Master's thesis

NTNU
Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Department of Engineering Cybernetics

Anders Kommedal

Development of Model Predictive Control for Setpoint Regulation in Offshore Drilling

Master's thesis in Cybernetics and Robotics
Supervisor: Professor Morten Hovd
Co-supervisor: Principal System Integration Advisor Karl Kristian Olsen (Halliburton), Technical Manager Arnfinn Grøtte (AkerBP)

June 2021

NTNU
Kunnskap for en bedre verden

Anders Kommedal

Development of Model Predictive Control for Setpoint Regulation in Offshore Drilling

Master's thesis in Cybernetics and Robotics
Supervisor: Professor Morten Hovd
Co-supervisor: Principal System Integration Advisor Karl Kristian Olsen (Halliburton), Technical Manager Arnfinn Grøtte (AkerBP)
June 2021

Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Department of Engineering Cybernetics

NTNU
Norwegian University of
Science and Technology

Abstract

Today's oil and gas companies are experiencing a highly uncertain and volatile market situation. With large fluctuations in the oil price, it is crucial to keep the operational costs at moderate levels to maintain profitability. Costs related to drilling make up a significant part of the overall expenditures; thus, improving drilling efficiency is essential to minimize costs. However, since most wells today are drilled manually by a driller, the efficiency is subjected to human factors, such as the drillers' experience, strategy, and relation to risks. A research project called *Performinator*, led by AkerBP, aims to autonomously control the drilling process using fully electric robots, thus removing the human factors and increasing the drilling operation's consistency, efficiency, and safety.

This thesis presents one possible solution to the control problem, which can potentially be used as a basis for further development. Results from simulations conducted in Matlab is presented, where a successive linearization-based model predictive control (SLMPC) approach is used to autonomously control the rate of penetration (ROP) to the desired value by adjusting the weight on bit (WOB) and top drive rotary speed (RPM) while subjected to operational and safety constraints imposed by the operation mode. This is done by linearizing a nonlinear Bourgoyne and Young ROP-model successively at each time step to formulate a convex QP problem that an MPC solves. In addition, a verification strategy was developed to simulate a real well response resulting from the calculated control inputs from the SLMPC. The results show that when updating the SLMPC model regularly with measurements from this simulated well, both through a trust-region parameter estimation technique and an extended Kalman filter, the simulated well ROP can efficiently be controlled to the desired value within the limits of the imposed constraints.

Sammendrag

Dagens olje- og gasselskaper opplever for tiden en svært usikker og volatil markedssituasjon. Med store svingninger i oljeprisen er det avgjørende å holde driftskostnadene på moderate nivåer for å opprettholde lønnsomheten. Kostnader knyttet til boring utgjør en betydelig andel av de samlede utgiftene og det er dermed essentielt å forbedre boreprosessen for å redusere kostnadene. Siden de fleste brønner i dag bores manuelt av en borer, påvirkes effektiviteten av menneskelige faktorer som borerens erfaring, strategi og forhold til risiko. Et forskningsprosjekt kalt Performinator, som ledes av AkerBP, har som mål å kontrollere boreprosessen autonomt ved hjelp av heleletriske roboter, og dermed fjerne de menneskelige faktorene med intensjon om å øke boreprosessens pålitlighet, effektivitet og sikkerhet.

Denne oppgaven presenterer én mulig løsning på kontrollproblemet, som potensielt kan brukes som grunnlag for videre utvikling. Resultater fra simuleringer utført i Matlab presenteres, hvor en suksessiv lineariseringsbasert modell prediktiv kontroll (SLMPC) brukes til å autonomt kontrollere gjennomtrengningshastigheten (ROP) til ønsket verdi ved å justere vekten på borekronen (WOB) og tårnboremaskinens rotasjonshastighet (RPM) mens drifts- og sikkerhetsbeskrankninger pålagt av en gitt operasjonsmodus opprettholdes. Dette gjøres ved at en ulineær Bourgoyne og Young ROP-modell lineariseres for hvert tidssteg slik at et konvekst QP-problem kan formuleres og løses av en MPC. I tillegg er det blitt utviklet en verifikasjonsstrategi for å simulere en brønnrespons som følge av de beregnede kontrollinngangene fra SLMPCen. Resultatene viser at dersom SLMPC-modellen jevnlig oppdateres av målinger fra den simulerte brønnen, både gjennom parameteres-timering basert på tillitsregions-metoden og via et utvidet Kalman-filter, kan gjennomtrengning-shastigheten til den simulerte brønnen styres til ønsket verdi innenfor beskrankningenes rekkevidde.

Preface

This master's thesis concludes the course *TTK4900 - Engineering Cybernetics, Master's Thesis* and thus the degree Master of Science in Engineering at Norwegian University of Science and Technology in Trondheim, under the Department of Engineering Cybernetics. The work was conducted in the spring semester of 2021 and is a continuation of a prior specialization project carried out in the fall semester of 2020 in the course *TTK4551 - Engineering Cybernetics, Specialization Project*. Further, the thesis was conducted in collaboration with AkerBP, which provided the necessary drilling data that forms the basis for the work and information and knowledge sharing regarding the drilling process and the petroleum industry. The program developed in this thesis was implemented in the Matlab environment.

I want to thank my supervisor Morten Hovd for the conversations and the feedback provided over the last year. Your many advice has been helpful for the progress of the thesis and is highly appreciated.

Further, I would like to thank my co-supervisor, Karl Kristian Olsen, for the genuine interest you have shown in my work and the follow-ups during the semester. I am very grateful for your continuous feedback and for sharing your technical knowledge and putting me in contact with the right people across various domains in AkerBP when I had questions. Also, I want to thank my co-supervisor, Arnfinn Grøtte, for both proposing and allowing me to write this thesis for AkerBP.

Lastly, I want to show my gratitude to Senior Drilling Engineer at AkerBP, Petter Kvandal. Your knowledge regarding the drilling operation and various domains within the petroleum industry has been invaluable. Thanks for all the time you have spent sharing this knowledge with me.

Anders Kommedal
Trondheim, June 2021

Table of Contents

List of Tables

List of Figures

Abbreviations

B&Y Bourgoyne and Young

BHA Bottom Hole Assembly

BHP Bottom Hole Pressure

BOP Blowout Preventer

CV Controlled Variable

ECD Equivalent Circulation Density

EKF Extended Kalman Filter

LEL Lower Explosive Level

LWD Logging While Drilling

MD Measured Depth

MPC Model Predictive Control

MPD Managed Pressure Drilling

MR Multiple Regression

MV Manipulated Variable

MW Mud Weight

MWD Measurements While Drilling

NPT Nonproductive Time

PDC Polycrystalline Diamond Compact

RCD Rotating Control Device

ROP Rate Of Penetration

SLMPC Successive Linearization based MPC

TR Trust-Region

TVD True Vertical Depth

UBD Underbalanced Drilling

1 Introduction

This section covers the motivation and a description of a project led by AkerBP called *Performinator*. Both these subsections are reproduced and elaborated from the specialization project (Kommedal [2020]). The new problem description is given in subsection 1.3, before the scope and report outline is presented in subsection 1.4.

1.1 Motivation

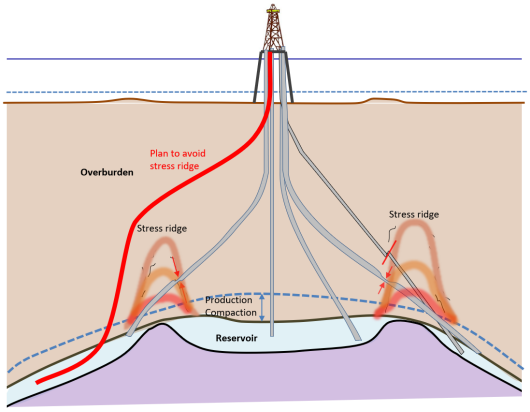
The oil and gas industry is facing significant challenges across the world. The COVID-19 pandemic and the resulting falling oil prices in the spring of 2020 due to reduced energy demand emphasize the market's volatility. Besides, many of the easy-to-get oil and gas fields are depleted, requiring drilling in more complex areas to obtain the same volumes as before, ultimately increasing the projects' break-even price. Furthermore, costs related to drilling make up a significant part of the overall expenditures for an exploration and production company. Over the years 2001-2016, the drilling costs made up on average 44% of the total investments towards oil and gas extraction[1] on the Norwegian continental shelf (Søbye [2017]). Hence, improving the drilling efficiency is of great importance to reduce overall costs, thus ensuring profitability.

Further, most wells today are drilled manually by a driller. This can cause varying drilling performance from day to day, depending on the driller's experience, strategy, and relation to risk. Thus, automating the drilling process and moving the driller to a supervisory role may provide consistency in drilling performance and increased personnel safety.

A digitalization project which aims to do so is the Valhall IP Performinator project (see subsection 1.2), led by AkerBP. They want to investigate the possibility of using an advanced regulation principle in conjunction with this project, namely Model Predictive Control (MPC), to autonomously control the drill string in different operating modes.

Valhall is a mature oilfield located in the southern part of the Norwegian sector of the North Sea, and is seen in figure 1.1 (a). It has been producing since 1982 and is expected to be producing oil and gas for many years to come. However, depletion of the reservoir has induced subsidence, which is a geological phenomenon where the ground's surface gradually sinks to a lower level (Reddish and Whittaker [1989]), leading to stress accumulation and resulting stress ridges shown in Figure 1.1 (b). Weaker formations and limited drilling windows (see subsection 2.1.4) increases the complexity of the drilling operations in the area, further emphasizing the need for new thinking and improved control strategies in the drilling operations.

(a) Valhall IP is platform number two from right (Source: AkerBP ASA)



(b) Well planning in order to avoid stress ridges (Illustration courtesy: Petter Kvandal, AkerBP)

Figure 1.1: Valhall Oilfield

[1] Costs related to decommissioning and piping not included

1.2 Performinator Project

Performinator is a research project led by AkerBP that aims to automate the drilling process using robotic automation of pipe handling for increased efficiency and safety on the drill floor. By using five fully electric robots, shown in figure 1.2, instead of the conventional and manually operated hydraulic pipe handling system, one can achieve more efficient operations with higher precision. Other benefits will be the improved personnel safety as the drillers office can be placed at a remote location, and that fewer drill floor personnel will be needed.

The whole system will integrate an open-source control system utilizing edge computing, simulators, and digital twins. Performinator yields autonomous operations like tripping drill pipe, connection of drill pipes, and optimization of the general drilling procedure.

The master thesis will be looking at the part concerning the drilling procedure. Specifically, it will consider what AkerBP has called an *Autodriller*, which is meant to provide the driller the ability to automatically control the drilling operations. However, this thesis assumes that the Autodriller interaction framework is in place, thus it only concerns the actual control problem.

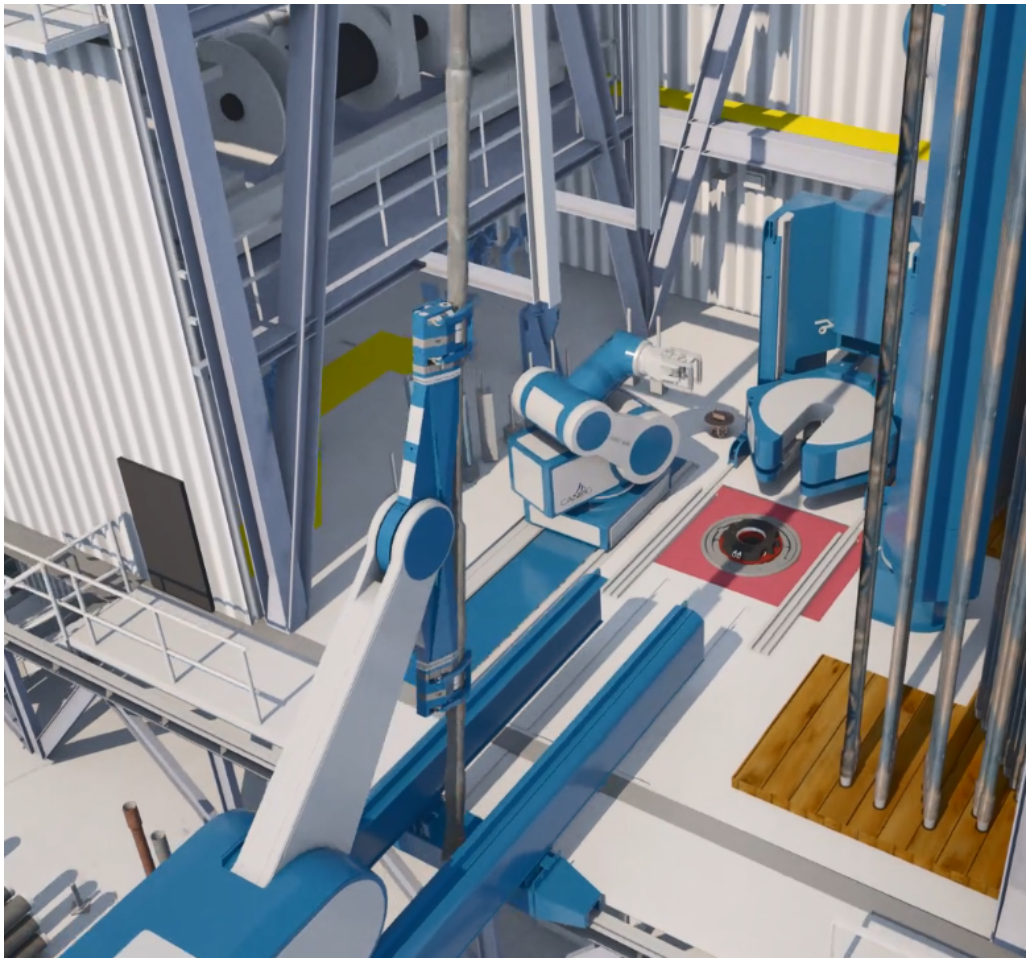


Figure 1.2: Illustration of robot setup on Valhall IP. (Courtesy: Arnfinn Grøtte, AkerBP)

1.3 Problem Description

The master's thesis is given in collaboration with AkerBP and is a continuation of a prior project thesis. The task is to develop an MPC regulator for the drilling process and to test the MPC on data from an actual drilled well. The drilled well data is made available by AkerBP. The task objectives are

- Establish a model of the drilling process based on established sources, such as the Bourgoyne and Young ROP-model.
- Formulate one or more MPC-problems based on different operating modes for the drilling process
- Develop a methodology for updating the MPC-model from online measurements
- Test and simulate the MPC regulator with measurements from the available well data

The final goal of the MPC is to be able to efficiently control the drilling process according to a specified operating mode without compromising safety, thus placing the driller in a supervisory role.

1.4 Scope and Report Outline

The following delimitations were made to narrow the scope of the task and to best meet the task objectives:

1. Only one MPC problem/formulation has been developed for one operating mode for the drilling process. Specifically, Rate of Penetration mode was developed as it is one of the most essential drilling parameters for cost efficiency and is likely to be the preferred operating mode under normal drilling conditions.

2. In lack of proper simulation environment of a proper wellbore, a verification strategy was developed to simulate the response from a "real well". The measurements from the available well data, as stated in the last bullet point in the task objectives, are thus made from this simulated "real well". This is described more thoroughly in subsection 6.2.2.

Further, the thesis has a total of nine sections, including the introduction. Section 2 provides theoretical background and covers the basics of the drilling process and the MPC. Next, section 3 presents the Bourgoyne and Young ROP-model and parameter estimation techniques. How this model can be applied to the MPC and the derivation of an MPC problem is given in section 4. The software used and how the drilling data provided by AkerBP were structured and handled are presented in section 5. Implementation, results, and discussion is given in section 6, 7, and 8, respectively. Lastly, a conclusion is drawn, and suggestions on future work are given in section 9.

2 Theoretical Background

This section provides necessary theory to understand some of the chief concepts regarding the drilling process. Further, the basics of model predictive control is covered.

Note that most of the theoretical background, except for subsection 2.1.5 (which is new), was found through a literature review conducted in the specialization project (Kommedal [2020]) as well as information shared by AkerBP, and is reproduced and elaborated where necessary.

2.1 The Drilling Process

2.1.1 Drilling Platform Types

In offshore drilling there exists several types of drilling rigs, which can be divided into fixed or floating rigs. When drilling at modest depths, it is usually preferable with fixed rigs, like the jacket rig illustrated in Figure 2.1 (a). This yields stable operations, even in hard weather conditions. As for deepwater operations, the floating rigs like the one depicted in Figure 2.1 (b) is required. Naturally, these will be more affected by weather conditions. They typically utilize a technique called *dynamic positioning* which allows the rig to maintain a stationary position in the sea, regardless of waves and currents, by active control of a set of thrusters located on the hull of the floating rig (King [2020]). However, the floating rig is still subjected to heave motions, meaning that it fluctuates vertically due to the waves. Nikoofard et al. [2013] shows how this motion can create a significant change in the pressure in the annulus, which is the void between the drill string and the walls of the drilled open hole or casing (see Figure 2.2), as a result of the drill string fluctuating vertically in the well. They used MPC to improve the attenuation of the heave motion.

However, the Valhall IP platform is a jacket platform, and is therefore not subjected to these motions. This means that the MPC developed in this thesis does not have to consider such vertical fluctuations.

(a) Valhall IP, AkerBP - Jacket platform
(Source: Norsk Oljemuseum)

(b) DSS-38, Keppel - Semisubmersible Rig
(Source: Rigzone)

Figure 2.1: Jacket Rig vs. Floating Rig

2.1.2 Setup of a Jacket Platform

A simplistic illustration of a typical setup of the drilling process on an offshore jacket platform is given in Figure 2.2:

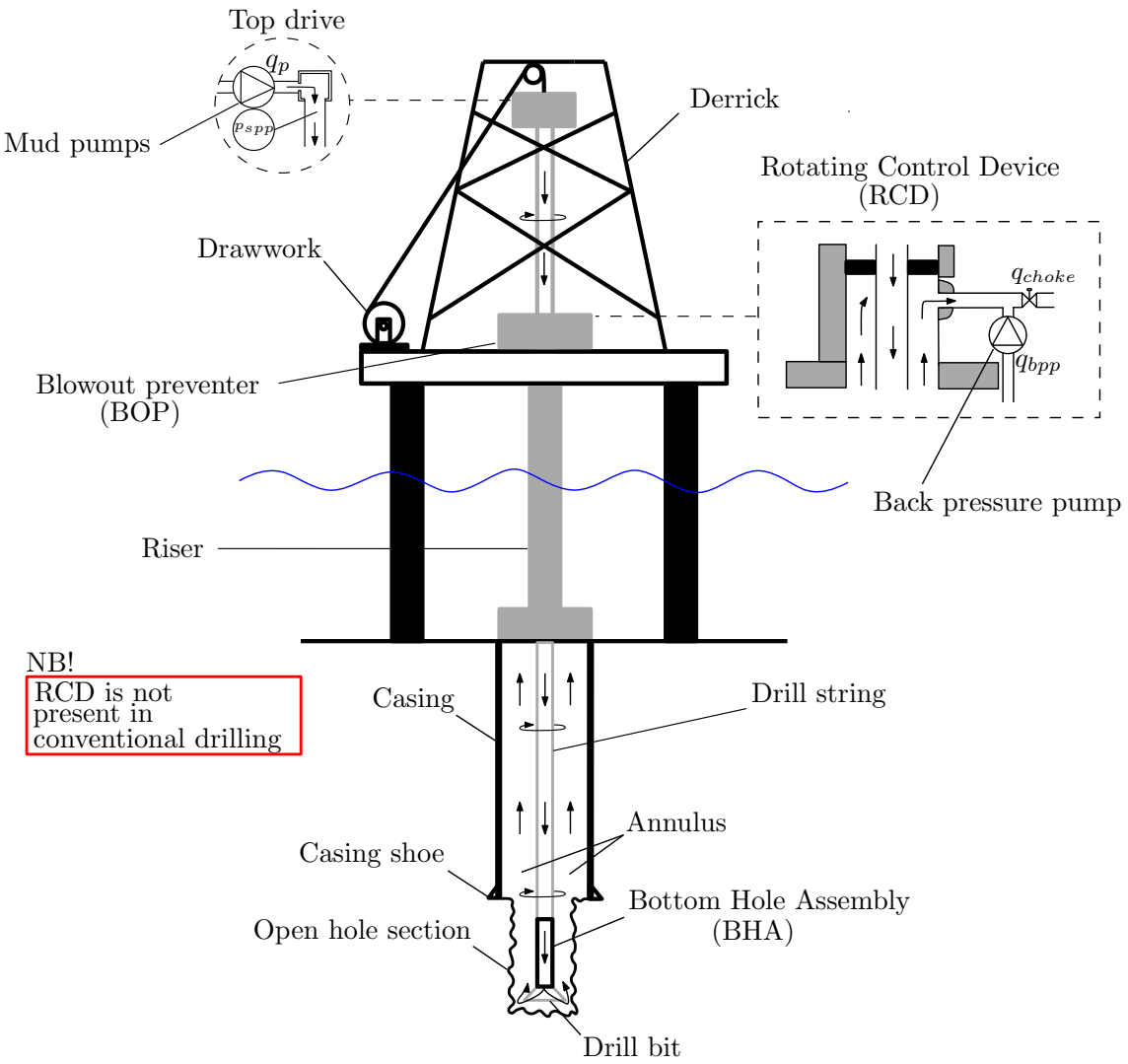


Figure 2.2: Typical setup of an offshore jacket platform. (Illustration adapted from: Stamnes [2011])

Drawworks

Located on the rig floor, one can see the drawworks. This is a large-diameter steel spool that is used to reel in and out a drilling line attached to the top drive. To ease the load of the drawworks, the drilling line is threaded through something called a traveling block. This is a multi-sheave pulley system that mechanically enables heavy loads like the drill string to be lifted.

Top drive and mud pumps

The top drive is the machinery that applies rotary motion to the drill string. It moves vertically along the derrick when it is lowered and hoisted by the drawworks. While drilling, a fluid called drilling mud, further elaborated in subsection 2.1.3, is continuously pumped through the top drive and further down the drill string by mud pumps.

Drill string, BHA, and drill bit

Attached to the top drive is the drill string. It consists of a series of hollow steel drill pipes, each at approximately *14m* at Valhall IP. In the last section of the drill string, one can find the Bottom Hole Assembly (BHA) before the drill bit at the very end. The BHA is a series of tools used for logging, steering, testing, and completion. All data regarding the condition of the wellbore usually is collected here. The drill bit is the component that transfers torque from the drill string to the bottom of the open hole section of the wellbore. Today, it is normal to use Polycrystalline Diamond Compact (PDC) bits, which are hard material bits designed to efficiently dig into the wellbore.

Riser, BOP, casing

The riser, Blowout preventer (BOP), and the casings are all considered safety barriers when drilling a well. These are supposed to prevent hazardous situations on the rig surface and avoid contamination of surrounding fauna. Located on the rig surface is the BOP, which essentially is a large safety valve. If the rig makes use of a Rotating Control Device (RCD), the BOP will be located beneath. It seals the well in uncontrolled situations, such as when there are severe erratic pressures or unexpectedly large volumes of fluids are flowing into the annulus from the surrounding formations (called a formation kick). The safety barrier that encloses the drill string between the BOP and the rig surface is called a riser. It ensures insulation between the fluids coming from the wellbore and the surrounding seawater. Lastly, the safety barrier between the wellbore and the annulus is called casing. These are large-diameter steel pipes, made to withstand the load from the surrounding formation movements.

Rotating Contol Device, choke, and back pressure pump

Subsection 2.1.4 introduces three different drilling techniques, where the main differences are how they utilize pressure control. Two of these, namely Underbalanced Drilling (UBD) and Managed Pressure Drilling (MPD), are dependant on a RCD, back pressure pump, and a choke. These techniques require that the circulation system, shown by the arrows in Figure 2.2, are closed against the atmosphere. The RCD allows the drill string to rotate and move vertically while ensuring a closed circulation system. The choke and the back pressure pump actively control the well's pressure, further elaborated in subsection 2.1.4. Note that the RCD, choke, and back pressure pump are not present when using the third drilling technique, namely conventional drilling. In this case, the circulation system is open to the atmosphere on the rig surface.

2.1.3 Drilling the Wellbore

A wellbore on Valhall is typically drilled in sections of approximately *28m*, which is the length of so-called stands, consisting of two drill pipes in series. After a section is drilled, the top drive comes to a halt allowing a new stand of drill pipe to be connected. This is done by wedging the drill string with three steel wedges, liberating the top drive. A new stand can then be connected to the drill string before the top drive is attached, allowing the next section to be drilled.

While each section is drilled, a fluid called drilling mud is continuously pumped through the drill string and tiny nozzles in the drill bit into the annulus and further up to the surface. This circulation system consists of tanks with drilling mud located on the rig surface, which feeds the mud pumps which again is circulating the mud. The drilling fluid may be composed of water-base muds, oil-base muds, or gas (Caenn et al. [2011]) and is usually a heavy and viscous fluid mixture. The main functions of the drilling mud can be listed as:

- Transporting the drilled cuttings, which is the crushed mass from the drilled borehole, to the surface and thereby providing a clean hole
- Providing hydrostatic pressure to prevent the formations of the open hole sections from collapsing into the wellbore
- Cooling, cleaning, and lubricating the drill bit and drill string
- Reducing friction between the drill string and the sides of the hole
- Used to assess the hole condition as loss or gain of drilling fluids may imply cracked formation or formation fluids entering the wellbore (called a kick) respectively
- Forms a low-permeability filter cake that seals openings in the formation

Thus the drilling mud needs to be sufficiently viscous to carry the drilled cuttings, and it should have a high enough density to apply sufficient hydrostatic pressure. Further exploration of the drilling mud properties is beyond the scope of this thesis and will not be considered, however Caenn et al. [2011] provides a good read on the subject.

After drilling an open hole segment, the casing is lowered into the open hole with the intent to protect the surrounding area from drilling mud and the wellstream from outside contaminations. It also provides stabilization, preventing the walls from collapsing into the wellbore. The outer diameter of the casing is intentionally smaller than the surrounding open hole. When the casing is set, this void is filled with cement. The cementing layer acts as an extra protective barrier as well as keeping the casing in place.

Before placing the casing, it is necessary to pull the drill string out of the wellbore. This procedure, as well as hoisting the drill string into the open hole, is called tripping. Tripping pipe in and out should be done with care, as it creates what is called swab and surge pressures. It can be seen as moving a piston out of or into a cylinder, it will create a pressure which can either crack the formation or make it collapse into the wellbore resulting in losses or kicks respectively.

2.1.4 Drilling Techniques and Pressure Control

As explained in the previous subsection, the drilling fluid is necessary to maintain stability in the open hole sections of the wellbore. The hydrostatic pressure increases with the true vertical depth (TVD [ft] shown in figure 2.4), and mud weight (MW [ppg]) with the following relation (Lapeyrouse [2002]):

$$p_{hyd} = 0.052 \times MW \times TVD \tag{1}$$

If the drill string is at rest, and the drill fluid flow is stopped, the bottom hole pressure (BHP) will equal the hydrostatic pressure. However, when the fluid flows, and the drill string rotates, the BHP will also be dependent on the pressure loss caused by annular friction (Malloy et al. [2009]):

$$p_{bh} = p_{hyd} + p_{af} \tag{2}$$

Note that p_{af}, commonly referred to as Equivalent Circulation Density (ECD), represents the pressure loss for both annular friction and due to any extra mass added by the transported cuttings. Equation (2) represents the BHP for *conventional drilling* setup. Here the mud exits the top of the wellbore open to the atmosphere before the mud is returned through a flowline to a mud-gas separator and handling equipment for the solids and then it is returned to the mud pit (Malloy et al. [2009]). Note that the way to control the BHP in conventional drilling is to adjust the MW which affect p_{hyd} and controlling the mud pump flow rates which affects p_{af}. The goal of conventional drilling is to stay well within the pressure limits of the formation known as the pore pressure and fracture gradient, illustrated in Figure 2.3. This is called overbalanced drilling, referring to the BHP exceeding the formation pore pressure.

On the other hand *underbalanced drilling* represents a drilling technique to stay below the pore pressure, hence bringing formation fluids (oil, gas, water) to the surface. This improves the rate of penetration (ROP), which is the speed the drill bit penetrates the bottom hole surface, due to reduced pressure against the open hole walls. This is where the RCD, choke, and back pressure pump are used, to create back pressure which is used for more efficient control of the BHP. Equation (2) can then be extended as following (Nauduri et al. [2009]):

$$p_{bh} = p_{hyd} + p_{af} + p_{bp} \tag{3}$$

Where p_{bp} serves as the surface back pressure applied, yielding yet a control parameter for the BHP.

Lastly, *managed pressure drilling* is a continuation of the UBD technology, but whereas UBD allows formation influx to utilize the effectiveness of the increased ROP, MPD will try to avoid influx by keeping the BHP above the pore pressure. An underlying risk of UBD is that it may cause too much formation influx, resulting in downhole problems and consequently non-productive time (NPT). The reduction of drilling costs due to less NPT is the main advantage of MPD. NPT is often a result of problems around depths where there is a close proximity between pore pressure and fracture pressure of the formation (Rehm et al. [2009]).

Møgster et al. [2013] showed how MPC could be used for MPD. Here the main mud pump and a choke were used to actively control the BHP, while all the imposed constraints were obeyed.

In the Valhall field most wells have been drilled with conventional overbalanced drilling. But with the increasing field complexity as explained in subsection 1.1, MPD has been used for some of the recent drilling operations on Valhall.

Figure 2.3: Drilling windows from well at Valhall field. The colored areas show approximately the operating area for conventional drilling, MPD, and UBD. Note that the pressure gradients are usually associated with TVD but is here expressed with measured depth.
(Illustration inspiration from: Malloy et al. [2009], Courtesy: Arnfinn Grøtte, AkerBP)

2.1.5 Directional Drilling and Wellbore Trajectory

According to Inglis [2013], wells were drilled only in the vertical direction in the late 1800s. Later, it was discovered that these "vertical" wells actually deviated a whole lot from a straight vertical line. Wells that deviate from this vertical line is called deviated wells. At first, deviated wells were considered to be a disadvantage as it meant more footage required to be drilled to reach the desired depth and more wear and tear on the drill pipes due to curvy paths. Around the 1930s, the first deliberately deviated wells were drilled, today known as directional drilling. Inglis [2013] lists some of the applications for directional drilling as

- **Sidetracking**
  Suppose that a part of the BHA is stuck in the bottom of the well. Then a sidetrack can be drilled, using a downhole motor to deflect the BHA, allowing for a new path to be drilled.
- **Avoiding geological problems**
  Certain areas of the formation may have geological properties, which are desirable to avoid. This can be seen as an obstacle between the rig and the target location, as illustrated in figure 2.4.
- **Controlling vertical holes**
  If the goal is to drill a straight vertical well, directional drilling can be used to assure that any deviation is minimized.
- **Offshore development drilling**
  When drilling offshore, as opposed to land-based drilling, drilling vertical wells from individual platforms is very expensive. Instead they utilize directional drilling to several holes from the same fixed platform, as seen in figure 1.1 (b).
- **Horizontal drilling**
  For thin oil column reservoirs, meaning that the height of the formation containing the oil is small, a horizontal well may be advantageous to increase the platform's drainage area.

The wellbore path can be described by the parameters shown in figure 2.4. The Measured Depth (MD) represents the wellbore's total depth, that is, the length of the drill string when the drill bit is at the bottom of the well. In comparison, TVD is the vertical depth of the wellbore. The inclination angle, I, is the angle at which the wellbore deviates from a straight vertical line at the point it is measured. Lastly, the azimuth angle, α, is the deviation between the measured point and the true or magnetic north.

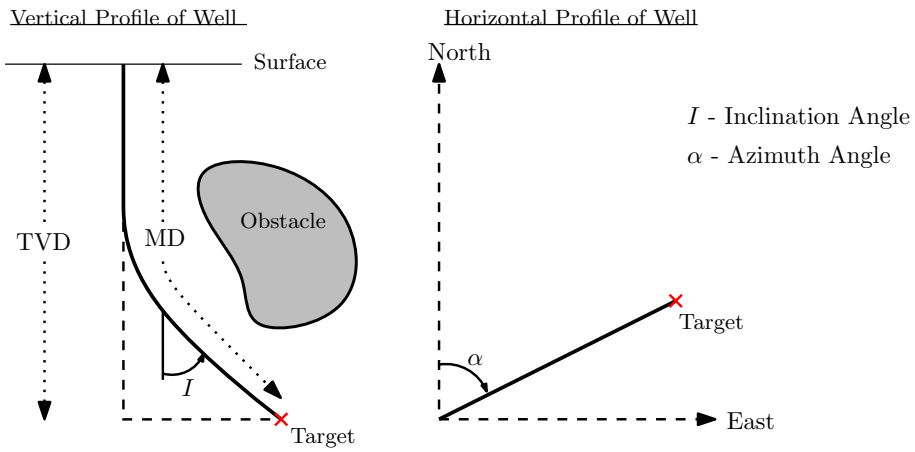


Figure 2.4: A vertical and horizontal profile view of the well showing the difference between TVD vs. MD, and inclination vs. azimuth angle (Illustration adapted from: Inglis [2013])

2.1.5.1 Well Surveying

To determine the drill bit's location and orientation and thus a "snapshot" of the wellbore trajectory, well surveying is done regularly. Surveying instruments are then used to measure inclination and azimuth angle at various depths, using measurement while drilling (see subsection 2.1.6). The length between the points of surveys, usually called survey stations, varies over the well's different sections. For critical sections, such as the start of the well or sections with much curvature, the survey stations may have intervals of approximately $12m$. For straight-line sections, every second or third drill pipe interval may be sufficient (Farah [2013]).

According to Bourgoyne et al. [1991] there are 18 or more techniques to calculate the wellbore trajectory between each survey station. These can be divided into two groups: those who use straight-line approximations and those who assume that the wellbore exists of curved segments. Three of these techniques are covered in the following three subsections. Further, figure 2.5 illustrates the main difference in the performance of the presented techniques in calculating the trajectory between survey stations S_1 and S_2.

2.1.5.2 Tangential Method

The tangential method, being amongst the simplest of the survey calculation methods, uses the inclination and azimuth angle at survey station S_2 and assumes constant angles over the whole length between the survey stations ΔMD. This yields the equations

$$\begin{aligned} \Delta N &= \Delta MD sin(I_2) cos(\alpha_2) \\ \Delta E &= \Delta MD sin(I_2) sin(\alpha_2) \\ \Delta TVD &= \Delta MD cos(I_2) \end{aligned} \tag{4}$$

This method is subject to substantial errors if the inclination and azimuth angles are different at the two survey stations, depending on the length between each survey station. It should therefore not be used, unless the length between the survey stations are no longer than the length of the survey tool (Farah [2013]).

2.1.5.3 Angle Averaging Method

To cope with the errors that the tangential method yields, by only considering the angles of the next survey station, the averaging method utilizes both survey stations' angles. It uses the average of the angles over the length ΔMD, resulting in the equations

$$\begin{aligned} \Delta N &= \Delta MD sin\left(\frac{I_2 + I_1}{2}\right) cos\left(\frac{\alpha_2 + \alpha_1}{2}\right) \\ \Delta E &= \Delta MD sin\left(\frac{I_2 + I_1}{2}\right) sin\left(\frac{\alpha_2 + \alpha_1}{2}\right) \\ \Delta TVD &= \Delta MD cos\left(\frac{I_2 + I_1}{2}\right) \end{aligned} \tag{5}$$

It is shown in Bourgoyne et al. [1991] that this method yields a considerably less error than the tangential method. However, it still uses a straight line approximation to calculate the trajectory between the two survey stations. This will result in a deviation from the actual wellbore path in between the survey stations, as illustrated in figure 2.5, if the azimuth and inclination angle have changed considerably from S_1 to S_2.

2.1.5.4 Minimum Curvature Method

The minimum curvature method developed by Taylor and Mason [1972] accounts for the curvature of the wellbore path instead of the tangential and angle averaging methods straight-line approximation approach. The method uses the following equations to describe the change of position

$$\begin{aligned} \Delta N &= \frac{\Delta MD}{2}\left[sin(I_1)cos(\alpha_1) + sin(I_2)cos(\alpha_2)\right] RF \\ \Delta E &= \frac{\Delta MD}{2}\left[sin(I_1)sin(\alpha_1) + sin(I_2)sin(\alpha_2)\right] RF \\ \Delta TVD &= \frac{\Delta MD}{2}\left[cos(I_1) + cos(I_2)\right] RF \end{aligned} \tag{6}$$

where

$$\begin{aligned} RF &= \frac{2}{\beta} tan\left(\frac{\beta}{2}\right) \\ \beta &= cos^{-1}\Big(cos(I_2 - I_1) - sin(I_1)sin(I_2)\left[1 - cos(\alpha_2 - \alpha_1)\right]\Big) \end{aligned} \tag{7}$$

Here RF is a ratio factor between the straight line segments $S_1B + BS_2$ and the curved line segments $S_1Q + QS_2$ and can be derived from geometrical relations between S_1, S_2, O, B, and Q which is shown in figure 2.5. To avoid singularity in RF, as $\beta \to 0$, RF is set to equal one for $\beta < 0.25$ (Bourgoyne et al. [1991]). Further, β represents the overall angle change of the drill pipe between the two survey stations.

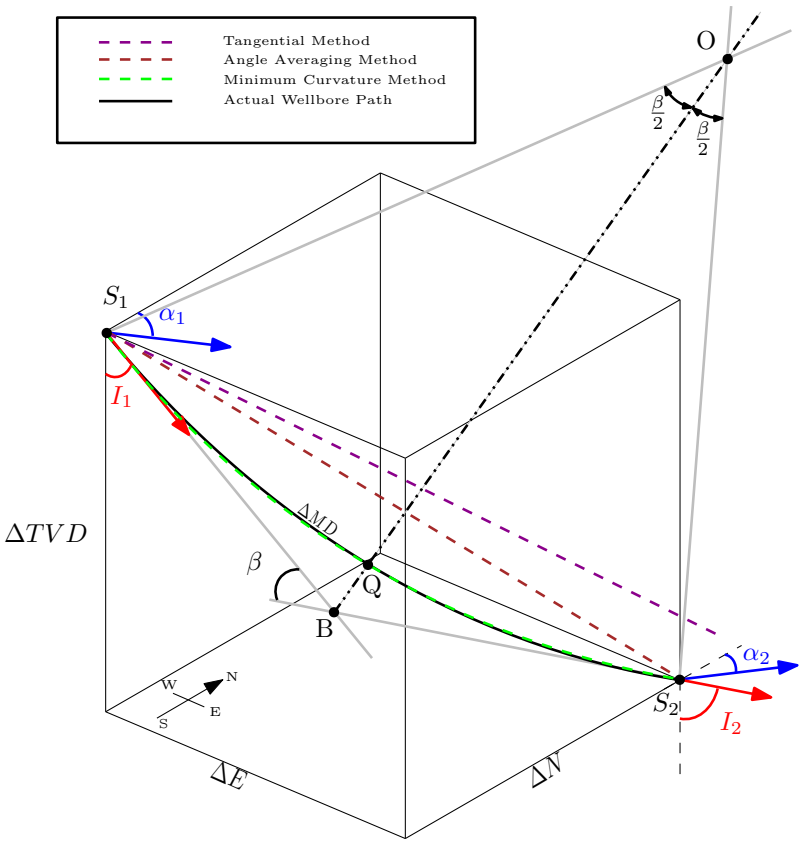


Figure 2.5: Calculated paths of tangential, angle averaging, and minimum curvature method compared to the actual wellbore path between two survey stations (Illustration adapted from: Farah [2013])

2.1.6 Measurement and Logging While Drilling

There are two measurement techniques typically used to evaluate the drilling conditions downhole. Measurements While Drilling (MWD) typically refers to the measurements regarding the wellbore geometry and trajectory. It yields the measurements required to evaluate the wellbore placement. Logging While Drilling (LWD) yields the measurements for evaluating formation and environmental conditions, like formation porosity, density, and resistivity, to mention a few. For the sake of simplicity, M/LWD will be used as a common name for both of them. A literature survey done by Pastorek et al. [2019] covers the capabilities and limitations of data collection equipment today. One of the tables presented yields the most commonly collected data for both surface and downhole tools in drilling operations:

	Surface Data	**Downhole Data**
Mud Data	Pit volume Mud temperature Mud pressure Mud weight Pump strokes	N/A
Well Data	Temperature Pressure Gas measurement	Temperature* Pressure
Directional Data		Inclination* Azimuth*
Drilling Mechanics	RPM Weight on bit Torque Bending moment Rotary torque Hook load ROP	RPM* Weight on bit Torque on bit Bending moment Downhole vibration*
Geological Data	Cuttings analysis	Density* Porosity* Resistivity* Gamma*

Table 1: Commonly collected data from M/LWD tools (Pastorek et al. [2019])

While most of the measurements are typically done on the rig surface, the M/LWD measurements yield the downhole data, which best represents the wellbore condition.

Note that the most common measurements downhole is marked with (*). Another important remark for the M/LWD technology is that it might be limited by the area of operation. In deep wells subjected to temperatures above 150^oC, the M/LWD tools may not be applicable due to the equipment's temperature limitations (Pastorek et al. [2019]).

2.1.7 Data Transfer

Real-time drilling data will be crucial to optimize the drilling process. The data transmission techniques are referred to as telemetry and can be done by wire or by wireless techniques. Telemetry is the process of transmitting measured data from a remote location in a suitable form to the surface (Schlumberger). Wired telemetry is typically done by fiber or cable incorporated in the drilling pipes and yields enhanced bandwidth. Wireless technologies encompass electromagnetic telemetry, acoustic telemetry, or pulsing in the mud (Shao et al. [2017]). An extensive literature review of mud pulse telemetry was done by Mwachaka et al. [2019], where the characteristics and capabilities of different communication technologies are compared. They presented the following table:

Features	**Electromagnetic**	**Acoustics**	**Mud pulses**	**Wired drill pipe**
Max. transm. data rate (bps)	10	20	20	57600
Maximum depth (meters)	5500	3700	12200	Unlimited
Data quantity	Medium	Low	High	Very high
Signal attenuation	High	High	Medium	N/A
Signal interference	High	Medium	Medium	Low
Costs	Medium	Medium	Low	High

Table 2: Comparing different M/LWD telemetry technologies (Mwachaka et al. [2019])

Table 2 shows the main differences by the most commonly used M/LWD telemetry technologies. Note how M/LWD by electromagnetic and acoustic telemetry only is applicable for depths less than 5500 and 3700 meters, respectively. They are also subjected to substantial signal attenuation and interference and are thus seldom used compared to M/LWD by mud pulsing or wired pipe.

While M/LWD by wired drill pipe is superior to mud pulsing in maximum transmission data rate, depth, and data quantity, it comes with a far greater maintenance cost. Thus mud pulsing telemetry has been extensively used due to simple operations and low costs.

Edwards et al. [2013] summarized BP's wired pipe trials from ten wells at five different locations between 2007 and 2010. Some of the apparent benefits of wired pipe were listed; Greater visibility of the wellbore allows measurements along the drill string and enables improved tools in the BHA with higher bandwidth requirements. They also state that while ordinary mud pulse typically updates the downhole parameters every two minutes at the rig surface, the wired pipe telemetry allows for updates every two seconds. Thus wired pipe telemetry might be required to provide an adequate amount of data to the MPC.

2.2 Model Predictive Control

The first use of Model Predictive Control strategies originates back to the late seventies. It has been applied in different fields as the chemical process industries, for robotic control, and within the cement industry, to mention a few (Camacho and Bordons [2007]). MPC makes use of a model of the process to predict future outputs within a finite time horizon, using an optimal control strategy to minimize an objective function. As seen in Figure 2.6, an open-loop optimization problem is solved at each timestep t, which yields a sequence of predicted optimal control signals u_t. Further, it applies the first control signal, $u_{t'}$, from this calculated sequence to the plant. This is all done while satisfying a set of constraints that is formulated by the optimization problem.

Almost every control problem imposes constraints; limitations in actuators, safety limits of the applications such as max pressure, temperature or velocities, or just physical limitations like pushing a gas pedal on the car to the bottom (Mayne et al. [2000]).

Note that in the literature, the input variables are normally referred to as *manipulated variables* (MV), output variables (states) which are going to be controlled as *controlled variables* (CV). These are further defined in section 4.

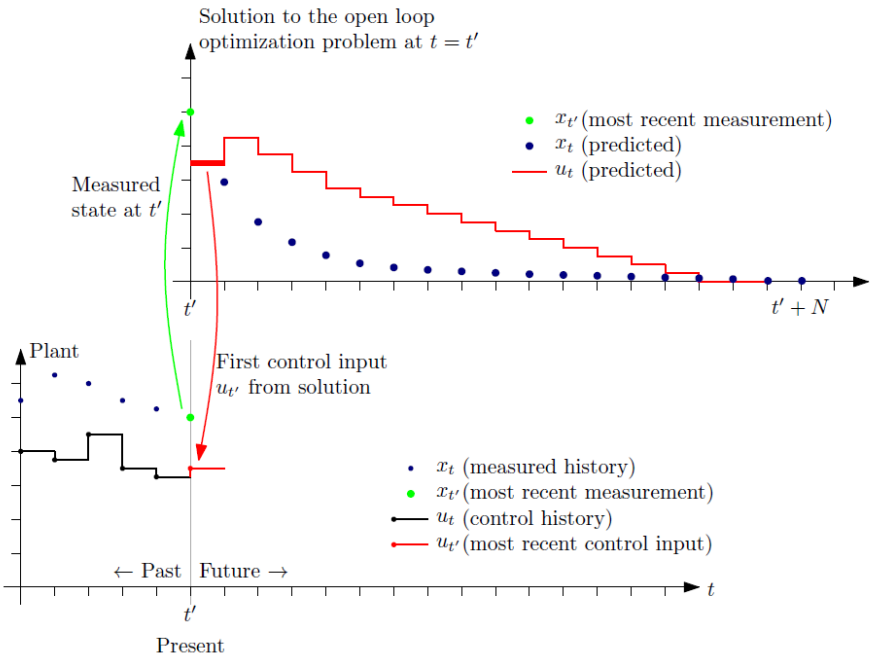


Figure 2.6: Illustration of the MPC algorithm (Foss and Aksel N. Heirung [2016]). The bottom graph shows the plant state and input signal for past, current and future time steps. The upper graph shows the open loop solution to the optimization problem at current time step t' until a finite time horizon $t' + N$ (also called prediction horizon), whereas the first control input is applied to the plant

The MPC strategy utilizes an old method of control design, as it essentially solves a standard optimal control problem, but it differs from other controllers due to the *on-line* solving of the problem at each time step whereas previous controllers used *off-line* feedback providing optimal control for each state (Mayne et al. [2000]). Some of the advantages and drawbacks of the MPC can be summarized as (Camacho and Bordons [2007]):

+ Can easily deal with multivariable cases, despite coupling between the states
+ Has the ability to deal with constraints
+ Solves an optimal control problem
+ Relatively intuitive control strategy and tuning parameters, with an easy-to-implement control law
− The derivation of the control law is more complex than for example the PID control
− For a dynamic process, all the computations must be carried out at every sampling time, which imposes high computational load.
− Requires a model representing the most important dynamics of the process to make sufficiently accurate predictions.

Note, however, that the MPC should be updated using available online measurement data. Even the most precise model of the process might yield inaccurate predictions over time without any measurement updates. In contrast, a very simplistic model might be sufficient if it is frequently updated with measurement data.

2.2.1 Mathematical Formulation of the MPC Problem

As described in subsection 2.2 the MPC solves an online dynamic optimization problem at each timestep, applying the first optimal control input to the plant. This can be mathematically formulated, as done by Foss and Aksel N. Heirung [2016], in the following way:

$$\min_{z\in\mathbb{R}^n} f(z) = \sum_{t=0}^{N_p-1} \underbrace{\frac{1}{2}X_{t+1}^T Q_{t+1} X_{t+1} + d_{xt+1}^T X_{t+1}}_{\text{Penalizing the CV}} + \underbrace{\frac{1}{2}U_t^T R_t U_t + d_{ut}^T U_t}_{\text{Penalizing the MVs}} + \underbrace{\frac{1}{2}\Delta u_t^T R_{\Delta t} \Delta u_t}_{\substack{\text{Penalizing the} \\ \text{MVs rate of change}}} + \underbrace{\rho^T \boldsymbol{\varepsilon} + \frac{1}{2}\boldsymbol{\varepsilon}^T S \boldsymbol{\varepsilon}}_{\substack{\text{Penalizing} \\ \text{constraint violations}}} \tag{8}$$

subject to

$$x_{t+1} = g(x_t, u_t), \qquad t = 0, \ldots, N_p - 1 \tag{9a}$$
$$x_0, u_{-1} = given \tag{9b}$$
$$\boldsymbol{\varepsilon} \geq 0 \tag{9c}$$
$$x^{low} - \boldsymbol{\varepsilon} \leq x_t \leq x^{high} + \boldsymbol{\varepsilon}, \qquad t = 0, \ldots, N_p \tag{9d}$$
$$u^{low} \leq u_t \leq u^{high}, \qquad t = 0, \ldots, N_p - 1 \tag{9e}$$
$$-\Delta u^{low} \leq \Delta u_t \leq \Delta u^{high}, \qquad t = 0, \ldots, N_p - 1 \tag{9f}$$

where

$$z^T = (x_1^T, \ldots, x_{N_p}^T, u_0^T, \ldots, u_{N_p-1}^T, \boldsymbol{\varepsilon}) \tag{10a}$$
$$n = N_p \cdot (n_x + n_u + n_{\boldsymbol{\varepsilon}}) \tag{10b}$$
$$X_{t+1} = x_{t+1} - x_{t+1}^{ref} \tag{10c}$$
$$U_t = u_t - u_t^{ref} \tag{10d}$$
$$\Delta u_t = u_t - u_{t-1} \tag{10e}$$

Here the *objective function* to be minimized, namely $f(z)$, is a quadratic objective function subjected to the constraints shown in 9a-9f. The variables, z, that are found by the chosen solver to minimize this function is called decision variables. Note that there exists a wide variety of dynamic optimization formulations and that this specific formulation is chosen to address some important properties. The objective function contains the matrices Q_{t+1}, R_t and $R_{\Delta t}$ which is always positive semidefinite and symmetric, for a convex objective function (see subsection 2.2.3). These matrices are called weighting matrices, containing the relative weighting of importance between each CVs/MVs in the diagonals, and is further elaborated in subsection 2.2.2. The first constraint in this formulation, equation 9a, represents the model of the process and can either be linear- or nonlinear dynamics.

Also worth noting is the two last terms of the objective function, and the added $\boldsymbol{\varepsilon}$ in 9d. The vector $\boldsymbol{\varepsilon}$ contains what is called slack variables, and is important in order to ensure the feasibility of the optimized solution at all times. In certain situations, the only way to find a feasible solution to the optimization problem requires a violation of one or more constraints. If all the constraints are hard, meaning that no violation is allowed, then in such situations there exists no feasible solution resulting in an error and the MPC will fail. However, by allowing a violation, the MPC is able to find a solution. The constraints containing the slack variables are called soft. The last two terms

in the objective function are added to penalize the violation of the soft constraints. Note that usually only one of these terms, that is either the linear or the quadratic term, is used at a time in the objective function. Further elaboration of sizes and structures of the variables and matrices in the optimization problem can be found in Foss and Aksel N. Heirung [2016].

Further, equations 10c-10d yields the error variables of the states and inputs. Here x_{t+1}^{ref} is the desired states at timestep $t+1$, while x_{t+1} represent the true states given by the chosen model $g(x_t, u_t)$. Similarly, u_t^{ref} is the desired input at timestep t, with u_t representing the true input. Δu_t represents the rate of change of the input.

It is usually desirable to define a terminal set in the MPC formulation. The terminal set is an essential component when doing stability analysis of the MPC. However, due to the wellbore's continuously changing formation and conditions while drilling, such a set may be challenging to implement consistently. Thus, a more comprehensive simulation approach may be necessary to ensure that the MPC will be stable for the whole operation.

2.2.2 MPC Design Parameters

Selecting the proper design variables on the MPC is important to achieve a good performance, but it will also affect the computational complexity of the MPC algorithm. Thus serving as a trade-off between the added complexity and the performance of the controller. Bemporad et al. [2005] has made a user's guide to Matlab's Model Predictive Control Toolbox, suggesting recommended practices to choose the design parameters of the MPC. Some of which are listed below:

Sample Time
It is normal to choose the sampling time T_s initially and keeping it constant while tuning the rest of the parameters. Smaller T_s will yield a higher rejection of unknown disturbances at the cost of a higher computational effort. The prediction horizon duration will be equal to $N_p \times T_s$, where N_p is the prediction horizon as shown in figure 2.6. The appropriate value of T_s will be highly dependent on the system dynamics, and bandwidth requirements for the specific application. As an example, in process control with the MPC acting as a supervisory controller the sample time might be in the area of minutes, while other applications might require $T_s < 1s$.

Prediction Horizon
An important feature of the MPC is its predictive capabilities. How far into the "future" the controller should look, is decided by the prediction horizon N_p. Just like the sample time, for most applications, it should be chosen early and kept constant in the design process. If the prediction horizon is too short, the controller loses its predictive capabilities, which might lead to control actions being applied too slowly. This in turn can lead to violations of constraints and infeasible solutions. On the other hand, a too long prediction horizon will lead to high computational efforts, and can also lead to unnecessarily large violations of the constraints. It should therefore be chosen just large enough to cover the significant dynamics of the system.

Control Horizon
The optimal control sequence, u_t, as depicted in the top graph of figure 2.6 is the predicted open-loop solution of the MVs which will yield the predicted evolution of x_t shown in the same graph. Note that in this illustration the control horizon, which represents the number of MV moves which should be optimized at each control interval, equals the prediction horizon. However, only the first MV move will be used and the rest are discarded. Hence it is normally beneficial to let the control horizon be sufficiently smaller than the prediction horizon (but larger than one) to avoid unnecessary computational expenses. Note also that the mathematical formulation in subsection 2.2.1 does not include a separate control horizon, and that it might be reformulated if need be.

Constraints
Usually, the constraints of an optimization problem are dictated by the application. As mentioned in subsection 2.2 these involve physical limitations on the actuators, safety limits of states et cetera. Yet there are some important considerations to take. Physical limitations should be included on all MVs as hard constraints. For the CVs, one should try to minimize the number of constraints to minimize the computational complexity of the optimization problem. The CV constraints may

possibly be softened, if the constraint is not critical in terms of safety or for other reasons, in order to ensure feasibility as discussed in subsection 2.2.1.

Weights
The relative importance of each CV and MV can, as stated in subsection 2.2.1, be expressed in the weighting matrices Q_{t+1}, R_t and $R_{\Delta t}$. Hence these are the primary tuning parameters in our control problem. Note that whether or not these matrices are time-dependent depends on the application. Consider an optimization problem, as stated in equation (8), with two error states $X_{t+1} = [X_{1,t+1}\ X_{2,t+1}]^T$ and the following time-invariant Q-matrix:

$$Q = \begin{bmatrix} 1 & 0 \\ 0 & 10 \end{bmatrix} \tag{11}$$

Based on this matrix one can see that the square of $X_{2,t+1}$ is weighted ten times more than the square of $X_{1,t+1}$. If we look at these terms isolated from the rest of the optimization problem, it means that deviations in $x_{2,t+1}$ from $x^{ref}_{2,t+1}$ will be penalized more then deviations in $x_{1,t+1}$ from $x^{ref}_{1,t+1}$. This will, combined with the R- and R_Δ-matrix, decide how much the CVs and MVs will be penalized as a function of their respective deviations.

2.2.3 Convexity vs. Non-Convexity

A critical factor for how fast the MPC solves the optimization problem is whether the problem is convex or not. Both sets and functions may be convex. According to Nocedal and Wright [2006], a set $S \in \mathbb{R}^n$ is convex if any straight line connecting two points in S lies entirely within S. Mathematically this can be expressed as if for any two points $x \in S$ and $y \in S$ then $\alpha x + (1-\alpha)y \in S, \quad \forall \alpha \in [0,1]$, and graphically expressed as in figure 2.7. It is further specified that a function is convex if for any two points $x \in S$ and $y \in S$, where S is a convex set, and

$$f(\alpha x + (1-\alpha)y) \leq \alpha f(x) + (1-\alpha)f(y), \quad \forall \alpha \in [0,1] \tag{12}$$

is satisfied. To summarize, a constrained optimization problem is convex if it satisfies the three conditions:

1) The objective function is convex

2) The equality constraints are linear

3) The inequality constraints are convex[2]

where the set of constraints 2-3 yields a convex set. The most significant feature of convexity is that any local solution is also a global solution, which is vital to reduce the time it takes to solve the optimization problem. This is not the case for a non-convex problem; thus, solving these optimization problems efficiently is much more complex and requires more sophisticated algorithms.

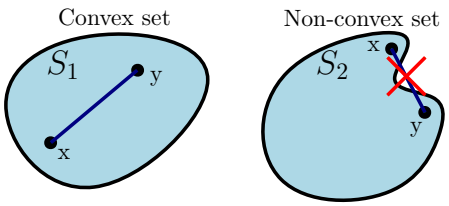


Figure 2.7: Convex set S_1 vs. non-convex set S_2

[2]Note that in Nocedal and Wright [2006] it says concave. This is because they have defined the inequality constraints as $c_i(x) \geq 0$, instead of the more common definition of $c_i(x) \leq 0$, and that the negated version of a convex function is concave.

3 Model and Parameter Estimation

This section, except for subsection 3.1.3 (which is new), has been reproduced and elaborated from Kommedal [2020].

As mentioned in subsection 2.2, the MPC requires a model of the process to make sufficient predictions. An extensive literature study was conducted by Eren and Ozbayoglu [2010], where the most important contributions to optimize drilling parameters in real-time during the drilling operation were addressed. They stated that one of the most significant contributions was made by Bourgoyne and Young [1999]. They performed a multiple regression analysis on historical drilling data in order to determine the optimal WOB, top drive RPM and bit hydraulics. Sui et al. [2013] further developed a moving-window method for the multiple regression analysis, considering only the most recent historical drilling data. Further, they employed an MPC strategy to optimize the ROP using the Bourgoyne and Young model (B&Y-model). Hence, the B&Y-model seems to be a promising prediction model for the MPC and will be further elaborated in this section.

3.1 Bourgoyne and Young ROP-Model

Bourgoyne and Young [1999] developed a mathematical drilling model that takes formation strength, formation depth, formation compaction, the pressure differential across the bottom hole, bit diameter and weight, rotary speed, bit wear, and bit hydraulics into account. ROP can be defined as

$$ROP(t) = \frac{dh}{dt} \tag{13}$$

where dh/dt is the change of bit depth h with small change in time t. The following model was then proposed by B&Y:

$$\widehat{ROP}(t) = Exp(a_1 + \sum_{j=2}^{8} a_j X_j) \tag{14}$$

where Exp(z) is the exponential e^z. An alternative formulation emphasizes the individual contribution from all the drilling effects included in the model as exponential functions:

$$\widehat{ROP} = f(1)f(2)...f(8) = \prod_{j=1}^{8} Exp(a_j X_j) \tag{15}$$

The model incorporates the effect of the drilling parameters, X_j, on the modeled ROP. Further elaboration of the drilling parameters are found in Table 3. An explanation of each individual contribution, $f(j)$, $\forall j = 1, ..., 8$ in equation (15) is given in subsection 3.1.1. The constants a_j, $\forall j = 1, ..., 8$ can be found through parameter estimation techniques, which is further elaborated in subsection 3.1.2 and 3.1.3. The coefficients definitions are given in table 5.

Parameters	Description		Sub-equations	Effect description
h	True vertical depth of bit		$X_1 = 1$	Dummy value
h_0	Point of normalization		$X_2 = h_0 - h$	Vertical depth
g_p	Pore pressure gradient		$X_3 = h^{0.69}(g_p - g_{p_0})$	Formation compaction
g_{p_0}	Point of normalization		$X_4 = h(g_p - p_{af})$	Differential pressure
p_{af}	Equivalent circulation density		$X_5 = ln(\frac{w/d_B - w_0/d_B}{4 - w_0/d_B})$	Bit type and weight
w	WOB		$X_6 = ln(r/r_0)$	Rotary speed
w_0	Threshold for bit weight		$X_7 = -H$	Bit tooth wear
d_B	Bit diameter		$X_8 = ln(\frac{F_j}{F_{j_0}})$	Bit hydraulics
r	RPM			
r_0	Given RPM		$F_j = 0.01823 C_d q \sqrt{\rho \Delta p_b}$	
H	Fractional tooth height worn away		$\Delta p_b = \frac{8.311 \times 10^{-5} \rho q^2}{C_d^2 A_T^2}$	
ρ	Mud density			
q	Flow rate			
F_j	Hydraulic impact force			
F_{j_0}	Point of normalization			
Δp_b	Pressure drop across the bit			
C_d	Discharge coefficient			
A_T	Total bit nozzle area			

Table 3: Drilling Parameters and ROP-model parameters adapted from Sui et al. [2013] and Bourgoyne et al. [1991]

3.1.1 The Effect of $f(j),\ \forall j = 1, ..., 8$ on the ROP-Model

Eren and Ozbayoglu [2010] lists the effects of each exponential function $f(j),\ \forall j = 1, ..., 8$ as formulated in equation (15), which forms the basis for the following subsection.

The first, $f(1)$, represents the formation strength, also known as the formation's drillability. Note that only the coefficient a_1 affects the drillability, as X_1 is just a dummy value that always equals one. a_1 includes the effect of formation strength and other drilling parameters that are not a part of the mathematical model, such as drilled cuttings, equipment efficiency, et cetera.

Furthermore, $f(2)$ and $f(3)$ incorporates the effect of formation compaction, resulting from compressed sediments, which further increase the formation strength. $f(2)$ yields the impact of what is called normal compaction, which assumes an exponential decrease in $\widehat{ROP}$ as a function of increasing depth. Abnormal pressures in the formation may produce further compaction and are expressed through $f(3)$.

As explained in subsection 2.1.4, a BHP close to or below the pore pressure gradient, will yield a higher ROP. This is essentially the effect that $f(4)$ yields. Hence, if the ECD is much higher than the pore pressure, $f(4)$ will yield a low contribution to the $\widehat{ROP}$.

The exponential function $f(5)$ represents the contribution from the WOB and includes the bit diameter. Note that $Exp(aln(b)) = b^a$, implying that $f(5)$ yields a contribution proportional to $(w/d_B)^{a_5}$ to the $\widehat{ROP}$ as a function of the WOB and the bit diameter. $X_5's$ specific formulation is normalized for $4000lbf/inch \approx 714.32kg/cm$ per bit diameter. Further, w_0/d_b represents the minimum WOB required for the bit to dig further into the wellbore. According to Bourgoyne et al. [1991] this WOB threshold is often very small, and normally negligible unless the formation is relatively hard.

Likewise, $f(6)$ will yield a relation proportional to r^{a_6} to the $\widehat{ROP}$. This function represents the rotary speed RPM. The function will be normalized to equal 1 for a given RPM r_0.

The function $f(7)$ incorporates the effect of tooth wear on the $\widehat{ROP}$. If the bit wear is not of significant size, a_7 can be assumed zero and thus removed from the multiple regression analysis.

The last function, $f(8)$, models the bit hydraulics effect on the $\widehat{ROP}$. The jet impact force F_j is the force that is produced by the flow of mud through the bit nozzles as it hits the bottom hole. The discharge coefficient C_d is typically introduced since the rest of the equation, given by F_j, assumes frictionless flow, which is not the case for real applications (Bourgoyne et al. [1991]). Eckel and Bielstein [1951] found this coefficient experimentally and recommended a discharge value of $C_d = 0.95$. Further, A_T is the total area of the bit nozzles in the drill bit. It can be found by $A_T = \frac{\pi}{4(32)^2}(N_1^2 + N_2^2 + ... + N_n^2)$, where n denotes the number of nozzles and N denotes the nozzle diameter expressed in 32nds of an inch (i.e. diameter of nozzle one is $N_1/32$ in.). Note that this function, $f(8)$, is different from the one presented in Kommedal [2020]. However, in Bourgoyne et al. [1991], it is shown that both these functions affect the ROP similarly and argues that the choice of the impact force function is arbitrary.

3.1.2 Determining $a_j,\ \forall j = 1, ..., 8$ Through Multiple Regression

In order to determine the exponent coefficients $a_j,\ \forall j = 1, ..., 8$, specified in table 5, a multiple regression analysis may be applied. The purpose of multiple regression is simply to find a linear approximation that best represent the data that is being regressed. This subsection summarize the multiple regression approach done by Bourgoyne and Young [1999].

By comparing the real ROP from equation (13) with the modeled ROP from equation (14) one get the following equation:

$$\frac{dh}{dt} = Exp(a_1 + \sum_{j=2}^{8} a_j X_j) \tag{16}$$

Applying the natural logarithm to both sides of equation (16) yields:

$$ln(\frac{dh}{dt}) = a_1 + \sum_{j=2}^{8} a_j X_j \tag{17}$$

Now we can define the residual error between the logarithmic of the actual ROP, $ln(dh/dt)$, and the model $\widehat{ROP}$ given by the right hand side of equation (17). Defining r_k for the kth datapoint:

$$r_k = a_1 + \sum_{j=2}^{8} a_j X_j - ln(\frac{dh}{dt}) \tag{18}$$

The goal is to find a linear approximation which minimizes the residuals for n historical datapoints, where n is some value greater than 8. Further, it is common to evaluate the square of the residuals. This can be advantageous both as the sign of the residual can be neglected, and so that large residuals (square of large deviations) will have a larger impact on the resulting linear approximation. Hence, choosing a_1 to a_8 such that the squared of the residuals are at a minimum yields

$$\frac{\partial \sum_{k=1}^{n} r_k^2}{\partial a_j} = \sum_{k=1}^{n} 2r_k \frac{\partial r_k}{\partial a_j} = \sum_{k=1}^{n} 2r_k X_j = 0, \qquad \forall j = 1, ..., 8 \tag{19}$$

By substituting for r_k from equation (19) with right hand side of equation (18), a system of equations are obtained which can be solved for a_1 to a_8:

$$a_1 n + a_2 \sum_{k=1}^{n} X_2 + a_3 \sum_{k=1}^{n} X_3 + ... + a_8 \sum_{k=1}^{n} X_8 = \sum_{k=1}^{n} ln(\frac{dh}{dt}) \tag{20a}$$

$$a_1 \sum_{k=1}^{n} X_2 + a_2 \sum_{k=1}^{n} X_2^2 + a_3 \sum_{k=1}^{n} X_2 X_3 + ... + a_8 \sum_{k=1}^{n} X_2 X_8 = \sum_{k=1}^{n} X_2 ln(\frac{dh}{dt}) \tag{20b}$$

$$a_1 \sum_{k=1}^{n} X_3 + a_2 \sum_{k=1}^{n} X_3 X_2 + a_3 \sum_{k=1}^{n} X_3^2 + ... + a_8 \sum_{k=1}^{n} X_3 X_8 = \sum_{k=1}^{n} X_3 ln(\frac{dh}{dt}) \tag{20c}$$

$$\vdots \tag{20d}$$

$$a_1 \sum_{k=1}^{n} X_8 + a_2 \sum_{k=1}^{n} X_8 X_2 + a_3 \sum_{k=1}^{n} X_8 X_3 + ... + a_8 \sum_{k=1}^{n} X_8^2 = \sum_{k=1}^{n} X_8 ln(\frac{dh}{dt}) \tag{20e}$$

Note that if any of the constants a_1 to a_8 is assumed known, the amount of equations to be solved are reduced. Note also that the set of equations (20), will only have the coefficients a_j as unknowns and that X_j and $ln(dh/dt)$ will be found as a sum of n historical data points. Hence, a system of eight equations are to be solved for eight unknowns at most.

If the model in eq. (14) reflected the true rotary drilling process with a 100% accuracy, only eight historical data points n would be needed. In reality, this is not the case. Bourgoyne and Young found, through a sensitivity study of the multiple regression analysis, that both the number of historical data points n and the range of values of the drilling parameters X_1,...,X_8 were important in order to obtain reasonable estimates of a_j, $\forall j = 1, ..., 8$. They presented the following table

Parameter	**Minimum Range**	**Number of Parameters a_j**	**Minimum Number of Data Points n**
X_2	2000	8	30
X_3	15000	7	25
X_4	15000	6	20
X_5	0.40	5	15
X_6	0.50	4	10
X_7	0.20	3	7
X_8	0.50	2	4

Table 4: Minimum data ranges recommended by Bourgoyne and Young [1999]

Hence the more parameters that are assumed to be known, meaning fewer parameters to be found in the multiple regression analysis, the fewer historical data points can be used. The minimum range of the parameters X_2,...,X_8 is the difference between the smallest parameter value and the largest, which is used in the regression analysis. Bourgoyne and Young [1999] further argues that if the drilling parameters, X_j, are close to constant over the regression interval, then a_j are better off estimated from values acquired from past drilled wells in the area. For some applications, they had to combine results from more than one well to obtain reasonable ranges for X_j.

3.1.3 Determining $a_j,\ \forall j = 1, ..., 8$ Through Trust-Region Method

A dataset fulfilling the recommendations of table 4 may not always be obtainable. This may lead to meaningless drilling coefficients when using multiple regression. Suppose, for example, that the bit weight exponent, a_5, were negative or zero. This would imply that increasing the WOB would reduce the ROP or yield zero change in ROP, respectively. Reasonable ranges for the drilling coefficients are given in Bourgoyne et al. [1991] and can be seen in the following table

Drilling Coefficients	Description	Minimum	Maximum
a_1	Formation strength exponent	0.5	1.9
a_2	Normal compaction exponent	0.000001	0.0005
a_3	Under-compaction exponent	0.000001	0.0009
a_4	Pressure differential exponent	0.000001	0.0001
a_5	Bit weight exponent	0.5	2
a_6	Rotary speed exponent	0.4	1
a_7	Tooth wear exponent	0.3	1.5
a_8	Hydraulic exponent	0.3	0.6

Table 5: Drilling coefficients and their reasonable ranges as proposed by Bourgoyne et al. [1991]

In order to ensure that the drilling coefficients are within reasonable ranges, Bahari and Baradaran Seyed [2007] used a trust-region approach to estimate the coefficients. The trust-region method allows for upper and lower bounds on the coefficients to be specified. An explanation of the algorithm is given, with the notation and an elaboration of Nocedal and Wright [2006].

The basic idea of trust region algorithms is to use a constructed model function that is easy to evaluate and behaves similarly, in a small neighborhood, to the objective function one wants to minimize. Suppose one wants to minimize the objective function $f(x)$, which represents a curve-fitting problem to minimize the residuals as in subsection 3.1.2. The model function around an iterate x_k can then be defined, based on a Taylor-series expansion of f. Nocedal and Wright [2006] defines such a model function as

$$m_k(p) = f_k + g_k^T p + \frac{1}{2} p^T B_k p \tag{21}$$

where $f_k = f(x_k)$ is a scalar, $g_k = \nabla f(x_k)$ is the gradient vector, and B_k is a symmetric matrix of the true hessian $\nabla^2 f(x_k)$ or some approximation of it. Further, p is the step towards the minimizer.

Remember that this model function will only be a reasonable approximation within a small region. Hence, the algorithm must minimize the model function iteratively until it is close enough to a solution of $f(x)$. The minimization problem performed at each iterate is given as

$$\min_{p \in \mathbb{R}^n} m_k(p) = f_k + g_k^T p + \frac{1}{2} p^T B_k p \qquad \text{s.t.} \quad ||p|| \leq \Delta_k \tag{22}$$

where Δ_k is the region where the model is trusted to be an adequate approximation of $f(x_k)$, i.e. the trust region. Here $|| \cdot ||$ is the Euclidian norm. Therefore, the constraint in this minimization problem will ensure that the solution is within a ball of radius Δ_k.

At each iteration, the size of the trust-region radius should be evaluated. If the step, p, that solves eq. (22) produces a sufficient decrease in f, that is $f(x_k + p) < f(x_k)$, the trust region radius Δ_k can be increased.

However, if it yields an unsatisfactory decrease, Δ_k should be reduced (Moré and Sorensen [1983]). To evaluate the validity of the model function relative to the actual function, given the step p_k, the ratio ρ_k is defined as

$$\rho_k = \frac{\text{actual reduction}}{\text{predicted reduction}} = \frac{f(x_k) - f(x_k + p_k)}{m_k(0) - m_k(p_k)} \tag{23}$$

Note that a ratio ρ_k near one means a good correspondence between the modeled function and the actual function, suggesting that the model is a good approximation within the specified trust region. However, if ρ_k is close to zero or negative, the step should be rejected, and the trust region shrunk.

A general algorithm can then be given

Algorithm 1: Trust Region (as given by Nocedal and Wright [2006])

Given $\hat{\Delta} > 0$, $\Delta_0 \in (0, \hat{\Delta})$, and $\eta \in [0, \frac{1}{4})$:
for *0,1,2,...* **do**
  Solve eq. (22) to obtain p_k;
  Evaluate ρ_k from eq. (23);
  if $\rho_k < \frac{1}{4}$ **then**
    $\Delta_{k+1} = \frac{1}{4}\Delta_k$
  else
    if $\rho_k > \frac{3}{4}$ *and* $||p_k|| = \Delta_k$ **then**
      $\Delta_{k+1} = min(2\Delta_k, \hat{\Delta})$
    else
      $\Delta_{k+1} = \Delta_k$
  if $\rho_k > \eta$ **then**
    $x_{k+1} = x_k + p_k$
  else
    $x_{k+1} = x_k$

Here $\hat{\Delta}$ is a maximum value for the trust-region, Δ_0 is the initial trust-region, and η specifies a minimum decrease in f as a fraction of the modeled prediction decrease.

One can see that if the ratio ρ_k is less than $\frac{1}{4}$, suggesting a low correspondence between the modeled and actual function, the trust-region is shrunk. Note also how the trust-region is only increased if the ratio is above $\frac{3}{4}$, and if the step length $||p_k||$ equals the radius of the trust-region. There is no need to increase the trust-region if the step length is well within its borders, as it does not interfere with the progress of the algorithm (Nocedal and Wright [2006]).

4 Model Predictive Control in Performinator

As previously stated, the Autodriller is supposed to allow autonomous control of the drilling operation. The idea is to let the driller interactively select the drilling variable to be controlled and choosing its desired value. The MPC should then drive the chosen variable to its reference value while obeying limitations imposed by the application.

Kommedal [2020] presented four different operating modes; Rate of Penetration, Torque of Rotary Drilling Equipment, Weight on Bit, and Mud Pump Standpipe Pressure mode. Each with its corresponding control objectives and limitations. However, only the ROP mode is developed in this thesis and is thereby reproduced and elaborated in subsection 4.1. Further, subsection 4.2 describes how a model can be developed to be used as a predictive model in the MPC. Here subsection 4.2.1, 4.2.2, and a minor part of 4.2.4 is reproduced and elaborated, while the rest is new. Lastly, subsection 4.3 shows how the quadratic program problem were derived for the MPC.

4.1 Rate of Penetration Mode and its Limits

In this mode, the ROP should be driven to its desired value which is chosen by the driller. If a harder formation is encountered and the ROP is reduced, proper control actions should be taken to maintain the ROP. This may involve increasing the WOB by reducing the amount of braking by the drawworks and increasing the RPM of the top drive. However, the WOB and RPM should at all times be maintained within the system safety limits.

In addition to the safety limits set by the driller, other limits apply to the ROP mode specifically. Firstly, the ROP will be highly restricted in terms of maintaining adequate hole cleaning. If the ROP is too high, it will generate more cuttings than the mud can transport to the surface, resulting in downhole problems and increased NPT.

Another limiting factor of the generated cuttings, assuming that the circulated mud can clean the hole properly, is the rig surface's cuttings handling capacity. In a report with the title *handling of cuttings in areas with vulnerable benthic fauna* (Øfjord et al. [2012]), different technologies for handling cuttings are addressed, as well as the regulations stated by the Norwegian law. Here it is stated that cuttings generated from drilling using oil-based mud, which is the preferred method on Valhall, cannot be released back into the seawater. Consequentially, a drill cuttings handling system, is used on the rig surface on Valhall. When the drilled cuttings are returned to the rig surface, it is separated from the mud on shale-shakers, which is vibrating screens where the mud falls into mud pits. Next, in what is called a slurrification system, the cuttings are ground to a slushy mass and added lubricant oil before it is re-injected into "waste" wells, or *drilled cuttings re-injection wells*. The shale-shakers, the slurrification system, and the wells will all have capacity limits. In some cases, tanker ships are used to transport the cuttings to shore and will be restricted by their respective load capacity.

The flow inlet of the returned mud is directed to a header box, a small box mounted on the shale-shakers used to distribute the mud to the screens evenly. A gas-trap is found in the header box, where two gas sensors are present. One of these is an online infrared sensor measuring the gas development trend in the returned drilling fluid. Hence, it will determine if the gas is increasing, stable, or decreasing. The other gas sensor is a so-called chromatograph measurement. It yields a more precise measurement of the total gas percentage, with each measurement taking up to two minutes. These are crucial to limit the rig surface's gas levels and thereby provide safety to the personnel. Besides, there are two gas detectors located on the rig surface. If one of these sensors measure 20% of Lower Explosive Level (LEL), which is the minimum level of a gas required to enable combustion, an alarm is activated. If they both measure 20% of LEL, a production shutdown sequence is activated. Consequentially, it might be necessary to reduce the drilling fluid flow rate to limit the amount of gas returned, yielding limitation to the ROP.

The upper bound on the ROP can be delivered to the MPC by a product called *Max ROP*, developed by one of AkerBP's alliance partners Halliburton. This is a program which calculates

the maximum achievable ROP based on the different safety and operational conditions as presented.

4.2 Using B&Y-model for Setpoint Control in ROP Mode

This subsection aims to explain how the MPC can utilize the B&Y-model to control the ROP. It summarizes the development of a nonlinear model to be used in a nonlinear MPC suggested by Sui et al. [2013], but presents an alternative approach named Successive Linearization based MPC (SLMPC).

4.2.1 Moving Data Window Method for the Multiple Regression and Trust-Region Method

When the coefficients a_j, $\forall j = 1, ..., 8$ are calculated once, based on historical data from one particular section of the well, it might yield insufficient ROP predictions for the ongoing drilling operation. This due to the formation's many layers with varying geological properties, where different coefficients will best represent these differences. A moving-window strategy, illustrated in figure 4.1, was proposed by Sui et al. [2013] to only include the most recent historical data. This can be done by defining the data set at time t

$$\Phi(t) = \{\frac{dh}{dt}(t-N), ..., \frac{dh}{dt}(t-1), X_j(t-N), ..., X_j(t-1), \forall j = 1, ..., 8\} \tag{24}$$

with the fixed window size here being N. This will be the data set which will be used to calculate the coefficients $a_j(t)$ with the methods explained in subsection 3.1.2 and 3.1.3 . Then the next coefficients $a_j(t+1)$ is found using the data set $\Phi(t+1)$, $a_j(t+2)$ is found using the set $\Phi(t+2)$ and so on.

Behind the moving-window proposition lies the assumption that the most recent data will yield the best approximations of the current wellbore properties. Naturally, a varying sized window could yield even better approximations. For example, a smaller sized window could better represent the wellbore's current conditions if the formation is very nonuniform (changing rapidly with increasing depth). However, for the sake of simplicity, a fixed window size is assumed to be sufficient.

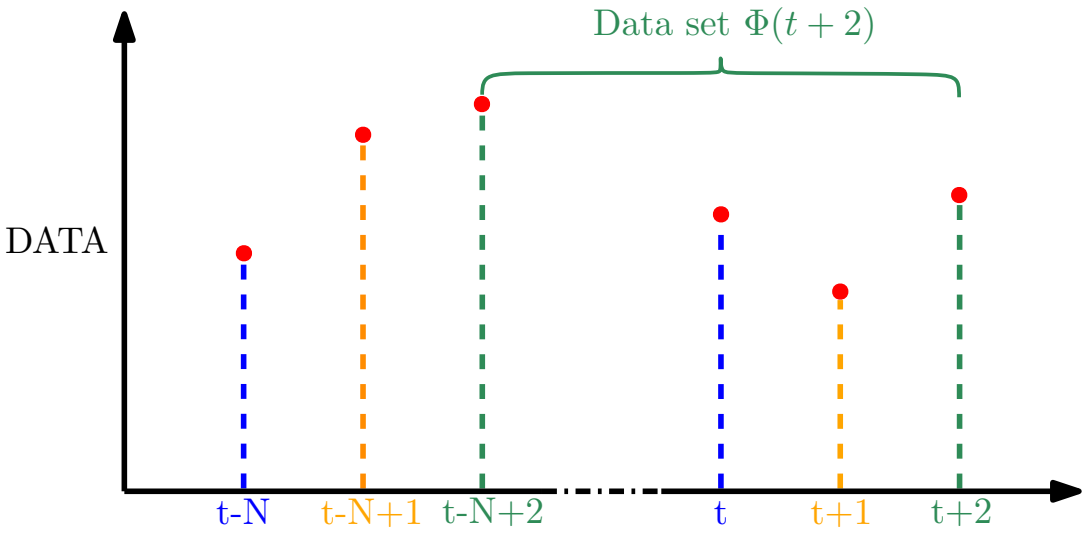


Figure 4.1: Fixed moving window. Red dots represents the data points.

4.2.2 ROP Calculation

First, one can define the error between the real ROP and the modeled ROP at time t to be

$$e(t) = \frac{dh}{dt}(t) - \widehat{ROP}(t) \tag{25}$$

Now using Eulers method on depth of bit h yields

$$h(t+1) = h(t) + \Delta t(\frac{dh}{dt}(t)) \tag{26}$$

where Δt is the time step length. Note that one can now use equation (25) to substitute for dh/dt in equation (26) resulting in the following equation

$$h(t+1) = h(t) + \Delta t(\widehat{ROP}(t) + e(t)) \tag{27}$$

This combined with an assumption that the error remains constant from t to $t+1$ allows for the following state space representation

$$x(t+1) = \begin{bmatrix} f_1(t) \\ f_2(t) \end{bmatrix} = \begin{bmatrix} h(t+1) \\ e(t+1) \end{bmatrix} = \begin{bmatrix} h(t) + \Delta t(\widehat{ROP}(t) + e(t)) \\ e(t) \end{bmatrix} \tag{28a}$$

$$y(t) = g(x(t), u(t), \alpha(t)) = \widehat{ROP}(t) + e(t) \tag{28b}$$

Note that $\widehat{ROP}(t)$ incorporates all the inputs $u(t)$ which should be used to control the ROP to the desired setpoint, aswell as the time dependant effects $a_j(t)$, $g_p(t)$, $p_{af}(t)$, $H(t)$, $q(t)$ and $\rho(t)$ as given in table 3. In short, the state space can be rewritten as

$$x(t+1) = f(x(t), u(t), \alpha(t)) \tag{29a}$$

$$y(t) = g(x(t), u(t), \alpha(t)) \tag{29b}$$

Lastly, the time dependant vectors are given as:

$$x(t) = \begin{bmatrix} h(t) \\ e(t) \end{bmatrix}, \qquad u(t) = \begin{bmatrix} w(t) \\ r(t) \end{bmatrix} \tag{30a}$$

$$\alpha(t) = [a_j(t), g_p(t), p_{af}(t), H(t), q(t), \rho(t)] \tag{30b}$$

assuming that only the WOB, $w(t)$, and the RPM, $r(t)$, is controllable variables. A possible extension to this formulation is to include the flow rate, $q(t)$, in $u(t)$, allowing yet another MV for the MPC. It should be noted that Sui et al. [2013] includes a disturbance term $d_e(t)$, associated with fluctuations in e, in the formulation of $f_2(t)$ in state space given in (28). In this thesis it is assumed to be incorporated in e as process noise, and is handled by the Extended Kalman Filter described in subsection 4.2.4.

4.2.3 Successive Linearization Based MPC

Sui et al. [2013] formulated a nonlinear optimization problem and solved it using nonlinear techniques. An alternative approach is to linearize the nonlinear functions of the equations (29) in order to formulate a quadratic optimization problem that satisfies the terms of convexity given in subsection 2.2.3.

Linearization is often used to bypass the mathematical and computational complexity of nonlinearity. When linearizing the nonlinear model of the process around the current states, at each time step, it is called successive linearization. Seki et al. [2004] used this technique to linearize a nonlinear model before solving the optimization problem to control a chemical reactor. Another example is Zhakatayev et al. [2017] which applied this technique to control variable stiffness actuated robots. They showed that the SLMPC provided similar performance as the nonlinear MPC while requiring less computation time.

A linear model can be used to approximate the nonlinear model for a small region around an operating point. A way of doing such a linearization is given in Hovd [2009]. Consider a nonlinear function $p(x), \quad x \in \mathbb{R}^n$. The change in $p(x)$ due to a small change in x can be approximated by the Taylor series expansion

$$p(x) = p(x_{op} + \delta x) \approx p(x_{op}) + \left.\frac{\partial p}{\partial x}\right|_{x=x_{op}} \delta x + \underbrace{\frac{1}{2}\delta x^{\intercal} \left.\frac{\partial^2 p}{\partial x^2}\right|_{x=x_{op}} \delta x + \dots}_{\text{Higher-order terms}} \tag{31}$$

where x_{op} is the point of linearization, called the operating point, and $\delta x = x - x_{op}$ represents the deviation from the operating point. Further, $|_{x=x_{op}}$ means that the derivatives are evaluated at the operating point. When the deviation δx is sufficiently small, the higher-order terms will be negligible and can be ignored. This yields the linear approximation

$$p(x) \approx p(x_{op}) + \left.\frac{\partial p}{\partial x}\right|_{x=x_{op}} (x - x_{op}) \tag{32}$$

This can be extended to include the input u, and applied to the nonlinear state space equation given in (29)

$$\begin{aligned} x(t+1) = f(x(t), u(t), \alpha(t)) \quad &\approx f(\ell_{op}) + \underbrace{\left.\frac{\partial f}{\partial x}\right|_{\ell=\ell_{op}}}_{A_t \in \mathbb{R}^{n_x \times n_x}} \delta x + \underbrace{\left.\frac{\partial f}{\partial u}\right|_{\ell=\ell_{op}}}_{B_t \in \mathbb{R}^{n_x \times n_u}} \delta u \\ y(t) = g(x(t), u(t), \alpha(t)) \quad &\approx g(\ell_{op}) + \underbrace{\left.\frac{\partial g}{\partial x}\right|_{\ell=\ell_{op}}}_{C_t \in \mathbb{R}^{n_y \times n_x}} \delta x + \underbrace{\left.\frac{\partial g}{\partial u}\right|_{\ell=\ell_{op}}}_{D_t \in \mathbb{R}^{n_y \times n_u}} \delta u \end{aligned} \tag{33}$$

Recall that $\alpha(t)$ only yields the parameters at time t and should therefore be evaluated as constants. Note that n_x, n_u, and n_y represents the dimensions of the states, inputs, and outputs, respectively. Further, the point of linearization is given by

$$\ell = \begin{bmatrix} x \\ u \end{bmatrix} = \begin{bmatrix} x_{op} \\ u_{op} \end{bmatrix} = \ell_{op} \tag{34}$$

This yields the linearized model

$$\begin{aligned} x(t+1) &= f(\ell_{op}) + A_t \delta x(t) + B_t \delta u(t) \\ y(t) &= g(\ell_{op}) + C_t \delta x(t) + D_t \delta u(t) \end{aligned} \tag{35}$$

with A_t, B_t, C_t, D_t given as

$$A_t = \left[\begin{array}{c:c} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} \\ \hdashline \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} \end{array}\right]\Bigg|_{\ell=\ell_{op}} = \left[\begin{array}{c:c} 1 + \Delta t \cdot \widehat{ROP}(t) \cdot \left(a_4 \cdot (g_p - p_{af}) - a_2 + 0.69 \cdot a_3 \cdot (\frac{g_p - g_{p0}}{x_1^{0.31}})\right) & \Delta t \\ \hdashline 0 & 1 \end{array}\right]\Bigg|_{\ell=\ell_{op}}$$

$$B_t = \left[\begin{array}{c:c} \frac{\partial f_1}{\partial u_1} & \frac{\partial f_1}{\partial u_2} \\ \hdashline \frac{\partial f_2}{\partial u_1} & \frac{\partial f_2}{\partial u_2} \end{array}\right]\Bigg|_{\ell=\ell_{op}} = \left[\begin{array}{c:c} \Delta t \cdot \widehat{ROP}(t) \cdot \left(\frac{a_5}{u_1 - w_0}\right) & \Delta t \cdot \widehat{ROP}(t) \cdot \left(\frac{a_6}{u_2}\right) \\ \hdashline 0 & 0 \end{array}\right]\Bigg|_{\ell=\ell_{op}}$$

$$C_t = \left[\begin{array}{c:c} \frac{\partial g}{\partial x_1} & \frac{\partial g}{\partial x_2} \end{array}\right]\Bigg|_{\ell=\ell_{op}} = \left[\begin{array}{c:c} \widehat{ROP}(t) \cdot \left(a_4 \cdot (g_p - p_{af}) - a_2 + 0.69 \cdot a_3 \cdot (\frac{g_p - g_{p0}}{x_1^{0.31}})\right) & 1 \end{array}\right]\Bigg|_{\ell=\ell_{op}}$$

$$D_t = \left[\begin{array}{c:c} \frac{\partial g}{\partial u_1} & \frac{\partial g}{\partial u_2} \end{array}\right]\Bigg|_{\ell=\ell_{op}} = \left[\begin{array}{c:c} \widehat{ROP}(t) \cdot \left(\frac{a_5}{u_1 - w_0}\right) & \widehat{ROP}(t) \cdot \left(\frac{a_6}{u_2}\right) \end{array}\right]\Bigg|_{\ell=\ell_{op}}$$

with f_1, f_2 and g as defined in eq. (28). Using the linearized state space, from equation (35), to formulate the MPC's optimization problem, at each time step, concludes the SLMPC strategy.

4.2.4 Extended Kalman Filter for State Estimation

Sui et al. [2013] showed that the performance of the MPC using a B&Y could be further improved by using observer methods to estimate the states. This since the B&Y-model is not a perfect representation of the real world. State observer methods are used to estimate the states of the system when the states can not be directly measured at each time step.

One such observer, meant for nonlinear systems, is the Extended Kalman Filter (EKF). It is an extension of the regular linear Kalman filter, developed by Kalman [1960], but where the nonlinear system is linearized around the current state estimate. Fossen [2011] was used as a basis for the implementation of the EKF, and is shown in the following algorithm with a slightly different notation

Algorithm 2: Extended Kalman Filter (Adapted from Fossen [2011])

Initial conditions: $\hat{x}^-(0) = x_0$, $\hat{P}^-(0) = P_0$, $Q_d = Q_d^T > 0$, $R_d = R_d^T > 0$

1. Take measurement if available
$y(t) = [h_{meas} \quad e_{meas}]^T$

2. Calculate the Kalman gain matrix
$K(t) = \hat{P}^-(t)C_d^T \left[C_d\hat{P}^-(t)C_d^T + R_d\right]^{-1}$

3. Correct the state and covariance error estimate
$\hat{x}(t) = \hat{x}^-(t) + K(t)\left[y(t) - C_d\hat{x}^-(t)\right]$

$\hat{P}(t) = \left[I - K(t)C_d\right]\hat{P}^-(t)\left[I - K(t)C_d\right]^T + K(t)R_dK^T(t)$

4. Predict next state estimate and covariance error
$\hat{x}^-(t+1) = f(\hat{x}(t), u(t), \alpha(t))$

$\hat{P}^-(t+1) = A_d\hat{P}(t)A_d^T + Q_d$

Here $\hat{x}^-(t)$ and $\hat{P}^-(t)$ is the state and covariance matrices before the measurement correction, while $\hat{x}(t)$ and $\hat{P}(t)$ is after the correction. Q_d is the covariance matrix for the process noise, and R_d the covariance matrix of the measurement noise. Kalman gain matrix is given by $K(t)$. For the measurement e_{meas}, being the difference between the real ROP and the modeled ROP, it should be emphasized that only the real ROP is measured while the modeled ROP from last time step is used. Further, A_d has the same structure as A_t from equation (35) but is linearized around the state estimate, while $C_d = diag([1 \quad 1])$.

The idea of the filter is to estimate the states recursively and correct the estimation with measurements. Thus, it is an iterative algorithm, where t is incremented after step 4 along with the progress of the simulation. If there are no new measurements at a particular time step, it will predict the following states using only step 4 in the algorithm without any measurement corrections.

The numeric size of the Kalman gain matrix $K(t)$ weighs the importance of the measurement compared to the state prediction. An intuitive interpretation can be made by rewriting the Kalman gain calculation in the following way

$$K(t) = \frac{\hat{P}^-(t)C_d^T}{C_d\hat{P}^-(t)C_d^T + R_d} \tag{36}$$

It is important to note that this is a misuse of notation and is not mathematically correct as one cannot perform normal division on matrices. However, it emphasizes the role of the covariance matrix for measurement noise R_d. If the measurement is very noisy, hence large R_d, the Kalman gain will be small. As shown in step 3 of the algorithm, the state will then be updated with the

last predicted value plus only a small contribution of the measurement correction. On the other hand, if the measurement noise is minimal, $K(t)$ will be significant and yield a higher measurement correction in this step. Further, the covariance matrix $\hat{P}(t)$ represents the uncertainty in the state estimate. As seen in step 4, Q_d feeds directly into the prediction $\hat{P}(t+1)$, as it represents the process noise in the state estimation model. A large Q_d will thus yield a large covariance matrix $\hat{P}(t)$ and cause a strong contribution from the measured values on the state estimates.

4.3 Deriving the Quadratic Program Problem

This subsection aims to describe the derivation of the quadratic program problem used in the MPC. Zhakatayev et al. [2017] provided a thorough derivation of the problem and was used as a basis. An augmentation is also done to include slack variables and is based on the formulations found in Perne et al. [2018]. For ease of notation, the state-space matrices, from equation (35), are hereafter denoted as

$$\begin{aligned} x_{k+1} &= f(\ell_{op}) + A_t(x_k - x_{op}) + B_t(u_k - u_{op}) \\ &= A_t x_k + B_t u_k + F \\ y_k &= g(\ell_{op}) + C_t(x_k - x_{op}) + D_t(u_k - u_{op}) \\ &= C_t x_k + D_t u_k + G \end{aligned} \tag{37}$$

where the subscript k denotes the kth step into "the future" in the MPC, and t is the time step of the drilling operation. Further, all the constant terms are collected into the matrices $F \in \mathbb{R}^{n_x}$ and $G \in \mathbb{R}^{n_y}$.

4.3.1 Evolution of the predicted ROP

Now one can find the evolution of the predicted ROP, given by the set $Y = \{y_0, y_1, ..., y_{N_p-1}\}$, where N_p is the prediction horizon by

$$\begin{aligned} y_0 &= C_t x_0 + D_t u_0 + G \\ y_1 &= C_t x_1 + D_t u_1 + G \\ &= C_t \underbrace{(A_t x_0 + B_t u_0 + F)}_{x_1} + D_t u_1 + G \\ &= C_t A_t x_0 + C_t B_t u_0 + D_t u_1 + C_t F + G \\ y_2 &= C_t x_2 + D_t u_2 + G \\ &= C_t(A_t x_1 + B_t u_1 + F) + D_t u_2 + G \\ &= C_t(A_t(A_t x_0 + B_t u_0 + F) + B_t u_1 + F) + D_t u_2 + G \\ &= C_t A_t^2 x_0 + C_t A_t B_t u_0 + C_t B_t u_1 + D_t u_2 + C_t A_t F + C_t F + G \\ &\vdots \\ y_{N_p-1} &= C_t A_t^{N_p-1} x_0 + C_t A_t^{N_p-2} B_t u_0 + C_t A_t^{N_p-3} B_t u_1 + \cdots + C_t B_t u_{N_p-2} + D_t u_{N_p-1} \\ &\quad + C_t A_t^{N_p-2} F + C_t A_t^{N_p-3} F + \cdots + C_t F + G \end{aligned} \tag{38}$$

Note that every x_k is replaced with its corresponding prediction equation $x_k = A_t x_{k-1} + B_t u_{k-1} + F$. This is repeated until x_0 and u_0 appears. Further, one can define the tracking error $y_k^e \triangleq y_k - y_k^r$, where the evolution of the tracking error is given by the set $Y^e = \{y_0^e, y_1^e, ..., y_{N_p-1}^e\}$, and the ROP reference is given by the set $Y^r = \{y_0^r, y_1^r, ..., y_{N_p-1}^r\}$.

The evolution can be compactly formulated, in terms of the tracking error, with matrices as

$$\underbrace{\begin{bmatrix} y_0^e \\ y_1^e \\ y_2^e \\ \vdots \\ y_{N_p-1}^e \end{bmatrix}}_{Y^e \in \mathbb{R}^{N_p \cdot n_y}} = \underbrace{\begin{bmatrix} C_t \\ C_t A_t \\ C_t A_t^2 \\ \vdots \\ C_t A_t^{N_p-1} \end{bmatrix}}_{P \in \mathbb{R}^{(N_p \cdot n_y) \times n_x}} x_0 + \underbrace{\begin{bmatrix} D_t & 0 & 0 & \cdots & 0 \\ C_t B_t & D_t & 0 & & \vdots \\ C_t A_t B_t & C_t B_t & D_t & & \vdots \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ C_t A_t^{N_p-2} B_t & C_t A_t^{N_p-3} B_t & C_t A_t^{N_p-4} B_t & \cdots & D_t \end{bmatrix}}_{H \in \mathbb{R}^{(N_p \cdot n_y) \times (N_p \cdot n_u)}} \underbrace{\begin{bmatrix} u_0 \\ u_1 \\ u_2 \\ \vdots \\ u_{N_p-1} \end{bmatrix}}_{U \in \mathbb{R}^{N_p \cdot n_u}}$$

$$+ \underbrace{\underbrace{\begin{bmatrix} 0 \\ C_t \\ C_t(I + A_t) \\ \vdots \\ C_t(I + \sum_{i=1}^{N_p-2} A_t^i) \end{bmatrix}}_{E \in \mathbb{R}^{(N_p \cdot n_y) \times n_x}} F + \underbrace{\begin{bmatrix} I \\ I \\ I \\ \vdots \\ I \end{bmatrix}}_{\mathbf{1} \in \mathbb{R}^{N_p \cdot n_y}} G - \underbrace{\begin{bmatrix} y_0^r \\ y_1^r \\ y_2^r \\ \vdots \\ y_{N_p-1}^r \end{bmatrix}}_{Y^r \in \mathbb{R}^{N_p \cdot n_y}}}_{M \in \mathbb{R}^{(N_p \cdot n_y)}}$$

Now all the constant terms can be collected in the matrix M, so that the evolution of the error is given by

$$\begin{aligned} Y^e &= Px_0 + HU + EF + \mathbf{1}G - Y^r \\ Y^e &= Px_0 + HU + M \end{aligned} \tag{39}$$

4.3.2 Unconstrained Control Objective

The control objective is to minimize the deviation between the estimated ROP and the desired ROP, hence minimizing Y^e. This should be done by choosing the proper control actions, U, over the prediction horizon N_p. This problem can be expressed as

$$\min_U J = \frac{1}{2} Y^{eT} Q Y^e + \frac{1}{2} U^{eT} R U^e \tag{40}$$

where J denotes the cost function to be minimized, and the matrix $U^e \in \mathbb{R}^{N_p \cdot n_u}$ is the error between the predicted inputs and the desired inputs over the prediction horizon. Often it will be desirable to achieve the control objective while minimizing the amount of input resources needed to do so. However, when considering the drilling process, too low RPM can cause poor hole cleaning and downhole problems. Setpoints on the inputs can therefore be specified to make sure that the input is minimized around desired values. Further, the weighting matrices $Q \in \mathbb{R}^{(N_p \cdot n_y) \times (N_p \cdot n_y)}$ and $R \in \mathbb{R}^{(N_p \cdot n_u) \times (N_p \cdot n_u)}$ are diagonal and positive definite.

By substituting Y^e from equation (39) into equation (40) one can derive the following expression for the cost function

$$\begin{aligned} J &= \frac{1}{2}(Px_0 + HU + M)^T Q (Px_0 + HU + M) + \frac{1}{2}(U - U^r)^T R (U - U^r) \\ &= \cancelto{0}{\frac{1}{2} x_0^T P^T Q P x_0} + \frac{1}{2} x_0^T P^T Q H U + \cdots + \frac{1}{2} U^T R U + \cdots + \cancelto{0}{\frac{1}{2} U^{rT} R U^r} \\ &= \frac{1}{2} U^T \underbrace{(H^T Q H + R)}_{\tilde{H} \in \mathbb{R}^{(N_p \cdot n_u) \times (N_p \cdot n_u)}} U + \underbrace{(x_0^T P^T Q H + M^T Q H - U^{rT} R)}_{\tilde{c}^T \in \mathbb{R}^{1 \times (N_p \cdot n_u)}} U \end{aligned} \tag{41}$$

Note that the terms that are not dependant on the minimization variable, U, will not affect the optimization problem and can be removed. So the unconstrained control problem can be formulated as

$$\min_U J = \frac{1}{2}U^T \tilde{H} U + \tilde{c}^T U \tag{42}$$

4.3.3 Adding Hard Constraints

As earlier mentioned, constraints can be added to ensure safe operation in terms of safety limits and physical limitations. Upper and lower limits on the WOB and RPM can be added as

$$\begin{bmatrix} \underline{WOB} \\ \underline{RPM} \\ \vdots \\ \underline{WOB} \\ \underline{RPM} \end{bmatrix} \leq U \leq \begin{bmatrix} \overline{WOB} \\ \overline{RPM} \\ \vdots \\ \overline{WOB} \\ \overline{RPM} \end{bmatrix} \tag{43}$$

where $\overline{WOB}$ and $\underline{WOB}$ are the upper and lower limits of WOB, respectively (and similar for RPM). Further, there should be limitations on how fast the WOB and RPM can increase, from one time step to another.

The evolution of the control vector over the prediction horizon can be described with

$$\begin{aligned} u_0 &= u_{-1} + \delta u_1 \\ u_1 &= u_0 + \delta u_2 \\ &= u_{-1} + \delta u_1 + \delta u_2 \\ &\vdots \\ u_{N_p-1} &= u_{-1} + \delta u_1 + \delta u_2 + \cdots + \delta u_{N_p} \end{aligned} \tag{44}$$

where u_{-1} is a known initial input and δu_k is the input increment from u_{k-2} to u_{k-1}. This can be formulated with the matrices

$$\underbrace{\begin{bmatrix} u_0 \\ u_1 \\ u_2 \\ \vdots \\ u_{N_p-1} \end{bmatrix}}_{U} = \underbrace{\begin{bmatrix} u_{-1} \\ u_{-1} \\ u_{-1} \\ \vdots \\ u_{-1} \end{bmatrix}}_{U_{-1} \in \mathbb{R}^{N_p \cdot n_u}} + \underbrace{\begin{bmatrix} I & 0 & \cdots & 0 \\ I & I & \cdots & \vdots \\ \vdots & \vdots & \ddots & \vdots \\ I & I & \cdots & I \end{bmatrix}}_{\Delta \in \mathbb{R}^{(N_p \cdot n_u) \times (N_p \cdot n_u)}} \underbrace{\begin{bmatrix} \delta u_1 \\ \delta u_2 \\ \delta u_3 \\ \vdots \\ \delta u_{N_p} \end{bmatrix}}_{\delta U \in \mathbb{R}^{N_p \cdot n_u}} \tag{45}$$

with Δ being a lower triangular matrix with elements of $\{I, 0\} \in \mathbb{R}^{n_u \times n_u}$. Now constraints on the rate of change δU can be specified in terms of U by

$$\begin{aligned} U &= U_{-1} + \Delta \delta U \\ \delta U &= \Delta^{-1}(U - U_{-1}) \\ &\Downarrow \ \textit{Add bounds and formulate in terms of } U \\ \underline{\delta U} + \Delta^{-1} U_{-1} &\leq \Delta^{-1} U \leq \overline{\delta U} + \Delta^{-1} U_{-1} \\ &\Downarrow \\ A_1 U &\leq b_1 \end{aligned} \tag{46}$$

where the upper and lower bounds on the rate of change is given by $\overline{\delta U}$ and $\underline{\delta U}$, respectively.

Further, the matrices $A_1 \in \mathbb{R}^{(2\cdot N_p \cdot n_u)\times(N_p\cdot n_u)}$ and $b_1 \in \mathbb{R}^{(2\cdot N_p\cdot n_u)}$ is given by

$$A_1 = \begin{bmatrix} \Delta^{-1} \\ -\Delta^{-1} \end{bmatrix}, \qquad b_1 = \begin{bmatrix} \overline{\delta U} + \Delta^{-1}U_{-1} \\ -\underline{\delta U} - \Delta^{-1}U_{-1} \end{bmatrix} \tag{47}$$

4.3.4 Augmented Formulation with Soft Constraints

A recommendation from the drill bit producer is to avoid running both the WOB and RPM on upper limits simultaneously, as this will reduce its lifespan. One way to achieve this is to add soft constraints with more strict upper limits on the inputs. This way, a violation of the soft constraints is allowed if the corresponding increase in ROP is beneficial based on the chosen weights. Consider the constraints

$$U \leq \begin{bmatrix} \overline{WOB} - WOB_M + \varepsilon_1^w \\ \overline{RPM} - RPM_M + \varepsilon_1^r \\ \vdots \\ \overline{WOB} - WOB_M + \varepsilon_{N_p}^w \\ \overline{RPM} - RPM_M + \varepsilon_{N_p}^r \end{bmatrix} \tag{48}$$

where WOB_M and RPM_M are margins in order to reduce the bit wear, and ε_k^w and ε_k^r is the slack variable for WOB and RPM at time step k, respectively. By augmenting the optimization problem to include the slack variable one can formulate this constraint as

$$\underbrace{\begin{bmatrix} I & 0 & \cdots & 0 & -I & 0 & \cdots & 0 \\ 0 & I & & \vdots & 0 & -I & & \vdots \\ \vdots & & \ddots & 0 & \vdots & & \ddots & 0 \\ 0 & \cdots & 0 & I & 0 & \cdots & 0 & -I \end{bmatrix}}_{\substack{I_1 \in \mathbb{R}^{(N_p\cdot n_u)\times(N_p\cdot n_u)} \\ -I_1 \in \mathbb{R}^{(N_p\cdot n_u)\times(N_p\cdot n_u)}}} \begin{bmatrix} U \\ \varepsilon_u \end{bmatrix} \leq \underbrace{\begin{bmatrix} \overline{WOB} - WOB_M \\ \overline{RPM} - RPM_M \\ \vdots \\ \overline{WOB} - WOB_M \\ \overline{RPM} - RPM_M \end{bmatrix}}_{b_2 \in \mathbb{R}^{(N_p\cdot n_u)}} \tag{49}$$

where $\varepsilon_u = [\varepsilon_1^w \quad \varepsilon_1^r \quad ... \quad \varepsilon_{N_p}^w \quad \varepsilon_{N_p}^r]^T$ is the vector of slack variables on u.

Next, in order to avoid wear and tear on the top drive and drawworks due to aggressive actuation, soft constraints on the inputs rate of change is added. This can be done using the same procedure as in equations (46) and (49), but without the upper and lower bounds and by adding slack variables

$$\begin{gathered} \delta U = \Delta^{-1}(U - U_{-1}) \\ \Downarrow \ \textit{Add slack and formulate in terms of } U \\ \Delta^{-1}U_{-1} + \varepsilon_{\delta u} \leq \Delta^{-1}U \leq \Delta^{-1}U_{-1} + \varepsilon_{\delta u} \\ \Downarrow \\ \underbrace{\begin{bmatrix} \Delta^{-1} \\ -\Delta^{-1} \end{bmatrix}}_{A_1} \quad \underbrace{\begin{bmatrix} -I_1 \\ -I_1 \end{bmatrix}}_{-I_{\delta u}} \begin{bmatrix} U \\ \varepsilon_{\delta u} \end{bmatrix} \leq \underbrace{\begin{bmatrix} \Delta^{-1}U_{-1} \\ -\Delta^{-1}U_{-1} \end{bmatrix}}_{b_3 \in \mathbb{R}^{(2\cdot N_p\cdot n_u)}} \end{gathered} \tag{50}$$

where $\varepsilon_{\delta u} = [\varepsilon_1^{\delta w} \quad \varepsilon_1^{\delta r} \quad \cdots \quad \varepsilon_{N_p}^{\delta w} \quad \varepsilon_{N_p}^{\delta r}]^T$ is the vector of slack variables on δu, $I_{\delta u} \in \mathbb{R}^{(2\cdot N_p\cdot n_u)\times(N_p\cdot n_u)}$, and the other matrices as defined above.

Lastly, constraint on the modeled ROP must be considered. Y as given in subsection 4.3.1 should be limited due to considerations presented in subsection 4.1. Further, it should be added as a soft constraint in order to avoid infeasibility due to conflicting constraints (further described in discussion (section 8)). Upper bound of Y can be written in terms of U as following

$$\begin{aligned} Y &= Px_0 + HU + EF + \mathbf{1}G \leq \overline{ROP} + \boldsymbol{\varepsilon}_y \\ &\Downarrow \\ HU &\leq \underbrace{\overline{ROP} - Px_0 - EF - \mathbf{1}G}_{b_4 \in \mathbb{R}^{(N_p \cdot n_y)}} + \boldsymbol{\varepsilon}_y \end{aligned} \tag{51}$$

where the vectors $\overline{ROP} = [\overline{y}_0 \quad \overline{y}_1 \quad ... \quad \overline{y}_{N_p-1}]^T$ and $\boldsymbol{\varepsilon}_y = [\boldsymbol{\varepsilon}_1^y \quad \boldsymbol{\varepsilon}_2^y \quad ... \quad \boldsymbol{\varepsilon}_{N_p}^y]^T$ yields the upper bound of ROP and the slack variables for each time step, respectively. Now one can augment the optimization problem once again to include the new slack variables, and equation (51) can be reformulated in the same way as equation (49) and (50) including U, $\boldsymbol{\varepsilon}_u$, $\boldsymbol{\varepsilon}_{\delta u}$, and $\boldsymbol{\varepsilon}_y$. Note that the slack variables must be constrained to be positive.

A final augmented formulation, including all constraints, can then be given as

$$\min_{U, \boldsymbol{\varepsilon}_u, \boldsymbol{\varepsilon}_{\delta u}, \boldsymbol{\varepsilon}_y} J = \frac{1}{2} \begin{bmatrix} U \\ \boldsymbol{\varepsilon}_u \\ \boldsymbol{\varepsilon}_{\delta u} \\ \boldsymbol{\varepsilon}_y \end{bmatrix}^T \underbrace{\begin{bmatrix} \tilde{H} & 0 & 0 & 0 \\ 0 & S_u & 0 & 0 \\ 0 & 0 & S_{\delta u} & 0 \\ 0 & 0 & 0 & S_y \end{bmatrix}}_{\Gamma} \begin{bmatrix} U \\ \boldsymbol{\varepsilon}_u \\ \boldsymbol{\varepsilon}_{\delta u} \\ \boldsymbol{\varepsilon}_y \end{bmatrix} + \underbrace{\begin{bmatrix} \tilde{c} \\ \rho_u \\ \rho_{\delta u} \\ \rho_y \end{bmatrix}^T}_{Z^T} \begin{bmatrix} U \\ \boldsymbol{\varepsilon}_u \\ \boldsymbol{\varepsilon}_{\delta u} \\ \boldsymbol{\varepsilon}_y \end{bmatrix}$$

$$s.t. \quad \begin{bmatrix} A_1 & 0 & 0 & 0 \\ I_1 & -I_1 & 0 & 0 \\ A_1 & 0 & -I_{\delta u} & 0 \\ H & 0 & 0 & -I_2 \end{bmatrix} \begin{bmatrix} U \\ \boldsymbol{\varepsilon}_u \\ \boldsymbol{\varepsilon}_{\delta u} \\ \boldsymbol{\varepsilon}_y \end{bmatrix} \leq \begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ b_4 \end{bmatrix} \tag{52}$$

$$\begin{bmatrix} \underline{U} \\ 0 \\ 0 \\ 0 \end{bmatrix} \leq \begin{bmatrix} U \\ \boldsymbol{\varepsilon}_u \\ \boldsymbol{\varepsilon}_{\delta u} \\ \boldsymbol{\varepsilon}_y \end{bmatrix} \leq \begin{bmatrix} \overline{U} \\ \infty \\ \infty \\ \infty \end{bmatrix}$$

where $S_u, S_{\delta u}, S_y$ is the quadratic weighting matrices and $\rho_u, \rho_{\delta u}, \rho_y$ is the linear weighting vectors. Note that either the quadratic or the linear term can be excluded by setting its corresponding weighting matrix to zero. Further, the upper and lower bounds on the decision variables are formulated explicitly on the form $lb \leq z \leq ub$. If the chosen optimization solver does not accept such constraints, these can be reformulated to be incorporated in the inequality constraints.

5 Software and Drilling Data Handling

5.1 Software

The program was developed and implemented in Matlab. Matlab is a programming platform that uses a language that lets one express matrix and array mathematics directly. Data analysis, developing algorithms, creating models and desktop application is amongst some of its capabilities (MathWorks inc.). It also has the benefit of having a vast amount of valuable toolboxes and functions which are well documented. The toolboxes used in the program developed in this thesis was *Optimization Toolbox, Signal Processing Toolbox, Mapping Toolbox,* and *Symbolic Math Toolbox.*

Further, Microsoft Excel were used to import and format .xml-files, as described in subsection 5.2.1. Further, it is a helpful tool to structure large data-files and to do quick calculations over repetitive data.

Lastly, the open source editor Notepad++[3] were used to structure .xml-files and make them readable as shown in figure 5.1 (a).

5.2 Data Files

AkerBP delivered the data files in various formats. Data regarding the wellbore geometry and BHA specifications were given in .xml-files, information regarding the pressure gradients were given in .xlsx-files, and lastly, the drilling data given by .csv-files.

5.2.1 Wellbore Geometry, BHA Specifications, and Drilling Windows

In order to calculate the wellbore trajectory, mathematical methods presented in subsection 2.1.5 were used to calculate the paths between survey points performed at regular depth intervals. These survey points were provided by AkerBP as .xml-files. XML stands for Extensible Markup Language. It is a text-based format that can represent structured information such as data (World Wide Web Consortium (W3C)). It is well suited to share information between computers and humans, as it is readable for both. An example of how the data is structured in .xml-files can be seen in figure 5.1 (a). It specifically shows the data at a survey point at a measured depth of $2869.9m$.

Information regarding the drill bit and other BHA equipment was also provided as .xml-files. Hence, drilling parameters required by the application could be read directly from these files and applied to the Matlab program. Note that the fractional tooth height worn away, H, was not included in the model (H was set to zero), assuming that the bit wear was insignificant. This since no information regarding the bits' current state was obtainable.

In order to determine the immediate depth of the bit at any time, the depths and angles at the survey points had to be made available to the Matlab program. This was done by using the import tools in Microsoft Excel to create a table in .xlsx-file format. Further, all unnecessary data were deleted to obtain a table, as shown in figure 5.1 (b). Note that the figure only shows a small section of all the survey points. The .xlsx-files are easily read by the Matlab application by using the function `readtable(filename,opts)`.

The drilling windows, as shown in figure 2.3, were also provided as .xlsx-files and could be read into Matlab in the same way. It could then be used as a lookup table to find the pressure gradients for a given depth. Note that the pressure gradients in the provided file were given in terms of MD, while these plots are normally expressed in terms of TVD.

[3]Further description of notepad++ found in: https://notepad-plus-plus.org/

```
<md uom="m">2869.9</md>
<tvd uom="m">2330.08</tvd>
<incl uom="dega">50</incl>
<azi uom="dega">280.72</azi>
<dispNs uom="m">394.45</dispNs>
<dispEw uom="m">-964.46</dispEw>
<dls uom="dega/30m">4.03</dls>
```

(a) Data extracted from one survey point in the provided .xml-file.

MD	UNIT	TVD	UNIT	INCL. ANGLE	UNIT	AZIM. ANGLE	UNIT
2814	m	2293,7	m	49,2	dega	272	dega
2842,1	m	2312,06	m	49,3	dega	275,91	dega
2869,9	m	2330,08	m	50	dega	280,72	dega
2897,8	m	2347,84	m	50,9	dega	285,45	dega
2925,7	m	2365,32	m	51,5	dega	287,69	dega
2953,6	m	2382,53	m	52,4	dega	291,63	dega
2982	m	2399,72	m	53,1	dega	292,5	dega
3009,9	m	2415,99	m	55,5	dega	293,78	dega
3038	m	2431,3	m	58,4	dega	294,65	dega

(b) Data from several survey points as presented in an .xlsx-file. The framed row is the one given in the .xml-example from (a).

Figure 5.1: Wellbore Geometry Data

5.2.2 Drilling Data

The rest of the drilling data were delivered in .csv-files [4]. CSV is short for Comma-Separated Values. They are typically text files that contain data separated by commas or other delimiters. In this way, large datasets can be exported from one application to another using .csv-files.

How the provided .csv-files were structured can be seen in figure 5.2. Note that the figure only shows a small section of the file. One can see that the first two lines are the title and units of the different data provided in the file. Further, all the data is provided with a timestamp given as the first value of the comma-separated values. While large .csv-files can be hard to read with the naked eye, they can easily be read and sorted in tables by programs like Matlab in the same way as for the .xlsx-files.

How the drilling data were imported to Matlab is shown in appendix B.2. `SimulateAll` specifies whether the whole imported data set should be used, or only a specified interval. If only a specified interval is to be used, the time interval specified by `FromDate`, `ToDate`, `FromTime`, and `ToTime` is found in the imported data set. To do this, the function `FindTimeIndexes(WellData,FromDate,FromTime,ToDate,ToTime)` were made and is shown in appendix B.3. Further, the Matlab function `extractBetween(str,startPos,endPos)` is used to extract all the time data in the interval on a specified format, converted to datetime. Lastly, the well data and time, in the specified interval, is forwarded to the interpolation covered in subsection 5.3.

```
TIME,BITDEP,ROP_AVG,GRAX,GRCX,BLOCKCOMP,TAVG_BITVEL,WOB_AVG,TORQUE,SURF_RPM,RPMX,F
yyyy-MM-dd"T"HH:mm:ss.fffzzz,m,m/h,gAPI,gAPI,m,m/s,klbf,kN.m,RPM,RPM,L/min,psi,ppg,ppg,ppg,psi
2018-03-22T07:18:56.000+01:00,2904.103,18.820,-999.25,-999.25,7.927,0.005,5.800,18.090,120.770,-
2018-03-22T07:18:59.000+01:00,2904.119,18.850,-999.25,-999.25,7.911,0.005,4.710,18.940,120.780,-
2018-03-22T07:19:00.000+01:00,-999.25,-999.25,-999.25,-999.25,-999.25,-999.25,-999.25,-999.25,-99
2018-03-22T07:19:02.000+01:00,2904.133,18.790,-999.25,-999.25,7.897,0.005,3.150,17.230,120.770,-
2018-03-22T07:19:05.000+01:00,2904.150,18.880,-999.25,-999.25,7.880,0.005,2.720,17.230,120.780,-
```

Figure 5.2: Data structure of the provided .csv-file

[4]For the reader who is not familiar with CSV-files, more can be read under https://www.howtogeek.com/348960/what-is-a-csv-file-and-how-do-i-open-it/

5.3 Drilling Data Processing and Operation Identification

When all the drilling data were assembled in Matlab, they had to be processed in order to be used by the MPC-application as described in section 4. One of the challenges in the data set is the nonuniform sample rate of the different drilling variables. Variables like the WOB, RPM, and bit depth, are logged every 2-3 seconds. However, variables like the ECD at bit, ECD at total depth, and ECD at casing shoe are only logged every 30 seconds. Further, a recurring value in the dataset is the value -999.25, hereby referred to as "NaN". This value emerges for every drilling variable which does not measure a new value at the current time step. Some values have more of these than others, depending on their respective sampling rates compared to the other drilling variables.

To create the moving data window $\Phi(t)$ as described in subsection 4.2.1, a uniform interpolated table with data at every time step was created. To achieve this, the function `InterpolateTable(PreData,Time)` seen in appendix B.4 were created. Here the `PreData` is data extracted from a desired time interval from the supplied drilling data files, while `Time` is the time steps in the same interval on `datetime`-format which is arrays representing points in time. A linear interpolation is done using the Matlab function `resample(x,tx,fs,p,q)` from *Signal Processing Toolbox* to create the table `InterData` with a uniform sample spacing of (p/q)/fs. It should be noted that the interpolated data table was created with a sample spacing of 1 Hz to ensure that no real measured value was lost due to insufficient sample rate. However, the data incorporated in the moving data window $\Phi(t)$ is spread over every two seconds[5], under the assumption that wired drill pipe is used to transfer downhole data to the platform every other second as presented in section 2.1.7.

The processing progress for WOB is illustrated step by step in figure 5.3. The first figure illustrates the raw data, including the NaN values. Removing the NaN values yields the figure in the middle. Lastly, the data is interpolated to yield one data point for each time step.

Next, the data files provided by AkerBP included 40 days and nights of different operations such as tripping pipe in or out of the hole, drilling a stand, connection of stands, et cetera. In order for the parameter estimation, presented in subsections 3.1.2 - 3.1.3, to make sense it is essential that they are estimated while drilling. To identify the drilling operation, only sections where the measured depth increased over time were considered. However, the measured depth will also increase during tripping operations. The other drilling variables were therefore investigated in the intervals of increasing depth. This is illustrated in figure 5.4, where one can see that $ROP > 0$, $WOB > 0$, and $RPM > 0$ in the intervals where the depth is increasing. This is identified by inspection as the process of drilling two stands in sequence. It can also be seen that the pump is shut down and that the top drive RPM is stopped during a connection. This also results in a drop in the ECD. During the connection of a new stand, surveys are taken to obtain new information of the angles at the given depth.

[5] $\Phi(t) = \{\frac{dh}{dt}(t-2\cdot N), ..., \frac{dh}{dt}(t-2), X_j(t-2\cdot N), ..., X_j(t-2), \forall j = 1, ..., 8\}$

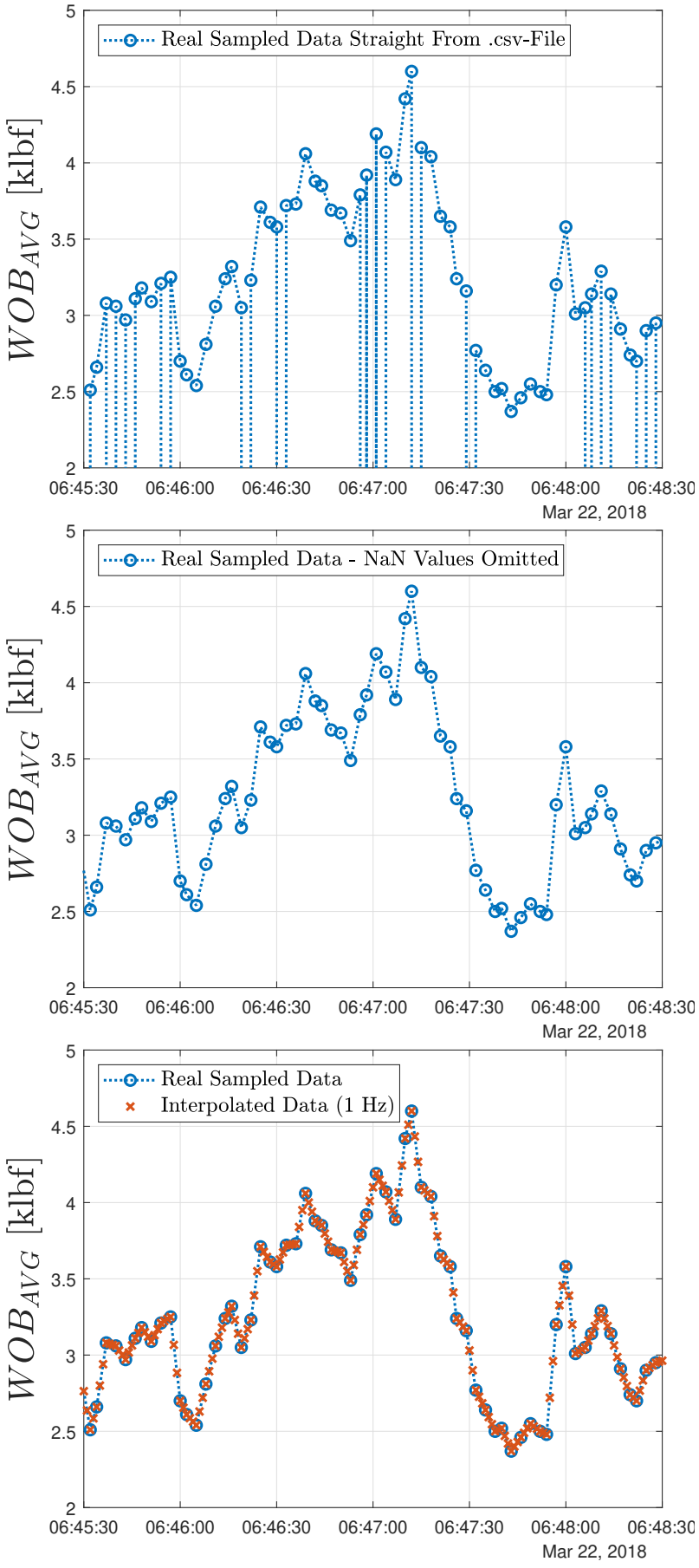


Figure 5.3: How the drilling data was treated in steps before it was used.

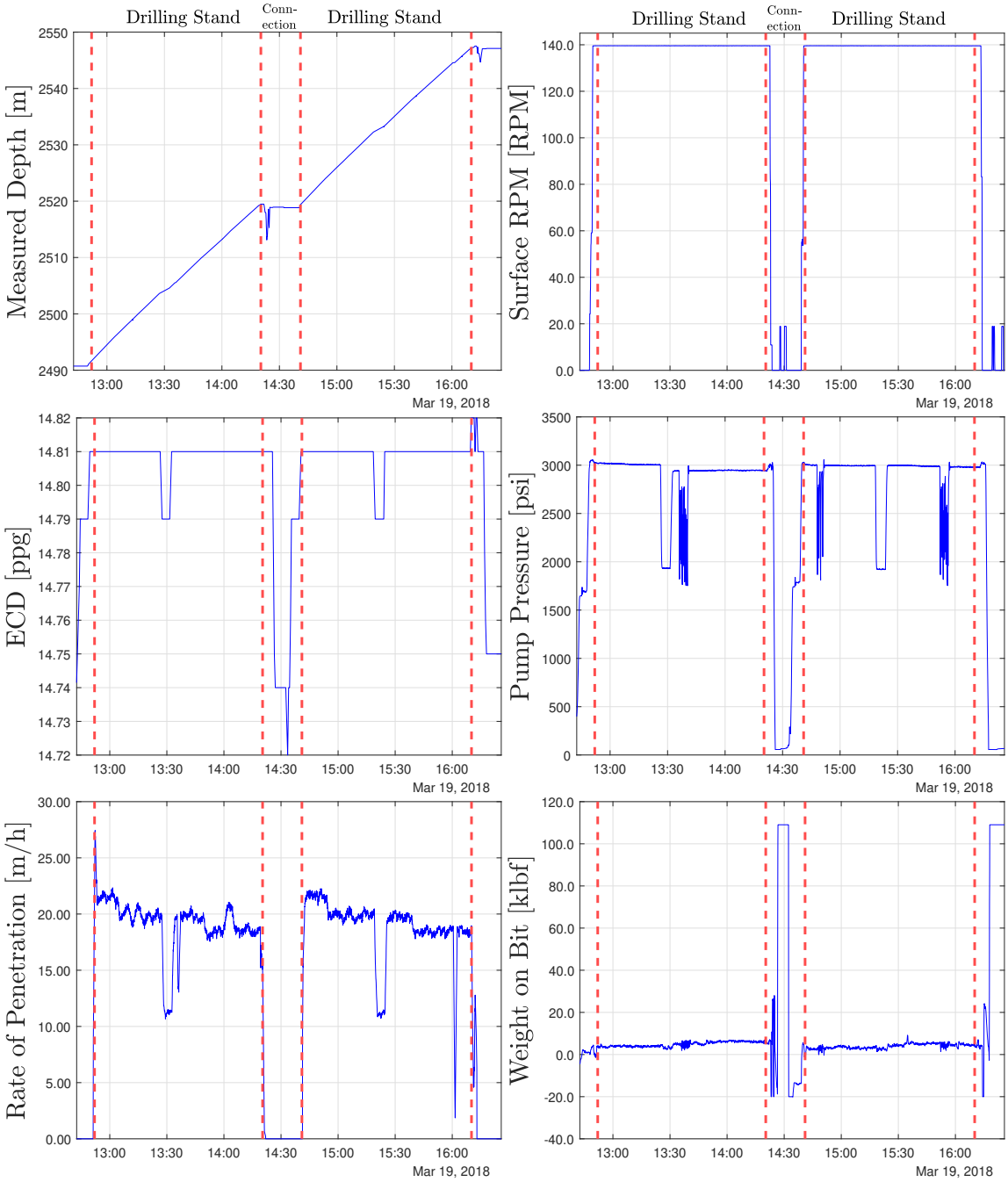


Figure 5.4: Drilling data from operation identified as drilling of two stands with connection in between.

6 Implementation and Verification

This section explains the model- and SLMPC setups in the Matlab program to get the results presented in section 7. Subsection 6.1 presents the model setup itself, while subsection 6.2 presents the setup and the verification strategy when using the SLMPC. Note that for the different subsections it is occasionally referred to the Matlab code which is found in appendix B.

6.1 B&Y-Model Setup and Verification

The setup shown in figure 6.1 illustrates how the modeled ROP was calculated in the developed Matlab program. Here, the interpolation of the drilling data is already done and is represented by InterData. The dashed arrows illustrate the data, which is only updated when a new measurement is available, while the regular arrows yield data at every time step. If there are no new measurement at the current time step, the drilling coefficients $a_1, ..., a_8$ and drilling parameters $X_1, ...X_8$ from last estimation and measurement is used.

The data set Φ used to estimate the coefficients $a_1, ..., a_8$ is found by the function `CreateDataSet(N,t,InterData)` as seen in appendix B.9. Note that the structure of this function and the function `DrillingParameters(h,t,InterData)`, given in appendix B.8, is similar. The only difference being that the latter finds the drilling parameters $X_1, ..., X_8$ and α of the current time step t, if available, while the data set extracts drilling parameters from N historical measurements. Note that α incorporates the rest of the drilling parameters shown in the output of the drilling parameters function.

Further, the depth from the provided drilling data was given as measured depth. Therefore, it could be used directly to find the pore pressure gradient (see figure 2.3), as the pressure gradients were expressed in terms of MD. However, the drilling parameters X_2, X_3, and X_4 are all expressed in terms of TVD. The function `MDtoTVD(MD)`, seen in appendix B.5, were therefore developed to convert from MD to TVD. The function was based on the angle averaging method, even though it does not include the curvature of the wellbore trajectory as presented in subsection 2.1.5. This method was chosen because the maximum overall angle change over the delivered wellbore trajectory data was $\beta = 0.1320 < 0.25$ rad. As presented in subsection 2.1.5.4, it is reasonable to set the ratio factor between the curved line segments and the straight line segments to $RF = 1$ for $\beta < 0.25$, so the minimum curvature method would only yield a straight-line approximation. It can also be argued that the deviation between the TVD acquired from the minimum curvature method and the angle averaging method is negligible to calculate the drilling parameters.

The drilling coefficients $a_1, ..., a_8$ is found in the function `FindDrillingParameters(DataSet,Method)` seen in appendix B.6, where `Method` determines whether multiple regression or trust-region method is used. Where multiple regression solves the equations in (20) to find $a_1, ..., a_8$ which minimizes the residuals, the trust-region method is used to solve the least-squares curve-fitting problem $\min_a \frac{1}{2}||Ca - d||_2^2 \quad s.t. \quad lb \leq a \leq ub$, by using the function `lsqlin(...)` from the *Optimization Toolbox*. Further, the upper and lower bounds were set according to Table 5. Note also that the last estimated parameters are stored and used as initial guess to the algorithm at next time instance through `persistent a0`.

Next, the EKF was implemented as shown in appendix B.10 in line with algorithm 2 shown in subsection 4.2.4. Note that calculation of the Kalman gain matrix $K(t)$ and the state and covariance correction step is only performed if a new measurement is available. For the simulations performed to evaluate the model performance, these measurements come from the interpolated data table `InterData`, which contains the actual drilling data. The modeled ROP can then be compared to the actual ROP, where all the drilling coefficients and parameters used in the model are based on the actual well data.

Note that when using multiple regression for parameter estimation, the estimated ROP had to be corrected in certain time steps due to sporadic large changes in the ROP estimates. This was done by setting $\widehat{ROP}(t) = \widehat{ROP}(t-1)$ if the difference between the two were larger than some bound (chosen as 5 m/h in the matlab code). The effect of this correction can be seen in the figures 7.2 and 7.3

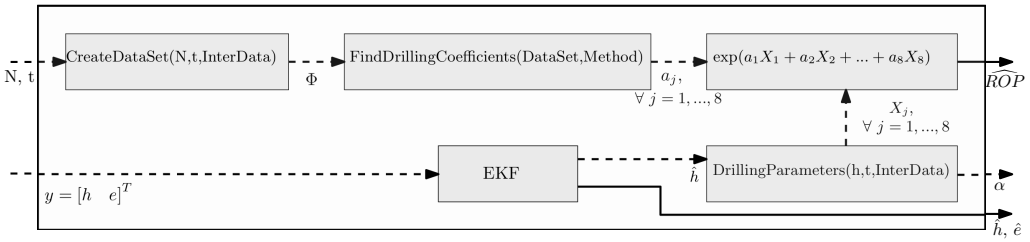


Figure 6.1: Model setup

6.2 Successive Linearization based MPC

This subsection explains the setup and verification strategy of the SLMPC. How the code was implemented for the SLMPC is given in appendix B.11. Subsection 6.2.1 aims to elaborate on the structure of this implementation, while subsection 6.2.2 elaborates how a verification strategy was developed in order to evaluate the SLMPC.

6.2.1 SLMPC Setup

The setup of the SLMPC is illustrated in figure 6.2, where the *Model* has the same structure as described in subsection 6.1. First, a linearization is performed based on the model parameters. The linearized matrices A_t, B_t, C_t, and D_t is then used to construct the different matrices described in subsection 4.3.

Next, the QP problem for the whole prediction horizon is constructed in *QP Formulation*. The function `InitMPC()` in appendix B.12 initializes the MPC with the length of the simulation, prediction horizon, size of states, inputs, and outputs, bounds, input reference values, the weight matrices, and a predetermined drilling plan.

Further, to find the P, H, E, and Δ-matrices from subsection 4.3 the functions `FindP(At,Ct,Np)`, `FindH(At,Bt,Ct,Dt,Np)`, `FindE(At,Ct,Np)`, and `FindDelta(Np,nu)` were created and is shown in appendix B.7. To create the matrix U_{-1} the Matlab function `repmat(u_op,[Np 1])` were used. The function simply creates a vector consisting of N_p rows of u_{op}. When constructing the matrix M, which collects all the constant terms, the reference ROP (Y^r) is constructed using a moving window that moves along the drilling plan defined in the initialization of the MPC. Next, $\tilde{H}, \tilde{c}$, the inequality constraints, upper and lower bounds, and the augmented matrices Γ and Z are constructed according to the formulations shown in subsection 4.3.

To solve the optimization problem the Matlab function `quadprog(H,f,A,b,Aeq,beq,lb,ub)` from *Optimization Toolbox* was used. The solver accepts quadratic objective functions with linear constraints. It also allows for the upper and lower bounds on the decision variables to be explicitly formulated. The `stop/TimerTicker` in the code was used to evaluate the average run speed of the solver.

At last, the first input from the open-loop solution of the MPC is applied to the system. In the simulation, this is done by adding the inputs w and r to the next time instance of InterData. In this way, the drilling parameters in the model will evaluate these in the next time step.

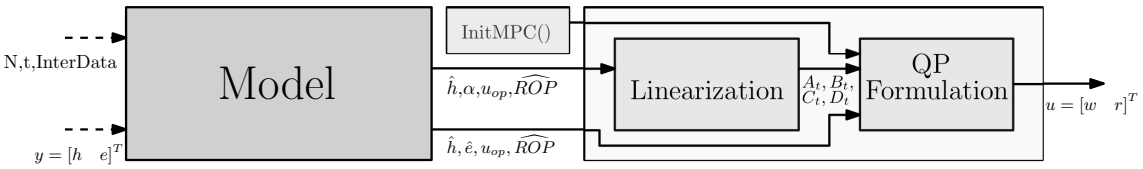


Figure 6.2: "Model" is structured as shown in figure 6.1, while the right hand side represents the SLMPC

6.2.2 SLMPC Verification

The model alone can be compared to the actual drilling data to evaluate its performance. However, other methods must be used when evaluating the model and SLMPC together. The challenge is to evaluate how a real well application would respond to the WOB and RPM applied by the SLMPC. It is already established that the B&Y-model is a simplification of the real world and that it will deviate from the response of an actual drilling process. Thus, measurements from the real world response are necessary to estimate the drilling coefficients, but also to correct the model estimates with the EKF.

However, in the absence of a proper simulation environment that can adequately determine the drilling response as a function of the drilling parameters, formation properties, et cetera, a verification strategy was developed. It does not, of course, replace the need for proper simulation tools but can suffice as a first step towards verification.

The verification setup is illustrated in figure 6.3. The coefficients and parameters of the simulated "real well", hereby denoted without quotation marks, are marked with W in super- or subscript. Both the model and the real well coefficients are first initialized in `InitDrillingCoefficients()` shown in appendix B.12. Note that the real well parameters are initialized based on 200 datapoints from the actual drilled well. This is done to get somewhat "realistic" initial coefficients. The model coefficients are initialized randomly within the reasonable range given in table 5. Further, the real well is quite comparable to the model setup, except for the `VaryingFormation()` which is used to simulate a nonuniform formation with varying properties. This is simply done by generating a random disturbance in the coefficients using the Matlab function `randn()` and can be seen in appendix B.1. The real well ROP_W is now found using the B&Y-model based on the drilling coefficients a_j^W and the parameters X_j^W. Further, the real depth h_W is propagated based on ROP_W before they are both injected to the next time instance of the InterData table, along with the optimal WOB and RPM calculated by the SLMPC. The inputs are then applied to both the real well and the model, in the next time instance, through InterData in the function `DrillingParameters`. The drilling coefficients in the model are then calculated by parameter estimation based on a moving horizon data window of size N propagating over the InterData table.

In this application, the flow rate q and the ECD p_{af} do not have any state models, so reasonable values for these as a function of other drilling conditions cannot be found. Therefore, their values at the next time instance in the `InterData` table are set to their value at the previous time instance and are thus held constant.

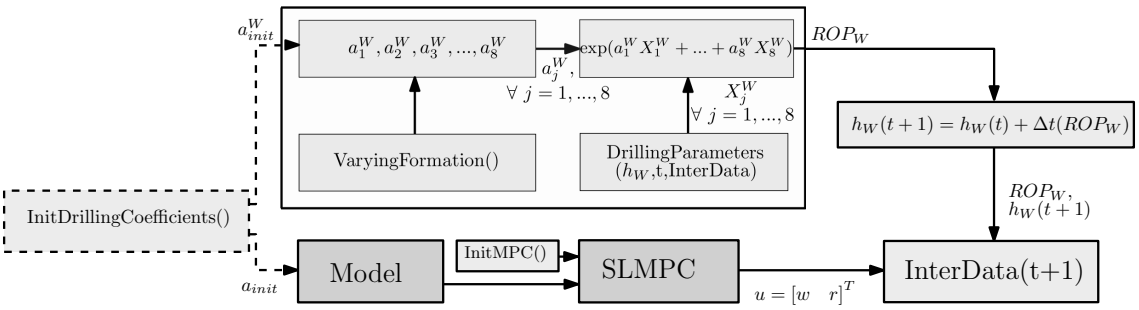


Figure 6.3: Strategy to verify the Model and SLMPC performance

7 Results

This section presents the results. Specifically, the results from the model itself are given in subsection 7.1, while the SLMPC results are given in subsection 7.2. The subsections are structured similarly. First, the relevant simulation parameters used to obtain the results are presented. Next, figures with results are given with brief figure texts and a more in-depth explanation either directly under each figure or at the end of the subsection. The results will further be discussed in section 8.

Note that the true values will be plotted in blue for all figures, while the estimated values are plotted in red.

7.1 Bourgoyne and Young ROP-Model

The B&Y ROP-model were tested on real well data, from a well drilled on 20th of March, 2018. The simulation length was $n = 4000$ with step size of 1 second, hence the data cover a little more than an hour (3600 seconds). The drilling data used to calculate the drilling coefficients and parameters over the simulation period are shown in figure 7.1.

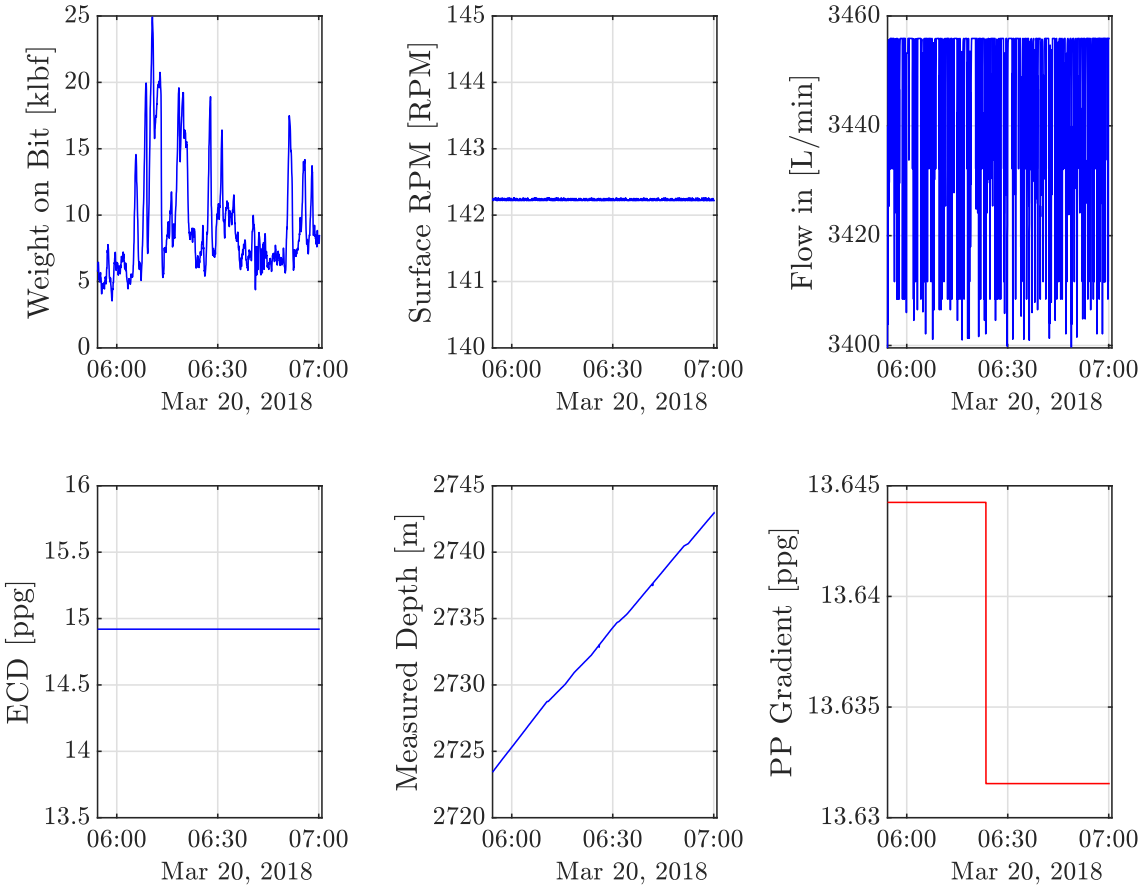
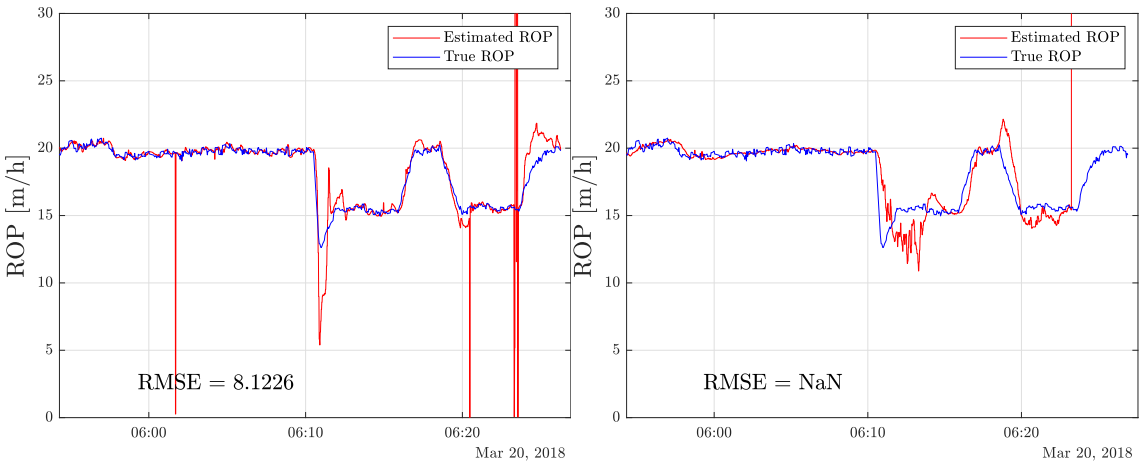


Figure 7.1: Drilling data over the simulated period

Note that the pore pressure gradient is red. This is because it was not included in the drilling data and had to be estimated as a function of the measured depth from the delivered pore pressure- and fracture-gradient table.

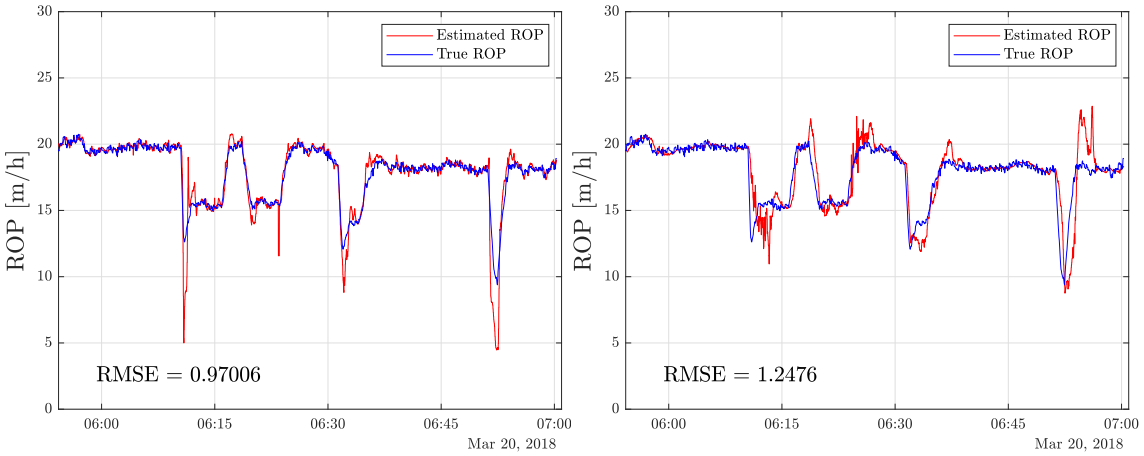
Further, the extended Kalman filter were initialized with $\hat{x}^-(0) = [h \quad 0]^T$, where h is the measured depth at the initial time step, $\hat{P}^-(0) = diag([1 \quad 10])$, $Q_d = diag([1 \quad 10])$, and $R_d = diag([1 \quad 1])$.

7.1.1 B&Y-Model Using Multiple Regression as Parameter Estimation Technique



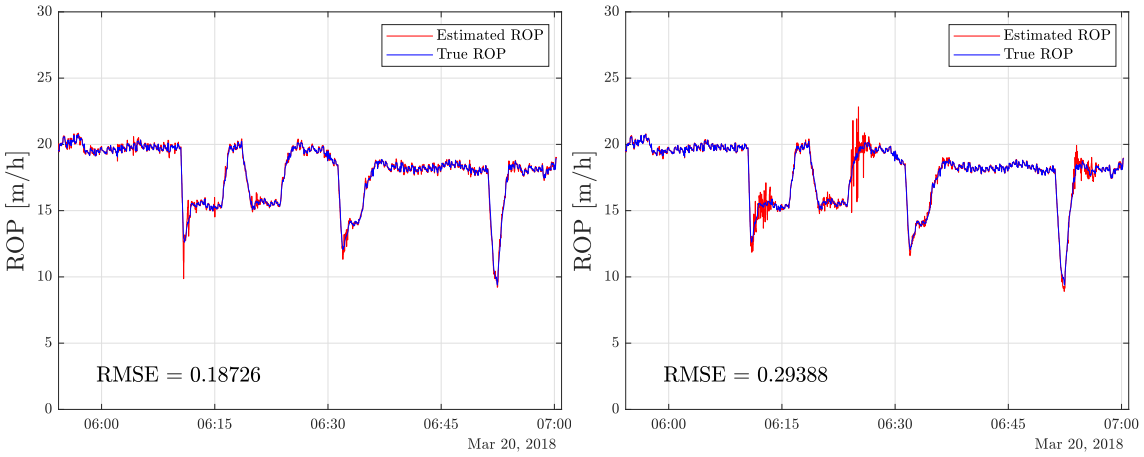
(a) Moving data window of $N = 30$ **(b)** Moving data window of $N = 100$

Figure 7.2: Model performance using MR for parameter estimation without EKF and no correction on sporadic ROP estimates.



(a) Moving data window of $N = 30$ **(b)** Moving data window of $N = 100$

Figure 7.3: Model performance using MR for parameter estimation without EKF and when correcting sporadic ROP estimates



(a) Moving data window of $N = 30$ **(b)** Moving data window of $N = 100$

Figure 7.4: Model performance using MR for parameter estimation with EKF and when correcting sporadic ROP estimates.

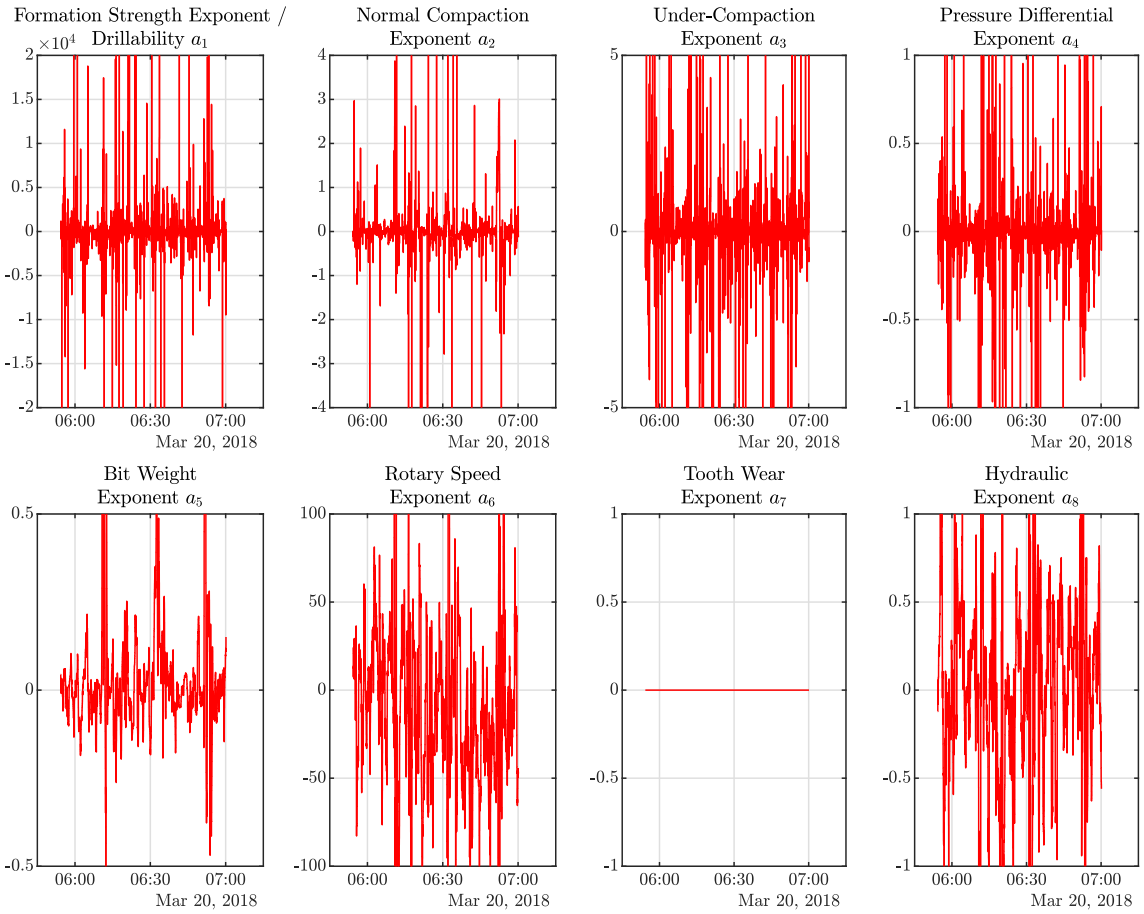


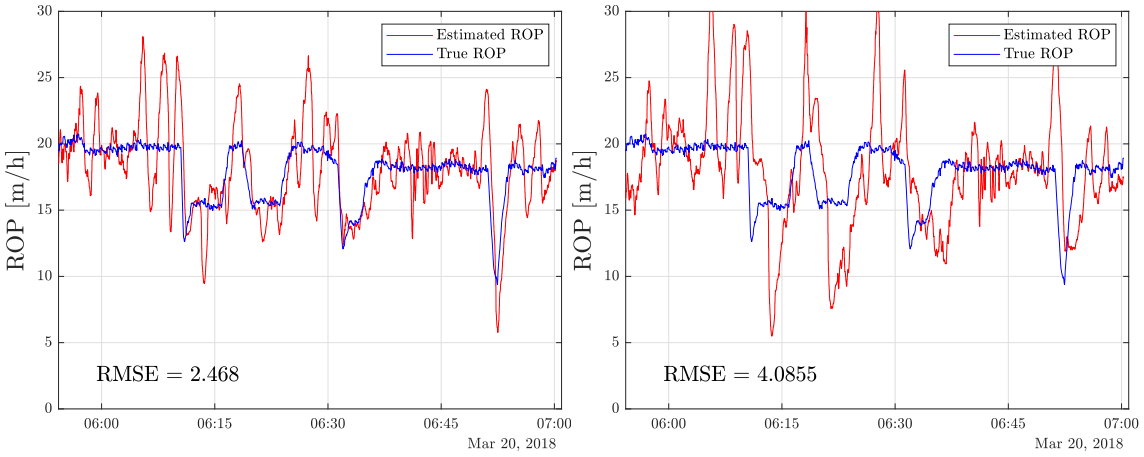
Figure 7.5: The estimated drilling coefficients $a_j \quad \forall j = 1, ..., 8$ using Multiple Regression with moving data window of $N = 30$.

Figure 7.2 shows the performance of the model when using multiple regression as parameter estimation technique. Some large sporadic deviations are seen, where for the case of moving data window of size $N = 100$, it caused the model to fail. Removing these sporadic deviations yields the performance of figure 7.3. Further, figure 7.4 shows the performance when the Kalman filter is used to further improve the estimates. The results are shown for two cases, (a) where the moving data window $\Phi(t)$ is of size $N = 30$ and (b) where the size is $N = 100$. Note that the root mean square error (RMSE) is given for each result. RMSE is a measure of how close the estimated values are to the real, and is calculated by $RMSE = \sqrt{\frac{\sum_{t=1}^{n}(ROP(t)-\widehat{ROP}(t))^2}{n}}$, where n is number of simulation steps. Hence RMSE close to zero implies that the model response is close to the true value, while large RMSE implies bad correspondence between the two.

It is seen that increasing the data window size yields more sluggish model estimates, yielding a larger RMSE. It also performs worse in intervals where the ROP changes rapidly. The model performance vastly improves when the EKF is applied, which is emphasized by the decrease in RMSE for both cases (a) and (b).

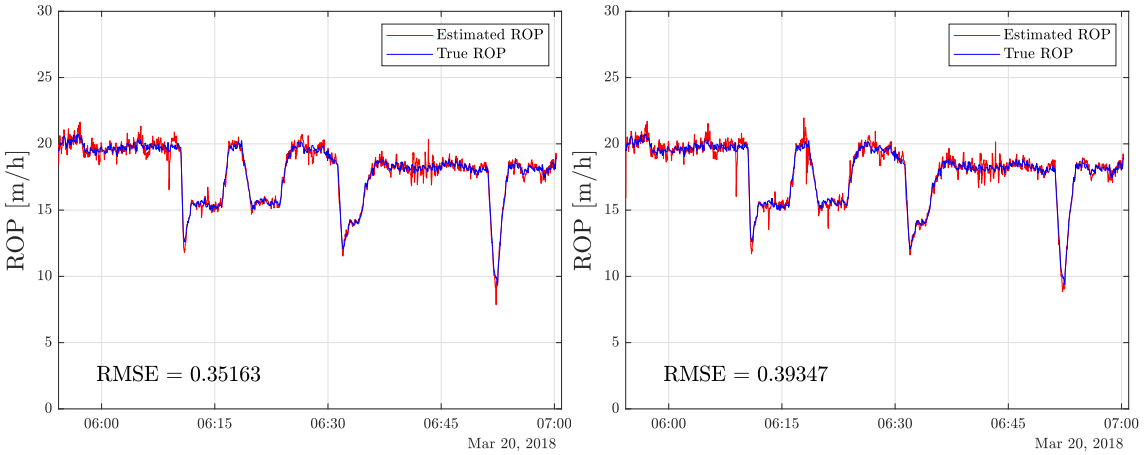
Figure 7.5 shows the estimated drilling coefficients when using moving data window of $N = 30$. It can be seen that these values rapidly fluctuates between positive and negative values. Further, it can be seen that the scales of the parameters are very different. While the drillability a_1 mainly varies between $\pm 2 \times 10^4$, the bit weight exponent a_5 ranges between ± 0.5. Note that the tooth wear exponent is constant, as it was not included in the model.

7.1.2 B&Y-Model Using Trust-Region Method as Parameter Estimation Technique



(a) Moving data window of $N = 30$ **(b)** Moving data window of $N = 100$

Figure 7.6: Model performance using TR for parameter estimation without EKF



(a) Moving data window of $N = 30$ **(b)** Moving data window of $N = 100$

Figure 7.7: Model performance using TR for parameter estimation with EKF

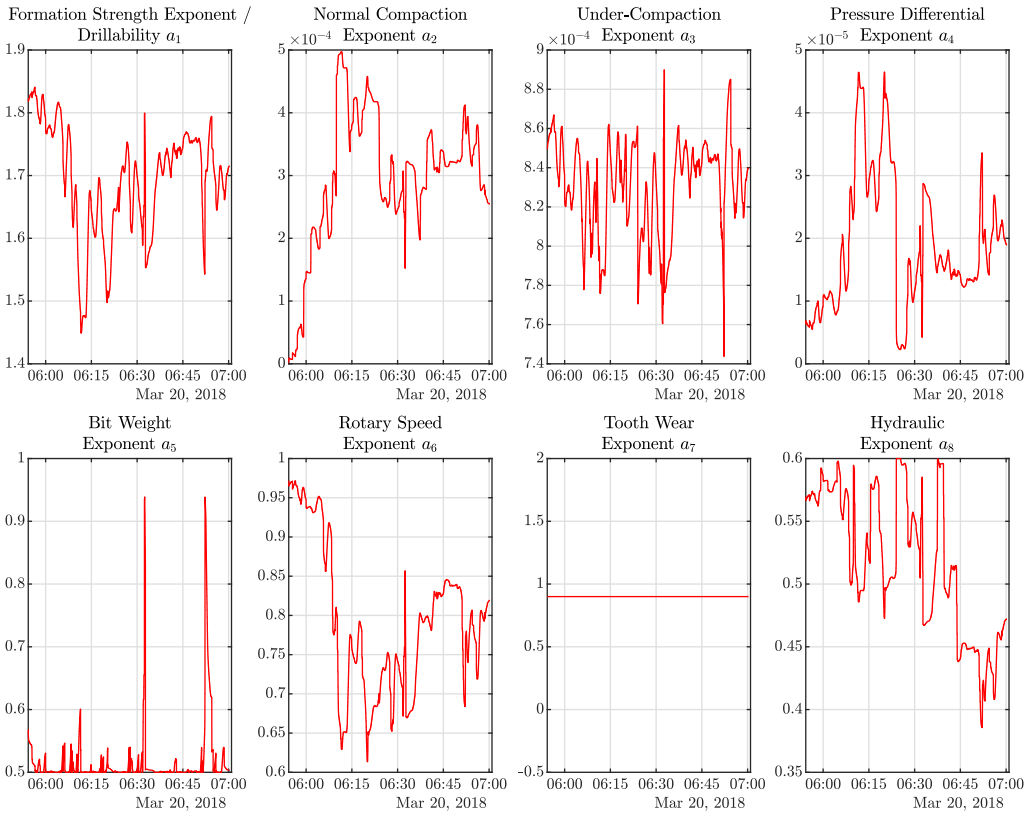


Figure 7.8: The estimated drilling coefficients $a_j \quad \forall j = 1, ..., 8$ using Trust-region method with moving data window of $N = 30$.

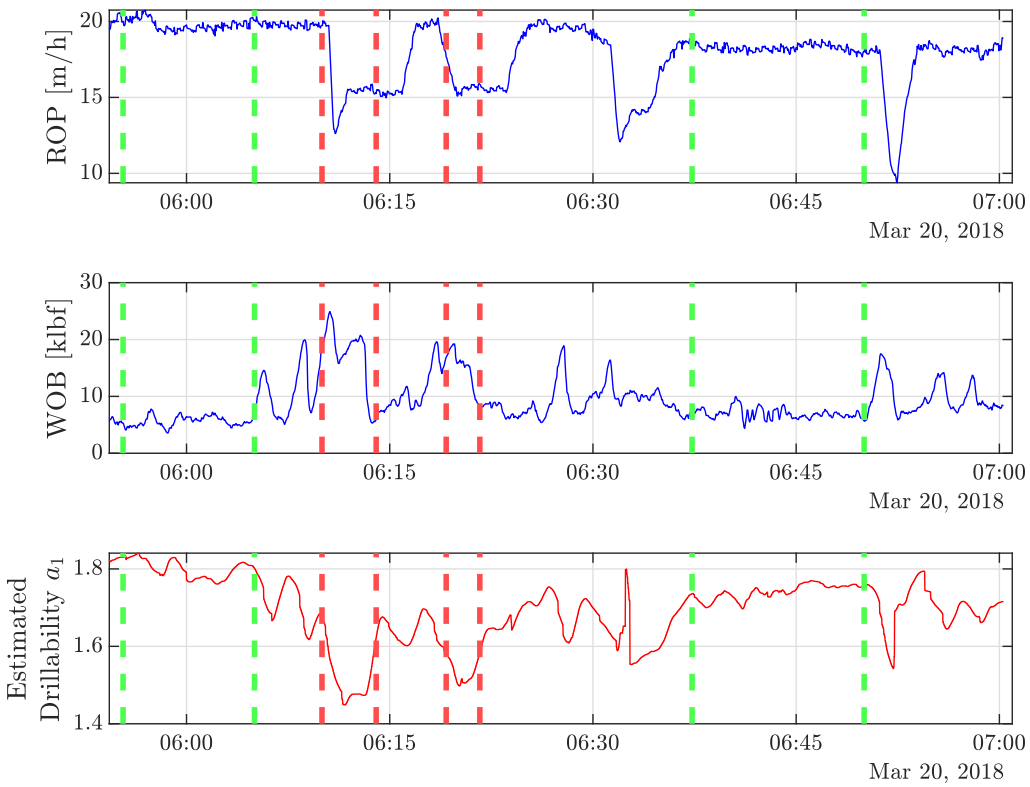


Figure 7.9: ROP and WOB with corresponding estimated drillability using TR with moving data window of $N = 30$. Green intervals where drillability is relatively high, red intervals where drillability is relatively low.

Figure 7.6 and 7.7 shows the model performance, without and with Kalman filter, respectively, when using the Trust-region method to estimate the drilling coefficients. It is seen that the model has some severe fluctuations around the real ROP value, resulting in deviations up to 10 m/h. However, for $N = 30$, the estimated ROP follows the trend of the real ROP, while $N = 100$ barely captures it, yielding nearly the doubled RMSE. The RMSE is vastly reduced for both cases when using the EKF, yielding 1/7th and 1/10th of the RMSE when not using EKF for (a) and (b), respectively.

Figure 7.8 shows the estimated drilling coefficients when using trust-region method as parameter estimation technique, and moving window of size $N = 30$. Note that the coefficients obey the upper and lower limits, given by table 5, over the whole simulation period. This also implies that they are never negative.

Further, figure 7.9 shows how the estimated drillability a_1 propagates over the simulation period and shows the evolution of ROP and WOB in the same period. The intervals given by the green dashed lines shows that the drillability a_1 is at its highest ($\sim$ 1.7-1.9 m/h) when the ROP is at its highest ($\sim$ 18-20 m/h), and the WOB is at its lowest ($\sim$ 5-10 klbf). Conversely, the red intervals shows that the drillability is low when the ROP is low, and the WOB is high. This makes sense as a_1 incorporates the effect of the formation strength as stated in subsection 3.1.1. One would expect that if the rock formation strength is high (low drillability), then even a high WOB would yield a low ROP.

7.2 Successive Linearization based MPC

Subsection 7.2.1 presents the results for a SLMPC using a static model. Next, the results from the verification strategy presented in subsection 6.2.2 is given in subsection 7.2.2. Note that the constraints are visually presented in the results and is therefore not specified by text, except for the upper and lower bound on input rate of change which were set to ± 0.5 for both WOB and RPM. Further, a sampling rate of $\Delta t = 1s$ were used for the SLMPC.

7.2.1 SLMPC with Static Model

This subsection presents the results for a test on the SLMPC alone. The setup is the same as shown in figure 6.2, but the model used is constant in the sense that the drilling coefficients $a_j \quad \forall j = 1, ..., 8$ are constant. The simulation length was $n = 4000$ and the prediction horizon set to $N_p = 10$. Further, since there is no real well in this simulation, the EKF was omitted. The drilling coefficients and the weight matrices used to achieve the results were

$$
\begin{aligned}
a_1 = 1.9, \quad a_2 = 0.000053, \quad a_3 = 0.0009, \quad a_4 = 0.000008, \\
a_5 = 0.6, \quad a_6 = 0.7, \quad a_7 = 0.9, \quad a_8 = 0.59
\end{aligned}
$$

$$
\begin{aligned}
q = 10^4, \quad r_1 = 10, \quad r_2 = 10, \quad s_{u_1} = 1, \quad s_{u_2} = 10^4, \\
s_{\delta u_1} = 10^6, \quad s_{\delta u_2} = 10^6, \quad s_y = 10^6
\end{aligned}
$$

$$
Q = \begin{bmatrix} q & & & \\ & q & & \\ & & \ddots & \\ & & & q \end{bmatrix}, \quad R = \begin{bmatrix} r_1 & 0 & \cdots & 0 \\ 0 & r_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & r_2 \end{bmatrix}, \quad S_u = \begin{bmatrix} s_{u_1} & 0 & \cdots & 0 \\ 0 & s_{u_2} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & s_{u_2} \end{bmatrix}, \tag{53}
$$

$$
S_{\delta u} = \begin{bmatrix} s_{\delta u_1} & 0 & \cdots & 0 \\ 0 & s_{\delta u_2} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & s_{\delta u_2} \end{bmatrix}, \quad S_y = \begin{bmatrix} s_y & 0 & \cdots & 0 \\ 0 & s_y & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & s_y \end{bmatrix}
$$

Further, ρ_u, $\rho_{\delta u}$, and ρ_y were all set to zero, as only the quadratic weight matrix was used. The matrix sizes are defined in subsection 4.3. Two cases are considered for testing all the different constraints in subsection 7.2.1. One was implemented with the weights above (green), while the other was implemented without the soft margin constraints, which is meant to avoid that max WOB and max RPM are run simultaneously (red). The latter were therefore implemented with the corresponding weights s_{u_1} and s_{u_2} being zero.

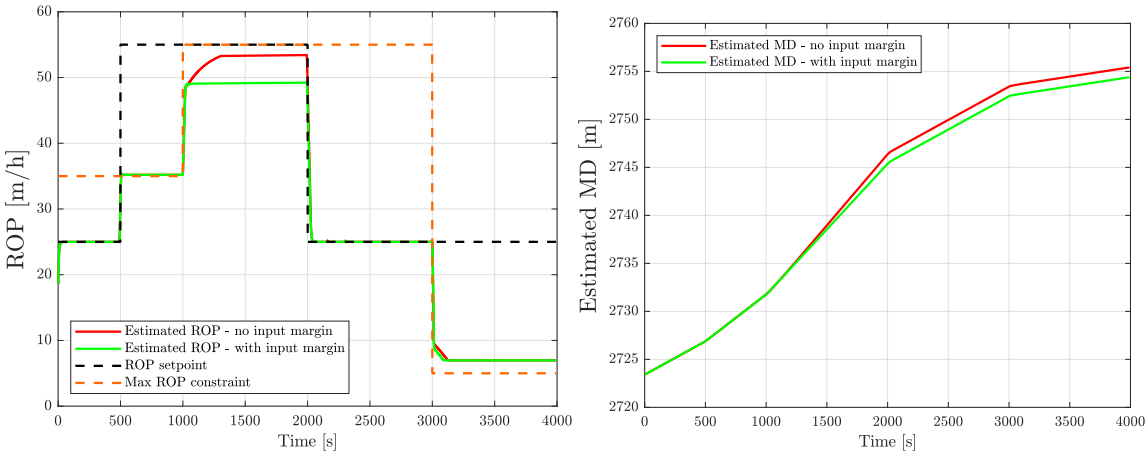


Figure 7.10: Estimated ROP and MD with (green) and without (red) margin constraints on the inputs

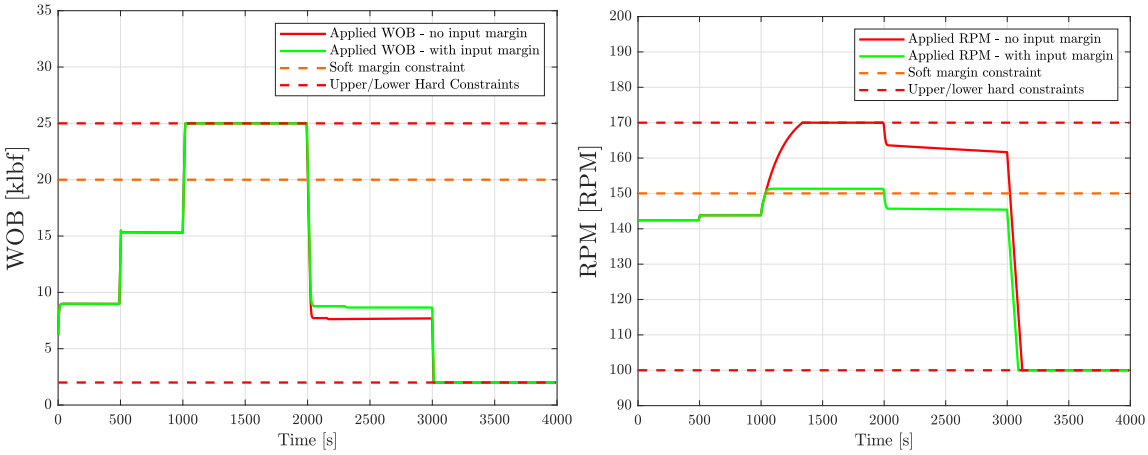


Figure 7.11: WOB and RPM applied to the model with (green) and without (red) margin constraints

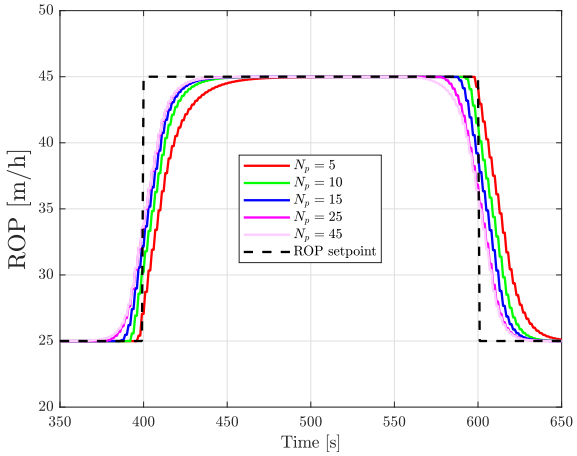


Figure 7.12: Estimated ROP for different prediction horizons N_p

Figure 7.10 shows the evolution of the modeled ROP and measured depth over the simulation period, where the inputs applied by the SLMPC can be seen in figure 7.11. It should be noted that all the soft constraints are plotted in orange, while the hard constraints are red.

For the first 500 seconds, the ROP tracks the reference value without any problems. After 500 seconds, the reference value is set to 55 m/h. However, the max ROP constraint is no higher than 35; hence the ROP is set to no higher than this. After 1000 seconds, a new max ROP is set to 55, so the ROP is yet again increased. This is where the two cases, with and without soft margin constraints, differ. As shown in figure 7.11, the RPM is held back to obey the margin constraint for the green case. The margin constraint effect is further shown in the time interval 2000-3000 seconds, where the solver finds it cheaper to increase the WOB to keep the RPM under the margin constraint. In the time interval 3000-4000 seconds, the max ROP constraint drops down to 5 m/h. However, the WOB and RPM have lower bounds governed by hard constraints at 2 klbf and 100 RPM, respectively. This results in the ROP not dropping below approximately 7 m/h. The red case is seen to drill the well slightly faster than the green in the time interval 1000-2000 due to the difference in ROP in the same period.

Figure 7.12 shows the difference in the ROP response when increasing the reference value at 400 seconds and decreasing it at 600 seconds. The figure illustrates the predictive capabilities of the MPC. The higher the prediction horizon N_p, the earlier the ROP starts to rise or fall depending on the setpoint change. However, with the rising prediction horizon, the number of decision variables in the problem also increases, and thus the time it takes to solve the optimization problem. The average time it took the SLMPC to solve the optimization problem at each time instance, and the RMSE for the different prediction horizons is given in table 6

Prediction Horizon N_p	**Number of Decision Variables** $N_p \cdot (3 \cdot n_u + n_y)$	**Average Solving Time** [s]	**RMSE** [m/h]
5	35	0.0062	3.8111
10	70	0.0208	2.9272
15	105	0.0369	2.4594
25	175	0.1313	2.2287
45	315	0.45	2.2293

Table 6: Tracking performance and solving time for the setpoint change shown in figure 7.12 in ROP with different prediction horizons

While the average solving time increases more rapidly as the prediction horizon and decision variables increase, the change in RMSE converges. It can also be seen in figure 7.12 that the response difference of prediction horizon $N_p = 10$ to $N_p = 45$ are not as significant as for $N_p = 5$ to the rest. The choice of proper prediction length is thus a tradeoff between solving time and performance.

7.2.2 Control of "real well" Using SLMPC

This subsection presents the results for the verification strategy presented in subsection 6.2.2, where the setup is given in figure 6.3. Here the drilling coefficients were estimated using the trust-region method with a moving data window of $N = 30$. Further, a varying formation were created by randomly altering the real well drilling coefficients a_1^W, a_5^W, and a_6^W for the first run, with the rest of the coefficients being constant. The drilling coefficients were saved and used for all the results below, and is shown in the following figure

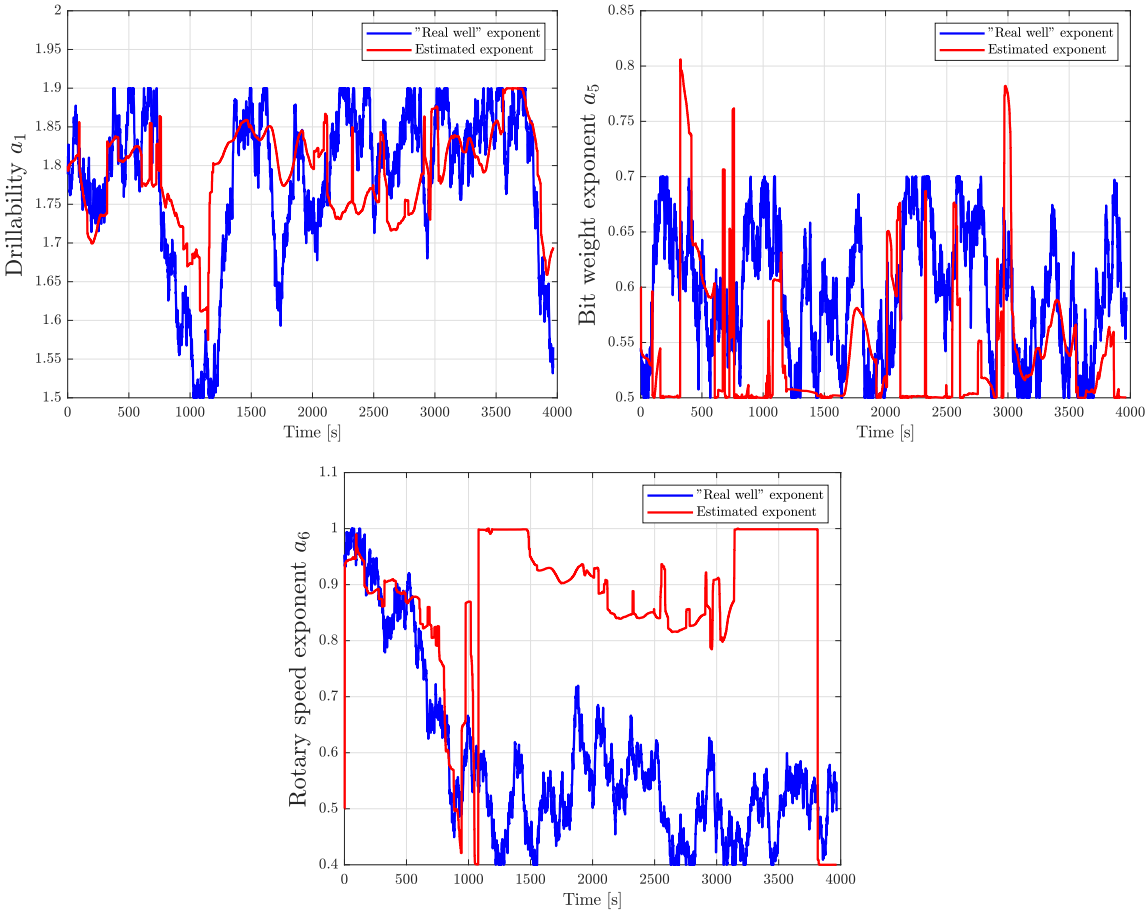


Figure 7.13: The simulated varying formation coefficients, and their corresponding estimates using TR-method with moving window of $N = 30$

The SLMPC were initialized with a prediction horizon of $N_p = 15$. Further, the drilling coefficients and weighting matrices used for the following results are defined in the same way as in equation (53), but with the values of

$$\begin{aligned} a_1 &= 1.5, \quad a_2 = 0.000053, \quad a_3 = 0.0009, \quad a_4 = 0.000008, \\ a_5 &= 0.6, \quad a_6 = 0.5, \quad a_7 = 0.9, \quad a_8 = 0.59 \end{aligned} \tag{54}$$

$$\begin{aligned} q &= 10^4, \quad r_1 = 10, \quad r_2 = 10, \quad s_{u_1} = 1, \quad s_{u_2} = 10^6, \\ s_{\delta u_1} &= 10^6, \quad s_{\delta u_2} = 10^6, \quad s_y = 10^6 \end{aligned}$$

The Kalman filter were initialized with $\hat{x}^-(0) = [h \quad 0]^T$, $\hat{P}^-(0) = diag([1 \quad 10])$, $Q_d = diag([1 \quad 10])$. Results from two different covariance matrices for the measurement noise $R_d = diag([1 \quad 1])$ and $R_d = diag([1 \quad 1000])$ are presented.

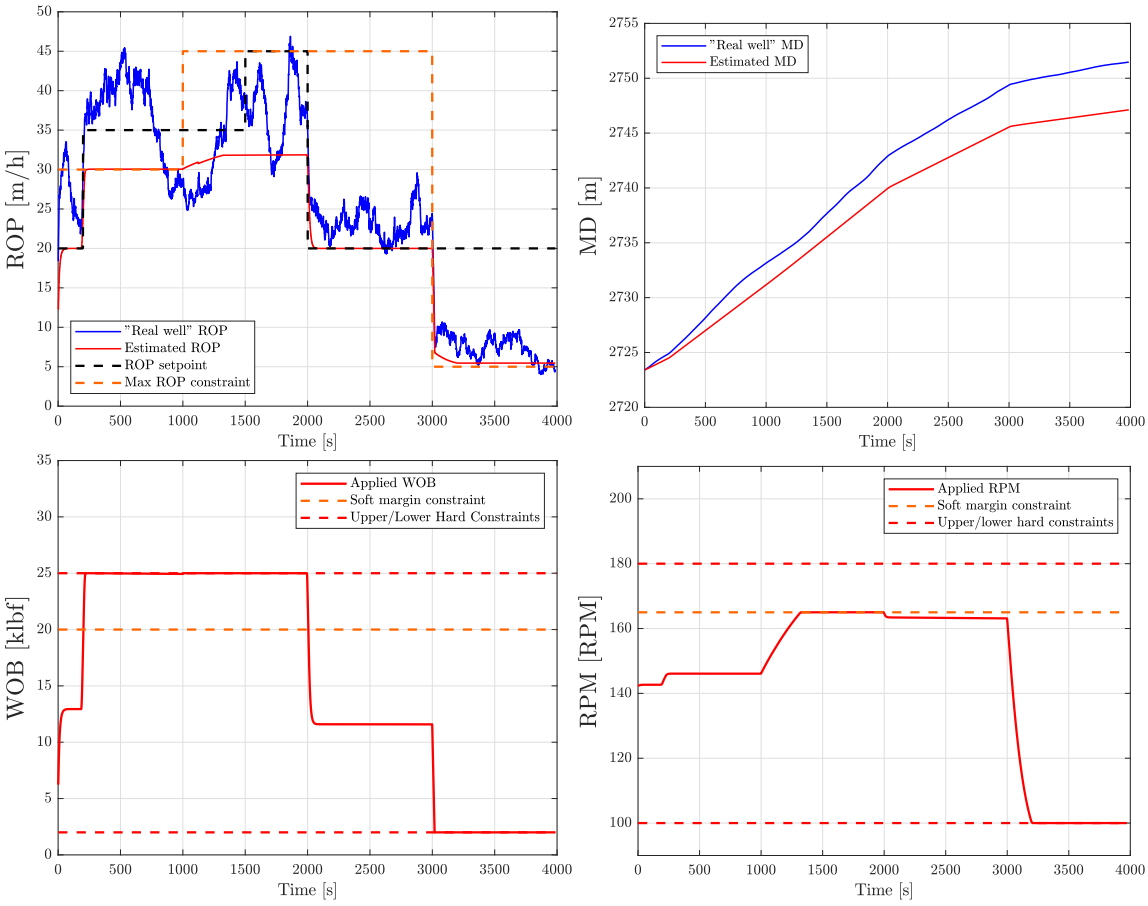


Figure 7.14: Control of real well using a static model (no parameter estimation or EKF)

Figure 7.14 shows the model's performance compared to the real well response when using a static model in the SLMPC. It is seen that the model poorly reflects the actual well behavior, with deviations up to 15 m/h in the time interval of 200-2000 seconds. On the other hand, the modeled ROP better approximates the real ROP behavior in the interval of 2000-4000 seconds. This can be caused by the model being a better approximation of the local drilling conditions in certain areas. For instance, it is seen that the static rotary speed exponent of $a_6 = 0.5$ is a reasonable approximation of the real exponent shown in figure 7.13[6] from the interval of 1000-4000 seconds. Using the static model in the SLMPC yields smooth control inputs, which only varies with the setpoint change. However, due to the large deviations in the true ROP and the modeled ROP, large deviations appear in the modeled and true measured depth.

[6]Note that the "estimated exponent" in this figure is the results of parameter estimation using TR-method, and is not relevant for the static model

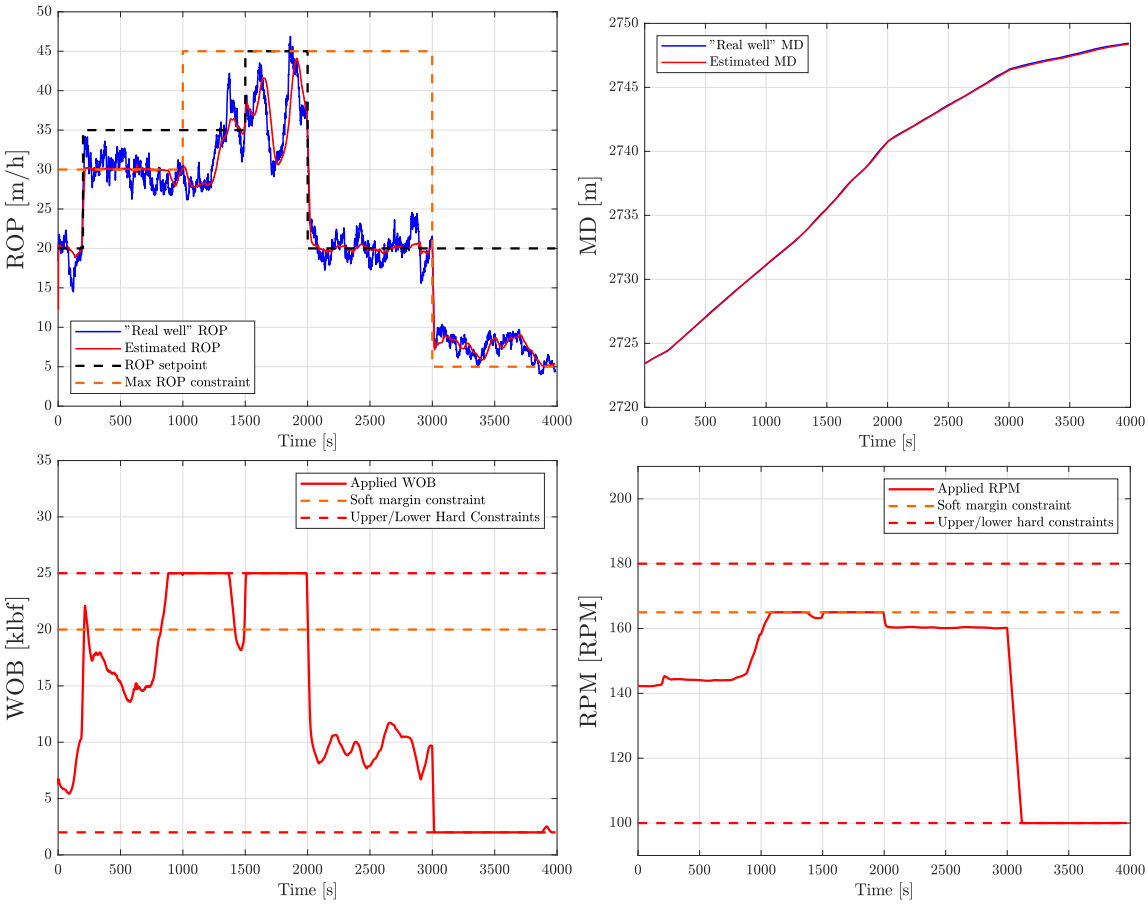


Figure 7.15: Control of real well using a model which is updated by TR parameter estimation over a moving horizon window of size $N = 30$ (no EKF)

Figure 7.15 shows that the performance can be vastly improved, relative to the static model, by estimating the local drilling coefficients using the trust-region method over a moving data window of size $N = 30$. This does, however, require more active control inputs. Note that for the areas where the control inputs are saturated, due to upper and lower bounds, the real well ROP is the same as when using the static model. Naturally, this makes sense as the same control inputs should yield the same real system response. A small deviation can be observed in the estimated and true measured depth, due to the deviations on the real well ROP and the estimated ROP.

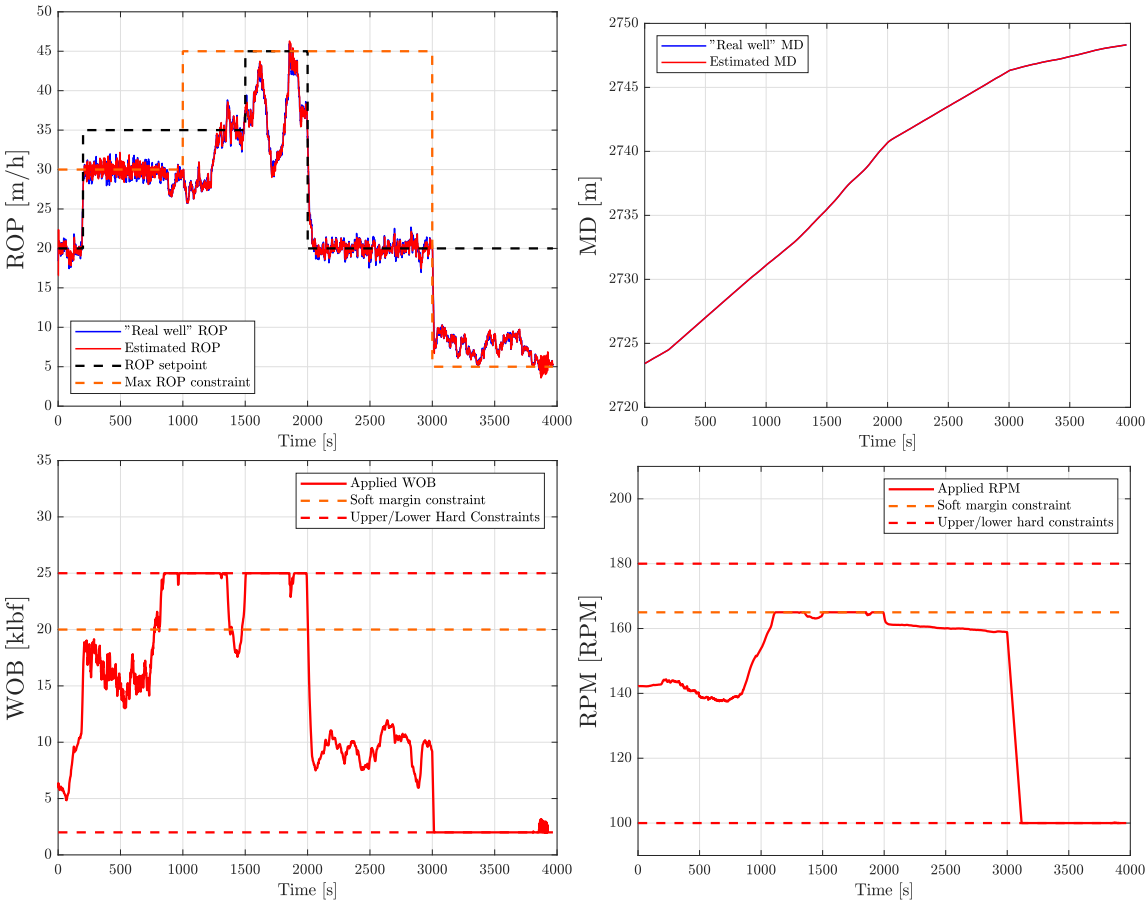


Figure 7.16: Control of real well using a model which is updated by TR parameter estimation over a moving horizon window of size $N = 30$ and extended Kalman filter with $R_d = diag([1 \quad 1])$

Figure 7.16 shows the performance while using both TR-method over a moving data window of size $N = 30$, and EKF with $R_d = diag([1 \quad 1])$. It can be seen that the modeled ROP are much closer to the real well ROP over the whole simulation period, relative to the modeled ROP in figure 7.14 and 7.15. It does, however, require a much more active use of the control inputs. In the time interval 200-1000 seconds it can be seen that the WOB and RPM have to rapidly fluctuate in order to make achieve the desired ROP. With the modeled ROP being so close to the actual ROP, and the measured depth being estimated through the EKF, no deviation in the estimated and true measured depth is observed. It is, however, shown in table 7 that there is a small deviation between the two.

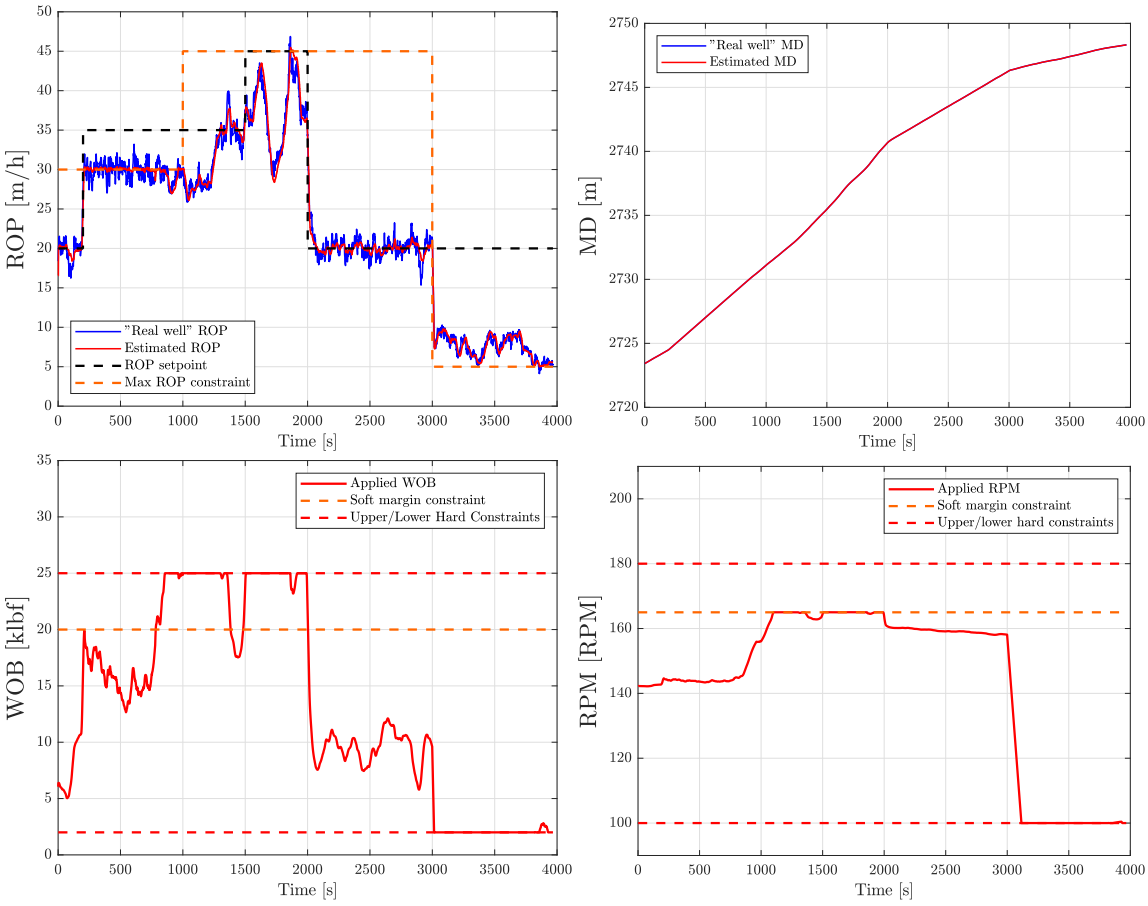


Figure 7.17: Control of real well using a model which is updated by TR parameter estimation over a moving horizon window of size $N = 30$ and extended Kalman filter with $R_d = diag([1 \quad 1000])$

Figure 7.17 shows the performance when using the exact same setup as in figure 7.16, but using a higher covariance on measurement uncertainty of the error state e. Here $R_d = diag([1 \quad 1000])$ were used. It can be seen that the largest deviations from figure 7.15, where no EKF were used, is removed. The control inputs are still more actively actuated then using no EKF, but the rapid fluctuation on WOB and RPM can be seen to be reduced within the time interval of 200-1000 seconds, relative to when $R_d = diag([1 \quad 1])$. No deviations in the modeled and real measured depth is observed, but can be seen in table 7.

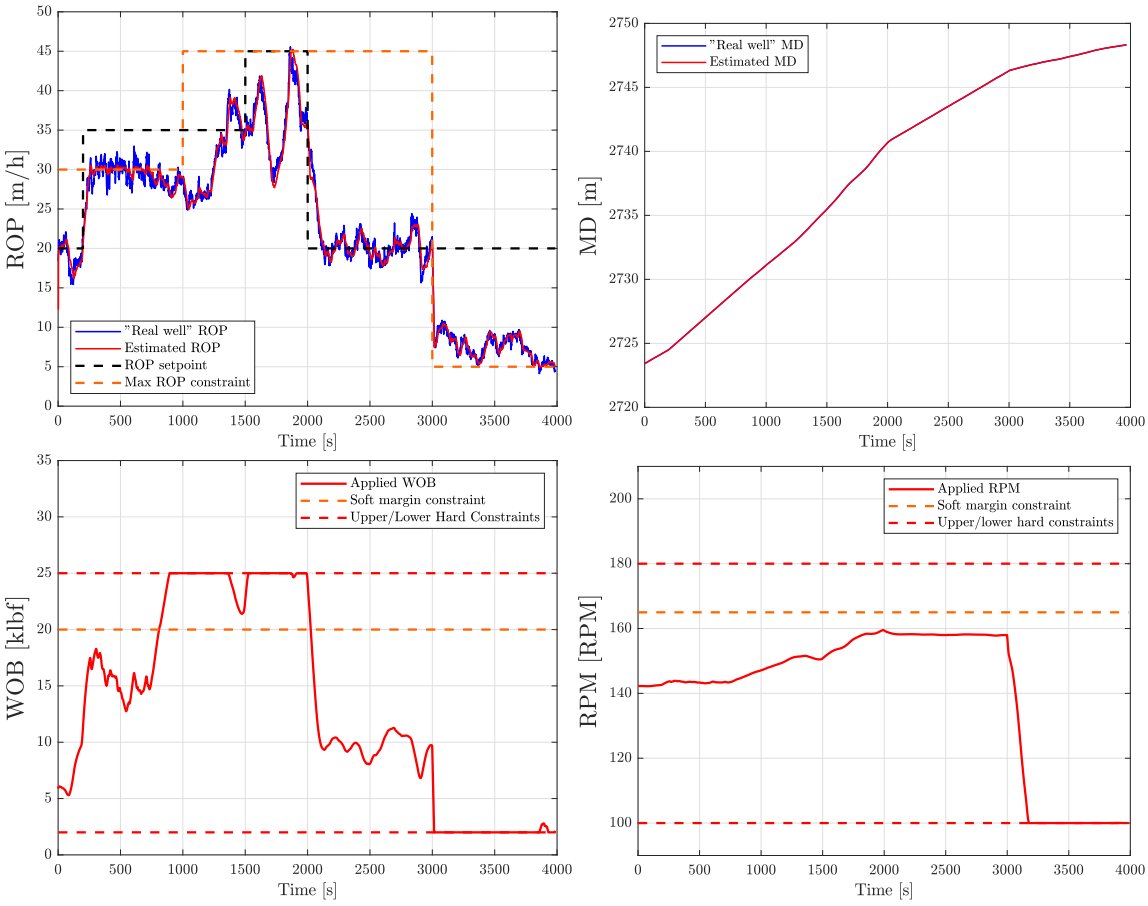


Figure 7.18: Control of real well using a model which is updated by TR parameter estimation over a moving horizon window of size $N = 30$ with extended Kalman filter using $R_d = diag([1 \quad 1000])$ and $s_{\delta u_1} = s_{\delta u_2} = 10^7$

Figure 7.18 shows the performance when using the exact same setup as in figure 7.17, but with higher weights on the rate of change of the control inputs. This means that deviations in input rate of change from zero are penalized more heavily. Even more conservative use of the control inputs can be seen, relative to figure 7.17. More significant deviations can thus be observed between the real well ROP and the reference. No deviation is observed in the measured depth, but is also given in table 7.

In order to compare the five different results presented above, table 7 was made. It presents the three errors $RMSE_P$, $RMSE_M$, and $RMSE_{MD}$. $RMSE_P$ yields the RMSE between the real well ROP and the ROP reference value in the time interval of 2000-3000 seconds. This is the longest run in the simulation where no constraints are limiting the reference tracking. Further, $RMSE_M$ yields the RMSE between the modeled ROP and the real well ROP over the whole simulation period. Lastly, $RMSE_{MD}$ yields the RMSE between the modeled and the real well measured depth.

Model Setup	**Figure**	$RMSE_P$ $[m/h]$	$RMSE_M$ $[m/h]$	$RMSE_{MD}$ $[m]$
Static Model	fig. 7.14	3.9824	5.8110	2.9808
Model using TR-method for parameter estimation	fig. 7.15	2.0451	1.7928	0.0580
Model using TR-method for parameter estimation and EKF with $R_d = diag([1 \quad 1])$	fig. 7.16	1.5151	0.5536	0.0002
Model using TR-method for parameter estimation and EKF with $R_d = diag([1 \quad 1000])$	fig. 7.17	1.7644	0.9920	0.0005
Model using TR-method for parameter estimation and EKF with $R_d = diag([1 \quad 1000])$ and $s_{\delta u_1} = s_{\delta u_2} = 10^7$	fig. 7.18	3.2828	0.9823	0.0005

Table 7: RMSE as a result of different model setups

It is seen that the static model has the highest $RMSE_p$, $RMSE_M$, and $RMSE_{MD}$ and thus the worst performance. The best performance, in terms of RMSE, is achieved when using the TR-method for parameter estimation and EKF using $R_d = diag([1 \quad 1])$, but the performance is not far from the model type using $R_d = diag([1 \quad 1000])$. Both the methods using EKF yields very low $RMSE_{MD}$ relative to the two model types where no EKF was used. This makes sense, as they are continuously updated by the real well measured depth. The last model type, using more conservative actuation, yields a higher $RMSE_P$. This is also expected since less active actuation will not counteract the formation variation to the same extent as for more active actuation.

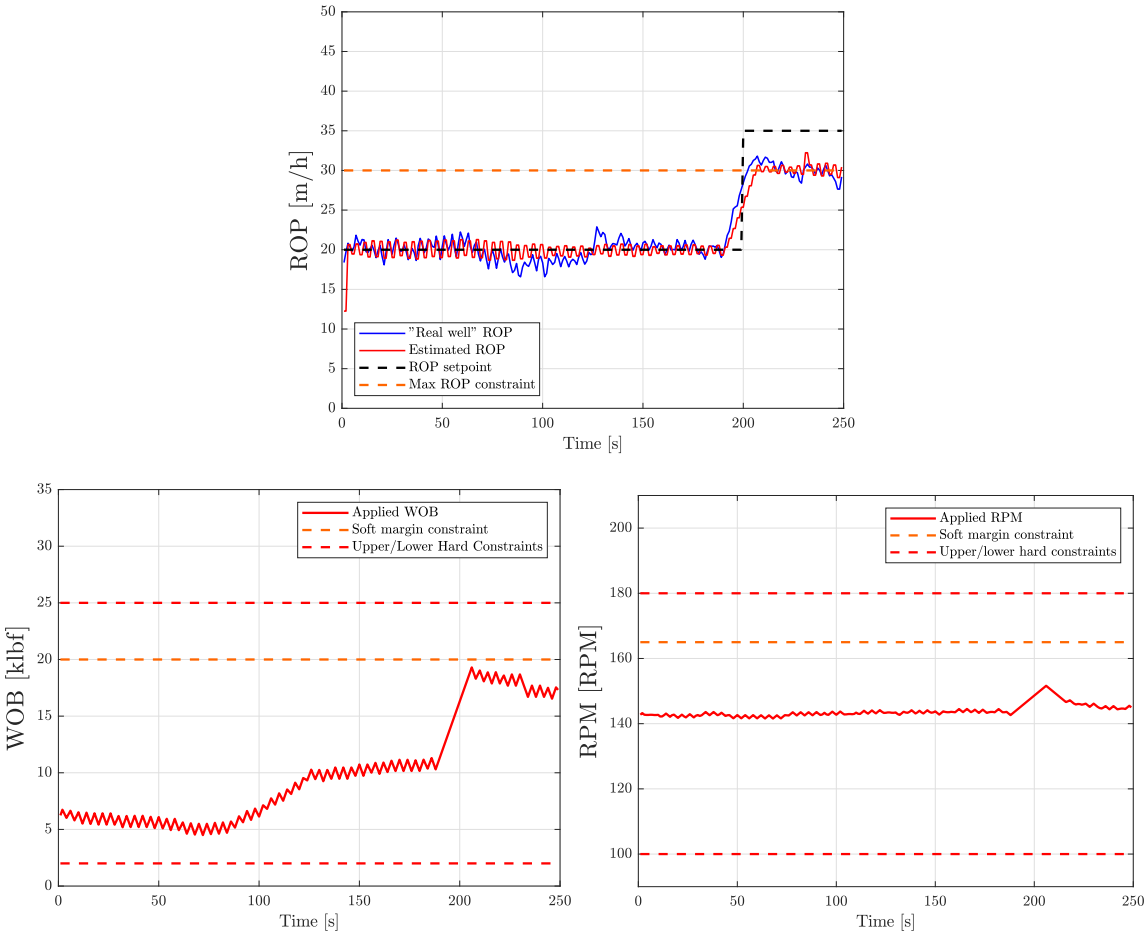


Figure 7.19: Control of real well with no soft constraint to penalize the rate of change of the inputs.

Figure 7.19 shows the consequence of not having a soft constraint (as given in eq. (50)) which is used to limit the rate of change of the control inputs. This was done by setting the weights used to penalize the deviation in the slack variables on control input rate of change, $s_{\delta u_1} = 0$ and $s_{\delta u_2} = 0$. As seen, this causes very rapid fluctuations in the control inputs and unnecessary wear and tear on the actuators.

8 Discussion

It is shown in the results that the developed SLMPC can drive a simulated well ROP to its desired values. There are, however, some limitations and potential weaknesses with the proposed solution that should be addressed. Thus, a discussion on the capabilities and drawbacks of the implemented B&Y-model and the SLMPC is given in the following subsections.

8.1 Bourgoyne and Young ROP-Model

The presented results show that the B&Y-model yields reasonable estimates of the real well ROP when frequently updated by recent measurements. In terms of RMSE, the best estimation result was achieved by using a moving data window of size $N = 30$ with a multiple regression approach. This can further be improved by the use of EKF to estimate the states. However, when looking at the estimated drilling coefficients
$a_j, \forall j = 1, ..., 8$ achieved by using multiple regression (figure 7.5), it is clear that the method cannot be used as a model for the SLMPC. As earlier described, negative drilling coefficients are physically meaningless. Furthermore, their numeric values are far beyond the reasonable ranges given by table 5. Since the moving data window, $\Phi(t)$, only considers 30-100 data points with a spread of 2 seconds (due to measurement taken every 2 seconds under the assumption of wired pipe), the variation in the drilling parameters X_j are nowhere near the recommended ranges given in table 4. This further emphasizes the uncertainty in the drilling coefficients found by the multiple regression method.

An alternative approach is to use the trust-region method to estimate the drilling coefficients a_j. The most significant advantage of the trust-region method is that one might specify boundaries for the parameters to ensure that they are always within the reasonable ranges. Even though the B&Y-model with trust-region method alone (figure 7.6) yields higher RMSE than the multiple regression approach, the parameter estimation yields more meaningful results, part of which is illustrated for a_1 in figure 7.9. Another essential feature of the trust-region method used in the program is that the last estimated drilling coefficients can be used as an initial guess for the trust-region method in the next iteration. This ensures that the contribution from the different drilling parameters on the estimated ROP changes more smoothly over time, instead of the large and rapid changes such as in the multiple regression case. Further, even though the estimated ROP deviates up to approximately 7-8 m/h from the true ROP, it can be sufficient if updated by measurement corrections through the EKF.

It should be stressed that due to lack of variation in some of the input variables as shown in figure 7.1, it is likely that the parameter estimation will still yield wrongful results at least for some of the drilling coefficients. As presented in subsection 3.1.2, Bourgoyne and Young [1999] argues that the coefficients which are based on practically constant drilling parameters X_j might be better off being set constant based on prior knowledge from other drilled wells in the area. Hence, it might be necessary to restrict the parameter estimation to some of the drilling coefficients.

Further, a linear interpolation was used between every real measured value in the provided drilling data to create a uniform drilling data table where the moving data window $\Phi(t)$ could move over. This introduces some "real measured" drilling values, which were never actually measured during the actual drilling process. While it may be a reasonable assumption that the actual value would not differ far from these, it does introduce an extra element of uncertainty in the simulation.

8.2 Successive Linearization based MPC

The first step of testing the SLMPC was done using a static model, where the results is presented in subsection 7.2.1. The drilling coefficients were chosen such that a high ROP ($\sim$ 50-55 m/h) could be obtained within the control input boundaries. The results shows that the static model can be driven to the desired reference value, within the control input boundaries and the max ROP constraint. Some considerations can be given to the results presented in figure 7.10 and 7.11. Two cases are shown, where one is obeying the margin constraints (green), and the other does not (red). Whether or not to include the soft constraints will merely depend on the situation. It would probably, in most cases, be wise to obey the recommendation by the bit producer by not running the max WOB and max RPM simultaneously to reduce wear and tear on the drill bit. For example, suppose that the drill bit is overloaded and has to be replaced. This would involve tripping the whole drill string out of the hole, replacing the bit, and then tripping the whole way back to the total depth of the hole. Such an operation may stretch over days, depending on the length of the drill string, and are thus very costly. On the other hand, if it is at the end of a drilling session, and the drill bit is planned to be replaced before the next drilling session, the soft constraint can be removed to drill the last section faster. It should be noted that only the RPM input obeys the soft constraint. This is chosen by design, to limit only one of the control inputs from running at maximum capacity, and is reflected by the weighting difference in $s_{u_1} = 1, s_{u_2} = 10^4$.

Further, the importance of implementing the max ROP as a soft constraint becomes clear in figure 7.10. Suppose that high concentrations of gas emerges to the platform, forcing the max ROP to immediately drop to 5 m/h at $t = 3000s$. The control inputs would not be able to drop fast enough, due to the constraints on the input rate of change, to obey the constraint. Further, the control inputs are saturated before reaching the max ROP constraint at 5 m/h. This would result in infeasibility and the controller would fail if the max ROP was implemented as a hard constraint. Instead, the max ROP constraint was implemented as a soft constraint, with a weight of $(s_y = 10^6) >> (q = 10^4)$, i.e. weighted 100 times more than the weight on the ROP reference q.

Next, the effect of different prediction horizons N_p is visualized in figure 7.12 and elaborated in table 6. The optimization problem is seen to be solved relatively efficient, with the average solving time of all prediction horizons being well within the sampling time of the SLMPC. While a small prediction horizon of $N_p = 5$ is solved on an average of 6.2 ms by the Matlab solver `quadprog`, it has quite poor predictive capabilities. On the other hand, a very large prediction horizon of $N_p = 45$ has far greater predictive capabilities at the cost of an average solving time as high as 450 ms. There are, however, numerous ways to improve the average solving time. While the optimization problem is easily formulated and solved using the Matlab `quadprog` solver, there exist other solvers that can solve the optimization problem faster. Further, modifying the optimization problem formulation can yield faster solving time. For example, the QP-formulation used in this thesis has equal prediction- and control horizon length. However, as presented in subsection 2.2.2, this may impose a higher computational complexity on the optimization problem than necessary. Hence, the average solving time can be reduced by formulating a separate and shorter control horizon. Another technique is called input blocking and involves forcing the control inputs to be constant over several time steps, thus reducing the number of decision variables, and is visualized in Foss and Aksel N. Heirung [2016]. For now, however, a prediction horizon of around 10-15 steps seems to suffice as a trade-off between time complexity and predictive capabilities.

A verification strategy was developed to determine the performance of the SLMPC when being continuously updated by measurements from a simulated real well with varying formation properties. The results presented in subsection 7.2.2 suggests that all the different model setups can control the real well ROP towards the ROP reference with varying degrees of success. Using just a static model without updating the MPC-model from online measurements yields the highest RMSE as shown in table 7. However, the performance of the static model will highly depend on the accuracy of the initial guess of the static model drilling coefficients relative to the real well coefficients and to what degree the formation properties vary. Needless to say, the SLMPC model should be updated against real well measurements as it is already stated that the B&Y-model is a simplification of the drilling process and thus will deviate from the actual real well response.

The RMSE is significantly improved when using the trust-region method to estimate the drilling

coefficients from a moving data window continuously updated by the control inputs and the real well response. There is, however, an imminent chance that the RMSE between the modeled ROP and the simulated well ROP is lower than it would be for a real world application. As seen in the B&Y-model using TR-method on the actual drilling data given in figure 7.7, the RMSE is approximately twice as large as for the simulated real well in figure 7.15. This captures the weakness of the verification strategy. Since the simulated real well ROP is based on the same model as the modeled ROP, only differing in the varying drilling coefficients a_j/a_j^w and measured depth $\hat{h}/h_W$, then the modeled ROP is likely to have a larger correlation to the simulated real well response than to an actual real well application. Furthermore, the simulated real well drilling coefficients a_1^W, a_5^W, and a_6^W were altered randomly, within their realistic ranges, to simulate the effects of a varying formation. This might yield unrealistic combinations of drilling coefficients which in addition potentially varies at unrealistic rates. It does, however, introduce and element of uncertainty and does illustrate the importance of updating the model used in the SLMPC by measurements from the real well response.

The extended Kalman filter further reduced the RMSE as seen in table 7. The best result, in terms of RMSE, was achieved by using $Q_d = diag([1 \quad 10])$, $\hat{P}^-(0) = diag([1 \quad 10])$, and $R_d = diag([1 \quad 1])$ and is seen in figure 7.16. A higher process noise and uncertainty in the state estimate e is assumed, as its propagation is solely based on being updated by measurement corrections as seen in section 4.2.2. It is, therefore, reasonable to trust the measurements more than the state estimate. However, due to the very nonuniform formation properties, it would require particularly active control inputs that could cause wear and tear on the actuators. By increasing the measurement uncertainty with $R_d = diag([1 \quad 1000])$, the most rapid changes in the ROP are ignored while the trends are still captured, which in turn requires less active actuators as seen in figure 7.17. If necessary, the input rate of change weights $s_{\delta u_1}$ and $s_{\delta u_2}$ can be increased to yield even more conservative actuation as seen in figure 7.18, at the cost of larger deviations from the desired ROP value. Removing these weights, however, may cause a severe fluctuation about the reference value which may be harmful to the actuators and potentially the drilling equipment.

Since there were no state models for the ECD, p_{af}, or the flow rate, q, these were set constant during the simulation with the SLMPC. It is, however, likely that the ECD would increase with larger RPM and as a function of the amount of transported cuttings in the mud as described in subsection 2.1.4. Since both the ECD and the flow rate are important factors for the pressure control in the wellbore, it would make sense to augment the state model to include these. This would potentially further limit the max ROP, and is in the program developed in this thesis assumed to be incorporated in the max ROP constraint.

Lastly, some thoughts on the difference between using a successive linearization based MPC approach as opposed to a nonlinear MPC approach. The obvious advantage of linearizing the model at each time step is to obtain the convexity property as presented in subsection 2.2.3, thus allowing highly efficient solving of the optimization problem. While the developed SLMPC had an average solving time of 36.9 ms with a prediction horizon of $N_p = 15$, Sui et al. [2013] reported an average solving time of 70 ms with the use of a nonlinear MPC with a prediction horizon of $N_p = 7$, fewer decision variables, and seemingly fewer constraints. Further, the convex QP problem is a well-documented area of research with highly efficient solvers available. However, the nonlinear MPC approach allows for nonlinear constraints to be directly formulated in the optimization problem. In the developed SLMPC, the max ROP constraint is assumed to be externally calculated by Halliburton's *Max ROP* and applied to the SLMPC as a linear constraint. Hence, the predictive capabilities of the SLMPC do not apply to the max ROP constraint. If these calculations were calculated directly as a nonlinear constraint on the max ROP, some predictive capabilities could be obtained and ultimately respond faster to unforeseen events. However, the feasibility of such a solution in terms of solving time and model complexity is unknown.

9 Conclusion and Future Work

9.1 Conclusion

This thesis presents results suggesting that a successive linearization-based MPC might be a promising candidate to control the drilling procedure. A Bourgoyne & Young ROP-model was established, where the drilling coefficients were found by parameter estimation techniques based on a moving data window. While the parameter estimation was performed both through a multiple regression method and a trust-region method, only the trust-region approach provided drilling coefficients within physically meaningful ranges and was thus the preferred method. Furthermore, the model was linearized at every time step to formulating a convex QP problem that the MPC could effectively solve.

Thus, an SLMPC-problem was formulated based on the ROP operating mode, where the control objective is to drive the ROP to the desired reference value, according to a pre-determined drilling plan, while obeying the imposed operational and safety constraints.

Further, a verification strategy was developed to test the SLMPC performance against a simulated "real well" response. The MPC model was updated from online measurements, both in terms of historical measurements in the moving data window and by the real-time measurements done in the extended Kalman filter. A prerequisite for the SLMPC scheme developed in this thesis is that a wired drill pipe is used, as the measurements in the simulation were done every two seconds. The results then showed that the SLMPC successfully controlled the simulated well to the reference value or the best obtainable value based on the constraints and the chosen weighting matrices.

While the simulation results suggest that the developed SLMPC can control the drilling process efficiently, according to the specified operating mode, it cannot be said to be done without compromising the safety in terms of pressure control. Therefore, augmenting the state-space model with states necessary to obtain proper pressure control might be a prerequisite for a fully autonomous operation, with the driller in a supervisory role.

9.2 Future Work

Suggestions to some potential improvements or next steps are provided in the following bulletpoints

- Formulate a separate control horizon or apply input blocking in the optimization problem developed in subsection 4.3 to further improve the average solving time of the SLMPC. Further, other optimization solvers should be considered.
- Augment the state space model to include ECD and flow rate to improve safety in terms of increased pressure control
- Evaluate, or develop methods to evaluate which drilling parameters X_j that should be excluded from the parameter estimation and thus which drilling coefficients a_j should be chosen based on prior knowledge of the drilling conditions and kept constant during the drilling process.
- Simulate and test the SLMPC in a proper well simulator environment.
- Formulate and develop an MPC for the rest of the operating modes that is mentioned in section 4 and further elaborated in Kommedal [2020].
- Formulate and develop a nonlinear MPC to test the performance relative to a SLMPC strategy.

Bibliography

AkerBP ASA. Valhall-området — Aker BP ASA. URL https://akerbp.com/asset/valhall/.

Abouzar Bahari and Abolfazl Baradaran Seyed. Trust-Region Approach To Find Constants of Bourgoyne and Young Penetration Rate Model in Khangiran Iranian Gas Field. In *All Days*. SPE, 4 2007. doi: 10.2118/107520-MS. URL https://onepetro.org/SPELACP/proceedings/07LACPEC/All-07LACPEC/Buenos%20Aires,%20Argentina/142079.

Alberto Bemporad, N Lawrence Ricker, and Manfred Morari. Model Predictive Control Toolbox™ User's Guide. Technical report, 2005. URL https://www.mathworks.com/help/pdf_doc/mpc/mpc_ug.pdf.

A. T. Bourgoyne and F. S. Young. A multiple regression approach to optimal drilling and abnormal pressure detection. *SPE Reprint Series*, 14(49):27–36, 8 1999. ISSN 08910901. doi: 10.2118/4238-pa.

A.T. Bourgoyne, K.K. Millhelm, M.E Chenevert, and F.S Young Jr. *Applied Drilling Engineering*, volume 2. Society of Petroleum Engineers, 1991. URL https://store.spe.org/Applied-Drilling-Engineering-P10.aspx.

R Caenn, HCH Darley, and GR Gray. *Composition and properties of drilling and completion fluids*. Sixth edition, 2011. ISBN 978-0-12-383858-2. URL https://books.google.com/books?hl=no&lr=&id=ohZanicqpmYC&oi=fnd&pg=PP1&ots=EbBzXoghDT&sig=KexvOneHhXluBJ57zzNg6rfV0p0.

E F Camacho and C Bordons. *Model Predictive Control*. Springer-Verlag London, second edition, 2007. ISBN 978-1-85233-694-3. doi: 10.1007/978-0-85729-398-5. URL https://books.google.com/books?hl=no&lr=&id=tXZDAAAAQBAJ&oi=fnd&pg=PA13&dq=model+predictive+control&ots=L2RwiDBWc1&sig=89aW3mZKi-OTvjBBq3CGPyF0cVQ.

John R. Eckel and W.J. Bielstein. Nozzle Design and its Effect on Drilling Rate and Pump Operation. Technical report, 1 1951. URL http://onepetro.org/APIDPP/proceedings-pdf/API51/All-API51/API-51-028/2066344/api-51-028.pdf.

Stephen T. Edwards, Christopher J. Coley, Nicholas A Whitley, Richard G. Keck, Vishwahnath Ramnath, Tommy Foster, Keith Coghill, and Mark Honey. A Summary Of Wired Drill Pipe (IntelliServ Networked Drillstring) Field Trials And Deployment In Bp. pages 5–7. Society of Petroleum Engineers (SPE), 3 2013. doi: 10.2118/163560-ms.

Tuna Eren and M. Evren Ozbayoglu. Real time optimization of drilling parameters during drilling operations. In *SPE Oil and Gas India Conference and Exhibition 2010, OGIC*, volume 2010-January. Society of Petroleum Engineers (SPE), 1 2010. ISBN 9781617382116. doi: 10.2118/129126-ms.

Farah Omar Farah. DIRECTIONAL WELL DESIGN, TRAJECTORY AND SURVEY CALCULATIONS, WITH A CASE STUDY IN FIALE, ASAL RIFT, DJIBOUTI. Technical report, Ministry of Energy and Natural Resources, 2013. URL http://www.os.is/gogn/unu-gtp-report/UNU-GTP-2013-27.pdf.

Bjarne Foss and Tor Aksel N. Heirung. Merging Optimization and Control. Technical report, Faculty of Information Technology, Mathematics, and Electrical Engineering Department of Engineering Cybernetics, 2016. URL https://www.researchgate.net/publication/301685613.

Thor I. Fossen. *Handbook of Marine Craft Hydrodynamics and Motion Control*. John Wiley & Sons, Ltd, Chichester, UK, 4 2011. ISBN 9781119994138. doi: 10.1002/9781119994138. URL http://doi.wiley.com/10.1002/9781119994138.

Morten Hovd. Lecture notes for the course Advanced Control of Industrial Processes, 11 2009. URL https://folk.ntnu.no/skoge/presentation/OLD-2017-and-earlier/plantwide-course-brasil-july2011/Hovd-Kompendium-2010.pdf.

T Inglis. *Directional drilling*, volume 2. Springer Science & Business Media, 2013, 2013. ISBN 9401712700, 9789401712705. URL https://books.google.com/books?hl=no&lr=&id=zM_rCAAAQBAJ&oi=fnd&pg=PR7&dq=directional+drilling&ots=BCTzzbRghr&sig=YgFhB6w6loALJdWr-uLF_w5FGnQ.

R. E. Kalman. A new approach to linear filtering and prediction problems. *Journal of Fluids Engineering, Transactions of the ASME*, 82(1):35–45, 3 1960. ISSN 1528901X. doi: 10.1115/1.3662552. URL http://asmedigitalcollection.asme.org/fluidsengineering/article-pdf/82/1/35/5518977/35_1.pdf.

Gregory King. 8.4.3.2: Semi-Submersible Rig — PNG 301: Introduction to Petroleum and Natural Gas Engineering, 10 2020. URL https://www.e-education.psu.edu/png301/node/909.

Anders Kommedal. TTK4551 - Model Predictive Control in Drilling Operations. Technical report, Norwegian University of Science and Technology, Trondheim, 12 2020.

NJ Lapeyrouse. *Formulas and calculations for drilling, production and workover*. Gulf Professional Publishing, 2nd edition, 11 2002. ISBN 9780080506760. URL https://books.google.com/books?hl=no&lr=&id=U1iGiLpF-MEC&oi=fnd&pg=PP9&dq=Lapeyrouse,+N.J.,+2002.+Formulas+and+calculations+for+drilling,+production+and+workover,+Boston:+Gulf+Professional+publishing.&ots=AJ5KFAM22Q&sig=1zx8NYQmaXGqUeuV_tn52tMlitg.

Kenneth P. Malloy, Rick Stone, George Harold Medley, Don M. Hannegan, Oliver D. Coker, Don Reitsma, Helio Mauricio Santos, Joseph Irvin Kinder, Johan Eck-Olsen, John Walton McCaskill, James Ronald May, Kenneth L. Smith, and Paul Sonnemann. Managed-Pressure Drilling: What It Is and What It Is Not. In *IADC/SPE Managed Pressure Drilling and Underbalanced Operations Conference & Exhibition*, pages 12–13. Society of Petroleum Engineers, 4 2009. doi: 10.2118/122281-MS. URL http://www.onepetro.org/doi/10.2118/122281-MS.

MathWorks inc. MATLAB - MathWorks - MATLAB & Simulink. URL https://se.mathworks.com/products/matlab.html.

D. Q. Mayne, J. B. Rawlings, C. V. Rao, and P. O.M. Scokaert. Constrained model predictive control: Stability and optimality. *Automatica*, 36(6):789–814, 6 2000. ISSN 00051098. doi: 10.1016/S0005-1098(99)00214-9.

J Møgster, J-M Godhavn, and L Imsland. Using MPC for Managed Pressure Drilling. *Identification and Control*, 34(3):131–138, 2013. doi: 10.4173/mic.2013.3.3.

Jorge J. Moré and D. C. Sorensen. Computing a Trust Region Step. *SIAM Journal on Scientific and Statistical Computing*, 4(3):553–572, 9 1983. ISSN 0196-5204. doi: 10.1137/0904038. URL https://epubs.siam.org/page/terms.

Saleh M. Mwachaka, Aiping Wu, and Qingqing Fu. A review of mud pulse telemetry signal impairments modeling and suppression methods, 3 2019. ISSN 21900566. URL https://doi.org/10.1007/s13202-018-0483-y.

Sagar Nauduri, George H. Medley, and Jerome J. Schubert. MPD: Beyond narrow pressure windows. In *Society of Petroleum Engineers - IADC/SPE Managed Pressure Drilling and Underbalanced Operations Conference and Exhibition 2009 - Drilling in India: Challenges and Opportunities*, pages 171–177. Society of Petroleum Engineers (SPE), 1 2009. ISBN 9781605609553. doi: 10.2118/122276-ms.

Amirhossein Nikoofard, Tor Arne Johansen, Hessam Mahdianfar, and Alexey Pavlov. Constrained MPC design for heave disturbance attenuation in offshore drilling systems. In *OCEANS 2013 MTS/IEEE Bergen: The Challenges of the Northern Dimension*, 2013. ISBN 9781479900015. doi: 10.1109/OCEANS-Bergen.2013.6608196.

Jorge Nocedal and Stephen J. Wright. *Numerical Optimization*. Springer Series in Operations Research and Financial Engineering. Springer New York, second edition edition, 2006. ISBN 978-0-387-30303-1. doi: 10.1007/978-0-387-40065-5. URL http://link.springer.com/10.1007/978-0-387-40065-5.

Norsk Oljemuseum. Valhall IP - Norsk Oljemuseum. URL https://www.norskolje.museum.no/3178_438998414e67401187b59962a69d606f-jpg/.

Gro D. Øfjord, Mathilde J. Lind, and Bent B. Skaare. Håndtering av borekaks i områder med sårbar bunnfauna. Technical report, Klima- og forurensingsdirektoratet (Klif), 11 2012. URL https://www.miljodirektoratet.no/globalassets/publikasjoner/klif2/publikasjoner/2984/ta2984.pdf.

Nathan Pastorek, Katherine R Young, and Alfred Eustes. Downhole Sensors in Drilling Operations. In *Conference: Presented at the 44th Workshop on Geothermal Reservoir Engineering, 11-13 February 2019, Stanford, California*, pages 1–9. Stanford University, 2 2019.

Matija Perne, Samo Gerkšič, and Boštjan Pregelj. Soft inequality constraints in gradient method and fast gradient method for quadratic programming. *Springer Science+Business Media, LLC, part of Springer Nature 2018*, pages 751–753, 12 2018. doi: 10.1007/s11081-018-9416-3. URL https://doi.org/10.1007/s11081-018-9416-3.

DJ Reddish and BN Whittaker. *Subsidence: occurrence, prediction and control*. Elsevier Science, 1st edition, 1 1989. ISBN 9780444598349. URL https://books.google.com/books?hl=no&lr=&id=J9B-iaMoUNwC&oi=fnd&pg=PP1&dq=subsidence&ots=IBW7DzT69K&sig=9if59Y3OV57WVJgrRULwfGykPXI.

B Rehm, J Schubert, A Haghshenas, and AS Paknejad. *Managed pressure drilling*. Gulf Publishing Company, 1st edition, 12 2009. ISBN 9780127999876. URL https://books.google.com/books?hl=no&lr=&id=NXejAQAAQBAJ&oi=fnd&pg=PP1&dq=managed+pressure+drilling&ots=NsskNqIt_t&sig=QzJxaxVYpwXFUHRSMF0q1Pzsbk0.

Rigzone. How Do Semisubmersibles Work? — Rigzone. URL https://www.rigzone.com/training/insight.asp?insight_id=338&c_id=.

Schlumberger. telemetry — Oilfield Glossary. URL https://www.glossary.oilfield.slb.com/en/terms/t/telemetry.

Hiroya Seki, Satoshi Ooyama, and Mosimasa Ogawa. Nonlinear Model Predictive Control Using Successive Linearization Application to Chemical Reactors. Technical report, Trans. of the Society of Instrument and Control Engineers, 2004. URL https://sice.sakura.ne.jp/e-trans/papers/E3-9.pdf.

Jing Shao, Zhidan Yan, Suli Han, Hui Li, Tingzheng Gao, Xiufeng Hu, and Chunming Wei. Differential signal extraction for continuous wave mud pulse telemetry. *Journal of Petroleum Science and Engineering*, 148:127–130, 1 2017. ISSN 09204105. doi: 10.1016/j.petrol.2016.09.047. URL http://dx.doi.org/10.1016/j.petrol.2016.09.047.

Espen Søbye. Borekostnadene på sokkelen: Knask eller knep? - Statistisk Sentralbyrå, 2 2017. URL https://www.ssb.no/energi-og-industri/artikler-og-publikasjoner/borekostnadene-pa-sokkelen-knask-eller-knep.

Øyvind Nistad Stamnes. *Øyvind Nistad Stamnes Nonlinear Estimation with Applications to Drilling*. Skipnes, 2011. ISBN 9788247126455. URL https://ntnuopen.ntnu.no/ntnu-xmlui/handle/11250/260303.

Dan Sui, Roar Nybo, and Vahid Azizi. Real-time optimization of rate of penetration during drilling operation. In *IEEE International Conference on Control and Automation, ICCA*, pages 357–362, 2013. ISBN 9781467347075. doi: 10.1109/ICCA.2013.6564893.

Howard L. Taylor and C. Mack Mason. SYSTEMATIC APPROACH TO WELL SURVEYING CALCULATIONS. *Soc Pet Eng J*, 12(6):474–488, 12 1972. ISSN 0197-7520. doi: 10.2118/3362-pa. URL http://onepetro.org/spejournal/article-pdf/12/06/474/2157061/spe-3362-pa.pdf.

World Wide Web Consortium (W3C). XML Essentials - W3C. URL https://www.w3.org/standards/xml/core.

Altay Zhakatayev, Bexultan Rakhim, Olzhas Adiyatov, Almaskhan Baimyshev, and Huseyin Atakan Varol. Successive linearization based model predictive control of variable stiffness actuated robots. In *IEEE/ASME International Conference on Advanced Intelligent Mechatronics, AIM*, pages 1774–1779. Institute of Electrical and Electronics Engineers Inc., 8 2017. ISBN 9781509059980. doi: 10.1109/AIM.2017.8014275. URL https://www.researchgate.net/publication/319122854_Successive_Linearization_Based_Model_Predictive_Control_of_Variable_Stiffness_Actuated_Robots.

Appendix

A Table of Dates Website References was Accessed

Due to difficulties of including the date a website reference was accessed with the chosen Latex reference framework, this table is included to provide this information.

As referenced in Bibliography	Date accessed
AkerBP ASA. Valhall-området	05/09/2020
Rigzone. How Do Semisubmersibles Work?	18/10/2020
Norsk Oljemuseum. Valhall IP	18/10/2020
Gregory King. 8.4.3.2: Semi-Submersible Rig	18/10/2020
IADC. Drilling Manual Drilling mechanics and performance.	08/11/2020
Schlumberger. telemetry	18/11/2020
World Wide Web Consortium (W3C). XML Essentials	24/04/2021
MathWorks inc. MATLAB - MathWorks - MATLAB & Simulink.	27/04/2021
Borekostnadene på sokkelen: Knask eller knep? - SSB	24/05/2021

Table 8: Dates of when the website references was accessed.

B Matlab Code

B.1 Simulating Varying Formation Properties

```
%% Simulating a non-uniform varying formation

awell(1) = awell(1) + randn()*0.01;
awell(5) = awell(5) + randn()*0.01;
awell(6) = awell(6) + randn()*0.01;

if awell(1) > 1.9
    awell(1) = 1.9;
elseif awell(1) < 1.5
    awell(1) =1.5;
end
if awell(5) > 0.7
    awell(5) = 0.7;
elseif awell(5) < 0.5
    awell(5) =0.5;
end
if awell(6) > 1.0
    awell(6) = 1.0;
elseif awell(6) < 0.4
    awell(6) =0.4;
end
```

B.2 Opening and Extracting Data from Drilling Data Files

```
%% Open and Extract Data From Files

% 0: Use data from specified interval, 1: Use all data
SimulateAll = 0;

FileToRead = 'WellData.csv';
FromDate   = '2018-03-20';
ToDate     = '2018-03-20';
FromTime   = '05:34:00';
ToTime     = '07:01:00';

% Import .csv-file properly.
opts     = detectImportOptions(FileToRead);
opts     = setvartype(opts, 'TIME', 'string');
WellData = readtable(FileToRead, opts);


%% Extract time on correct datetime format YYYY-MM-DDTHH:MM:SS

if SimulateAll
    t1 = 1;
    t2 = height(WellData);
    Time=datetime(extractBetween(WellData.TIME(:),"","."),...
        'Format','yyyy-MM-dd''T''HH:mm:ss');
else
    [t1,t2] = FindTimeIndexes(WellData,FromDate,FromTime,ToDate,ToTime)
    Time=datetime(extractBetween(WellData.TIME(t1:t2),"","."),...
        'Format','yyyy-MM-dd''T''HH:mm:ss');
end


%% Creating Interpolated Data Sample (Uniform Data at 1 Hz)

InterData = InterpolateTable(WellData(t1:t2,:),Time);
```

B.3 Find Time Indexes

```
% Find data between specified date and time
% Date format: 'YYYY-MM-DD', Time format: 'HH:MM:SS'
function [t1, t2] = FindTimeIndexes(G22A,FromDate,FromTime,ToDate,ToTime)

FromDateTime = sprintf([FromDate 'T' FromTime '.000+01:00']);
ToDateTime   = sprintf([ToDate 'T' ToTime '.000+01:00']);

t1 = find(G22A{:,1} == FromDateTime);
t2 = find(G22A{:,1} == ToDateTime);

end
```

B.4 Interpolating the Drilling Data Table

```
function InterpolatedTable = InterpolateTable(PreData,Time)
% Function made to create the interpolated data table

% Data prior to interpolation. Zeroes are added in first and last row
% to ensure that there exist no NaN data in these rows prior to the
% interpolation
PreData{1,2:19} = 0;
PreData{height(PreData),2:19} = 0;

% Dimensions of the new interpolated table.
InterHeight = height(resample(PreData{:,2}...
    (PreData{:,2} > -999),Time(PreData{:,2} > -999),1,1,1));
InterWidth = width(PreData);

% Variable types of new interpolated table
VarTypes = {'datetime','double','double','double','double',...
    'double','double','double','double','double','double','double',...
    'double','double','double','double','double','double','double'};

% Defining the new interpolated table
InterpolatedTable = table('Size',[InterHeight InterWidth]...
    ,'VariableTypes',VarTypes);
InterpolatedTable.Properties.VariableNames = ...
    PreData.Properties.VariableNames;

% Removing all "NaN" values, and creating the interpolated data using
% Matlab function resample()
for i = 2:19
    [InterpolatedTable{:,i},InterpolatedTable{:,1}] = ...
        resample(PreData{:,i}(PreData{:,i} > -999),...
        Time(PreData{:,i} > -999),1,1,1); %Interpolated data (1 hz)
end

% Omit the first and last two rows of the interpolated data
InterpolatedTable(1:3,:) = [];
InterpolatedTable(height(InterpolatedTable)-2:...
    height(InterpolatedTable),:) = [];
end
```

B.5 Converting from Measured Depth to True Vertical Depth

```
function TVD = MDtoTVD(MD)
% Converts MD to TVD using Angle Averaging Method from MD1 to MD2
% i.e. does not include curvature.

global MDvsTVD

% Finding the index %
[ ~, index] = min( abs( MDvsTVD.MD(:)-MD ) );

if MDvsTVD.MD(index) > MD
    index = index - 1;
end

% TVD calculation based on current MD %
TVD = MDvsTVD.TVD(index)...
    + (MD - MDvsTVD.MD(index))*(cosd((MDvsTVD.INCLINATIONANGLE(index)...
    + MDvsTVD.INCLINATIONANGLE(index + 1))/2));
end
```

B.6 Finding the Drilling Coefficients $a_j,\ \forall j = 1, ..., 8$ Through MR and TR Methods

```
function sol = FindDrillingCoefficients(DataSet,Method)
%Function for Finding a(j) For All j = 1,...,8

persistent a0

%% Determining a(j) For All j = 1,...,8 Through Multiple Regression
if Method
    syms a1 a2 a3 a4 a5 a6 a7 a8;

    eq1 = a1*sum(DataSet.X1(:)) + a2*sum(DataSet.X2(:)) + a3*sum(DataSet.X3(:)) + a4
    eq2 = a1*sum(DataSet.X2(:)) + a2*sum(DataSet.X2(:).*DataSet.X2(:)) + a3*sum(Data
    eq3 = a1*sum(DataSet.X3(:)) + a2*sum(DataSet.X2(:).*DataSet.X3(:)) + a3*sum(Data
    eq4 = a1*sum(DataSet.X4(:)) + a2*sum(DataSet.X2(:).*DataSet.X4(:)) + a3*sum(Data
    eq5 = a1*sum(DataSet.X5(:)) + a2*sum(DataSet.X2(:).*DataSet.X5(:)) + a3*sum(Data
    eq6 = a1*sum(DataSet.X6(:)) + a2*sum(DataSet.X2(:).*DataSet.X6(:)) + a3*sum(Data
    eq7 = a1*sum(DataSet.X7(:)) + a2*sum(DataSet.X2(:).*DataSet.X7(:)) + a3*sum(Data
    eq8 = a1*sum(DataSet.X8(:)) + a2*sum(DataSet.X2(:).*DataSet.X8(:)) + a3*sum(Data

    sol = solve([eq1, eq2, eq3, eq4, eq5, eq6, eq7, eq8],...
        [a1, a2, a3, a4, a5, a6, a7, a8]);

    sol = [sym2poly(sol.a1) sym2poly(sol.a2) sym2poly(sol.a3) ...
           sym2poly(sol.a4) sym2poly(sol.a5) sym2poly(sol.a6) ...
           sym2poly(sol.a7) sym2poly(sol.a8)]';



    %% Determining a(j) For All j = 1,...,8 Through Trust-Region Method
    % min  0.5*(NORM(C*x-d)).^2 with upper and lower bounds using TR method
    %  x
else

    C = [sum(DataSet.X1(:)) sum(DataSet.X2(:)) sum(DataSet.X3(:)) sum(DataSet.X4(:))
        sum(DataSet.X2(:)) sum(DataSet.X2(:).*DataSet.X2(:)) sum(DataSet.X3(:).*Data
        sum(DataSet.X3(:)) sum(DataSet.X2(:).*DataSet.X3(:)) sum(DataSet.X3(:).*Data
        sum(DataSet.X4(:)) sum(DataSet.X2(:).*DataSet.X4(:)) sum(DataSet.X3(:).*Data
        sum(DataSet.X5(:)) sum(DataSet.X2(:).*DataSet.X5(:)) sum(DataSet.X3(:).*Data
        sum(DataSet.X6(:)) sum(DataSet.X2(:).*DataSet.X6(:)) sum(DataSet.X3(:).*Data
        sum(DataSet.X7(:)) sum(DataSet.X2(:).*DataSet.X7(:)) sum(DataSet.X3(:).*Data
        sum(DataSet.X8(:)) sum(DataSet.X2(:).*DataSet.X8(:)) sum(DataSet.X3(:).*Data

    d = [sum(log(DataSet.ROP{:,:}));
        sum(log(DataSet.ROP{:,:}).*DataSet.X2(:));
        sum(log(DataSet.ROP{:,:}).*DataSet.X3(:));
        sum(log(DataSet.ROP{:,:}).*DataSet.X4(:));
        sum(log(DataSet.ROP{:,:}).*DataSet.X5(:));
        sum(log(DataSet.ROP{:,:}).*DataSet.X6(:));
        sum(log(DataSet.ROP{:,:}).*DataSet.X7(:));
        sum(log(DataSet.ROP{:,:}).*DataSet.X8(:))];

    % Lower and Upper Bounds on a(j) %
    %     a1     a2      a3       a4      a5  a6  a7  a8
    lb = [0.5;0.000001;0.000001;0.000001;0.5;0.4;0.3;0.3];
    ub = [1.9;0.0005  ;0.0009  ;0.0001  ;2.0;1.0;1.5;0.6];

    options = optimoptions('lsqlin','Algorithm'...
        ,'trust-region-reflective','Display','Off');
    sol = lsqlin(C,d,[],[],[],[],lb,ub,a0,options);
    a0 = [sol(1) sol(2) sol(3) sol(4) sol(5) sol(6) sol(7) sol(8)]';

end



end
```

B.7 Finding the Matrices; P, H, E, and Δ

```
function P = FindP(At,Ct,Np)
% Find P matrix
% Inspired by obsv() matrix (J.C.Slater, J.A. Leitner)

nx = size(At,1);
ny = size(Ct,1);

% Allocate Fc and compute each C A^k term
P = zeros(Np*ny,nx);
P(1:ny,:) = Ct;

for k=1:Np-1
    P(k*ny+1:(k+1)*ny,:) = P((k-1)*ny+1:k*ny,:) * At;
end

end
```

```
function H = FindH(At,Bt,Ct,Dt,Np)
% Find H matrix

ny = size(Ct,1);
nu = size(Bt,1);

H = kron(eye(Np),Dt);

for k = 2:Np*ny
    indx = 1;
    for j = 1:2:Np*nu
        if k-1 >= indx
            H(k,j:j+1) = Ct*At^(k-1-(indx))*Bt;
        end
        indx = indx + 1;
    end
end
end
```

```
function E = FindE(At,Ct,Np)
% Find E matrix

nx = size(At,1);
ny = size(Ct,1);

E = zeros(Np*ny,nx);

E(1:ny,:) = 0;
E(2,:) = Ct;


for k = 3:Np*ny
    SumOfA = 0;
    for i = 1:k-2
        SumOfA = SumOfA + At^i;
    end
    E(k,:) = Ct + Ct*SumOfA;
end

end
```

```
function Delta = FindDelta(Np,nu)
% Find Delta matrix
%
% kronD produces
%
%          |I 0 .. 0|
%          |I I .. 0|
% kronD =  |. . .. .| , size(kronD) = [N*nu*nu,N*nu*nu]
%          |I I .. I|
%
% Delta extracts the upper left quadrant of the matrix
%
%          |I 0 .. 0|
%          |I I .. 0|
% Delta =  |. . .. .| , size(Delta) = [N*nu,N*nu]
%          |I I .. I|
%


kronD = kron(tril(ones(Np*nu)),eye(nu));

Delta = kronD(1:Np*nu,1:Np*nu);

end
```

B.8 Finding the Drilling Parameters $X_1, X_2, ..., X_8$ at Time t

```
function [X,w,r,gp,gp0,ECD,w0] = DrillingParameters(h,t,InterData)
% Provides X1,X2,...,X8 and other drilling parameters
% for time t used in ROP estimate

global PPFG
%% Constants and variables

if(length(class(h)) < 6)
    h = sym2poly(h);                % Ensures numeric value for h
end

h0TVD = MDtoTVD(InterData{1,2});    % Point of normalization for TVD [m]
hMD = h;                            % Measured depth [m]
hTVD = MDtoTVD(h);                  % True Vertical Depth [m]
ECD = InterData{t,14};              % ECD at bit [ppg]
gp0 = 9;                            % Point of normalization of PP [ppg]
w = InterData{t,8};                 % Weight on Bit [klbf]
w0 = 0;                             % Bit weight threshold before
                                    %   the bit is effective
r = InterData{t,10};                % Surface RPM [RPM]
r0 = 100;                           % Point of normalization of RPM [RPM]
q = InterData{t,12};                % Mud flow [L/min]
d_B = 12.25;                        % Bit outer diameter [in]
H = 0;                              % Bit tooth wear
rho = 14.6;                         % Mud Weight
A_T = (15^2 + 15^2 + 15^2 + ...     % Total area flow [in^2]
       15^2 + 16^2 + 16^2 + ...
       16^2)*pi*(1/(4*32^2));
Cd = 0.95;                          % Discharge coefficient
LPMtoGPM = 0.26417;                 % Conversion factor for L/min
                                    %  to gallons/min
PD = (8.311e-5)*...                 % Bit pressure drop[lbs/in^2]
    (rho*(q*LPMtoGPM)^2)/...
    ((Cd^2)*(A_T^2));
Fj = 0.01823*Cd*(q*LPMtoGPM)*...    % Jet impact force [lbf]
    (rho*PD)^.5;
Fj0 = 1000;                         % Point of normalization for jet impact
                                    %  force
%% Finding approximate PP gradient corresponding to given measured depth

[ ~, PPindex] = min( abs(PPFG.depth(:) - hMD));
gp = PPFG.MLPP(PPindex,1);      % Pore pressure gradient [ppg]

%% Drilling Parameters
X(1) = 1;
X(2) = distdim(h0TVD-hTVD,'m','ft');
X(3) = (distdim(hTVD,'m','ft')^(0.69))*(gp - gp0);
X(4) = distdim(hTVD,'m','ft')*(gp - ECD);
X(5) = log((w/d_B - w0/d_B)/(4 - w0/d_B));
X(6) = log(r/r0);
X(7) = -H;
X(8) = log(Fj/Fj0);

end
```

B.9 Finding the Dataset $\Phi(t)$

```
function DataSet = CreateDataSet(N,t,InterData)
% Creating the moving window data set
% Creating a data set of N steps, from timestep t.

global PPFG
DataSpread = 2;                         % Spread in the historical data
                                        % measurements
% Spread of historic data if desired
SetHeight = t-N*DataSpread:DataSpread:t-DataSpread;

%% Constants and variables
h0MD = table(ones(length(SetHeight),1)*InterData{1,2}); % Normalized MD
h0TVD = table(ones(length(SetHeight),1)*InterData{1,2});% Normalized TVD
hMD = InterData(SetHeight,2);           % Measured depth [m]
hTVD = zeros(height(hMD),1);            % True Vertical Depth [m]
ECD = InterData(SetHeight,14);          % ECD at bit [ppg]
gp = zeros(height(hMD),1);              % Initilalize PP gradient
gp0 = 9;                                % Point of normalization of PP[ppg]
w = InterData(SetHeight,8);             % Weight on Bit [klbf]
w0 = 0;                                 % Bit weight threshold before
                                        %  the bit is effective
r = InterData(SetHeight,10);            % Surface RPM [RPM]
r0 = 100;                               % Point of normalization of RPM
q = InterData(SetHeight,12);            % Mud flow [L/min]
d_B = 12.25;                            % Bit outer diameter [in]
H = 0;                                  % Bit tooth wear
rho = 14.6;                             % Mud Weight
A_T = (15^2 + 15^2 + 15^2 + ...         % Total area flow [in^2]
       15^2 + 16^2 + 16^2 + ...
       16^2)*pi*(1/(4*32^2));
Cd = 0.95;                              % Discharge coefficient
LPMtoGPM = 0.26417;                     % Conversion factor for L/min
                                        %  to gallons/min
PD = (8.311e-5).*...                    % Bit pressure drop [lbs/in^2]
    (rho.*(q{:,:}.*LPMtoGPM).^2)./...
    ((Cd^2)*(A_T^2));
Fj = 0.01823.*Cd.*(q{:,:}.*LPMtoGPM).*... % Jet impact force [lbf]
    (rho.*PD).^.5;
Fj0 = 1000;
%% Converting From Measured Depth to Total Vertical Depth
for i = 1:height(hMD)
   h0TVD{i,1} = MDtoTVD(h0MD{i,1});
   hTVD(i) = MDtoTVD(hMD{i,1});
end

%% Finding approximate PP gradient corresponding to given depth
for i = 1:height(hMD)
    [ ~, PPindex] = min( abs(PPFG.depth(:) - hMD{i,1}));
    gp(i) = PPFG.MLPP(PPindex,1);
end
%% Defining Data Set Table
DataSet = table('Size',[N 10],'VariableTypes',...
    {'datetime','double','double','double','double',...
    'double','double','double','double','double'});
DataSet.Properties.VariableNames = ...
    {'Time','ROP','X1','X2','X3','X4','X5','X6','X7','X8'};

%% Drilling Parameters
DataSet.Time = InterData(SetHeight,1);
DataSet.ROP = InterData(SetHeight,3);
DataSet.X1(:) = 1;                                  % Dummy value
DataSet.X2 = distdim(h0TVD{:,:}-hTVD(:),'m','ft');
DataSet.X3 = (distdim(hTVD(:),'m','ft').^(0.69)).*(gp(:) - gp0);
DataSet.X4 = distdim(hTVD(:),'m','ft').*(gp(:) - ECD{:,:});
DataSet.X5 = log((w{:,:}./d_B - w0(:)./d_B)./(4 - w0(:)./d_B));
DataSet.X6 = log(r{:,:}./r0(:));
DataSet.X7(:) = -H;
DataSet.X8 = log(Fj./Fj0);
end
```

B.10 Extended Kalman Filter

```
if NewMeasurement
    y = [hwell; ROPwell - ROPest];

    % Calculate the Kalman gain matrix
    K = P_prd * Cd' * inv( Cd * P_prd * Cd' + Rd );

    % Correct the state and covariance error estimate
    x_hat = x_prd + K * (y - Cd * x_prd);
    P_hat = ...
    (eye(2) - K*Cd) * P_prd * (eye(2) - K*Cd)' + K * Rd * K';
else
    % If there are no new measurements
    % use only the predicted estimate
    x_hat = x_prd;
    P_hat = P_prd;
end

% Linearized df/dx around x = x_hat
Ad   = [ 1+dt*(-a(2) + ...
    0.69*a(3)*((gp - gp0)/(x_hat(1)^0.31))...
    + a(4)*(gp-ECD))*ROPest   dt
    0  1 ];

% Predict next state estimate and covariance error
x_prd = x_hat + dt*[ROPest + x_hat(2);0];
P_prd = Ad * P_hat * Ad' +  Qd ;
```

B.11 Successive Linearization based MPC

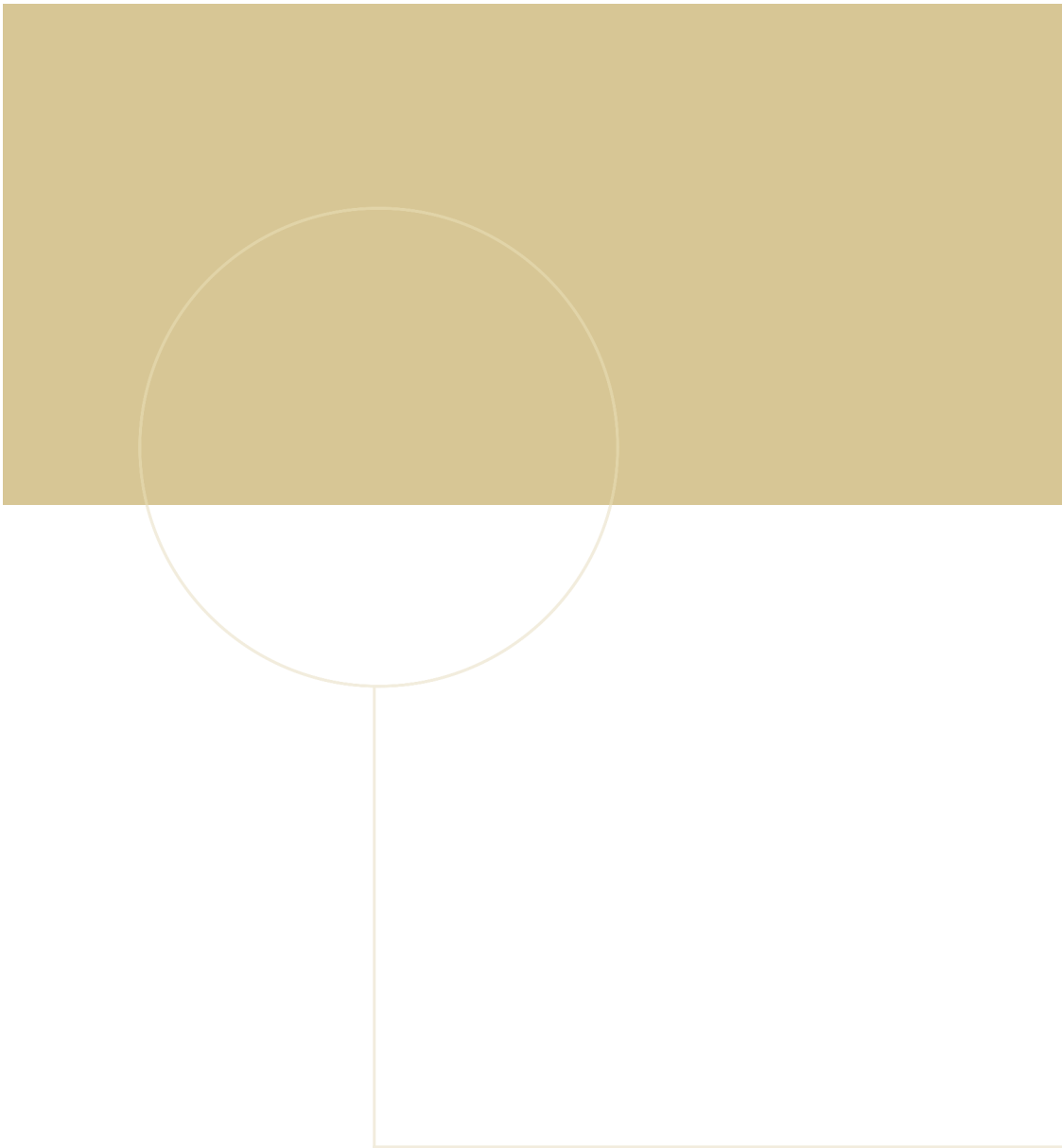
```
%% Model Predictive Controller (SLMPC)
if T > StartSetpointRegulation

    % Linearization %
    At = [1 + dt*ROPest*(a(4)*(gp-ECD) - a(2) ...
        + 0.69*a(3)*((gp - gp0)/(x_hat(1)^0.31))) dt;0 1];

    Bt = [dt*ROPest*(a(5)/(w - w0)) dt*ROPest*(a(6)/r);0 0];

    Ct = [ROPest*(a(4)*(gp-ECD) - a(2) ...
        + 0.69*a(3)*((gp - gp0)/(x_hat(1)^0.31))) 1];
    Dt = [ROPest*(a(5)/(w - w0)) ROPest*(a(6)/r)];

    % QP-formulation %
    f0 = [x_hat(1) + dt*(ROPest + x_hat(2) );x_hat(2)];
    g0 = ROPest + x_hat(2)  ;
    x0 = [x_hat(1);x_hat(2)];
    x_op = x0;                          % Using x0 as operating point
    u_op = [w;r];                       % Using last applied w,r as
                                        % operating point

    F = f0 - At*x_op - Bt*u_op;
    G = g0 - Ct*x_op - Dt*u_op;

    P = FindP(At,Ct,Np);
    H = FindH(At,Bt,Ct,Dt,Np);
    E = FindE(At,Ct,Np);
    Delta = FindDelta(Np,Nu);

    U_1 = repmat(u_op,[Np 1]);          % Repeats the u_op over
                                        % the whole horizon

    % Collecting all constant terms, including the reference ROP
    M = E*F + ones(Np*Ny,1)*G- yref(T-StartSetpointRegulation:...
        T-StartSetpointRegulation+Np*Ny-1);

    % Weighting matrices for the objective function before
    % augmentation
    H_tilde = H'*Q*H + R;
    c_tilde = H'*Q'*(P*x0 + M) - R'*Uref;


    % Building the inequality constraints Az <= b
    A1 =  [inv(Delta);-inv(Delta)];
    I1  = eye(Np*Nu);
    Idu = repmat(eye(Np*Nu),2,1);
    I2  = eye(Np);

    b1 =  [dUub + Delta\U_1;-dUlb - Delta\U_1;];
    b2 = Uub - repmat([WOBmargin;RPMmargin],Np,1);
    b3 = [Delta\U_1;-Delta\U_1];
    b4 = ones(Np*Ny,1)*MaxROP - P*x0 - E*F - ones(Np*Ny,1)*G;

    Aineq = ...
    [A1 zeros(height(A1),2*width(I1) + width(I2));
    I1 -I1 zeros(height(I1),width(I1) + width(I2));
    A1 zeros(height(A1),width(I1)) -Idu zeros(height(A1),width(I2));
    H zeros(height(H),2*width(I1)) -I2];

    bineq = [b1;b2;b3;b4];

    % Upper and lower bounds on U and slack variables
    AugUlb = [Ulb; zeros(Np*Nu,1); zeros(Np*Nu,1); zeros(Np,1)];
    AugUub = ...
        [Uub; ones(Np*Nu,1)*inf; ones(Np*Nu,1)*inf;ones(Np,1)*inf];
```

```
% Building the quadratic weighting matrix
Gamma = ...
[H_tilde zeros(height(H_tilde),width(Su)+width(Sdu)+width(Sy));
 zeros(height(Su),width(H_tilde)) Su ...
 zeros(height(Su),width(Sdu)+width(Sy));
 zeros(height(Sdu),width(H_tilde)+width(Su)) Sdu...
 zeros(height(Sdu),width(Sy));
 zeros(height(Sy),width(H_tilde)+width(Su)+width(Sdu)) Sy];

% Building the linear weighting matrix
Zeta  = [c_tilde; rho_u; rho_du; rho_y];

% Solving the open loop optimization problem
tic;
[U,lambda,exitflag,output] = quadprog(Gamma,Zeta...
    ,Aineq,bineq,[],[],AugUlb,AugUub);
stop = stop + toc;
TimerTicker = TimerTicker + 1;

% Applying only the first control input to the system
w =  U(1);
r =  U(2);
```

B.12 Initializing the MPC and Drilling Coefficients

```
%% Initialize MPC
%%
% Length of Simulation
n = 4000;

% Prediction Horizon and Number of States, Inputs and Outputs
Np = 10;
Nx = 2;
Nu = 2;
Ny = 1;

%% Upper and Lower Bounds
MaxROP = 30;           % Max ROP constraint can be delivered by
                       % by Halliburton "Max ROP" software.
WOBmargin = 5;         % WOB margin to avoid excessive wear and tear on bit
HardWOB_UB = 25;       % Max WOB tolerance
HardWOB_LB = 2;        % Minimum WOB tolerance
RPMmargin = 15;        % RPM margin to avoid excessive wear and tear on bit
HardRPM_UB = 180;      % Max RPM tolerance
HardRPM_LB = 100;      % Minimum RPM tolerance

Ulb = zeros(Np*Nu,1);
Uub = zeros(Np*Nu,1);
Ulb(1:2:end) = HardWOB_LB;     % Lower bound on WOB
Ulb(2:2:end) = HardRPM_LB;     % Lower bound on RPM
Uub(1:2:end) = HardWOB_UB;     % Upper bound on WOB
Uub(2:2:end) = HardRPM_UB;     % Upper bound on RPM
dUlb = zeros(Np*Nu,1);
dUub = zeros(Np*Nu,1);
dUlb(1:2:end) = -.5;          % Lower bound on WOB rate of change
dUlb(2:2:end) = -.5;          % Lower bound on RPM rate of change
dUub(1:2:end) = .5;           % Upper bound on WOB rate of change
dUub(2:2:end) = .5;           % Upper bound on RPM rate of change

%% Input Reference Values
Uref = zeros(Np*Nu,1);
Uref(2:2:end) = 142.22;     % High RPM desirable for proper hole cleaning

%% Weighting Matrices
Q = eye(Np*Ny)*1e4;                          % Weight on ROP
r1 = 10;                                     % Weight on WOB
r2 = 10;                                     % Weight on RPM
R1 = [r1 0;0 r2];
R = diag_repeat(R1,Np);

% Weight matrices for slack on inputs U

s_u1 = 1;                                    % Weight on WOB slack
s_u2 = 1e6;                                  % Weight on RPM slack
Su = diag_repeat(diag([s_u1 s_u2]),Np);      % Quadratic weight matrix
rho_u = repmat([s_u1;s_u2],Np,1)*0;          % Linear weight matrix

% Weight matrices for slack on rate of change of inputs dU

s_du1 = 1e6;                                 % WOB rate of change slack
s_du2 = 1e6;                                 % RPM rate of change slack
Sdu = diag_repeat(diag([s_du1 s_du2]),Np); % Quadratic weight matrix
rho_du =  repmat([s_du1;s_du2],Np,1)*0;      % Linear weight matrix

% Weight matrices for slack on Max ROP.
Sy = eye(Np)*1e6;                            % Quadratic weight matrix
rho_y = ones(Np,1)*0;                        % Linear weight matrix

%% Drilling Plan
yref = ones(n,1)*20;
yref(200:1500) =35;
yref(1501:2000) =45;
```

```
%% InitDrillingCoefficients()
% Initializing the Drilling Coefficients for both "Real well" and the Model

% Initialization for "Real well" coefficients
% based on 200 datapoints from actual drilled well
DataSet = CreateDataSet(200,t,InterData);
RegSol = FindDrillingCoefficients(DataSet,EstimationMethod);
awell(1) = RegSol(1);
awell(2) = RegSol(2);
awell(3) = RegSol(3);
awell(4) = RegSol(4);
awell(5) = RegSol(5);
awell(6) = RegSol(6);
awell(7) = RegSol(7);
awell(8) = RegSol(8);

% Initialization of Model coefficients by guess.
a = [1.5,0.000053,0.0009,0.000008,0.6,0.5,0.9,0.59];
```



NTNU

Kunnskap for en bedre verden