NTNU
Norwegian University of Science and Technology

Stick-Slip Prevention of Drill Strings Using Nonlinear Model Reduction and Nonlinear Model Predictive Control

Morten Krøtøy Johannessen
Torgeir Myrvold

Master of Science in Engineering Cybernetics
Submission date: May 2010
Supervisor: Jan Tommy Gravdahl, ITK
Co-supervisor: Morten Hovd, ITK

Norwegian University of Science and Technology
Department of Engineering Cybernetics

Problem Description

This assignment is given by the Department of Engineering Cybernetics in cooperation with National Oilwell Varco (NOV).

Stick-slip oscillations are severe, self-sustained and periodic torque fluctuations of the drill string torque. They are driven by nonlinear downhole friction and characterized by large bit speed variations. Stick-slip oscillations are recognized as being a major source of problems, such as fatigue failures, excessive bit wear and poor drilling rate. This problem can be addressed by implementing a nonlinear model predictive controller (NMPC). NMPC is a model-based controller which uses an internal model of the process to be controlled to find an optimal input vector. The NMPC s main strength is that it handles constraints both on inputs, states and outputs.

The topic regarding stick-slip and NOV s stick-slip prevention system, SoftSpeed, has been treated by the project group in TTK4530 - Control Engineering, Specialization Project in the fall of 2009. That work was mainly a literature study and the intention was to get familiar with SoftSpeed and to find its weaknesses and strengths and compare it with other available stick-slip prevention systems. This master thesis can be regarded as a continuation of the project.

The main tasks for this master thesis are:
- Verification and validation of the drill string model using operational data from NOV
- Nonlinear model reduction of the drill string model for the purpose of using it in an NMPC
- Develop an NMPC for the purpose of curing stick-slip in drill strings
- Implement the NMPC in MATLAB and illustrate the performance using simulation
- Compare the NMPC with the stick-slip prevention system, SoftSpeed by NOV

Assignment given: 11. January 2010
Supervisor: Jan Tommy Gravdahl, ITK

Abstract

The main focus of this thesis is aspects in the development of a system for prevention of stick-slip oscillations in drill strings that are used for drilling oil wells. Stick-slip is mainly caused by elasticity of the drill string and changing frictional forces at the bit; static frictional forces are higher than the kinetic frictional forces which make the bit act in a manner where it sticks and then slips, called stick-slip. Stick-slip leads to excessive bit wear, premature tool failures and a poor rate of penetration. A model predictive controller (MPC) should be a suitable remedy for this problem; MPC has gained great success in constrained control problems where tight control is needed.

Friction is a highly nonlinear phenomenon and for that reason is it obvious that a nonlinear model is preferred to be used in the MPC to get prime control. Obviously it is of great importance that the internal model used in the MPC is of a certain quality, and as National Oilwell Varco (NOV) has developed a nonlinear drill string model in Simulink, it will be useful to check over this model. This model was therefore verified with a code-to-code comparison and validated using logging data provided from NOV.

As the model describing the dynamics of the drill string is somewhat large, a nonlinear model reduction is needed due to the computational complexity of solving a nonlinear model predictive control problem. This nonlinear model reduction is based on the technique of balancing the empirical Gramians, a method that has proven to be successful for a variety of systems.

A nonlinear drill string model has been reduced and implemented to a nonlinear model predictive controller (NMPC) and simulated for different scenarios; all proven that NMPC is able to cope with the stick-slip problem. Comparisons have been made with a linear MPC and an existing stick-slip prevention system, SoftSpeed, developed by National Oilwell Varco.

Preface

The work presented in this thesis was carried out at the Department for Engineering Cybernetics at the Norwegian University of Science and Technology (NTNU) during the spring of 2010. The work has been carried out in collaboration between the students Torgeir Myrvold and Morten Johannessen. The assignment is given by Department for Engineering Cybernetics in cooperation with National Oilwell Varco represented by Pål Jacob Nessjøen.

Several people have been involved in this assignment, and without their support and assistance would not the outcome of this thesis be of such high quality. The following people deserve acknowledgement for there help:

We would like to thank our supervisors at NTNU Tommy Gravdahl and Morten Hovd for useful comments and helpful pointers.

Co-supervisor Pål Jacob Nessjøen has provided us with logging data and insightful comments regarding the stick-slip problem.

The group is very thankful to associate professor Juergen Hahn at Texas A&M University for his assistance. His help gave us insight about nonlinear model reduction based on balancing of empirical Gramians, and he also provided us with code examples which made the theory easier to understand and the code was used as a basis when the theory was implemented.

Professor Lars Imsland at the Department for Engineering Cybernetics, NTNU provided us with a code example for a nonlinear model predictive controller to use as basis when implementing the NMPC. He also gave helpful comments regarding model predictive control of oscillating systems.

We would finally like to thank our fellow students and especially Håvard Pehrson which we have shared office with. He has offered great support and help with the model predictive controller and not at least been a competent opponent in our lunchtime card games.

Torgeir Myrvold and Morten Johannessen
Trondheim, May 2010

"Don't accept the old order. Get rid of it."
- Johnny Rotten

Contents

1 INTRODUCTION 1

1.1 STICK-SLIP 2
1.2 STICK-SLIP IN DRILLING OPERATIONS 5
1.3 DRILLING ASSEMBLY 6
1.4 PROBLEMS RELATED TO STICK-SLIP 7
1.5 BACKGROUND 8
1.6 STICK-SLIP PREVENTION 8
1.7 MOTIVATION 11
1.8 CONTRIBUTIONS 12
1.9 SOURCE CODE 12

2 VERIFICATION AND VALIDATION 13

2.1 THE USE OF V&V 15
2.2 V&V OF DRILL STRING MODEL 16

3 MODEL REDUCTION 23

3.1 BALANCED TRUNCATION FOR LINEAR SYSTEMS 25
3.2 BALANCING OF EMPIRICAL GRAMIANS FOR NONLINEAR SYSTEMS 37
3.3 FULL SCALE MODEL REDUCTION 48

4 MPC 53

4.1 RECEDING HORIZON CONCEPT 54
4.2 OBJECTIVE FUNCTION 55
4.3 CONSTRAINTS 55
4.4 PREDICTION 57
4.5 COMPLETE FORMULATION 59
4.6 CONTROL HIERARCHY 59
4.7 ESTIMATION 61
4.8 STABILITY 64
4.9 TUNING 65
4.10 NMPC 66

5 NMPC FOR DRILL STRING 69

6 SIMULATIONS OF THE NMPC AND COMPARISONS 71

6.1 NMPC 71
6.2 LINEAR MPC 78
6.3 SOFTSPEED 83
6.4 COMPARISON 85

7 DISCUSSIONS 87

7.1 VERIFICATION AND VALIDATION 87
7.2 MODEL REDUCTION 87
7.3 MPC 88

8 CONCLUSION 89

8.1 CONCLUDING REMARKS 89
8.2 CONTRIBUTIONS PROVIDED BY THIS THESIS 90
8.3 SUGGESTIONS FOR FURTHER WORK 90

BIBLIOGRAPHY 93

A PROOF OF ASYMPTOTIC STABILITY OF STRICTLY PASSIVE SYSTEM 97

B MATLAB/SIMULINK FILES 99

Abbreviations

AVD……………………...	Active Vibration Damper
BHA……………………...	Bottom Hole Assembly
D-OSKIL………………...	Drilling Oscillation Killer
EKF……………………....	Extended Kalman Filter
HIL…………………….....	Hardware-In-the-Loop
LQR……………………....	Linear Quadratic Regulator
LP…………………….......	Linear Programming
MRF……………………...	Magnetorheological Fluid
MPC……………………...	Model Predictive Control
MWD……………………..	Measurement While Drilling
NMPC…………………....	Nonlinear Model Predictive Control
PDC……………………....	Polycrystalline Diamond Compact
PDE……………………....	Partial Differential Equation
PID…………………….....	Proportional-Integral-Derivative
QP…………………….…...	Quadratic Programming
ROP………………………	Rate Of Penetration
RPM……………………...	Revolutions Per Minute
SISIO……………………..	Single Input – Single Output
STRS……………………..	Soft Torque Rotary System
SQA………………………	Software Quality Assurance
SQP………………………	Sequential Quadratic Programming
SVD………………………	Singular Value Decomposition
UKF………………………	Unscented Kalman Filter
V&V……………………...	Verification and Validation
WOB……………………..	Weight On Bit

1 Introduction

Most of this chapter is taken from the report in the subject *TTK4530 - Control Engineering, Specialization Project* by Johannessen & Myrvold (2009).

There has for the past 50 years been conducted extensive research on the subject of torsional vibrations on drill strings used by the oil industry. These torsional vibrations are an important cause for deteriorated drill string performance. They can lead to premature failure of bits, motors and other expensive components used in drilling operations.

One of the main reasons for torsional vibrations is the stick-slip phenomenon. The phenomenon is characterized by stick-phases, where the rotation comes to a complete stop, and slip-phases where the angular velocity of the bit increase up to three times its nominal value. This undesirable motion of the bit will not only lead to unwanted wear, but also significantly reduce the rate of penetration (ROP), which is an important consideration financially associated with drilling operations.

Figure 1.1: Offshore oil rig

1.1 Stick-Slip

Stick-slip is a phenomenon that can appear when two surfaces are sliding in contact with each other, and the surfaces are alternating between sticking and sliding. This will result in a change in the force of friction since the static friction is usually larger than the kinetic friction. As an example it can be considered an object lying on a flat surface. If a force is applied to the object that is large enough to overcome the static friction, the reduction to the kinetic friction can cause a sudden jump in the velocity from standstill. If there are some elasticity between the source of the force and the point where the friction acts, the system can at a specific frequency alternate between sticking and slipping.

The stick-slip phenomenon is present in many different settings in our everyday life; some of them will be presented here.

Violin

When a violin player makes beautiful music it is stick-slip vibration that make the violin sing. The friction between the bow and the string causes stick-slip to occur. With the higher static friction the string "sticks" to the bow and the bow drags the string with it until the energy in the string is large enough and the string "slips"; it brakes free from the bow and slides past it with low kinetic friction. When the string has lost its energy it sticks to the bow again and the cycle continues. When the violinist changes the tone of the instrument, he moves his finger back and forth along the violin neck while pressing the string against the neck, which will change the effective length of the string. This manoeuvre forces the stick-slip to occur at different frequencies, and by performing these string-length-variations the right way, the violin will make beautiful music. Also the speed of the bow will have effect on the stick-slip frequency, so in a simplified manner it can be seen as the violinist has two degrees of freedom to manipulate the sound from the violin.

Figure 1.2: A famous Norwegian violin player utilizing the stick-slip phenomenon

Panulirus argus

The phenomenon of stick-slip is also present in the nature, and the spiny lobster (Panulirus argus) can be such an example (Patek, 2001). The lobster takes advantage of the stick-slip phenomenon to produce a loud, abrasive sound to scare off predators. By rubbing a plectrum (a basal extension on the antenna of soft elastic tissue) over microscopic shingles on a file (located below its eyes), the lobster can produce high frequency sounds using the stick-slip mechanism. The energy is stored in the soft, elastic tissue of the plectrum during the stick-phase, and then being released during the slip-phase; a sound pulse is then generated whenever the two surfaces slip. The spiny lobster can effectively scare off predators by using the stick-slip mechanism during the moult cycle when their exoskeleton is soft.

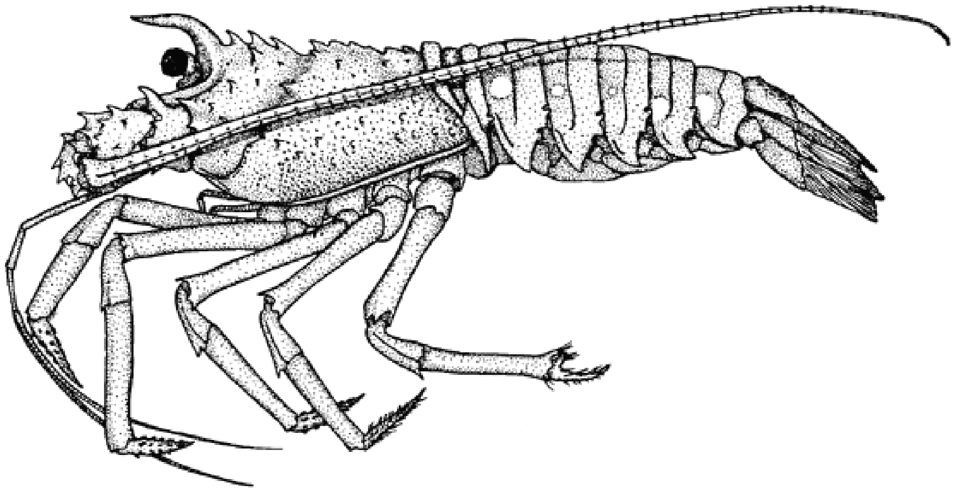

Figure 1.3: The spiny lobster

Tectonic earthquakes

Another situation in nature where the stick-slip vibrations can be remarkable is when earthquakes occur. The hard outer shell of the Earth is currently broken into eight major, and many minor plates. A tectonic earthquake is caused by a sudden release of energy in the Earth's crust that creates seismic waves. The energy is released as a result of a sudden slippage along the boundary between two plates. As the plates have asperities along their boundaries, they will tend to stick or lock up to each other as they move. The continued relative motion between the plates will lead to increasing stress and a force is built up as the plates deform elastically. When the force is sufficient to make the plates break free, the stored energy is released as seismic waves that cause the ground to vibrate. The plates will eventually stick and lock up again in a stick-slip behaviour and yet more earthquakes will happen as time goes by.

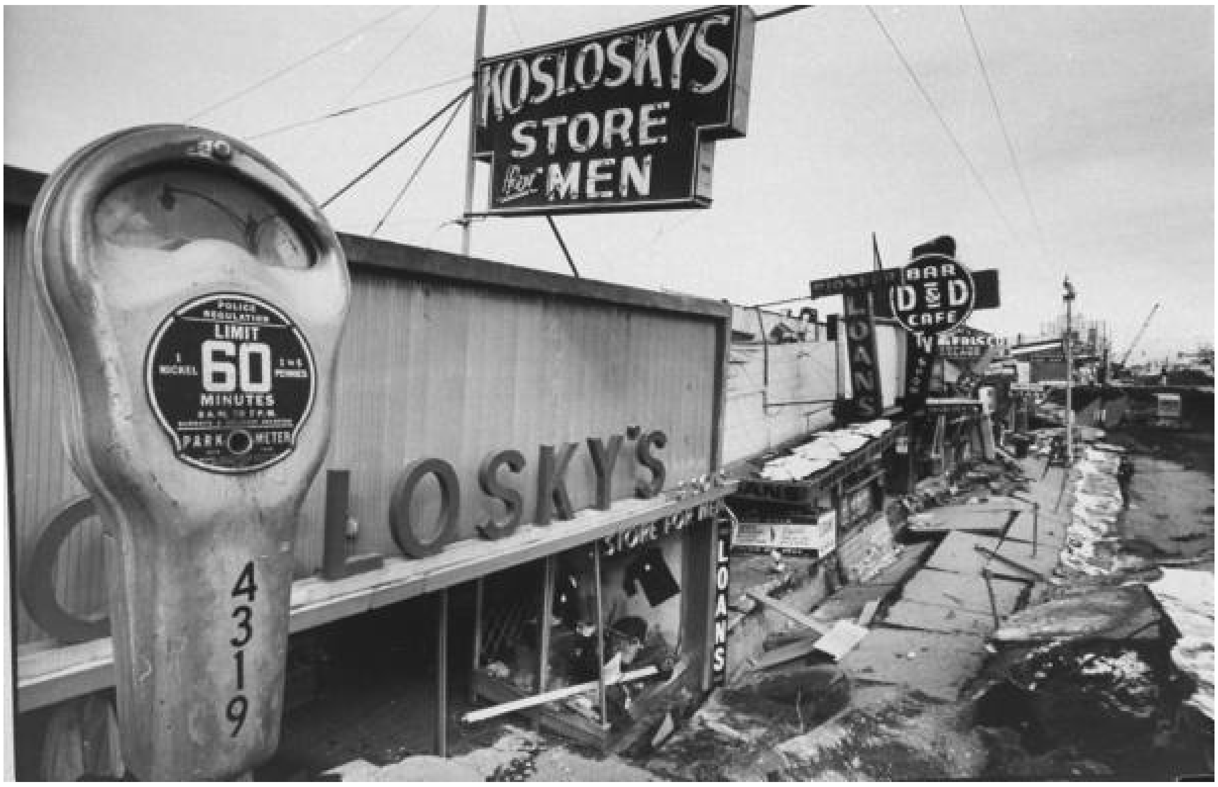

Figure 1.4: Stick-slip can cause great damage when generating earthquakes

1.2 Stick-slip in drilling operations

The stick-slip phenomenon is also a challenge when drilling for oil and gas reservoirs. For the exploration of oil and gas, wells are drilled, which connects the reservoir to the earth's surface. In this case the main problem appears in the friction force between the drilling bit and the rock it is drilling through; there will be a large friction at the diamond cutters on the bit that shear the rock. As stated earlier, stick-slip could appear if there are some elasticity between the source of the force and the point where the friction acts. In this situation it will be the top drive at the drilling rig that applies torque to the string, and large friction forces emerge at the bit. As the string length increases, up to several thousand meters, the elasticity of the string will eventually result in stick-slip occurrence.

Not only the bit, but also the string itself can induce stick-slip. The string will to some extent be in contact with the well, even though the well diameter is larger than the diameter of the string. There are mainly two reasons why this contact occur; with long drill strings the string will to some degree obtain a buckled shape down the well (e.g. caused by centrifugal forces and unbalance in the string), and in the latter years it has also become more and more common to drill sloped wells. If the contact force between string and well get large enough, stick-slip can occur anywhere along the string.

To fully understand the stick-slip phenomenon when drilling oil and gas wells, a description of how to consider the drilling arrangement is addressed in the next section.

1.3 Drilling assembly

In a simple consideration of the drilling assembly it consists of a top drive, rotary table, drill string, drill collar and drill bit. A sketch of this can be seen in Figure 1.5.

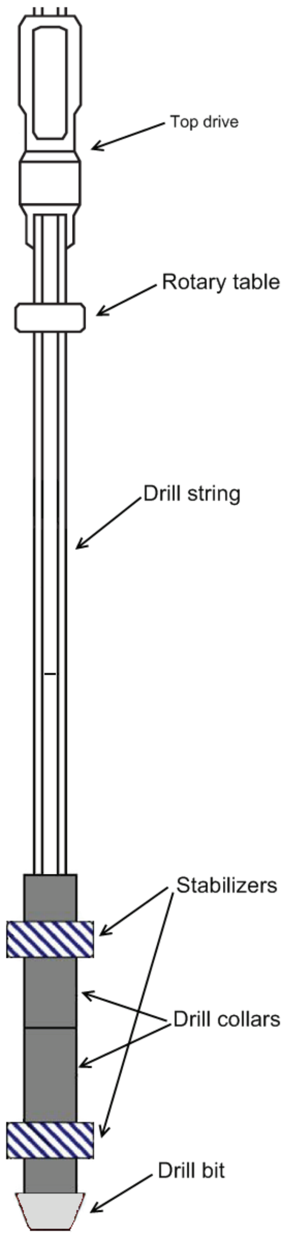

Figure 1.5: Drilling assembly

The top drive is usually a large electric motor that can revolve at an approximately steady rotation speed, regardless of what is happening down the well. The drill string is a relatively slender pipe that connects the top drive to the drill collar. Typically the pipe can have an outer diameter of 127 mm (5”), a wall thickness of 9 mm and can be many km long. In fact the string consists of many pipes that are linked together with treaded connections, where each pipe usually is about 9 m long. To get a picture of how long and slender the drill string can be, it could be considered a drill string that has a diameter of 1 mm. If a 5” bore string is 5000 m, the 1 mm string would be about 40 m long, assuming the same proportions.

The lower part of the drill string, the collar and the bit, are often called the Bottom Hole Assembly (BHA). This part could typically have an outer diameter of 5” – 10” and an inner diameter of 2.5” – 3”. This part is often considered as torsional rigid in simulations. The bit at the end of the BHA is a rock cutting tool and it is here the borehole is created. Modern bits that are used today, so-called PDC bits (Polycrystalline Diamond Compact), consists of a steel body with artificial diamond cutters that shear the rock.

The drill string will also act as a transport channel for mud that is being pumped from the drilling rig down to the drill bit and then pumped through nozzles in it. The mud’s purposes is to lubricate the bit and to remove the cutting material from the borehole; when the mud flows back up again it will bring the rock particles to the surface. The mud is pumped through the string with high pressure and sometimes the mud flow is also used to run a mud motor placed near the bit which will add extra torque and speed to the bit.

At the upper end the drill string is connected to a hook which pulls the string upwards. Because of this hook, most of the drill string is in a constant tension, except for the BHA which will be partly compressed. The tension of the drill string is used to prevent the weight on bit (WOB) becoming too large, and reduces the string’s chance of buckling down the well. Since the BHA is compressed, stabilizers are used to reduce buckling, see Figure 1.5. The stabilizers are short sections which nearly has the same diameter as the drill bit.

Then back to the stick-slip problem associated with drilling operations. The problem has been known for several decades and a large number of papers and articles have addressed the stick-slip problem, like for example Kyllingstad & Halsey (1988), Dufeyte & Henneuse (1991), Brett (1992), Sananikone, Kamoshima & White (1992), Robnett, Hood, Heisig & Macpherson (1999), Baumgart (2000), Challamel, Sellami & Gossuin (2000), Leine, van Campen & Keultjes (2002), Khulief, Al-Sulaiman & Bashmal (2007).

With drill strings that exceed a length of about 2000 m the stick-slip problem starts to become significant. In such cases, when starting from standstill, the top drive will have to rotate many times before enough torque is applied to the bit so it can overcome the static friction between the bit and rock. When the bit slips and starts to rotate, it will get a high acceleration. Often the angular velocity of the bit can reach up to three times the velocity of the top drive before the bit speed reduces again and reaches the stick-situation. Then the top drive will ‘spin up’ the string again until the bit slips, and so it goes on. As mentioned above the string and BHA can be considered as a torsion spring with a dynamic viscous friction component in addition to the static friction.

1.4 Problems related to stick-slip

The existence of stick-slip when performing oil well drilling can cause problems, such as excessive bit wear, premature tool failures and poor rate of penetration. The problems are closely related to the high peak speeds occurring during the slip-phase. The high rotation speeds in turn lead to extreme accelerations and forces, both in axial and lateral directions. In some extreme slip-phases the separated pipes that the drill string consists of can be unscrewed, or the string can simply break. If this happens the well can be lost, and the work will then be set back for several weeks.

1.5 Background

Stick-slip oscillations when drilling oil wells has been a known problem for the last decades and there has been conducted a serious amount of research on the matter. There have been different approaches to solve the problem with stick-slip oscillations; indeed many of the methods have different drawbacks.

Oil history

Oil has been used by the mankind for thousands of years and the first oil wells were drilled in China in 347 BC using bits attached to bamboo poles. The extracted oil was burned to evaporate brine and produce salt. After that, no considerable development in the oil industry was done for the next 2000 years, until the first modern commercial oil well was drilled on the Aspheron Peninsula northeast of Baku, Asia in 1848. Then the rest of the world followed and soon there was drilled for oil onshore throughout the world.

The first offshore oil well was drilled around 1897 in Summerland, California. The idea of drilling offshore came when the early oilmen noticed that the wells that were close to the ocean were the most productive ones. Instead of just drilling wells on the beach, H. L. Williams came up with the idea of building a wharf and erect a drill rig on it. The well was drilled about 90 m into the Pacific Ocean and proved to be a good oil producer.

It was not before the end of World War 2 that the first real offshore oil wells were drilled out of sight from land. In 1947 the Kerr-McGee Corporation drilled the first well from a fixed platform almost 18 km off the Louisiana shore. This led to a boom in the offshore oil drilling industry and soon many offshore oil fields were in production.

Since the first oil well was drilled in China there has been a huge development in both the production and the usage of oil and gas. As the years has went by the technology has evolved, and the demand for higher effectiveness in the production has got more important.

1.6 Stick-slip prevention

Because of the problems related to stick-slip, many methods and systems have been proposed through the years to eliminate, or anyhow reduce the problem. Drilling operators have made their own methods by for instance reducing the weight on bit, or changing the speed of the top drive. These uncontrolled operator activities will cure the stick-slip problem, but often it will result in a poor drilling rate. More intelligent systems have therefore been made to prevent stick-slip, still maintaining a good drilling rate. Some of these systems are presented in the following.

Torque feedback

One of the first methods that were presented used torque feedback from the applied string torque (Halsey, Kyllingstad & Kylling, 1988). With a torque feedback, the stick-slip oscillations can be prevented by allowing the rotary table speed to respond to dynamic torque variations. Since the torque is used as feedback in the control loop, a compromise between two contradictory requirements is bound to happen; the control loop needs to maintain the speed set point while also maintaining a constant torque. To prevent stick-slip oscillations the speed is adjusted so the torsional waves are dampened at the rotary table instead of being reflected back to the drill string. A drawback with this method is that it needs a good measurement of the string torque, which in practice can be very hard to get.

Soft Torque Rotary System

Shell Research was aware of the problems with the measurement of the drill string torque and focused their research on improving the torque feedback in the 1990s. They saw two major drawbacks with the torque feedback; the measurement of drill string torque, and the control algorithm which was based on zero reflection of the torsional waves. By using the motor current they computed the drill string torque, and used this in the feedback loop. By tuning the controller (which could be considered as an active damper), the system dampens stick-slip oscillations in an effective manner. This will make the system behave as a tuned damper, similar to the passive dampers which often are used to avoid wind-induced oscillations in power-transmission lines. This system was called Soft Torque Rotary System (STRS) and field testing has shown that the system dampens stick-slip oscillations effectively, and that ROP is increased and downhole equipment failures are reduced. The system has been commercially available for many years and is used on different locations around the world.

PID-control

A simple method for curing stick-slip was presented by Pavone and Desplans (1994). By doing thorough analyzes of data obtained by TRAFOR, a Measurement While Drilling (MWD) system operated by the Institut Français du Pétrole, they found that by using stability analysis they could avoid stick-slip. From analyzes they derived parameters for a PID-controller to control the rotary table speed that cured stick-slip, but the damping was very low so the drilling had to stand high vibrations. Exactly how they derived the PID-parameters is not explained. It seems that their main focus was on using the data they got from TRAFOR and from this derive a good model of the drill string that describe stick-slip instead of finding an effective way to suppress stick-slip oscillations.

H_∞-control

In 1998 Serrarens, van de Molengraft and van den Steen proposed a H_∞-control method to suppress stick-slip oscillations on a contemplated system. H_∞-control has been a widely used solution in controlling vibration problems, such as in cutting processes where it is used to suppress machine tool chatter. The H_∞-controller suppressed the stick-slip oscillations in spite of the controller being linear and time-invariant; the friction at the bit is indeed strongly nonlinear. There have only been performed experiments with the H_∞-controller on a semi-real process, and as far as the authors of this thesis know the controller has not been used on a real process. But the results from the experiments gave a promising perspective that the solution could be developed further and be implemented on a real process. The H_∞-controller also has robust qualities which are desirable.

D-OSKIL

Another method to suppress stick-slip oscillations is to use the weight on bit as an additional control variable. This method, called Drilling OScillation KILler (D-OSKIL), was introduced by Canudas-de-Wit et al. (2005). A sustainable reason for the stick-slip oscillations are the friction torque produced by the contact between the rock cutting tool and the rock. It is needed to keep the value of the WOB large to get a good ROP, but as the WOB increases it will enhance the possibility of stick-slip to occur. From this it is obvious that introducing a control strategy that manipulates the WOB is a smart way to go to find an optimal compromise between WOB and ROP. When stick-slip oscillations occur, the WOB is decreased by manipulating the force from the hook lifting the drill string at the rig, e.g. the drill string is pulled upwards to reduce the WOB. Simulations have shown that this is an effective method for suppressing stick-slip oscillations. Another interesting fact with the control law that D-OSKIL uses is that the resulting feedback system has been proven to be globally asymptotically stable. A formal stability analysis is provided by Corchero, Canudas-de-Wit and Rubio (2006), something that is not often seen in the rest of the methods proposed.

Active vibration damper
An active vibration damper (AVD) was presented by Cobern and Wassell in 2005. The aim of the method is to reduce drill string vibration, which will lead to a reduction in stick-slip oscillations. The AVD is mounted between the drill string and the BHA, having a structure that is similar to a shock sub consisting of a spring-fluid damper. Instead of using hydraulic oil in the damper, a magnetorheological fluid (MRF) is used which allows the viscous properties of the fluid to be changed. By varying the viscosity of the fluid, the damping coefficient of the AVD can be manipulated. Because of the fluid's properties, its viscosity can be changed when it is influenced by electromagnetic fields. Measurements of the relative motion of the bit and drill string are taken in real-time during the drilling operation and from this the damping properties are continuously modified. Results from drilling tests have been very promising, and has shown that by varying the damping coefficient, vibration and variation of WOB can be reduced. It is not only in drilling the magnetorheological fluid abilities are utilized, it is also used in earthquake protection systems for buildings and bridges, and sophisticated adaptive automotive suspensions, like the one found in the Ferrari 599 GTB Fiorano.

Figure 1.6: Ferrari 599 GTB Fiorano

Modelling error compensation
In 2008 Puebla and Alvarez-Ramirez introduced a control approach based on modelling error compensation to suppress stick-slip. Since uncertainty always exists when modelling friction components, the suggested feedback control scheme should deal with uncertainties in the friction model and drill string parameters. The drill string is modelled as a simple torsional pendulum driven by an electric motor. The model comprises damped inertias which are mechanically coupled by an elastic shaft that can be viewed as a spring/damper. By introducing modelling error functions, two different control schemes are given, cascade control and decentralized control. The cascade control scheme uses a single control input, the electrical properties of the motor, and a full state feedback. The states that are used in the

feedback are the relative position between coupled inertias and the angular velocity of the inertias. The decentralized control scheme on the other hand uses two inputs, the electrical properties of the motor and WOB. Rotary table oscillations are regulated with the motor, whereas BHA oscillations are regulated with the WOB. The same full state feedback is also used in the decentralized scheme. Numerical simulations have shown that the method gives satisfying suppression of stick-slip oscillations with uncertainties in the control design and changes in model parameters, but a commercially available system based on this method is not available as far as the authors of this thesis know.

SoftSpeed
National Oilwell Varco has developed their own stick-slip prevention system called SoftSpeed, which they presented in 2009 (Kyllingstad & Nessjøen, 2009). The system is basically a PI-controller with an acceleration feedback controlling the speed of the drive, which is tuned so that it dampens torsional oscillations in the drill string, and by this cures stick-slip. More theory and advantages of the use of such acceleration feedbacks can be studied in the dissertation by Lindegaard (2003). The more advanced part of this system is the online tuning of the PI-controller. When SoftSpeed is in use on a real system, there are several algorithms that execute to keep the controller tuned. One algorithm calculates the stick-slip period and another algorithm uses this as a basis for tuning the controller. More on how these algorithms work, and the controller setup, can be found in the report by Johannessen & Myrvold (2009).

1.7 Motivation

Stick-slip oscillations in drill strings are a well studied topic, and a number of methods have been suggested through the years to eliminate the problem. As the technology in the oil industry has developed, the complexity in drilling oil wells has expanded. New technology has made it possible to drill deeper and deeper wells which demand that the stick-slip prevention system has to provide high performance. The authors of this thesis have studied several of the existing methods, but none of these utilizes optimal control. It was therefore of great interest to study this control strategy more closely using the well known Model Predictive Controller (MPC). Since the friction forces acting on the bit are of a highly nonlinear manner, a nonlinear MPC (NMPC) was preferable to use.

When NMPC should be used for this purpose, which requires a model of the process that should be possible to run in real-time, it also turns up some other very interesting areas that have to be investigated.

The drill string model provided by NOV is made from physical considerations and had not earlier been verified and validated (V&V). But it will in fact be very important to know if the model is correct before it can be used in an NMPC.

A good model describing the drill string will very quick become large and demand large computational power. To overcome this problem the model will have to be reduced to some extent by applying a technique for nonlinear model reduction. Model reduction techniques show promising results for implementation in MPC; the computational load is reduced significantly.

Besides safety is efficiency, the ROP, one of the most important factors when it comes to the drilling of oil wells. An MPC can be the answer to maintain a high efficiency, and therefore it is of great interest to investigate the possibility of using MPC in a stick-slip prevention system.

1.8 Contributions

The contributions of this thesis are divided in four main parts as shown below:

Verification and Validation
In Chapter 2 is a V&V experiment conducted on the drill string model. The verification is done using code-to-code comparison and validation using operational data provided by NOV.

Model reduction
Chapter 3 is dedicated to model reduction of a drill string model. Both the theory for a linear and a nonlinear method are presented with complimentary examples. Finally the nonlinear method is used on a complete drill string setup.

Model predictive control
Chapter 4 provides an introduction to model predictive control. The theory for both the linear and nonlinear case is presented.

Chapter 5 deals with the NMPC problem for a rotating drill string and the MATLAB implementation of the controller.

Chapter 6 contains the various simulations of the NMPC, together with simulations of a linear MPC and NOV's stick-slip prevention system, SoftSpeed.

Discussion and conclusion part
Chapter 7 gives summarizing discussions on the different results obtained in the work.

Chapter 8 concludes the work done in this thesis giving concluding remarks, contributions provided by the work and some suggestions for further work.

1.9 Source code

A ZIP-file is attached electronically which contains the source code that has been used in this thesis. All implementations are done in MATLAB/Simulink and an overview over the different files, with a brief description of what they do, can be found in Appendix B.

2 Verification and validation

One of the tasks of this master thesis has been to undertake a model verification and validation experiment of the Simulink drill string model created by NOV. The use of models to simulate physical events and engineering systems are increasingly being used in problem solving and has a great importance in making critical decisions. To fully rely on the model and its results, some kind of quality assurance is needed; there is no use for a model if the results it produces are wrong. How to cope with this topic is addressed through model verification and validation.

One of the first to address verification of computer simulation models where Naylor & Finger (1967). They proposed a three stage process of verification concerning verification in economics which relates to the verification of computer models of industrial systems. The three stages proposed where rationalism, empiricism and positive economics.

In V&V is verification often defined as "ensuring that the computer program of the computerized model and its implementation is correct" (Schlesinger et al., 1979) while validation is defined as "substantiate that a computerized model within its domain of applicability possesses a satisfactory range of accuracy consistent with the intended application of the model".

V&V serve different purposes as shown below;

Verification is concerned with building the model right;

- Is the model programmed correctly?
- Are the algorithms implemented properly?
- Does model contain errors or bugs?
- Are the input parameters and logical structure of the model correctly represented?

Validation is concerned with building the right model;

- Is the model an accurate representation of the real system?
- Does the model meet its intended requirements in terms of the methods employed and the results obtained?

A graphical representation of V&V, called the Sargent Circle, was developed by the Society for Computer Simulations and can be seen in Figure 2.1. This graphical representation gives a simple view of the activities taking place in V&V; the modelling and simulation activities are connected through dashed lines and the assessment activities are connected through the solid lines.

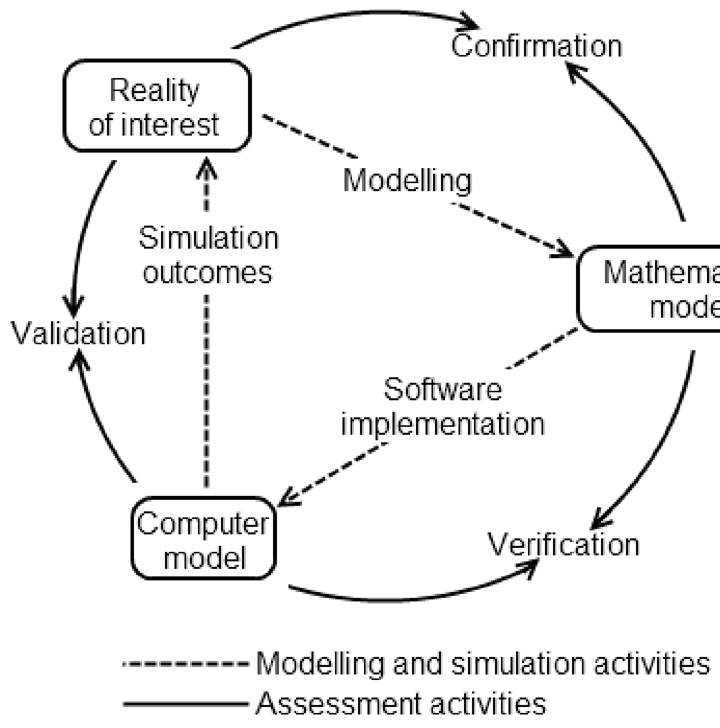

Figure 2.1: A graphical representation of V&V, the Sargent circle

Reality of interest represents the physical system or process, i.e. the particular problem being studied. This could range from a small unit to a complete system.

The *mathematical model* is composed of all mathematical modelling data and mathematical equations that describe the system of interest, typically in the form of partial differential equations (PDEs). There are different ways of producing the mathematical model; it can be produced by modern system identification methods or it can be based on the more classical methods using conservation of mass, momentum and energy.

The *computer model* is an implementation of the mathematical model as an operational computer program. In modern terminology is the computerized model referred to as the computer model or code. It is usually in the form of numerical discretizations, algorithms, convergence criteria and miscellaneous parameters associated with numerical approximations.

A modelling process is necessary to represent the reality of interest as a mathematical model, termed modelling in Figure 2.1. The correctness of the modelling is assessed by the confirmation.

The mathematical model is implemented as a computer model through software implementation. Verification is done to identify and remove errors in the implementation process.

The outcome of the simulation of the computer model is compared with experimental data, the validation. Validation is an ongoing activity as experiments are improved and parameter ranges extended.

2.1 The use of V&V

Since V&V has different purposes are also different methods used; verification is an activity which mostly takes place under the implementation of the model, while validation is a more ongoing activity as there eventually will be more data available to improve the validity of the model.

Verification
Verification is often split up into two parts, code verification and calculation verification. The purpose of the code verification is as the name implies to confirm that the software is working as intended without bugs. To accomplish code verification, software quality assurance (SQA) procedures are used. This is performed by the code developer and ensures that the code is implemented correctly. SQA has a focus on the code as a software product and is only concerned whether it is reliable from the perspective of computer science. Methods for SQA are typically configuration management, static and dynamic testing.

Calculation verification deals with the accuracy of the numerical simulations and should if possible give an estimation of the numerical errors induced by the model. Therefore it is also referred to as numerical error estimation because the primary goal is to estimate the numerical accuracy of a given solution, typically for nonlinear PDEs with discontinuities and singularities. In calculation verification can errors like insufficient convergence tolerance, insufficient spatial or temporal discretization, incorrect input options and finite precision arithmetic be identified and removed.

A popular way of performing calculation verification is to do a code-to-code comparison. This is done by comparing the results obtained from two different codes solving the same problem. It should be mentioned that there is no guarantee that either of them are correct even if they give the same solution, they could incorporate the same error, e.g. if it is an error in the paper they are based on. This method should therefore be used with great care and only when there is insufficient verification evidence from other sources.

Validation
Validation is done to determine how accurate a model represents the real system. This is done by comparing computational results with experimental data. An important issue here is to appropriately design and execute validation experiments that allow for precise and conclusive comparisons between the computational results and the experimental data. This means that the validation experiments have to produce high-quality data for the purpose of assessing the accuracy of a model prediction. In other words, “the code is the customer”. Validation tests are performed on the real system to obtain experimental data that can be used in the validation experiments. It is important that initial conditions, boundary conditions and input parameters are prescribed accurately for the validation test; this will have to be taken into consideration when performing the validation experiments.

There are three aspects that should be taken into consideration when performing a validation experiment:

1. Use the code and define the expected results from the validation experiment
2. Design a specific validation experiment by using the code in a predictive sense
3. Develop a well-thought-out plan for analyzing the computational and experimental results

Figure 2.2 shows the validation process as given in the article by Oberkampf & Trucano (2002). The computational results of the modelling and simulation process are compared with the experimental data; a validation experiment.

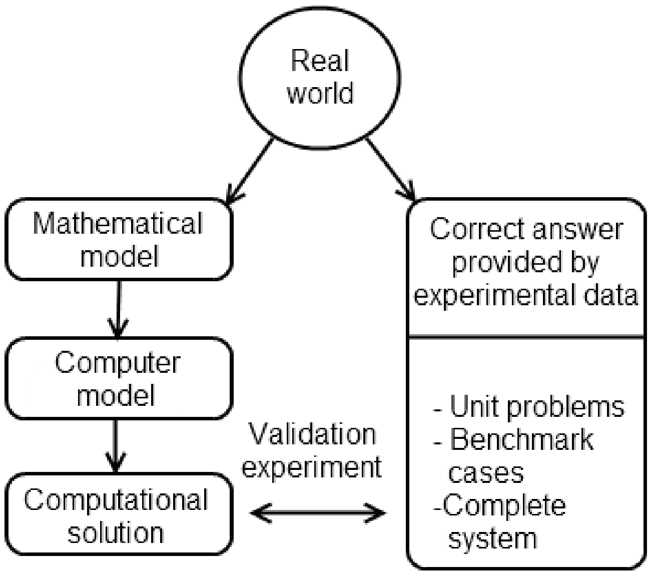

Figure 2.2: Validation process

2.2 V&V of drill string model

In this section the theory behind V&V is used on NOV's drill string model to verify and validate it. It should be noticed that the MATLAB/Simulink files used for V&V purposes are not included in the ZIP-file because the data provided by NOV is under settlement.

Verification:
The model of the drill string consists of mathematical equations which are based on the conservation of momentum. The equations have been implemented into Simulink, which is MathWorks' tool for modelling, simulation and analyzing. Implementation of the equations was done by NOV and verification was also performed; the Simulink model has been used in several simulations and runs without any bugs or problems.

A verification procedure done in this thesis is a code-to-code comparison. Mathematical equations of the drill string were derived using conservation of momentum and implemented in the MATLAB Editor as a script. The results when the models were simulated in Simulink or as a script were practically the same, a good indicator that the implementations of the mathematical equations are correct. There are some deviations because in the script model is not all the decimals of the constants included and a different solver is used. Simulations when an input step from 10 to 15 rad/sec is applied to both the Simulink model and the model implemented in a script, can be viewed in Figure 2.3.

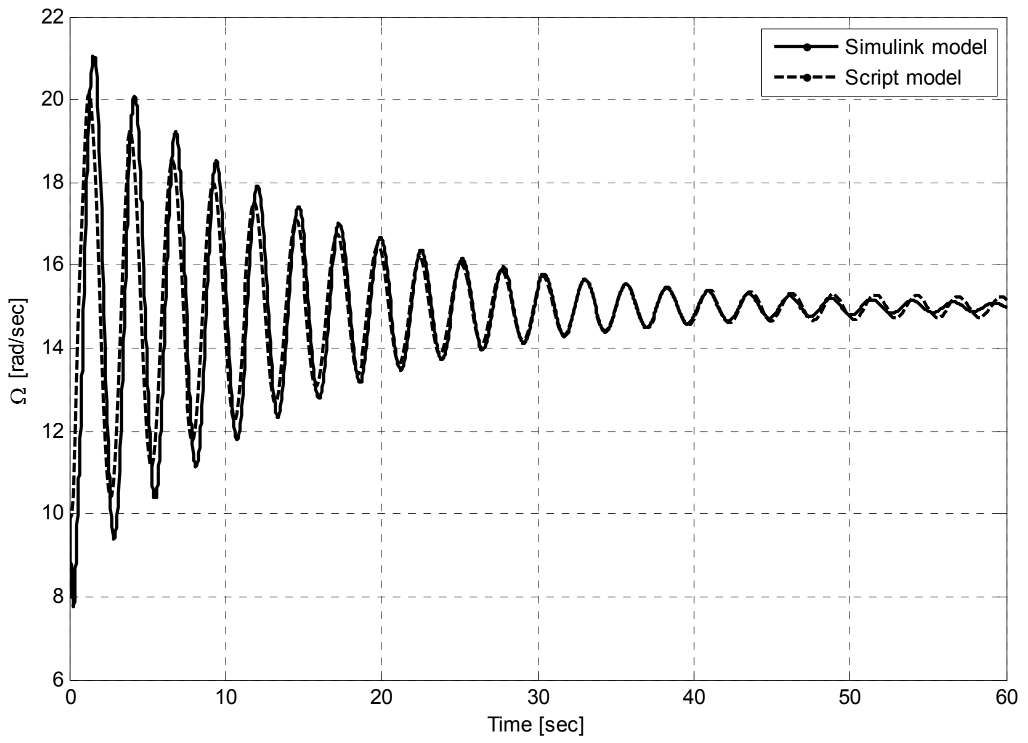

Figure 2.3: Simulation of Simulink model and script model

Validation:
Two sets of operational data were provided from NOV to undertake validation of the drill string model. The data were unfortunately not captured in a validation experiment; the data is just logs from a drilling operation.

One set were from an installation and commissioning of SoftSpeed which only consisted of data from the top of the string, i.e. top drive speed and torque. This will not be sufficient for the purpose of validation; the model has the top drive speed as input and bit speed as output and this logging series were not applicable for this purpose and were therefore rejected.

The other set also included data from the bit, i.e. the bit speed. This data was captured using NOV's drilling research tool DRT 2.0. This tool is placed at the end of the BHA and is capable of logging at very fast rates for a short time period (20-120 seconds). It should be kept in mind that the operational data provided was not from an appropriately designed and executed experiment, which makes the validation procedure limited. There were also used a mud motor at the bottom of the string which made large sections of the log useless. Fortunately the mud motor torque was included in the log so it was possible to pick out the sections where the mud motor was inoperative.

The well that the operational data originates from is somewhat sloped as it can be seen in the sketch in Figure 2.4. The slopes will lead to contact between the drill string and the walls of the well which makes the system more complicated. To get a proper validation of a model using data from a sloped well it would be necessary with measurements of the speed at several sections of the drill string.

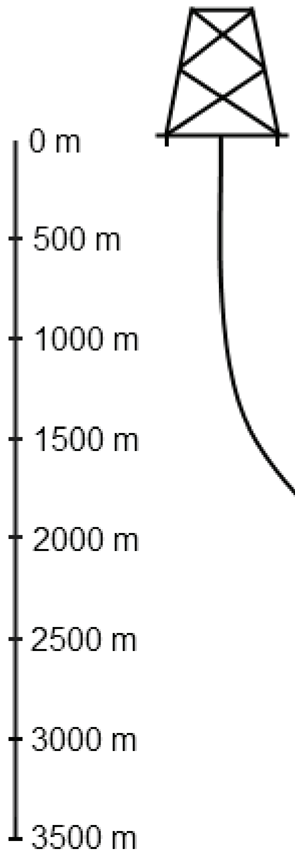

Figure 2.4: Slope of the well where logging was performed

The group considered two different possibilities for conducting the validation experiment; either use the operational data to develop a model using system identification or use the operational data as input on the Simulink model and compare the output from the model with the output in the log. The two methods are shown below.

System identification
One way to conduct a model validation is to adapt a model from logged data, and compare this model's dynamics with the dynamics of the model to be validated. To do this a system identification experiment using the logging data provided by NOV was performed using MathWorks' *System Identification Toolbox 7.3.1*. There was found an appropriate part from the log where the mud motor torque was zero and some perturbations were done. The log did unfortunately not contain any situations with stick-slip. As the dynamics that let stick-slip occur is included in the model created by NOV, it is necessary that a model created by system identification also contains this dynamics. It would be hard to believe that a model produced from this data should be a good model to simulate the stick-slip phenomenon.

Different methods from the toolbox were tested, and the best fit was achieved when using nonlinear ARX. There is also a choice on how many regressors that should be used; this was set to 50 both for inputs and outputs. This gave a 52.05 % fit, and the plot showing the output in the log and the output from the ARX model applying the same input is showed in Figure 2.5 (it should be noticed that rpm is the speed unit in the treatment of this system identification in distinction to the rest of the thesis where rad/sec is used).

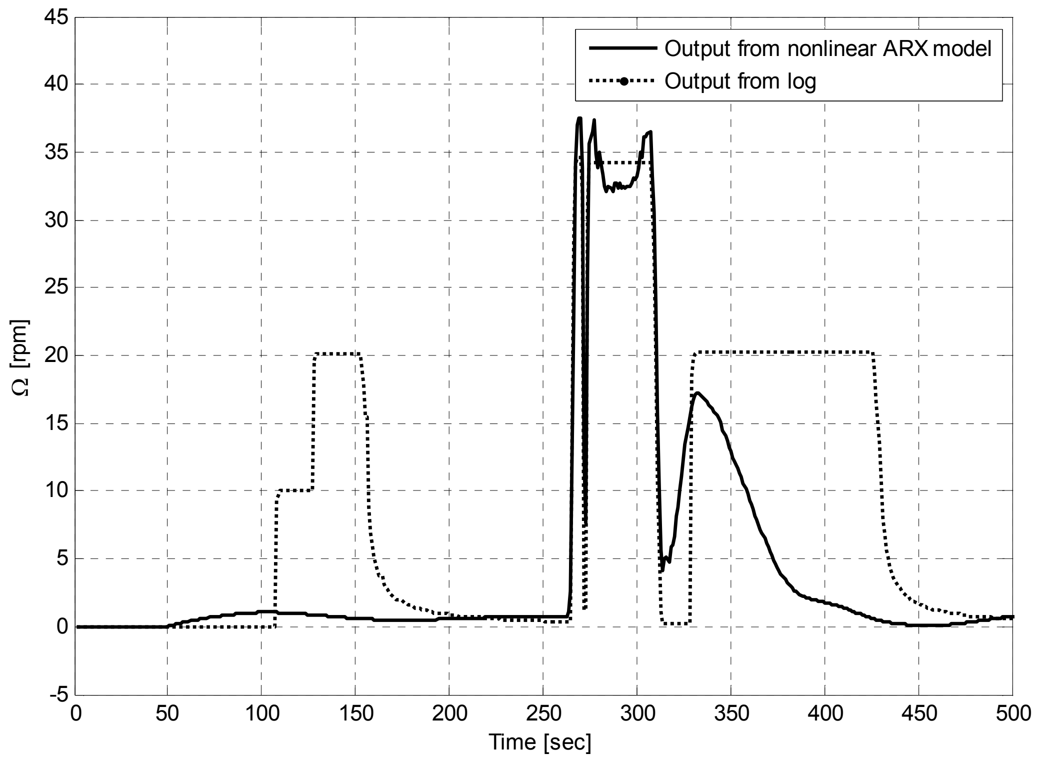

Figure 2.5: Comparison of model and logged output

As Figure 2.5 shows the output from the model follows the real output relatively good when large perturbations occurs. With smaller perturbations the model does not track the logged output to the same degree.

To compare the ARX model with the drill string model created by NOV, a step response was applied to both models. The slope of the well (Figure 2.4) was applied to the NOV model and is therefore expected to be similar to the ARX model. A constant input of 100 rpm was applied until the input steps to 120 rpm after 30 seconds. The plot of the models’ responses is shown in Figure 2.6.

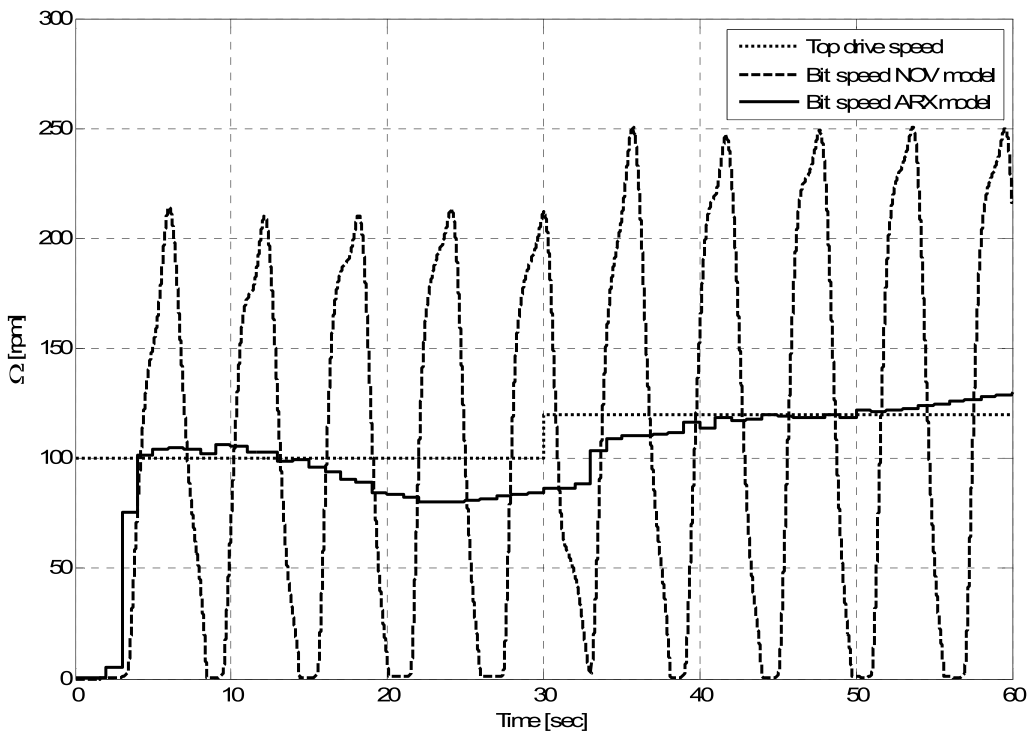

Figure 2.6: Comparison of NOV and ARX model

As Figure 2.6 shows, the ARX model does not contain dynamic information that let stick-slip occur, which is not a big surprise. To get an ARX model with this dynamics it is probably necessary that the data used in the system identification includes stick-slip, which was not the case with the operational data provided by NOV.

Direct comparison
Here the second possibility is tested using the operational data as input to NOV's Simulink model and comparing the output from the model with the output in the log. If the Simulink model is a good representation of a drill string, the bit speed will behave in a similar manner to the bit speed from the log.

All the initial conditions and input parameters that were available were implemented into the Simulink model and the top drive speed from the operational data was used as input. The output, the bit speed, was compared with the actual bit speed from the operational data. Both the input and the two outputs can be viewed in Figure 2.7. As the figure clearly shows are the outputs not exactly identical, but they behave in a similar manner. The bit speed of the NOV model has a more oscillating behaviour than the logged bit speed. There can be several reasons for this; e.g. there were not given any information about the physical measures of the drill string. String diameter, string wall thickness, length of BHA, weight on bit, steel density, etc. were not given and therefore such things were assumed to some reasonable standard values. Such parameters would of course influence the response of the string. By considering the plots in Figure 2.7 it seems like the drill string were the logging is taken is stiffer than the drill string in the NOV model.

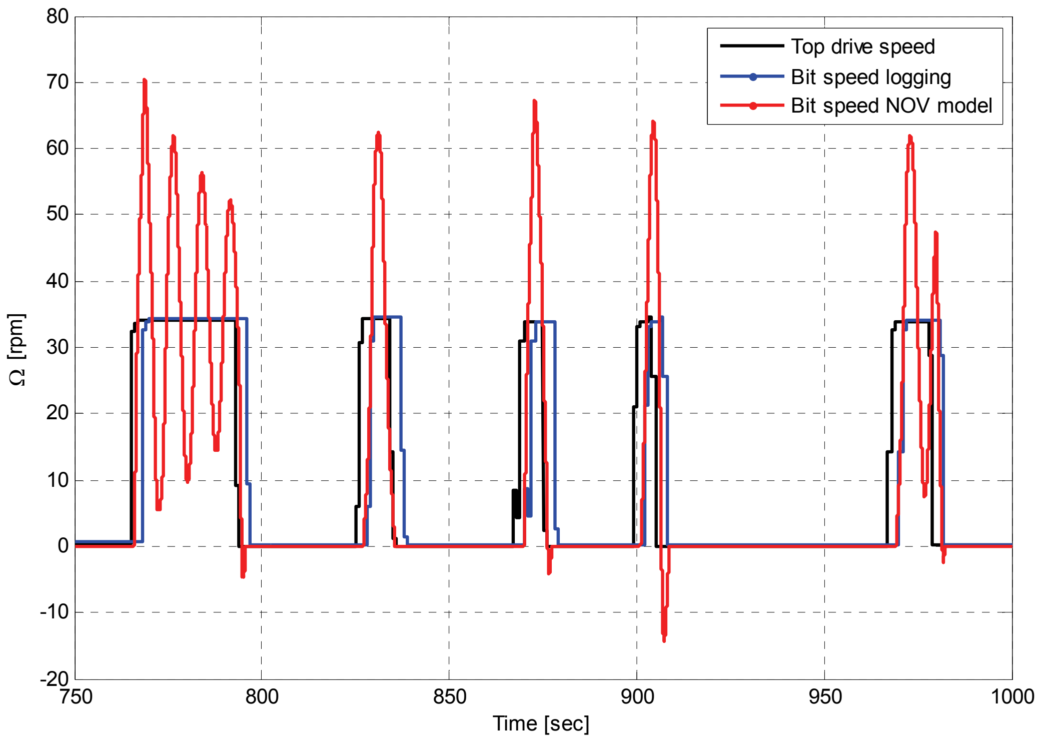

Figure 2.7: Simulations with operational data

There is too much uncertainty in this validation experiments to give a conclusion whether the NOV model is a good or poor approximation of a drill string. To do this there will be necessary with a planned experiment to gather data where the purpose is to use those data for validation. Complete information about the actual drill string and more measurements is also needed.

3 Model reduction

Modelling of dynamic systems is one of the most important issues when design, analysis and implementation of control systems are performed. Traditionally this type of modelling has been used as a representation of the real system, and quite simple PID-controllers have been designed from simulations of this model. In such design methods the computational time is less important; for example if the simulation takes 5 seconds or 2 minutes is in most cases irrelevant.

As modern computers are becoming faster and faster, it has been more usual to make use of a mathematical model of the process that is to be controlled, as a part of the controller, so-called model-based controllers. The model of the process that is included in the controller, often called internal model, is then processed in real-time. Therefore the internal model has to be of a somewhat low dimension to reduce the computational time, or else it will not be suitable for real-time applications.

Lately there has been established theory behind the algorithms used in model-based controllers that has proven why they work, and the algorithms are also proved to be stable. This has also led to a gain in usage of such types of controllers. Complex models still demand high computational requirements that make them unsuited for implementation in a model-based controller. One approach to solve this problem is by applying a model reduction technique and use the reduced model in the controller.

Figure 3.1: A reduced model of a somewhat complex system

Model reduction techniques will reduce the system, but still maintain the original system's most important input-output behaviour. Different methods have been suggested throughout the years, such as balanced truncation (Moore, 1981) and balanced residualization for linear systems (Fernando & Nicholson, 1982) and proper orthogonal decomposition (Karhunen, 1946; Loève, 1946) and balancing of empirical Gramians for nonlinear systems (Hahn & Edgar, 2002).

Balanced truncation is only suitable for linear systems and is based on a transformation of the original system. The transformed system is somewhat balanced; the states that are highly influenced by the input has also a great influence on the output. To do this the controllability and observability Gramians have to be found. A reduced model is obtained by simply truncate the states that have a small influence on the dynamics.

Balanced residualization is similar to balanced truncation, but it is based on the idea that derivates of the states with small influence on the dynamics can be approximated to zero while the rest of the system is retained. As for the balanced truncation are the less important states found through balancing and transformation of the Gramians.

Proper orthogonal decomposition, or Karhunen-Loève expansion, is a widely used method when it comes to model reduction; it has been used in fluid dynamics, compressible flow, aerodynamics and optimal control. One of the reasons for its broad usage is that it handles both linear and nonlinear systems. The method constructs a low-dimensional approximation in a reduced space. Snapshots (samples of trajectories) are taken either from experimental or simulation data. The proper orthogonal decomposition is then found by minimizing the error between the original snapshots and their representation in the reduced space. More about the proper orthogonal decomposition and the diversity of use of this method can be found in the articles by Kirby & Sirovich (1990), Glavaški & Marsden (1998), Rowley & Marsden (2000), Glegg & Devenport (2001), Krysl, Lall, & Marsden (2001), Azeez & Vakakis (2001) and Kerschen, Golinval, Vakakis, & Bergman (2005).

The method of balancing the empirical Gramians is similar to balanced truncation of linear systems. The main difference is how the Gramians are found; with a linear system can the Gramians be derived directly from the system equations, while they has to be found empiric based on experimental or simulation data for a nonlinear system.

In this thesis a model of a drill string has been reduced for the purpose of implementing it into a model predictive controller. This was done by the use of the method by balancing the empirical Gramians. This method is closely related to the method of balanced truncation used on linear systems, so the linear method will be explained first.

There were mainly two reasons why the method of balancing the empirical Gramians were chosen; it is a relatively new method it will be interesting to get familiar with and there where several good papers describing the method including complimentary examples. Another advantage of the method is that it only requires linear matrix computations which make it suitable for implementation. The principle around Gramians is also relatively easy to comprehend which makes this method suitable in an educational environment.

3.1 Balanced truncation for linear systems

The idea behind this method is to find a reduced model that still contains the most important dynamics found in the original system. The reduction is done by removing some of the states to achieve a lower order model. Since the goal is to retain the dynamics as much as possible, it will be necessary to find the states that have the largest contribution to the system's input-output behaviour. To do this job the method finds a state space representation with an equal number of states in addition to the Hankel singular values connected to each of these states, and the states are sorted in a descent way according to the singular values (balancing). These values are then used to decide which states that can be eliminated and which that needs to be retained (truncation).

In this section the balanced truncation method will be used to reduce a linear, time invariant system. First the theory will be explained, and then the theory will be applied to an example. Consider a linear system represented as a state space model of the standard form given by equation (3.1).

$$\begin{aligned} \dot{x} &= Ax + Bu \\ y &= Cx \end{aligned} \tag{3.1}$$

where

x	is the state vector
A	is the system matrix
u	is the input vector
B	is the input matrix
y	is the measurement vector
C	is the measurement matrix

To reduce such systems with this method one could follow the step-by-step procedure described below:

1. Compute Gramians using the original system equations
2. Compute balancing transformation
3. Balance Gramians and system
4. Determine size of reduced-order model
5. Define equations for the reduced-order model

In the following the theory behind the different steps will be derived.

Compute Gramians using the original system equations
In control theory it is well known that the Gramian matrices can be used to determine whether a system is controllable and/or observable. The controllability Gramian W_C and observability Gramian W_O are defined by equation (3.2) and (3.3).

$$W_C \triangleq \int_0^\infty e^{At} BB^T e^{A^T t} dt \tag{3.2}$$

$$W_O \triangleq \int_0^\infty e^{A^T t} C^T C e^{At} dt \tag{3.3}$$

To decide if a system is controllable and/or observable using W_C and W_O one have to determine if the Gramians are positive definite, i.e.

- if $W_C > 0$ the pair (A,B) is state controllable
- if $W_O > 0$ the pair (A,C) is state observable

As stated by Skogestad & Postlethwaite (2005), the linear Gramians can also be found as the unique positive semi definite solutions of the Lyapunov equations defined in (3.4) and (3.5)

$$AW_C + W_C A^T = -BB^T \tag{3.4}$$

$$A^T W_O + W_O A = -C^T C \tag{3.5}$$

The singular values of the controllability Gramian W_C can be used as a measure of how much energy that has to be applied to the system in order to move the corresponding state, and the singular values of the observability Gramian W_O is in distinction a measure of the energy that is produced by the corresponding state.

In the recent years there are introduced new areas in control theory where the Gramian matrices are utilized. One of these employments is to use them when working with different model reduction techniques, and in this thesis they will be used in the method of balanced truncation.

Compute balancing transformation
The balanced realization is by Skogestad & Postlethwaite (2005) defined as an asymptotically stable minimal realization in which the controllability and observability Gramians are equal and diagonal. This means that the balanced Gramian is given as showed in equation (3.6).

$$W_C = W_O = \Sigma = diag(\sigma_1, \sigma_2, ..., \sigma_n) = \begin{bmatrix} \sigma_1 & 0 & \cdots & 0 \\ 0 & \sigma_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \sigma_n \end{bmatrix} \tag{3.6}$$

where

$$\sigma_1 \geq \sigma_2 \geq ... \geq \sigma_n \geq 0$$

The σ_i's are called the Hankel singular values. When the Gramians are found using equation (3.4) and (3.5), they will almost certainly not be diagonal and equal, and it will be necessary to compute the balancing transformation T.

The calculation of T and T^1 is described by Phillips, Daniel & Silveira (2003), and the following steps needs to be conducted in order to find those matrices:

- Compute Cholesky factors $W_C = L_C L_C^T$ and $W_O = L_O L_O^T$
- Compute the singular value decomposition (SVD) of the Cholesky product $U\Sigma V = L_O^T L_C$ where Σ is diagonal positive and U, V have orthonormal columns

- Compute the balancing transformations *T* and T^{1} from the following formulas:
  $T = L_C V \Sigma^{-1/2}$ and $T^{-1} = \Sigma^{-1/2} U^T L_O^T$

Balance Gramians and system
The transformed Gramians will then be given by equations (3.7) and (3.8)

$$\overline{W}_C = T W_C T^T \tag{3.7}$$

$$\overline{W}_O = (T^{-1})^T W_O T^{-1} \tag{3.8}$$

The state vector for the balanced system will be given by

$$\overline{x} = Tx \tag{3.9}$$

and the balanced system will be given by

$$\begin{aligned} \dot{\overline{x}} &= T^{-1} A T \overline{x} + T^{-1} B u = \overline{A}\overline{x} + \overline{B} u \\ y &= C T \overline{x} = \overline{C}\overline{x} \end{aligned} \tag{3.10}$$

As one could notice from equation (3.10), the only symbols that are not noted with a bar are the system input *u* and measurement *y*; this means that all system matrices and states are changed. This is one of the disadvantages with this balanced representation of the system, all the states has lost their physical meaning.

Determine size of reduced-order model
When the Gramians are balanced they will be equal, and are often defined as *Σ* (as shown in equation (3.6)). The Hankel singular values along the diagonal gives a suggestion of how the states will influence the system; changes in the control signal will affect the state with the largest singular value, and the output will also be most sensitive to changes in this state. This relation can be utilized to determine how many states that can be removed but still maintain the most important dynamics of the original system. An often used approach is to keep the *i* first states, where σ_i is the first Hankel singular value that is at least one order of magnitude larger than σ_{i+1}.

Define equations for the reduced-order model
To find the equations for a reduced-order model originally consisting of *n* states, the balanced model should be partitioned as shown in equation (3.11)

$$\begin{aligned} \begin{pmatrix} \dot{\overline{x}}_1 \\ \dot{\overline{x}}_2 \end{pmatrix} &= \begin{pmatrix} \overline{A}_{11} & \overline{A}_{12} \\ \overline{A}_{21} & \overline{A}_{22} \end{pmatrix} \begin{pmatrix} \overline{x}_1 \\ \overline{x}_2 \end{pmatrix} + \begin{pmatrix} \overline{B}_1 \\ \overline{B}_2 \end{pmatrix} u \\ y &= \begin{pmatrix} \overline{C}_1 & \overline{C}_2 \end{pmatrix} \begin{pmatrix} \overline{x}_1 \\ \overline{x}_2 \end{pmatrix} \end{aligned} \tag{3.11}$$

where

$\overline{x}_1$ is a vector representing the *i* first states

$\overline{x}_2$ is a vector representing the states from *i+1* to *n*

The truncated model will then be given by

$$\begin{aligned} \dot{\bar{x}}_1 &= \bar{A}_{11}\bar{x}_1 + \bar{B}_1 u \\ y &= \bar{C}_1\bar{x}_1 \end{aligned} \tag{3.12}$$

Example 3.1: Linear string model with 6 states
In this example a drill string model consisting of 3 string elements will be considered. The model is developed as a mass-spring-damper-system, the same that is done in the Simulink model created by NOV. First the state space model needs to be derived; the system can be considered as a motor with several loads and elastic connections between the loads. This is illustrated in Figure 3.2 with *n* loads.

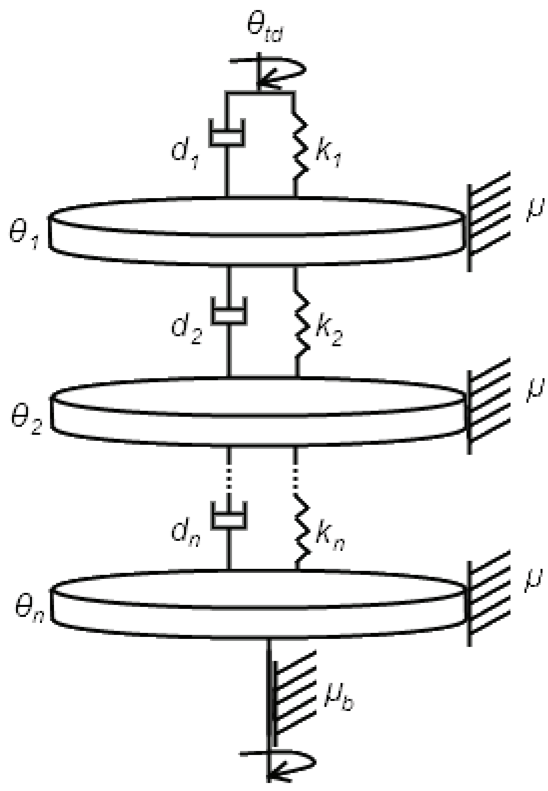

Figure 3.2: String model consisting of *n* elements

The model with 3 string elements will be given by the following equations:

$$\begin{aligned} j_1\ddot{\theta}_1 &= T_1 - T_2 - \mu\dot{\theta}_1 \\ j_2\ddot{\theta}_2 &= T_2 - T_3 - \mu\dot{\theta}_2 \\ j_3\ddot{\theta}_3 &= T_3 - (\mu + \mu_b)\dot{\theta}_3 \end{aligned} \tag{3.13}$$

$$\begin{aligned} T_1 &= d_1(\dot{\theta}_{td} - \dot{\theta}_1) + k_1(\theta_{td} - \theta_1) \\ T_2 &= d_2(\dot{\theta}_1 - \dot{\theta}_2) + k_2(\theta_1 - \theta_2) \\ T_3 &= d_3(\dot{\theta}_2 - \dot{\theta}_3) + k_3(\theta_2 - \theta_3) \end{aligned} \tag{3.14}$$

where

θ_{td}	is the angle of the top drive
θ_i	is the angle of joint i
k_i	is the stiffness coefficient for string section i
d_i	is the internal damping coefficient for string section i
μ_b	is the friction coefficient for the bit
μ	is the wall friction coefficient for the string sections
T_i	is the torque from string section i to $i+1$
j_i	is the inertia for string section i

If (3.13) and (3.14) are combined the model will become

$$\begin{aligned} \ddot{\theta}_1 &= \frac{1}{j_1}\left(d_1(\dot{\theta}_{td} - \dot{\theta}_1) + k_1(\theta_{td} - \theta_1) - d_2(\dot{\theta}_1 - \dot{\theta}_2) - k_2(\theta_1 - \theta_2) - \mu\dot{\theta}_1\right) \\ \ddot{\theta}_2 &= \frac{1}{j_2}\left(d_2(\dot{\theta}_1 - \dot{\theta}_2) + k_2(\theta_1 - \theta_2) - d_3(\dot{\theta}_2 - \dot{\theta}_3) - k_3(\theta_2 - \theta_3) - \mu\dot{\theta}_2\right) \\ \ddot{\theta}_3 &= \frac{1}{j_3}\left(d_3(\dot{\theta}_2 - \dot{\theta}_3) + k_3(\theta_2 - \theta_3) - (\mu + \mu_b)\dot{\theta}_3\right) \end{aligned} \tag{3.15}$$

To use standard notation for the states and input in (3.15), the definitions below are applied:

$$\begin{aligned} &\dot{\theta}_1 \triangleq x_1,\ \dot{\theta}_2 \triangleq x_2,\ \dot{\theta}_3 \triangleq x_3 \\ &\theta_{td} - \theta_1 \triangleq x_4,\ \theta_1 - \theta_2 \triangleq x_5,\ \theta_2 - \theta_3 \triangleq x_6 \\ &\dot{\theta}_{td} = u \end{aligned}$$

Inserting those definitions into (3.15) will give

$$\begin{aligned}
\dot{x}_1 &= \frac{1}{j_1}\left(d_1(u-x_1)+k_1x_4-d_2(x_1-x_2)-k_2x_5-\mu x_1\right)\\
\dot{x}_2 &= \frac{1}{j_2}\left(d_2(x_1-x_2)+k_2x_5-d_3(x_2-x_3)-k_3x_6-\mu x_2\right)\\
\dot{x}_3 &= \frac{1}{j_3}\left(d_3(x_2-x_3)+k_3x_6-(\mu+\mu_b)x_3\right)
\end{aligned} \tag{3.16}$$

To get a square system matrix the definitions of *x₄*, *x₅* and *x₆* above need to be included to the state space model. If they are differentiated, they can be expressed as shown in (3.17).

$$\begin{aligned}
\dot{x}_4 &= \dot{\theta}_{td}-\dot{\theta}_1 = u-x_1\\
\dot{x}_5 &= \dot{\theta}_1-\dot{\theta}_2 = x_1-x_2\\
\dot{x}_6 &= \dot{\theta}_2-\dot{\theta}_3 = x_2-x_3
\end{aligned} \tag{3.17}$$

If (3.16) and (3.17) are merged and reformulated the state space model can be written as shown in (3.18).

$$\begin{aligned}
\dot{x}_1 &= -\frac{1}{j_1}\left(d_1+d_2+\mu\right)x_1+\frac{d_2}{j_1}x_2+\frac{k_1}{j_1}x_4-\frac{k_2}{j_1}x_5+\frac{d_1}{j_1}u\\
\dot{x}_2 &= \frac{d_2}{j_2}x_1-\frac{1}{j_2}\left(d_2+d_3+\mu\right)x_2+\frac{d_3}{j_2}x_3+\frac{k_2}{j_2}x_5-\frac{k_3}{j_2}x_6\\
\dot{x}_3 &= \frac{d_3}{j_3}x_2-\frac{1}{j_3}\left(d_3+\mu+\mu_b\right)x_3+\frac{k_3}{j_3}x_6\\
\dot{x}_4 &= -x_1+u\\
\dot{x}_5 &= x_1-x_2\\
\dot{x}_6 &= x_2-x_3
\end{aligned} \tag{3.18}$$

It is desirable to use the standard matrix formulation as in (3.1) with bit speed x_3 as measurement, and state vector defined by

$$x = \begin{bmatrix} x_1\\ x_2\\ x_3\\ x_4\\ x_5\\ x_6 \end{bmatrix}$$

The system matrices will be given by

$$A = \begin{bmatrix} -\frac{1}{j_1}\left(d_1 + d_2 + \mu\right) & \frac{d_2}{j_1} & 0 & \frac{k_1}{j_1} & -\frac{k_2}{j_1} & 0 \\ \frac{d_2}{j_2} & -\frac{1}{j_2}\left(d_2 + d_3 + \mu\right) & \frac{d_3}{j_2} & 0 & \frac{k_2}{j_2} & -\frac{k_3}{j_2} \\ 0 & \frac{d_3}{j_3} & -\frac{1}{j_3}\left(d_3 + \mu + \mu_b\right) & 0 & 0 & \frac{k_3}{j_3} \\ -1 & 0 & 0 & 0 & 0 & 0 \\ 1 & -1 & 0 & 0 & 0 & 0 \\ 0 & 1 & -1 & 0 & 0 & 0 \end{bmatrix}$$

$$B = \begin{bmatrix} \frac{d_1}{j_1} \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}$$

$$C = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix}.$$

To be able to perform simulations, it will be necessary with numerical values for the different constants. In this simple approach it will be assumed that the two upper elements is the drill string and the lower part is the BHA. As the BHA has larger diameter than the drill string, the values are in this example assumed to be as follows:

$$k_1 = 1,\ k_2 = 1,\ k_3 = 10$$
$$j_1 = 5,\ j_2 = 5,\ j_3 = 10$$
$$d_1 = 0.5,\ \mathrm{d}_2 = 0.5,\ \mathrm{d}_3 = 1$$
$$\mu = 0.5$$
$$\mu_b = 10$$

With these values the system matrices will be:

$$A = \begin{bmatrix} -0.3 & 0.1 & 0 & 0.2 & -0.2 & 0 \\ 0.1 & -0.4 & 0.2 & 0 & 0.2 & -2 \\ 0 & 0.1 & -1.15 & 0 & 0 & 1 \\ -1 & 0 & 0 & 0 & 0 & 0 \\ 1 & -1 & 0 & 0 & 0 & 0 \\ 0 & 1 & -1 & 0 & 0 & 0 \end{bmatrix},\ B = \begin{bmatrix} 0.1 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \end{bmatrix},\ C = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix}$$

Now the system can be reduced by using the theory derived at the earlier in this section.

Compute Gramians using the original system equations
The controllability Gramian is found using (3.4)

$$AW_C + W_C A^T = -BB^T$$

$$\begin{bmatrix} -0.3 & 0.1 & 0 & 0.2 & -0.2 & 0 \\ 0.1 & -0.4 & 0.2 & 0 & 0.2 & -2 \\ 0 & 0.1 & -1.15 & 0 & 0 & 1 \\ -1 & 0 & 0 & 0 & 0 & 0 \\ 1 & -1 & 0 & 0 & 0 & 0 \\ 0 & 1 & -1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} w_{c11} & w_{c12} & w_{c13} & w_{c14} & w_{c15} & w_{c16} \\ w_{c21} & w_{c22} & w_{c23} & w_{c24} & w_{c25} & w_{c26} \\ w_{c31} & w_{c32} & w_{c33} & w_{c34} & w_{c35} & w_{c36} \\ w_{c41} & w_{c42} & w_{c43} & w_{c44} & w_{c45} & w_{c46} \\ w_{c51} & w_{c52} & w_{c53} & w_{c54} & w_{c55} & w_{c56} \\ w_{c61} & w_{c62} & w_{c63} & w_{c64} & w_{c65} & w_{c66} \end{bmatrix} +$$

$$\begin{bmatrix} w_{c11} & w_{c12} & w_{c13} & w_{c14} & w_{c15} & w_{c16} \\ w_{c21} & w_{c22} & w_{c23} & w_{c24} & w_{c25} & w_{c26} \\ w_{c31} & w_{c32} & w_{c33} & w_{c34} & w_{c35} & w_{c36} \\ w_{c41} & w_{c42} & w_{c43} & w_{c44} & w_{c45} & w_{c46} \\ w_{c51} & w_{c52} & w_{c53} & w_{c54} & w_{c55} & w_{c56} \\ w_{c61} & w_{c62} & w_{c63} & w_{c64} & w_{c65} & w_{c66} \end{bmatrix} \begin{bmatrix} -0.3 & 0.1 & 0 & 0.2 & -0.2 & 0 \\ 0.1 & -0.4 & 0.2 & 0 & 0.2 & -2 \\ 0 & 0.1 & -1.15 & 0 & 0 & 1 \\ -1 & 0 & 0 & 0 & 0 & 0 \\ 1 & -1 & 0 & 0 & 0 & 0 \\ 0 & 1 & -1 & 0 & 0 & 0 \end{bmatrix}^T = - \begin{bmatrix} 0.1 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \end{bmatrix} \begin{bmatrix} 0.1 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}^T$$

This will give 36 unknown and 36 equations, which means that it can be found a unique solution for the controllability Gramian. Since the Gramians are symmetric matrices, it will only be necessary to calculate the diagonal elements and the elements above or below the diagonal; the values can then be inserted directly to the opposite part of the matrix. So for an *n*x*n* Gramian it will be necessary to solve $0.5(n^2+n)$ equations; in this example the number of equations will be 21. In the same way (3.5) can be used to find the observability Gramian. After some algebra, the Gramians are found to be

$$W_C = \begin{bmatrix} 0.1832 & 0.0294 & 0.0134 & 0.5000 & 0.2648 & 0.0249 \\ 0.0294 & 0.0239 & 0.0232 & 0.2352 & 0.2648 & 0.0255 \\ 0.0134 & 0.0232 & 0.0242 & 0.2102 & 0.2643 & 0.0255 \\ 0.5000 & 0.2352 & 0.2102 & 3.5220 & 2.4734 & 0.2316 \\ 0.2648 & 0.2648 & 0.2643 & 2.4734 & 2.9777 & 0.2872 \\ 0.0249 & 0.0255 & 0.0255 & 0.2316 & 0.2872 & 0.0278 \end{bmatrix}$$

$$W_O = \begin{bmatrix} 0.0258 & 0.0118 & 0.0242 & 0.0000 & 0.0066 & -0.0137 \\ 0.0118 & 0.1045 & 0.0641 & -0.0066 & 0.0066 & 0.0408 \\ 0.0242 & 0.0641 & 0.3750 & -0.0104 & 0.0022 & 0.0815 \\ 0.0000 & -0.0066 & -0.0104 & 0.0239 & 0.0194 & 0.0155 \\ 0.0066 & 0.0066 & 0.0022 & 0.0194 & 0.0235 & 0.0067 \\ -0.0137 & 0.0408 & 0.0815 & 0.0155 & 0.0067 & 0.1612 \end{bmatrix}$$

Compute balancing transformation
As seen above, the controllability and observability Gramians are not diagonal and equal (as expected) so the balancing transformation needs to be computed. The step-by-step procedure described earlier is used to find *Σ*, *T* and *T*$^{-1}$, and they are given by

$$\Sigma = \begin{bmatrix} 0.5326 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0.0812 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.0494 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.0023 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.0022 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0.0009 \end{bmatrix}$$

$$T = \begin{bmatrix} -0.3106 & 0.7341 & -1.3344 & -0.1199 & 0.1155 & 0.0252 \\ -0.2045 & -0.1316 & -0.0522 & 0.1518 & 0.0939 & -0.0366 \\ -0.1941 & -0.2131 & 0.0918 & -0.0361 & -0.0027 & -0.0448 \\ -2.3994 & 2.3287 & 0.5446 & 0.0812 & 0.3998 & -0.2276 \\ -2.2333 & -1.9802 & -0.2062 & -0.0096 & -0.4961 & 0.1861 \\ -0.2129 & -0.2093 & -0.0325 & -0.0707 & 0.0416 & 0.0803 \end{bmatrix}$$

$$T^{-1} = \begin{bmatrix} -0.0505 & -0.0847 & -0.1706 & -0.1891 & -0.1960 & -0.1997 \\ 0.0260 & -0.6835 & -1.4304 & 0.2105 & 0.0087 & -0.5402 \\ -0.6828 & -0.4374 & -0.2020 & 0.1601 & -0.0685 & 0.5152 \\ -0.5460 & 3.7350 & -5.7615 & 0.0185 & 0.4336 & -2.2936 \\ 0.0908 & 3.0414 & 2.6527 & -0.0064 & -1.0059 & 5.1510 \\ -0.8706 & -0.4722 & -10.7154 & 0.1318 & 0.3784 & 6.0388 \end{bmatrix}$$

Balance Gramians and system
The transformed Gramians will now be given by *Σ* and the state vector $\bar{x}$ can be found using equation (3.9), and in this case the new state vector is given by

$$\bar{x} = Tx$$

$$\begin{bmatrix} \bar{x}_1 \\ \bar{x}_2 \\ \bar{x}_3 \\ \bar{x}_4 \\ \bar{x}_5 \\ \bar{x}_6 \end{bmatrix} = \begin{bmatrix} -0.3106x_1+0.7341x_2-1.3344x_3-0.1199x_4+0.1155x_5+0.0252x_6 \\ -0.2045x_1-0.1316x_2-0.0522x_3+0.1518x_4+0.0939x_5-0.0366x_6 \\ -0.1941x_1-0.2131x_2+0.0918x_3-0.0361x_4-0.0027x_5-0.0448x_6 \\ -2.3994x_1+2.3287x_2+0.5446x_3+0.0812x_4+0.3998x_5-0.2276x_6 \\ -2.2333x_1-1.9802x_2-0.2062x_3-0.0096x_4-0.4961x_5+0.1861x_6 \\ -0.2129x_1-0.2093x_2-0.0325x_3-0.0707x_4+0.0416x_5+0.0803x_6 \end{bmatrix}$$

The transformed system matrices $\bar{A}$, $\bar{B}$ and $\bar{C}$ are found by using equation (3.10), and will be given by

$$\bar{A} = \begin{bmatrix} -0.0354 & -0.0917 & 0.0306 & -0.0131 & -0.0010 & -0.0163 \\ 0.0917 & -0.2795 & 0.6149 & -0.0975 & -0.0068 & -0.1162 \\ 0.0306 & -0.6149 & -0.0853 & 0.0641 & 0.0052 & 0.0846 \\ -0.0131 & 0.0975 & 0.0641 & -0.2817 & -1.3908 & -1.1090 \\ 0.0010 & -0.0068 & -0.0052 & 1.3908 & -0.0016 & -0.0386 \\ 0.0163 & -0.1162 & -0.0846 & 1.1090 & -0.0386 & -1.1666 \end{bmatrix}$$

$$\bar{B} = \begin{bmatrix} -0.1941 \\ 0.2131 \\ 0.0918 \\ -0.0361 \\ 0.0027 \\ 0.0448 \end{bmatrix}$$

$$\bar{C} = \begin{bmatrix} -0.1941 & -0.2131 & 0.0918 & -0.0361 & -0.0027 & -0.0448 \end{bmatrix}$$

Since this only will be another state space representation of the original system, step responses and simulations will be equal ($eig(A) = eig(\bar{A})$). It was performed different simulations just to confirm this, and one of these plots is found in Figure 3.3. Here both systems' outputs are steady at 7 rad/sec before a step to 10 rad/sec is applied to the input after 20 seconds. As the figure shows the responses are almost identical; the small deviation (almost impossible to see in the plot) between them is because the numbers in the transformed system matrices are rounded off to four decimals.

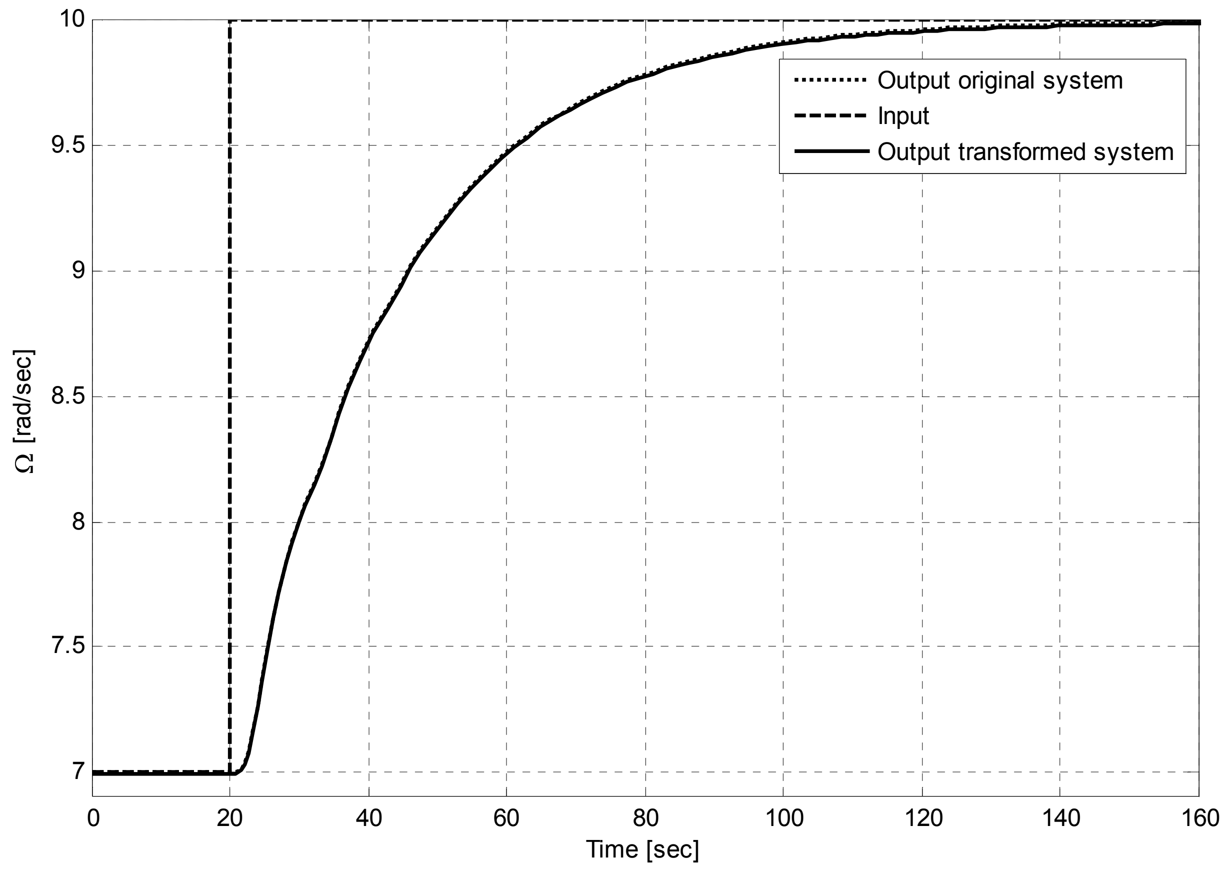

Figure 3.3: Comparison of original and transformed system

Determine size of reduced-order model
As seen from *Σ*, there is about one order of magnitude between Hankel singular value 1 and 2. From the approach presented earlier in this section, this means that if the system is reduced to 1 state, the most important dynamics of the system will still be maintained. The 3rd singular value has the same order of magnitude as the 2nd, but the 4th is again one lower than the 3rd.

To compare with different sizes of the reduced model it was from these considerations conducted simulation with the $\overline{A}_{11}$ matrix as a scalar (see equation (3.11)), 2x2 and 4x4 with the corresponding $\overline{B}_1$'s and $\overline{C}_1$'s. The simulations were done with the input signal

$$u = \begin{cases} 1 & 0 \le t < 150 \\ 2 & 150 \le t < 300 \end{cases}.$$

The simulations are compared in Figure 3.4.

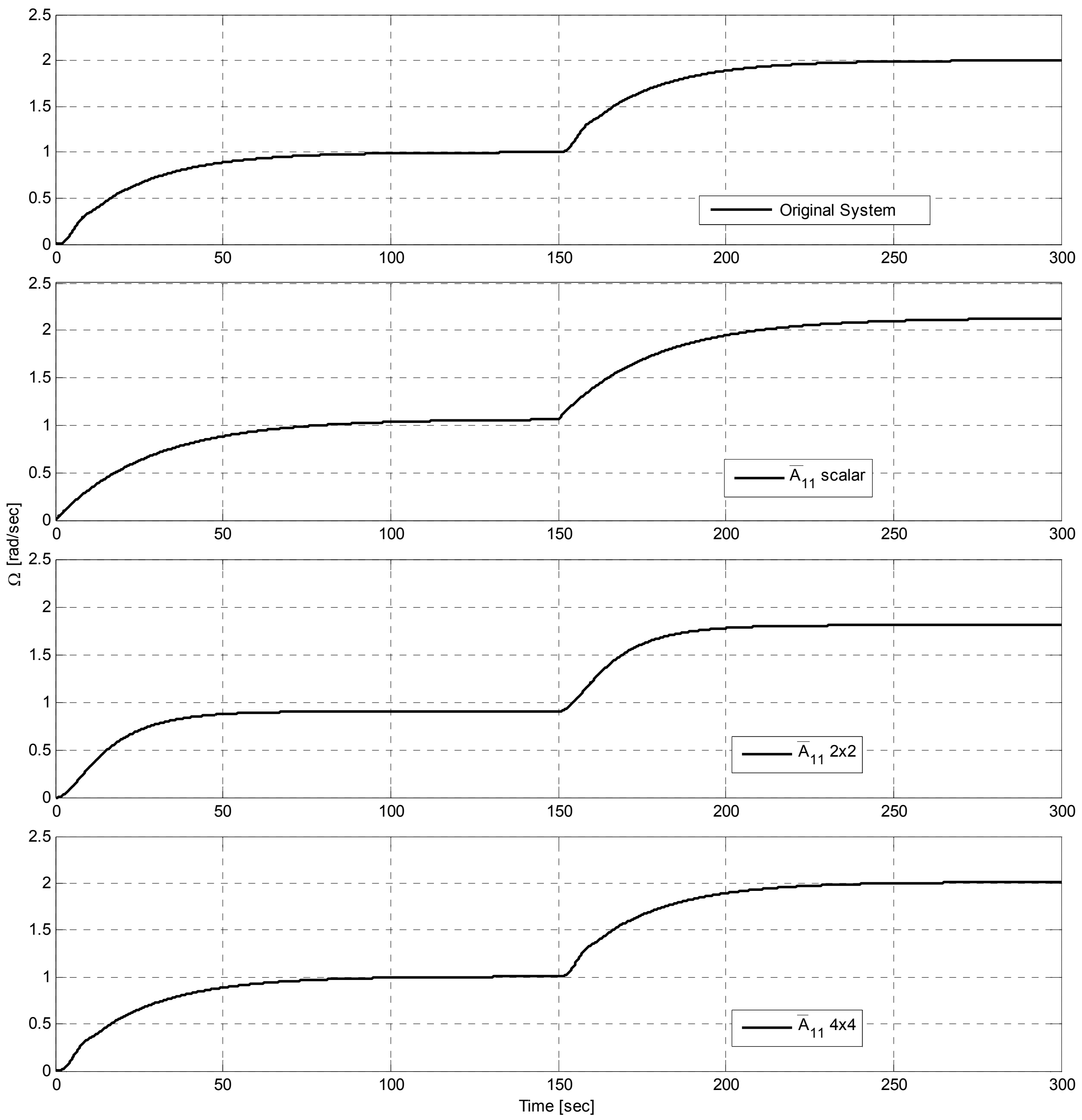

Figure 3.4: Comparison of bit speed with different sizes of truncated models

As seen in Figure 3.4, already with $\bar{A}_{11}$ as a scalar, the dominating time constant and most important dynamics are taken care of. There will indeed be a small error stationary, both with $\bar{A}_{11}$ as a scalar and as a 2x2 matrix. When $\bar{A}_{11}$ is extended to a 4x4 matrix this error will in fact disappear.

Define equations for the reduced-order model

Since there are dealt with model reduction with the purpose of using the reduced models in an MPC it is not required that the model is perfect; it will be mechanisms included in the MPC that will take care of this uncertainty.

So in this example the scalar model will give sufficient information to be used in an MPC; consequently the truncated model will be given by

$$\begin{aligned}\dot{\bar{x}}_1 &= \bar{A}_{11}\bar{x}_1 + \bar{B}_1 u = -0.0354\bar{x}_1 - 0.1941u \\ y &= \bar{C}_1\bar{x}_1 = -0.1941\bar{x}_1\end{aligned}$$

3.2 Balancing of empirical Gramians for nonlinear systems

Most systems exhibits some kind of nonlinear behaviour and it can in many cases be sufficient to linearize this and still get a satisfactory model to be used in a model-based controller, but in many cases will this be insufficient and give a unsatisfactory performance. A model-based controller that is growing in popularity is nonlinear model predictive control. Many complex models require too much computational time and therefore have to undergo some kind of reduction to be used in NMPC.

When working with nonlinear systems the balancing routines for linear systems are insufficient. The method that will be presented here for reducing nonlinear systems has the same framework as the balanced truncation method for linear systems; the main difference is how the Gramians are calculated. Hahn & Edgar (2002) has shown some interesting results using this method and the possibilities it gives.

As for all control problems stability of the system is of great concern and it is therefore natural to consider the following; will the reduced model be stable if the original system is stable? There does not exist a definite answer to this today as far as the authors of this thesis know. There is no guarantee that stabilizing states are included in the reduced model, but in general will the method give stable reductions. There are however some cases where the reduced models can become unstable. Instability can occur in cases where

- the original model is highly nonlinear
- the reduced model is operated in a region that is larger than the region used when constructing the empirical Gramians
- the model is reduced too much which can result in a reduced model where some of the stabilizing behaviour are cut off

Instead of using the ordinary Gramians, is the empirical Gramians used as defined by Lall et al. (1999). As in the previous section will the theory of the method be presented first, and then some examples will follow. Consider a nonlinear system given by equation (3.19)

$$\begin{aligned}\dot{x}(t) &= f\big(x(t), u(t)\big) \\ y(t) &= h\big(x(t)\big)\end{aligned} \tag{3.19}$$

where

x	is the state vector
u	is the input vector
y	is the measurement vector
f, h	are nonlinear functions

As for the case with a linear system, can this system be reduced following almost the same step-by-step procedure as described below:

1. Compute Gramians using experimental or simulation data
2. Compute balancing transformation
3. Balance Gramians and system
4. Determine size of reduced-order model
5. Define equations for the reduced-order model

Compute Gramians using experimental or simulation data

The following sets have to be defined to compute the empirical Gramians:

$$T^n = \left\{T_1, ..., T_r;\ T_i \in \mathbb{R}^{n\times n},\ T_i^T T_i = I,\ i = 1, ..., r\right\}$$

$$M = \{c_1, ..., c_s;\ c_i \in \mathbb{R},\ c_i > 0,\ i = 1, ..., s\}$$

$$E^n = \left\{e_1, ..., e_n;\ \text{standard unit vectors in } \mathbb{R}^n\right\}$$

$$r = \text{ number of matrices for excitation/perturbation directions}$$

$$s = \text{ number of different excitation/perturbation sizes for each direction}$$

$$n = \text{ number of inputs } (\text{controllabilty}) \text{ or number of states } (\text{observability})$$

where

T is a set containing *r* orthogonal *n*x*n* matrices; the matrices for perturbation/excitation directions.

M is a set containing *s* positive constants where *s* represents the number of different perturbation/excitation sizes for each direction.

E is a set containing *n* unit vectors.

Definition 3.1
Let T^p, E^p, and M be given sets as described above, where p is the number of inputs. The empirical controllability Gramian is defined by equation (3.20).

$$W_C = \sum_{l=1}^{r} \sum_{m=1}^{s} \sum_{i=1}^{p} \frac{1}{rsc_m^2} \int_0^\infty \Phi^{ilm}(t)\, dt \tag{3.20}$$

where

$\Phi^{ilm}(t) \in \mathbb{R}^{n\times n}$ is given by $\Phi^{ilm}(t) = \left(x^{ilm}(t) - x_0^{ilm}\right)\left(x^{ilm}(t) - x_0^{ilm}\right)^T$

and $x^{ilm}(t)$ is the state of the nonlinear system corresponding to the impulse input given by $u(t) = c_m T_l e_i \delta(t) + u_0$ and x_0^{ilm} refers to the steady state of the system

Definition 3.2
Let T^n, E^n, and M be given sets as described above, where n is the number of states. The empirical observability Gramian is defined by equation (3.21).

$$W_O = \sum_{l=1}^{r} \sum_{m=1}^{s} \frac{1}{rsc_m^2} \int_0^\infty T_l \Psi^{lm}(t) T_l^T \, dt \tag{3.21}$$

where

$\Psi^{lm}(t) \in \mathbb{R}^{n \times n}$ is given by $\Psi_{ij}^{lm}(t) = \left(y^{ilm}(t) - y_0^{ilm}\right)\left(y^{jlm}(t) - y_0^{jlm}\right)$

and $y^{ilm}(t)$ is the output of the nonlinear system corresponding to the initial condition given by $x(0) = c_m T_l e_i + x_0$ and y_0^{ilm} refers to the output measurement corresponding to the steady state of the system

---

As opposed to the Gramians for the linear case, the empirical Gramians for the nonlinear case is somewhat more complicated. As seen from definition 3.1 and 3.2 they will be based on information from the system, either from experimental or simulation data. An important issue regarding the empirical Gramians are that the data has to be collected within the region of operation for the process to capture the nonlinear behaviour of the process within that region.

For non control-affine systems the covariance matrices are used instead of the empirical Gramians (Hahn & Edgar, 2002); the difference being that the covariance matrices can be computed for different input types and therefore are more applicable to systems that are not control affine. For simplicity are both referred to as empirical Gramians in this thesis; they are both used as a description of the input-output behaviour.

A question arising is how to choose *r* and *s* and the constants $c_1,\dots,c_s$. There is not an exact science for choosing these values; they have to be chosen for the specific system and the specific operating region. This is usually done with a trial-and-error method.

It is shown by Lall et al. (1999) that both the empirical controllability Gramian and observability Gramian reduce to linear Gramians for linear systems.

As for the linear case, the singular values of the controllability Gramian W_C can be used as a measure of how much energy that has to be applied to the system in order to move the corresponding state, and the singular values of the observability Gramian W_O is in distinction a measure of the energy that is produced by the corresponding state.

Compute balancing transformation
The balancing transformation are computed in the same way as for the linear case, see section 3.1.

Balance Gramians and system
The transformed Gramians will as for the linear case be given by

$$\overline{W}_C = TW_C T^T \tag{3.7}$$

$$\overline{W}_O = (T^{-1})^T W_O T^{-1} \tag{3.8}$$

and the state vector for the balanced system will be given by

$$\overline{x} = Tx \tag{3.9}$$

The transformed system is then given by

$$\begin{aligned} \dot{\overline{x}} &= Tf\left(T^{-1}\overline{x}(t), u(t)\right) \\ y &= h\left(T^{-1}\overline{x}(t)\right) \end{aligned} \tag{3.22}$$

Determine size of reduced-order model
The size of the reduced-order model is determined in the same way as for the linear case; by looking at the Hankel singular values of the balanced Gramian, see section 3.1.

Define equations for the reduced-order model
The equations for the reduced-order model are found using the following equation

$$\begin{aligned} \dot{\overline{x}}_1(t) &= PTf\left(T^{-1}\overline{x}(t), u(t)\right) \\ \overline{x}_2(t) &= \overline{x}_{2ss}(0) \\ y(t) &= h\left(T^{-1}\overline{x}(t)\right) \end{aligned} \tag{3.23}$$

where

$P = \begin{bmatrix} I & 0 \end{bmatrix}$ is a projection matrix which has the rank of the original system where I is the identity matrix with the same rank as the reduced system

This will lead to the following reduced model

$$\begin{aligned} \dot{\overline{x}}_1(t) &= PTf\left(T^{-1}\overline{x}(t), u(t)\right) \\ y(t) &= h\left(T^{-1}\overline{x}(t)\right) \end{aligned} \tag{3.24}$$

The method described in this chapter was implemented in MATLAB, see Appendix B for information about the codes. It should be noticed that before computing the empirical Gramians and balancing, the system of interest is scaled. This is done because a state changing by orders of magnitude can be more important than a state that hardly changes, even if the steady state value of it has a smaller absolute value. For a general nonlinear system given by (3.19) where x_{ss} and u_{ss} represent the steady state values of x and u is two quantities introduced;

$$
\begin{aligned}
T_x &= diag\left(x_{ss}\right) \\
T_u &= diag\left(u_{ss}\right)
\end{aligned}
$$

The scaled system is then given by

$$
\begin{aligned}
&\tilde{x} = T_x^{-1} \\
&\tilde{u} = T_u^{-1} \\
&\Downarrow \\
&\dot{\tilde{x}} = T_x^{-1} f\left(T_x \tilde{x}, T_u \tilde{u}\right) \\
&y = h\left(T_x \tilde{x}\right)
\end{aligned}
$$

In the following example a small system is treated to see how the model is reduced using balancing of the empirical Gramians.

Example 3.2: Theoretical example with two states
In this theoretical example a small two-state system is treated. This is done just for simplicity, so it should be possible to do the calculations explicit to get a complete review of the theory.

Consider the system given by equation (3.25).

$$
\begin{aligned}
\dot{x}_1 &= -x_1 x_2 - x_1 + u \\
\dot{x}_2 &= x_1^2 - x_2 \\
y &= x_1
\end{aligned}
\tag{3.25}
$$

This system has one scalar input and one scalar output, in addition to the two states. This is just a theoretical example (no physical meaning) with the purpose of showing how the method of balancing the empirical Gramians works and what the reduced system will look like.

The system can be proven to be asymptotically stable by considering the following storage function

$$
V\left(x\right) = \frac{x_1^2}{2} + \frac{x_2^2}{2} \tag{3.26}
$$

As stated in Khalil (2002) is a system on the form in (3.19) said to be passive if

$$
u^T y \geq \dot{V} = \frac{\partial V}{\partial x} f\left(x,u\right) \ \forall \left(x,u\right) \in \mathbb{R}^n \times \mathbb{R}^p
$$

And strictly passive if

$$u^T y \geq \dot{V} + \psi(x) \text{ for some positive definite function } \psi$$

By using the storage function in (3.26) is it clear that

$$\dot{V}(x) = x_1\dot{x}_1 + x_2\dot{x}_2 = -x_1^2 x_2 - x_1^2 + x_1 u + x_1^2 x_2 - x_2^2 = x_1 u - x_1^2 - x_2^2 = u^T y - x_1^2 - x_2^2$$

The system is passive and also strictly passive. This can be used to say something about the stability properties of the system on the form (3.19). The origin of $\dot{x} = f(x,0)$ is asymptotically stable if the system is strictly passive; which it is in this example. A proof of this can be found in Appendix A.

Compute Gramians using experimental or simulation data
When the empirical Gramians are computed from definition 3.1 and 3.2 they result in

$$W_C = \begin{bmatrix} 0.2345 & 0.3287 \\ 0.3287 & 0.8737 \end{bmatrix}$$
$$W_O = \begin{bmatrix} 0.1571 & -0.0182 \\ -0.0182 & 0.0084 \end{bmatrix}$$

As already mentioned it is important to find the empirical Gramians for the region of operation for where the process is to be controlled. For this specific example the system was implemented in Simulink where steady state values for input and states were found. Definition 3.1 and 3.2 were used with two directions (up/down) and only one small excitation size; i.e. impulse up/down.

Compute balancing transformation
As expected are the controllability and observability Gramians not diagonal and equal, so the transformation matrix that balances the system has to be computed. *Σ*, *T* and T^{1} will be given by

$$\Sigma = \begin{bmatrix} 0.1701 & 0 \\ 0 & 0.0576 \end{bmatrix}$$

$$T = \begin{bmatrix} 0.9189 & -0.0499 \\ -0.4834 & 0.3724 \end{bmatrix}$$

$$T^{-1} = \begin{bmatrix} 1.1708 & 0.1568 \\ 1.5197 & 2.8890 \end{bmatrix}$$

Balance Gramians and system
The next step will be to balance the system using equation (3.22), but first some calculations need to be done.

$$T^{-1}\bar{x} = \begin{bmatrix} 1.1708\bar{x}_1 + 0.1568\bar{x}_2 \\ 1.5197\bar{x}_1 + 2.8890\bar{x}_2 \end{bmatrix}$$

$$f\left(T^{-1}\bar{x}(t), u(t)\right) = \begin{bmatrix} -(1.1708\bar{x}_1 + 0.1568\bar{x}_2)(1.5197\bar{x}_1 + 2.8890\bar{x}_2) - 1.1708\bar{x}_1 - 0.1568\bar{x}_2 + u \\ (1.1708\bar{x}_1 + 0.1568\bar{x}_2)^2 - 1.5197\bar{x}_1 - 2.8890\bar{x}_2 \end{bmatrix}$$

$$\Downarrow$$

$$f\left(T^{-1}\bar{x}(t), u(t)\right) = \begin{bmatrix} -1.779\bar{x}_1^2 - 3.620\bar{x}_1\bar{x}_2 - 0.453\bar{x}_2^2 - 1.1708\bar{x}_1 - 0.1568\bar{x}_2 + u \\ 1.37\bar{x}_1^2 + 0.367\bar{x}_1\bar{x}_2 + 0.0246\bar{x}_2^2 - 1.5197\bar{x}_1 - 2.8890\bar{x}_2 \end{bmatrix}$$

The balanced system is then given by

$$\dot{\bar{x}} = Tf\left(T^{-1}\bar{x}(t), u(t)\right) = \begin{bmatrix} \dot{\bar{x}}_1 \\ \dot{\bar{x}}_2 \end{bmatrix}$$

$$\Downarrow$$

$$\dot{\bar{x}} = \begin{bmatrix} -1.6984\bar{x}_1^2 - 0.417\bar{x}_2^2 - 3.51\bar{x}_1\bar{x}_2 - \bar{x}_1 + 0.9189u \\ 1.3702\bar{x}_1^2 + 0.228\bar{x}_2^2 + 1.887\bar{x}_1\bar{x}_2 + 2.844\cdot 10^{-5}\bar{x}_1 - \bar{x}_2 - 0.4834u \end{bmatrix}$$

Determine size of reduced-order model
Since there are only two states in the original system, is the only possibility to reduce it to one state. As seen from *Σ* is the singular value related to the 2nd state one order of magnitude smaller than for the 1st state. This indicates that the 2nd state will contribute significantly less to the input-output behaviour of the system. A system with only one state should therefore be able to describe the system's input-output behaviour appropriately.

Define equations for the reduced-order model
When reducing the system to one state the projection matrix *P* will be given by

$$P = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}$$

The truncated model will then be given by

$$\dot{\bar{x}}_1 = PTf\left(T^{-1}\bar{x}(t), u(t)\right) = -1.6984\bar{x}_1^2 - 0.417\bar{x}_2^2 - 3.51\bar{x}_1\bar{x}_2 - \bar{x}_1 + 0.9189u$$

$\bar{x}_2$ is still left in the truncated system equation, and it can be seen from (3.23) that the steady state value of $\bar{x}_2$ will be needed. This steady state value was found from simulation of the balanced system, and was found to be

$$\bar{x}_2 = \bar{x}_{2ss} = -0.1482$$

Inserting this into the system equation will give

$$\dot{\bar{x}}_1 = -1.6984\bar{x}_1^2 - 0.48\bar{x}_1 - 0.00916 + 0.9189u$$

The measurement will be given by

$$y = h\left(T^{-1}\bar{x}\right) = 1.1708\bar{x}_1 + 0.1568\bar{x}_2 = 1.1708\bar{x}_1 - 0.02324$$

The response for the full-order model and the reduced-order with a 10% change in the input can be seen in Figure 3.5. As seen in the figure, some of the dynamics are lost, and there is also an error in the steady state value. This is to be expected, a reduction from two to only one state is a significant reduction and some of the dynamics will be lost.

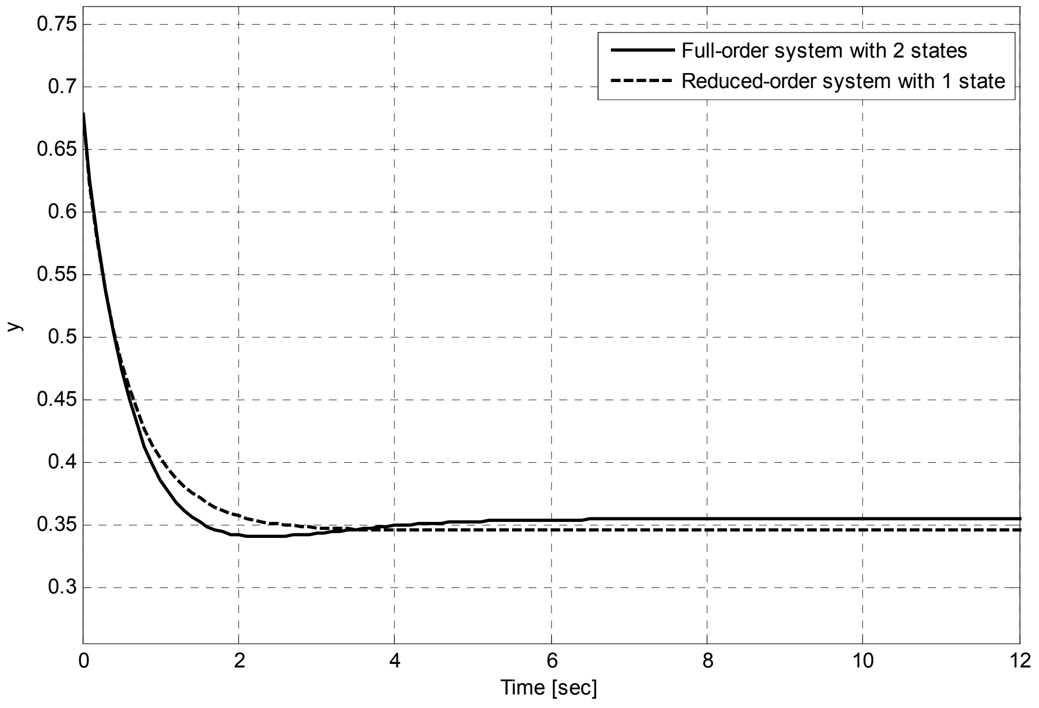

Figure 3.5: Comparison of full-order and reduced-order theoretical example

Example 3.3: String model with 6 states and nonlinear friction component at bit
In this example will the same string model that was used in Example 3.1 be examined, but this time with a nonlinear friction component representing the friction at the bit. The system is represented by the following equations

$$\dot{x}_1 = \frac{1}{j_1}\left(d_1(u - x_1) + k_1x_4 - d_2(x_1 - x_2) - k_2x_5 - \mu x_1\right)$$
$$\dot{x}_2 = \frac{1}{j_2}\left(d_2(x_1 - x_2) + k_2x_5 - d_3(x_2 - x_3) - k_3x_6 - \mu x_2\right)$$
$$\dot{x}_3 = \frac{1}{j_3}\left(d_3(x_2 - x_3) + k_3x_6 - \mu x_3 - T_b\right)$$
$$\dot{x}_4 = \dot{\theta}_{td} - \dot{\theta}_1 = u - x_1$$
$$\dot{x}_5 = \dot{\theta}_1 - \dot{\theta}_2 = x_1 - x_2$$
$$\dot{x}_6 = \dot{\theta}_2 - \dot{\theta}_3 = x_2 - x_3$$

where

T_b is representing the nonlinear friction component at the bit and is given by

$$T_b = \mu N r \left(\frac{x_3}{\sqrt{x_3^2 + \Omega_0^2}} + \frac{p\Omega_0 x_3}{x_3^2 + \Omega_0^2} \right) - Dx_3\left(\frac{x_3}{\Omega_1} - 1 \right)$$

where

μ	is the friction coeffcient
N	is the force vector
r	is the contact radius vector
Ω_0	is the transistion speed for the string
Ω_1	is the transistion speed for the well
D	is the linear damping vector
p	is the start friction parameter

T_b is the same friction term that is used in NOV's model, but for simplicity it is only added to the last element of the string, i.e. the bit. In NOV's model it is added to every element of the string, but only with large influence to the parts of the string which are bent and likely to come in contact with the walls of the well. In this example is a vertical string considered, so the only significant contribution to the friction will be from the bit.

The parameters for the string is the same as in Example 3.1, while the parameters for the nonlinear bit friction were chosen somewhat randomly and from the data in the Simulink model provided by NOV.

The empirical Gramians were computed and balanced which give the following singular values of the balanced Gramian

$$\Sigma = \begin{bmatrix} 451.63 & 0 & 0 & 0 & 0 & 0 \\ 0 & 6.1344 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2.2381 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.1886 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.1544 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0.1121 \end{bmatrix}$$

From Σ it is clear that the 1st state of the balanced system has the largest contribution to the input-output behaviour. Then the 2nd and 3rd state will have somewhat equal contribution, but less than the 1st state since they are two orders of magnitude smaller than the 1st state. The 4th, 5th and 6th state will have the smallest contribution to the input-output behaviour; they are at least one order of magnitude smaller than the previous states. A reduced system with only three states should therefore give a satisfactory description of the system input-output behaviour.

Another solution to the problem could be to linearize the system around an operating point. The linear system will require less computational power, but it will in some cases nevertheless not be sufficient to characterize the original systems input-output behaviour needed for tight control. The string model used in this example was linearized around its operating point, 0.8 rad/sec, to be able to compare it with the nonlinear reduction. Simulations of the full-order system, the reduced system and the linearized system can be viewed in Figure 3.6. As the figure clearly shows the reduced system is a better representation of the full-order system than the linearized system. The full-order system has an inverse response which also the reduced-order model includes, whilst the linearized has a lack of this dynamics; making it a poorer representation of the original system. The nonlinearity causing the inverse response is lost in the linearized system. The reduced-order model gives in general a better description of the original system's dynamics than the linearized one, making it more suitable for implementation in model-based controllers.

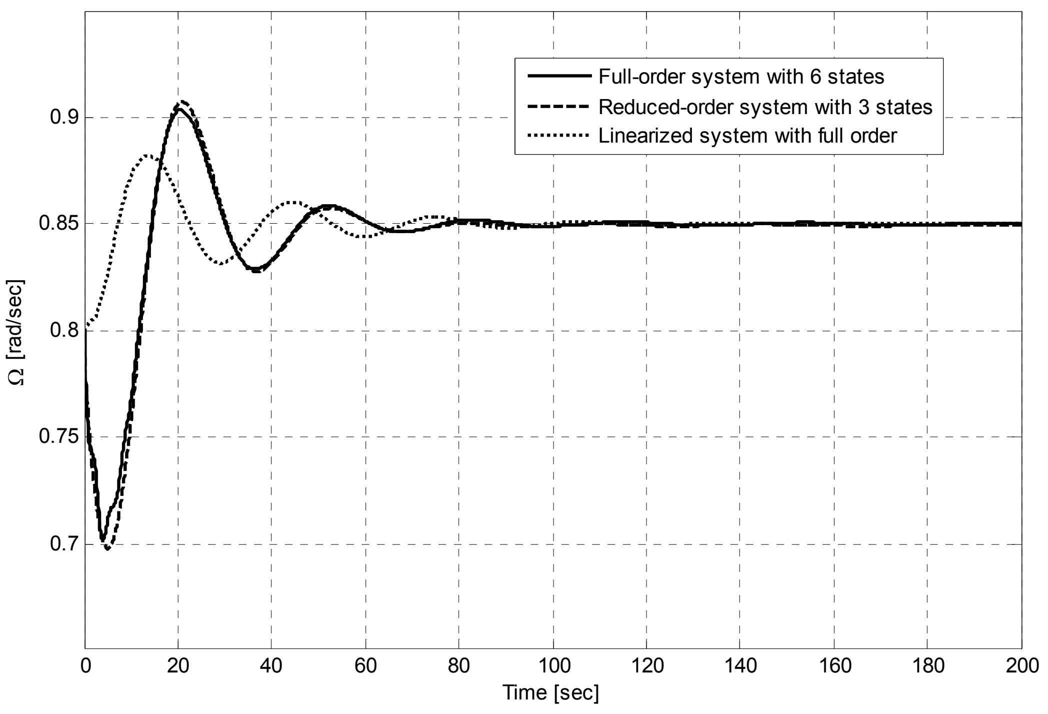

Figure 3.6: Comparison of bit speed for full-order system, reduced-order system, and linearized system

3.3 Full scale model reduction

In this section a full scale model representing a drill string with a length of 1200 m will be reduced. The equations for this model will have the same setup as the equations used in Example 3.1 and Example 3.3, and with an element length of 28 m it will result in 86 differential equations. For simplicity is the nonlinear friction term only added to the bit, the friction from the wall is next to nothing since the well used here is vertical.

As already mentioned will the system be represented by 86 equations, where 43 of them will describe the acceleration of the 43 different elements while the remaining 43 will describe the velocity difference between elements connected to each other. Equation (3.27) shows how the system will be represented.

$$
\begin{array}{l}
\text{String}\begin{cases}
\dot{x}_1 = \dfrac{1}{j_s}\left(d_s k_s (u - x_1) + k_s x_{44} - d_s k_s (x_1 - x_2) - k_s x_{45}\right) \\
\dot{x}_2 = \dfrac{1}{j_s}\left(d_s k_s (x_1 - x_2) + k_s x_{45} - d_s k_s (x_2 - x_3) - k_s x_{46}\right) \\
\vdots \\
\dot{x}_{35} = \dfrac{1}{j_b}\left(d_s k_s (x_{34} - x_{35}) + k_s x_{78} - d_b k_b (x_{35} - x_{36}) - k_b x_{79}\right)
\end{cases} \\
\text{BHA}\begin{cases}
\dot{x}_{36} = \dfrac{1}{j_b}\left(d_b k_b (x_{35} - x_{36}) + k_b x_{79} - d_b k_b (x_{36} - x_{37}) - k_b x_{80}\right) \\
\vdots \\
\dot{x}_{42} = \dfrac{1}{j_b}\left(d_b k_b (x_{41} - x_{42}) + k_b x_{85} - d_b k_b (x_{42} - x_{43}) - k_b x_{86}\right) \\
\dot{x}_{43} = \dfrac{1}{j_b}\left(d_b k_b (x_{42} - x_{43}) + k_b x_{86} - T_b\right)
\end{cases} \\
\Delta\text{speed}\begin{cases}
\dot{x}_{44} = u - x_1 \\
\dot{x}_{45} = x_1 - x_2 \\
\vdots \\
\dot{x}_{86} = x_{42} - x_{43}
\end{cases}
\end{array}
\tag{3.27}
$$

where

k_s	is the stiffness coefficient for the string section
d_s	is the internal damping coefficient for the string section
k_b	is the stiffness coefficient for the bottom hole section
d_b	is the internal damping coefficient for the bottom hole section
T_b	is representing the nonlinear friction component at the bit and is given by

$$
T_b = \mu N r\left(\frac{x_3}{\sqrt{x_3^2 + \Omega_0^2}} + \frac{p\Omega_0 x_3}{x_3^2 + \Omega_0^2}\right) - D x_3\left(\frac{x_3}{\Omega_1} - 1\right)
$$

where

μ	is the friction coeffcient
N	is the force vector
r	is the contact radius vector
Ω_0	is the transistion speed for the string
Ω_1	is the transistion speed for the well
D	is the linear damping vector
p	is the start friction parameter

Values for the different constants are taken from the Simulink model provided by NOV; a 1200 m vertical string was implemented which gave the constants needed. The different values represent the same as they would on a physical string.

The empirical Gramians were computed and balanced with the method already described in the previous section. An important matter that has to be considered is the values used when computing the empirical Gramians, i.e. the simulation data. As already explained is the empirical Gramians found in the area of operation, so it is important that this area is the same as the one the reduced system is supposed to operate in. For this system 10 rad/sec was chosen as steady state of the speed (x_1-x_{43}) and input u. Different values were used for the steps/impulses where the series of 0.1, 0.5 and 1 rad/sec gave the best result. There does not exist a clear method for choosing this data as far as the authors of this thesis know; trial-and-error is usually the approach.

The following singular values of the balanced empirical Gramian was then found

$$diag\left(\Sigma\right)=\begin{bmatrix} 9968.3 \\ 44.9 \\ 19.9 \\ 2.7 \\ 0.88 \\ 0.27 \\ 0.17 \\ 0.021 \\ 0.0054 \\ 0.0020 \\ 0.00054 \\ 0.00003 \\ \vdots \end{bmatrix}$$

As seen from the singular values it is clear that the first states of the balanced system have the largest contribution to the input-output behaviour. A reduced model with between 4-10 states should therefore be sufficient to describe this system.

Figure 3.7 shows a comparison between different reduced models with a change in the input from 10 to 15 rad/sec. A reduced model with only one state is clearly not sufficient to describe the full-order system's dynamics. With 4 states are most of the dynamics captured, but there is a deviation of approximately 4 from the full-order steady state value. This bias gets smaller and smaller as the number of states increases and with 10 states is the bias completely gone. With 4 states there are also an inverse response which neither the full-order system nor the system with 10 states has. One should notice that with 10 states there is a second frequency coming into play, this is not seen with 4 states. One can therefore argue over which would give the best result when implemented in a controller; having only 4 states will reduce the computational complexity dramatically and will therefore be the preferred choice. The bias should not be a problem; integral action is easily implemented in model predictive controllers.

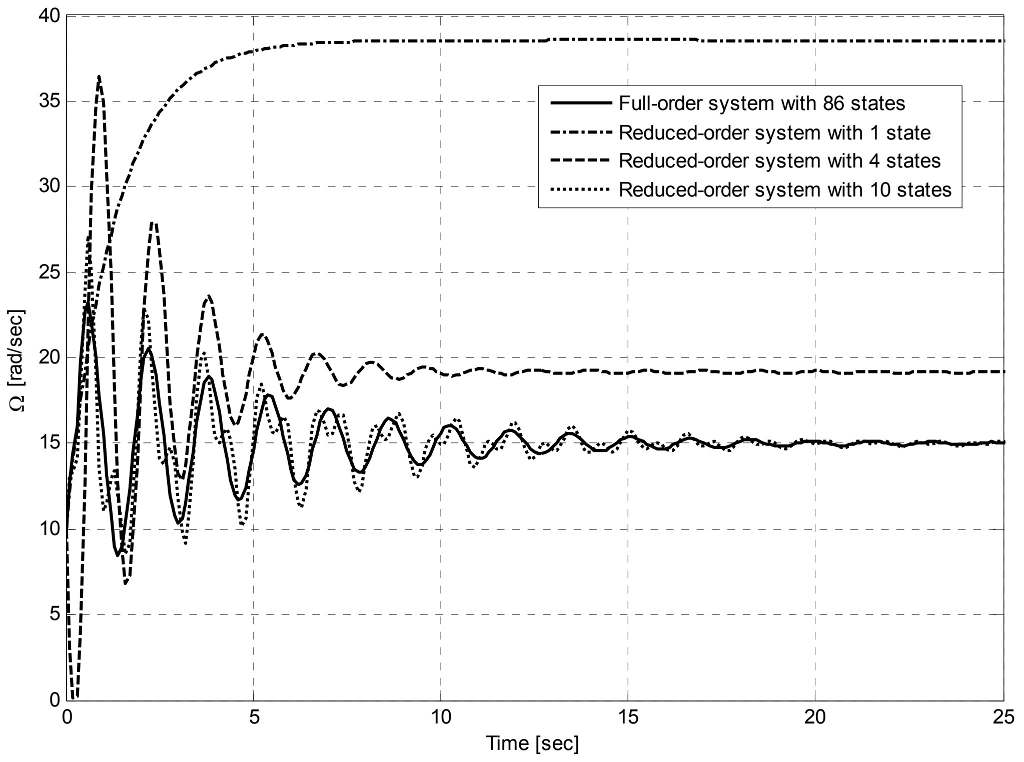

Figure 3.7: Comparison of bit speed between different reduced-order models

As mentioned earlier is the region of operation an important issue when the empirical Gramians are computed. The empirical Gramians were therefore computed with different steps with different magnitudes. Figure 3.8 and 3.9 shows the response of the reduced-order model with 4 states compared to the full-order system.

In Figure 3.8 the input is 5 rad/sec. With this input both the full-order model and the reduced-order model will give stick-slip. The reduced-order model is a good description of the full-order model; it captures the stick-slip phenomenon as it was intended to do. The oscillations are however amplified, but this should not be a problem.

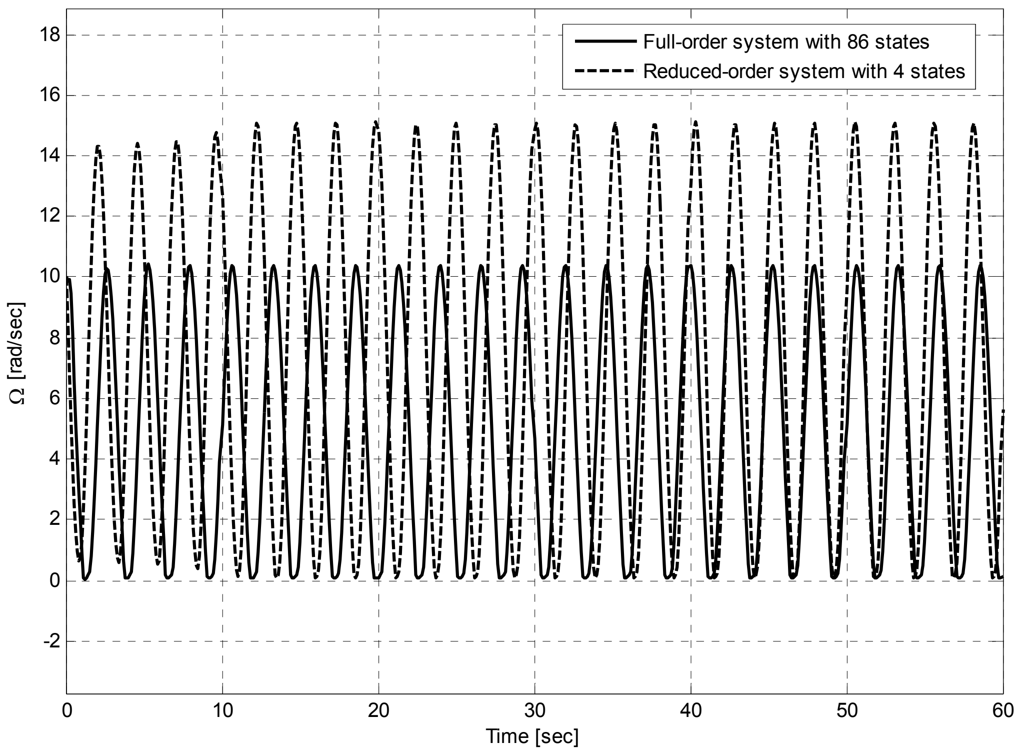

Figure 3.8: Both full-order model and reduced-order model exhibiting the stick-slip phenomenon

In Figure 3.9 there is a large step in the input; it goes from 10 to 25 rad/sec. The reduced-order model gives also in this region of operation a good description of the full-order system.

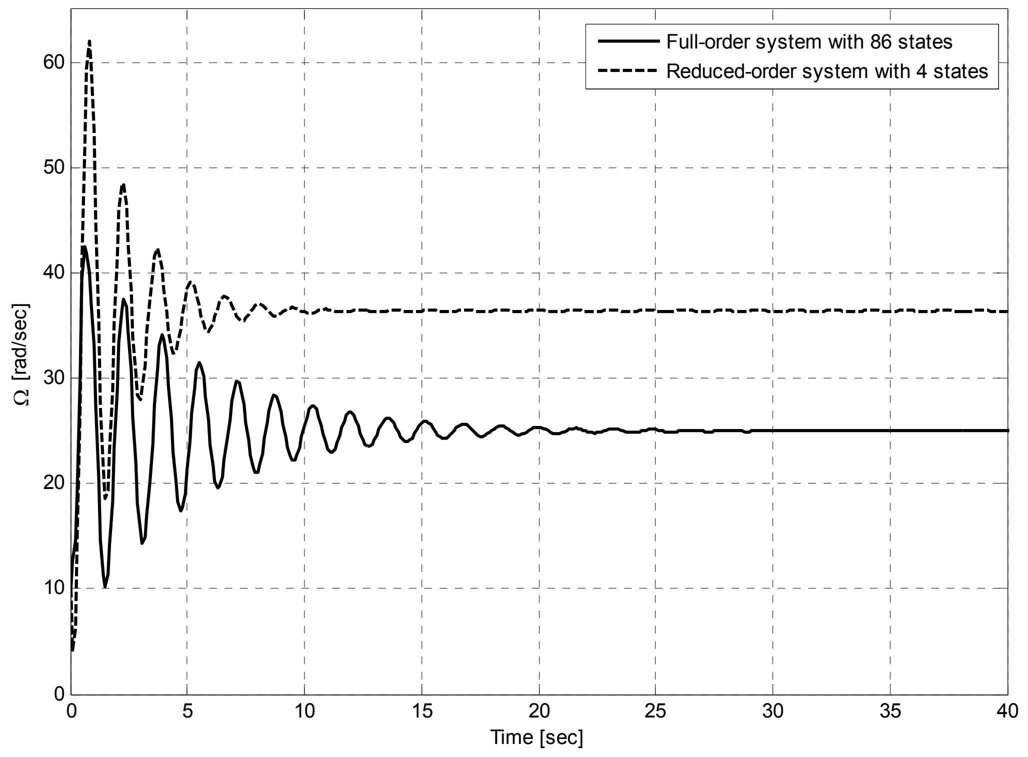

Figure 3.9: Comparison between full-order model and reduced-order model with a large step in the input

It is clear from the figures above that the reduced-order model with only 4 states works well in different regions of operation, and is capable to demonstrate the stick-slip phenomenon. This reduced-order model will be used in Chapter 5 were it will be implemented into an NMPC.

4 MPC

Model predictive control is one of the advanced control methods that has gained foothold in modern industrial control engineering. There are many reasons for this success, and the main reasons are listed as follows by Maciejowski (2002);

1. *It handles multivariable control naturally*
2. *It can take count of actuator limitations*
3. *It allows operation closer to constraints (compared with conventional control), which frequently leads to more profitable operation. Remarkably short pay-back periods have been reported.*
4. *Control update rates are relatively low in these applications, so that there is plenty of time for the necessary on-line computations.*

The authors of this thesis believe that item 2 and 3 in the list above are the most significant due to the results NOV desire from this study of the MPC strategy.

MPC has been in use in industrial control since the 1980s and it was in the first years principally applied to chemical plants. A reason for this is that chemical plants are generally slow with large time delays. The technology did also arise in the process industry and has therefore been somewhat tied to this industry, a tie that currently is being broken down. MPC is being applied to many other segments of control engineering nowadays, such as control of vehicles, aircrafts and in robotics.

Figure 4.1: Model predictive control is gaining a broader usage; such as in aeroplanes

MPC makes use of a dynamic model of the process to predict the process responses due to the applied inputs. There must be defined an objective function, and from the predicted process responses the MPC utilize a numerical minimization algorithm that will calculate the optimal process input that minimizes this objective function. This optimization is performed each time step with the same horizon and with updated measurements/estimates of states and disturbances. At each time step the first computed value of the optimal input trajectory is applied to the process. The optimization algorithm will also take the system's constraints into account, and compute the input trajectory without violating them. All these aspects will be described more closely in this chapter.

4.1 Receding horizon concept

As mentioned earlier, the optimization is performed with the same horizon at each time step. This concept is often called receding horizon, corresponding to the behaviour of the Earth's horizon; as one go one step towards the horizon, it will recede and still be the same distance away. This means that when the optimization is done at current time step *k*, the objective function will be penalized by the algorithm up to time $k + H_p$ where H_p is the prediction horizon. This concept is illustrated in Figure 4.2, and for simplicity is the SISO (single input – single output) case considered here. The set point trajectory is denoted by *s(t)* (in some cases can the set point also be a fixed value), and this is the trajectory that the measured process output *y(t)* in the ideal case should follow. The reference trajectory, represented by *r(t|k)* in the figure, defines the desired trajectory for the process output to follow to return to *s(t)*. The notation *r(t|k)* shows that it depends on the process condition at time *k*, and will be calculated at every time step. The starting point for the reference trajectory will always be the process output *y(t)* at time *k*. It will not always be best to choose the reference trajectory to be a straight line from *y(k)* to *s(k+1)*, but rather let it converge to *s(t)* within a number of time steps. A common approach is to let it converge exponentially with a time constant just about the process' time constant.

The predicted process output $\hat{y}(t \mid k)$ is determined from the model, where the optimal input trajectory $\hat{u}(k+i \mid k)$ will be applied ($i = \{0,1,...,H_p-1\}$). Also here the hat-notation implies that the input trajectory is the best estimate at time *k*, which means that the prediction can be changed in the following time steps.

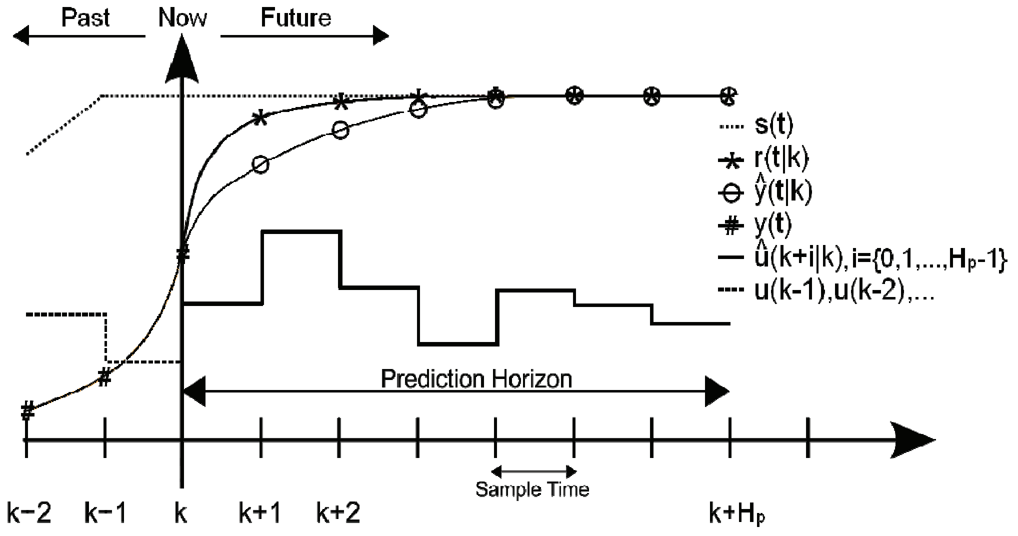

Figure 4.2: The receding horizon concept

4.2 Objective function

When the optimization is executed will the intention be to minimize the objective function. The objective function can be formulated in many different ways, depending on the optimization method. The most common formulations are the linear programming (LP) and quadratic programming (QP) based methods. The advantage of the LP method is that both objective function and constraints are linear, which means that the computational time is low and convergence to a global minimum is always guaranteed. The QP method is still considered to give better performance in general, and therefore will this method be dealt with in this thesis.

In distinction to LP, will the objective function in a QP formulation be quadratic, but the constraints will still be linear. Before the objective function is defined, two other frequently used horizons will be presented. Because there always will be a delay when the input is changed until the process responds, a reasonable approach is to wait some time before the deviation between the reference trajectory and process output is penalized. This time to wait will be represented by the window parameter H_w, and the objective function "window" will be the time horizon from H_w to H_p. It is also common to define a control horizon H_u. In many situation will $H_u = H_p$, but in some cases, e.g. when it is desirable to reduce computational time, it will be natural to set $H_u < H_p$. With these assumptions in mind the objective function can be defined as follows.

$$V(k) = \sum_{i=H_w}^{H_p} \left\| \hat{y}(k+i \mid k) - r(k+i \mid k) \right\|_Q^2 + \sum_{i=0}^{H_u-1} \left\| \Delta \hat{u}(k+i \mid k) \right\|_R^2 \qquad (4.1)$$

where Q and R are weight matrices. The norms used in equation (4.1) are short notation for the quadratic terms, and are defined by $\|x\|_Y^2 = x^T Y x$. The weight matrices are tuning parameters which can be adjusted to give the MPC satisfactory performance; this will be treated in more detail in section 4.9.

4.3 Constraints

Closely related to MPC is the Linear Quadratic Regulator (LQR). This controller also minimizes a quadratic objective function, and penalizes the input and state deviations for each sample step. In distinction to MPC, the LQR will not handle the system's constraints when calculating the next system input; so one could simply say that MPC is a constrained LQR.

In practice a system's constraints can be divided in two parts, equipment constraints and constraints that the process is wanted to operate within to for example ensure a specific quality of the product. Examples of equipment constraints could be

- actuator constraints like fully open/closed valve, a motors min/max speed, an actuator's slew rate etc.
- state constraints like empty/full tank, a pipe's maximum allowed pressure, a drill string's max torque etc.

and typical operating constraints could be

- to keep the controlled variables within certain limits

- to keep the mixture of two or more liquids within a certain composition
- to let the controlled variables follow a set point trajectory

A mathematical representation of the inequality constraints could be

$$
\begin{aligned}
&y_{\min} \le y(k) \le y_{\max} \\
&\Delta u_{\min} \le \Delta u(k) \le \Delta u_{\max} \\
&u_{\min} \le u(k) \le u_{\max}
\end{aligned}
\tag{4.2}
$$

where

y_{min}/ y_{max}	are vectors containing the controlled variables' min/max values
$y(k)$	is a vector containing the state values at time step *k*
$\Delta u_{min}/\Delta u_{max}$	are vectors containing the actuators min/max slew rates
$\Delta u(k)$	is a vector containing the actuator's applied slew rates at time step *k*
u_{min}/u_{max}	are vectors containing the actuators min/max values
$u(k)$	is a vector containing the actuators input signal at time step *k*

It should not be hard to realize that the equipment constraints mentioned above are not possible to violate under any conditions; because of this fact such constraints are called hard constraints. The operating constraints are possible, but not desirable to violate. In some cases, for example if health and safety are in danger, it would be necessary to violate these constraints and they are therefore called soft constraints. If the process operates close to the constraints, a large disturbance can move the controlled variables outside the feasible area. Also if the internal model has become incorrect due to changes in the process dynamics the MPC will give wrong predictions and can possibly result in constraint violation. It will be important that the MPC handles these infeasible situations, because it can give serious consequences if the algorithm just stops. There are many ways to deal with such scenarios; one could just use the previous feasible input or use a feedback controller until a feasible solution is achieved. A more intelligent approach would be to implement soft constraints in the algorithm.

When softening the constraints the algorithm allows the constraints to be violated, but only if it is necessary and the violations should be held to a minimum. One way of softening the constraints are by adding new variables to the objective function, called slack variables. If the constraints are violated the slack variables will be non-zero, but they will be very heavily penalized by the objective function so the optimizer has a strong incentive of keeping the slack variables zero if possible. As some of the constraints are hard constraints, those cannot be violated and the corresponding slack variables are always zero. One method for softening the constraints is described as follows by Maciejowski (2002).

First consider a standard QP problem of the form

$$\min_{\theta} \frac{1}{2}\theta^T \Phi \theta + \phi^T \theta$$

subject to

$$\Omega\theta \le \omega$$

This will not handle a situation where the constraints are violated. When slack variables ε are introduced, the QP formulation can be modified to

$$\min_{\theta,\varepsilon} \frac{1}{2}\theta^T \Phi \theta + \phi^T \theta + \rho \|\varepsilon\|^2$$

subject to

$$\begin{aligned} &\Omega\theta \le \omega + \varepsilon \\ &\varepsilon \ge 0 \end{aligned}$$

where

ε	is a non-negative vector with the same dimention as ω (elements corresponding with hard constraints are always zero)
ρ	is a non-negative scalar

4.4 Prediction

One of the key-components of the MPC is the prediction of the future values of the controlled variables. This is done using measurements or estimates of the current state and the assumed future inputs.

Assume that all states are measured and that nothing is known about any disturbances or measurement noise. Then the prediction of the states is found by iterating the model given by (4.3)

$$\begin{aligned} x_{k+1} &= Ax_k + Bu_k \\ y_k &= Cx_k \end{aligned} \tag{4.3}$$

where

x	is the state vector
u	is the input vector
y	is the output vector

This will lead to the following:

$$\begin{bmatrix} \hat{x}(k+1|k) \\ \vdots \\ \hat{x}(k+H_u|k) \\ \hat{x}(k+H_u+1|k) \\ \vdots \\ \hat{x}(k+H_p|k) \end{bmatrix} = \begin{bmatrix} A \\ \vdots \\ A^{H_u} \\ A^{H_u+1} \\ \vdots \\ A^{H_p} \end{bmatrix} x(k) + \begin{bmatrix} B \\ \vdots \\ \sum_{i=0}^{H_u-1} A^i B \\ \sum_{i=0}^{H_u} A^i B \\ \vdots \\ \sum_{i=0}^{H_p-1} A^i B \end{bmatrix} u(k-1) +$$

$$\begin{bmatrix} B & \cdots & 0 \\ AB+B & \cdots & 0 \\ \vdots & \ddots & \vdots \\ \sum_{i=0}^{H_u-1} A^i B & \cdots & B \\ \sum_{i=0}^{H_u} A^i B & \cdots & AB+B \\ \vdots & \vdots & \vdots \\ \sum_{i=0}^{H_p-1} A^i B & \cdots & \sum_{i=0}^{H_p-H_u} A^i B \end{bmatrix} \begin{bmatrix} \Delta\hat{u}(k|k) \\ \vdots \\ \Delta\hat{u}(k+H_u-1|k) \end{bmatrix} \tag{4.4}$$

It is clear that the first two terms are based on the past, while the last term is based on the future. There are seldom that no disturbances act on a system. An output disturbance can easily be added to the predictions, even without a measurement of it, by just taking the difference between the measured and the estimated output. If not all the states are measured, which typically is the case, an observer or another estimation scheme have to estimate the full state vector. This is usually possible to do, but there can be different approaches. Different methods for estimating the full state vector, including methods for nonlinear systems, are given further investigation in section 4.7.

4.5 Complete formulation

In the previous sections the different aspects regarding the model predictive controller has been shown, such as the objective function, constraints, predictions etc. In this section will they be put together to see how the complete MPC problem can be formulated.

Consider the linear time invariant state space-space model as given by (4.3). The MPC problem is then formulated as the following

$$\min \sum_{i=H_w}^{H_p} \left\| \hat{y}(k+i \mid k) - r(k+i \mid k) \right\|_Q^2 + \sum_{i=0}^{H_u-1} \left\| \Delta\hat{u}(k+i \mid k) \right\|_R^2$$

subject to

$$x_{k+1} = Ax_k + Bu_k$$
$$y_k = Cx_k$$
$$y_{\min} \leq y(k) \leq y_{\max}$$
$$\Delta u_{\min} \leq \Delta u(k) \leq \Delta u_{\max}$$
$$u_{\min} \leq u(k) \leq u_{\max}$$

4.6 Control hierarchy

If one should consider the MPC in a control hierarchy, it has traditionally been placed in a layer above the regulatory control layer (where the single loop PI(D) controllers are found), and the MPC calculates the set points for this lower layer. In such setup it will be the regulatory level that stabilizes the process and gives the control inputs applied to the actuators (valve servos, drives, etc.). The reader is referred to the article by Skogestad (2004) for more information about control structure design. The main reason why it has been done this way is because after the introduction of MPC, there did not exist any proofs showing that it would stabilize the process. The operating companies did not rely 100% on the MPC, and by doing it this way the process was still stable (although not optimal), even if the MPC for some reason was not operational. There are in the latter days been established a satisfactory stability theory for the MPC, but there is quite rarely seen applications where the MPC outputs are applied directly to the actuators. The reason for this practice could maybe be just due to traditional aspects; Maciejowski (2002) supposes a future trend where the regulatory control level is removed and the MPC output are applied directly to the actuators. Figure 4.3 shows a traditional setup for a control hierarchy, mostly used in large processes.

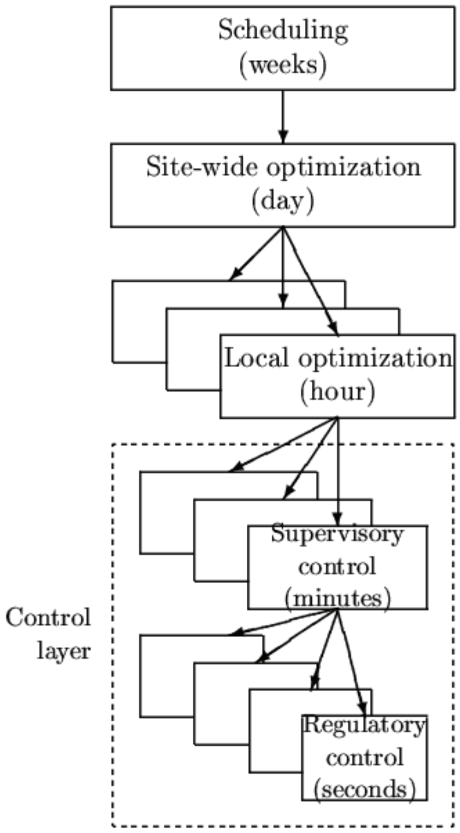

Figure 4.3: Control hierarchy as proposed by Skogestad (2004)

In this thesis will the MPC output be used directly to control the top drive speed of the drill string, which is rather unconventional, but it is of large interest to see how an MPC can deal with the stick-slip phenomenon. The output of the MPC will go directly to the top drive and the controlled variable will be the bit speed. Figure 4.4 shows a traditional drill string with input u and output y. In Chapter 5 will the complete problem formulation for this case be derived.

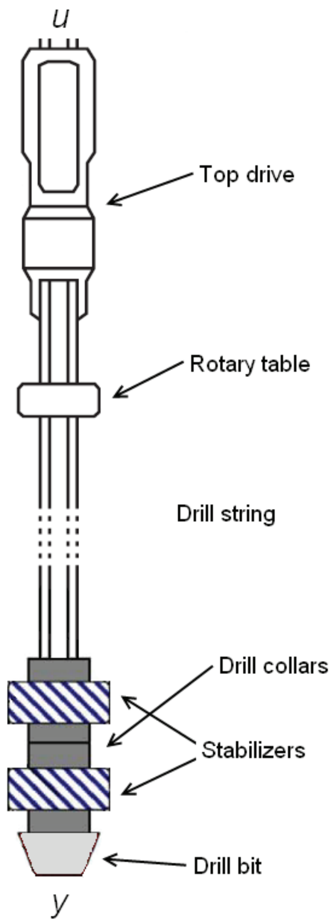

Figure 4.4: Drill string with input *u* and output *y*

4.7 Estimation

In MPC is the predicted values of the controlled variables computed using the current states and assumed future inputs. It is rather unrealistic that a measurement for every state is available, so the unmeasured states have to be estimated. There are different strategies for estimating the states; which method that gives the best results depends on the system. If the system is linear there exists rather simple strategies; a Kalman filter usually gives satisfactory results. If the system is nonlinear the story will be somewhat different. Two estimators based on the Kalman filter that have shown good results are the Extended Kalman Filter (EKF) and the Unscented Kalman Filter (UKF), where the UKF seems to be the best to handle systems with high nonlinearity. Since the interest of this thesis is a highly nonlinear system only the UKF will be given further attention.

Unscented Kalman Filter

The first thing that catches ones eye when the UKF is considered is the strange name; an unscented Kalman filter? This needs some further comments. Scented are used about things with a smell or an odour, and have in principle nothing to do with state estimation problems. UKF was developed by the Robotics Research Group at University of Oxford and it was also members of this group that gave the method its name. The method was first called the New Filter, but due to the ambiguous nature of the name it was decided that a new and better name was needed. After a democratic vote amongst the group the name was chosen to be Unscented Kalman Filter, exactly why that name was chosen is somewhat unclear. There has been speculated that the name was chosen to imply that EKF stinks, but this is just speculations.

This funny anecdote is taken from the PowerPoint presentation Pekka Jänis used at a Postgraduate Seminar on Signal Processing in 2007 at the Helsinki University of Technology. The question about the origin of the name still remains a riddle for the authors of this thesis.

As apposed to the EKF which uses Taylor series linearization, is statistical linearization used in the UKF. A nonlinear function of a random variable is linearized through a linear regression between *n* points drawn from the prior distribution of the random variable. The state distribution is specified with a minimal set of chosen sample points which captures the true mean and covariance for the variable. When there are at least *2n+1* sampling points are the approximations accurate to the 3rd order (Taylor series expansion) for Gaussian inputs and at least to the 2nd order for non-Gaussian inputs. This is an advantage over the EKF which only achieves 1st order accuracy. The algorithm presented here is based on the one in the article written by Wan & van der Merwe (2000).

Algorithm 4.1: Unscented Kalman Filter

Consider the nonlinear system given by:

$$\begin{aligned} x_{k+1} &= f\left(x_k, u_k\right) + w_k \\ y_k &= h\left(x_k\right) + v_k \end{aligned} \tag{4.5}$$

The algorithm is initialized with:

$$\begin{aligned} \hat{x}_0 &= E\left(x_0\right) \\ \mathrm{P}_0 &= E\left[\left(x_0 - \hat{x}_0\right)\left(x_0 - \hat{x}_0\right)^T\right] \\ \hat{x}_0^a &= E\left[x^a\right] = \left[\hat{x}_0^T \quad 0 \quad 0\right]^T \\ \mathrm{P}_0^a &= E\left[\left(x_0^a - \hat{x}_0^a\right)\left(x_0^a - \hat{x}_0^a\right)^T\right] = \begin{bmatrix} \mathrm{P}_0 & 0 & 0 \\ 0 & \mathrm{P}_w & 0 \\ 0 & 0 & \mathrm{P}_v \end{bmatrix} \end{aligned} \tag{4.6}$$

where

$$\begin{aligned} x^a &= \left[x^T \quad w^T \quad v^T\right]^T \\ \mathrm{P}_w &= \text{ Process noise covariance} \\ \mathrm{P}_v &= \text{ Measurement noise covariance} \end{aligned} \tag{4.7}$$

The Sigma points are calculated:

$$\mathrm{X}_{k-1}^a = \left[\hat{x}_{k-1}^a \quad \hat{x}_{k-1}^a \pm \sqrt{\left(L+\lambda\right)\mathrm{P}_{k-1}^a}\right], k \in \{1, \ldots, \infty\} \tag{4.8}$$

where

$$\begin{aligned} \mathrm{X}^a &= \left[\left(\mathrm{X}^x\right)^T \quad \left(\mathrm{X}^v\right)^T \quad \left(\mathrm{X}^n\right)^T\right]^T \\ L &= \text{ dimension of augmented state} \\ \lambda &= \text{ composite scaling parameter} \end{aligned} \tag{4.9}$$

Time update:

$$
\begin{aligned}
&\mathrm{X}_{k|k-1}^{x} = F\left[\mathrm{X}_{k-1}^{x}, \mathrm{X}_{k-1}^{v}\right] \\
&\hat{x}_k^- = \sum_{i=0}^{2L} W_i^{(m)} \mathrm{X}_{i,k|k-1}^{x} \\
&\mathrm{P}_k^- = \sum_{i=0}^{2L} W_i^{(c)} \left[\mathrm{X}_{i,k|k-1}^{x} - \hat{x}_k^-\right]\left[\mathrm{X}_{i,k|k-1}^{x} - \hat{x}_k^-\right]^T \\
&\mathrm{Y}_{k|k-1} = \mathrm{H}\left[\mathrm{X}_{k|k-1}^{x}, \mathrm{X}_{k|k-1}^{n}\right] \\
&\hat{y}_k^- = \sum_{i=0}^{2L} W_i^{(m)} \mathrm{Y}_{i,k|k-1}
\end{aligned}
\tag{4.10}
$$

where W_i are weights that are calculated by:

$$
\begin{aligned}
&W_0^{(m)} = \frac{\lambda}{L+\lambda} \\
&W_0^{(c)} = \frac{\lambda}{L+\lambda} + \left(1 - \alpha^2 + \beta\right) \\
&W_i^{(m)} = W_i^{(c)} = \frac{1}{2\left(L+\lambda\right)}, i = 1, \ldots, 2L
\end{aligned}
\tag{4.11}
$$

where

$$
\begin{aligned}
&\alpha = \text{ the spread of the sigma points around } \overline{x} \\
&\beta = \text{ prior knowledge of the distrubution of } x
\end{aligned}
\tag{4.12}
$$

The measurements are updated by:

$$
\begin{aligned}
&\mathrm{P}_{\tilde{y}_k \tilde{y}_k} = \sum_{i=0}^{2L} W_i^{(c)} \left[\mathrm{Y}_{i,k|k-1} - \hat{y}_k^-\right]\left[\mathrm{Y}_{i,k|k-1} - \hat{y}_k^-\right]^T \\
&\mathrm{P}_{x_k y_k} = \sum_{i=0}^{2L} W_i^{(c)} \left[\mathrm{X}_{i,k|k-1} - \hat{x}_k^-\right]\left[\mathrm{Y}_{i,k|k-1} - \hat{y}_k^-\right]^T \\
&\kappa = \mathrm{P}_{x_k y_k} \mathrm{P}_{\tilde{y}_k \tilde{y}_k}^{-1} \\
&\hat{x}_k = \hat{x}_k^- + \kappa\left(y_k - \hat{y}_k^-\right) \\
&\mathrm{P}_k = \mathrm{P}_k^- - \kappa \mathrm{P}_{\tilde{y}_k \tilde{y}_k} \kappa^T
\end{aligned}
\tag{4.13}
$$

where

$$
\begin{aligned}
&x^a = \left[x^T \quad v^T \quad n^T\right]^T \\
&\mathrm{X}^a = \left[\left(\mathrm{X}^x\right)^T \quad \left(\mathrm{X}^v\right)^T \quad \left(\mathrm{X}^n\right)^T\right]^T
\end{aligned}
\tag{4.14}
$$

There can be a challenge to estimate the states of a reduced model. When using reduced nonlinear models the states will loose their physical meaning. The only variables that make sense are the system's input and output; the rest has just some value that cannot be related to anything physical. This will of course be a condition that makes state estimation harder since it will be difficult to say whether the observer gives good state estimates or not.

Integral action
Another motivation for utilizing estimation in MPC is the implementation of integral action. The main reason for steady state error is due to model errors. In practice it will be impossible to get a model perfectly matching the real process, and the process dynamics will also change by time. To achieve offset-free regulation it will therefore be necessary to apply such integral action. It is shown by Pannocchia & Rawlings (2003) that this can be done by adding integrating disturbances to the process model. They prove in their article that a number of integrating disturbances equal to the number of measured variables are shown to be sufficient to guarantee zero offset in the controlled variables.

4.8 Stability

A challenge with MPC has been stability and in the latter days has there been established a great amount of theory on this subject. Predictive control with the receding horizon idea is a feedback control policy. Feedback can be dangerous in the sense that the resulting closed loop could become unstable, but feedback is needed to effectively reduce the effects of unexpected and immeasurable disturbances and to reduce the effects of the uncertainties of the system. In this section will different strategies that guarantee nominal stability be investigated, with the assumption that the model is perfect.

Terminal constraints
By adding a terminal constraint will the state be forced to take a particular value at the end of the prediction horizon, for example drive the state to the origin. This can easily be proved using a Lyapunov function (Keerthi & Gilbert, 1988). Some assumptions are made, such that the optimization problem has a solution at every step and that the global optimum can be found at every step. With constraints can this be difficult, and sometimes even infeasible.

Terminal cost
A strategy that often is used together with the terminal constraint is a terminal cost. The terminal cost is added to the objective function and will penalize the non-zero states at the end of the prediction. This will make the optimization drive the states to zero.

Infinite horizons
Making the horizons infinite will guarantee stability for the closed loop because the optimal trajectory will not change. When the process is stable is the control problem with infinite horizons solved by reparameterization of the control problem to a finite number of parameters. Then the optimization can be performed over a finite-dimensional space. When the process is unstable the problem is not so trivial. The process is divided into two parts, one stable and one unstable. On the stable part is still an infinite horizon used. A terminal equality constraint is used on the unstable modes; this makes the unstable modes go to zero at the end of the control horizon. This extra constraint will be imposed to the optimization problem and there can be problems with infeasibility.

4.9 Tuning

As for conventional controllers is tuning an important issue to get an optimal performance of the MPC. There is no point in spending time and effort on deriving an MPC if it is put in use with somewhat random parameters. Remember that one of the major reasons for choosing an MPC is get optimal control close to constraints. There are several different parameters that can be chosen in an MPC, and in this section will the most important be explained; how they can be found and how they influence the system and the performance.

Weight matrices *Q* and *R*

The weight matrices Q and R are important parameters for the closed loop performance of the controlled system. Tuning Q and R are similar to the LQR-case, and they are often chosen as diagonal matrices. If Q is increased relative to R there will be a more aggressive closed loop behaviour as the MPC tries to reduce the tracking error. On the other hand if R is increased relative to Q the control activity will be reduced as the MPC tries to reduce the change in u. The task of finding the correct values for Q and R and the ratio between them is another story; there is no straightforward method for choosing them. The desired closed loop response and knowledge about the process can be a basis for choosing the parameters, but in the end is trial-and-error by closed loop simulation the way to ensure optimal performance.

Prediction horizon H_p

Affecting both closed loop performance and computational load makes the prediction horizon one of the most important parameter in MPC tuning. Large H_p will give a more complex optimization problem which will clearly affect the computational load as the number of free variables increase. One would think that a larger H_p would increase the optimality of the MPC as this will make the closed loop behaviour closer to the infinite horizon controller, but this is not always the case (Di Palma & Magni, 2007). Another issue for not choosing a too long prediction horizon is mismatch between process and model; effects from model uncertainties tends to enhance if the prediction horizon is too long.

Control horizon H_u

The control horizon will as the prediction horizon affect both closed loop performance and computational load. The prediction horizon and control horizon are closely related as seen in the objective function (4.1); choosing one of them will inflict the other. An important issue when choosing both H_p and H_u are the process time delays. It is important that H_p is larger than the time delay and that H_p+ H_u are larger than the time delay to force the controller to consider the full effect of each move. Consider the following example:

Example 4.1: Time delays

$H_p = 10$
$H_u = 6$
Delay = 5

Six different control moves has to be calculated, but the 6th control move will not have any impact within the prediction horizon since the delay is 5. This makes the 6th control move indeterminate (this situation is illustrated in Figure 4.5); it just increases the computational load. This could easily be avoided if H_p had been larger or H_u smaller.

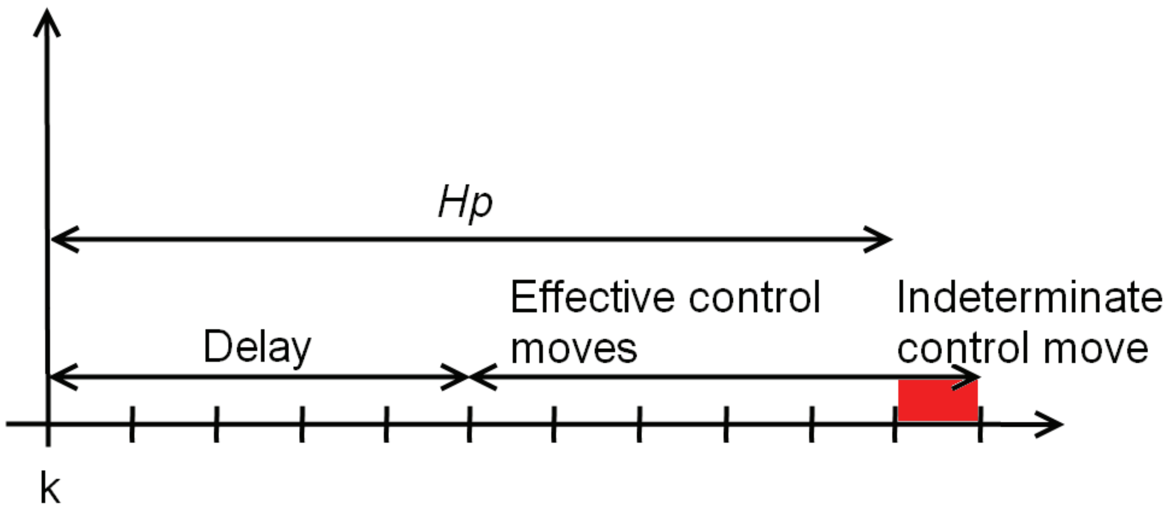

Figure 4.5: The effect of time delays

As this simple example shows time delays are an important issue when choosing the horizons. There are some rules of thumb for choosing the horizons, but in the end they have to be chosen individually for the specific process.

The "window" parameter H_w
The choice of H_w will influence the closed loop performance; if it is chosen too low will this lead to a deviation between the reference trajectory and process output which again will result in a reduction of the controller performance. As already mentioned there can be different time delays, so there is no point in penalizing the deviation between the reference trajectory and process output before the input has any chance on affecting the output.

Reference trajectory *r(k+i)*
Closed loop aggresivity will also be influenced by the reference trajectory; a slower reference trajectory will lead to a less aggressive closed loop behaviour. The reference trajectory has to be chosen in consideration with the desired closed loop response.

4.10 NMPC

In all physical systems there exist nonlinearities to some extent. Only in simple mechanical systems and simple electrical circuits consisting of passive components, the system's behaviour will be close to linear over a substantial region of operation. In such applications a linear representation of the system can be used, even if some of the operating conditions are changed. In reality will systems to be controlled usually be complex systems with significant nonlinear behaviour, whether it is a large chemical process, transportation systems, social models, or whatever it should be. When dealing with these kinds of strongly nonlinear processes, an MPC based on a linear model will give large limitations on the controller's performance.

An intuitive solution to the issue mentioned above is to use a nonlinear internal model in the MPC; NMPC. Of course it involves some challenges to use nonlinear models, unless there should not be any reasons to use linear models. In this chapter some of these challenges are discussed, and methods describing how one can overcome these problems are considered.

Challenges
When an MPC is running in real-time, one of the main problems using a nonlinear model is that it will be hard, or sometimes impossible, to say something about the computational time each optimization step will take to complete and even if the optimization ever will terminate. It will be a firm requirement that the MPC has its next output signal ready for every sample step, especially when the output is applied directly to the actuators (see section 4.6 about *Control hierarchy*); this will actually be the case when the MPC output is used directly as control signal to the top drive of a drill string.

Another drawback when convexity of the optimization problem is lost is that the problem no longer has one optimal solution, but can have many local optimums in addition to the global optimum which is preferred to find. If the optimization algorithm converges to a solution, one cannot say if the solution is the global optimum or not; the algorithm can as well converge to a local solution.

Similar to the linear case, NMPC will need the full state vector for every sampling. Usually it will not be the case that all states are measured directly, so it will be necessary to use a state estimator. For the nonlinear case usually the Extended Kalman Filter (EKF) has been used as state estimator, but more recently the Unscented Kalman Filter (UKF) has gained popularity (find more about this in section 4.7 about *Estimation*).

SQP
There are developed several methods for solving optimization problems using nonlinear internal models, and one of the most effective method so far is an algorithm generating steps by solving quadratic sub-problems (Nocedal & Wright, 2006). This method is called Sequential Quadratic Programming (SQP) and this method will be used in this thesis. In all iterations of the SQP method there will be made a quadratic approximation to the objective function and linear approximations to the nonlinear constraints, namely the method will make an approximation to a standard QP problem. Many variations of the method are available, but the basic idea of SQP is presented as follows by Maciejowski (2002):

Suppose that a general constrained optimization problem of the form showed in (4.15) is to be solved.

$$\min_{x} \left\{ V(x_k) : H_i(x_k) = 0, \Omega_j(x_k) \leq 0 \right\} \tag{4.15}$$

where

$\left\{ H_i(\cdot) = 0 \right\}$ are sets of nonlinear equality constraints

$\left\{ \Omega_j(\cdot) \leq 0 \right\}$ are sets of nonlinear inequality constraints

The SQP algorithm makes a quadratic approximation *$q_k(d)$* to the objective function *$V(x_k)$*:

$$q_k(d) = \frac{1}{2} d^T \nabla_{xx}^2 L(x_k, \lambda_k) d + \nabla V(x_k)^T d$$

where

$$L(x,\lambda) = V(x) + \sum_i \lambda_i H_i(x) + \sum_i \lambda_i \Omega_i(x) \qquad \text{is the Lagrangian}$$

The next iterate x_{k+1} is given by

$$x_{k+1} = x_k + d_k$$

where d_k is found by solving the QP problem which results from minimizing $q_k(d)$ subject to local linear approximations of the constraints, given by

$$\min_d \left\{ q_k(d) : H_i(x_k) + \nabla H_i(x_k)^T d = 0, \Omega_j(x_k) + \nabla \Omega_i(x_k)^T d \le 0 \right\}$$

Since this is just a sketch of how an SQP algorithm works, nothing about how to find the Lagrangian multipliers λ_k or the Hessian matrix $\nabla_{xx}^2 L(x_k, \lambda_k)$ are described. This will in fact vary from algorithm to algorithm, and in this thesis is MATLAB's SQP solver *fmincon* used in the NMPC. *fmincon* uses the well known BFGS method, which is considered as the most popular quasi-Newton method (Nocedal & Wright, 2006) for solving nonlinear optimization problems. The method is named from its discoverers Broyden, Fletcher, Goldfarb and Shannon.

5 NMPC for drill string

The theory presented in the previous chapter will now be used to control a rotating drill string.

Consider the nonlinear system given by

$$\begin{aligned}\dot{x} &= f\left(x(t), u(t)\right) \\ y &= h\left(x(t)\right)\end{aligned} \tag{5.1}$$

where

x	is the state vector
u	is the input (top drive speed, ω_{td})
y	is the measurement (bit speed, ω_{bit})
f, h	are nonlinear functions

The nonlinear system here is in reduced form, consisting of only 4 states, whereas the original system consists of 86 states. The system under investigation is the same as in Chapter 3, a 1200 m drill string with a highly nonlinear component representing the friction at the bit.

There are different sets of constraints acting on this system, where the torque alterations are believed to be the most critical one. The equipment constraints acting on the system is maximum top drive speed and top drive slew rate (which will influence the torque alterations). The constraints defined for this system are all linear and will be

$$\begin{aligned}&0 \le \omega_{td}(k) \le \omega_{td,\max} \\ &\left|\omega_{td}(k) - \omega_{td}(k-1)\right| \le \Delta\omega_{td,\max}\end{aligned}$$

There were not found data about slew rate on a real top drive, but $\Delta\omega_{td,\max}$ is here supposed to be 1 rad/sec per second.

The objective of the controller is to keep the bit speed constant at set point and to keep the rate of penetration as high as possible. A suitable objective function is then given by

$$V(k) = Q\sum_{i=H_w}^{H_p}\left(\omega_{bit}(k+i) - \omega_r(k+i)\right)^2 + R\sum_{i=0}^{H_u-1}\left(\omega_{td}(k+i) - \omega_{td}(k+i-1)\right)^2$$

The NMPC was implemented in MATLAB using *fmincon* as the SQP solver. There were some difficulties implementing the last term in the objective function, Δu, so this was not included. This would anyway have little influence on the value of the objective function since most weight would be on keeping a constant set point, and because the constraints on Δu will be active most of the time when stick-slip occurs. Information about the complete codes for the controller can be found in Appendix B.

6 Simulations of the NMPC and comparisons

The NMPC developed in the previous chapter is simulated to see how the controller copes with the stick-slip phenomenon when a reduced model of the system is used. The NMPC will also be compared with a linear MPC and NOV's patented stick-slip prevention system, SoftSpeed. First are the different controllers tuned, thereafter simulated and lastly a comparison is made between the three.

6.1 NMPC

The NMPC implemented in MATLAB is simulated to see how it performs on a rotating drill string. First the controller is tuned, and then the controller is simulated when set point changes are applied to verify its performance.

Tuning

There are several parameters that influence the response of an NMPC as presented in Chapter 4. In this section different parameters will be under investigation, the motivation being to find the optimal set, and keeping the computational complexity as low as possible.

Since only the set point changes were implemented (R=0) in the objective function is the value of the weight matrix Q irrelevant. It just has to be non-zero.

The most important parameters in this perspective are the horizons. Long horizons will increase the computational complexity, but if the horizons are too low will the NMPC not be able to cure the stick-slip oscillations. As already discussed in Chapter 4, there is no fixed rule for finding the horizons, but a rule of thumb is to choose the prediction horizon 4-10 times larger than the systems largest time constant.

The following figures show plots with different horizons. The prediction horizon and control horizon are equal within the different simulations, but the length of them changes from simulation to simulation, as does also the period between each time a new input value is computed, h_c. In all the simulations there are at the beginning a constant input of 7 rad/sec, which gives stick-slip with a period of about 1.5 seconds. The NMPC is turned on after approximately 10 seconds.

In Figure 6.1 are the prediction and control horizons 4.5 seconds and h_c=1.5 seconds, which enables the NMPC to cure stick-slip without any problems. The input is stable after 7-8 seconds, while the output is stable after 11-12 seconds.

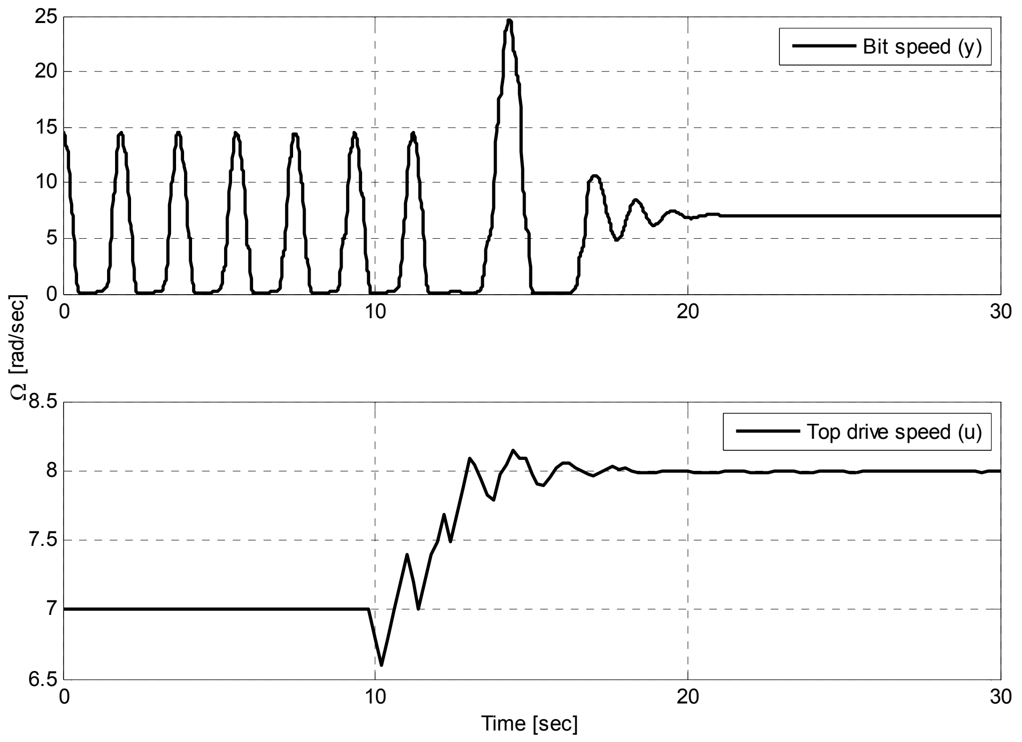

Figure 6.1: Prediction and control horizon 4.5 seconds, h_c = 1.5 seconds

Figure 6.2 shows the simulation where the prediction and control horizons are 6 seconds and h_c=2 seconds. The output settles faster and the maximum value of the output is lower than in the previous plot. Also the input settles faster, but is on the other hand much more erratic than the previous plot.

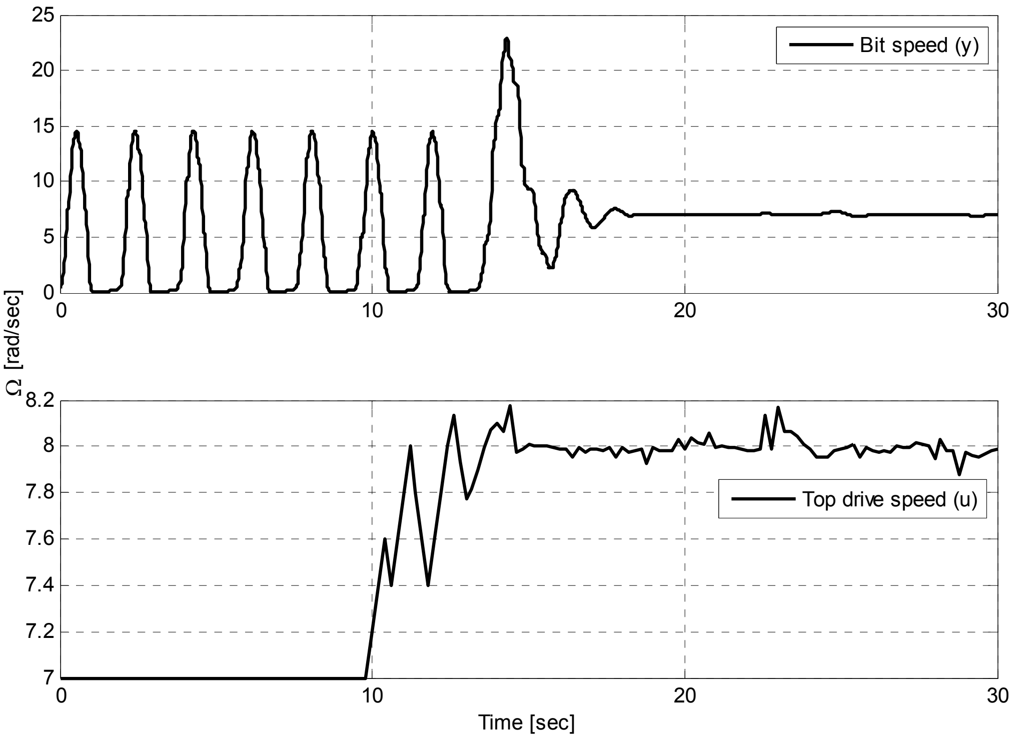

Figure 6.2: Prediction and control horizon 6 seconds, h_c = 2 seconds

In Figure 6.3 are the prediction and control horizons still 6 seconds, but the input now changes six times (h_c=1) over the prediction horizon instead of three which has been the case for the previous simulations (all the slices have the same length). The input is somewhat more aggressive and peaks up to around 9.2 rad/sec. This gives a lower maximum value of the output, but it is slower than the others and the output is somewhat fluctuating.

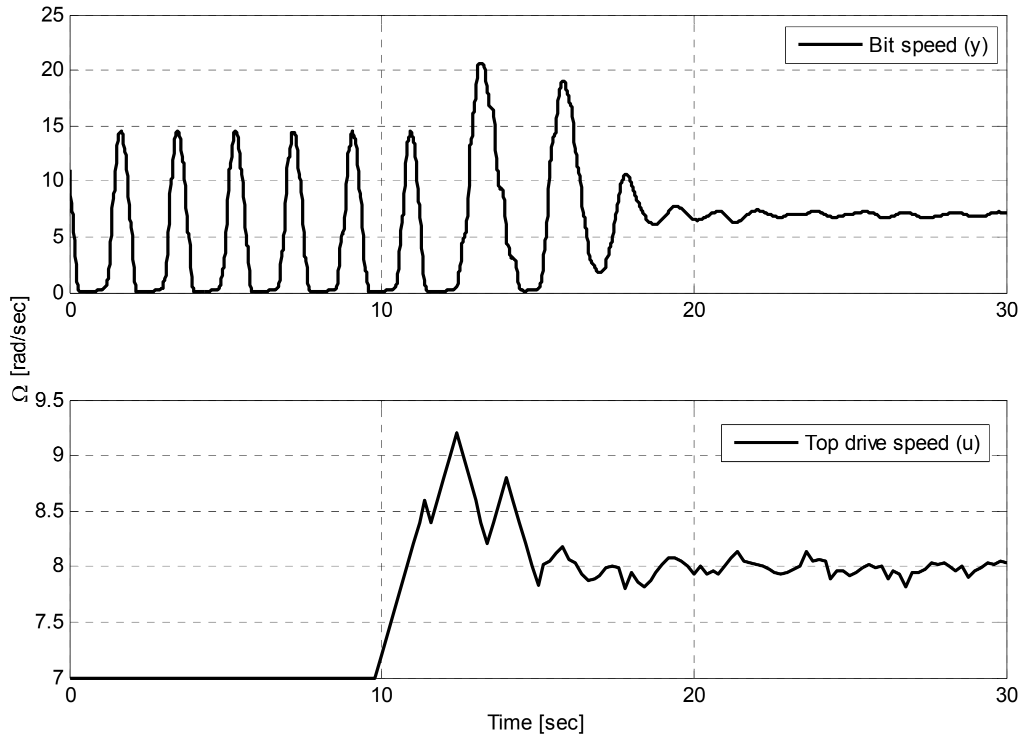

Figure 6.3: Prediction and control horizon 6 seconds, h_c = 1 second

The different simulations show that a prediction and control horizon of 4.5 to 6 seconds gives a satisfactory cure of the stick-slip problem. One should keep in mind that the simulations shown here is when stick-slip has occurred because there is applied a constant input before the NMPC is activated; in the real world would the NMPC run all time and hopefully preventing stick-slip from ever occurring.

Simulations

As mentioned above, the usual case would not be to control the bit speed from stick-slip and then to settle at set point; these simulations were conducted mostly to prove that the controller is able to cure stick-slip if it for some reason should appear. In normal operation will the controller be active all the time and the bit speed will be kept around set point. Therefore it will be natural to look at how the controller performs when set point changes are applied. In the following simulations the bit speed will be controlled to a constant value before a set point change occurs. In the following simulations are the prediction and control horizon 4.5 seconds and h_c = 1.5 seconds.

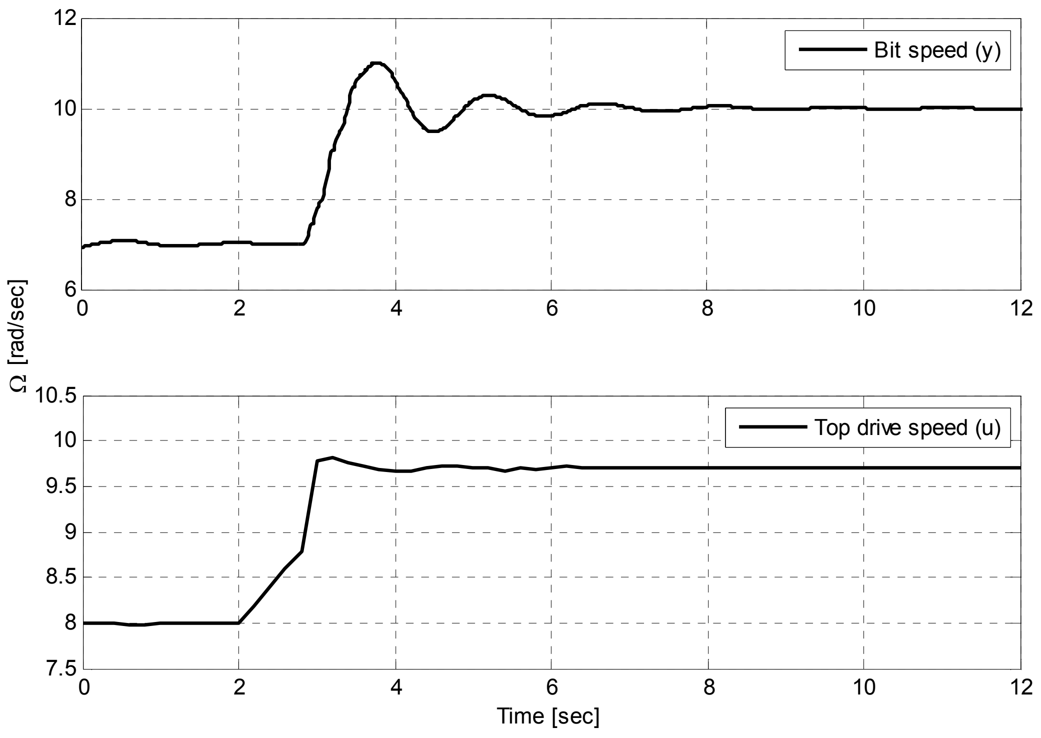

Figure 6.4: Set point change at 2 seconds from 7 to 10 rad/sec

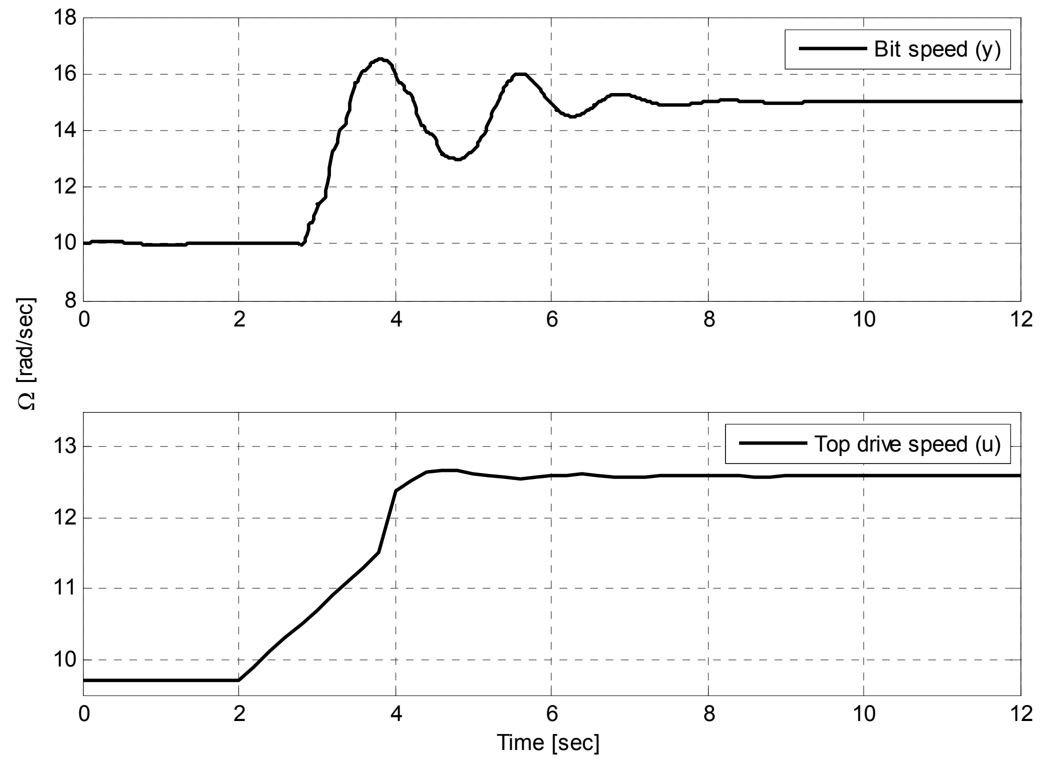

Figure 6.5: Set point change at 2 seconds from 10 to 15 rad/sec

In the previous simulations are both the internal model of the NMPC and the process the same, i.e. the reduced model with 4 states. This is done since the NMPC needs all the states in the prediction and have to get the states from the simulation of the reduced model. There is no physical meaning of the states of the reduced-order model, so the states from the full-order system with 86 states can only be used with an estimator which estimates the 4 states of the reduced-order model from the measurement vector of the full-order system. This was not implemented in this thesis, but could be done with for example an Unscented Kalman Filter; an algorithm for doing this is proposed in Chapter 4. For this reason is the full-order system only controlled open loop by applying the same input as for the closed loop case with 4 states. In Figure 6.6 is the original system exhibiting stick-slip; the input is first held constant at 7 rad/sec before the NMPC is turned on after 10 seconds. Stick-slip is clearly not cured; it just gets a higher frequency and amplitude. This is because of the error in the reduced model; to keep the bit speed constant at 7 rad/sec for the reduced model the input is 8 rad/sec (Figure 6.1). Because of this error and model uncertainties, the NMPC is not able to cure stick-slip for the original system in open loop. Stick-slip is indeed cured for one period after the NMPC is turned on, but the stick-slip behaviour resumes when the input settles again.

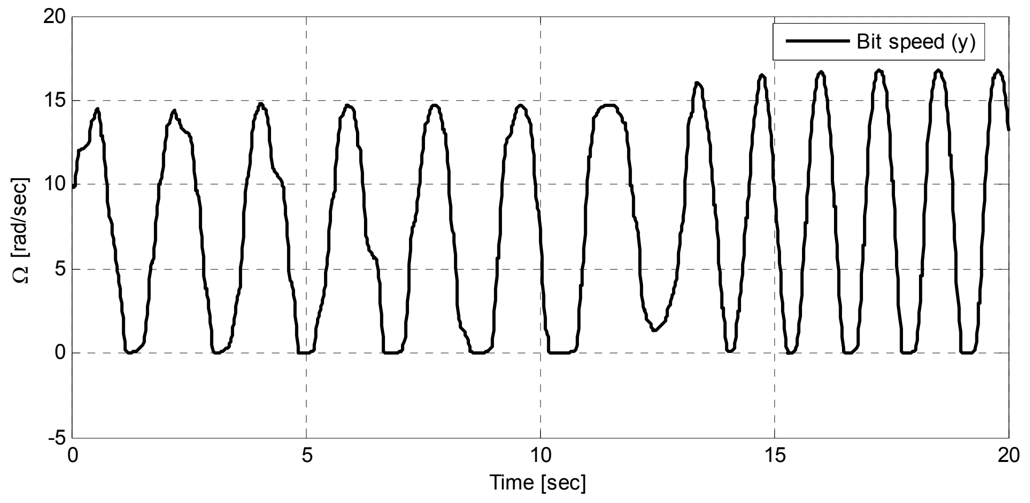

Figure 6.6: Open loop of the original system with 86 states with stick-slip

In Figure 6.7 a set point change from 10 to 12 rad/sec is applied after 100 seconds. Both the closed loop with the reduced-order model and the open loop with the full-order model become stable; the closed loop faster than the open loop. Because of the error in the reduced-model does not both models stabilize at the same value. With closed loop and integral action the original system would have a faster response and no steady-state error. It should still be noticed that the contour of the first period after the set point change is applied is very similar on both, indicating that the models behave quite similar.

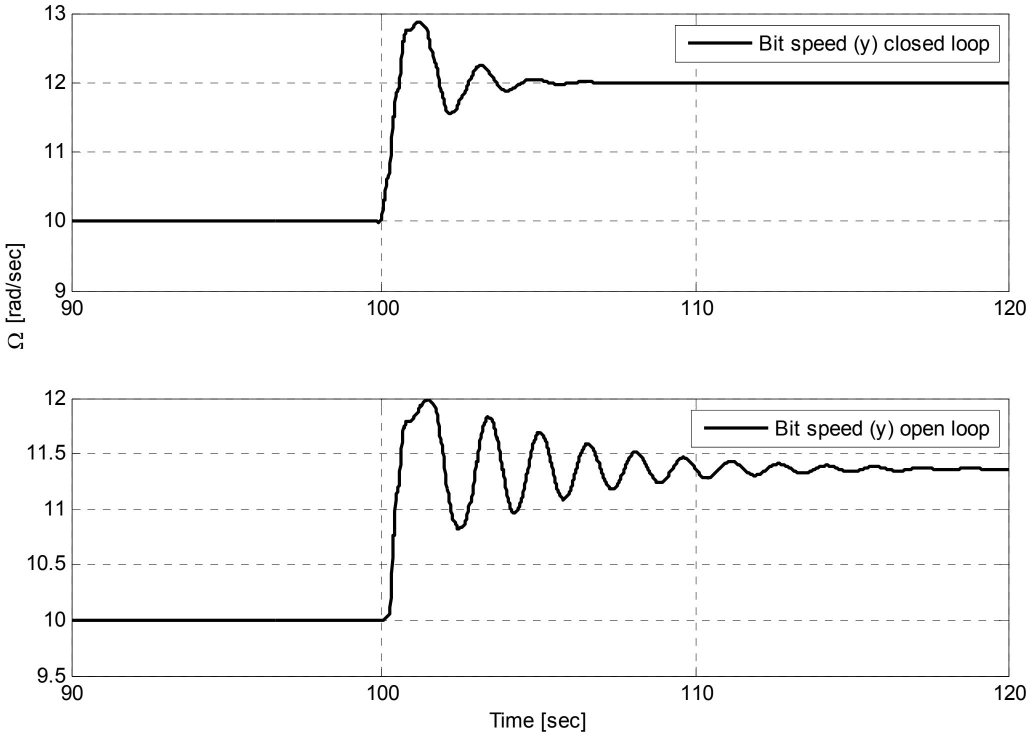

Figure 6.7: Set point change from 10 rad/sec to 12 rad/sec for closed loop and open loop

Remarks
Below there are mentioned some issues regarding this NMPC which need some further comments.

Δu is not included in the objective function; by including it would the changes in u be penalized. One can argue if it necessary to include it in the objective function, the most important thing is the ROP which is kept high if the bit speed is constant at the set point. Rapid changes in the input are already taken care of through the constraints so it should not be a problem.

The drill string used in the NMPC is modelled so that the speed from the top drive is the input, while the bit speed is the output. Therefore there are no constraints on the torque directly which could be desirable. The input constraints will indeed in an indirectly manner represent constraints on the torque; constraining the speed and the speed change will constrain the torque.

In the implementation there are no constraints on the output (the speed of the drill bit). This could be desirable, for instance constrain the output to ±20% of the set point. This would of course make the optimization problem harder to solve. Constraints on the output would prevent high speeds and accelerations, which can be potentially harmful for the drill string.

As the different plots shows there is an error between the steady state input and output. This is of course not the case for a true system; the speed at the bit will be the same as the speed at the top drive. The reason for the error is that the reduced-order model with 4 states has a deviation as clearly shown in Figure 3.7 in Chapter 3. This should of course be no problem for the NMPC; integral action is easily implemented.

It could also be interesting to see how the unconstrained NMPC performs. In Figure 6.8 there is showed a plot where the input first is 10 rad/sec and the unconstrained NMPC with set point 15 rad/sec is turned on after 5 seconds. The stick-slip will here be cured very quickly and the output will settle at steady state in “no time”. Here the input (top drive speed) drops from 10 rad/sec to about 2 rad/sec in one sampling (0.2 seconds). This case will of course not be realistic, but still it is interesting to see the significance of the top drive’s slew rate and the importance of implementing constraints in the controller.

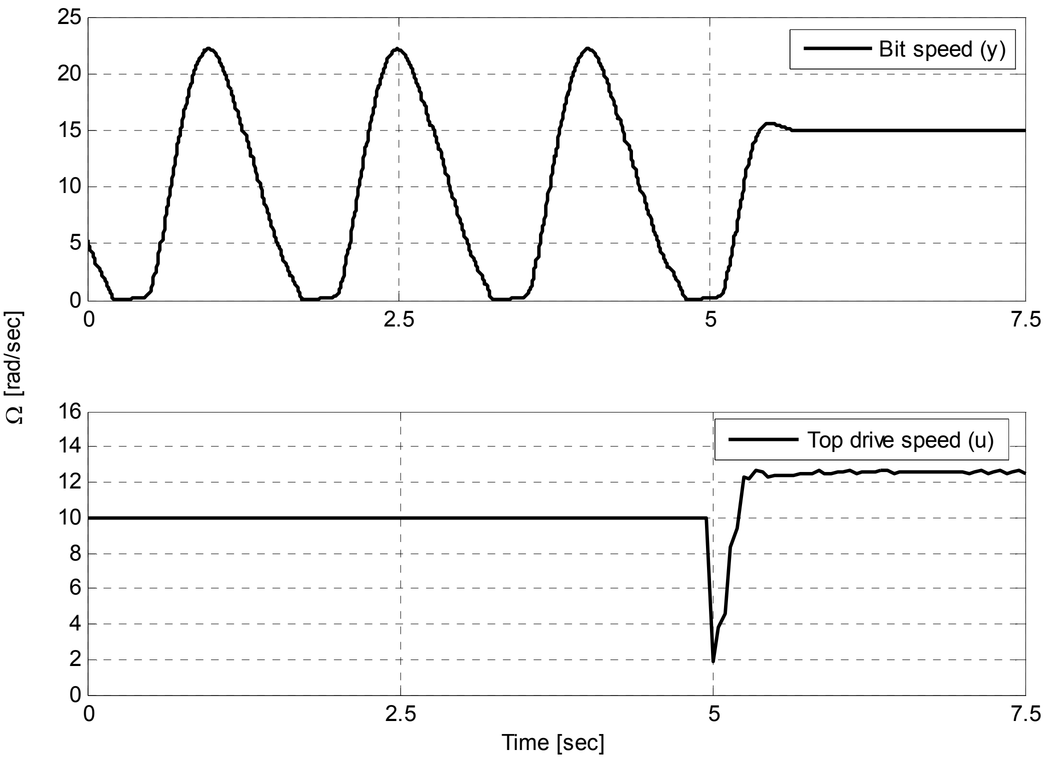

Figure 6.8: Unconstrained NMPC

6.2 *Linear MPC*

In this section is a linear MPC used to control the drill string, which will be of great interest to compare with the NMPC. The linear internal model will only be correct in a certain operating point, and in principle should the linear MPC operate best around this point in distinction to the nonlinear model which is supposed to be a good approach over a larger range of operation. To make the implementation as simple as possible there was used a standard MPC toolbox and linearization tool in Simulink.

Linearization
First a linear model of the drill string to use as the internal model in the MPC had to be found. The nonlinear model of the string was therefore linearized using the tool *Simulink Control Design 3.0*. This tool makes the linearization simple; one just has to define input and output points in the Simulink diagram and then start the design manager. This is an intuitively tool to use with a graphical user interface. The operating point was set to 7 rad/sec, and there was found a full-order model with 86 states. There was also found a reduced-order linear model with 4 states like for the NMPC; this was done using the MATLAB-function *reduce*. All plots were taken with both the model with 86 states and 4 states but the plot were nearly the same which implies that the main dynamics are included in the reduced-order model with 4 states; therefore all plots included in this thesis is with the reduced-order model.

MPC tuning
The MPC was made with the *Model Predictive Control Toolbox 3.1.1* in Simulink. To make the simulations as reasonable as possible the constraints and sampling period were set equal to the NMPC (see section 6.1 for details). The objective function was also in this controller set to only penalize output deviation, which means that the weight tuning will lose its significance. There is indeed a slightly different way to tune this controller compared to the NMPC. There will not be possible to change the period between each time the input should change in the MPC toolbox, but a new output has to be to be calculated with the same frequency as the prediction of the output. There is indeed a choice of how many control moves to apply, which had great impact on control performance. After the last input is computed the input will be kept constant for the rest of the prediction horizon.

Quite surprisingly the prediction horizon did not influence the control performance notably; the output had best performance and was nearly the same with prediction horizon in the range from 5 to 20 seconds. Because of this the prediction horizon was set to 10 seconds and only the control horizon was changed in the rest of the tuning.

In the following simulations the linear MPC is used to control the drill string model created by NOV. This will not be exactly the same model as the one used with the NMPC. If one should be able to use the NMPC to control the full-order model it will be necessary to implement a state estimator which is not done here. If one compares Figure 6.1 and 6.9, the stick-slip behaviour with a constant input of 7 rad/sec will be slightly different; the stick time is longer with the reduced model. But the stick-slip frequency and bit speed amplitude are nearly the same, so it will still make good sense to compare the controllers' performance. Figure 6.9, 6.10 and 6.11 show the plots from the tuning. A constant input of 7 rad/sec is applied until the MPC is activated after 10 seconds.

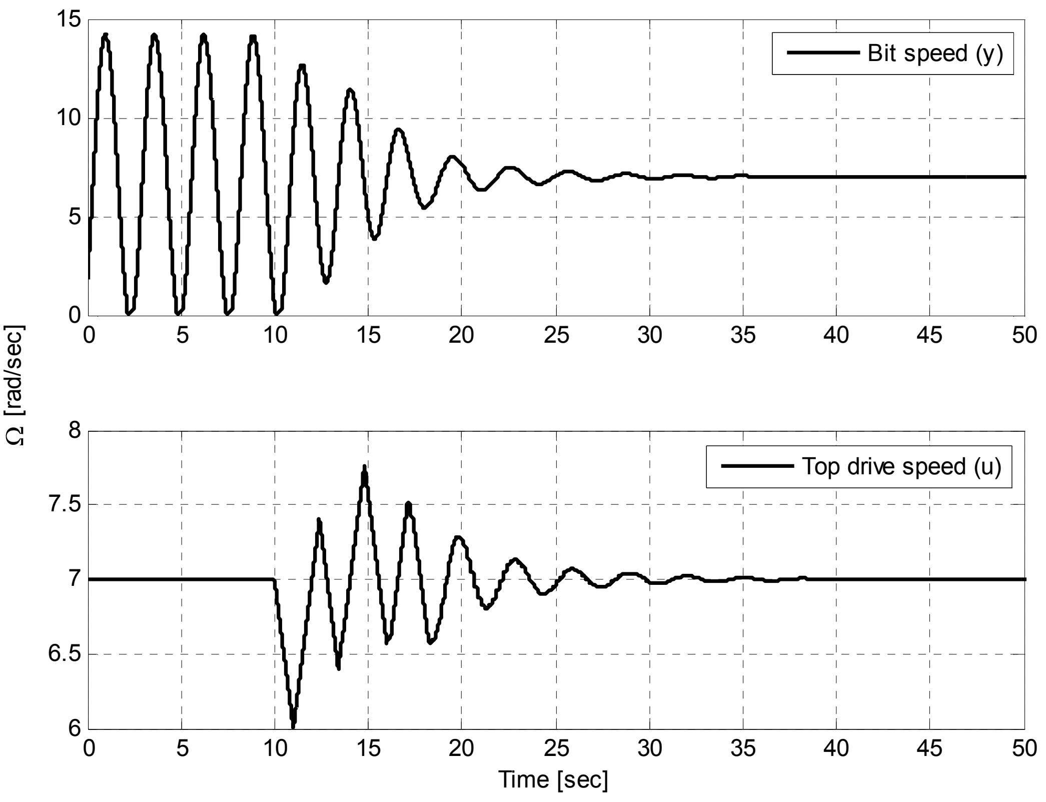

Figure 6.9: Control horizon = 1

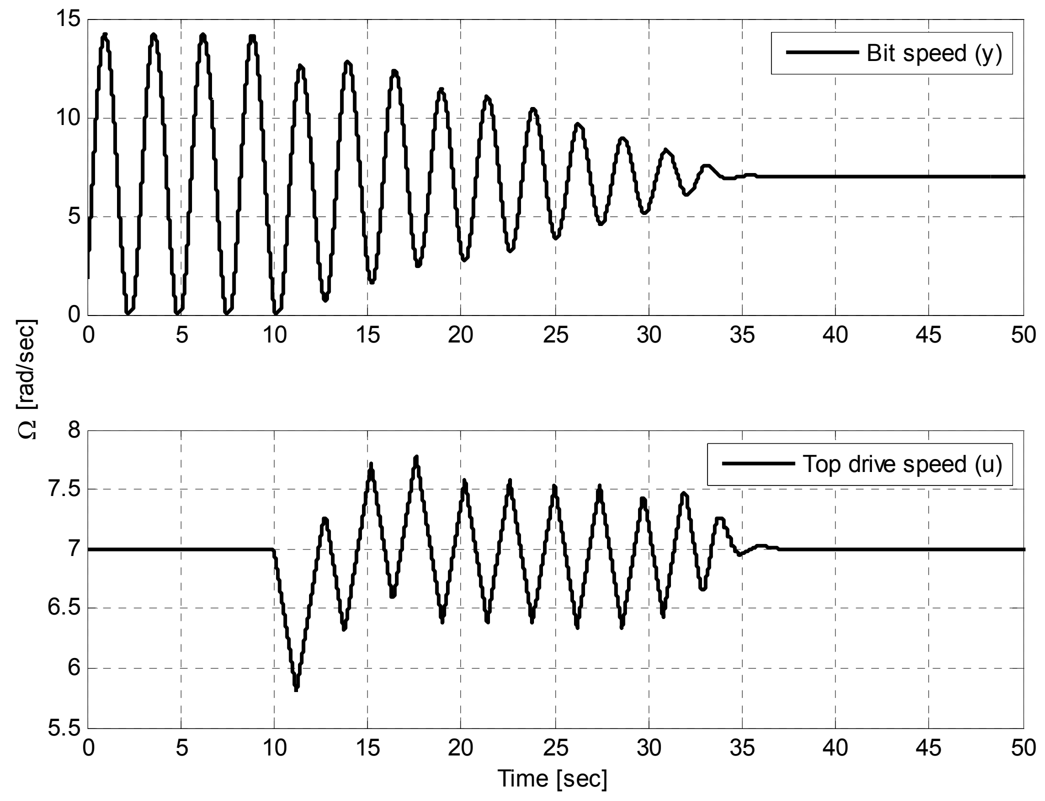

Figure 6.10: Control horizon = 2

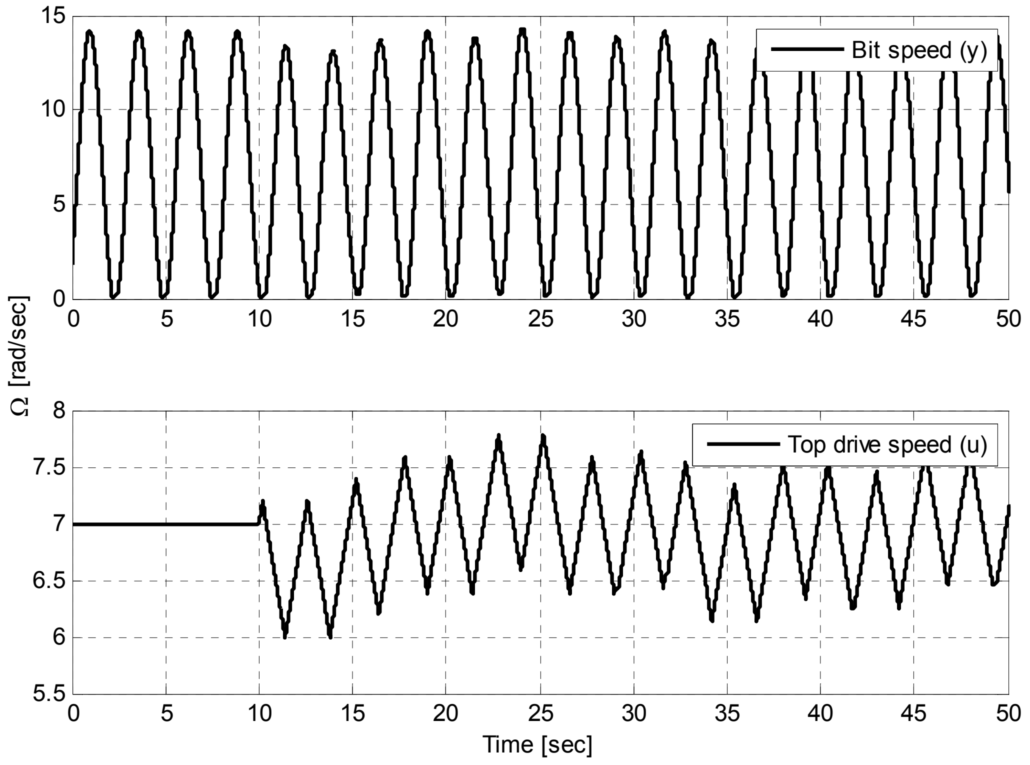

Figure 6.11: Control horizon = 3

As the plots illustrate is the best performance achieved with control horizon = 1. The performance reduces as the control horizon is extended, and when it is set to 3 time steps the MPC will not even cure stick-slip. When the input looks more like a sawtooth wave, it is because of the active constraint on Δu.

Simulations
To compare this controller's performance with the NMPC the same set point changes are applied. The controller has the same parameters as in Figure 6.9. These plots are found in Figure 6.12 and 6.13.

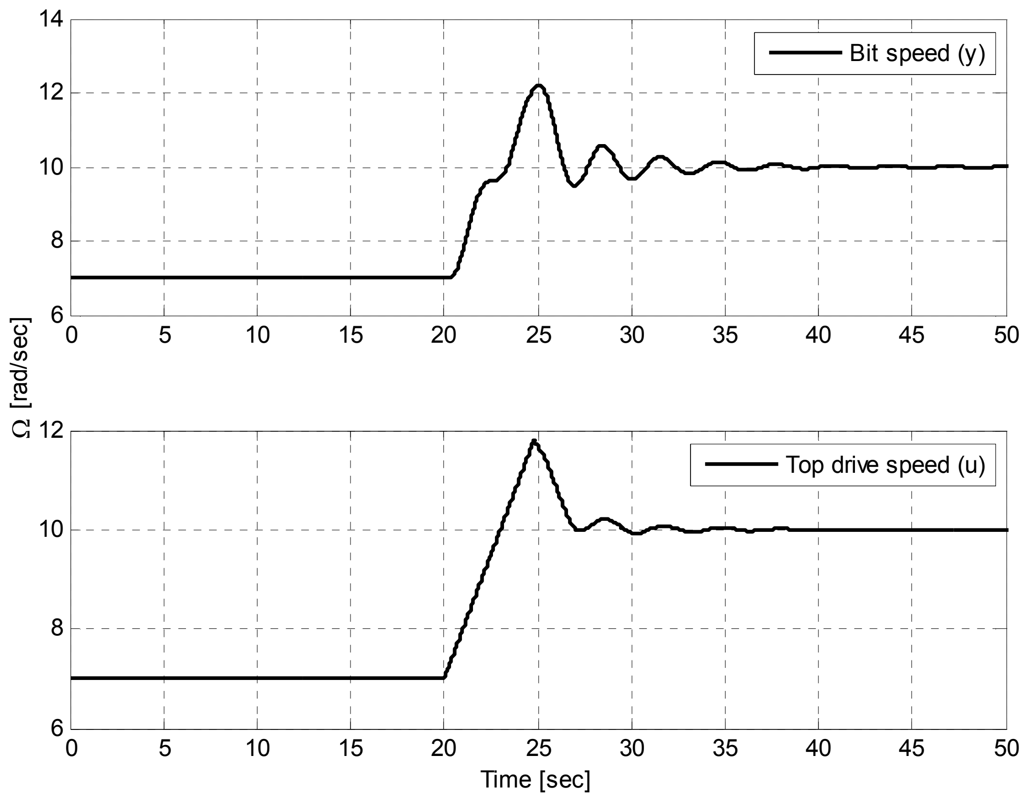

Figure 6.12: Set point change at 20 seconds from 7 to 10 rad/sec

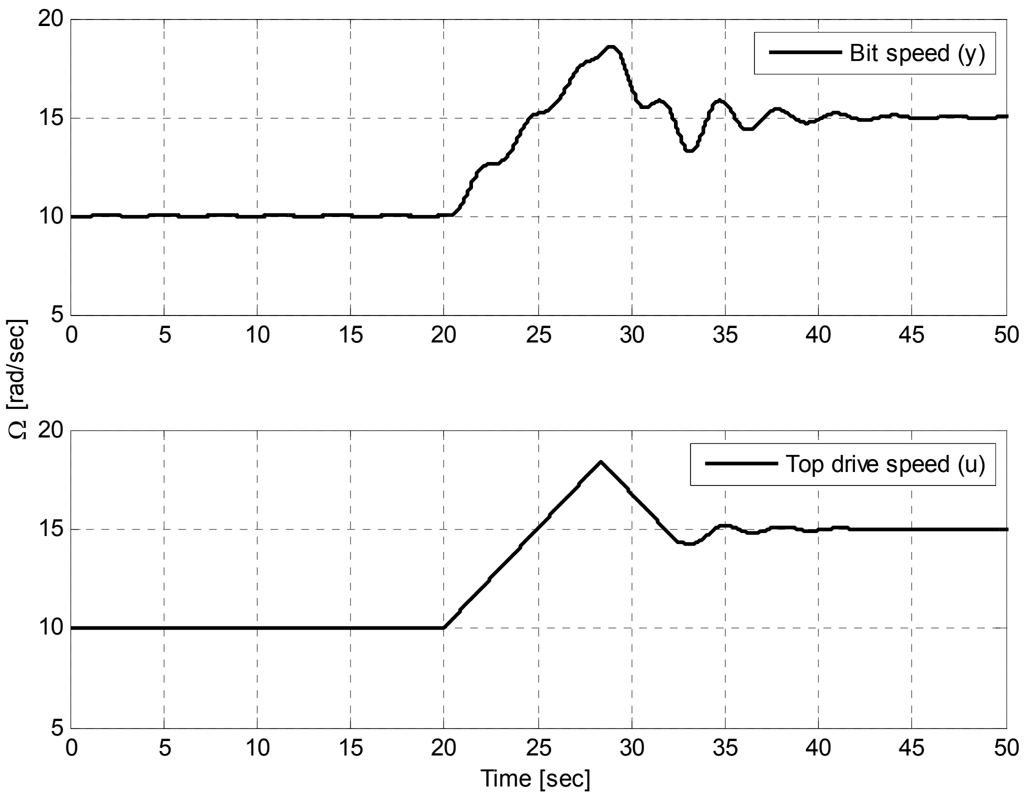

Figure 6.13: Set point change at 20 seconds from 10 to 15 rad/sec

6.3 SoftSpeed

NOV's patented stick-slip prevention system SoftSpeed will in this section be simulated with the same scenarios as the NMPC and linear MPC. Then it will be easy to see which of the systems that performs best.

Simulations

Similar to the simulations with the linear MPC it is the full-order drill string model that is controlled with SoftSpeed.

In the Simulink model of SoftSpeed one has to define the stick-slip period before the simulation is started; it was here set to 1.5 seconds. One also have to define the well geometry, which in this situation is a straight well of 1200 m, and from this the SoftSpeed controller will be tuned automatically. Because of this no tuning had to be conducted on the SoftSpeed controller, but the same scenario with stick-slip and a constant input of 7 rad/sec is plotted to be able to compare the different systems' performance. This plot is found in Figure 6.14 where SoftSpeed is turned on after 10 seconds.

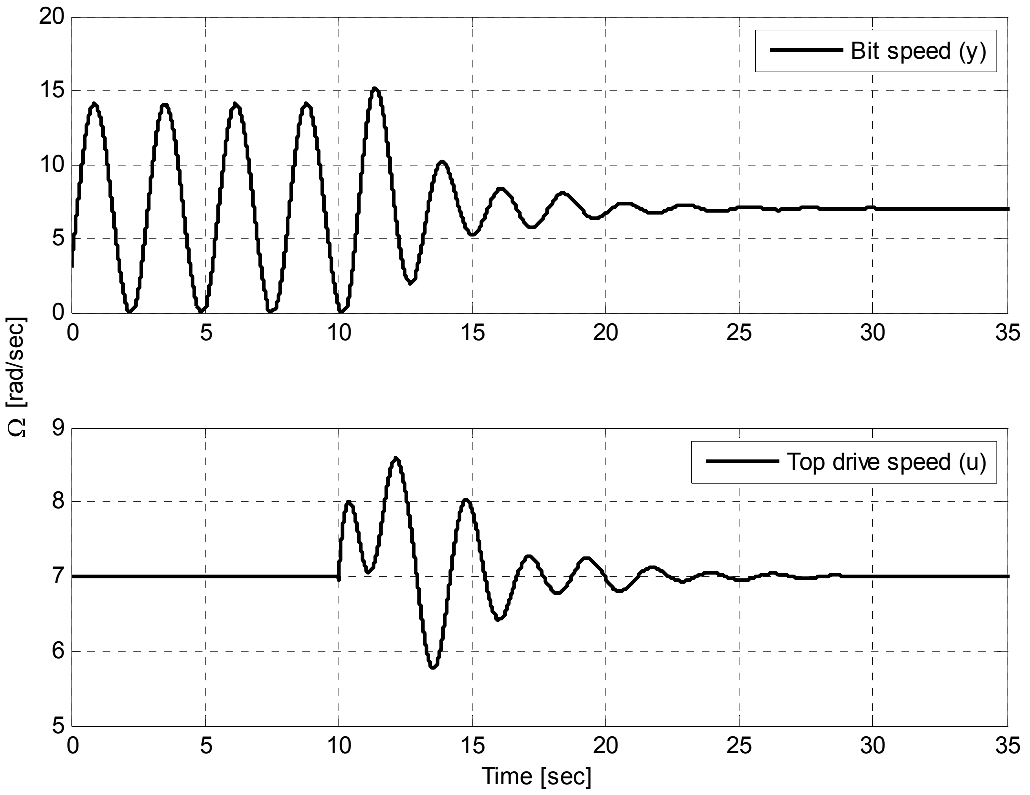

Figure 6.14: SoftSpeed

There were also conducted simulations with the same set point changes as for the previous controllers. These are found in Figure 6.15 and 6.16.

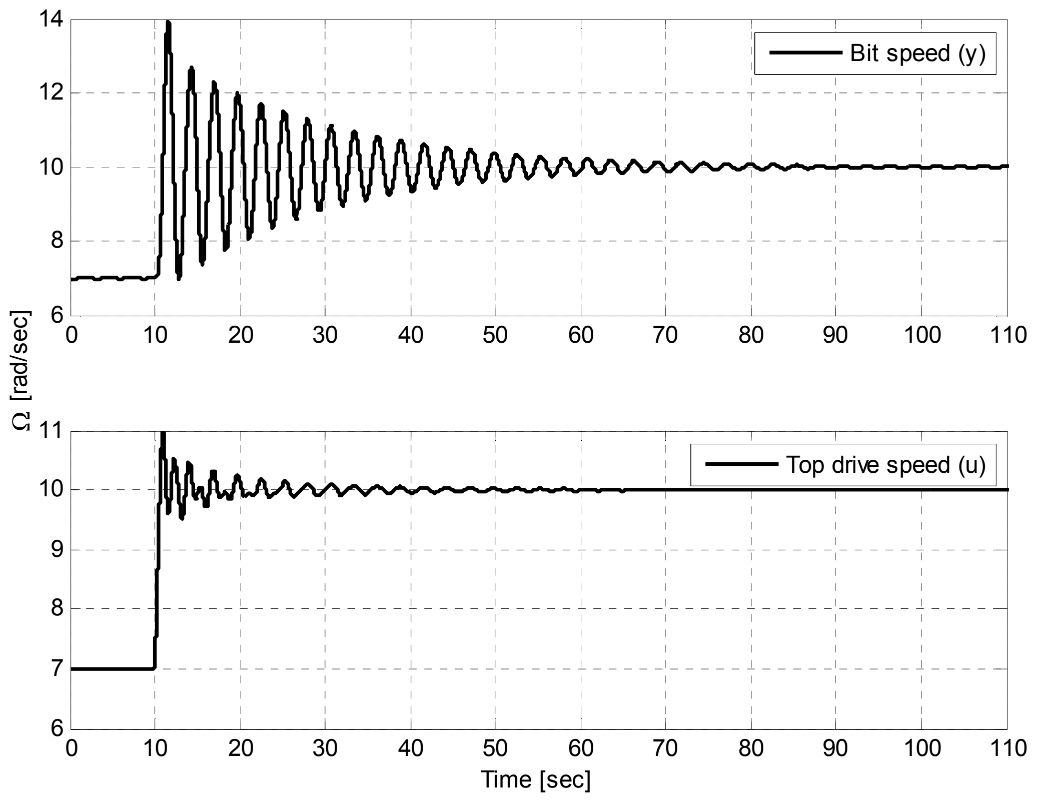

Figure 6.15: Set point change at 10 seconds from 7 to 10 rad/sec

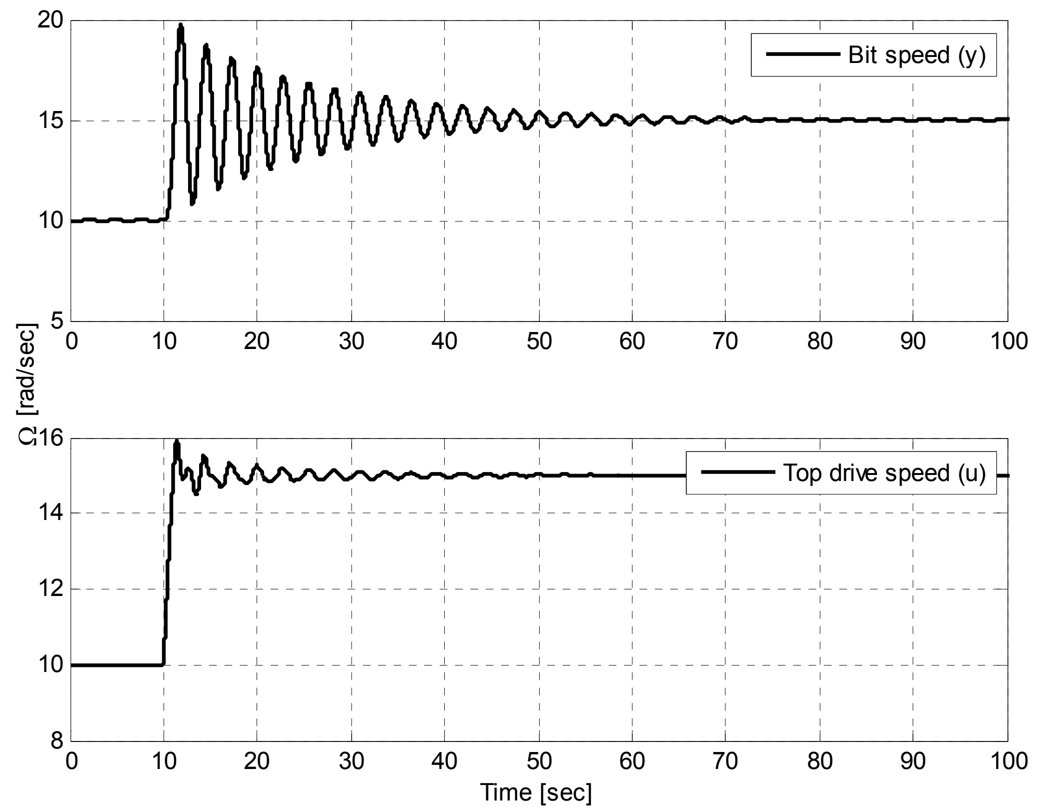

Figure 6.16: Set point change at 10 seconds from 10 to 15 rad/sec

6.4 Comparison

The stick-slip problem has now been addressed with three different solutions; NMPC, MPC and SoftSpeed. In this section a comparison of the systems are done.

Stick-slip

The three different solutions are all capable of curing stick-slip, but there are some differences in the performance. The NMPC is the fastest to reach set point; it takes only about 10 seconds (Figure 6.1) before it is stable. SoftSpeed is somewhat slower than the NMPC; it takes approximately 10-15 seconds before it is stable (Figure 6.14). The slowest of the three to cure stick-slip is the linear MPC; it takes 15-20 seconds before it is stable (Figure 6.9).

Step response

As already mentioned stick-slip should never occur with these prevention systems active, so set point changes are a more realistic approach to determine the performance of the different solutions. Two set point changes were applied to the controllers, 7 to 10 rad/sec and 10 to 15 rad/sec. Before the steps were applied all inputs were steady state at set point. Their performances are compared below.

When a step from 7 to 10 rad/sec is applied

- the NMPC settles to 10 rad/sec after ~6 seconds (Figure 6.4)
- the linear MPC settles to 10 rad/sec after ~20 seconds (Figure 6.12)
- SoftSpeed settles to 10 rad/sec after ~80 seconds (Figure 6.15)

When a step from 10 to 15 rad/sec is applied

- the NMPC settles to 15 rad/sec after ~6 seconds (Figure 6.5)
- the linear MPC settles to 15 rad/sec after ~25 seconds (Figure 6.13)
- SoftSpeed settles to 15 rad/sec after ~70 seconds (Figure 6.16)

As for the case when curing stick-slip, the NMPC is also the fastest to settle at its new set point. In contradictory to the case when curing stick-slip is the linear MPC faster than SoftSpeed when it comes to set point changes. It should be noticed that a set point change from 10 to 15 rad/sec uses more time to settle than from 7 to 10 rad/sec for the linear MPC. With the NMPC is the time it takes about the same. A reason for this is that the linear internal model used in the MPC is linearized around 7 rad/sec, making the performance lower at other areas of operation. The SoftSpeed has a very oscillating response and uses much more time to settle after set point changes than the NMPC and the linear MPC.

7 Discussions

In this chapter the results from the main topics of this thesis are discussed briefly. Because the results and simulations are discussed successively as they appear in the text, this chapter can be regarded as a summarizing discussion.

7.1 Verification and validation

In Chapter 2 the verification and validation of NOV's drill string model was conducted.

Verification
It was performed a code-to-code comparison of the Simulink model provided from NOV and a model implemented as a MATLAB script. Simulations showed that both models comprised the same dynamics, only small deviations were present, which probably are caused by numerical inaccuracy.

Validation
A validation experiment was also undertaken using operational logging data provided by NOV. Such validation experiment shall preferably be done using logging data produced with the only purpose of using it for validation. The logging data was unfortunately not of this quality, which made the validation somewhat limited. The well where the logging came from had also a complicated slope and only measurements from the top and bottom; with non-straight wells the drill string's dynamics will be more complicated and it will be necessary to have several measurements along the string to get a good validation.

The logging data were used in a system identification experiment to find a model describing the logged drill string's behaviour. The intention with this was to be able to compare this model with the model from NOV. Since the logging data did not contain any situations where stick-slip appeared, the model from the system identification lacks much of the information included in the model from NOV, which made them hard to compare.

Simulations were also conducted when using the input from the log as input to the model and the model's output was compared with the logged output. The output (bit speed) from the model was more oscillating than the real drill bit, but it is hard to state if the model is a good or bad representation of a drill string from this observation, not knowing anything about what happens with the rest of the drill string.

7.2 Model reduction

Model reduction is treated in Chapter 3. The main intention was to reduce a large drill string model to a smaller nonlinear model fitted for use in an NMPC. Nonlinear model reduction is quite heavy theoretically, but the method used is closely related to a linear method which is described first. The chapter contains several examples to see how the reduced models perform compared to the original models. From the simulations it can be stated that the most important dynamics will still be present in a reduced model if the states with least contribution to the original model's input-output behaviour is removed.

7.3 MPC

Simulations of MPC to control the drill string are addressed in Chapter 6. Since the bit friction is highly nonlinear it was chosen to use an NMPC. The performance of this controller was compared with how a linear MPC and NOV's patented system SoftSpeed performed. All the methods are able to cure stick-slip if the controllers are tuned properly, but a more realistic scenario is when a set point change is applied. In this situation the NPMC performs best; it makes the output settle faster to the new set point and gives the least aggressive response. It should be brought to mind that in the NMPC is the model and the process the same i.e. the reduced model. In real life would this not be the case. The NMPC should however perform well when used on a real system, but this would require an estimator and integral action in the NMPC.

8 Conclusion

In this thesis the main focus has been on developing a system to use for controlling a drill string that should be able to cure stick-slip at the bit. Different interesting topics have been investigated under the development, such as verification and validation, model reduction and model predictive control. The work has been demanding and rewarding; demanding in the sense that several of the topics are topics which the authors were not familiar with and had to gain knowledge about. This again has been a reward; the topics have been interesting to study and are an important part of cybernetics. Model reduction techniques are very useful and the authors of this thesis believe that the topic will become more and more important as model-based controllers are gaining popularity. Other rewards has been that the authors has developed a better understanding for already known topics, such as predictive control, and have become fairly good in implementing the problems into source code; both should come in hand when dealing with other control problems later in life. Since topics new to the authors of this thesis have been investigated during the work, much time has been spent on understanding the theory thourough.

8.1 Concluding remarks

The outcome from the verification part showed good results; a code-to-code comparison indicated that the model is implemented properly. When the model was to be validated using the provided operational logging data it was hard to say if the model is a good representation of a drill string or not. The data had not the quality needed to carry out the validation properly.

The nonlinear model reduction technique used in this thesis showed promising results in the simulations performed in Chapter 3. When models were reduced quite heavy, still the main dynamics were present. Some reductions gave in fact a bias compared to the original model, but this can be disregarded since integral action in the controller will eliminate this problem.

From the simulations performed in Chapter 6, the NMPC gives best responses compared to the other controllers when set point changes are applied. The main drawback with the NMPC compared to the other solutions is that it is impossible to say anything about computational time for the optimization. These optimizations have to be performed at every time step and needs to be completed faster than the controller's sampling time; this will have great importance especially since the NMPC's control signal is applied directly to the actuator (top drive). How long it will take also varies depending on the system's region of operation. When the simulations were performed, it was noticed a large deviation in the time the SQP solver used to complete the individual optimizations.

In this thesis only a 1200 m vertical well is under consideration. When drilling oil wells would not this always be the case; the well can be much longer and contain slopes. This would of course affect the model and it will most likely not be possible to get a good reduced model with as few as 4 states to represent the system. Slopes would add friction forces acting between the walls and the drill string, not only the bit, and therefore cause more nonlinearity. A reduced model consisting of even more states would consequently contribute to more computational complexity for solving the optimizations in the NMPC. With a 1200 m vertical well is the stick-slip problem rather small, but the work done in this thesis shows some promising results for further research and development. With the use of an NMPC as stick-slip prevention system, it should be possible to maintain an optimal ROP. The nonlinear method used in this thesis is shown to be versatile and can be applied to all kinds of nonlinear systems.

The main conclusions in this thesis can from the statements above be summarized as follows:

- The results from the verification and validation of the drill string model demonstrate the importance of having good logging data to be able to conduct the validation properly.
- By using the selected method for nonlinear model reduction, models with satisfactory dynamics for use in an NMPC were achieved.
- The development of the NMPC was successful and gave promising results; it cures stick-slip efficiently and settled faster than the other systems when set point changes were applied. Before such controller is applied to a real drilling rig it would in fact be necessary to test if the controller is able to run real-time (this could for example be done in a hardware-in-the-loop setup, see Section 8.3).

8.2 Contributions provided by this thesis

Nonlinear model reduction is applicable in many different cybernetic problems. Since nearly all real systems that are to be controlled behave nonlinear, a nonlinear representation will represent the dynamics better than a linear model and will be preferable in many situations. To make the model easier to handle it will be useful to have a method for nonlinear model reduction. This topic is not treated in any subjects at the Department of Engineering Cybernetics, and by the authors of this thesis are this area understood to be quite unknown at the department in general. Hopefully this thesis will contribute to more knowledge about this interesting and appropriate topic, and that it will be included in relevant subjects or generate ideas to future thesis and projects.

The authors of this thesis have not found other places where a model predictive controller has been used to control a drill string's top drive. Furthermore is the practice of applying an MPC's control signal directly to the actuator in general not frequently seen in control systems used in the real world. This thesis can in this setting be regarded as an important contribution for making the control engineering industry familiar with this extended way of utilizing model predictive control in the future.

8.3 Suggestions for further work

One of the most important limitations of the results in this thesis is that the NMPC is not used to control the full scale drill string model. In the simulations the NMPC controls the reduced model, which is exactly the same model as the NMPC's internal model. To be able to use it to control the full model it will be necessary to estimate the states in the reduced model based on measurements from the full model. There exist rather simple strategies for doing this if the model is linear; then a common approach is to use a Kalman filter. If the model is nonlinear the situation will be somewhat more complex. The more recent UKF has shown good results when dealing with nonlinear models.

When oil and gas well drilling are performed, the well geometry changes gradually. Also the drill string dynamics will then be changed depending on how the slope of the well is. The NMPC presented in this thesis is designed for operating with just one specific well geometry.

If it should be used on a real drilling rig there is not implemented a mechanism that can take care of updating the internal model as the system's dynamics changes. There could be several ways to do this. If the complete well geometry is known before the drilling starts, could both a complete and parts of the string be modelled and reduced in advance. Then there could be a mechanism that changes the internal model as the well changes. The NMPC's tuning parameters will also have to be updated as the system and model change. Such auto tuning is neither implemented.

As mentioned in section 8.1, before the NMPC is applied to a real drilling rig, it would be necessary to test if the controller is able to run real-time. Even if the NMPC's performance seems to be good in the simulations performed in this thesis, the controller will be useless if it is not able to run real-time. A cost efficient tool for such tests can be to run Hardware-In-the-Loop (HIL) simulations. NOV is already in possession of a HIL setup used when developing their stick-slip prevention system SoftSpeed. A paper about this system (Kyllingstad & Nessjøen, 2010) was presented at the SPE/IADC Drilling Conference and Exhibition held in New Orleans in 2010, and details about this system can be studied there.

The nonlinear model reduction is based on empirical Gramians which are computed using experimental or simulation data. There can be difficult to select the correct data to get a good reduced model. A method for doing this should be of great interest; in this thesis is a trial-and-error approach used. An effective method for choosing data would be preferred if the method should be used in the industry and gain popularity in control societies.

From the discussions above the following suggestions for further work arises:

- Create an UKF to estimate the states in the reduced internal model of the NMPC.
- Implement an adaptive mechanism that adapts the NMPC's internal model and parameters.
- Conduct HIL simulations to verify the NMPC's real-time performance.
- Develop a method for selection of simulation/experimental data when performing nonlinear model reduction.

Bibliography

Azeez, M. F. A. & Vakakis, A. F. (2001). Proper Orthogonal Decomposition (POD) of a Class of Vibroimpact Oscillations. *Journal of Sound and Vibration, 240*(5), 859-889.

Baumgart, A. (2000). Stick-Slip and Bit-Bounce of Deep-Hole Drillstrings. *Journal of Energy Resources Technology, 122*(2), 78-82.

Brett, J. F. (1992). The Genesis of Torsional Drillstring Vibrations. *SPE Drilling Engineering, 7*(3), 168-174.

Canudas-de-Wit, C., Corchero, M. A., Rubio, F. R. & Navarro-López, E. M. (2005). D-OSKIL: a New Mechanism for Suppressing Stick-Slip in Oil Well Drillstrings. *Paper presented at the 44th IEEE Conference on Decision and Control*, Seville, Spain, 12-15 December.

Challamel, N., Sellami, H., Chenevez, E. & Gossuin, L. (2000). A Stick-Slip Analysis Based on Rock/Bit Interaction: Theoretical and Experimental Contribution. *Paper IADC/SPE 59230 presented at the IADC/SPE Drilling conference* held in New Orleans, Louisiana, 23-25 February.

Cobern, M. E. & Wassell, M. E. (2005). Laboratory Testing of an Active Drilling Vibration Monitoring & Control System. *Paper AADE-05-NTCE-25 presented at the AADE 2005 National Technical Conference and Exhibition* held in Houston, Texas, 5-7 April.

Corchero, M. A., Canudas-de-Wit, C. & Rubio, F. R. (2006). Stability of the D-OSKIL Oscillation Supression Mechanism for Oil Well Drillstrings. *Paper presented at the 45th IEEE Conference on Decision and Control*, San Diego, California, 13-15 December.

Di Palma, F. & Magni, L. (2007). On Optimality of Nonlinear Model Predictive Control. *Systems & Control Letters, 56*(1), 58-61.

Dufeyte, M.-P.& Henneuse, H. (1991). Detection and Monitoring of the Stick-Slip Motion: Field Experiments. *Paper SPE/IADC 21945 presented at the SPE/IADC Drilling conference* held in Amsterdam, The Netherlands, 11-14 March.

Fernando, K. V. & Nicholson, H. (1982). Singular Perturbational Model Reduction of Balanced Systems. *IEEE Transactions on Automatic Control, AC-27*(2), 466-468.

Glavaški, S., Marsden, J. E. (1998). Model Reduction, Centering, and the Karhunen-Loeve Expansion. *Proceedings of the IEEE Conference on Decision and Control,* Tampa, Florida, USA, 16-18 December.

Glegg, S. A. L. & Devenport, W. J. (2001). Proper Orthogonal Decomposition of Turbulent Flows for Aeroacoustic and Hydroacoustic Applications. *Journal of Sound and Vibration, 239*(4), 767-784.

Hahn, J. & Edgar, T. F. (2002). Balancing Approach to Minimal Realization and Model Reduction of Stable Nonlinear Systems. *Industrial & Engineering Chemistry Research, 41*(9), 2204-2212.

Hahn, J. & Edgar, T. F. (2002). An Improved Method for Nonlinear Model Reduction Using Balancing of Empirical Gramians. *Computers and Chemical Engineering, 26*(10), 1379-1397.

Halsey, G. W., Kyllingstad, Å. & Kylling, A. (1988). Torque Feedback Used to Cure Stick-Slip Motion. *Paper SPE 18049 presented at the 63rd Annual Technical Conference and Exhibition of SPE* held in Houston, Texas, 2-5 October.

Helsinki University of Technology (n.d.). *Unscented Kalman Filter.* Retrieved 08.04.2010 from http://signal.hut.fi/kurssit/s884221/ukf.pdf.

Johannessen, M & Myrvold, T. (2009). *Stick-Slip Prevention System.* Project work, Department of Engineering Cybernetics, Norwegian University of Science and Technology.

Julier, S. J. & Uhlmann, J. K. (1997). A New Extension to the Kalman Filter to Nonlinear Systems. *Signal Processing, Sensor Fusion, and Target Recognition VI (Proceedings Volume), 3068*, 182-193.

Karhunen, K. (1946). Zur spektral theorie stochasticher prozesse. *Annales Academiae Scientiarum Fennicae, 37*(A1), 1-7.

Keerthi, S. S. & Gilbert, E. G. (1988). Optimal Infinite-Horizon Feedback Laws for a General Class of Constrained Discrete-Time Systems: Stability and Moving-Horizon Approximations. *Journal of Optimization Theory and Applications, 57*(2), 265-293.

Kerschen, G., Golinval, J-C, Vakakis, A. F. & Bergman L. A. (2005). The Method of Proper Orthogonal Decomposition for Dynamical Characterization and Order Reduction of Mechanical Systems: An Overview. *Nonlinear Systems, 41*(1-3), 147-169.

Khalil, H. K. (2002). *Nonlinear Systems.* NJ: Prentice Hall.

Khulief, Y. A., Al-Sulaiman, F. A. & Bashmal, S. (2007). Vibration Analysis of Drillstrings with Self-Excited Stick-Slip Oscillations. *Journal of Sound and Vibration, 299*(3), 540-558.

Kirby, M. & Sirovich, L. (1990). Application of the Karhunen-Loève Procedure for the Characterization of Human Faces. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 12*(1), 103-108.

Krysl, P., Lall, S. & Marsden, J. E. (2001). Dimensional Model Reduction in Non-Linear Finite Element Dynamics of Solids and Structures. *International Journal for Numerical Methods in Engineering, 51*(4), 479-504.

Kyllingstad, Å. & Halsey, G. W. (1988). A Study of Slip/Stick Motion of the Bit. *SPE Drilling Engineering, 3*(4), 369-373.

Kyllingstad, Å. & Nessjøen, P. J. (2009). A New Stick-Slip Prevention System. *Paper SPE/IADC 119660 presented at the SPE/IADC Drilling Conference and Exhibition* held in Amsterdam, The Netherlands, 17-19 February.

Kyllingstad, Å. & Nessjøen, P. J. (2010). Hardware-in-the-Loop Simulations Used as a Cost-Efficient Tool for Developing an Advanced Stick-Slip Prevention System. *Paper SPE/IADC 128223 presented at the SPE/IADC Drilling Conference and Exhibition* held in New Orleans, Louisiana, USA, 2-4 February.

Leine, R. I., van Campen, D. H. & Keultjes, W. J. G. (2002). Stick-Slip Whirl Interaction in Drillstring Dynamics. *Journal of Vibration and Acoustics, 124*(2), 209-220.

Lindegaard, K.-P. W. (2003). *Acceleration Feedback in Dynamic Positioning.* PhD Thesis, Department of Engineering Cybernetics, Norwegian University of Science and Technology, Norway.

Loève, M. (1946). Fonctions al´eatoires de second ordre. *Revue Sci.,* 84, 195-206.

Maciejowski J. M. (2002). *Predictive Control with Constraints.* Essex: Pearson Edu. Ltd.

Moore, B. C. (1981). Principal Component Analysis in Linear Systems: Controllability, Observability, and Model Reduction. *IEEE Transactions on Automatic Control, AC-26*(1), 17-32.

Naylor, T. H. & Finger, J. M. (1967). Verification of Computer Simulation Models. *Management Science, 14*(2), B92-B106.

Nocedal, J. & Wright, S. J. (2006). *Numerical Optimization.* New York: Springer.

Oberkampf, W. L. & Trucano, T. G. (2002). Verification and Validation in Computational Fluid Dynamics. *Progress in Aerospace Science, 38*(3), 209-272.

Pannocchia, G. & Rawlings, J. B. (2003). Disturbance Models for Offset-Free Model Predictive Control. *AIChE Journal, 49*(2), 426-437

Patek, S. N. (2001). Spiny Lobsters Stick and Slip to Make Sound. *Nature, 411*(6834), 153-154.

Pavone, D. R. & Desplans, J. P. (1994). Application of High Sampling Rate Downhole Measurements for Analysis and Cure of Stick-Slip in Drilling. *Paper SPE 28324 presented at the SPE 89th Annual Technical Conference and Exhibition* held in New Orleans, LA, 25-28 September.

Phillips, J. R., Daniel, L. & Silveira, L. M. (2003). Guaranteed Passive Balancing Transformations for Model Order Reduction. *IEEE Transactions on Computer-aided Design of Integrated Circuits and Systems, 22*(8), 1027-1041.

Puebla, H. & Alvarez-Ramirez, J. (2008). Suppression of Stick-Slip in Drillstrings: A Control Approach Based on Modeling Error Compensation. *Journal of Sound and Vibration, 310*(4-5), 881-901.

Robnett, E. W., Hood, J. A., Heisig, G. & Macpherson, J. D. (1999). Analysis of the Stick-Slip Phenomenon Using Downhole Drillstring Rotation Data. *Paper SPE/IADC 52821 presented at the SPE/IADC Drilling Conference* held in Amsterdam, The Netherlands, 9-11 March.

Rowley, C. W. & Marsden, J. E. (2000). Reconstruction Equations and the Karhunen-Loève Expansion for Systems with Symmetry. *Physica D: Nonlinear Phenomena, 142*(1-2), 1-19.

Sananikone, P., Kamoshima, O. & White, D.B. (1992). A Field Method for Controlling Drillstring Torsional Vibrations. *Paper IADC/SPE 23891 presented at the IADC/SPE Drilling Conference* held in New Orleans, Louisiana, 18-21 February.

Schlesinger, S., et al (1979). Terminology for Model Credibility. *Simulation, 32*(3), 103-104.

Serrarens, A. F. A., van de Molengraft, M. J. G., Kok, J. J. & van den Steen, L. (1998). H_∞ Control for Supressing Stick-Slip in Oil Well Drillstrings. *IEEE Control System Magazine 18*(2), 19-30.

Skogestad, S. (2004). Control Structure Design for Complete Chemical Plants. *Computers and Chemical Engineering 28*(1-2), 219-234.

Skogestad, S. & Postlethwaite, I. (2005). *Multivariable Feedback Control.* West Sussex: Wiley & Sons.

Wan, E. A. & van der Merwe, R. (2000). The Unscented Kalman Filter for Nonlinear Estimation. *Proceedings of the IEEE 2000 Adaptive Systems for Signal Processing, Communications, and Control Symposium*, Lake Louise, Alberta, Canada, 1-4 October.

A Proof of asymptotic stability of strictly passive system

In Chapter 3, Example 3.2 is the system shown to be strictly passive. Here is the proof of asymptotic stability of strictly passive system as given in Khalil (2002).

Consider a system given by

$$\begin{aligned} \dot{x} &= f(x,u) \\ y &= h(x,u) \end{aligned} \tag{A.1}$$

The system is strictly passive with $V(x)$ as its storage function.
With $u = 0$ will $\dot{V}$ satisfy the inequality $\dot{V} \le -\psi(x)$ where $\psi(x)$ is positive definite.
The inequality can be used to show that $V(x)$ is positive definite.
For any $x \in \mathbb{R}^n$ will the equation $\dot{x} = f(x,0)$ have a solution $\phi(t;x)$
starting from x at $t = 0$ and defined on some interval $[0,\delta]$.
Integrating the inequality $\dot{V} \le -\psi(x)$ gives

$$V(\phi(\tau,x)) - V(x) \le -\int_0^\tau \psi(\phi(\tau;x))dt, \forall \tau \in [0,\delta]$$

By using $V(\phi(\tau,x)) \ge 0$ implies that $V(x) \ge \int_0^\tau \psi(\phi(\tau;x))dt$
Suppose that there is a $\bar{x} \ne 0$ such that $V(\bar{x})$=0.
The foregoing inequality implies that

$$\int_0^\tau \psi(\phi(\tau;\bar{x}))dt = 0, \forall \tau \in [0,\delta] \Rightarrow \psi(\phi(\tau;\bar{x})) \equiv 0 \Rightarrow \phi(\tau;\bar{x}) \equiv 0 \Rightarrow \bar{x} \equiv 0$$

which contradicts the claim that $\bar{x} \ne 0$. Thus, $V(x) > 0$ for all $x \ne 0$.
This qualifies $V(x)$ as a Lyapunov function canidate and since $\dot{V}(x) \le -\psi(x)$
the conclusion can be drawn that the origin is asymptotically stable.

B MATLAB/Simulink files

When submission of this thesis was done in DAIM, MATLAB/Simulink files were attached electronically as a ZIP-file. In this appendix these files are listed and a description of each file is given.

Linear mod.red
Stringdata.m: Contains data needed to run drill string model.
SetupLinearizing.mdl: Simulink setup to use when linearizing drill string with *Simulink Control Design*.
LinearizedStringmodel.mat: Contains full scale linear model.
LinModRed.m: Calls the MATLAB routine for linear model reduction.
workspace.mat: Contains the full scale and reduced linear models.

Nonlinear mod.red
nonlinear_model_reduction.m: Routine for the complete reduction.
simulations.m: Routine for simulating the reduced and original system.
drillstring.m: The ODE function for the original scaled system.
drillstring_reduced.m: The ODE function for the reduced scaled system.
control_gramian.m: Routine for computing the controllability gramian.
observ_gramian.m: Routine for computing the observability gramian.
balancing.m: Routine for computing the transformation matrix that balances the system.

Linear MPC
Stringdata.m: Contains data needed to run drill string model.
LinMPC.mdl: Simulink setup with MPC and drill string.
MPCdesign4.mat: MPC with 4-state internal model.
MPCdesign86.mat: MPC with 86-state internal model.
plotting.m: Routine for plotting input and output.

NMPC
nmpc.m: Contains the nonlinear model predictive controller.
objfun.m: Contains the objective function which is minimized to get the optimal input.
samplehold.m: Contains the routine for
drillstring.m: The ODE function for the full-order model.
drillstring_reduced.m: The ODE function for the scaled reduced-order model.