB,RPM state space representation of two different formations encountered by the agent. Formation A has wider vibration constraints, thereby allowing higher rates of penetration.

When the environment changes for the worse, the agent immediately observes the impact of this change upon its received reward. However if prior to the environment change the agent has spent all of its time exploring high-potential actions, its understanding of the rest of the state-space is weak. Therefore, the agent will struggle to find rewarding actions after the environment changes. This idea is conveyed by the experiment shown in 3.14. Two exploration strategies are used: first is the familiar ϵ-greedy strategy with $\epsilon = 0.25$, and the second is the described Smart-greedy strategy where Υ is the probability that the agent emphasizes on the second best performing cluster of past actions. Initially, before the environment changes, a lower value of Υ is doing better since the agent is strictly focused on the best performing cluster of past actions. However, after the change, the agent with $\Upsilon = 0.75$ begins to collect non-zero rewards much faster, since it had taken the time to learn a more diverse set of high performing actions. The learning of the ϵ-greedy is slower than the Smart greedy strategy with $\Upsilon = 0.75$ both before and after the environment change.

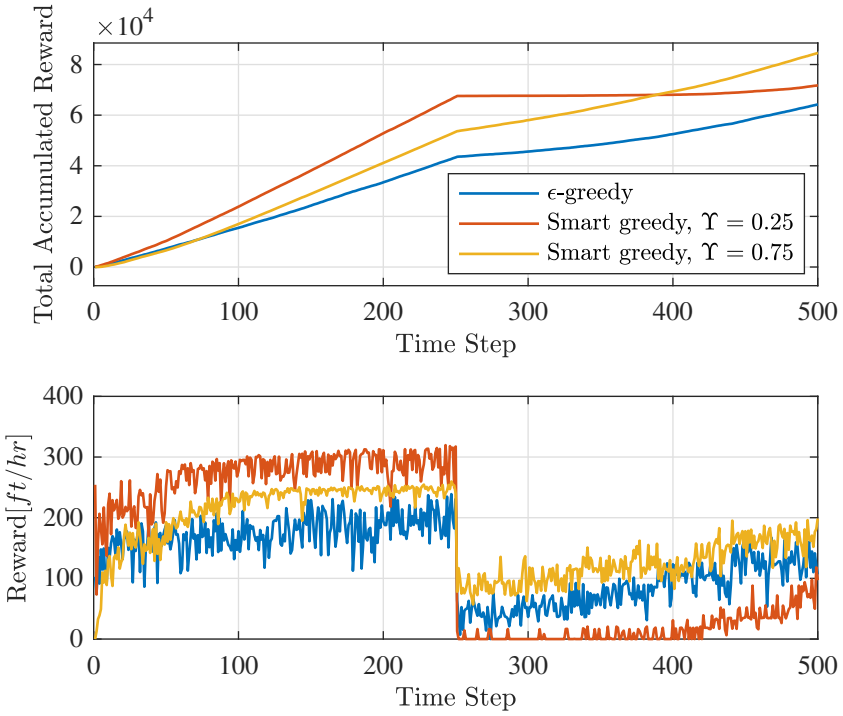

Figure 3.14: Performance results of three different qDrill agents with different exploration strategies, in the case of transitioning from Formation A to B. The Smart greedy agent give preference to second-best performing past actions with probability Υ . The results are averaged from 10 experiments for each agent.

The opposite scenario is when the environment changes for the better (more forgiving dysfunction constraints) from Formation B to Formation A. In this case, as shown in Figure 3.15, all three agents identify the change and begin to obtain higher reward immediately. In this case, however, the ϵ-greedy agent is the worst performing of the three, both before and after the change. The $\Upsilon = 0.75$ agent beats the $\Upsilon = 0.25$ agent after the formation change, because of its more diverse understanding of past high-potential actions.

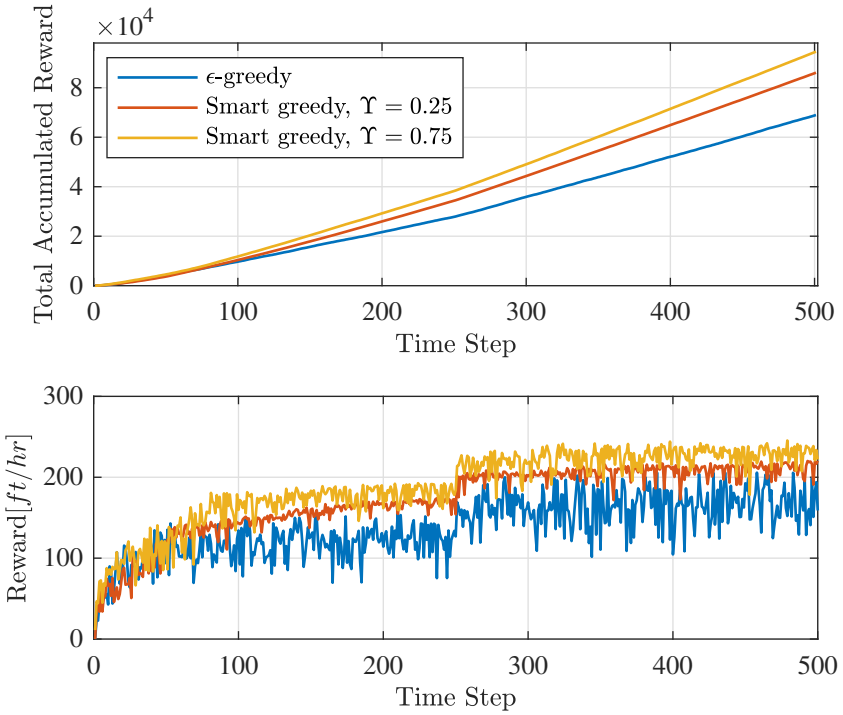

Figure 3.15: Performance results of three different qDrill agents with different exploration strategies, in the case of transitioning from Formation B to A. The Smart greedy agents give preference to second-best performing past actions with probability Υ . The results are averaged from 10 experiments for each agent.

In essence, while the Smart greedy technique is still a better performing technique than the simpler ϵ-greedy strategy, a modification to the idea was required to ensure that it can effectively handle environment changes. It was shown that exploration in the highest-performing cluster of past actions is not the best choice, and that the agent actually performs best when it explores promising past actions, but not the most promising.

3.7 Tripping Optimization

In the last section of this chapter, the optimization problem is considered for the case of pipe tripping. The optimization process in tripping differs from the drilling task in that it does not involve simultaneous learning since no sensory feedback is available during this process. However, the physics-based model of Chapter 2 can be reliably used for prediction and design of an optimal tripping trajectory that satisfies downhole pressure constraints, given that the parameter estimation of the model has been completed during the pumps-on period (when standpipe pressure is available). Mathematically, the virtual driller's goal is to solve the following optimization problem before tripping in/out of hole:

$$\begin{aligned} &\text{minimize} \quad t_{trip}(v_{des}, t_{ramp}) \\ &\text{subject to} \quad \delta P_t(v_{des}, t_{ramp}) \leq P_{max},\ t = 0, \ldots, t_{trip}. \end{aligned} \tag{3.12}$$

where t_{trip} is the total tripping time for each stand, v_{des} is the steady state tripping velocity, t_{ramp} is the ramping time to the desired velocity, δP_t is the dynamic downhole pressure change due to tripping, and P_{max} is the maximum allowable pressure drop/gain that can be tolerated before the bottom-hole pressure goes below the pore pressure, or exceeds the fracture gradient (the pressure drop occurs when the drillstring is tripped out, and the pressure gain occurs when the drillstring is tripped in) . The velocity trajectory of the tripping process is assumed to be of a trapezoidal shape, where a stand of pipe is accelerated to a constant desired velocity, and decelerated at the same rate to a stop (In Chapter 2, this trajectory appeared to be optimal for parameter

learning purposes). This trajectory can be designed such that the area of this trapezoid is equal to the total length of a stand. δP_t is an output of the learned hydraulics model, and t_{trip} can be found as follows:

$$\begin{aligned} t_{trip} &= t_{const} + 2t_{ramp} \\ t_{const} &= \frac{d_{const}}{v_{des}},\ d_{const} = d_{trip} - 2d_{ramp} \\ d_{ramp} &= \frac{\phi_1}{4}t_{ramp}^4 + \frac{\phi_2}{3}t_{ramp}^3 + \frac{\phi_3}{2}t_{ramp}^2 + \phi_4 t_{ramp} \end{aligned} \tag{3.13}$$

where $d_{trip} \approx 90ft$ is the length of a stand, and $\phi_1 ... \phi_4$ are the parameters of a smooth cubic function, which are found by solving the following system of linear equations:

$$\Phi_{trip} = A_{trip}^{-1} b_{trip} \tag{3.14}$$

where,

$$\Phi_{trip} = \begin{bmatrix} \phi_1 \\ \phi_2 \\ \phi_3 \\ \phi_4 \end{bmatrix} \tag{3.15}$$

$$A_{trip} = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ t_{ramp}^3 & t_{ramp}^2 & t_{ramp} & 1 \\ 3t_{ramp}^2 & 2t_{ramp} & 1 & 0 \end{bmatrix}, b_{trip} = \begin{bmatrix} 0 \\ 0 \\ v_{des} \\ 0 \end{bmatrix}, \tag{3.16}$$

The optimization problem presented in Equation 3.12 can be solved through prediction using the learned hydraulics model, and the relationships presented in Equations 3.13 through 3.16. Since the problem is inherently nonlinear, no closed-form solution is available. The problem can, however, be solved using the same heuristic search method the virtual driller employs

while drilling, namely the PSO technique. The optimizer accepts as input a pressure constraint, P_{const}, and returns the optimal tripping trajectory that satisfies this pressure constraint. In Figure 3.16, tripping optimization results using the PSO technique are presented with $P_{const} = 150\,psi$. It can be seen that the optimal trajectory converges to a time of $35.31\,s$, in 10 iterations with $\beta = 25$ and 50. With a two-decimal point discretization of a $1\,m/s \times 30s$ search space, an exhaustive search would have required 3×10^5 simulations to find the optimal trajectory, as opposed to the 500 simulations performed by the PSO with $\beta = 50$.

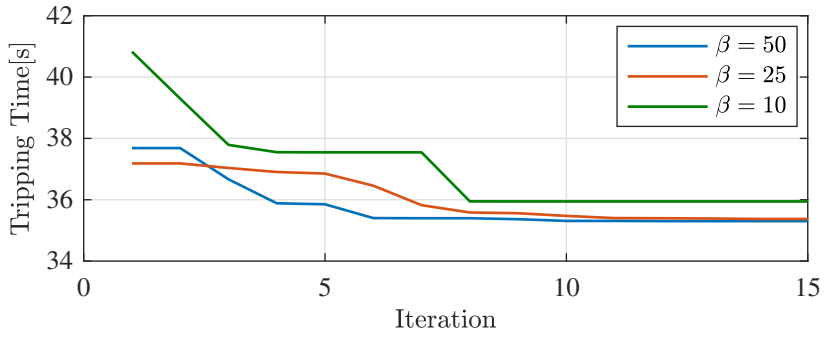

Figure 3.16: PSO convergence results for tripping time optimization with different values of population size β.

3.7.1 Accounting for Uncertainty

The formulation above is a simple yet powerful approach that enables time-optimal pipe tripping, without relying on real-time feedback during the tripping process itself. However, the optimization results presented in Figure 3.16 assume deterministic model parameters when predicting the pressure re-

sponse of the environment. To address and overcome this assumption, let us refer back to Chapter 2 and consider the unknown/uncertain hydraulic parameters, K, PV and YP. The estimated model parameters, through learning can be approximated as:

$$\begin{aligned} K &= \mu_K \pm \sigma_K \\ PV &= \mu_{PV} \pm \sigma_{PV} \\ YP &= \mu_{YP} \pm \sigma_{YP} \end{aligned} \tag{3.17}$$

where σ is the standard deviation of the parameter, approximated from its time-domain estimation. Upon inspection of Equation 2.18, it is trivial to see that a "95% worst case" pressure change scenario occurs when $PV = \mu_{PV} + \sigma_{PV}$, and $YP = \mu_{YP} + \sigma_{YP}$. Similarly, a "95% best-case" is obtained when $PV = \mu_{PV} - \sigma_{PV}$, and $YP = \mu_{YP} - \sigma_{YP}$. In the case of K however, the effect on the maximum pressure change is not obvious as the the parameter only affects the transient response of pressure, and not its steady state value. A risk averse virtual driller therefore:

- **For maximum risk (shortest trip time):** solves the optimization problem with $PV = \mu_{PV} - \sigma_{PV}$, and $YP = \mu_{YP} - \sigma_{YP}$, with the maximum pressure evaluated at $K = \mu_K \pm \sigma_K$.
- **For minimum risk (longest trip time):** solves the optimization problem with $PV = \mu_{PV} + \sigma_{PV}$, and $YP = \mu_{YP} + \sigma_{YP}$, with the maximum pressure evaluated at $K = \mu_K \pm \sigma_K$.

Figure 3.17 demonstrates the risk-averse tripping methodology. Through the consideration of the uncertainty in the physics-based model's parameters, and based on the desired level of risk, the virtual driller can design three different tripping trajectories, all of which satisfy the maximum pressure change of $150\,psi$. The best case solution is faster than the worst case solution by $1.5\,s$.

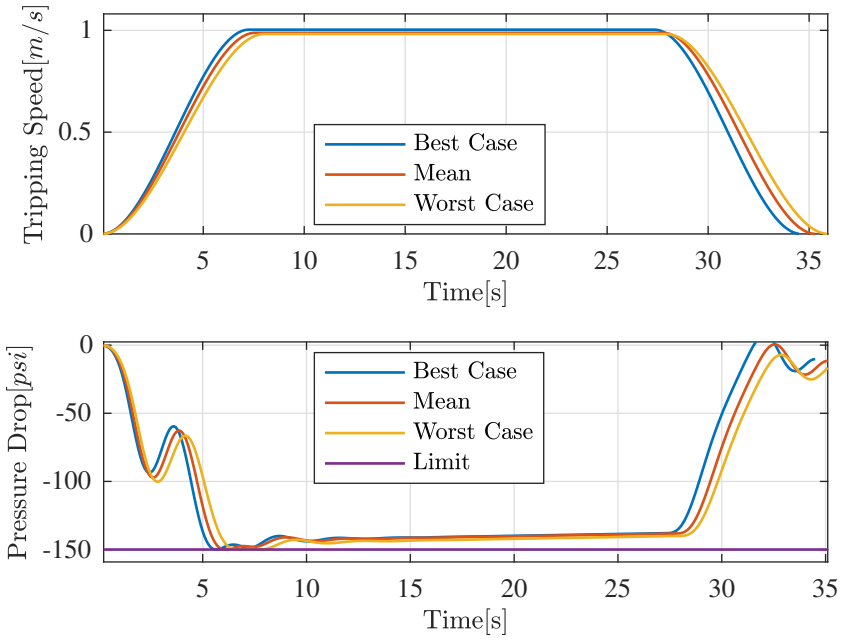

Figure 3.17: Optimal tripping trajectories for three different cases of risk consideration. To ensure the satisfaction of pressure constraints, the worst-case solution is slower by $1.5\,s$. Each uncertain parameter of the hydraulic model has a $\pm 10\%$ uncertainty.

The reader can easily see that the uncertainty of the pressure constraint itself can be included in the risk-averse formulation, where P_{max} is represented by: $\mu_{P_{max}} \pm \sigma_{P_{max}}$. The base case scenario can then be evaluated at $\mu_{P_{max}} +$

$\sigma_{P_{max}}$, and worst case scenario at $\mu_{P_{max}} - \sigma_{P_{max}}$.

3.8 Conclusion

In this chapter, a solution to drilling optimization without complete *a priori* understanding of the drilling environment, specifically regarding the relationship between control inputs and performance metrics, is presented. The reinforcement learning approach enables the use of a regression model (trained by the action values observed while interacting with the environment) for real-time optimization. The solution is also shown to be effective in dealing with a non-stationary environment, when various parameters such as formation or drill-bit properties change. Additionally, a heuristic -based solution for the tripping optimization was presented, which used the model learned in Chapter 2 to optimize the process with respect to time. In the next chapter, the problem of tracking the *Optimizer's* set-points is explored.

Chapter 4

Controller: Robust, Closed-Loop Tracking of *Optimizer* Set-points

In the previous chapter, we concluded that the qDrill formulation cannot control the environment directly because of the assumption that taking a desired action puts it into the state $s = A$, with $\mathcal{P} = 1$. Therefore, as proposed in Figure 1.2, the critic would first assign a set-point to the controller and, through closed-loop feedback, the controller would ensure that the system state does in fact become the same as the desired set-point. The controller would have to ensure that a large transition in WOB and RPM is achievable. While downhole RPM tracking and active mitigation of torsional stick-slip vibrations is a popular research area in drilling, our emphasis in this dissertation will be on the complications that arise in controlling downhole WOB by adjusting the drawworks feed-rate velocity[1]. Hence, we have assumed that by avoiding drilling dysfunctions, and with the use of an appropriately tuned top-drive controller, any downhole RPM is attainable.

[1]The work in Section 4.4.2 has been presented in the conference publication: P. Pournazari, B. R. Fernandez, and E. van Oort. Robust weight-on-bit tracking in drilling operations: A stochastic, nonlinear approach. In *ASME 2017 Dynamic Systems and Control Conference*, pages V003T43A003-V003T43A003. American Society of Mechanical Engineers, 2017. The author of this dissertation contributed to that work through theoretical analysis, controller design, and simulation studies.

4.1 Introduction

Accurate WOB tracking is important for an automated drilling system, not only due to its impact on the performance of ROP optimization routines, but also because oscillatory action of downhole WOB results in bit torque fluctuations that can potentially induce torsional vibrations. In a study by Pastusek et al. (2016), the authors mention several shortcomings associated with present-day auto-drillers (a closed-loop drilling controller that most commonly aims to track a desired WOB set-point by getting feedback from the hook-load sensor). Most systems on the market rely on basic first-order dynamics and utilize controllers based on bang-bang or PID techniques. As a result, such systems do not take into account the complex dynamics associated with the drillstring compliance, the wellbore friction forces, fluid viscosity, and bit-rock interaction. Furthermore, uncertainty associated with hook-load measurements is not quantified, and its effects on controller performance are unknown. Finally, the controller gains have to be frequently tuned to provide acceptable performance (Pastusek et al., 2016).

To investigate the problems regarding the relationship between poor WOB control and torsional vibrations, the study in Figure 4.1 is conducted. After tagging bottom and increasing WOB, the oscillatory behavior caused by poor controller tuning results in severe torque-on-bit fluctuations. These fluctuations force the bit into a stick-slip cycle, where the bit stores enough energy to speed up to almost three times the RPM-set-point, and then decelerates back to zero where the sticking occurs. The stick-slip cycle essentially

continues until the WOB has stabilized.

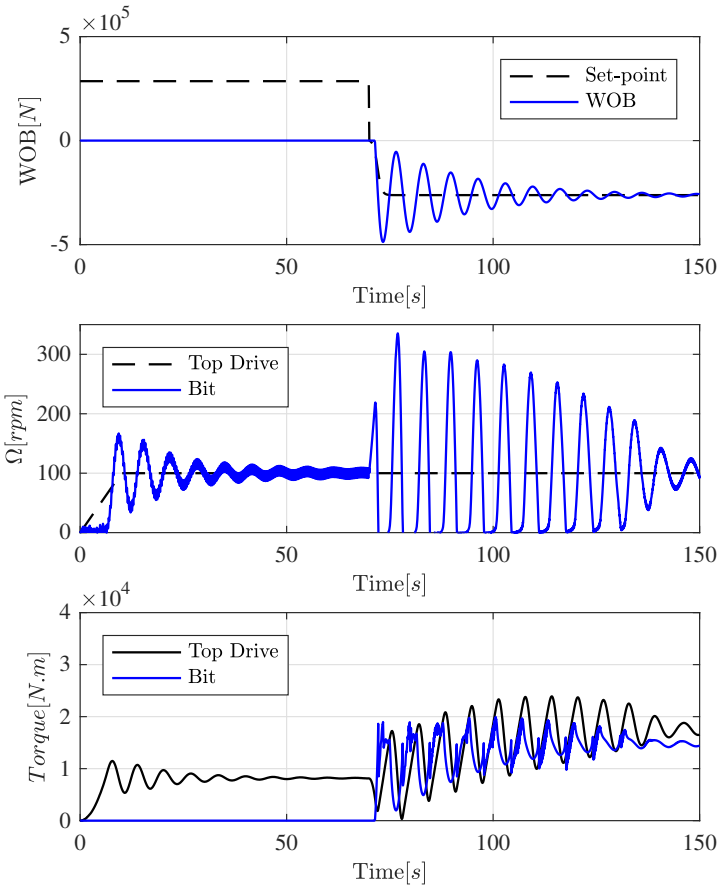

Figure 4.1: Simulation of torsional stick-slip vibrations (middle plot), as a result of TOB fluctuations (lower plot), when the WOB controller behaves poorly (upper plot).

A much better RPM response is observed in Figure 4.2, where the WOB only marginally overshoots after tagging bottom. The torque-on-bit, as seen in the lower plot of Figure 4.2, has a smoother transition to the steady-state

torque. Therefore, the bit only experiences a $< 30s$ stick-slip cycle.

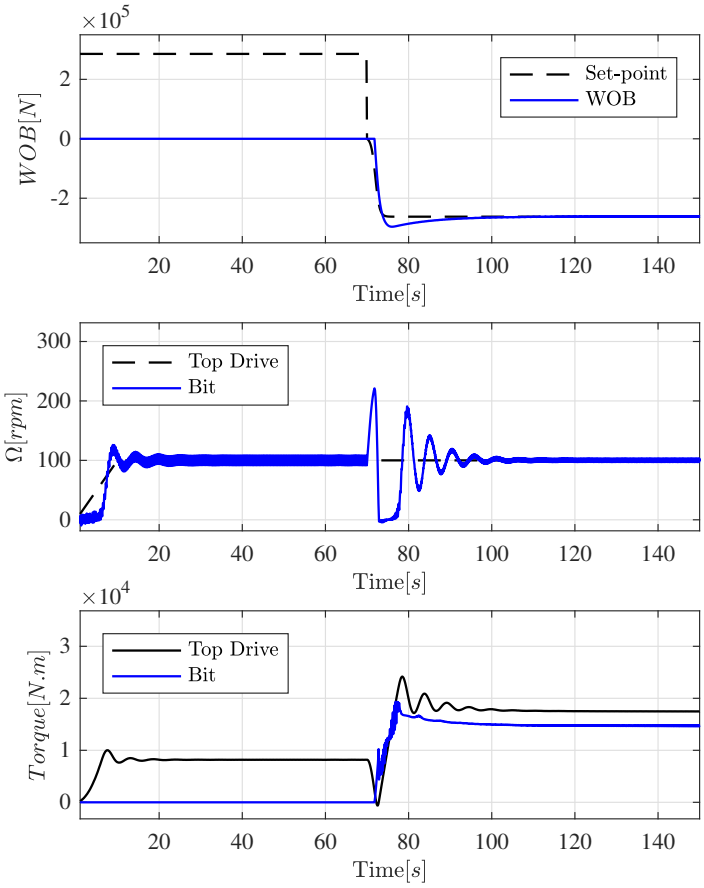

Figure 4.2: Simulation of torsional stick-slip vibrations (middle plot), as a result of TOB fluctuations (lower plot), when the WOB controller behaves well without significant overshoot (upper plot).

The simulations above demonstrate that the performance of a WOB controller considerably affects torsional vibrations, which could result in poor drilling performance and reduced bit life. Robust WOB tracking is therefore

also important regarding its coupling to torsional dynamics.

This chapter will focus on developing the necessary tools that equip the drilling automation system with the ability to perform robust set-point tracking of downhole weight-on-bit (WOB). The idea in this chapter is that through model learning (as addressed in Chapter 2), a model-based robust controller can be automatically re-designed at every bit run. This controller would then take into account the approximate dynamics of the system, and enable robust WOB tracking. We begin by introducing the problem and summarizing the motivation for a robust WOB controller. In Section 4.1.1, an overview of various previously studied control approaches for WOB tracking is presented. Section 4.3 presents several different control methodologies based on linear control theory. A more elaborate nonlinear controller is then presented in Section 4.4, which improves system performance in the presence of nonlinear frictional forces. In Section 4.5, we compare the performance of all controllers in terms of robustness to parametric uncertainty and tracking capability. In the end of this chapter, we study the full control problem including downhole disturbance rejection, and make recommendations for practical field implementation.

4.1.1 Literature Review

Boyadjieff et al. (2003) presented a linear control system for WOB tracking and a linearized plant model that includes an empirical ROP model based on the work in Jorden et al. (1966). Their model also takes into ac-

count the dynamics of the traveling equipment and the drawworks braking system. Field test results in this paper demonstrate significant improvements over similar wells without an auto-driller system. A shortcoming of this work is the missing treatment of wellbore friction forces, which makes their approach inapplicable to directional drilling. In addition, the authors do not address controller robustness when subject to plant variations such as changes in formation hardness.

The work in Yigit and Christoforou (2006) considers a similar problem, but from a bit-bounce mitigation perspective. The authors develop a lumped parameter model of the drillstring, but bit-rock interaction is ignored since they are only concerned with stabilizing the axial movement of the drill-bit. Similar to the previous paper, consideration of nonlinear damping forces is missing. A linear controller is presented based on pole-placement techniques, but observer design (therefore treatment of disturbance from bit-rock interaction) was not discussed.

The most recent advance in WOB control is achieved by Pink et al. (2013) via the use of downhole WOB measurements and real-time data transfer using wired-pipe telemetry. Even though the authors do not provide any details on modeling and control system design, their results demonstrate significant gains in performance regarding downhole WOB stability and tracking. Although the benefits of real-time downhole WOB measurements are well-understood as they provide the ability for true feedback control, the economic value of wired-pipe telemetry is not widely justified for lower cost well con-

struction.

The work in this chapter adopts an approach similar to that of Boyadjieff et al. (2003), but relies on a more comprehensive model of drilling dynamics to account for nonlinear friction forces during directional drilling. From a controls perspective, we develop a nonlinear full-state feedback controller as opposed to the linear method used in Yigit and Christoforou (2006) to identify and feedback-linearize the nonlinear dry friction forces. Also in comparison to the work of Yigit and Christoforou (2006), the controller in this chapter does not assume available measurements along the drillstring, and uses the Bayesian filtering technique outlined in Chapter 2 to estimate system states. Finally, unlike Pink et al. (2013), we do not rely on real-time downhole measurements for robust downhole tracking. With that said, in presence of wired-drill telemetry, the presented controller in this chapter can be complemented with real-time downhole data as well for improved performance.

4.2 Control Problem Formulation

The models used for controller design and testing were developed in Chapter 2. In addition, we outlined the state estimation tools that can be used to estimate unmeasurable states of the system from surface measurements. Figure 4.3 describes the controller architecture of interest that is developed in this chapter.

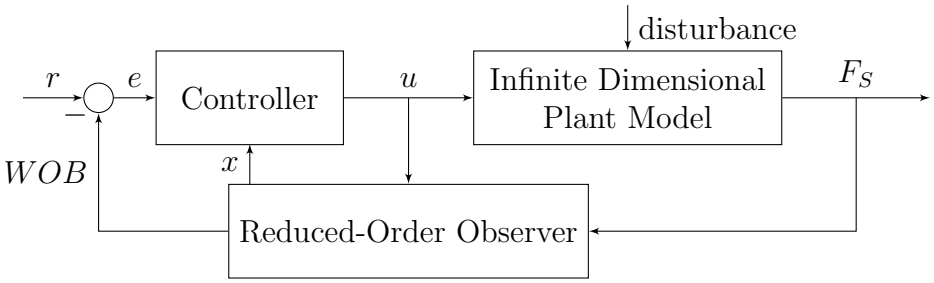

Figure 4.3: Block diagram of the WOB feedback controller. The reduced-order observer provides the system with an estimate of downhole WOB and the disturbance.

Later in this chapter, for testing of various controllers, the infinite dimensional model of Chapter 2 is used as a model of the plant to provide a realistic representation of the actual dynamics and delays of the environment. The proposed UKF-based state estimator is used to provide the controller with an estimate of the system states, with downhole WOB as the desired input to the controller. The controller then outputs the appropriate drawworks feed-rate to drive the system to the desired WOB set-point. For controller design, a reduced-order model of the axial drillstring dynamics is given by:

$$d\mathbf{x} = F(\mathbf{x}, u, t)dt + d\beta \tag{4.1a}$$

$$\mathbf{z}_k = H\mathbf{x} + \mathbf{v} \tag{4.1b}$$

where $\mathbf{z}_k$ is the measurement vector, H is the measurement matrix, β is a zero-mean random variable of variance q that represents process uncertainty, and v is the measurement uncertainty which can be written as:

$$E[\mathbf{v}] = 0, E[\mathbf{v}\mathbf{v}^T] = R = \sigma_{hkl}^2 \tag{4.2}$$

and,

$$H = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{4.3}$$

The state vector $\mathbf{x}$ is given by:

$$x = \begin{bmatrix} F_{ds} \\ v_{ds} \\ F_c \\ v_c \\ F_{fr} \end{bmatrix} \tag{4.4}$$

and the transition function, $F(\mathbf{x})$ by:

$$F(\mathbf{x}) = \begin{bmatrix} k_{ds}\big(v_{input} - v_{ds}\big) \\ \frac{1}{m_{ds}}\big(F_{ds} - F_c - b_{ds}v_{ds} - \Psi_{ds}(v_{ds})\big) \\ k_c\big(v_{ds} - v_c\big) \\ \frac{1}{m_c}\big(F_c - F_{fr} - b_c v_c - \Psi_c(v_c)\big) \\ k_{fr}\big(v_c - v_{disturbance}\big) \end{bmatrix} \tag{4.5}$$

$$v_{disturbance} = \Gamma(WOB_{applied}, \Omega) \tag{4.6}$$

For nonlinear controller design, the goal is to design the system such that model nonlinearities are explicitly handled by the controller. Therefore, no approximations to the presented model are needed. For linear controller design, the system of equations can be linearized, and represented in state space form, with A, B, C and D matrices given by:

$$A = \begin{bmatrix} 0 & -k_{ds} & 0 & 0 & 0 \\ \frac{1}{m_{ds}} & -\frac{1}{m_{ds}}\big(b_{ds} + \frac{\partial \Psi_{ds}}{\partial x_2}\big|_{x_{2,eq}}\big) & -\frac{1}{m_{ds}} & 0 & 0 \\ 0 & k_c & 0 & -k_c & 0 \\ 0 & 0 & \frac{1}{m_c} & -\frac{1}{m_c}\big(b_c + \frac{\partial \Psi_c}{\partial x_4}\big|_{x_{4,eq}}\big) & -\frac{1}{m_c} \\ 0 & 0 & 0 & k_{fr} & 0 \end{bmatrix} \tag{4.7}$$

$$B = \begin{bmatrix} k_{ds} & 0 & 0 & 0 & 0 \end{bmatrix}^T, C = \begin{bmatrix} 0 & 0 & 0 & 0 & 1 \end{bmatrix}, D = 0 \tag{4.8}$$

Since the desired control variable is F_{fr}, the C matrix is changed, such that F_{fr} is treated as the virtual output, even though the actual output of the system is the hook-load measurement F_{ds}.

4.3 Linear Control Design

In the following section, several suitable control algorithms for WOB tracking using the model described in the previous section are introduced. The simplest linear control technique is the proportional-integral-derivative (PID) control, which is also the most widely used controller for WOB tracking in the drilling industry. The PID approach is based on classical control theory, where a controller is designed for the transfer function of the system in the frequency domain. Therefore the feedback loop is closed on the output variable only. More sophisticated modern control techniques, including the LQI (linear quadratic integral) and the MPC (model-predictive control) theoretically enable better performance, since they stabilize all of the system states. In addition, the aforementioned techniques are designed by solving an optimization problem that provides optimal control gains based on a user-specified cost function. The MPC approach also enables the explicit handling of actuation constraints, which is a problem of interest in this dissertation. The performance of these controllers will be compared in the following sections, and will be used for benchmarking with the nonlinear controller.

4.3.0.1 PID Control

PID controllers are the most commonly used commercial controllers in the drilling industry, and most of the existing auto-driller systems rely on them for WOB tracking. The transfer function of a PID controller is given by:

$$K_p + \frac{K_i}{s} + K_d s \tag{4.9}$$

where K_p is the proportional gain, K_i is the integral gain, and K_d is the derivative gain. To design a controller, we take note of the transfer function of the axial system, given by:

$$G(s) = \frac{K}{s^5 + a_1 s^4 + a_2 s^3 + a_3 s^2 + a_4 s + a_5} \tag{4.10}$$

and add an integrator to achieve zero steady-state tracking error. Since the integrator adds an unstable zero to the system, a proportional term, K_p, is then used to add a zero and stabilize the system. The gains K_p and K_i are then chosen to achieve the desired time-domain performance, and provide sufficient gain and phase margins for robustness. Several criteria exist for appropriate tuning of PID control gains, including the root-locus approach. We used a commercial software PID tuner by MATLAB to perform the tuning of the PID controller in the rest of this chapter.

4.3.1 Linear-Quadratic-Integral (LQI) Control

LQI control is based on the linear quadratic regulator (LQR), with integral action to improve controller robustness when the actual model of the

environment differs from the model used for controller design (Zhou et al., 1996). The LQR control problem is concerned with solving the infinite horizon, continuous-time optimal control problem given by:

$$\dot{x} = Ax + Bu \tag{4.11}$$

with the cost function defined as:

$$J = \int_0^\infty (x^T Q x + u^T R u + 2x^T N u) dt \tag{4.12}$$

where Q and R are the weight matrices. The control law given by:

$$u = -\mathrm{K}x \tag{4.13}$$

minimizes the value of the cost function in Equation 4.12, where K is given by:

$$\mathrm{K} = R^{-1}(B^T P + N^T) \tag{4.14}$$

and P is obtained by solving the algebraic Riccati equation:

$$A^T P + PA - (PB + N)R^{-1}(B^T P + N^T) + Q = 0 \tag{4.15}$$

To incorporate integral action, the original state space can be augmented such that:

$$\frac{d}{dt}\begin{bmatrix} x \\ z \end{bmatrix} = \begin{bmatrix} Ax + Bu \\ Cx - r \end{bmatrix} \tag{4.16}$$

where the new state z is the integral of the error between the desired output r and the actual output y. The LQR problem is then solved for the new augmented system in order to find the controller gain K.

An LQR controller for a single-input single-output plant is supposed to possess a guaranteed gain margin of -6 to $+\infty$, and phase margin of $-60°$ to $+60°$ in all channels (Zhou et al., 1996). The robustness of the controller is an important criteria for the WOB tracking controller, especially regarding uncertainty in model parameters, because of the learning process that was outlined in Chapter 2. The frequency response of the closed-loop system with a sample LQI controller, subject to plant perturbations is shown in Figures 4.4 through 4.6.

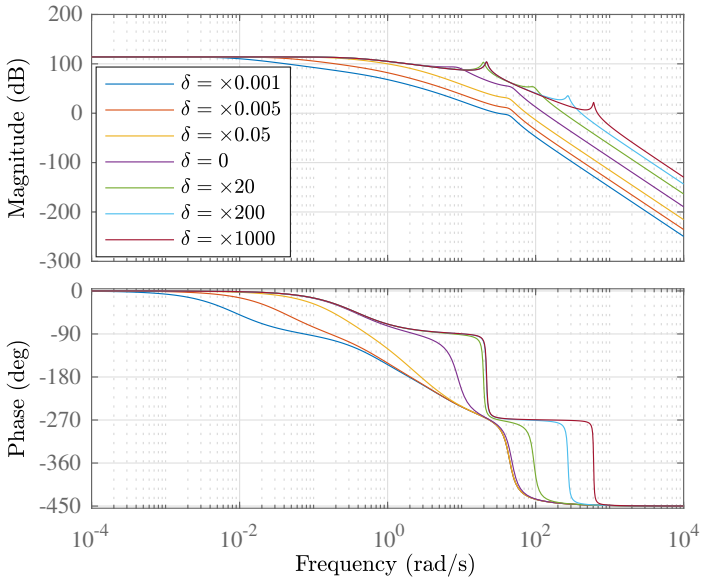

Figure 4.4: Bode diagram of the closed-loop system response with various levels of perturbations in the formation stiffness parameter, k_{fr}.

The Bode plot in Figure 4.4 demonstrates the sensitivity of the closed-loop system to perturbations in the formation stiffness k_{fr}. While it is clear

that the time-domain performance of the LQI controller deteriorates when $\delta > \times 20$ as shown by the resonant peaks of the magnitude plot, the closed-loop response becomes unstable at $\delta < \times 0.005$, as demonstrated by the shifted cross-over frequency. Perturbations in the mass parameter of the drill-collars and the BHA are analyzed in Figure 4.5. With the variation in phase response in the high frequency range (100Hz), changes in the m_c parameter are not expected to affect the time-domain response of the system significantly. Stability of the system due to these perturbations is not affected.

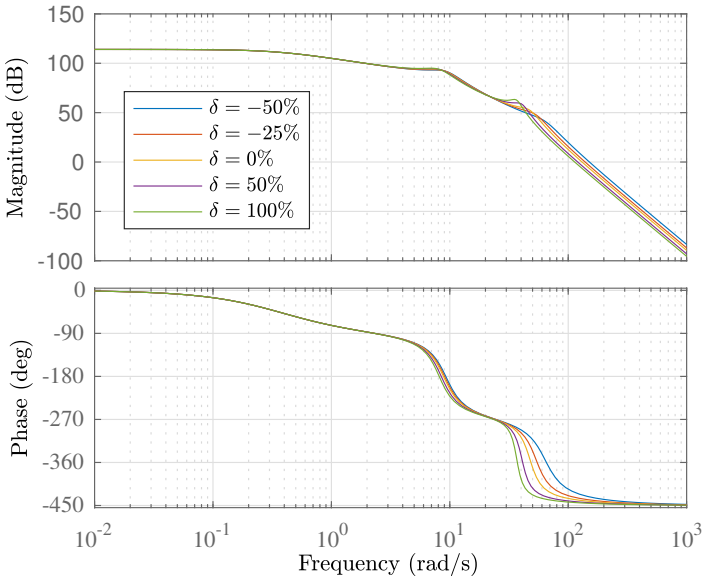

Figure 4.5: Bode diagram of the closed-loop system response with various levels of perturbations in the drill-collar and BHA mass parameter, m_c.

The maximum expected perturbations in the damping parameter associated with the drill-collars and the BHA, b_c, are shown in Figure 4.6. These

changes do not impact the stability of the system, similar to the case of the mass parameter. However, the time-domain response of the closed-loop system is expected to become more oscillatory with negative variations, as demonstrated by the gain plots.

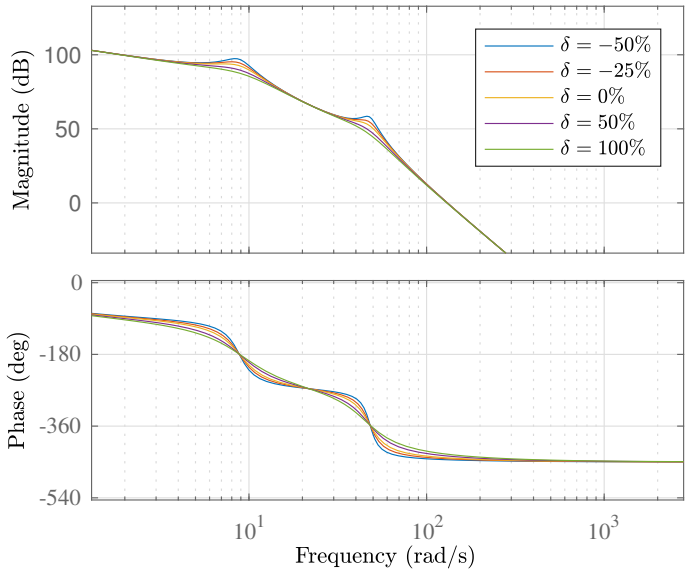

Figure 4.6: Bode diagram of the closed-loop system response with various levels of perturbations in the drill-collar and BHA damping parameter, b_c.

4.3.2 Model Predictive Control

The general objective of a discrete model predictive controller (MPC) is to compute a trajectory of a future manipulated variable u to optimize the future behavior of the plant output y. The difference in the optimization approach in MPC to that of the LQR is that the optimization is performed within

a limited time window, by implementing the information at the beginning of each window (Wang, 2009). Therefore, the optimal input to the plant is re-computed for each window, and revised when feedback arrives. To solve the optimization problem, the MPC uses a model of the environment to compute its future behavior. To improve controller robustness when the model does not match the exact behavior of the plant, the model is augmented with an integrator, similar to the formulation in 4.24 but for a discrete state space formulation.

The prediction of system behavior within an optimization window can be performed using the state space model such that:

$$Y = Fx(k_i) + \phi\Delta U \tag{4.17}$$

where

$$F = \begin{bmatrix} CA \\ CA^2 \\ CA^3 \\ \vdots \\ CA^{N_p} \end{bmatrix}; \phi = \begin{bmatrix} CB & 0 & 0 & \dots & 0 \\ CAB & CB & 0 & \dots & 0 \\ CA^2B & CAB & CB & \dots & 0 \\ \vdots & & & & \\ CA^{N_{p-1}}B & CA^{N_p-2}B & CA^{N_p-3}B & \dots & CA^{N_p-N_c}B \end{bmatrix} \tag{4.18}$$

where N_p is the prediction horizon, and N_c is the control horizon. For unconstrained optimization, the cost function J can be defined as:

$$J = (R_s - Y)^T(R_s - Y) + \Delta U^T \bar{R}\Delta U \tag{4.19}$$

where $R_s^T = [11...1]r(k_i)$ is the data vector that contains the set-point information, and the optimal ΔU is found by setting the first derivative of J with

respect to ΔU to zero, which yields:

$$\Delta U = (\phi^T \phi + \bar{R})^{-1} \phi^T (R_s - Fx(k_i)) \tag{4.20}$$

An important capability of the MPC approach is the explicit handling of input and output constrains (Wang, 2009). For the WOB control problem, handling of actuation constraints can improve controller robustness, as the output of the drawworks system is prone to saturation and has limited bandwidth. The actuation limits can be added as linear inequality constraints in the optimization problem, such that:

$$\Delta U_{min} \leq \Delta U \leq \Delta U_{max} \tag{4.21}$$

and in matrix form:

$$\begin{bmatrix} -I \\ I \end{bmatrix} \Delta U \leq \begin{bmatrix} -\Delta U_{min} \\ \Delta U_{max} \end{bmatrix} \tag{4.22}$$

In a similar manner for the saturation limits:

$$U_{min} \leq U \leq U_{max} \tag{4.23}$$

and in matrix form:

$$\begin{bmatrix} -I \\ I \end{bmatrix} U \leq \begin{bmatrix} -U_{min} \\ U_{max} \end{bmatrix} \tag{4.24}$$

Aggregation of all the constraints, into one compact expression results in the following hard constraint expression for the optimization problem:

$$M\Delta U \leq \gamma \tag{4.25}$$

where M reflects the constraints, and γ the constraint values.

Solving the optimization problem with inequality constraints is of course a more involved process, and requires the use of numerical solutions. The quadratic programming (QP) formulation provides a solution to the optimization problem. In the QP notation, the problem of interest to be minimized can be represented using the cost function:

$$J = \frac{1}{2}x^T Ex + x^T F \tag{4.26}$$

$$Mx \leq \gamma, \tag{4.27}$$

where E, F, M and γ are compatible matrices and vectors in the QP problem. The simplest form of QP is to find the constrained minimum of a positive definite quadratic function with equality constraints that each represent a hyperplane. With inequality constrains, however, the number of constrains could be more than the number of decision variables, and may contain active and inactive constraints (Wang, 2009). To handle this issue, a common approach is to use Hildreth's QP procedure for solving the dual problem that arises from the Lagrangian formulation.

4.4 Nonlinear Control Design

To design a controller for the nonlinear system represented by:

$$\dot{\mathbf{x}} = f(\mathbf{x}) + g(\mathbf{x})u \tag{4.28}$$

We put the system in normal form such that:

$$y = h(\mathbf{x}) = F_{fr} = L_f^0(h) \tag{4.29}$$

$$\frac{dy}{dt} = L_f(h) \tag{4.30}$$

$$\frac{d^2y}{dt^2} = L_f^2(h) \tag{4.31}$$

$$\frac{d^3y}{dt^3} = L_f^3(h) \tag{4.32}$$

$$\frac{d^4y}{dt^4} = L_f^4(h) \tag{4.33}$$

$$\frac{d^5y}{dt^5} = L_f^5(h) + L_g(L_f^4(h))u \tag{4.34}$$

$$\tag{4.35}$$

where $L_f()$ is the Lie derivative. Since the control term appears in the fifth order output, no zero-dynamics remain to be stabilized (Isidori, 2013). Given no unstable zero-dynamics exist to be stabilized, the nonlinear terms can be feedback-linearized if the system is locally controllable and observable. A controllability matrix can be constructed by using the Lie bracket, which is defined as:

$$[f, g] = \frac{\partial g}{\partial x} f - \frac{\partial f}{\partial x} g \tag{4.36}$$

$$(ad_f^0, g) \equiv g \tag{4.37}$$

$$(ad_f^1, g) \equiv [f, g] \tag{4.38}$$

$$(ad_f^k, g) \equiv [f, ad_f^{k-1}, g)] \tag{4.39}$$

to yield the controllability matrix of Equation 4.40.

$$C = \left[g, ad_f^0, g, ad_f^1, g...ad_f^5, g\right] \tag{4.40}$$

The system is then locally controllable if matrix C is full rank. Similarly, an observability matrix, O as in Equation 4.41 is constructed to ensure observability of all system states from surface measurements.

$$O = \begin{bmatrix} \frac{\partial L_f^0(h)}{\partial x_1} & \cdots & \frac{\partial L_f^0(h)}{\partial x_n} \\ \cdots & \cdots & \cdots \\ \frac{\partial L_f^{n-1}(h)}{\partial x_1} & \cdots & \frac{\partial L_f^{n-1}(h)}{\partial x_n} \end{bmatrix} \tag{4.41}$$

4.4.1 Pre-filter Design

The role of the pre-filter is to receive a static WOB set-point, and output a carefully designed ramped set-point to ensure a smooth transition from each set-point so that controller bandwidth restrictions are satisfied. The reference is a generated as a function of the set-point WOB_{des}, and the desired ramping time, t_{ramp}. (the cubic spline approach was discussed in Chapter 3). The following system of the linear equations can be solved to determine the cubic spline trajectory:

$$\Phi_{wob} = A_{wob}^{-1} b_{wob} \tag{4.42}$$

where,

$$\Phi_{wob} = \begin{bmatrix} \phi_1 \\ \phi_2 \\ \phi_3 \\ \phi_4 \end{bmatrix} \tag{4.43}$$

$$A_{wob} = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ t_{ramp}^3 & t_{ramp}^2 & t_{ramp} & 1 \\ 3t_{ramp}^2 & 2t_{ramp} & 1 & 0 \end{bmatrix}, b_{wob} = \begin{bmatrix} 0 \\ 0 \\ \text{WOB}_{des} \\ 0 \end{bmatrix}, \quad (4.44)$$

Figure 4.7 shows a comparison of a simple set-point ramping trajectory, and the cubic spline trajectory discussed above. As evident, the rate of change of the WOB set-point in the cubic case is a non-constant smooth parabola, which results in the second derivative of the set-point trajectory to not diverge to very high values (unlike the linear case).

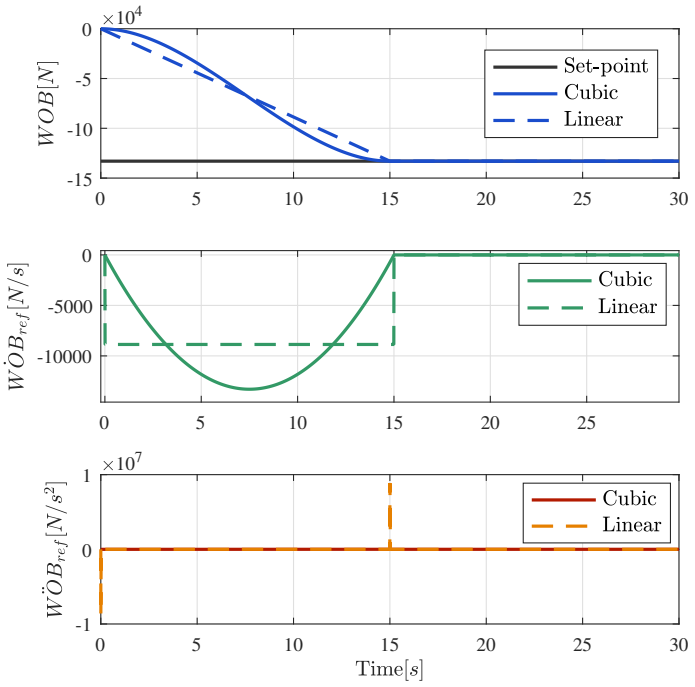

Figure 4.7: Smooth, time-dependent WOB set-point generation using a cubic spline function.

4.4.2 Sliding Mode Control

In dealing with the uncertain terms and the nonlinearity of system equations, we adopt the sliding mode approach for controller design. The main benefits of using this control technique are the direct incorporation of model nonlinearity and uncertainty into controller design (Slotine et al., 1991). In addition, all of the dynamics and parameters of the system get directly incorporated inside the controller, thus minimizing the need for controller tuning when the plant characteristics change. Developing a sliding mode controller is achieved through designing a sliding surface, $S(x_1...x_n)$, so that the relative degree of the output S is one, and the controller input u appears explicitly in the $\dot{S}$ expression. The goal of the design process is to make $S = 0$ an attractive surface that guarantees system stability through attracting, and forcing the system to slide along the surface to zero. This can be mathematically expressed as:

$$S\dot{S} \leq \eta_1|S| \tag{4.45}$$

After studying the system equations, we realize that to have matching conditions for controller input and process uncertainty, the controller has to be divided into two nested controllers, one that provides tracking for downhole WOB=$F_{fr} = x_5$, and another for $F_c = x_3$. For x_3, the control system input $u_1 = v_{in}$, and an outer-loop sliding mode controller can be designed such that:

$$S_1 = \lambda_1\ddot{e_1} + \lambda_2\dot{e_1} + \lambda_3 e_1 + \lambda_4 \int e_1 \tag{4.46}$$

where

$$
\begin{aligned}
e_1 &= r_1 - x_3 \\
\dot{e_1} &= \dot{r}_1 - k_c(x_2 - x_4) \\
\ddot{e_1} = \ddot{r}_1 &- \Big(\Big(k_c\big(x_5 - x_3 + b_c x_4 + aq_2 \tanh(bq_2 x_4)\big) \Big) / m_c \\
&- \Big(k_c\big(x_3 - x_1 + b_{ds} x_2 + aq_1 \tanh(bq_1 x_2)\big) \Big) / m_{ds} \Big)
\end{aligned}
$$

and $\lambda_1 \ldots \lambda_4$ are constants that determine the roots of the characteristic equation. For the drillstring and drill-collars, the $aq\tanh(bqx)$ terms serve as a smooth approximation to the nonlinear friction terms, Ψ. Depending on the position of the drill-pipes and and drill-collars in the wellbore, aq_1, aq_2, or both can be zero. The goal of the sliding mode controller is to ensure:

$$\dot{S}_1 = -\eta_1 \frac{|S_1|}{S_1} \tag{4.47}$$

The dynamics of the sliding surface can be described in terms of the error states such that:

$$-\eta_1 \frac{|S_1|}{S_1} = \lambda_1 \dddot{e_1} + \lambda_2 \ddot{e_1} + \lambda_3 \dot{e_1} + \lambda_4 e_1 \tag{4.48}$$

Since the control input u_1 shows up in $\dddot{e_1}$, the expression above can be rewritten as:

$$-\eta_1 \frac{|S_1|}{S_1} = \lambda_1(\alpha_1 + \beta_1 u_1) + \lambda_2 \ddot{e_1} + \lambda_3 \dot{e_1} + \lambda_4 e_1 \tag{4.49}$$

A second controller can be designed, with the control variable $u_2 = x_3$, such that:

$$S_2 = \lambda_5 \dot{e_2} + \lambda_6 e_2 + \lambda_7 \int e_2 \tag{4.50}$$

where

$$e_2 = r_{wob} - x_5$$

$$\dot{e}_2 = \dot{r}_{wob} - k_{fr} x_4$$

and $\lambda_5 \ldots \lambda_7$ are constants that determine the roots of the characteristic equation. The goal of the sliding mode controller is to ensure:

$$\dot{S}_2 = -\eta_2 \frac{|S_2|}{S_2} \tag{4.51}$$

$$-\eta_2 \frac{|S_2|}{S_2} = \lambda_5 \ddot{e}_2 + \lambda_6 \dot{e}_2 + \lambda_7 e_2 \tag{4.52}$$

$$-\eta_2 \frac{|S_2|}{S_2} = \lambda_5 (\alpha_2 + \beta_2 u_2) + \lambda_6 \dot{e}_2 + \lambda_7 e_2 \tag{4.53}$$

where α_2 includes all the nonlinear terms and the process uncertainty in $\ddot{e}_2$. Therefore, as long as the maximum uncertainty in the model is bounded and less than η, attraction and sliding of the system dynamics along S is guaranteed (Slotine et al., 1991).

With conventional sliding mode control, the $\frac{|S|}{S}$ term can cause high chatter in the controller input. In order to avoid the high chattering in cases where actuator bandwidth is low, the $\frac{|S|}{S}$ term can be replaced by $tanh(cS)$ with c equal to some large positive constant. Another solution to the chatter avoidance problem is the super twisting sliding mode algorithm, STSMC, where similar stability guarantees are provided, but the controller input is

modulated by an integral term which grows larger with higher switching frequencies (Shtessel et al., 2010). The STSMC algorithm is given by:

$$u = -c_1\sqrt{|S|}sign(S) + v \tag{4.54a}$$

$$v = -\int c_2 sign(S) \tag{4.54b}$$

where c_1 relates to the η parameter in the original SMC algorithm, and c_2 is related to the chatter reducing capability of the STSMC algorithm.

4.4.2.1 Simulation Results

To assess the applicability of the sliding mode approach, and its benefits in minimizing the impacts of model uncertainty and dry friction, the discussed sliding mode controller was used in a simulation study where its performance was compared to a simple PID controller. The PID controller was designed to produce a 60° phase margin for robustness, as shown in the Bode diagram of Figure 4.8.

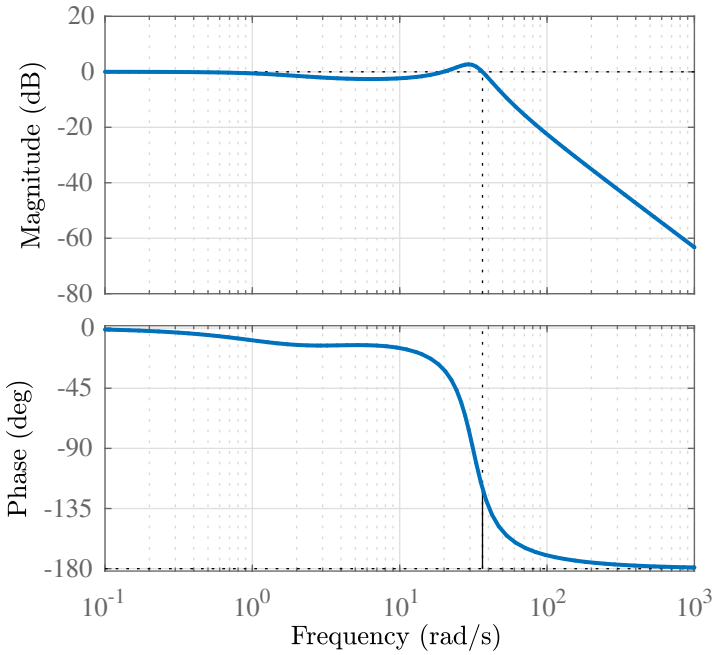

Figure 4.8: Bode diagram of the closed-loop PID system, designed to have infinite gain margin, and 60° phase margin (at 36.5 rad/s)

The effect of various of super-twisting algorithms is investigated in Figure 4.9, where a phase-plane analysis of the sliding behavior is shown. The original sliding mode approach, shown in blue, provides the fastest convergence to the sliding surface, followed by significant chatter when approaching $x_1 = 0$. The other three lines represent the super-twisting approach, where a smaller ratio of c_2 to c_1 corresponds to higher twisting. With $c_2 = 0.5c_1$ and $c_2 = 0.3c_1$, a smooth phase-plane behavior is observed, where a slower reaching time is observed, but minimal chatter is present in comparison to the original

SMC. In the case of $c_2 = 0.1c_1$, a twisting effect that is too high results in slow reaching time and high over-shoot, as obvious by the purple line crossing zero velocity and high chatter around $x_1 = 0$.

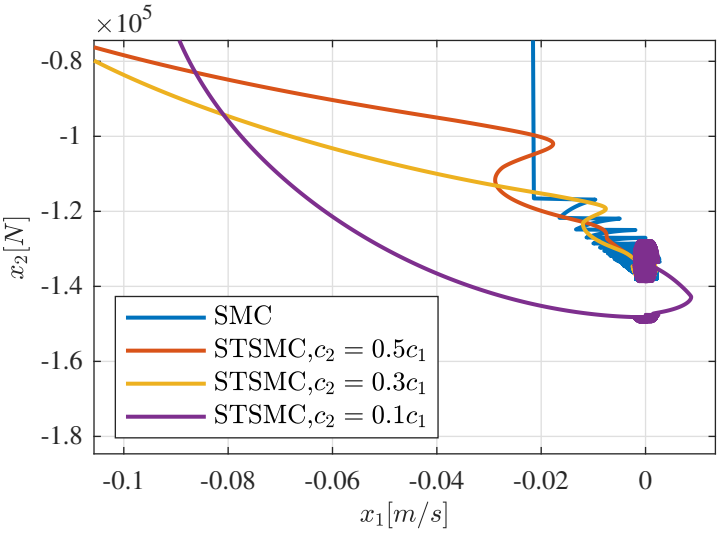

Figure 4.9: Phase plane analysis of different SMC techniques, including the super-twisting algorithms with varying c_2/c_1 ratios.

For a simple performance comparison to a PID controller, Figure 4.10 shows the step response of both controllers. The sliding mode controller avoids the initial overshoot of the PID, and allows for a steady convergence to the desired set-point. By doubling the nonlinear component of friction, the performance difference between the two controllers is more obvious, as shown in Figure 4.11. The PID controller takes much longer to reach the WOB set-point, and continues to overshoot for a long period before converging at $t = 10$.

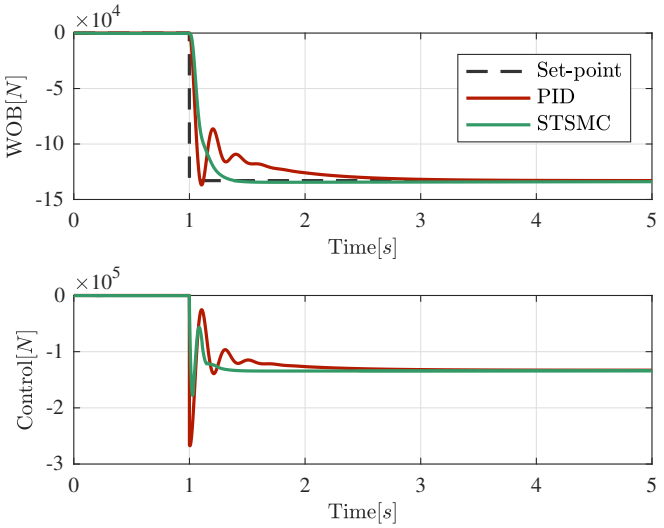

Figure 4.10: Step response of the PID and STSMC controllers, when the coefficient of dry friction is 0.2

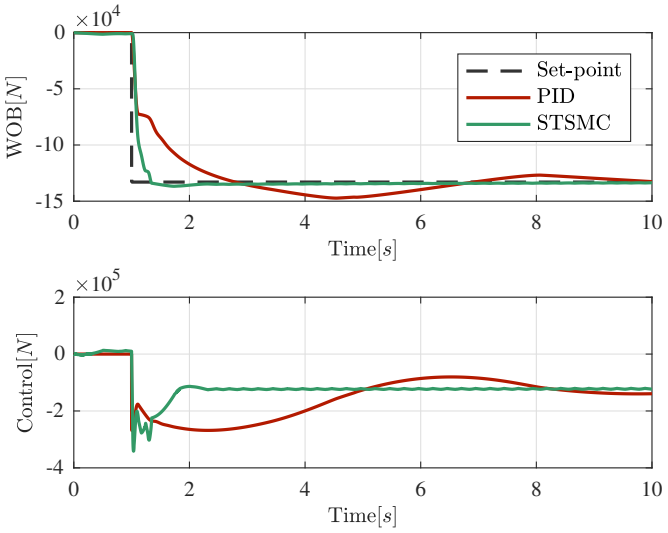

Figure 4.11: Step response of the PID and STSMC controllers, when the coefficient of dry friction is 0.4

The results in Figure 4.11 support our initial intuition about the performance of the sliding mode controller: that its performance margin becomes more obvious when the plant behaves more nonlinearly. A robustness study is shown in Figure 4.12, where the plant model is inaccurate by 200%, a randomly distributed disturbance of variance $0.002m/s$ exists, and formation stiffness changes by$\times 5$ from $15s$ to $17s$. While the transient behavior of the sliding mode controller is much better due to higher robustness to parameter uncertainty, the steady state performance of both controllers is quite similar, with both controllers reacting similarly to the formation stiffness change.

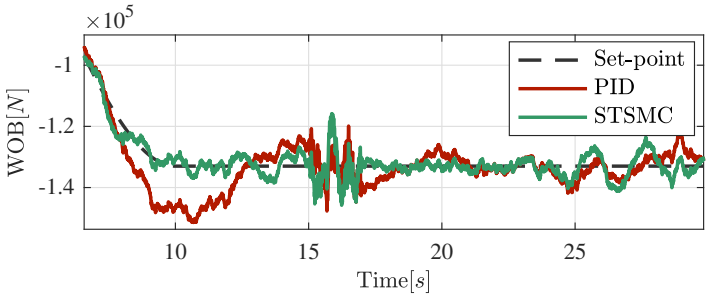

Figure 4.12: Tracking performance comparison of the PID and STSMC controllers, where there is a 200% perturbation in the drill-collar mass parameter, and the formation stiffness changes from $15 < t < 17$. The controller is subject to a $0.002m/s$ randomly distributed disturbance.

4.4.2.2 Handling Process Delays

The infinite-gain concept of sliding mode control leads to the question of how such a technique would handle a delay that arises from either delayed actuation, delayed measurement, or both. This is an important consideration for a field-level WOB controller, as measurement is often delayed due to the

time it takes for downhole dynamics to be reflected in surface measurements. While certain control techniques are better suited to handling delays, others can significantly suffer and force the system into instability. In Figure 4.13, the response of both sliding mode and PID controllers to a $500ms$ output delay is shown. While the PID controller's performance has clearly deteriorated and exhibits a limit-cycle behavior, the sliding mode controller has become unstable.

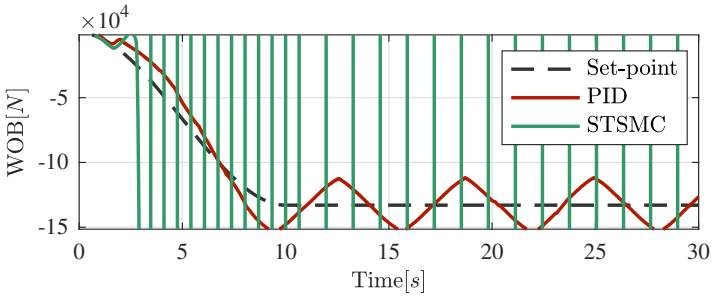

Figure 4.13: Tracking response comparison of the controllers when exposed to a $500ms$ input delay.

Sliding mode control can be made applicable in the presence of delays by including a predictive element into the controller. This predictive element would then cancel out the delayed response of the plant, and prevent the system from going unstable. This is the idea behind a Smith Predictor, as shown in Figure 4.14.

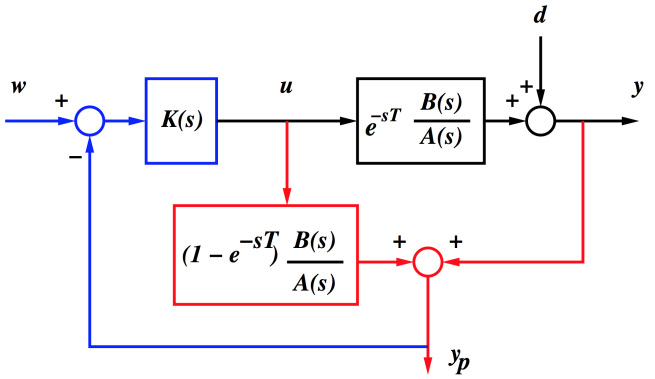

Figure 4.14: Block diagram of a Smith predictor to handle known system delays. $K(s)$ corresponds to the controller, and e^{-sT} represents the model of the delay, where T is the known system delay.

The output of the controller in Figure 4.14 is sent to both the actual and the model of the plant, where e^{-sT} represents the process delay. Through this prediction, the delay component of the output y is subtracted, and the feedback now only consists of the impact of disturbance d, and the delay-free response of the plant. The challenging aspect of implementing a Smith predictor is that the model of the plant as well as the estimate of the process delay have to be sufficiently accurate. In Figure 4.15, the performance of the aforementioned controllers, both complemented with a Smith predictor is shown with a $500ms$ process delay. When the model of the plant is fairly representative of the actual plant, the original performance benefits of the sliding mode controller are recovered. However, when the model of the plant is inaccurate, Figure 4.15 demonstrates that the sliding mode controller only performs marginally better in the transient state, followed by a poor settling

time in comparison to the PID controller. The robustness of the sliding mode approach therefore is only a major advantage when the plant does not contain significant process delays that have to be mitigated using a Smith predictor approach.

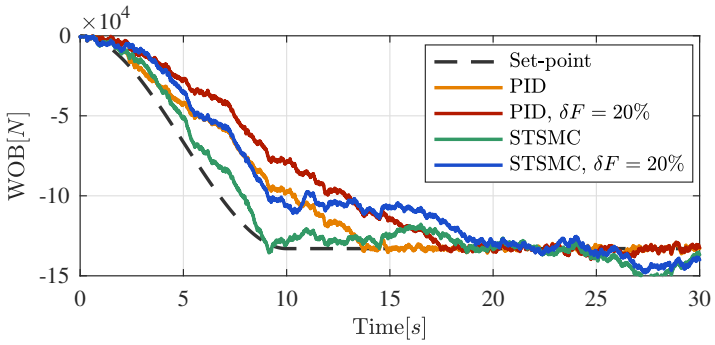

Figure 4.15: Tracking response comparison of the controllers with a Smith predictor, when exposed to a $500ms$ input delay, with various levels of model uncertainty.

4.5 Performance Assessment on High Fidelity Plant Model

In the last part of this chapter, the control approaches that we explored earlier are implemented on a realistic model of the plant that was developed in Chapter 2. The use of this model exposes the WOB controller to latencies present in the actual plant, and highlights the challenges involved in state estimation and control of distributed drill-string dynamics.

We will briefly discuss some practical modifications to the nonlinear controller presented in Section 4.4, as well as state estimation methodology in Chapter 2, to handle the challenges of the infinite-dimensional model. The

performance of the nonlinear control in various practical scenarios is explored, and benchmarked against the linear control techniques discussed earlier in this chapter.

4.5.1 Observer Considerations

The problem of state estimation for state-feedback control was already addressed in Chapter 2. The role of the state estimator, or the "observer", is to provide an estimate of the un-measureable states of the plant, given the input and output of the plant. As noted in Chapter 2, the infinite-dimensional model can be estimated using any order of finite approximations, as well as an infinite-dimensional estimator, called the DDE observer.

In the previous section, we also discussed the challenges associated with controlling a system that has a combination of input/output and process delays. In drilling, such delays are virtually present at all times, due to communication complications as well as the distributed nature of the physical system. Therefore, any control algorithm in the drilling environment has to be amended with a predictive element to handle delays. The idea of a Smith predictor was to essentially erase the delayed-open loop response of the system, and add a delay-free response based on a model of the system. The challenge with employing such an idea is that if there are any discrepancies between the model and the plant, robustness of the controlled is jeopardized. For a practical implementation of Smith prediction, a modified high-gain UKF observer will be employed in the following section.

The high-gain UKF observer utilizes a model of the same order as the controller itself. In order to remove the delayed open-loop response of the plant, the HG-UKF employs an arbitrary measurement noise, R_{delay}, in addition to the actual measurement noise $R = \sigma_{hlk}^2$ to increase the observer gain. In this manner, the correction made to the estimate of the reduced-order observer is limited to include only the effects of disturbances on the plant, and the delayed-response of the plant is filtered.

4.5.2 Controller Modifications

For implementation on the high-fidelity model, several modifications were made to the sliding mode controller proposed in the previous section. First, the cascade formulation is reduced to a surface controller only, to mitigate the complications that arise from bandwidth matching of the two controllers. To handle the matching conditions problem, the sliding surface of the surface controller is modified, such that includes $e_f + \int e_f$, where e_f is the downhole tracking error represented by $WOB_{ref} - F_{fr}$.

Additionally, due to the uncertainty that arises in estimating the velocity terms of the plant model as well as velocity dependent force terms including dry friction, the u_{fbl} term of the surface controller now excludes these terms, and instead u_{smc} is modulated in real-time to cancel the dynamics associated with these terms.

4.5.3 WOB Tracking while Tagging Bottom

The first benchmarking experiment implements all 4 controllers on a tagging-bottom scenario, where force control is enabled, and the WOB reference is increased initially to $100kN$, followed by an increase to $200kN$. The plots in Figure 4.16 demonstrate the response and control effort of each controller. In the first transient period, with $t < 10$, the STSMC controller has an obvious superior performance. From a control effort perspective, the LQI controller uses the least energy, but at the cost of slow settling time in comparison to all other controllers. The best steady state performance is also achieved by the STSMC controller, as evident in the magnified portion of the experiment from $33 < t < 50$. Due to a $0.5s$ discretization, the MPC controller inevitably exhibits oscillatory behavior in the steady state, which is increased further in the transient.

A summary of the performance of all controllers in this experiment is shown in Table 4.1. For this task, the STSMC controller has a significantly smaller mean squared error, followed by the LQI controller.

Controller	**MSE** $[N^2] \times 10^7$
STSMC	0.67
MPC	29.56
LQI	9.23
PID	21.71

Table 4.1: Mean squared tracking error for all controllers in ideal conditions.

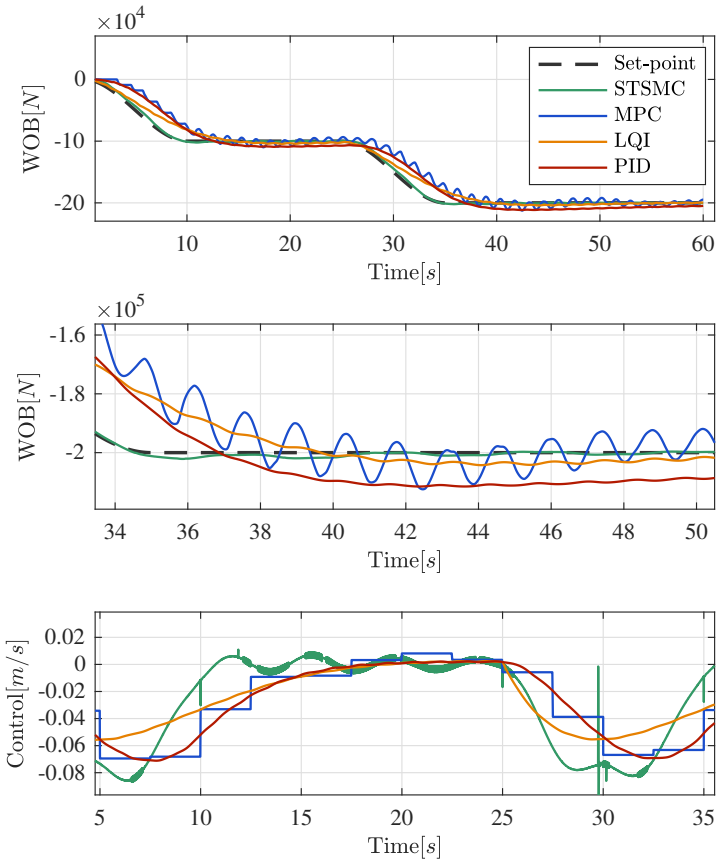

Figure 4.16: WOB tracking response of the controllers with no disturbances. The middle plot is a magnification of the upper plot from 34 to 50 seconds. The lower plot shows the control inputs from 5 to 35 seconds, during the first transient period.

4.5.4 Robustness to Parametric Uncertainty

An analysis of controller performance when there is major discrepancy between the model used for controller and observer initialization and the actual

plant, is essential for practical field level implementation, as well as to account for the uncertainty that resulted from the learning of the model parameters in Chapter 2. Figure 4.17 demonstrates such a scenario, beginning at $t = 30$, where a 100% change in the parameters of the drill-collars and the BHA is experienced by the control system. The highest robustness is demonstrated by the STSMC approach, as minimal functions are observed when the mentioned plant changes are enforced. The MPC controller is the least robust to plant changes, as demonstrated by the prolonged oscillations.

In Table 4.2, the mean squared tracking error of all controllers is shown.

Controller	**MSE** $[N^2] \times 10^7$
STSMC	0.65
MPC	30.54
LQI	8.98
PID	21.29

Table 4.2: Mean squared tracking error for all controllers when subject to parametric uncertainty.

4.5.5 Robustness to Actuation Constraints

The high variability in rig-site equipment prevents the plug-&-play application of control algorithms without considering equipment power limits. In the case of a WOB controller, the power limits of a drawworks system translates into both input acceleration and velocity limits. In Figure 4.18, we demonstrate a scenario when the controller input saturates at $0.04m/s$. As is evident, the performance of all controllers, except the MPC controller, drastically suffers when the controller input saturates. The MPC controller is able

to plan a better control trajectory and achieve the desired $200kN$ set-point despite the input constraint, since it handles the actuation constraint explicitly at each optimization window. The STSMC controller performs the worst in this scenario, and drives the system into instability.

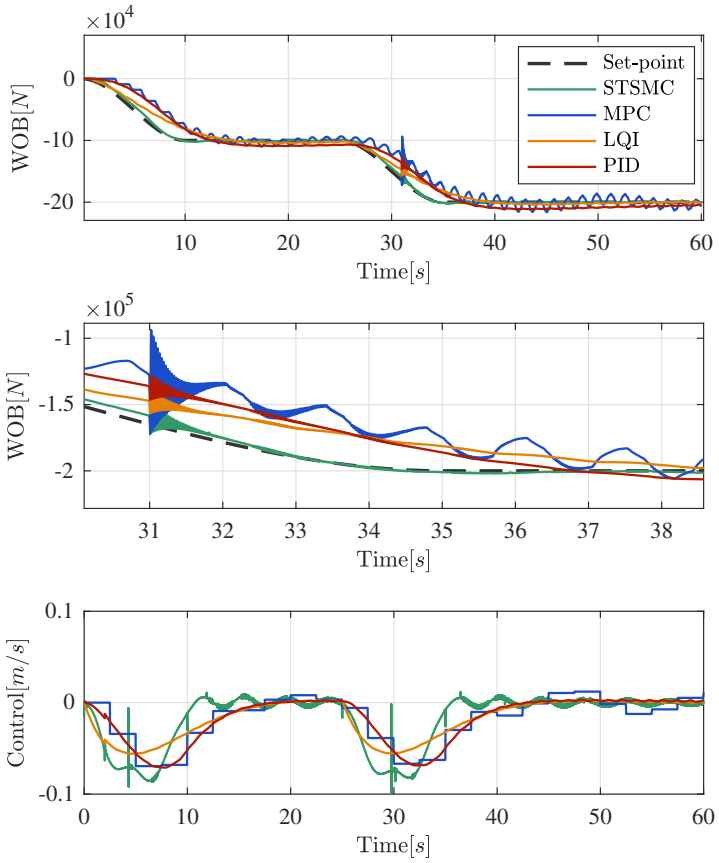

Figure 4.17: WOB tracking response of the controllers when exposed to plant perturbations that result in parametric uncertainty. The middle plot is a magnification of the upper plot from 31 to 38 seconds.

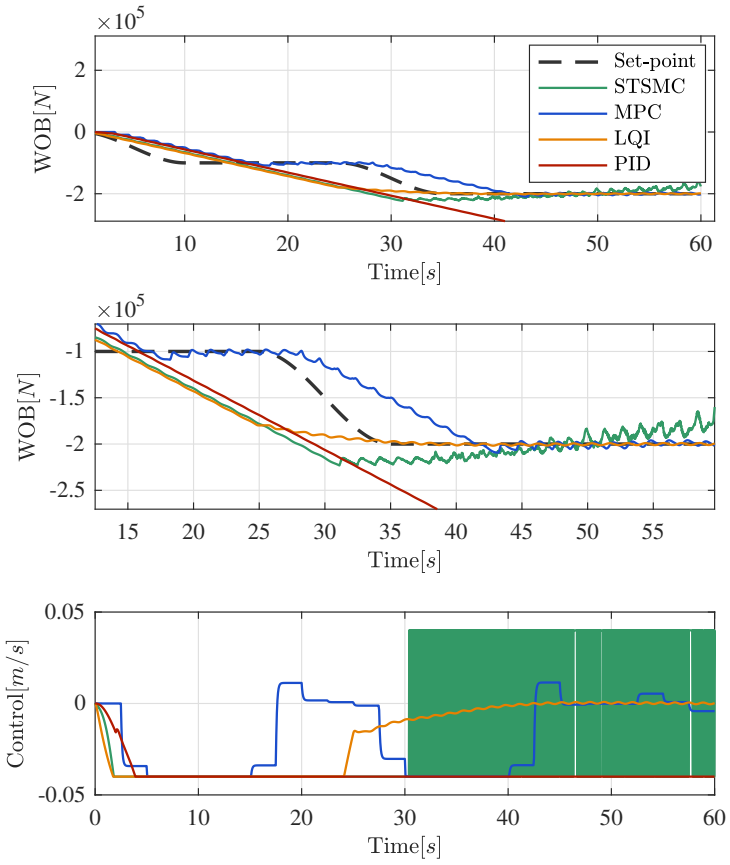

Figure 4.18: WOB tracking response of the controllers with constrained actuation power. The middle plot is a magnification of when the controllers try to achieve the final desired set-point around $t = 30s$.

4.5.6 Disturbance Rejection

A fundamental assessment of the WOB controller is concerned with the ability to handle disturbances that appear in the form of varying ROPs when the applied WOB and RPM change. In Figure 4.19, an increase in

ROP at $t = 41s$ causes a slip of the drill-bit, and WOB decreases to $190kN$. All four controllers begin to force the system to the original WOB set-point. Although the PID controller seems to converge faster to the original set-point, this is due to the fact that the controller was originally operating above the desired WOB set-point. Amongst the other controllers, the STSMC provides the fastest WOB recovery. At $t = 55s$, ROP decreases and the WOB is now increased to $220kN$. Similar to the previous case of PID, the MPC returns fastest to the WOB set-point, only because it was originally operating below the WOB set-point. The STSMC recovers from the disturbance in $9s$, followed by the LQI and then the PID. The tracking error results of the disturbance rejection scenario are shown in Table 4.3. The marginal performance difference between the LQI and the STSMC controllers are now reduced compared to the previous test.

Controller	**MSE** $[N^2] \times 10^7$
STSMC	4.04
MPC	25.48
LQI	12.46
PID	17.85

Table 4.3: Mean squared tracking error for all controllers when subject to slowly varying disturbance. The results were averaged from 10 experiments.

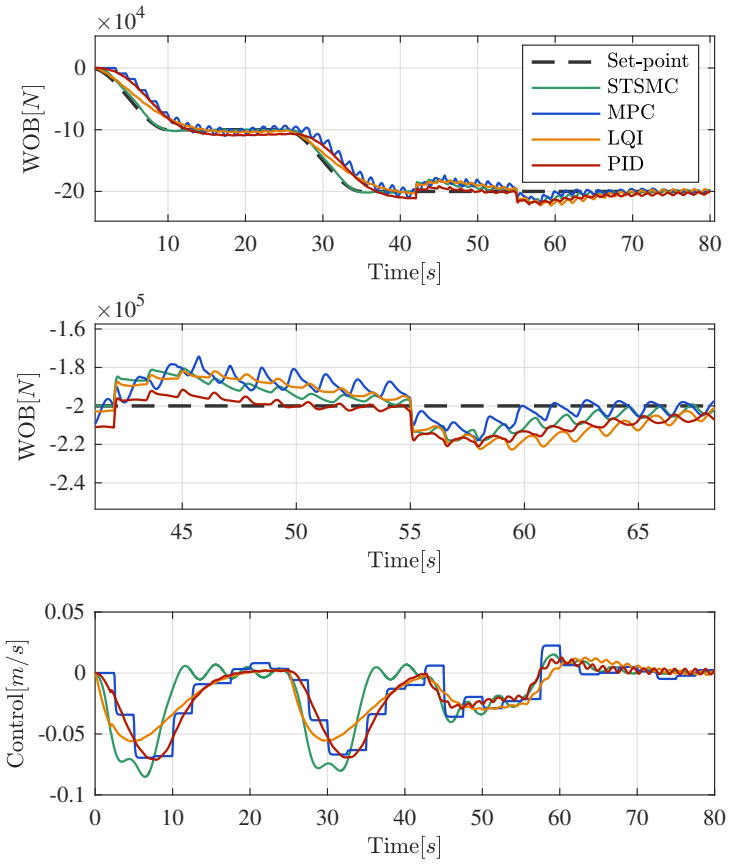

Figure 4.19: Disturbance rejection ability of the controllers with a slowly varying ROP. The middle plot is a magnification of the upper plot from 42 to 68 seconds. The lower plot shows the control inputs from 42 to 68 seconds.

A final test for the WOB controllers is their ability to quickly handle changing disturbances, and the effect of disturbance handling on variations in angular velocity. The enforced disturbances are shown in the upper right plot in Figure 4.20. As is evident from the estimation of the disturbances by the

observer, the oscillatory behavior of all controllers can be attributed to the noisy estimation of the disturbance, and not the controllers themselves.

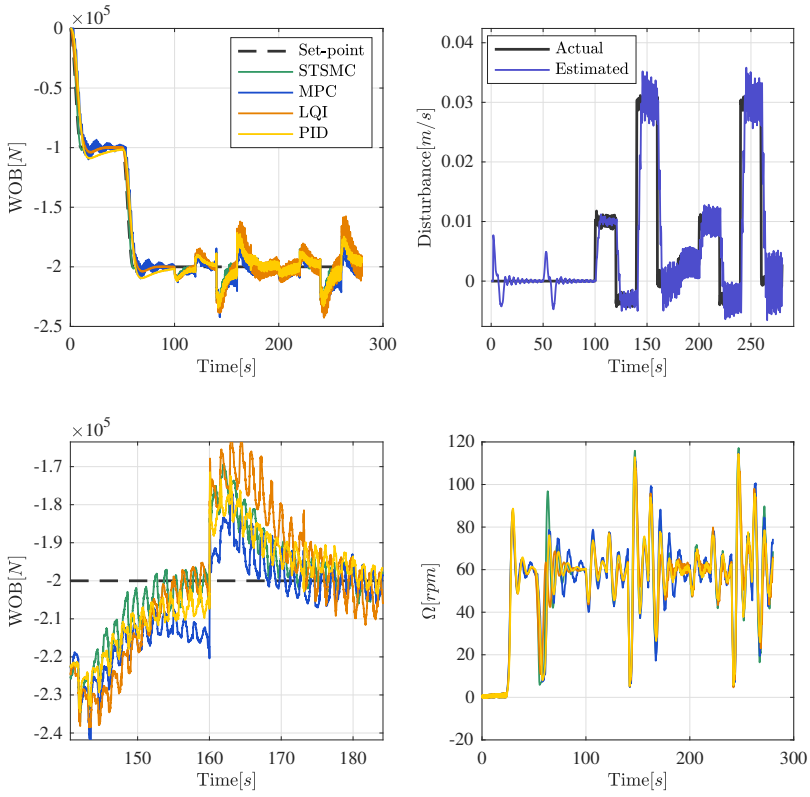

Figure 4.20: Disturbance rejection ability of the controllers with a fast varying ROP. The upper right plot demonstrates the actual and estimate of the disturbance. The lower right plot demonstrates the fluctuations in bit angular velocity, Ω, as a result of WOB variations.

In the context of recovery from quickly varying disturbances, the LQI controller has now become the worst performing controller (as is evident from results in Table 4.4) due to its slow transient performance. However, from

a torsional vibrations perspective, the LQI controller seems to provide the most damped response, especially in comparison to the MPC and STSMC controllers, as shown in the lower right plot of Figure 4.20.

Controller	**MSE** $[N^2] \times 10^7$
STSMC	6.73
MPC	13.35
LQI	14.81
PID	11.75

Table 4.4: Mean squared tracking error for all controllers when subject to quickly varying disturbance. The results were averaged from 10 experiments.

4.6 Conclusion and Recommendations

In this chapter, a robust control methodology was developed for control of axial drillstring dynamics with the purpose of tracking the WOB setpoints that are assigned by the qDrill algorithm. We hypothesized that due to the nonlinearities in drillstring dynamics, a robust nonlinear approach to the tracking control problem would yield the best performance results. Through development of several candidate controllers, it was shown that the proposed nonlinear control technique is indeed the best approach from several perspectives.

A summary of the findings from the analysis of all controllers is shown in Table 4.5. As already noted, the STSMC approach offers high levels of transient performance and robustness in comparison to the other linear control techniques. From a practical standpoint, however, the STSMC approach

is not a fully model-based approach, and its design and commissioning might require the involvement of a human expert. This arises from the fact that the choice of controller parameters such as c_1 and c_2 has to be verified in simulation prior to controller deployment. The inferior MPC approach despite its less robust performance, can be easier to commission on rig-site systems, and automatically re-tuned by the automation system itself. Future research work should therefore explore more sophisticated variations of the MPC approach, for comparison to STSMC in terms of robustness and tracking performance.

Controller	**Robustness**	**Tracking Capability**	**Tuning Difficulty**	**Computational Power Needs**
STSMC	High	High	Medium	Low
MPC	Low	Medium	Low	High
LQI	Medium	Medium	Medium	Medium
PID	Medium	Low	High	Low

Table 4.5: Comparison of all control techniques from a practical standpoint for field-level implementation.

Chapter 5

Event Detector: Missed/False Alarm Minimization in Drilling Event Detection

During an automated drilling operation, various critical trouble events can potentially require the attention of both the automation system and the human operator. In the architecture shown in Figure 1.2, the *Event Detector* was proposed for the detection of such events. In the case of an event where its handling falls outside of the automation system's scope, the role of the automation system is to simply bring its state to a halt, and give control to the human operator. The *Optimizer* was presented in Chapter 3, which through iterative learning addressed the drilling optimization problem. From an operational standpoint, however, the *Optimizer* can only become truly optimal if it is not excessively interrupted by the *Event Detector* due to false alarms. We explore this problem in this chapter[1], and present an event detection system that addresses the challenges of robust event detection. The *Event Detector*

[1]Most of the work in Chapter 5 has been presented in the conference publication: P. Pournazari, P. Ashok, E. van Oort, S. Unrau, S. Lai, et al. Enhanced kick detection with low-cost rig sensors through automated pattern recognition and real-time sensor calibration. In *SPE Middle East Intelligent Oil and Gas Conference and Exhibition*. Society of Petroleum Engineers, 2015b. The author of this dissertation contributed to that work through study of the literature for kick detection systems, design of the event detection system, and data analysis.

focuses on kick and lost circulation events as they are amongst the most difficult drilling events to detect, and account for high false alarm rates on both offshore and onshore drilling rigs.

5.1 Introduction

During a drilling operation, early detection of trouble events such as kicks (an undesirable influx from downhole formation(s)) and lost circulation (loss of drilling fluid to downhole formation(s)) is important in minimizing the associated safety risks and non-productive time. Early detection of these events depends mainly on the quality of mud circulation sensors and the robustness of the detection algorithm. The challenging aspect of this problem is not necessarily detecting the gain/loss of fluid, but accurately distinguishing undesirable gains/loss events from events that are similar in signature such as wellbore breathing. Specifically, while wellbore breathing events also result in mud gains during a drilling operation, the consequences are much less serious than a kick. In wellbore breathing, mud gains occur as a result of the formation fractures pushing back the drilling fluid that was keeping them open. The signature of a wellbore breathing event differs from that of a kick due to a decreasing pressure as the fractures close, which causes the pit volume trend to increase with an exponentially decaying shape. Therefore, the event detection system should not only be robust to unreliable sensor measurements, but it should also be able to distinguish less important events such as wellbore breathing from more serious ones such as kicks.

This chapter is organized as follows: after presenting a review of various techniques for kick and lost circulation detection, we present a solution based on the idea of condition pattern regoncitnion that forms the basis of the *Event Detector*. The various subsystems are explained in detailed, including the change detection technique, and the pattern recognition approach. The fuzzy logic based classifier is then presented, and the *Event Detector* is demonstrated on field data for performance evaluation. Additionally, we explore the benefits of self-learning for sensor calibration to enable the use of delta or processed-flow methods for reduction of false/missed alarms.

5.1.1 Literature Review

In the past, the industry has experimented with several methodologies to reduce missed/false alarms when detecting kick and lost circulation events. Historically, variations of the pit volume have been relied upon extensively to provide a clear indication of gains and losses from the wellbore (Anfinsen et al., 1992; Maus et al., 1979). Theoretically, these changes should match the volume of the drilled hole during drilling and the volume of the drillstring during pipe tripping. Therefore, abnormal variations in the pit volume or the trip tank are a direct sign of gain/loss problems. However, other drilling processes such as addition and removal of drilling fluid by the drilling crew and variations of the drilling fluid density (e.g. due to fluid compression and/or expansion) also have effects on the pit volume and make event detection difficult. It has been therefore common practice to monitor the flow-rate measurements out of

the well and compare it to the flow-rate reading from the mud pumps to get an indication of mud gain/losses (i.e the so-called delta-flow method) (Maus et al., 1979). This method can be erroneous during transient periods when the value of flow-in and flow-out do not match or when there is significant noise associated with the flow measurements (Schafer et al., 1992). Obtaining accurate flow readings from the high-pressure side of the mud circulation line is difficult, since measurement typically relies on counting the mud pump strokes and assuming an pump efficiency factor to derive volume pumped and flow rate. Accurate flow-meters for the low-pressure side such as a Coriolis flow-meter are costly and economically challenging for use on low cost drilling operations. Therefore, it is still common to rely on low accuracy sensors such as flow-paddles to have a measure of the flow-rate out of a well.

The use of pattern matching algorithms to identify gain patterns in the flow-paddle readings has been proposed to reduce the false alarms associated with using the delta-flow method for kick detection (Hargreaves et al., 2001). Although experimentally verified, these methods have not been widely adopted due to being computationally intensive for rig-site computers. Le Blay et al. (2012) and Daireaux et al. (2013) later proposed the use of detailed hydraulic models to calculate an estimate of the flow-rate out of the well and pit volume, and comparing these to sensor readings to detect abnormal variations (i.e. the processed-flow method). These methods are advantageous in that transients can be predicted to avoid false alarms. However, detailed hydraulics models require many well parameters to be entered and also require accurate flow-

meters for reasonable comparisons.

The use of annular discharge pressure and standpipe pressure for event detection was proposed by Reitsma et al. (2011) in order to avoid the use of a Coriolis flow-meter. For instance, an increase in both standpipe pressure and annular discharge pressure would signify the occurrence of a kick event. This method, however, is more applicable to managed pressured drilling, where the annulus is sealed and there is enough pressure build-up above the sensor to get reliable readings. In conventional drilling, the pressure sensor is open to the atmosphere. Therefore, it is difficult to obtain reliable readings of the annular pressure. Downhole pressure sensors have also been accepted as a reliable kick detection solution. The concept remains economically challenging for most drilling operations, as downhole pressure sensing can be costly. In addition, downhole pressure data is often only available during drilling when communication is possible with the downhole tool.

Several other methods have shown potential for early kick detection, but none of them has been widely adopted. Without being exhaustive, several are mentioned below. Acoustic sensors, for instance, can be used to measure the speed of a traveling pressure wave in the drilling fluid (Bang et al., 1994). In the case of a gas influx, the pressure wave speed would vary and the ultrasonic sensor would then detect the kick. This method was only tested in a laboratory environment, and was shown to work only for water-based drilling fluids. Recently, the use of electrical capacitance measurements was demonstrated for kick detection (Trivedi et al., 2014). By using a downhole tool,

the capacitance of the drilling fluid was measured in real-time with a lower capacitance signifying a kick. This method has not been field tested yet, and requires re-calibration of the sensor with different drilling fluids.

Pit volume gain/loss and delta-flow methods are widely used in the industry due to their relative simplicity and cost efficiency. Transient hydraulic models and pattern matching algorithms have also been employed to help reduce false alarms associated with these methods. However, there are two main issues that complicate these attempts: 1) the available pattern-matching methods alone appear insufficient in reducing false alarms, as they cannot deal with transient phenomena and require a lot of computational power at the rig-site; the transient hydraulics models can often be complex, requiring a large amount of input information and computational power as well; 2) the accuracy of an advanced hydraulics model must be matched with an accurate flow-meter. In the case of most land rigs which use simple flow-meters such as a flow paddle, the processed-flow method using an advanced hydraulics model does not provide an advantage.

Moreover, in discussing the results of event detection systems, the literature has often lacked sufficient analysis of the results. Early kick detection methodologies usually provide only a few examples of the algorithm detecting a kick. Most discussions also fail to present the number of false alarms and missed alarms of the detection algorithms. In the case of a rare event such as a kick, a detection algorithm not only has to be accurate but also has to minimize the number of false positives to e.g. avoid the "cry wolf" effect. A

recent paper by Brakel et al. (2015) addressed this issue by examining the false alarm rate of various kick detection indicators separately over a six months time period. It was shown that the false alarm rate can vary significantly, depending on the specific rig activity and the kick indicator used. It is therefore important to provide performance measures more than just accuracy, i.e. a measure of how the algorithm performs regarding both false and missed alarm rates in presence of imbalanced data.

5.2 Solution Overview

The foundation for the *Event Detector* is the idea of conditional pattern recognition (CPR). In CPR, pattern extraction and matching to pre-defined models of sensor trends only occurs upon observation of statistically significant change points in relevant signals. For example in the case of kick detection, a positive change in flow-out has to be observed before the classification task is conducted. Otherwise, events such as the intentional transfer of drilling fluid by the crew could result in false alarms. Therefore, the solution in this chapter employs a three-step process for detection of kick and lost circulation events during circulation periods, which consists of:

- Statistical change detection to find abrupt changes in the mud circulation sensors. This is an essential step to avoid false alarms that arise from normal rig activities that emit the same trends as the trouble events. Change detection triggers the pattern-marching and the classifier algorithms.

- Feature creation, specifically for the shape of the mud volume trend, which reveals important information about the potential trouble event.
- Classification of events, based on the features created during the previous step, as well as the dynamics of other mud circulation sensors.

In addition to event detection during circulation periods, a second module facilitates the detection of kick and wellbore breathing events during connections. The module has a fingerprinting feature, which through the establishment of a model, compares flowback trends at the onset of a connection to historical flowbacks. Meanwhile, the same feature creation technique used during circulation periods is used to assess the shape of the mud volume trend, and notify the user of trends that might raise the concern of a true kick. Finally, the event detection methodology in this chapter explores the concept of self-learning for calibration of the flow-out sensor, in order for the system to have the ability of performing a direct flow-in/flow-out comparison.

Figure 5.1 demonstrates the user interface of the RAPID EDS program, which was developed to facilitate historical and real-time testing of the event detection system at industry real-time operating centers. Aside from showing the alarms associated with both circulation and connection events, the UI allows the user to interact with the program and select various operation modes of the fingerprinting module.

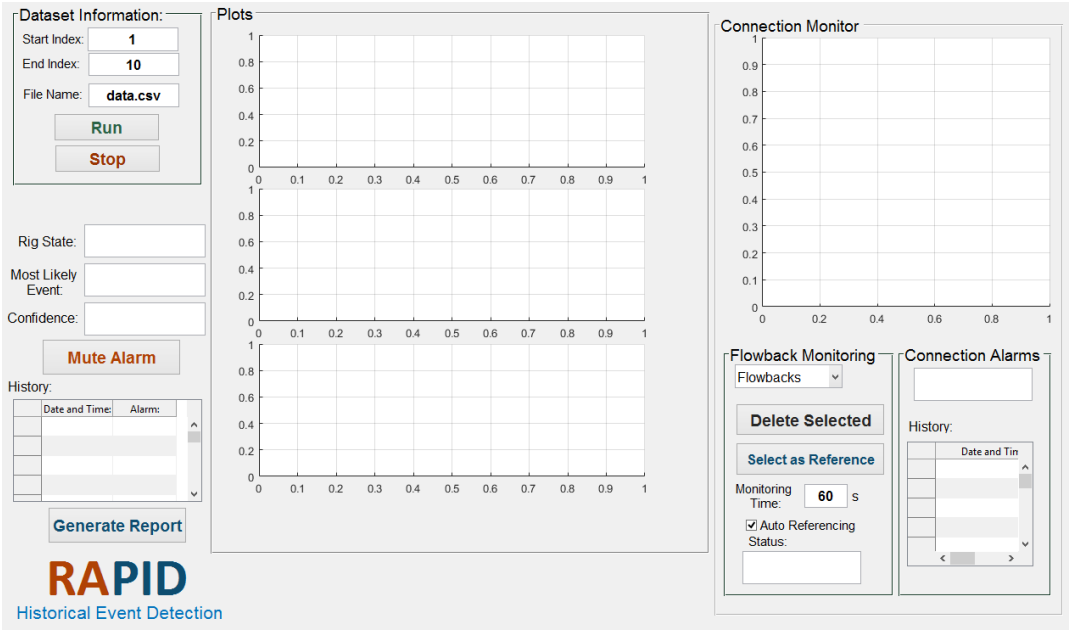

Figure 5.1: Overview of the RAPID EDS user interface developed for deployment at industry real-time operating centers (RTOC).

5.3 Statistical Gain/Loss Detection

The role of gain/loss detection is to find the abrupt changes that occur in the mud circulation sensors. In drilling event detection, this step is of paramount importance because the EDS shall only analyze mud volume changes for their shape that correspond to true statistical changes in flow and pressure sensors. Otherwise, other events such as pit transfers could be falsely identified as kick or lost circulation. In the EDS program, change detection is specifically applied to the flow-out, standpipe pressure and the flow-in sensors. Later in this chapter, we will discuss how the combination of various changes can be used to activate a different event classifier.

Several methods are available for change detection in pattern recognition literature. A simple method would use conventional outlier detection techniques that rely on a moving average window, where a prediction at time t_{i+1} is made based upon an average of the observations at $t_i : t_(i + w)$,where w is the window size:

$$x_{t+1} = \frac{1}{W}(x_t + x_{t-1} + \ldots + x_{t-W-1}) \tag{5.1}$$

A gain alarm can then be raised by fitting a Gaussian distribution to the selected window with mean, μ, and standard deviation, σ, and assessing if:

$$\text{Alarm Level} = \phi(\max(0,x_{t+1} - \mu)/\sigma) \tag{5.2}$$

where ϕ is the cumulative distribution function for N samples. This method, however, can be prone to high false alarm rates if the level of noise and periodic changes in the input signal is high (Gottman, 1981). This method can be improved through various dimensionality reduction techniques that essentially provide a filtering medium for the data (namely the SAX method, detailed in the next section).

In comparison to frequentist methods, a Bayesian change point detector focuses on causal inference through the generation of a distribution of the next unseen datum in the data sequence. In this section, we will also adopt an algorithm developed by Adams and MacKay (2007) to the problem of detecting abrupt changes in the mud circulation sensor data. Let us assume that a sequence of flow-out data is represented by $x_1, x_2, ...x_T$, and that the

data is divided into several partitions by change points, some of which might correspond to the onset of the connection making period, or a kick event. We can then assume that for each partition, ρ, the data points within are independently represented by a distribution $P(x_t|\eta_p)$ where the parameters, η_p, are assumed to be independent as well. We are now concerned with the problem of estimating the posterior distribution over the current time since the last change point, given the data observed so far.

The algorithm of Adams and MacKay (2007), is visually described in Figure 5.2 which shows how a change point model is expressed in terms of run lengths (duration of the signal before a change point occurs). In Figure 5.2(a), an arbitrary univariate data is divided into three segments of run lengths 4, 6 and 4 respectively. These run lengths are shown as a function of time in Figure 5.2(b) where they drop to zero when the change point occurs.

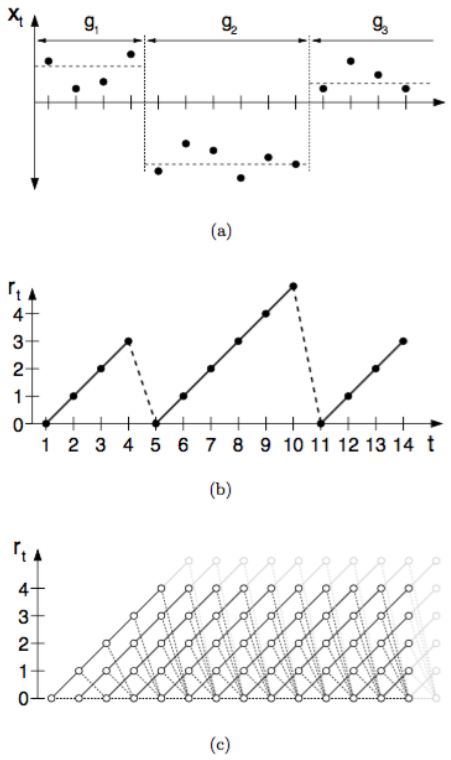

Figure 5.2: Demonstration of Adam's change point detection algorithm (Adams and MacKay, 2007).

Figure 5.2(c) is a depiction of the algorithm, where solid lines indicate that probability is being passed upwards and the run length increases as a result. The dotted lines in turn demonstrate the possibility that the current run is over and the run length changes to zero. The recursive estimation of a run length is based on the idea of computing the predictive distribution, conditional on a given run length r_t. The algorithm then integrates over the posterior distribution on the current run length to find the marginal predictive

distribution given by:

$$\pi(t) = P(x_{t+1}|X_{1:t}) = \sum_{r_t} P(x_{t+1}|r_t, X_t^r,)P(r_t|X_{1:t}) \tag{5.3}$$

Conjugate-exponential models are used in this algorithm for integrating with the change point detection scheme. This is due their ability to allow inference using sufficient statistics which can be incrementally calculated (Adams and MacKay, 2007). For a univariate Gaussian distribution, the conjugate prior for the precision is given by a gamma distribution of the form:

$$Gam(\lambda|a, b) = \frac{1}{\gamma(a)} b^a \lambda^{a-1} e^{-b\lambda} \tag{5.4}$$

where γ is the gamma function. With a univariate Gaussian together with a Gamma prior and integrating out the precision, we obtain the marginal distribution in the form of:

$$St(x|\mu, \lambda, \varphi) = \frac{\gamma(\frac{\varphi}{2} + \frac{1}{2})}{\gamma(\frac{\varphi}{2})} (\frac{\lambda}{\pi\varphi})^{1/2} [1 + \frac{\lambda(x-\mu)}{\varphi}]^{-\frac{\varphi}{2}-\frac{1}{2}} \tag{5.5}$$

which is known as the Student's t-distribution (Bishop, 2007). In the equation above, $\lambda = a/b$ and $\varphi = 2a$. The Student's t-distribution can be used for the estimation of the predictive probability expression in Equation 5.3. In the algorithm described here, the change point prior, $P(r_t|r_{t-1})$, takes the form of (Adams and MacKay, 2007):

$$P(r_t, r_{t-1}) = \begin{cases} H(r_{t-1}+1), & \text{if } r_t = 0 \\ 1 - H(r_{t-1}+1), & \text{if } r_t = r_{t-1} + 1 \\ 0, & \text{otherwise} \end{cases}$$

where $H(\tau)$ is the hazard function. In the case of a discrete exponential distribution, $H(\tau)$ is given by: (Adams and MacKay, 2007):

$$H(\tau) = \frac{1}{\alpha} \tag{5.6}$$

where α is a timescale constant. Given we have the expressions for the predictive probability distribution, and the prior for run length growth and change points, we can recursively estimate the growth and change point probabilities by (Adams and MacKay, 2007):

$$P(\text{Growth}) = P(r_t | r_{t-1} + 1, X_{1:t}) = P(r_{t-1} + 1, X_{1:t-1})\pi(t)(1 - H(r_{t-1} + 1)) \tag{5.7}$$

$$P(\text{Change}) = P(r_t = 0, X_{1:t}) = \sum_{r_{t-1}} P(r_{t-1} + 1, X_{1:t-1})\pi(t)H(r_{t-1} + 1) \tag{5.8}$$

Both the outlier detection and the Bayesian change detection algorithms discussed above are implemented on 4 different cases of statistically changing flow-out signals. Two cases are shown in Figure 5.3, where moderate levels of noise on the scale of $\pm 10\%$ exist in the flow-out signal. Evidently, both the SAX-based outlier detection and the CPD algorithm perform similarly, with the outlier detection algorithm identifying false changes when the noise levels increase around $t = 1000s$.

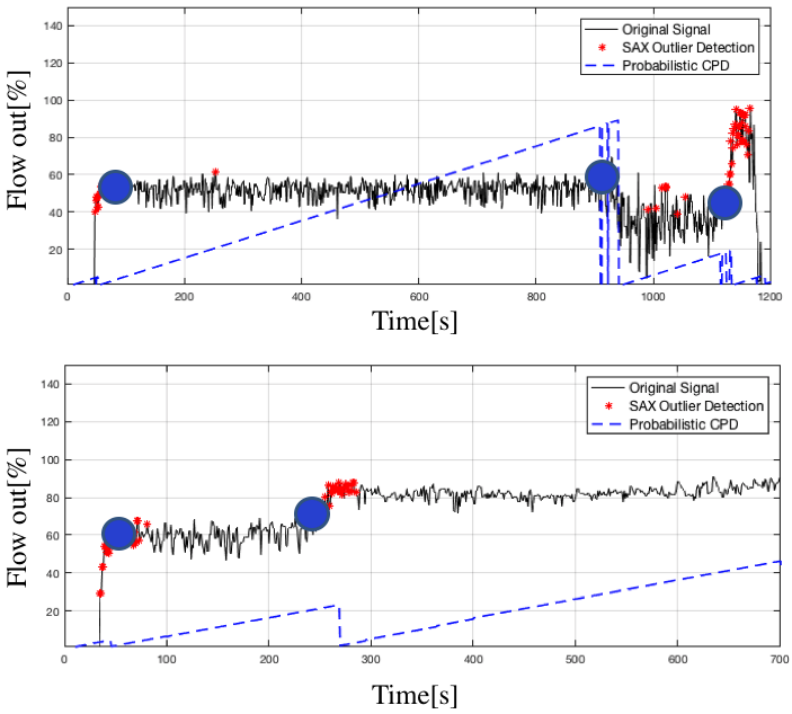

Figure 5.3: Comparison of the SAX based outlier detection (shown in red) and the change-point detection algorithms (shown in blue) with a moderately noisy flow paddle signal. The dashed blue line represents the run-lengths of the CPD algorithm.

In Figure 5.4, two different cases of flow-out changes are shown: the one on the top portrays a case of severe noise, and the one on top bottom shows very low noise levels of $\pm 2\%$. In both cases of Figure 5.4, the outlier detection algorithm performs significantly worse than the CPD, where it either produces too many false change points, or fails to catch a subtly changing flow-out trend. The performance benefits of the CPD algorithm are therefore evident in a variety scenarios when the signals are either too noisy or when

the changes are small and slow. Since robustness to such scenarios is key for reduction in missed and false alarms, the CPD algorithm was selected for the EDS program.

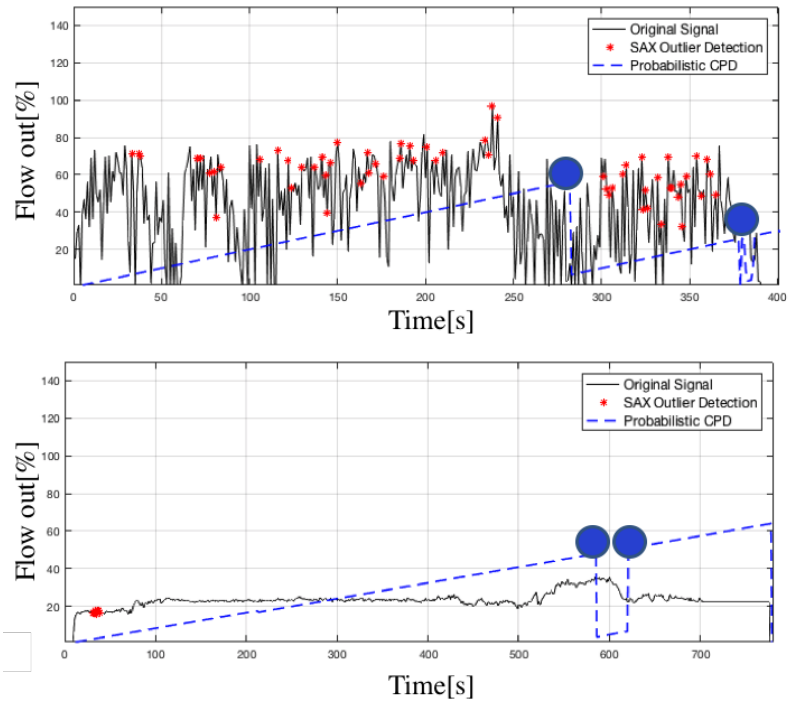

Figure 5.4: Comparison of the SAX based outlier detection (shown in red) and the change-point detection algorithms (shown in blue) with both a highly noisy (upper plot) and noise-free (lower plot) flow paddle signal. The dashed blue line represents the run-lengths of the CPD algorithm.

5.4 Pattern Matching

The role of the pattern matching algorithm is to create the features used by the classifier to perform event classification. Pattern matching requires *a*

priori knowledge in regards to the shape of a sensor signal that corresponds to a particular event. This prior knowledge can then be combined with observed trends in the sensor signal segments at the classification stage for decision making.The most important pattern feature is the shape of the mud volume trend, which can indicate the difference between an influx and a wellbore breathing event.

We propose the use of the Symbolic Aggregate Approximation (SAX) method to perform this task. SAX has been recently adopted as a powerful technique in time-series data mining because of its simplicity and allowing for dimensionality reduction (Lin et al., 2007). The SAX method transforms a time-series X of length n into a string of arbitrary length ω, where $\omega \ll n$, using an alphabet A of size $a > 2$. The SAX method relies on the fact that normalized time-series signals have a high Gaussian distribution. By finding the breakpoints that correspond to the alphabet size, we can obtain equal-sized areas under the Gaussian curve. In the first step of the SAX algorithm, the signal is normalized and transformed into a PAA (piecewise aggregate approximation). In PAA, the signal is divided into equal-sized frames, and the mean value of the points that lie in every frame is computed (Lin et al., 2007). The lower dimensional vector of the original time series signal is the vector whose components are the means of all successive frames of the signal. Use of PAA at the first step brings the advantage of a simple and efficient dimensionality reduction while providing the important lower bounding property. In the last step, the PAA representation is discretized and converted into an alpha-

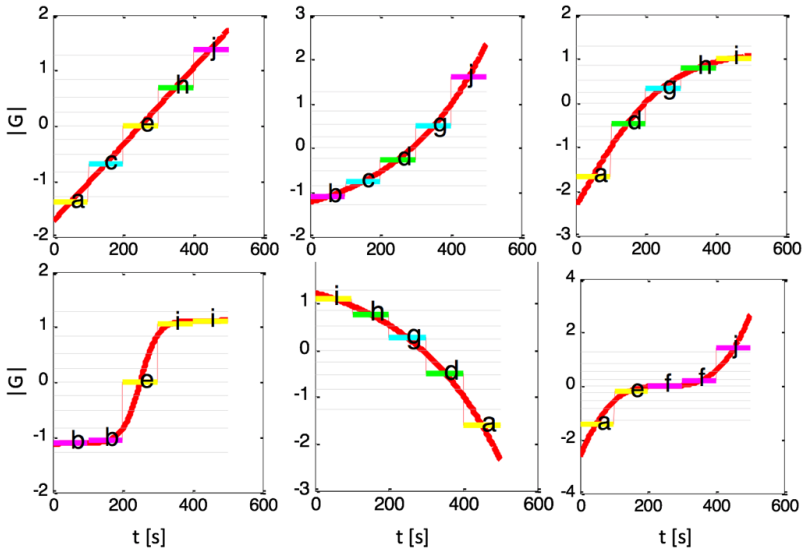

Figure 5.5: SAX demonstration of various potential pit volume trends, including linear (upper left), exponentially increasing (upper middle), and exponentially decaying (upper right).

betical representation using a look-up table. Figure 5.5 demonstrates various signal trends and their corresponding alphabetical representation. These alphabetical representations can be acquired in real-time for signal segments and compared to a set of base alphabetical representations of known models for signal.

Several measures can indicate a similarity between two signals. A cosine similarity, for instance, measures the cosine angle between two vectors. A FFT (Fast Fourier Transform) analysis represents a non-periodic signal by a continuous superposition of complex exponentials, thereby enabling a direct comparison of the two signals. In this work, we employ the Edit Distance (ED) algorithm to determine the similarity between two sensor signals. This algo-

rithm builds on the benefits of the SAX method and compares the similarity between two strings by counting the number of operations required to convert one string to the other (Marzal and Vidal, 1993). These operations can include the "insertion", "deletion", and "substitution" of a character. The algorithm therefore starts at an initial state and finds the shortest path to converting the first string to the second. Let us assume comparison to a linear trend as shown in Figure 5.5. If the obtained signal has a growing exponential shape, it will take 3 operations to convert the first string to the second. In the case of a *tanh* signal, however, the number of operations to convert the two strings is 4. The counting of operations can be performed in an automated manner using dynamic programming.

5.5 Event Classification

Classification of specific trouble events can be made using a supervised-learning type classifier, or a rule-based system. A trained classifier can be simpler to design and implement, but it requires access to large training data, with a balanced distribution of all the events of interest. Rule-based systems require the designer to essentially think of all the various combinations of the feature set, and associate each combination with the appropriate event. A benefit of such a system is that expert knowledge can be directly built into the system, and the need for training data is avoided.

Fuzzy classification based on fuzzy logic provides a comprehensive way of embedding linguistic knowledge into a classifier and aggregating the feature

set to arrive at a final outcome, with a quantifiable degree of certainty. Since the problem of classifying hydraulic events does not involve a large feature space, fuzzy logic is an appropriate framework for the classification task. In addition, a fuzzy based classification allows for associating a degree of certainty with each output decision.

In a Mamdani fuzzy inference system (FIS), fuzzy rules enable the creation of linguistic statements that describe how the classifier should make its decisions (Kuncheva, 2000). Through a fuzzification process, numerical inputs from hydraulic sensors can be converted into linguistic terms using a set of membership functions. These functions can represent fuzzy concepts such as "small", "large", "high". In making a fuzzy rule, these memberships can be combined using conventional "and", "or", and "not" operators. For instance, "and" can be computed using the product operator $u_A(x) \times u_B(x)$, where $u_A(x)$ represents the membership in class A, and the "or" can be computed using $u_A(x) + u_B(x) - u_A(x) \times u_B(x)$ (Kuncheva, 2000).

The classification of kick and lost circulation events is performed using two different fuzzy inference systems. For the kick detection task, the input variables include the shape and size of the mud volume change, the size of the flow-out increase, and the sign of the standpipe pressure change. For the lost circulation FIS, the input variables are the size and shape of the mud volume change. Table 5.1 demonstrates the input and output variables of the FIS systems, as well as the activation conditions for each FIS. For the Gain FIS system to be activated, a positive change in the flow-out trend with no

changes in flow-in must be observed within a certain time-frame. The Loss FIS is similarly activated when both a negative change in flow-out and standpipe pressure are observed.

FIS	**Change Activator**	**Input Variables**	**Output Variables**
Gain	$+f_{out} \cap \bar{f_{in}}$	mv_{size}, mv_{shape}, spp_{sign}, $f_{out,size}$	P(gas influx), P(influx) , P(breathing)
Loss	$(-f_{out} \cap -spp) \cap \bar{f_{in}}$	mv_{size}, mv_{shape}	P(loss)

Table 5.1: Activation conditions, and input/output variables of the fuzzy inference systems for gains and losses.

The rule surface of the Loss FIS is simpler to display, because it only has two input variables as shown in Figure 5.6 . To demonstrate how certain events are inferred in the Gain FIS system, various input-output relationships are shown in Figures 5.7 through 5.10. In Figure 5.7 for instance, the output variables all display an event probability of less than 0.5, with all events being similarly likely, due to the specific combination of the input variables,

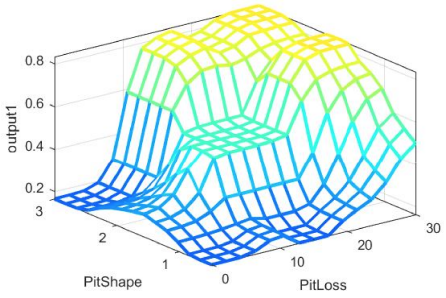

Figure 5.6: Fuzzy decision surface for lost circulation classification.

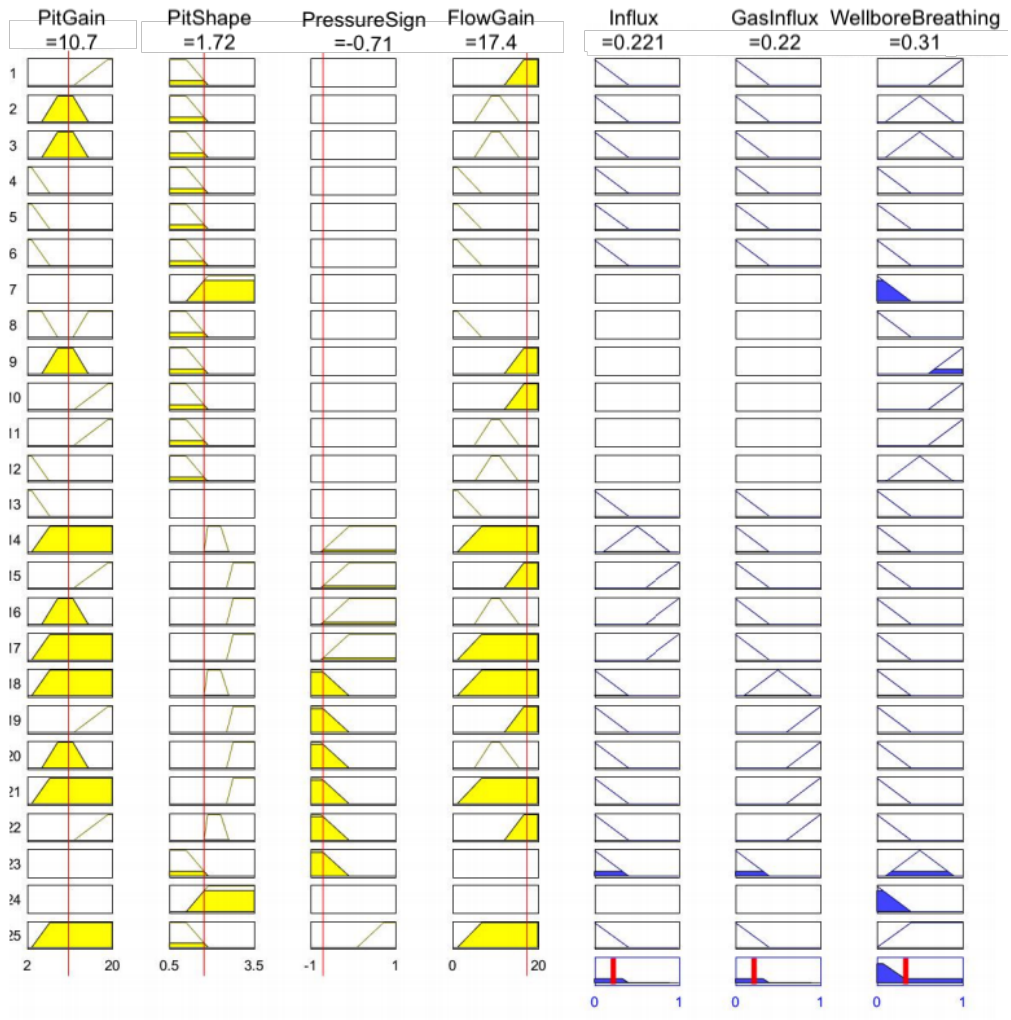

Figure 5.7: Particular combination of the input variables of the Gain FIS system that results in all events having a probability less than 0.5.

In the case of Figure 5.8, the highest probability output is identified as a wellbore breathing event, due to a positive sign of pressure change as well as a decaying mud volume shape.

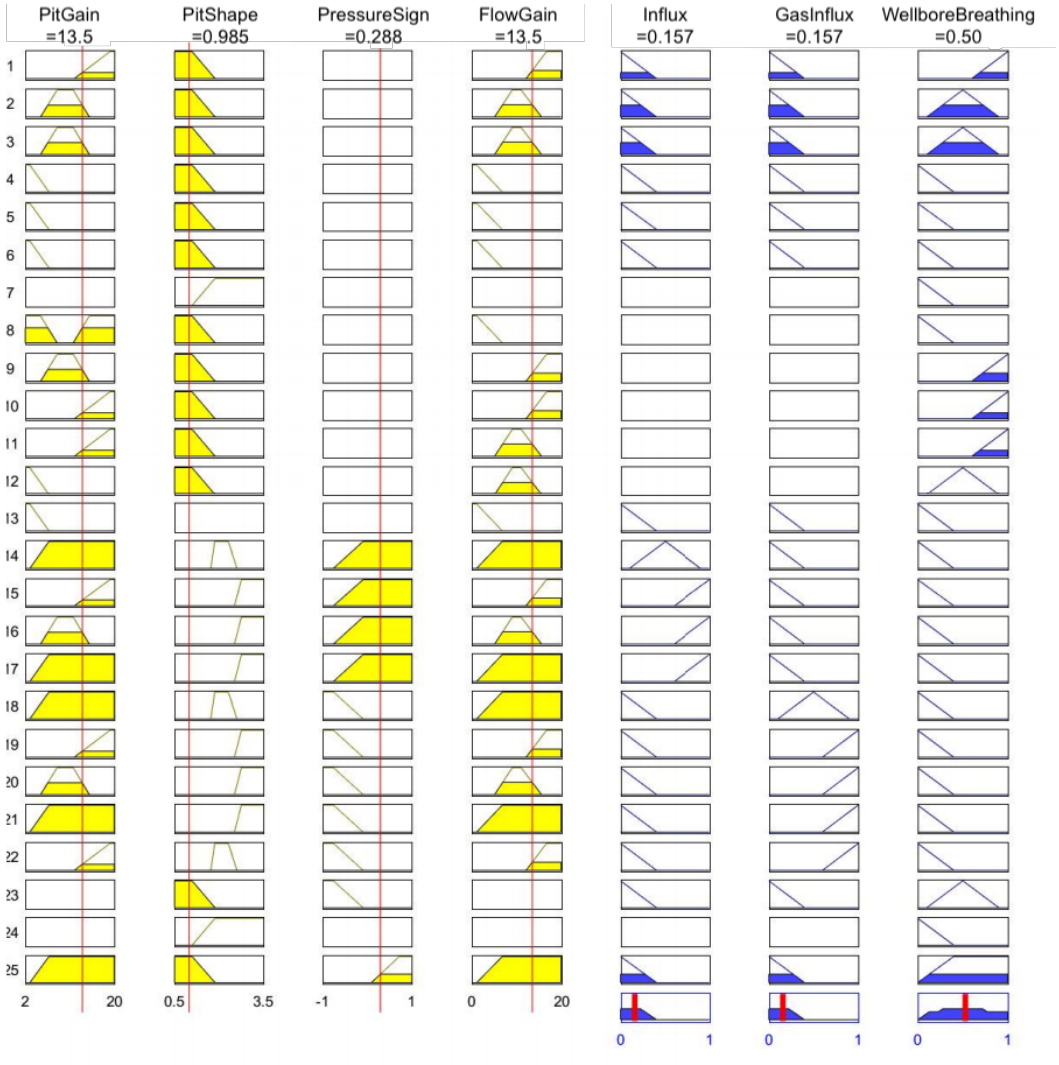

Figure 5.8: Particular combination of the input variables of the Gain FIS system that results in a wellbore breathing event being the most probable event with $P(\text{breathing}) = 0.5$.

With a change in the shape of the mud volume trend to exponentially increasing, the probability of the wellbore breathing is reduced, and the most likely event becomes an influx, as shown in Figure 5.9.

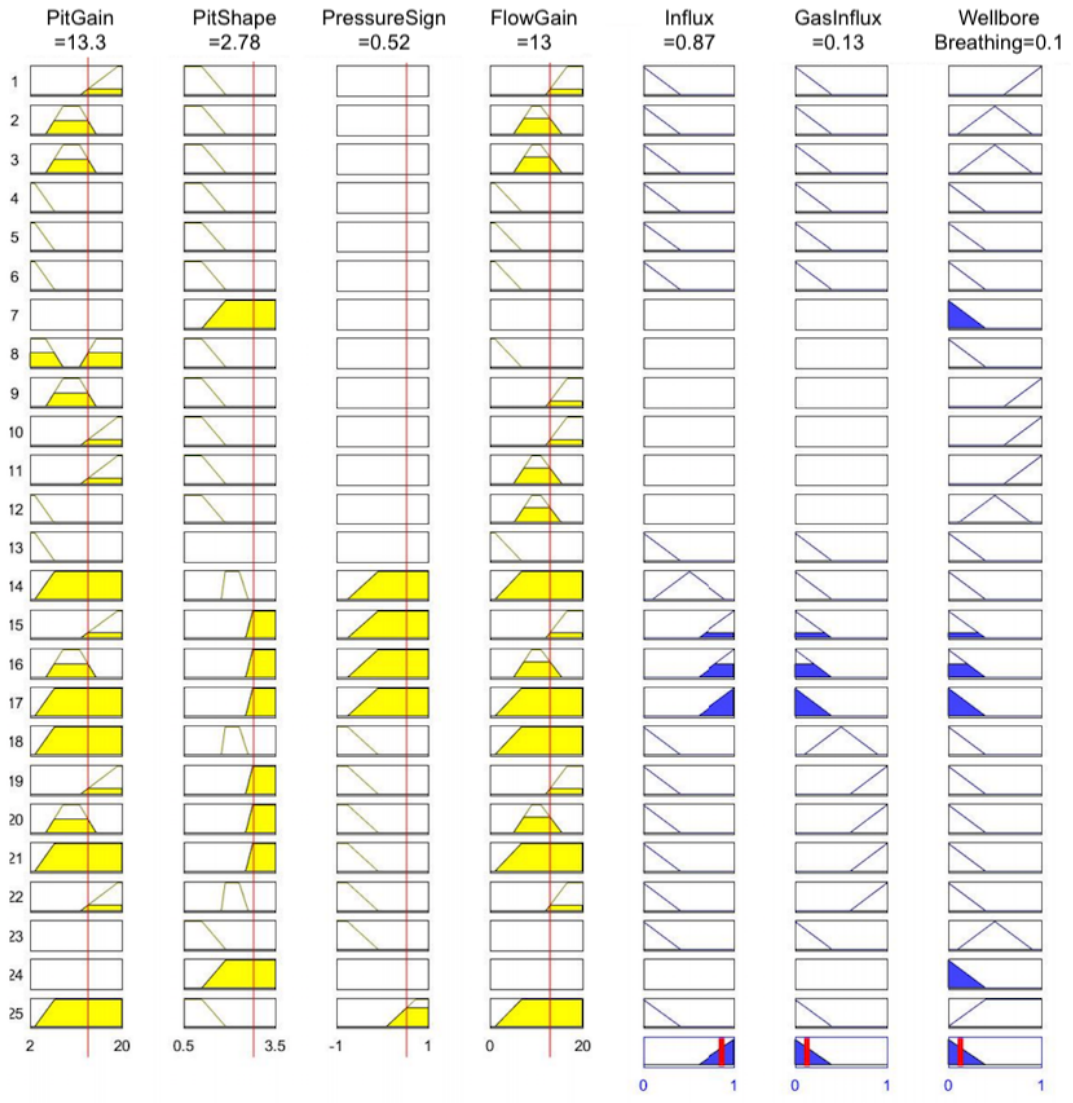

Figure 5.9: Particular combination of the input variables of the Gain FIS system that results in an influx event being the most probable event with $P(\text{influx}) = 0.87$.

Finally in Figure 5.10, a switch in the sign of the pressure change and increased gain in flow-out and mud volume signals results in the inference of

a gas influx event with a probability of 0.7.

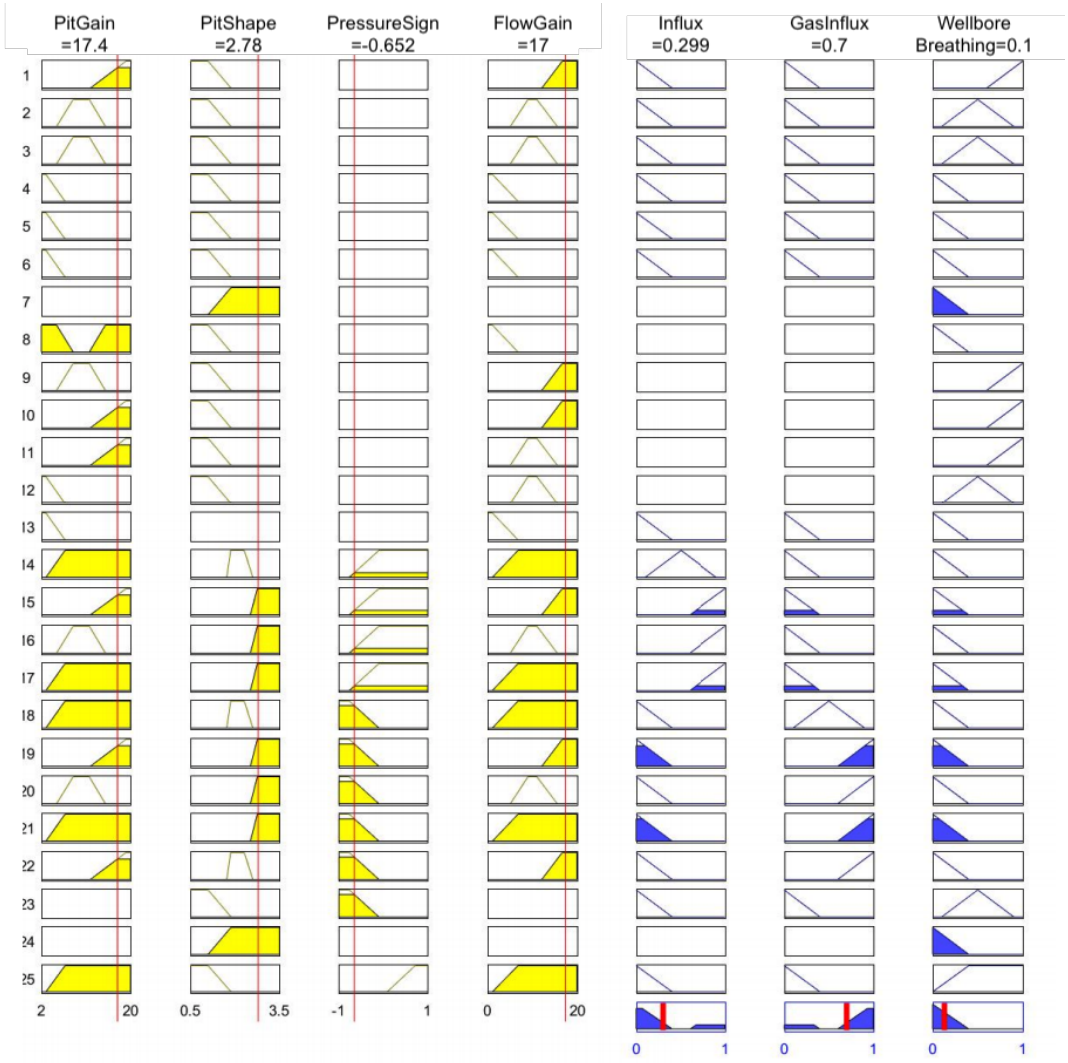

Figure 5.10: Particular combination of the input variables of the Gain FIS system that results in a gas influx event being the most probable event with $P(\text{gas influx}) = 0.7$.

5.6 Sensor Calibration for Direct Flow-in/Flow-out Comparison

As noted earlier in this chapter, a separate module in EDS is devoted to performing sensor calibration, such that a direct comparison of the flow-in and flow-out signals (delta-flow) can be performed. While the delta-flow comparison is a popular method for identification of kick and lost circulation events in the drilling literature, it is only possible when accurate flow sensors are present both on the inlet and the outlet of the mud circulation loop. Most commonly however, the measurement of the flow-out from the annulus is performed using a flow-paddle sensor. A flow paddle measures the height of the returning drilling mud from the annulus that is traveling in an open conduit. Therefore, the measurement is performed in the units of [%] of a total height, and does not directly translate to a flow-rate unit. For the calibration task, we are therefore interested in learning a nonlinear mapping from flow-out in [%] units to the actual value of flow-out in *gpm*. The target, which is the flow-rate in *gpm* units, is available to us as the flow-in reading itself, given no deviations between flow-in and flow-out trends exist. This assumption holds true through the simple relationship of $f_{in} = f_{out}$ in steady state conditions, when no losses or gains are present in the mud circulation system. In Figure 5.11, the idea is that the steady state value of flow-out should match flow-in during normal operation, except during the transient periods.

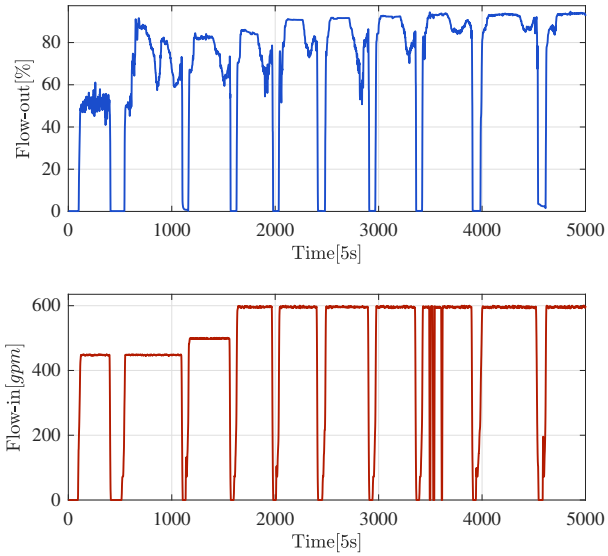

Figure 5.11: Uncalibrated flow-out (upper plot) and flow-in (lower plot) data from drilling 10 stands. The majority of the data only covers a 400gpm to 600gpm range.

One way of performing this calibration task is to use the nonlinear regression techniques already discussed in Chapter 3, and apply them to historical data with the flow-out data in [%] as the input, and the flow-in data in gpm as the target. A calibration performed in this manner should only contain data since the last change in the rheological properties of the drilling fluid. Figure 5.12 shows the results of this calibration, on data from 10 stands of drilling activity. The calibration function was trained on the first 4 stands, and tested on all 10 stands. Although the support vector method works sufficiently well in comparison to the random forest and the ANN, the performance of calibration task has room for improvement. Specifically, the imbalance in

the "levels" of input and target data, seems to not present the learner with enough examples to learn the true nonlinear function from.

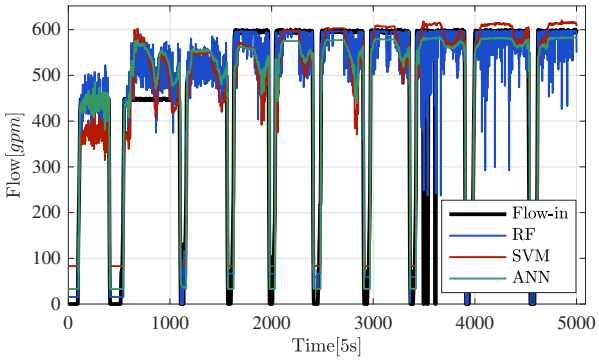

Figure 5.12: Calibrated flow-out measurements using various regression techniques, and flow-in data from drilling 10 stands.

Consider the case of Figure 5.13. The training task for calibration is performed on the first 80s of the data during an active period where the flow-rate is ramped from 0 to 800gpm. The calibration is then used for prediction of the flow-out signal for $t > 200$. With the training using examples only from $t < 80$, the prediction results resemble the same performance depicted in Figure 5.12. Similarly, the SVM with a polynomial kernel overestimates the flow-out response, but still performs better than the other methods.

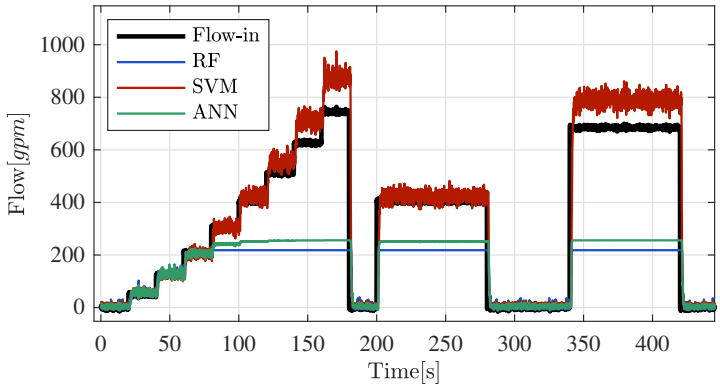

Figure 5.13: Action-driven calibration of flow-out, using data from a flow-in range of 0-200gpm. Training performed for $t < 80$.

When the training period is increased to include more training examples that span a higher flow range, the prediction performance of the SVM method is significantly improved. In the final case in Figure 5.15, where the training includes the full flow range, all predictions have improved and can trivially predict the flow-out response.

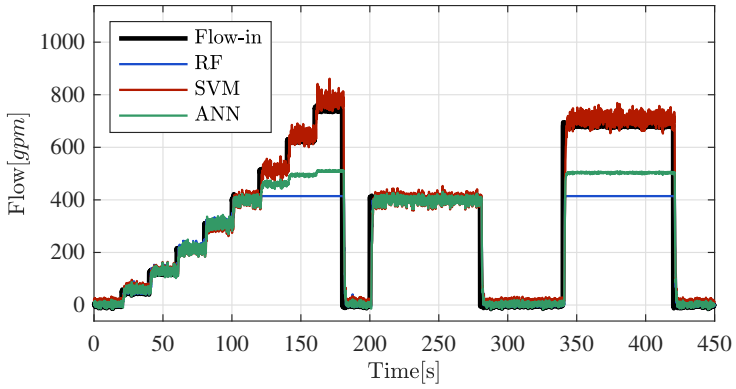

Figure 5.14: Action-driven calibration of flow-out, using data from a flow-in range of 0-400gpm. Training performed for $t < 120$.

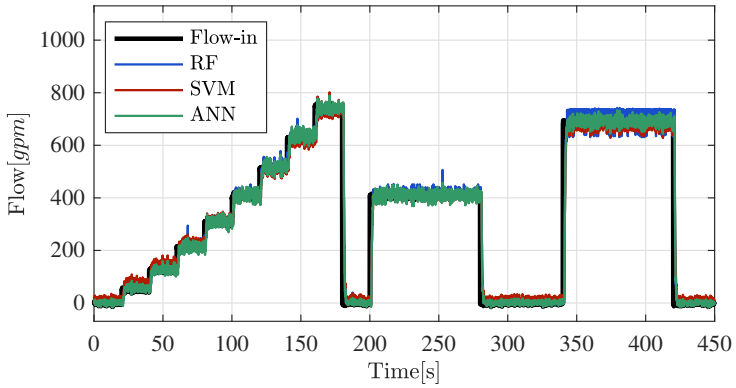

Figure 5.15: Action-driven calibration of flow-out, using data from a flow-in range of 0-750gpm. Training performed for $t < 180$.

The finding from Figures 5.13 through Figure 5.15 is that while it is preferred for the calibration task to use data from the full range of possible flow values for optimum prediction results, it is also possible, by using the right calibration method, to achieve sufficient results by training on at least 50% of the maximum expected flow range. This finding is also shown in Table 5.2, where the mean squared prediction results of all calibration methods are shown. Not only is the SVM method the most accurate, but its performance stays fairly consistent when going from 120s training to a 180s training. Due to overfitting to the smaller range of data in 80s and 120s training, the ANN and RF methods are not capable of predicting higher-than-experienced flow-rates, and therefore only perform well when they are exposed to a 180s training.

Learner	**180s training**	**120s training**	**80s training**
Random Forest	5.8510e+03	2.3484e+04	7.1722e+04
Neural Network	5.4972e+03	1.3974e+04	5.9870e+04
Support Vector Regression	5.4782e+03	5.8731e+03	9.4354e+03

Table 5.2: Mean squared prediction error of three regression techniques for calibration of the flow-out sensor.

To demonstrate the benefits of the delta-flow comparison, note the field scenario in Figure 5.16. In this scenario, an accidental change in the flow-out signal occurs just prior to when the crew begins adding mud to the pit tanks. As is evident, a purely pattern-based logic would produce a false alarm, since it associates the detected change in flow-out with the increase in mud volume. However, a threshold-based comparison of flow-out and flow-in (50gpm in this case) serving as an additional feature to the event classifier would prevent the false alarm by highlighting the fact that the gain in flow-out is physically insignificant.

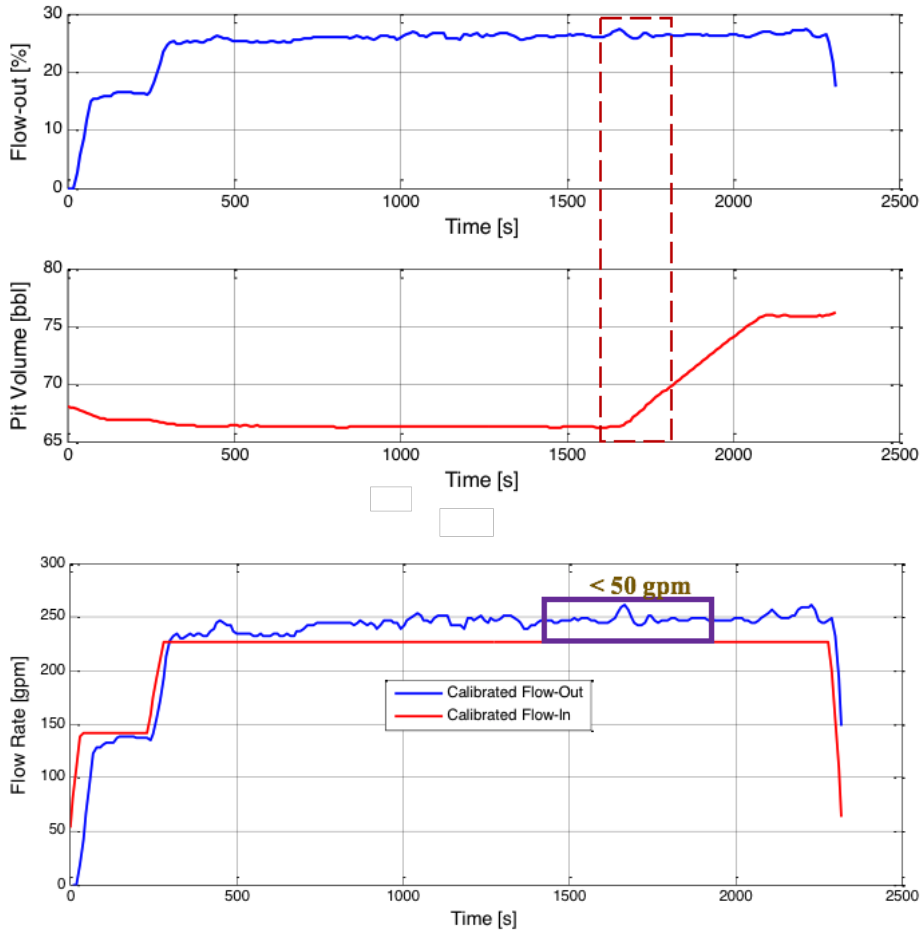

Figure 5.16: Demonstration of delta-flow method benefit for avoiding a false alarm. The upper plot shows an uncalibrated flow-out, and a change-point followed by mud transfer in the pits (shown in the middle plot). The bottom plot shows the overlay of calibrated flow-out and flow-in trends, and the fact that the change-point corresponds to a small change in absolute flow.

In essence, an active calibration of the flow-out sensor using a ramping procedure shown in Figure 5.14 enables a direct flow-out/flow-in comparison. This additional comparison can assist in false alarm avoidance by identifying

changes in the flow-out signal that are statistically significant, but physically negligible.

5.7 Flowback Fingerprinting

The fingerprinting feature of EDS enables historical analysis of mud flowback trends during the onset of connections. The purpose of this analysis is to distinguish between normally occurring flowbacks due to the mud in return lines, and wellbore breathing events. Flowbacks due to the mud in return lines are ideally repeatable, whereas wellbore breathing, while still having a decaying trend, can be much bigger in volume. When the first flowback is observed, the program fits an exponential model to the trend, using nonlinear least squares. An error bar diagram is then created, as shown in Figure 5.17 which signifies the upper-bound of a safe-zone for consecutive flowback trends. If future trends have a similar shape to the previous flowback, the model can be automatically updated to take statistical variations into account. Otherwise, the flowback is considered "abnormal", and an alert is generated. The EDS user can also discard flowback trends manually, or select a specific trend as the sole reference for the model. A separate module continuously checks for shape of the flowback trend, to ensure it does not show the signature of a kick.

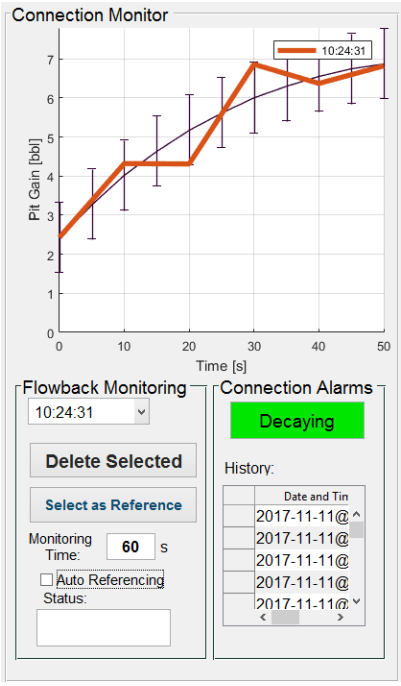

Figure 5.17: Flowback monitoring screen in the RAPID EDS program. The main plot shows the mud gains during the onset of connection making. The user can select a specific flowback trend from the past using the drop-down menu as shown, or delete it if deemed irrelevant. Auto-referencing allows the user to include all similar looking past flowbacks in derivation of the statistical model. After the flowback "monitoring time" has ended, EDS continuously checks if the connection flowback starts exhibiting an exponentially increasing shape.

5.8 Case Studies

In the final section of this chapter, we discuss the testing results of the EDS program on several data-sets from onshore and offshore drilling operations that contain real kick, lost circulation and wellbore breathing events. These data-sets were generously provided by the sponsors of the RAPID consortium,

including the Hess Corporation, Pason Systems, and the Apache Corporation. The scenario depicted in Figure 5.18 shows an example of a real influx being detected. The influx triggers positive change-points in both the flow-out and standpipe pressure signals, and shortly after a $2bbl$ gain in the mud volume is observed. The system successfully ignores the mud transfer that occurs at $t = 1600s$.

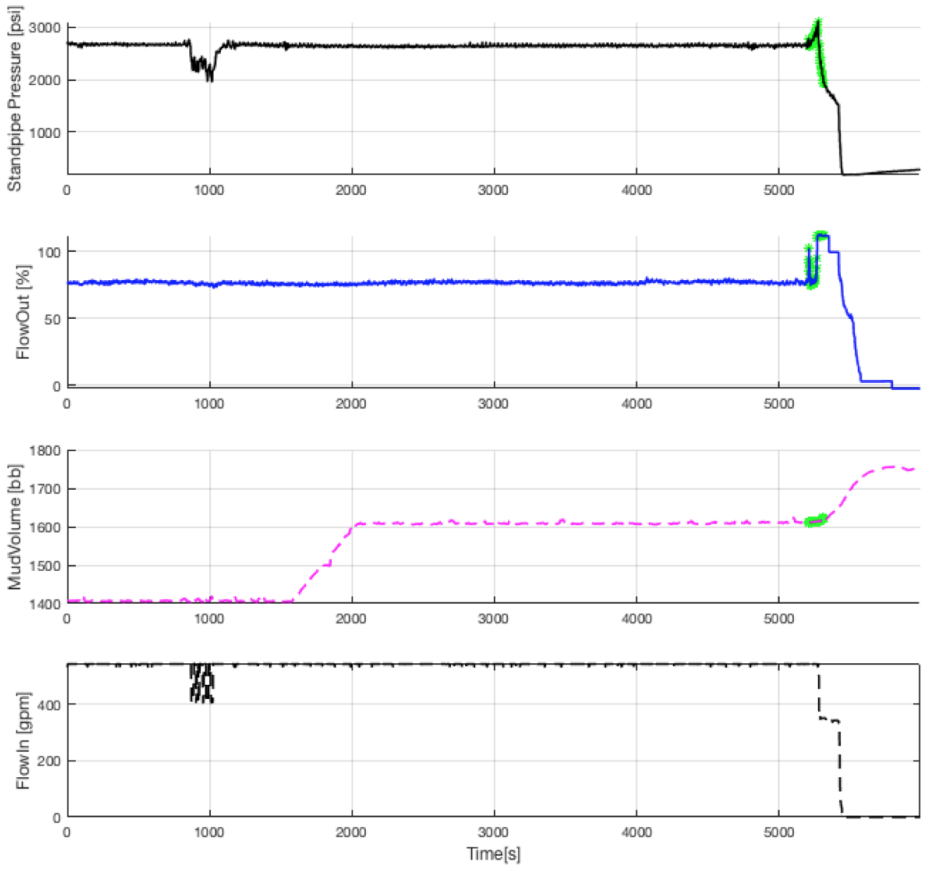

Figure 5.18: Demonstration of the 4 mud circulation sensor trends when an influx occurs, and is detected by EDS.

Figure 5.19 is the case of a lost circulation event. Prior to catching the event at $t = 6300s$, EDS encounters several periodic fluctuations in both flow-out and mud volume, which could have potentially caused false alarms. However, when a simultaneous negative change-point in flow-out and standpipe pressure is detected at $t = 6300s$, the algorithm begins to look for losses in mud volume. The severity of the alarm generated for this event is a function of the shape and size of the loss, as already noted in Section 5.5.

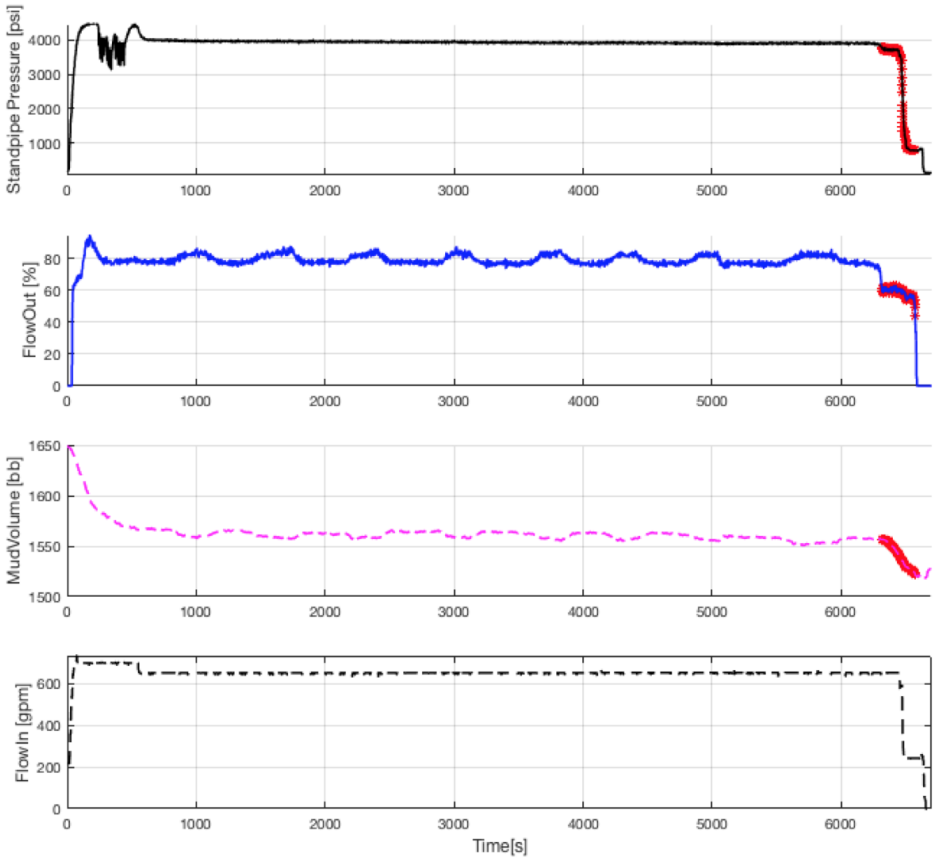

Figure 5.19: Demonstration of the 4 mud circulation sensor trends when a lost circulation event occurs, and is detected by EDS.

Figure 5.20 shows the output the EDS program in response to a gas influx. The alerts for this event were generated at 7:23, roughly 2 minutes before the pumps are shut off, in response to a rapidly increasing flow-out and slowly dropping standpipe pressure. What is interesting to note is the flowback signature at 6:54, when after 20 seconds the trend diverges from the fingerprinting model and keeps increasing, only to be interrupted by the starting of the pumps again. With access to a fingerprinting tool such as the one in the EDS, an obvious divergence of the flowback signature could have potentially provided an indication to the crew several minutes ahead of the actual event.

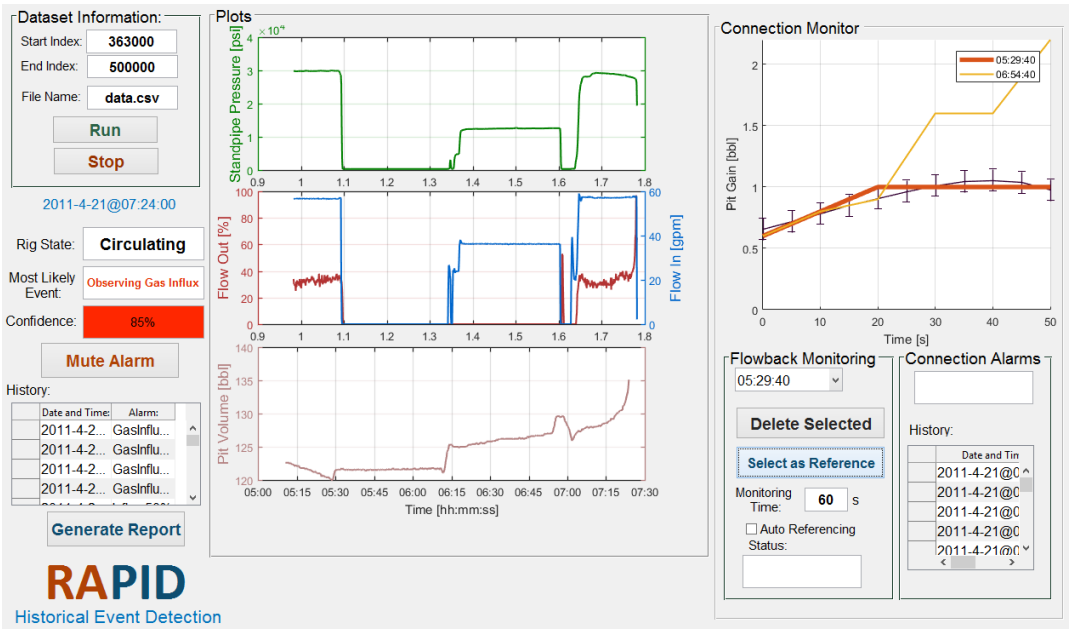

Figure 5.20: Correct identification of a gas influx event with an 85% confidence. Note the flowback at 6:54 demonstrates an abnormal signature compared to the flowback model.

Figure 5.21 depicts a scenario where a very large $> 100bbl$ wellbore breathing incident occurs during an offshore operation. The event is first identified as an influx at 13:17 by EDS. Once the pumps are shut off however, the mud volume keeps increasing. The circulation alerts of the EDS stop and as shown, the connection alarms indicate a decaying a flowback trend, which signifies correctly a wellbore breathing event.

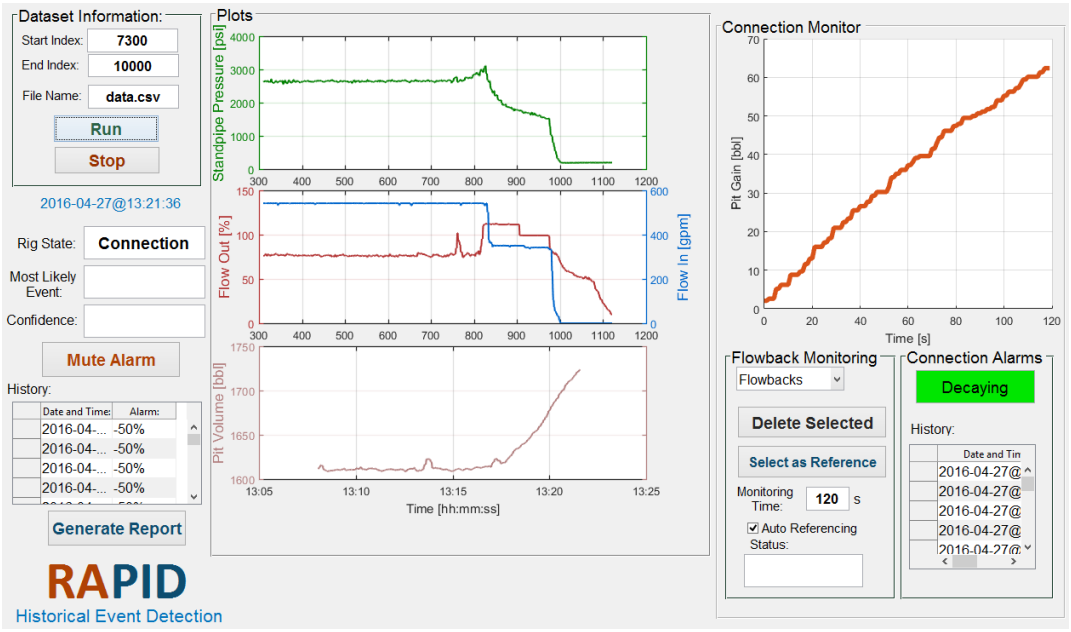

Figure 5.21: Correct identification of a wellbore breathing event during pumps-off, after initially indicating an influx event. The large gains continue well into $120s$ after the pumps are shut-off, but EDS shows a safe "decaying" status.

Figure 5.22 is another example of a much more subtle wellbore breathing event. The connection fingerprinting feature has initially updated the model to include the flowbacks that occurred at 10:24, and 12:10. Starting at

14:03, all future flowbacks begin to fall outside the model's safe zone. The rig crew began reporting wellbore breathing prior to midnight.

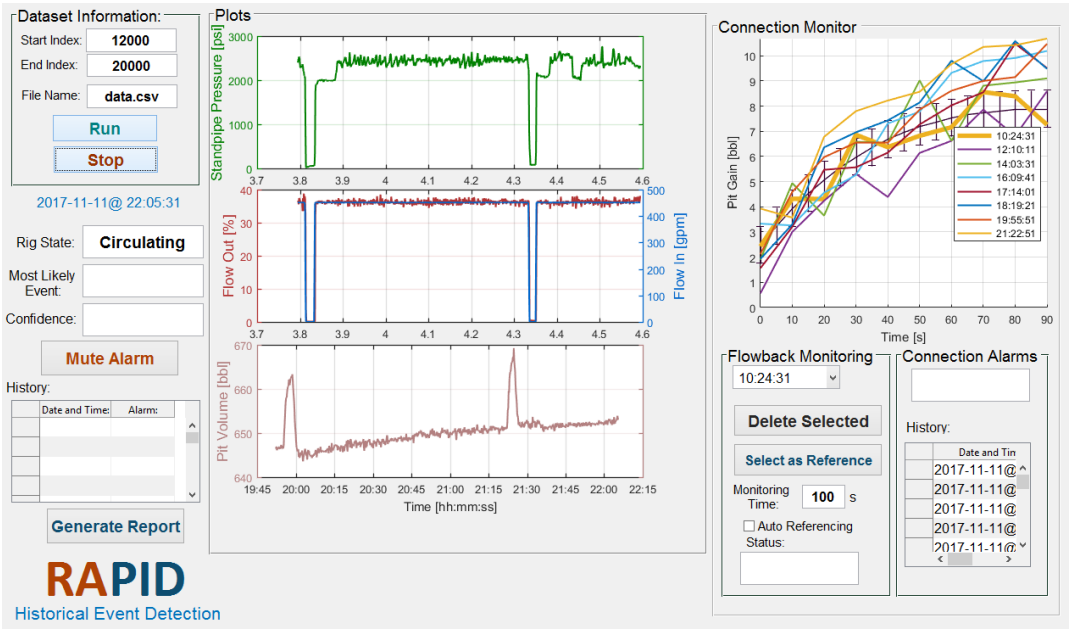

Figure 5.22: Correct identification of wellbore breathing events during connections using the flowback monitoring tool. After 14:03, all flowbacks have a higher steady-state gain than the previous ones.

The RAPID EDS program has been tested on several historical datasets, as well as in real-time in industry RTOCs. The results obtained from the historical testing are summarized in the Table 5.3. The highest rate of false-alarms in testing was observed in Well G when the flow-out sensor was severely noisy at ±30%. The missed alarms from datasets Well E and Well K were due to data quality issues including saturating flow-out sensors that did not exhibit the necessary change required to trigger the FIS system. The

wellbore breathing event in Well H was detected during the connections and corresponds to the example in Figure 5.22.

Dataset	**Number of Days**	**Number of Events**	**Missed Alarms**	**False Alarms**	**Comments**
Well A	12	0	0	0	-
Well B	2	1 (Influx)	0	0	-
Well C	3	1 (Loss)	0	0	-
Well D	4	0	0	1 (Influx)	-
Well E	3	1 (Influx)	1	0	Event not visible in RTC data
Well F	5	1 (Influx)	0	3 (Influx), 1(Loss)	Severely noisy flow-paddle
Well G	25	1 (Gas Influx)	0	0	-
Well H	5	1 (Wellbore Breathing)	0	0	-
Well I	15	1 (Gas Influx)	0	0	-
Well J	56	0	0	1	Rig state missclassification
Well K	1	1 (Gas Influx)	1	0	Saturating flow-paddle

Table 5.3: Testing results summary of the RAPID EDS program on multiple offshore and onshore wells.

The algorithmic details of event detection systems are usually not published in drilling literature, and reproducing results from other similar systems is not possible. However, several authors in recent years have begun to publish

information on the missed/false alarm performance of their systems. Unrau et al. (2017) for instance reported a false alarm rate of 1 per 5 hours for kicks, and 1 per 10 hours for losses using tight alarm settings with their adaptive alarms system. Andia et al. (2018) reported a false alarm rate of 1.53 per 12 hours using a model-based early kick detection system and flow-paddle measurements. The results shown in Table 5.3 demonstrate the strong performance of the EDS system and its ability to avoid false alarm generation. In the worst example of Table 5.3 which corresponds to Well F, EDS produces 0.4 false alarms per 12 hours. In the best example of Table 5.3 which corresponds to Well G, EDS produced 0 false alarms over a 25 day period while detecting the true gas influx event.

5.9 Conclusion

In this chapter, we developed EDS, a set of tools that together form the *Event Detector*. EDS is effective at detecting drilling trouble events with a low rate of missed/false alarms. In addition, reduction of false alarms was shown to be improved by sensor calibration, through the enabling of the delta-flow method. The proposed *Event Detector* was tested on several data-sets from field operations to demonstrate its feasibility for practical implementation.

Chapter 6

Conclusion

This dissertation explores the problem of real-time learning for a drilling automation system, and ways in which the automation system can leverage this learning to enhance optimization, control and event detection performance. As a solution to this problem, this dissertation described a self-learning control methodology that encourages the system to actively learn about its environment by taking appropriate actions. The various subsystems of this control system were developed and tested, and the importance of each subsystem to the control system as a whole was highlighted.

Initially, it was shown that learning the parameters of a drilling physics-based model requires suitable learning algorithms and the right transition function structure to address the complexities of the drilling environment. In addition, effective learning was shown to require a calculated sequential approach, in which the agent takes the appropriate actions in the right scenarios for optimal learning.

Optimization was explored in Chapter 3, and it was shown that by strategically striking the right balance between exploration and exploitation, the *Optimizer* can perform iterative learning and optimization to find optimal

drilling parameters. Through the use of the model learned in Chapter 2, *Optimizer* was also shown to be capable of predicting the environment's response to control actions and ensure constraint satisfaction.

The closed-loop tracking of *Optimizer* set-points was targeted in Chapter 4, and a nonlinear control methodology based on the sliding mode technique was shown to handle the challenges of drilling process control better than linear control techniques. Benchmarking was performed on a high-fidelity dynamics model using various test cases.

The importance of the *Event Detector* and its ability to robustly avoid missed/false alarms in drilling event detection was explored in Chapter 5. An event detection system capable of leveraging real-time learning was developed, and implemented for field data from drilling operations.

6.1 Contributions

Each individual chapter in this dissertation made the following contributions to the existing drilling engineering literature:

- Chapter 2 presented an action-driven, sequential learning approach for learning of physics-based drilling models. This methodology was applied to a novel real-time physics-based model of the drilling environment, which in accordance with the scope, focused on a coupled axial dynamics and drilling fluid hydraulics model.

- Chapter 3 introduced a drilling optimization technique (named qDrill)

that does not rely on complete *a priori* knowledge of the environment and adapts itself to environment changes. Chapter 3 also formulated the first predictive optimization procedure for automated tripping.

- Chapter 4 presented a nonlinear WOB controller to address the challenges of tracking the set-points of the *Optimizer*. This controller was shown to perform better than conventional techniques from a tracking, robustness and disturbance rejection perspective.

- Chapter 5 presented a drilling event detection system (named RAPID EDS) that has proven to produce a very low number of missed/false alarms in comparison to other popular event detection software in the industry.

6.2 Recommendations and Future Work

The work presented in this dissertation is readily applicable to field operations. For effective implementation, the following points should be considered.

The learning algorithms presented in Chapter 2 require complex mathematical operations and high sampling rates, which are not attainable with rig-site programmable logic controllers. Industrial embedded PC systems, however, can address both of the aforementioned requirements, as they allow direct connectivity with rig sensors and can perform the required computations. The nonlinear WOB process level control of Chapter 4 can be implemented directly

on a PLC system, but it will require direct communication with the estimator on the embedded PC. For a standalone controller/estimator layout without the use of an embedded PC, a FPGA implementation can also be explored.

The event detection system of Chapter 5 has been successfully implemented in an RTOC setting with the aid of a WITSML data acquisition application. A similar approach can be taken for rig-site implementation, provided the rig has reliable internet connectivity. However, the WITSML data communication protocol does not provide deterministic data transfer for applications that benefit from timely decisions. For early detection of trouble events in critical applications such as offshore operations, it is recommended that the EDS be implemented on a rig-site embedded PC system to avoid the complications of the WITSML protocol.

The optimization algorithms of Chapter 3 require high computational and memory power, but low communication speeds. They can therefore be implemented on office computers, and communicate with the rig-site control system through various existing protocols such as WITSML. Alternatively, an additional onsite computer can be used for implementation of the optimization algorithm in the case of no internet connectivity.

From an academic standpoint, it is recommended to focus future research on:

- Laboratory experiments to further validate the action-driven learning approach presented in Chapter 2.

- Field implementation of the qDrill algorithm on an open industry application platform to experimentally assess the proposed technique in real drilling scenarios.

- Exploration of a look-up table based PID gain scheduling as an alternative to the proposed nonlinear controller for WOB control. If comparable in terms of performance, rig-site implementation of such a control technique can be potentially less complicated than the rest of the control techniques presented in this dissertation.

- Application of RAPID EDS to other drilling events in a variety of drilling operational states.

Appendices

Appendix A

List of Symbols and Abbreviations

A.1 Symbols

Symbol	Meaning
t	time
z	spatial coordinate along the vertical
$\mathbf{z}$	measurement vector
h	spatial coordinate along the horizontal
x	state variable
s	Laplace variable
v	velocity
P	pressure
Q	flow rate
Ω	angular velocity
F	force
m	mass
$\bar{m}$	mass per unit length
w	weight per unit length
m_f	fluid mass
L	length
b	damping parameter
k	stiffness parameter
$\bar{k}$	stiffness times length

K	bulk modulus of elasticity
PV	plastic viscosity of a Bingham plastic fluid
YP	yield point of a Bingham plastic fluid
V	volume
A	cross-sectional area
C_z	discharge efficiency
Ψ	nonlinear damping function
κ	nonlinear friction force
μ	coefficient of friction
β_B	coefficient of bouyancy
Z_{ds}	characteristic impedance of drillstring
T	characteristic time delay
$\mathbf{P}$	state covariance matrix
Q	process noise covariance matrix
R	measurement noise covariance matrix
H	linear observation matrix
$\mathcal{R}$	reward
$\mathcal{P}$	transition probability
α	learning rate
ϵ	probability of random action
Υ	probability of selecting sub-optimal cluster
β	PSO population parameter
S	sliding surface
η	sliding mode control paramater
c_1	super-twisting sliding mode control paramater
c_2	super-twisting sliding mode control paramater
λ	characteristic equation root

f_{out}	flow-rate out
f_{in}	flow-rate in

A.2 Subscripts

Subscript	**Meaning**
ds	drillstring
c	drill collars
fr	formation
in	input
f	frictional loss
hkl	hook-load
spp	standpipe pressure
n	lumped model element
q	lumped estimator element
md	measured depth
tvd	true vertical depth

A.3 Abbreviations

Abbreviation	**Meaning**
WOB	weight-on-bit
ROP	rate of penetration
RPM	revolutions per minute
CV	control volume
EKF	extended Kalman filter
UKF	unscented Kalman filter
PF	particle filter
EUL	ensemble unscented learner

ODE	ordinary differential equation
DDE	delay differential equation
BD	bit depth
HD	hole depth
SPP	standpipe pressure
MDP	markov decision process
ANN	artificial neural network
SVR	support vector regression
BTR	bagged tree regression
RF	random forest
MSE	mean squared error
PSO	particle swarm optimization
PID	proportional integral derivative control
LQI	linear quadratic integral control
MPC	model predictive control
SMC	sliding mode control
STSMC	super-twisting sliding mode control
EDS	event detection system
SAX	symbolic aggregare approximation
CPD	change point detection
RTC	real-time centre
WITSML	wellsite information transfer standard markup language

Appendix B

Hydraulic Pressure Drop Calculations

For Bingham Plastic fluids, the pressure drop for a pipe of inner diameter, D, in field units can be written as:

$$\Delta p_{f,lam} = \left(\frac{\mu_p v}{1500 D^2} + \frac{\tau_0}{225 D} \right) \Delta L \tag{B.1}$$

for laminar flow, and:

$$\Delta p_{f,turb} = \left(\frac{\rho^{0.75} v^{1.75} \mu_p^{0.25}}{18000 D^{1.25}} \right) \Delta L \tag{B.2}$$

for turbulent flow. For fluid flow inside the annulus, the pressure drop can be written as:

$$\Delta p_{f,lam} = \left(\frac{\mu_p v}{1000 (D_2 - D_1)^2} + \frac{\tau_0}{200 (D_2 - D_1)} \right) \Delta L \tag{B.3}$$

for laminar flow, and:

$$\Delta p_{f,turb} = \left(\frac{\rho^{0.75} v^{1.75} \mu_p^{0.25}}{1396 (D_2 - D_1)^{1.25}} \right) \Delta L \tag{B.4}$$

for turbulent flow. The flow regime can be determined by evaluating the Reynolds number. For inside the drillstring, the Reynolds number can be written as:

$$N_{Re} = 928 \frac{\rho v D}{\mu_a} \tag{B.5}$$

where the apparent viscosity, μ_a, is given by:

$$\mu_a = \mu_p + \frac{6.66\tau_0 D}{v} \tag{B.6}$$

Similarly, the Reynolds number inside the annulus can be evaluated by:

$$N_{Re} = 757\frac{\rho v(D_2 - D_1)}{\mu_a} \tag{B.7}$$

where the apparent viscosity, μ_a, is given by:

$$\mu_a = \mu_p + \frac{5\tau_0(D_2 - D_1)}{v} \tag{B.8}$$

Appendix C

Published Work

1. A. Ambrus, P. Pournazari, P. Ashok, R. Shor, E. van Oort, et al. Overcoming barriers to adoption of drilling automation: Moving towards automated well manufacturing. *In SPE/IADC Drilling Conference and Exhibition*. Society of Petroleum Engineers, 2015.

2. P. Pournazari, P. Ashok, and E. van Oort. Modeling and control of automated pipe hoisting in oil and gas well construction. In *ASME 2015 Dynamic Systems and Control Conference*, pages V002T20A004 V002T20A004. American Society of Mechanical Engineers, 2015a.

3. P. Pournazari, P. Ashok, E. van Oort, S. Unrau, S. Lai, et al. Enhanced kick detection with low-cost rig sensors through automated pattern recognition and real-time sensor calibration. In *SPE Middle East Intelligent Oil and Gas Conference and Exhibition*. Society of Petroleum Engineers, 2015b.

4. P. Pournazari, D. Adams, P. Ashok, E. v. Oort, K. Holliday, et al. Realtime health monitoring of top drives for offshore operations using physics based models and new sensor technology. In *SPE Deepwater*

Drilling and Completions Conference. Society of Petroleum Engineers, 2016.

5. P. Pournazari, B. R. Fernandez, and E. van Oort. Robust weight-on-bit tracking in drilling operations: A stochastic, nonlinear approach. In *ASME 2017 Dynamic Systems and Control Conference*, pages V003T43A003-V003T43A003. American Society of Mechanical Engineers, 2017.

6. E. Cayeux, R. Shor, A. Ambrus, P. Pournazari, P. Ashok, and E. van Oort. From shallow horizontal drilling to erd wells: How scale affects drillability and the management of drilling incidents. *Journal of Petroleum Science and Engineering*, 160:91-105, 2018.

C.1 Planned Publications

1. P. Pournazari, P. Ashok, and E. van Oort. An Action-Drive, Self-Learning Auto-Driller for Robust Weight-on-Bit Tracking and Real-Time Drilling Optimization. *ASME Journal of Dynamic Systems, Measurement, and Control*.

2. P. Pournazari, P. Ashok, and E. van Oort. False and Missed Alarm Minimization in Drilling Event Detection. *SPE Drilling & Completion Journal*.

Bibliography

O. M. Aamo. Disturbance rejection in 2 x 2 linear hyperbolic systems. *Automatic Control, IEEE Transactions on*, 58(5):1095–1106, 2013.

U. J. F. Aarsnes, O. M. Aamo, and A. Pavlov. Quantifying error introduced by finite order discretization of a hydraulic well model. In *Control Conference (AUCC), 2012 2nd Australian*, pages 54–59. IEEE, 2012.

U. J. F. Aarsnes, F. Di Meglio, S. Evje, and O. M. Aamo. Control-oriented drift-flux modeling of single and two-phase flow for drilling. In *ASME 2014 Dynamic Systems and Control Conference*, pages V003T37A003–V003T37A003. American Society of Mechanical Engineers, 2014.

R. P. Adams and D. J. MacKay. Bayesian online changepoint detection. *arXiv preprint arXiv:0710.3742*, 2007.

R. Al Seyab. Differential recurrent neural network based predictive control. *Computers & Chemical Engineering*, 32(7):1533–1545, 2008.

A. Ambrus, P. Pournazari, P. Ashok, R. Shor, E. van Oort, et al. Overcoming barriers to adoption of drilling automation: Moving towards automated well manufacturing. In *SPE/IADC Drilling Conference and Exhibition*. Society of Petroleum Engineers, 2015.

P. Andia, R. Israel, et al. A cyber-physical approach to early kick detection. In *IADC/SPE Drilling Conference and Exhibition*. Society of Petroleum Engineers, 2018.

C. Andrieu and A. Doucet. Online expectation-maximization type algorithms for parameter estimation in general state space models. In *ICASSP (6)*, pages 69–72. Citeseer, 2003.

B. Anfinsen, R. Rommetveit, et al. Sensitivity of early kick detection parameters in full-scale gas kick experiments with oil-and water-based drilling muds. In *SPE/IADC Drilling Conference*. Society of Petroleum Engineers, 1992.

I. Arasaratnam and S. Haykin. Cubature kalman filters. *Automatic Control, IEEE Transactions on*, 54(6):1254–1269, 2009.

A. Arnaout, R. Fruhwirth, M. Winter, B. Esmael, and G. Thonhauser. Model-based hookload monitoring and prediction at drilling rigs using neural networks and forward-selection algorithm. In *EGU General Assembly Conference Abstracts*, volume 14, page 12209, 2012.

J. Bailey. An analytical study of drill-string vibration. 1960.

J. Bang, S. Mjaaland, A. Solstad, P. Hendriks, L. Jensen, et al. Acoustic gas kick detection with wellhead sonar. In *SPE Annual Technical Conference and Exhibition*. Society of Petroleum Engineers, 1994.

C. Bishop. Pattern recognition and machine learning. 2007.

G. Boyadjieff, D. Murray, A. Orr, M. Porche, P. Thompson, et al. Design considerations and field performance of an advanced automatic driller. In *SPE/IADC Drilling Conference*. Society of Petroleum Engineers, 2003.

J. Brakel, B. Tarr, W. Cox, F. Jorgensen, H. V. Straume, et al. Smart kick detection: First step on the well-control automation journey. *SPE Drilling & Completion*, 30(03):233–242, 2015.

L. Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

J. T. Bryson, X. Jin, and S. K. Agrawal. Optimal design of cable-driven manipulators using particle swarm optimization. *Journal of mechanisms and robotics*, 8(4):041003, 2016.

J. Burkhardt et al. Wellbore pressure surges produced by pipe movement. *Journal of petroleum technology*, 13(06):595–605, 1961.

H. Caicedo, W. Calhoun, and R. Ewy. Unique rop predictor using bit-specific coefficient of sliding friction and mechanical efficiency as a function of confined compressive strength impacts drilling performance. In *SPE/IADC Drilling Conference*, 2005.

E. Cayeux, B. Daireaux, et al. Early detection of drilling conditions deterioration using real-time calibration of computer models: field example from north sea drilling operations. In *SPE/IADC Drilling Conference and Exhibition*. Society of Petroleum Engineers, 2009.

E. Cayeux, R. Shor, A. Ambrus, P. Pournazari, P. Ashok, and E. van Oort. From shallow horizontal drilling to erd wells: How scale affects drillability and the management of drilling incidents. *Journal of Petroleum Science and Engineering*, 160:91–105, 2018.

C. D. Chapman, J. L. Sanchez, R. De Leon Perez, H. Yu, et al. Automated teller-loop drilling with rop optimization algorithm significantly reduces drilling time and improves downhole tool reliability. In *IADC/SPE Drilling Conference and Exhibition*. Society of Petroleum Engineers, 2012.

Z. Chen et al. Bayesian filtering: From kalman filters to particle filters, and beyond. *Statistics*, 182(1):1–69, 2003.

W. C. Chin, X. Zhuang, et al. Advances in swab-surge modeling for managed pressure drilling. In *Offshore Technology Conference*. Offshore Technology Conference, 2011.

M. Costa, E. Pasero, F. Piglione, and D. Radasanu. Short term load forecasting using a synchronously operated recurrent neural network. In *Neural Networks, 1999. IJCNN'99. International Joint Conference on*, volume 5, pages 3478–3482. IEEE, 1999.

F. E. Crespo, R. M. Ahmed, A. Saasen, M. Enfis, M. Amani, et al. Surge-and-swab pressure predictions for yield-power-law drilling fluids. *SPE Drilling & Completion*, 27(04):574–585, 2012.

B. Daireaux, E. Cayeux, et al. Precise gain and loss detection using a transient hydraulic model of the return flow to the pit. In *SPE/IADC Middle East Drilling Technology Conference & Exhibition*. Society of Petroleum Engineers, 2013.

D. Dashevskiy, V. Dubinsky, J. Macpherson, et al. Application of neural networks for predictive control in drilling dynamics. In *SPE Annual Technical Conference and Exhibition*. Society of Petroleum Engineers, 1999.

D. Dashevskiy, J. Macpherson, V. Dubinsky, and P. McGinley. Real-time drilling optimization based on mwd dynamic measurements?field test results. In *AADE 2003 National Technical Conference ?Practical Solutions for Drilling Challenges?, April*, pages 1–3, 2003.

F. Daum and J. Huang. Curse of dimensionality and particle filters. In *Aerospace Conference, 2003. Proceedings. 2003 IEEE*, volume 4, pages 4_1979–4_1993. IEEE, 2003.

V. Dunayevsky, A. Judzis, W. Mills, et al. Onset of drillstring precession in a directional borehole. In *SPE Annual Technical Conference and Exhibition*. Society of Petroleum Engineers, 1984.

V. Dunayevsky, F. Abbassian, et al. Application of stability approach to bit dynamics. *SPE Drilling & Completion*, 13(02):99–107, 1998.

J. Dunlop, R. Isangulov, W. D. Aldred, H. A. Sanchez, J. L. S. Flores, J. A. Herdoiza, J. Belaskie, C. Luppens, et al. Increased rate of penetration

through automation. In *SPE/IADC Drilling Conference and Exhibition*. Society of Petroleum Engineers, 2011.

G. Evensen. The ensemble kalman filter: Theoretical formulation and practical implementation. *Ocean dynamics*, 53(4):343–367, 2003.

B. Fernández, A. V. Prabhudesai, V. V. Murty, R. Gupta, and W. Chang. Neurobondgraphs: Modeling environment of nonlinear dynamic systems using neural networks and bond graphs. 1993.

B. R. Fernández, C. Gupta, and J. Krol. extreme control: The emergent behavior of robust-optimal adaptation synergy. 2001.

T. C. Fonseca, J. P. Mendes, A. Serapião, and I. R. Guilherme. A genetic neuro-model reference adaptive controller for petroleum wells drilling operations. In *Computational Intelligence for Modelling, Control and Automation, 2006 and International Conference on Intelligent Agents, Web Technologies and Internet Commerce, International Conference on*, pages 3–3. IEEE, 2006.

Z. Ghahramani and S. T. Roweis. Learning nonlinear dynamical systems using an em algorithm. *Advances in neural information processing systems*, pages 431–437, 1999.

Y. K. Gidh, A. Purwanto, H. Ibrahim, et al. Artificial neural network drilling parameter optimization system improves rop by predicting/managing bit

wear. In *SPE Intelligent Energy International*. Society of Petroleum Engineers, 2012.

C. L. Giles, S. Lawrence, and A. Tsoi. Rule inference for financial prediction using recurrent neural networks. In *Computational Intelligence for Financial Engineering (CIFEr), 1997., Proceedings of the IEEE/IAFE 1997*, pages 253–259. IEEE, 1997.

K. Gjerstad, D. Sui, K. S. Bjørkevoll, and R. W. Time. Automatic prediction of downhole pressure surges in tripping operations. In *IPTC 2013: International Petroleum Technology Conference*, 2013.

J. M. Gottman. *Time-series analysis: a comprehensive introduction for social scientists*. Number 519.55 G6. 1981.

J. E. Gravdal, R. J. Lorentzen, K. K. Fjelde, E. H. Vefring, et al. Tuning of computer model parameters in managed pressure drilling applications using an unscented kalman filter technique. In *SPE Annual Technical Conference and Exhibition*. Society of Petroleum Engineers, 2005.

T. O. Gulsrud, R. Nybø, K. S. Bjørkevoll, et al. Statistical method for detection of poor hole cleaning and stuck pipe. In *Offshore Europe*. Society of Petroleum Engineers, 2009.

D. Hargreaves, S. Jardine, B. Jeffryes, et al. Early kick detection for deepwater drilling: New probabilistic methods applied in the field. In *SPE Annual Technical Conference and Exhibition*. Society of Petroleum Engineers, 2001.

M. W. Harris, B. Açıkmeşe, and E. van Oort. Lmi based control of stick-slip oscillations in drilling. In *ASME 2014 Dynamic Systems and Control Conference*, pages V001T09A004–V001T09A004. American Society of Mechanical Engineers, 2014.

T. Hastie and R. Tibshirani. *Generalized additive models*. Wiley Online Library, 1990.

S. S. Haykin, S. S. Haykin, S. S. Haykin, and S. S. Haykin. *Neural networks and learning machines*, volume 3. Pearson Education Upper Saddle River, 2009.

G. Heisig, M. Neubert, et al. Lateral drillstring vibrations in extended-reach wells. In *IADC/SPE drilling conference*. Society of Petroleum Engineers, 2000.

S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

N. Hovakimyan, F. Nardi, A. Calise, and N. Kim. Adaptive output feedback control of uncertain nonlinear systems using single-hidden-layer neural networks. *Neural Networks, IEEE Transactions on*, 13(6):1420–1431, 2002.

Y. Hu, Q. Di, W. Zhu, Z. Chen, and W. Wang. Dynamic characteristics analysis of drillstring in the ultra-deep well with spatial curved beam finite element. *Journal of Petroleum Science and Engineering*, 82:166–173, 2012.

T.-M. Huang, V. Kecman, and I. Kopriva. *Kernel based algorithms for mining huge data sets*, volume 1. Springer, 2006.

R. Irani and R. Nasimi. Application of artificial bee colony-based neural network in bottom hole pressure prediction in underbalanced drilling. *Journal of Petroleum Science and Engineering*, 78(1):6–12, 2011.

A. Isidori. *Nonlinear control systems*. Springer Science & Business Media, 2013.

M. S. Johannes and N. Polson. Particle filtering and parameter learning. *Available at SSRN 983646*, 2007.

J. Jorden, O. Shirley, et al. Application of drilling performance data to overpressure detection. *Journal of Petroleum Technology*, 18(11):1–387, 1966.

T. M. Josserand. *Optimally-robust nonlinear control of a class of robotic underwater vehicles*. ProQuest, 2006.

G.-O. Kaasa, Ø. N. Stamnes, O. M. Aamo, L. S. Imsland, et al. Simplified hydraulics model used for intelligent estimation of downhole pressure for a managed-pressure-drilling control system. *SPE Drilling & Completion*, 27 (01):127–138, 2012.

L. Kuncheva. *Fuzzy classifier design*, volume 49. Springer Science & Business Media, 2000.

A. C. Lage, K. K. Fjelde, R. W. Time, et al. Underbalanced drilling dynamics: Two-phase flow modeling and experiments. In *IADC/SPE Asia Pacific Drilling Technology*. Society of Petroleum Engineers, 2000.

I. S. Landet. *Modeling and control for managed pressure drilling from floaters*. PhD thesis, diploma thesis, NTNU, 2011.

L. Lang, W.-s. Chen, B. R. Bakshi, P. K. Goel, and S. Ungarala. Bayesian estimation via sequential monte carlo sampling?constrained dynamic systems. *Automatica*, 43(9):1615–1622, 2007.

F. Le Blay, E. Villard, S. C. Hilliard, T. Gronas, et al. A new generation of well surveillance for early detection of gains and losses when drilling very high profile ultradeepwater wells, improving safety, and optimizing operating procedures. In *SPETT 2012 Energy Conference and Exhibition*. Society of Petroleum Engineers, 2012.

J. Lin, E. Keogh, L. Wei, and S. Lonardi. Experiencing sax: a novel symbolic representation of time series. *Data Mining and knowledge discovery*, 15(2): 107–144, 2007.

Z. Ma and D. Chen. Modeling of coupled axial and torsional motions of a drilling system. In *ASME 2015 Dynamic Systems and Control Conference*, pages V002T20A005–V002T20A005. American Society of Mechanical Engineers, 2015.

A. Marzal and E. Vidal. Computation of normalized edit distance and applications. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 15(9):926–932, 1993.

L. Maus, J. Tannich, W. Ilfrey, et al. Instrumentation requirements for kick detection in deep water. *Journal of Petroleum Technology*, 31(08):1–029, 1979.

D. McCann, D. White, L. Marais, G. Rodt, et al. Improved rig safety by rapid and automated kick detection. In *SPE/IADC Drilling Conference*. Society of Petroleum Engineers, 1991.

R. Miri, J. H. Sampaio, M. Afshar, A. Lourenco, et al. Development of artificial neural networks to predict differential pipe sticking in iranian offshore oil fields. In *International Oil Conference and Exhibition in Mexico*. Society of Petroleum Engineers, 2007.

R. Mitchell et al. Dynamic surge/swab pressure predictions. *SPE drilling engineering*, 3(03):325–333, 1988.

A. W. Moore and C. G. Atkeson. Prioritized sweeping: Reinforcement learning with less data and less time. *Machine Learning*, 13(1):103–130, 1993.

H. Moradkhani, S. Sorooshian, H. V. Gupta, and P. R. Houser. Dual state–parameter estimation of hydrological models using ensemble kalman filter. *Advances in Water Resources*, 28(2):135–147, 2005.

K. R. Muske and T. F. Edgar. Nonlinear state estimation. In *Nonlinear process control*, pages 311–370. Prentice-Hall, Inc., 1997.

M. Nakhaeinejad. Fault detection and model-based diagnostics in nonlinear dynamic systems. 2010.

E. M. Navarro-López and D. Cortés. Avoiding harmful oscillations in a drill-string through dynamical analysis. *Journal of sound and vibration*, 307(1): 152–171, 2007.

A. S. Nejad and K. Shahbazi. A new approach for estimating free point in fishing of stuck pipe using artificial neural network. *International Journal of Computer Applications*, 82(6), 2013.

A. Nikoofard, U. J. F. Aarsnes, T. A. Johansen, and G.-O. Kaasa. Estimation of states and parameters of a drift-flux model with unscented kalman filter. *IFAC-PapersOnLine*, 48(6):165–170, 2015.

R. Nybø. Efficient drilling problem detection: Early fault detection by the combination of physical models and artificial intelligence. 2009.

Y. Pan and J. Wang. Model predictive control of unknown nonlinear dynamical systems based on recurrent neural networks. *Industrial Electronics, IEEE Transactions on*, 59(8):3089–3101, 2012.

R. Pascanu, T. Mikolov, and Y. Bengio. On the difficulty of training recurrent neural networks. *arXiv preprint arXiv:1211.5063*, 2012.

P. Pastusek, M. Owens, D. Barrette, V. Wilkins, A. Bolzan, J. Ryan, K. Akyabi, M. Reichle, D. Pais, et al. Drill rig control systems: Debugging, tuning, and long term needs. In *SPE Annual Technical Conference and Exhibition*. Society of Petroleum Engineers, 2016.

G. S. Payette, D. Pais, B. Spivey, L. Wang, J. R. Bailey, P. Pastusek, M. Owens, et al. Mitigating drilling dysfunction using a drilling advisory system: Results from recent field applications. In *International Petroleum Technology Conference*. International Petroleum Technology Conference, 2015.

T. P. Pink, W. L. Koederitz, A. Barrie, D. R. Bert, D. C. Overgaard, et al. Closed loop automation of downhole weight on bit improves sliding performance and reduces conservatism in unconventional horizontal well development. In *SPE Annual Technical Conference and Exhibition*. Society of Petroleum Engineers, 2013.

D. S. Pixton, R. Asgharzadeh Shishavan, H. D. Perez, J. D. Hedengren, A. Craig, et al. Addressing ubo and mpd challenges with wired drill pipe telemetry. In *SPE/IADC Managed Pressure Drilling & Underbalanced Operations Conference & Exhibition*. Society of Petroleum Engineers, 2014.

P. Pournazari, P. Ashok, and E. van Oort. Modeling and control of automated pipe hoisting in oil and gas well construction. In *ASME 2015 Dynamic Systems and Control Conference*, pages V002T20A004–V002T20A004. American Society of Mechanical Engineers, 2015a.

P. Pournazari, P. Ashok, E. van Oort, S. Unrau, S. Lai, et al. Enhanced kick detection with low-cost rig sensors through automated pattern recognition and real-time sensor calibration. In *SPE Middle East Intelligent Oil and Gas Conference and Exhibition*. Society of Petroleum Engineers, 2015b.

P. Pournazari, D. Adams, P. Ashok, E. v. Oort, K. Holliday, et al. Real-time health monitoring of top drives for offshore operations using physics based models and new sensor technology. In *SPE Deepwater Drilling and Completions Conference*. Society of Petroleum Engineers, 2016.

P. Pournazari, B. R. Fernández, and E. van Oort. Robust weight-on-bit tracking in drilling operations: A stochastic, nonlinear approach. In *ASME 2017 Dynamic Systems and Control Conference*, pages V003T43A003–V003T43A003. American Society of Mechanical Engineers, 2017.

J. Prakash, S. C. Patwardhan, and S. L. Shah. Constrained nonlinear state estimation using ensemble kalman filters. *Industrial & Engineering Chemistry Research*, 49(5):2242–2253, 2010.

J. Prakash, R. Gopaluni, S. C. Patwardhan, S. Narasimhan, and S. L. Shah. Nonlinear bayesian state estimation: Review and recent trends. In *Advanced Control of Industrial Processes (ADCONIP), 2011 International Symposium on*, pages 450–455. IEEE, 2011.

G. V. Puskorius and L. A. Feldkamp. Parameter-based kalman filter training: theory and implementation. *Kalman filtering and neural networks*, page 23, 2001.

G. V. Puskorius, L. A. Feldkamp, and L. I. Davis Jr. Dynamic neural network methods applied to on-vehicle idle speed control. *Proceedings of the IEEE*, 84(10):1407–1420, 1996.

D. Reitsma et al. Development of an automated system for the rapid detection of drilling anomalies using standpipe and discharge pressure. In *SPE/IADC Drilling Conference and Exhibition*. Society of Petroleum Engineers, 2011.

J. Rivera, J. J. Raygoza, C. Mora, L. Garcia, and S. Ortega. *Super-twisting sliding mode in motion control systems*. INTECH Open Access Publisher, 2011.

R. Rooki, F. D. Ardejani, and A. Moradzadeh. Hole cleaning prediction in foam drilling using artificial neural network and multiple linear regression. *Geomaterials*, 2014, 2014.

D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning representations by back-propagating errors. *Cognitive modeling*, 5(3):1, 1988.

G. R. Samuel, A. Sunthankar, G. McColpin, P. Bern, T. Flynn, et al. Field validation of transient swab-surge response with real-time downhole pressure data. *SPE drilling & completion*, 18(04):280–283, 2003.

H. Santos et al. Differentially stuck pipe: early diagnostic and solution. In *IADC/SPE Drilling Conference*. Society of Petroleum Engineers, 2000.

D. Schafer, G. Loeppke, D. Glowka, D. Scott, E. Wright, et al. An evaluation of flowmeters for the detection of kicks and lost circulation during drilling. In *SPE/IADC Drilling Conference*. Society of Petroleum Engineers, 1992.

R. J. Shor, M. Pryor, and E. Van Oort. Drillstring vibration observation, modeling and prevention in the oil and gas industry. In *ASME 2014 Dynamic Systems and Control Conference*, pages V003T37A004–V003T37A004. American Society of Mechanical Engineers, 2014.

Y. B. Shtessel, J. A. Moreno, F. Plestan, L. M. Fridman, and A. S. Poznyak. Super-twisting adaptive sliding mode control: A lyapunov design. In *Decision and Control (CDC), 2010 49th IEEE Conference on*, pages 5109–5113. IEEE, 2010.

E. Skaugen et al. The effects of quasi-random drill bit vibrations upon drillstring dynamic behavior. In *SPE Annual Technical Conference and Exhibition*. Society of Petroleum Engineers, 1987.

J.-J. E. Slotine, W. Li, et al. *Applied nonlinear control*, volume 199. prentice-Hall Englewood Cliffs, NJ, 1991.

A. J. Smola and B. Schölkopf. A tutorial on support vector regression. *Statistics and computing*, 14(3):199–222, 2004.

Ø. N. Stamnes, O. M. Aamo, and G.-O. Kaasa. Redesign of adaptive observers for improved parameter identification in nonlinear systems. *Automatica*, 47 (2):403–410, 2011.

R. S. Sutton and A. G. Barto. *Reinforcement learning: An introduction*, volume 1. MIT press Cambridge, 1998.

P. P. Trivedi et al. Innovative kick detection system for hp/ht ultradeepwater wells using a section of the bha. In *Offshore Technology Conference*. Offshore Technology Conference, 2014.

S. Unrau, P. Torrione, M. Hibbard, R. Smith, L. Olesen, J. Watson, et al. Machine learning algorithms applied to detection of well control events. In *SPE Kingdom of Saudi Arabia Annual Technical Symposium and Exhibition*. Society of Petroleum Engineers, 2017.

P. Vachhani, R. Rengaswamy, V. Gangwal, and S. Narasimhan. Recursive estimation in constrained nonlinear dynamical systems. *AIChE Journal*, 51 (3):946–959, 2005.

P. Vachhani, S. Narasimhan, and R. Rengaswamy. Robust and reliable estimation via unscented recursive nonlinear dynamic data reconciliation. *Journal of process control*, 16(10):1075–1086, 2006.

R. Van Der Merwe, A. Doucet, N. De Freitas, and E. A. Wan. The unscented particle filter. In *Advances in neural information processing systems*, pages 584–590, 2001.

H. Van Seijen and R. S. Sutton. Efficient planning in mdps by small backups. 2013.

E. A. Wan and R. Van Der Merwe. The unscented kalman filter for nonlinear estimation. In *Adaptive Systems for Signal Processing, Communications, and Control Symposium 2000. AS-SPCC. The IEEE 2000*, pages 153–158. Ieee, 2000.

E. A. Wan, R. Van Der Merwe, and A. T. Nelson. Dual estimation and the unscented transformation. In *NIPS*, pages 666–672, 1999.

L. Wang. *Model predictive control system design and implementation using MATLAB®*. Springer Science & Business Media, 2009.

R. J. Williams and D. Zipser. A learning algorithm for continually running fully recurrent neural networks. *Neural computation*, 1(2):270–280, 1989.

A. S. Yigit and A. P. Christoforou. Stick-slip and bit-bounce interaction in oil-well drillstrings. *Journal of Energy Resources Technology*, 128(4):268–274, 2006.

M. Yilmaz, S. Mujeeb, and N. R. Dhansri. A h-infinity control approach for oil drilling processes. *Procedia Computer Science*, 20:134–139, 2013.

V. M. Zavala, C. D. Laird, and L. T. Biegler. A fast moving horizon estimation algorithm based on nonlinear programming sensitivity. *Journal of Process Control*, 18(9):876–884, 2008.

K. Zhou, J. C. Doyle, K. Glover, et al. *Robust and optimal control*, volume 40. Prentice hall New Jersey, 1996.

Vita

Parham Pournazari attended the West Vancouver Secondary School and graduated with an International Baccalaureate diploma. He then matriculated at the University of British Columbia, and received a degree in Mechanical Engineering with the Mechatronics option in May 2013. During his time as an undergraduate student, he held part-time and co-op positions at university research labs, as well as process control and manufacturing companies. Subsequently, he began his graduate studies at the University of Texas at Austin in the Mechanical Engineering department, with a specialization in Dynamic Systems and Control.

Permanent address: parhamp@utexas.edu

This dissertation was typeset with LaTeX[†] by the author.

[†]LaTeX is a document preparation system developed by Leslie Lamport as a special version of Donald Knuth's TeX Program.