FACULTY OF SCIENCE AND TECHNOLOGY

BACHELOR'S THESIS

Study programme / specialisation: Energy and Petroleum technology	*Spring* semester, 2023 Open
Author: Edward Larsen	
Supervisor at UiS: Hans Joakim Skadsem Co-supervisor: N/A External supervisor(s): N/A	
Thesis title: Prediction of Mechanical Friction Forces on Casing being run	
Credits (ECTS): 20	
Keywords: Casing operation, drilling, torque, drag, modelling	Pages: 46 + appendix: 23 Stavanger, *14/06/2023*

Abstract

In recent years, advancements in drilling technology have led to increasingly complex wellbore trajectories, particularly in the case of horizontal wells and extended reach wells (ERWs). This complexity necessitates highly accurate models for predicting forces exerted on downhole equipment during drilling operations, especially given the amplified gravitational and frictional forces encountered. This thesis presents an advanced torque and drag model, capable of forecasting the drag forces experienced by casing during tripping operations. Despite notable technological advancements, the core mathematical principles underpinning most torque and drag models have largely remained unaltered since their inception. Building upon these foundational principles, the code developed in this project provides improved predictive accuracy which is critical during the planning and design stages of well construction. The developed code applies the classic torque and drag model to quantify the drag on casing during running operations, using well paths from the Ullrigg drilling and well site. The model is intended for use in the early stages of well planning, where it assists in assessing casing choices and the potential necessity for specific techniques, such as flotation or rotation. By providing the ability to predict potential complications—like pipe sticking, buckling, or failure due to excessive downhole forces—the model can guide preventive measures that enhance the efficiency, safety, and cost-effectiveness of casing operations. Hence, this model offers a valuable tool for minimizing equipment damage and optimizing modern well construction practices [1] [2]

Acknowledgements

I would like to thank my supervisor Hans Joakim Skadsem for invaluable guidance and support throughout the work on the thesis.

List of Tables

List of Figures

Nomenclature

$_i$	Inner pipe area, nominal ID
$_o$	Outer pipe area, nominal OD
κ	Wellbore curvature
μ	Friction coefficient
ϕ	Wellbore inclination
ϕ_n	new wellbore inclination
ρ_i	Density inside pipe
ρ_m	Density of mud
ρ_o	Density outside pipe
ρ_s	Density of steel
θ	Wellbore azimuth
θ_n	new wellbore azimuth
$\vec{t}$	Tangent vector
$\vec{w}_c$	Contact force vector
$\vec{w}_d$	Friction force vector
BF	Buoyancy factor
n_z	Z-component of normal vector
RF	ratio factor
s	measured depth
s_target	measured depth to target
w_p	Weight of pipe in air per unit
w_{bp}	Bouyed weight of segment per unit
w_c	Contact force
b_z	Z-component of binormal vector
t_z	Z-component of unit tangential vector

Contents

Chapter 1

Introduction

By virtue of better techniques and technologies that have evolved throughout the decades, modern wells are far more intricate than they were just 50 years ago, horizontal wells and extended reach wells (ERW) have become more common throughout the years. With the increasing prevalence of horizontal and Extended Reach Well (ERW) drilling techniques, long horizontal sections have become a common feature in modern wells[1]. This poses significant operational challenges as these sections typically generate high frictional forces acting on downhole equipment such as casing, drill pipe, and drill string. The change in trajectory and its inclination means that both friction and gravity work against the casing, potentially causing it to get stuck. This context has fueled a demand for precise models capable of accurately predicting these forces, to mitigate risks and ensure safe and efficient operations.

A reliable torque and drag model plays an essential role in the planning and design stages of well construction. The importance of these models is anchored in their ability to estimate the forces that are anticipated during drilling and completion. This foresight is crucial in the creation of a safe and efficient well trajectory. Moreover, it guides the selection of appropriate drilling equipment to match the specific challenges of each operation. Another vital aspect of torque and drag models is their contribution to the prevention of equipment damage[1]. Drilling is a demanding process that involves high mechanical stresses. Excessive torque and drag can lead to significant wear and tear on drilling equipment. Such wear can escalate into equipment failure, which can not only be costly, but also dangerous[3]. By utilizing models, these problems can be anticipated and preventive measures can be put in place, protecting both personnel and valuable drilling equipment. Torque and drag modelling is an important tool for optimizing drilling and completion operations. The models play a critical role not only in ensuring safety but also in enhancing operational efficiency. An example of how these models play an integral role in drilling operations can be seen in their application during the drilling phase. One notable application during the drilling phase is the generation of "hookload roadmaps"[4]. These roadmaps outline the expected hookload — the weight experienced by the hoisting system — at various depths during lifting, lowering, and rotating operations. Any significant discrepancy between the field measurements and the model's prediction can indicate potential issues within the well, such as an accumulation of cuttings particles causing increased friction, or a poorly aligned well resulting in greater contact forces and friction. By identifying these anomalies early, the model facilitates prompt intervention and minimizes operational delays, contributing to

overall productivity and cost-effectiveness. [4]

Torque and drag models also play a key role in predicting and managing operational issues. These models provide insights into potential problems like pipe sticking, buckling, or failure due to excessive downhole forces. They are equally vital during casing operations, where the forces at play can result in the casing becoming stuck, damaged, or falling short of the target depth. Accurate prediction and management of the drag forces during casing operations can greatly reduce the risk of such problems. These predictions allow for the timely implementation of measures to manage or even avoid these issues, enhancing the reliability and continuity of drilling operations. [1] [3] [5]

The program presented in this paper, which builds upon over two decades of development in torque and drag models, is adept at calculating these crucial drag forces during casing operations. Even though technology and computational capabilities have advanced significantly over the years, the fundamental mathematical principles underpinning the soft string torque and drag model remain consistent. The key to these models' enduring utility lies in their ability to adapt and account for the diverse range of operational scenarios encountered in well construction, from drilling to casing operations.

1.1 Objective

The objective of this thesis is to implement, verify, and apply a computational model for calculating the frictional forces encountered during the tripping operations in deviated and extended reach wells. Through a meticulous study of the principles of drilling dynamics and by using the minimum curvature, soft-string model, the aim is to gain insights into how the well path trajectory influence drag forces. The model will take into account parameters such as wellbore geometry, casing string properties, and frictional coefficients.

Utilizing this model, the intention is to investigate and illustrate the disparity between tripping in and tripping out operations in terms of drag forces for three different well paths, namely well U1, U2, and U6B from Ullrig Test Centre [6]. The case study findings are expected to demonstrate the complexities associated with different well trajectories, thus providing crucial insights into the challenges faced during casing running operations. Ultimately, this research seeks to contribute valuable insights to the industry, aiding in the design and planning of future well operations.

Chapter 2

Literature Review

2.1 Torque and drag fundamentals

2.1.1 Drag

Drag refers to the frictional resistance experienced by the casing or drill string as it moves along the wellbore. This force can either contribute to increasing tension (when pulling the drill string out) or increasing compression (when sliding the drill string in) [2]. In the models developed over the years, both torque and drag are attributed solely to the sliding friction forces that occur due to the drill string's contact with the wellbore [7]. This interaction becomes especially significant in deviated wells, as gravity pulls the drill string downwards, causing it to contact the wellbore wall and leading to the development of drag forces. Understanding and managing these forces is crucial for the efficiency, safety, and overall success of both casing-running and drilling operations.

2.1.2 Torque

Torque is typically the product of force and the length of the lever arm. In the context of drilling, torque refers to the rotational force required to turn the drill pipe. This force is essential for overcoming the friction that arises within the wellbore and at the drill bit while rotating. As torque is applied, some of it is lost during transmission, which reduces the amount of torque available at the bit for breaking the rock. High torque and drag forces often occur simultaneously. In an ideal vertical well, torque losses would be minimal, with only a slight reduction due to the viscous force exerted by the drilling mud. However, in deviated wells, particularly in extended reach wells, torque loss can be significant. This loss can be a limiting factor in casing running operations, as well as the drilling process. Torque is directly proportional to the radius at which rotation takes place, the coefficient of friction, and the side (normal) force exerted by the casing against the wellbore wall. [8] [7]

What causes torque and drag

Torque and drag in drilling and casing operations are primarily caused by the friction that arises from the contact between the drill string or casing and the wellbore. This frictional force is generally modeled using the Coulomb friction model. According to the Coulomb friction model, the frictional force (F) is the product of the normal force

(N) and the friction coefficient (μ), mathematically represented as $F = \mu N$. Here, N is the normal force, which represents the contact force between the drill string or casing and the wellbore, and μ is the friction coefficient that characterizes the frictional resistance between the two contacting surfaces. Importantly, the frictional force operates in the opposite direction to the movement, thus either increasing the tension when pulling out or reducing it and thereby increasing compression during run-in operations. While the Coulomb model offers a basic framework for understanding the interactions leading to torque and drag, it's important to note that there are additional factors not considered in this model which may also influence the behavior of torque and drag in wellbore operations [9][7][3]. There are still other factors contributing to an increase in drag forces, such as:

- Restrictive hole conditions
- Sloughing holes
- Doglegs
- Keyseat - A keyseat refers to a ledge or groove carved into the wall of a wellbore. This can occur when there is a drastic change in direction during drilling, known as a dogleg, or if a hard formation ledge is left between softer formations that enlarge over time. During tripping operations tools such as drill collars, stabilizers, tool joints, and bits, that are larger in diameter may get stuck. In order to prevent keyseating, it is advised to ensure that changes in direction in the wellbore are gradual and smooth. If keyseating is to occur, the solution is to widen the eroded channel to a size that can accommodate the tools.[10]
- Differential sticking - Differential sticking is a condition where the drill string cannot be moved. Differential sticking is a product of the differential pressure between the reservoir and the wellbore, and the area that the pressure is acting upon. This causes the pipe to be pushed into the mud filter cake.
- Cuttings accumulation and poor hole cleaning
- Wellbore trajectory
- Loss of circulation

2.2 Wellbore Trajectory

Inclination (ϕ) is the angle between the vertical and the tangent of the wellbore. A vertical well has an inclination of 0°, while a horizontal well has an inclination of 90°. The azimuth (θ)denotes the angle between true north and the tangent of the wellbore projected onto a horizontal plane, beginning with 0° at north and moving clockwise, with west at 270°. The usual representation of a directional well is shown in figure 2.1[1]

A variety of surveying instruments are used in the oil and gas industry to measure inclination and azimuth angles. There are several types of surveying instruments, magnetic devices, including single-shot, and multi-shot, as well as more advanced

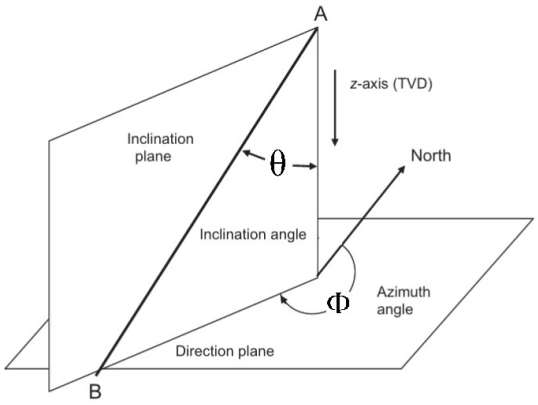

Figure 2.1: Hole inclination and azimuth [1]

gyroscopic tools. Magnetic instruments rely on an inclinometer, compass, timer, and camera for their measurements, while gyroscopic devices are based on the spinning mass principle. To accomodate magnetic instruments, the bottomhole assembly (BHA) must include drill collars made of nonmagnetic materials, such as stainless steel and "monel" metals so that they do not interfere with the surveying instrument. Gyroscopic instruments, on the other hand, do not require nonmagnetic equipment as they are resistant to magnetic interference. The angle between true north and magnetic north is called the declination angle. Measurement locations are called stations, at each station, the measured depth (MD), inclination angle, and azimuth angle are recorded. Depending on the complexity of the well path, measurements are taken at intervals between 30 and 300 feet, or even less if necessary. This data is then used to calculate true vertical depth (TVD), and the geographic positioning in the North-South direction and East-West direction. Using this data it is also possible to calculate dogleg (DL) and dogleg severity (DLS). [1]

Dogleg (DL) is used to describe the curvature or bending of the wellbore typically in degrees. The dogleg (DL) is the angle between the tangents of two wellbore positions. It represents the change in direction of the wellbore between two survey stations. A large dogleg indicates a larger change in direction, while a smaller dogleg indicates a more gradual change. Dogleg severity (DLS) is a measure of the wellbore curvature rate, representing the change in the dogleg per unit length of the wellbore, typically expressed in degrees per 100 feet or degrees per 30 meters. [1]

There are several methods developed to analyse the trajectory. One of the first developed is the Average angle method (AAM). This approach models the well as a series of straight segments in a vertical and horizontal plane. It is assumed that the inclination and azimuth angles are constant and equal to the average value for two subsequent points. [1] Another method is the Radius-of-curvatore method (RCM), this method assumes constant build rates and turn rate along the trajectory. The method that is currently the most established in the oil and gas industry is the minimum curvature method (MCM). This method assumes that two successive points on the trajectory lie on a circular arc located in a plane, resulting in a constant curvature between the successive stations. This angle is called the dogleg (β). [1]

2.3 Casing Operations

2.3.1 Casing Running Techniques

In deviated wells, traditional casing running techniques may prove inadequate due to the increased drag forces encountered within the well. As a result, various specialized methods have been devised to tackle this challenge.

Rotation is one such method that has gained widespread acceptance, not only for use in horizontal wells but also in conventional vertical wells that present complicated conditions or encounter operational issues. The rotational technique can significantly reduce the drag forces by changing the nature of the frictional interaction in the wellbore. Rather than dealing with the static friction that occurs when the casing is stationary, rotation introduces a dynamic frictional component. This dynamic friction is considerably lower than its static counterpart, which allows the available weight of the casing to be more effectively utilized for downward penetration. in the well, offering the additional benefit of mitigating buckling-related problems. Furthermore, equipment designed for casing rotation permits circulation during tripping, assuming no flotation is involved [3] [11].

Alternatively, applying additional pushing force serves as a simple yet effective means of counteracting the drag forces. Such additional weight can be introduced through various approaches. For instance, tapered casing can be employed, whereby the top part of the casing is heavier than the bottom section. Another option involves the use of flotation. Floatation is a technique centered around the principle of buoyancy. By filling the casing with a lighter fluid or even partially with air, the overall weight of the casing string that must be managed is reduced. This effectively provides a degree of 'uplift' or 'float' to the casing, counteracting the force of gravity and subsequently the drag forces experienced during casing running operations.[3]

In practice, the internal volume of the casing string is filled with a gas, such as air, or a fluid that is less dense than the drilling mud. This can be done in sections or across the whole casing string. The resultant buoyant force can significantly alleviate the axial load on the casing string, reducing the risk of buckling, and can also facilitate casing running in challenging sections of the wellbore. [3].

Lastly, reducing the existing side force presents another strategy for diminishing drag force. This can be achieved through the use of lighter pipe materials or by filling the pipe with a fluid that is less dense than the drilling mud, a technique also known as flotation. This is evident when exploring the the equation used in the model: Equation 3.11.

These methods, either singly or in combination, offer promising avenues for overcoming the challenges posed by drag forces in deviated wells.

Buckling

Applying additional pushing force as a technique to overcome drag forces, while effective, is not without its limitations. One notable complication is the potential onset of a phenomenon known as buckling, a condition where the pipe exhibits bending or coiling behavior within the wellbore.

Buckling generally manifests in two stages: sinusoidal and helical. Sinusoidal buckling represents the initial stage, occurring when the pipe is subjected to a

relatively lower compressive load. During this phase, the pipe starts to deviate from its straight configuration, instead adopting a sinusoidal, or wave-like, shape.

As the compressive load continues to increase, the pipe may progress to the second stage, helical buckling. This phase marks a significant loss of axial stability for the pipe. Under the influence of the heightened compression, the pipe can rapidly transition into a twisted, spiral formation, which resembles a helix.

Understanding these buckling phenomena is crucial for mitigating potential complications associated with the application of additional pushing force during casing operations. With this knowledge, operators can design and implement measures to manage these challenges effectively[3].

2.4 Extended reach wells

An extended reach well (ERW) is considered as a well with horizontal departure 2 times (or more) longer than true vertical depth (TVD). [3] Extended reach well's are often used nowadays to reach onshore and offshore oil and gas deposits in order to minimize infrastructure and operational footprints. In 2022 the Abu Dhabi National Oil Company (ADNOC) proudly announced their achievement of drilling the world's longest oil and gas well, located within its Upper Zakum Concession. The well extends an impressive 15240 meters [12]. In the past decades, multilateral-well technology has also matured. Today, technologies surrounding this topic have significantly improved, paving the way for more efficient and effective oil and gas production. Maximizing reservoir recovery is the primary benefit of drilling deviated wells [13]. Horizontal wells and extended reach wells are considered crucial tools for improved oil recovery. Modern horizontal wells have a higher inclination than vertical wells, resulting in higher friction forces. This further emphasises the necessity of a model capable of accurately predicting torque and drag forces along drill string and casing string. A model such as this helps ensure wellbore stability by allowing engineers to plan, design, and execute tripping operations with the appropriate specifications for casing and mud in order to minimize risks. Secondly, by predicting friction forces, operators can optimize drilling parameters such as weight on bit (WOB).

2.5 Casing Selection

2.5.1 Size

The selection of appropriate casing sizes is central to a successful casing running operation. The size of the casing string hinges on two key factors: the necessary inner diameter of the production string, and the number of intermediate casing strings needed to reach the target depth. [1]

To ensure the production casing reaches the intended depth, the drill bit size utilized for the last segment needs to be slightly larger than the outer diameter of the casing connectors. This provision is critical as it enables the mudcake on the borehole wall and casing appliances such as centralizers to be accommodated. It

also provides sufficient clearance beyond the outer diameter of the coupling, thereby allowing for efficient casing operations. [1].

2.5.2 Weight, Grade, and Couplings

Upon determination of the casing's size parameters, the design of the weight, grade, and couplings for each string becomes feasible. As a general principle, every casing string must be robust enough to bear the most intense load conditions anticipated during its installation and throughout the well's life cycle. Predominantly, loading conditions such as burst, collapse, and tension are taken into account. When necessary, conditions such as bending and buckling must also be evaluated. Interestingly, a more cost-effective casing design can often be achieved by integrating various weights, grades, and couplings within a single casing string. When selecting casing, it is crucial to consider the yield strength within a safety factor, which signifies the safety margin between the applied load and design rating. Perhaps the most significant aspect of casing selection is cost-efficiency. The chosen casing design must meet the required design specifications based on the mentioned criteria while being as cost-effective as possible. Typically, the most cost-effective design features the minimum possible weight per meter in the minimum grade [1].

Chapter 3

Mathematical modelling

3.1 Interpolating between survey stations

3.2 Torque and drag modelling

The torque and drag model is the standard drill string model in the oil and gas industry, largely because of its simplicity and widespread accessibility. Despite its general effectiveness, substantiated by field evidence, there have been reported instances of the model under performing. This is particularly evident in its prediction of loads for casing running in horizontal wells, where discrepancies between the model's predictions and actual field data have been observed [2]. Drilling complicated wells require precise planning to decrease friction forces and in order to accomplish set goals. A torque and drag model is often utilized during the well planning stage in order to assess whether the planned route is possible to drill to completion or not. It's also used in order to track torque and drag values in real-time. This makes it possible for the driller to apply preventative measures when the values exceed the permissible safety margins.

Over the past three decades, an array of computer models for the drill string have been developed, with the torque and drag model, initially developed by Dawson and Morehead, being the most prevalent [2]. Within these models, both torque and drag are attributed exclusively to the sliding friction forces resulting from the drill string's contact with the wellbore. The direction of the friction forces is typically unknown, as the drill string or casing can make contact with the wellbore at various points, either in point or continuous contact. The friction coefficient between contacting surfaces and the normal contact force are the two main factors influencing sliding wellbore friction, with the dot product of these two factors yielding the sliding friction force's magnitude [7]. In order to combat this uncertainty the model assumes that the drill string follows the same trajectory as the wellbore, with continuous contact.

In the planning phase of the well construction process, torque and drag models are applied in order to check the proposed well path. Forces along the trajectory are analyzed, which are then used to determine equipment size, and rig capabilities. If it is determined that the proposed well path is not possible to complete without failure (issues such as stuck pipe, too much buckling forces), the trajectory design can be optimized and changed in order to fit the operations required specifications. The Minimum Curvature Method (MCM) is a common technique used to interpolate

the well path between discrete survey points. While this method assumes that the bending moment is discontinuous at these survey stations, it considers the drill string or casing as a "single-force" beam, meaning the force is presumed to be tangent to the well path trajectory. This approach is what characterizes the soft-string model, which assumes a continuous adherence of the drill string or casing to the wellbore trajectory [2].

Although these two concepts share similarities, they serve different roles. MCM is primarily a survey interpolation technique used to map out the well path, whereas the soft-string model is a torque and drag model used to evaluate the forces acting on the drill string or casing. In the work presented here, we have combined these two methods. We employed MCM to interpolate the well path and used a variation of the soft-string model, taking the well path determined by MCM, to calculate torque and drag. This approach optimizes the predictive capabilities of both methods[1].

There are several other torque and drag models that have been developed throughout the years one of them is the stiff-string model, first presented by Ho in 1988, improves on the soft-string model by introducing bending stiffness of the tubular string into consideration [14]. Instead of simplifying the tubular string as a soft rope with zero bending stiffness, the stiff-string model views the string as a thin elastic rod, offering a more realistic representation especially in high dogleg severity wells. This added complexity allows the model to better handle situations where significant bending and contact forces are involved. Despite its increased accuracy under certain conditions, the stiff-string model, much like the soft-string model, still assumes that the drill string or casing maintains continuous contact with the wellbore, which might not always be the case.[15]

In response to the limitations of the static models, a more advanced dynamic torque and drag model has been proposed. Initially introduced by Miska et al. in 2015[16], this model incorporated the effects of velocity and acceleration, representing a significant leap from the static equilibrium approach. However, it assumed a constant axial velocity across the entire tubular string, leading to some discrepancies in the representation of complex mechanical behaviors of the tubular string[15].

The application of torque and drag model's also allow wells to be monitored in real-time. This allows problems to be detected during early stages, and allows for correction in order to prevent a domino effect of smaller issues compounding into major problems. During a real-time monitored operations the driller is able to compare values calculated by the model with the real-time values, if the values match it can indicate good wellbore stability and hole conditions. When values deviate from each other this could indicate a problem downhole [7].

3.2.1 Interpolation between survey stations

The survey data received is from NORCE research's Ullrigg test wells [6]. Interpolating between survey stations as I have done in the code is not strictly necessary in order to calculate drag forces, but is useful data to have available. The code uses interpolated inclinations at every point, instead of interpolating directly in z-component vector formulas. The interpolated inclination is given by:

$$\phi = \arccos\left\{\cos(\phi_1)\cdot\cos\left[\kappa(s-s_1)\right] + \frac{\cos(\phi_2)-\cos(\phi_1)\cdot\cos(\beta)}{\sin(\beta)}\cdot\sin\left[\kappa(s-s_1)\right]\right\} \tag{3.1}$$

In this equation, the variables are defined as follows:

- ϕ: This is the interpolated inclination at any point between two survey stations.
- ϕ_1 and ϕ_2: These represent the inclinations at the first and second survey stations, respectively, providing the start and end inclinations for the segment under consideration.
- θ_1 and θ_2: These represent the azimuths at the first and second survey stations, respectively, providing the start and end azimuths for the segment under consideration.
- s: This represents the measured depth at any point within the span of the two survey stations.
- s_1: This represents the measured depth at the first survey station, serving as the reference point from which the change in depth is calculated.
- β: This represents the dogleg severity or the total angular change in the borehole direction between the two survey stations.
- κ: This represents the wellbore curvature per unit length, obtained by dividing β by the distance between the two survey stations, $s - s_1$.

In the process of interpolating between stations and determining wellbore coordinates, the initial step involves calculating the dogleg angle (β) between the survey stations, and the wellbore curvature κ. The dogleg angle proposed by Sawaryn [17] is the preferred formula in this thesis. The distinct characteristic of this formula is its generality; it does not make assumptions about the shape of the wellbore between the stations. The dogleg β is given by:

$$\beta = 2\cdot\arcsin\left(\sqrt{\left(\sin\left(\frac{\phi_2-\phi_1}{2}\right)\right)^2 + \sin(\phi_1)\cdot\sin(\phi_2)\cdot\left(\sin\left(\frac{\theta_2-\theta_1}{2}\right)\right)^2}\right) \tag{3.2}$$

Wellbore curvature κ is given by[1]:

$$\kappa = \frac{\beta}{s_2 - s_1} \tag{3.3}$$

3.2.2 Torque and Drag modelling in three-dimensional well profiles

The most well known method for analyzing torque and drag was developed by Johancsik [7]. The model developed has the following assumptions:

- The drill string or casing is in continuous contact with the wellbore.
- The drill string or casing is "smooth", the effects of tool joints, couplings, irregularities in the wellbore such as keyseatings are ignored.
- Intertial effects due to pipe sliding or rotation are ignored.
- Flow effects from fluids are ignored.
- The friction factor is modeled using Coulomb friction concept.

The model created in this paper is based on the soft-string sliding-pipe model, which means the model assumes that the pipe behaves as a flexible cable with no bending rigidity or shear forces acting on it. In the case of tripping a casing or drill string, torque equals zero. Therefor only three force equilibrium formulas are necessary [1]:

$$\frac{dF}{ds} + w_{bp}t_z + \vec{w}_d \cdot \vec{t} = 0 \tag{3.4}$$

$$F\kappa + w_{bp}n_z + \vec{w}_c \cdot \vec{n} = 0 \tag{3.5}$$

$$w_{bp}b_z + \vec{w}_c \cdot \vec{b} = 0 \tag{3.6}$$

The term w_{bp} denotes the buoyed weight per unit length of a wellbore segment, taking into account the buoyancy effect of the drilling fluid. The vectors $\vec{t}$, $\vec{n}$, and $\vec{b}$ represent the unit tangential, normal, and binormal vectors, respectively, defining the wellbore's trajectory. Their respective z-components are denoted as t_z, n_z, and b_z.

With these force balances in mind and the assumptions stated earlier it was possible for Johanscik to create a differential formula for calculating axial drag force acting on a segment during tripping operations[1]:

$$\frac{dF}{ds} + w_{bp}t_z \pm \mu\sqrt{(F\kappa + w_{bp}n_z)^2 + (w_{bp}b_z)^2} = 0 \tag{3.7}$$

$$w_c = \sqrt{(F\kappa + w_{bp}n_z)^2 + (w_{bp}b_z)^2} \tag{3.8}$$

$\frac{dF}{ds}$ is the change in force over the change in position, which could be approximated as $dF = F_2 - F_1$ where F_2 and F_1 are the forces at the top of the segment, and the bottom of the segment respectively. Then the equation becomes[1]:

$$\frac{dF}{ds} = -w_{bp}t_z \pm \mu\sqrt{(F\kappa + w_{bp}n_z)^2 + (w_{bp}b_z)^2} \tag{3.9}$$

Solving for the change in force F_2, using $ds = 1$. The equation becomes:

$$F_2 - F_1 + w_{bp}t_z \pm \mu\sqrt{(F\kappa + w_{bp}n_z)^2 + (w_{bp}b_z)^2} = 0 \tag{3.10}$$

Solving for F_2, we obtain:

$$F_2 = F_1 + w_{bp}t_z \pm \mu\sqrt{(F\kappa + w_{bp}n_z)^2 + (w_{bp}b_z)^2} \tag{3.11}$$

In equation (3.11), the $\pm$ sign in front of the term $\mu\sqrt{(F\kappa + w_{bp}n_z)^2 + (w_{bp}b_z)^2}$ denotes the direction of the force being applied. The positive sign (+) corresponds to hoisting while the negative sign (-) corresponds to lowering.

3.2.3 Friction coefficient

The coefficient of friction (μ) is a crucial parameter in wellbore operations, reflecting the ratio of frictional forces that resist the movement between two surfaces in contact to the normal force pressing these surfaces together [18]. During drilling operations, μ is determined under borehole conditions, being a factor that correlates the measured forces at the top of the hole. Assumptions on the friction coefficient generally fall within a range of 0.2 to 0.4, this estimate is influenced by numerous factors including the rock type and properties of the drilling fluid. However, it should be noted that literature sources have reported friction factors as high as 0.8 under certain conditions [7]. This thesis primarily focuses on the dynamic coefficient of friction, which comes into play when the system is already in motion, as opposed to the static coefficient, which is typically larger and pertains to initiating movement. A thorough understanding of these coefficients, their variations, and their impacts are pivotal in the precise planning and execution of drilling operations. [1]

3.2.4 Buoyed weight of segment

The buoyed weight of a segment w_{bp}, is given by[19]:

$$BFw_p \tag{3.12}$$

Solving for buoyancy factor (BF) using the densities of mud ρ_m and steel ρ_s, gives:

$$BF = (1 - \frac{\rho_m}{\rho_s}) \tag{3.13}$$

If there is a disparity between the densities of fluids inside ρ_i, and outside ρ_o of the segment for example after cementing, or during circulation of mud with different mud weight, the buoyancy factor BF becomes:

$$BF = 1 - \frac{\rho_o A_o - \rho_i A_i}{\rho_s (A_o - A_i)} \tag{3.14}$$

Where A_o, and A_i represent the outer, and inner area.

In this model the casing or drill string is assumed to be run into the well, with mud of same density circulating. In the case of a real world scenario, there would certainly be different densities. There are many situations where the mud weight needs to be regulated, just to mention a few:

- drilling cuttings or weighting materials settling
- liquid or gas inflow into the wellbore
- surface fluid dilution

As seen in formula 3.9 a torque and drag analysis typically consists of only three external forces. The force due to the weight of the segment w_{bp} multiplied by the z-component of binormal vector b_z, the force due to friction μ, and the force due to the pipes normal force w_c. It is therefor very important to have the correct unit weight and selecting the correct mud weight. Heavier mud will tend to lower the buoyed weight of a segment, leading to a decrease in pipe normal force w_c, friction and torque. Heavier mud will have more weighting material, which could lead to an increase in friction μ.

3.2.5 Unit vectors

To correctly compute the drag forces along the wellbore trajectory, it is essential to determine the z-components of the normal vector n_z, binormal vector b_z, and tangent vector t_z.

The tangent vector component t_z is typically aligned with the wellbore's tangent

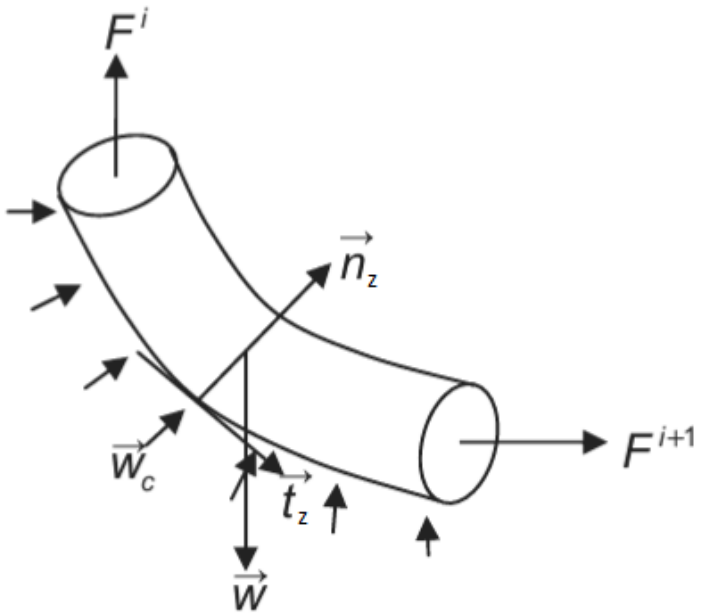

Figure 3.1: Free body diagram of a sliding pipe - soft string model[1]

.

line and varies according to the wellbore's inclination and the measured depth along the wellbore. It is mathematically represented as[1]:

$$t_z = \cos(\phi_1)\cos[\kappa(s - s_1)] + \frac{\cos(\phi_2) - \cos(\phi_1)\cos(\beta)}{\sin(\beta)}\sin[\kappa(s - s_1)] \tag{3.15}$$

The normal vector component n_z usually points in the direction of the local vertical (toward the Earth's center) and is given by[1]:

$$n_z = -\cos(\phi_1)\sin[\kappa(s - s_1)] + \frac{\cos(\phi_2) - \cos(\phi_1)\cos(\beta)}{\sin(\beta)}\cos[\kappa(s - s_1)] \tag{3.16}$$

The binormal vector component b_z typically points in the local horizontal plane, varying with the azimuth. It is computed as[1]:

$$b_z = \sin(\phi_1)\sin(\phi_2)\sin(\theta_2 - \theta_1)/\sin(\beta) \tag{3.17}$$

- ϕ_1 and ϕ_2: These represent the inclinations at the first and second survey stations, respectively, providing the start and end inclinations for the segment under consideration.
- θ_1 and θ_2: These represent the azimuths at the first and second survey stations, respectively, providing the start and end azimuths for the segment under consideration.
- s: This represents the measured depth at any point within the span of the two survey stations.
- s_1: This represents the measured depth at the first survey station, serving as the reference point from which the change in depth is calculated.
- β: This represents the dogleg severity or the total angular change in the borehole direction between the two survey stations.
- κ: This represents the wellbore curvature per unit length, obtained by dividing β by the distance between the two survey stations, $s - s_1$.

These calculations are integral to accurately estimate the distribution of drag forces along the wellbore trajectory.

In the wellbore trajectory analysis, the code incorporates a unique approach where inclinations are interpolated at every meter of the survey station data prior to the implementation of calculations for tz, nz, and bz. This interpolation technique enhances the accuracy and granularity of the data, thereby ensuring a holistic understanding of the wellbore trajectory[1].

The mathematical formulas incorporated in this computational process are:

$$tz = \cos(\phi) \tag{3.18}$$

$$nz = -\frac{\sin(\phi)}{\kappa} \cdot \frac{\Delta\phi}{\Delta s} \tag{3.19}$$

$$bz = \frac{1}{k} \cdot \sin^2(\phi) \cdot \frac{\Delta\theta}{\Delta s} \tag{3.20}$$

Given the pre-calculated interpolated inclinations at each meter of the survey station, the execution of these equations enables the efficient calculation of the z-component vectors. The calculated vectors have a crucial role in providing a comprehensive illustration of the wellbore path, thereby enabling accurate estimation of the drag forces influencing the trajectory.[1]

Through the integration of interpolated inclinations and the aforementioned equations, the code demonstrates an efficient approach to calculating the z-component vectors that form an integral part of drag force estimation. Thus, this methodology significantly contributes to the precision of wellbore trajectory analysis in drilling and completion operations, potentially informing adjustments to improve operational outcomes.

3.2.6 Visual representation

The Ratio Factor (RF) in wellbore trajectory calculations is a geometric factor that relates the incremental changes in the wellbore's inclination and azimuth angles to the corresponding incremental changes in the wellbore's Cartesian coordinates (x, y, z). The RF is specifically used in the application of the Tangential method and the Average Angle method of calculating wellbore trajectory. It's essential for these methods because they apply simplifying assumptions to the curvature of the wellbore path and use these assumptions to approximate the wellbore's spatial trajectory[1].

The Ratio Factor is calculated using the formula:

$$RF = \frac{d}{\Delta\theta} \tan\left(\frac{\Delta\theta}{2}\right) \tag{3.21}$$

where d is the measured depth difference between two points along the wellbore and $\Delta\theta$ is the change in angle (in radians) between these two points[1].

After calculating the Ratio Factor, it is possible to estimate the incremental changes in the Cartesian coordinates of the wellbore trajectory, namely x, y, and z. This is done using the following formulas[1]:

$$\Delta x = (\sin\theta_1 \cos\phi_1 + \sin\theta_2 \cos\phi_2) \cdot RF \tag{3.22}$$

$$\Delta y = (\sin\theta_1 \sin\phi_1 + \sin\theta_2 \sin\phi_2) \cdot RF \tag{3.23}$$

$$\Delta z = (\cos\theta_1 + \cos\theta_2) \cdot RF \tag{3.24}$$

where θ_1 and θ_2 are the initial and final inclination angles of the wellbore segment, ϕ_1 and ϕ_2 are the initial and final azimuth angles, and RF is the Ratio Factor calculated earlier. These calculated increments Δx, Δy, and Δz are then used to update the wellbore's position[1].

Consequently, visualization of the wellbore's trajectory in 3D space provides an invaluable tool for understanding the physical behavior of the casing or drill string. The provided code snippet enables the creation of a 3D plot that accurately displays the changes in inclination and azimuth angles along the wellbore's path. By coupling the derived equations for axial drag force with these visual aids, the implemented solution allows for comprehensive exploration and interpretation of drilling operations.

3.3 Minimum Curvature Method

The minimum curvature method is a widely accepted industry standard used for modelling and planning of wellbore trajectory. It is a mathematical method used to estimate the path of a wellbore between two survey stations. The essence of this method is that two successive points on the trajectory are assumed to lie on a circular arc, located in a plane. The arc is the smoothest possible curve connecting the two points. The two survey points define the start and end point of the arc, and the curvature (Dogleg angle) between the points on the segment is constant. The method then uses the properties of the arc, its length, curvature, and the directions at the end points to estimate the spatial coordinates of points along the trajectory.

This method as mentioned widely accepted and used in the oil and gas industry. However it is important to note that while the minimum curvature method is a powerful tool, it is a generalization, and uses certain assumptions as discussed earlier. Like any model, it might not fully represent the actual wellbore path in all situations. [17] As illustrated in Figure 3.2, points 1, and 2 reside on the same plane, with a constant curvature between points 1 and 2. The inclination and azimuth are denoted as ϕ and θ respectively. The measured depth between points 1 and 2 is denoted as Δs, while the radius of the circular arc connecting these points is represented by R. The angle β is termed as the dogleg [1].

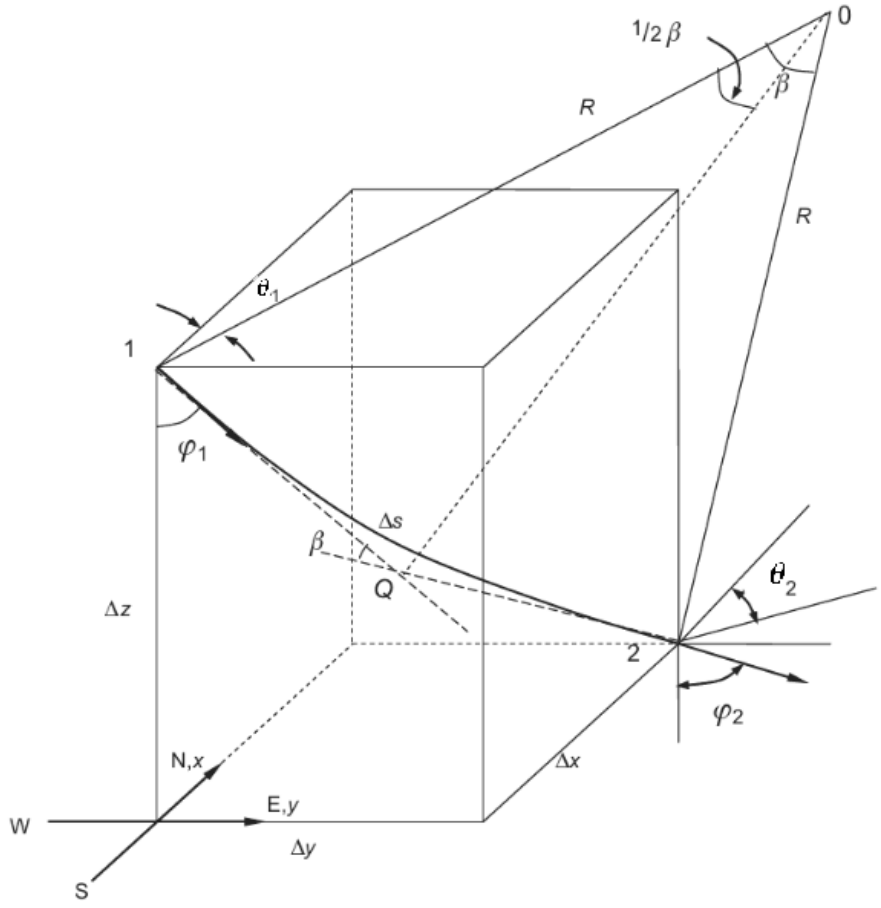

Figure 3.2: Schematic diagram of a wellbore segment for the minimum curvature method[1]

.

Chapter 4

Methodology

4.1 Python

Python is a widely used, high-level, and versatile programming language that has gained immense popularity due to its efficiency, readability, and extensive library support. In this project, Python was used to create a predictive model for estimating friction forces on casing during well operations. Python was chosen as the appropriate programming language for this project because of its ease of use, and rich ecosystem of scientific computing libraries.

4.1.1 Libraries and Packages

To streamline the development process and leverage existing tools, a variety of Python libraries are utilized in the project. The libraries used were NumPy, Pandas, matplotlib, math, Welly, Lasio, and prettytable.

math

The math library is a standard Python module that provides mathematical functions and operations. "Math" includes functions for trigonometry, logarithmic and exponential operations, and other common mathematical calculations. Math was used to perform trigonometric calculations when determining dogleg severity, wellbore trajectory coordinates, and torque and drag. Functions such as "math.radians()", "math.sin()", and "math.cos()" are used to convert angles to radians and to perform sine and cosine operations, which are essential in calculating the wellbore trajectory as well as torque and drag forces acting on the casing. One important thing to note is that Python calculated trigonometric functions in radians by default, hence the importance of using the proper built-in math functions to convert from radians to degrees, and degrees to radians where necessary. [20]

matplotlib

"matplotlib" is a plotting library that allows for the creation of a wide variety of static, animated, and interactive visualizations. "matplotlib" was used to visualize the wellbore trajectory, and certain qualities of the calculated coordinates. "matplotlib" and "mpl_toolkits.mplot3d" are used together to generate 2D and 3D plots of the wellbore trajectory. The library offers a high level of customization which

allowed customization of the plot to create a clear and informative visual representation of the wellbore data. The various plots shown in **??** provide insight into the wellbore's trajectory, lateral, and vertical movements. [21]

numpy

"Numpy" (Numerical Python) is a fundamental library for scientific computing in Python. It provides support for multi-dimensional arrays and matrices, as well as a collection of mathematical functions to operate these data structures. In this project "numpy" is used to store and manipulate the calculated X,Y, and Z coordinates of the wellbore trajectory points. By utilizing "numpy", the script can efficiently perform calculations required for generating the wellbore trajectory [22]

Welly

The Welly package is a Python package developed for managing well data. Welly offers a range of features to handle well log data, including loading, processing, and visualizing data. One of the key capabilities is the ability to load well data from LAS files. Additionally the package provides various functions for processing well log data, including computing petrophysical properties, generating synthetic seismograms, and correcting for missing or invalid data. Lastly, the package offers ways to visualize well log data in a clear and organized way. Overall the package provides a comprehensive set of tools to work with well data in Python. [23]

Lasio

Lasio is another package that was used in the code, Lasio is a package for Python 3.7 and above that allow the user to read and write Log ASCII Standard (LAS) files, which are commonly used in the oil and gas industry. LAS files are used for storing borehole data for geological, geophysical, and petrophysical logs. The primary purpose of the Lasio package is to facilitate the reading and writing of both data and metadata in LAS files. The package is designed to handle as many LAS files as possible, even those with errors or non-standard formatting. While the package can be used directly, there are other packages available that may be better suited to specific needs. In this project Lasio is used to generate, and write a LAS file, which is used to store data for the next steps in the code. [24]

prettytable

"Prettytable" is a library for creating simple and visually appealing tables. It allows users to generate formatted table with various styles and alignments, making it easier to display data in a clear and organized manner. In this project "prettytable" is used to display the wellbore trajectory points in a formatted table, including MD, inclination, azimuth, x, y, and z coordinates. The table provides an accessible overview of the data, and makes it easier to analyze said data.[25]

4.2 Code Description and Operation

This section provides a comprehensive description of the operational steps embedded in the code and the underlying calculations.

Data Preprocessing and Unit Conversion

The algorithm begins by converting all units to radians to ensure consistency across the board. This is crucial for mathematical operations that follow. Survey data is manually filled in and used as input.

Distance Calculation and Dogleg Function

Next, the code constructs a list to store the distances between survey stations (denoted as s-s1), which are fundamental for subsequent calculations. The Dogleg function is executed on the survey data, and the resulting dogleg values are also stored in a separate list for further computation.

Wellbore Curvature and Ratio Factor

Utilizing the derived dogleg values and distance list, the wellbore curvature, denoted by κ, is calculated and similarly stored in a list. Following this, the Ratio Factor is determined using equation 3.20.

Wellbore Trajectory Coordinates

With these variables now readily available in lists, the code is able to compute the coordinates of the wellbore trajectory using equations 3.21, 3.22, and 3.23. These coordinates are then preserved in an array, enabling the construction of a 3D wellbore trajectory plot. In the next part of this phase, a LAS file is initiated using the LASFile function from the lasio library. This file serves as a standardized format for storing and sharing well log data.

Adding Curves to LAS File

Subsequently, curves are added to the LAS file. These curves represent different data variables, each with a designated unit of measurement and a brief description. They include: Measured Depth (DEPT), Inclination (INCL), Azimuth (AZIM), and X, Y, Z offsets. The data for these curves is taken from the array of calculated coordinates.

Writing and Loading LAS File

The populated LAS file is then saved in a designated location on the local storage, and immediately loaded back into the program. This process effectively saves all the data and calculation results in a LAS file, which can be accessed and manipulated later as needed.

3D Plot Construction

Following the data storage, the algorithm proceeds to visualize the wellbore trajectory through 3D plotting. A 3D plot is generated, illustrating the X, Y, Z offsets, which represent the well path. The start and end points of the well path are marked distinctly. The plot's labels and scales are set appropriately.

Additional Plots

In addition to the 3D plot, the code also generates three 2D plots to visualize the relationships between different dimensions: X vs Y locations, X location vs TVD, and Y location vs TVD. Each plot provides a different perspective on the wellbore trajectory.

Data Interpolation and Storage

The code employs a segment length of s-s1 = 1 meter for the upcoming computations as opposed to using the distance between survey stations. This approach ensures accurate calculations and yields more detailed data points that may be utilized in later operations. A Python function is devised to interpolate the survey data for every meter of the total measured depth (MD) and saves this in an array of length MD.

True Vertical Depth

To enhance the clarity of the data, the algorithm computes the True Vertical Depth (TVD) using equation 3.23 and incorporates it into the array. Additionally, equations 3.17, 3.18, and 3.19 are applied to the interpolated inclinations to calculate tz, nz, and bz respectively. These values are subsequently stored in the array.

Drag Force Calculation

Finally, the algorithm calculates the drag forces along the trajectory using equation 3.10. It's crucial to note that these calculations start from the bottom of the casing or drill string, given that the force at the bottom is known to be zero. The code works its way upwards, incrementally by 1 meter, accumulating drag force along the trajectory. The calculated forces are then saved in an array, which provides the drag force at every depth, thereby offering valuable insights into wellbore conditions.

Creation of Well Plots

Finally, the algorithm facilitates the creation of well plots. These plots visualize the calculated forces, allowing the user to observe the changes in drag force along the wellbore trajectory. Notably, the code provides the flexibility to generate these plots with different buoyed weight of segment (w_{bp}) and friction factors. Varying these parameters can lead to different output, providing a way to observe the influence of these variables on the drag force. This flexibility could be crucial for optimizing drilling operations, by providing an easy means to understand how changes in these variables could affect the forces encountered in the wellbore.

Chapter 5

Results and Discussion

This chapter embarks on an exploration of practical case studies of various wells, with data obtained from Ullrigg Test Centre. Ullrigg Test Centre, managed by NORCE Norwegian Research Centre AS, is a prominent facility that provides a comprehensive archive of test wells [6]. The data from these wells offer valuable real-world insights into the multifaceted nature of wellbore trajectory planning, design, and management. These case studies serve as the embodiment of the theoretical principles and computational methodology outlined in the preceding chapters. Specifically, we explore variations in the friction coefficient, spanning the standard range of 0.2 to 0.4. By analyzing these cases, I am aiming to validate the developed algorithm and demonstrate its effectiveness in varied real-world scenarios, while highlighting any potential areas of enhancement.

A key focus of these case studies is an exploration of the impact of variations in the friction coefficient. The analysis encompasses the standard range of 0.2 to 0.4, and further delves into more extreme cases to test the robustness of the model. The overarching objective of these case studies is not only to substantiate the algorithm developed in this study but also to illustrate its versatility across different real-world scenarios. Furthermore, these studies offer an opportunity to identify potential areas for model refinement and enhancement.

> *Disclaimer: The well paths represented in the figures of this thesis is generated from a 3D model in Python. It is essential to note that the interactive nature of the model, including the capability to manipulate and observe different perspectives of the well path, is not fully captured by the static screenshots provided here. For a comprehensive understanding of the well path dynamics, it is recommended to engage with the actual interactive 3D model.*

5.1 Well U1

In this study, we turn our attention to Well U1, a wellbore with a total measured depth (MD) of 1324 m. This well is structurally characterized by a casing shoe set at 354 m MD. The casing is designed as a vertical 13 3/8" section, which boasts a nominal weight (w_{bp}) of 72 lb/ft, as specified by the Ullrig Test Centre [6]. This weight is a crucial factor, as it significantly affects the overall stability and stress resistance of the well structure.

Given the sensitivity of well performance to friction coefficients, we opted to run simulations that encompassed a range of plausible scenarios. These included the industry-standard friction coefficients range of 0.2 - 0.4, as well as an elevated coefficient of 0.8 to explore the impact of extreme conditions on well integrity and operational efficiency. Furthermore, we investigated the influence of varying the casing's nominal weight w_{bp} by comparing outcomes for different nominal weights of 13 3/8" casing, 72, 68, and 61 lb/ft. For the sake of consistency across all parameters and calculations, the casing weight was converted to metric units, resulting in 107.2, 101.2, and 90.8 kg/m, respectively. It should be noted that these conversions were rounded to the nearest tenth for simplicity and clarity.

The intent, through the simulation of these diverse scenarios, is to provide a more holistic understanding of the dynamics within the well environment. By doing so, we aim to delineate the parameters that have the greatest influence on well performance and stability, thereby informing more effective design and operational strategies for future well construction and management.

5.1.1 Results Well U1

The table 5.1 showcases Well U1's survey data. This dataset, acquired from Ullrigg Test Centre [6], delineates the along hole depth (AHD), inclination, and azimuth of the well at various points. Through a detailed analysis of this data, we continue to assess the performance and validity of the developed model.

The model-generated well path for Well U1 is illustrated in Figure 5.1. This visual representation encompasses multiple interrelated plots, illustrating the X-axis and Y-axis locations against TVD (true vertical depth), as well as their comparative positioning. This ensemble of graphics not only provides a comprehensive spatial perspective of the well trajectory but also serves as a critical foundation for subsequent analyses.

AHD (M)	INC (DEG)	AZ (DEG)
0.0	0.0	0.0
50.0	1.5	143.7
100.0	4.4	154.2
150.0	6.3	153.3
200.0	7.0	150.7
250.0	5.5	151.0
300.0	5.2	155.9
350.0	4.7	175.7
400.0	1.7	175.9
450.0	1.3	173.6
500.0	1.4	-175.0
550.0	3.6	-170.4
600.0	4.5	-149.9
650.0	2.9	-132.9
700.0	1.1	-131.8
750.0	0.2	236.5
800.0	1.7	-127.8
850.0	1.8	-112.5
900.0	1.4	-47.5
950.0	2.4	50.5

Table 5.1: Survey Data for Well U1: MD (m), Inclination (degrees), Azimuth (degrees)

The investigation into Well U1 continues with an analysis of the influence of varying friction coefficients, as given a nominal casing weight, w_{bp}, of 90.8 kg/m. This analysis illuminates an evident correlation between the friction coefficient (μ) and the discrepancy between forces during tripping in and out operations.

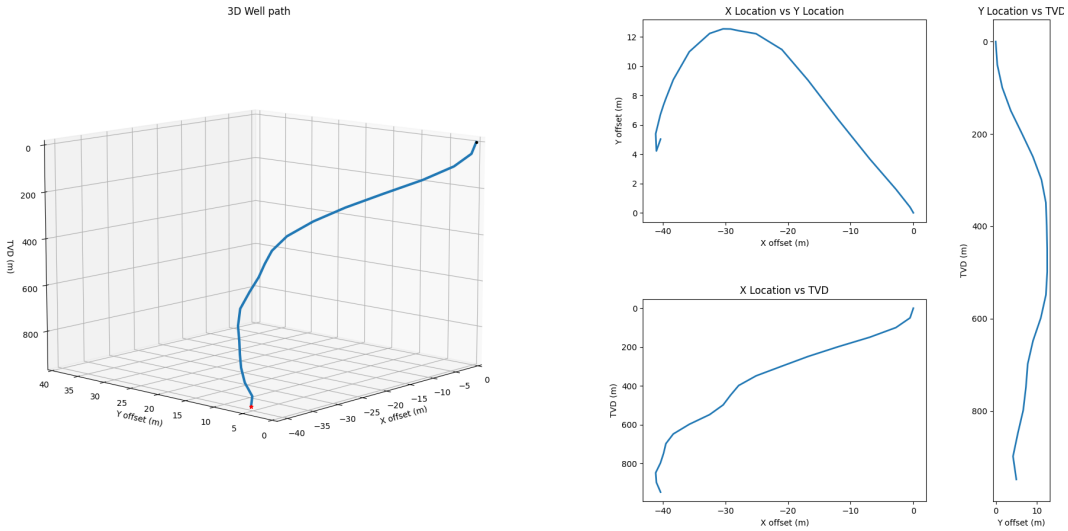

Figure 5.1: Model-generated well path for Well U1, featuring X-axis and Y-axis locations plotted against TVD.

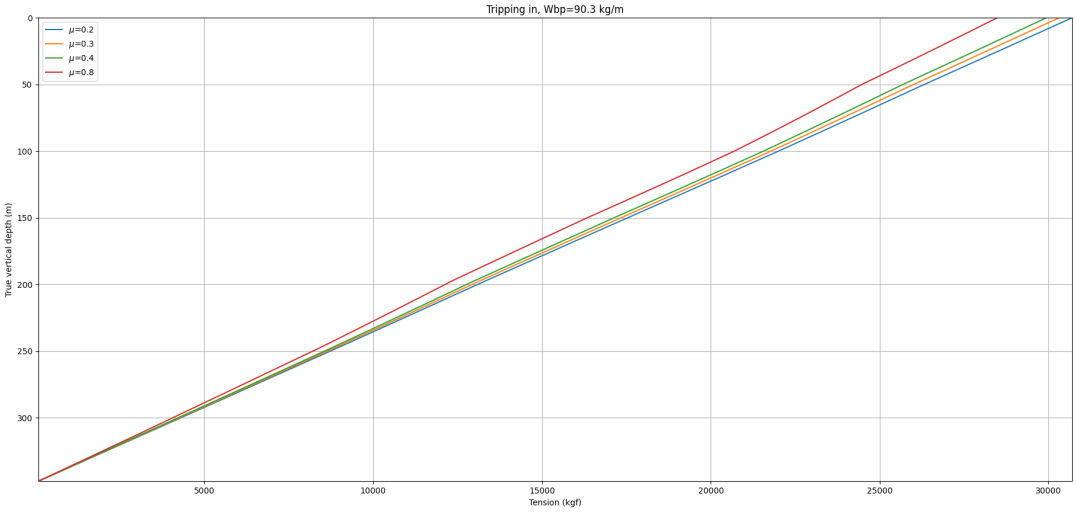

Figure 5.2: Impact of varying friction coefficients on drag forces for w_{bp} of 90.8 kg/m, tripping in, Well U1

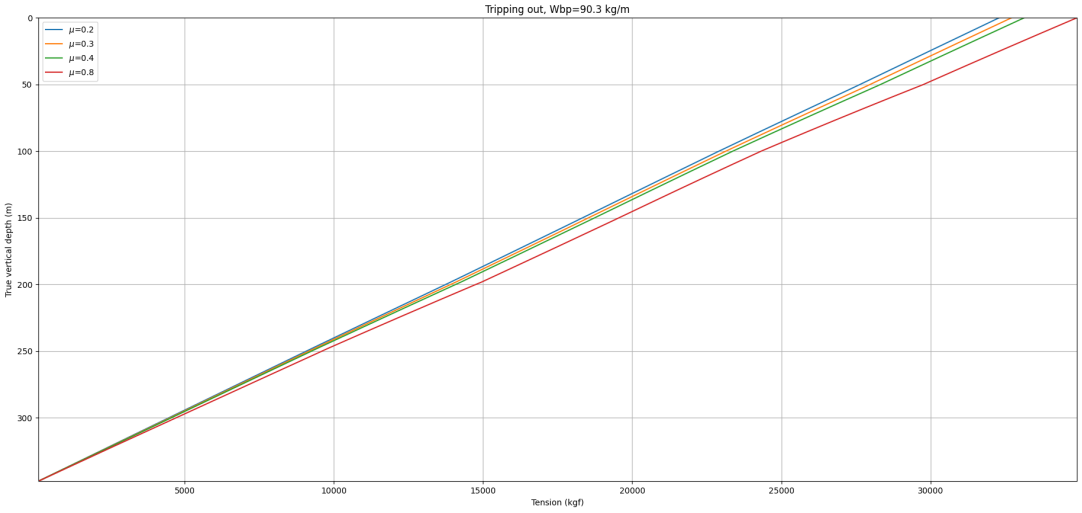

Figure 5.3: Impact of varying friction coefficients on drag forces for w_{bp} of 90.8 kg/m, tripping out, Well U1

This behavior can be rationalized by taking a look at Equation 3.10:

$$F_2 = F_1 + w_{bp}t_z \pm \mu\sqrt{(F\kappa + w_{bp}n_z)^2 + (w_{bp}b_z)^2} \tag{5.1}$$

which quantifies the frictional force experienced by the casing during well operations. As μ escalates, the overall contribution of the frictional force component is amplified, leading to an increased disparity between the forces experienced during tripping in and out operations. It is crucial to note that a positive sign in this equation corresponds to hoisting (tripping out) operations, whereas a negative sign refers to lowering (tripping in) operations. The visual representation in Figure **??** substantiates this, revealing higher total force at MD = 0 during hoisting.

Moreover, it is worth noting that Well U1 exhibits a predominantly vertical trajectory, as evidenced by the survey data (Table 5.1) and figure 5.1. The inclination values are mostly below 10 degrees, indicating a significantly vertical well path. As we proceed with the analysis of wells featuring higher inclinations, the influence of inclination on wellbore forces, in tandem with friction coefficients, will be further elaborated upon.

5.2 Well U2

We now direct our attention to Well U2, which presents a different set of conditions. This well extends to a total depth of 2020 m MD, with a 13 3/8" casing shoe positioned at 1520 m MD. The well employs a 13 3/8" casing with a nominal weight (w_{bp}) of 72 lb/ft, as dictated by the specifications provided by Ullrig Test Centre [6]. Notably, Well U2 features a tangential section with an inclination ranging from 60 to 63 degrees, encountered at approximately 1650 m along hole depth (AHD).

Mirroring the analysis carried out for Well U1, the model for Well U2 is also tested with varied nominal casing weights, specifically, 72, 68, and 61 lb/ft These weights translate to 107.2, 101.2, and 90.8 kg/m, respectively, providing further scenarios to test the flexibility and accuracy of the computational model.

5.2.1 Results Well U2

For Well U2, we have a markedly different well trajectory, with a higher degree of horizontal displacement compared to Well U1. The survey data provided for Well U2, illustrated in Table 5.2, reveals this distinction. While the well begins in a similar vertical orientation as U1, the inclination values gradually increase, indicating the well's transition towards a more horizontal trajectory. The Azimuth (Az) values also exhibit considerable variation, reflecting the well's directional change. As we proceed to analyze the impact of varying friction coefficients on Well U2, the influence of these horizontal deviations will become more apparent.

AHD (M)	INC (DEG)	AZ (DEG)
0.0	0.0	0.0
50.0	0.9	-29.7
100.0	0.4	-95.1
150.0	0.7	142.5
200.0	0.9	58.2
250.0	0.6	173.2
300.0	1.3	143.6
350.0	6.6	155.6
400.0	11.2	143.7
450.0	12.7	145.2
500.0	12.0	150.8
550.0	12.5	154.7
600.0	12.2	158.4
650.0	13.1	159.0
700.0	14.6	153.9
750.0	16.2	152.2
800.0	16.4	146.1
850.0	20.7	138.3
900.0	24.6	129.8
950.0	27.8	123.2
1000.0	31.0	119.2
1050.0	32.8	115.6
1100.0	36.5	113.7
1150.0	38.5	109.1
1200.0	40.3	106.5
1250.0	43.0	104.0
1300.0	46.2	104.0
1350.0	50.5	100.6
1400.0	52.3	102.4
1450.0	55.1	100.5
1500.0	58.6	100.2
1550.0	61.8	90.6
1600.0	61.9	88.8
1650.0	63.4	88.4
1700.0	60.1	91.7
1750.0	57.8	93.5
1800.0	57.6	98.9
1850.0	58.2	102.7
1900.0	59.9	105.8
1950.0	60.8	110.5
2000.0	61.3	116.6

Table 5.2: Survey Data for Well U2: AHD (m), Inclination (degrees), Azimuth (degrees)

The planning and implementation of well trajectories have a profound effect on the efficiency and success of a drilling operation. To further elucidate this, we

present the model-generated well path for Well U2. In Figure 5.4, the well path's X-axis and Y-axis locations are plotted against True Vertical Depth (TVD). The illustration provides a visual representation of the drilling trajectory, which offers crucial insights into the geometric and spatial path of Well U2. Understanding the intricacies of this path aids in foreseeing any potential challenges and opportunities during drilling and completion operations.

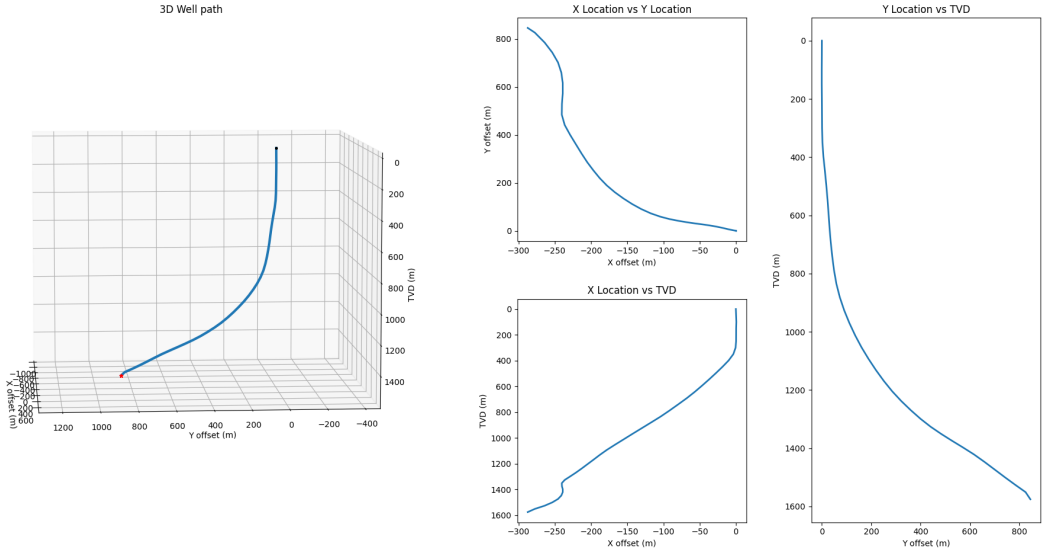

Figure 5.4: Model-generated well path for Well U2, featuring X-axis and Y-axis locations plotted against TVD.

As observed in Figure 5.4, the well path of Well U2 significantly differs from that of Well U1, particularly due to a substantial horizontal section. Scrutiny of the X location versus TVD plot discloses a marked change in the X offset at around a TVD of 300m. Furthermore, the Y location versus TVD plot shows a substantial increase in the Y offset at a TVD close to 800m, marking the commencement of a significantly deviated section.

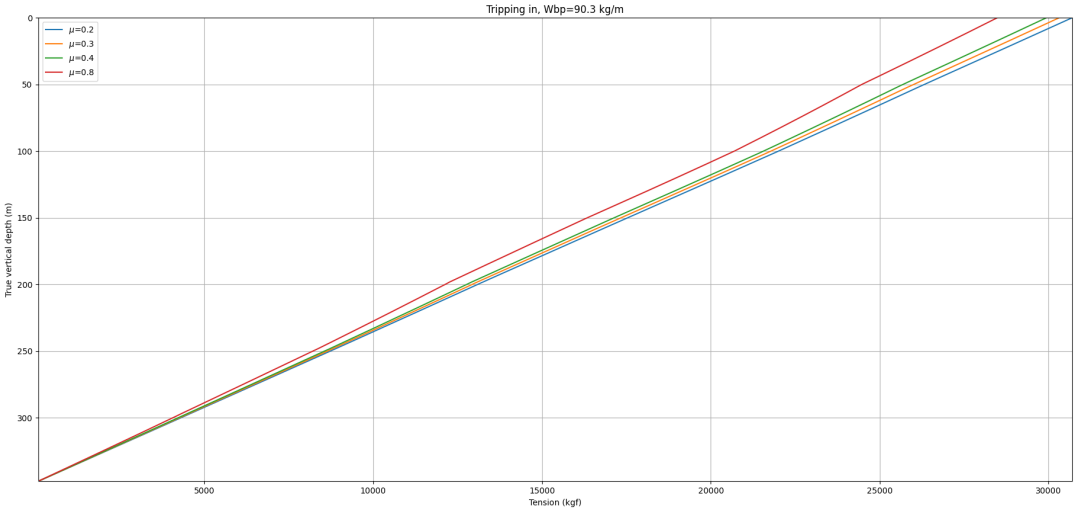

Figure 5.5: Impact of varying friction coefficients on drag forces for w_{bp} of 90.8 $\frac{kg}{m}$ Well U2

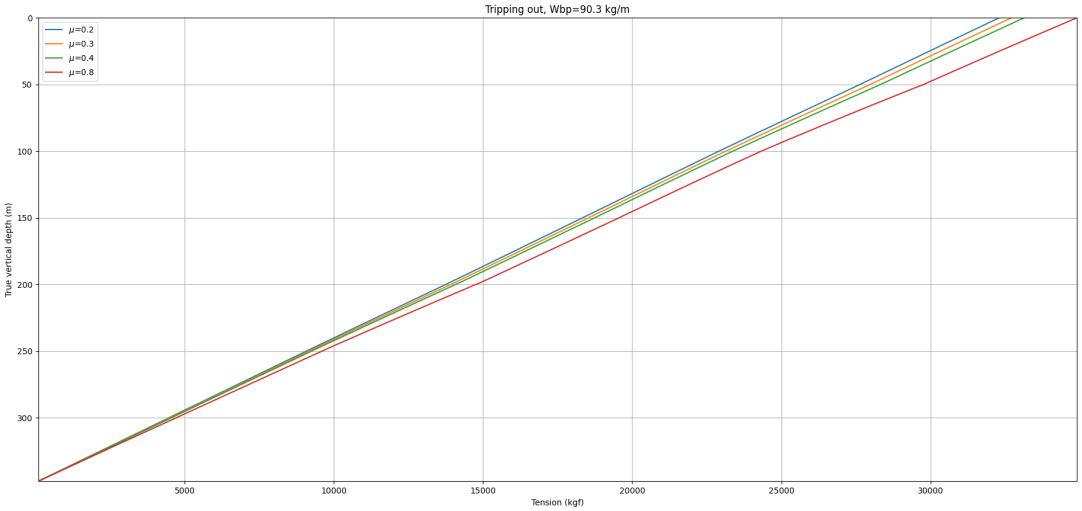

Figure 5.6: Impact of varying friction coefficients on drag forces for w_{bp} of 90.8 $\frac{kg}{m}$ Well U2

As shown in Figures 5.6, 5.5, there is a notable increase in drag forces around these depths. This is even more apparent when looking at the plots with a higher friction coefficient μ. As the normal force, an integral component of equation 3.11, escalates this leads to a widening disparity between hoisting and lowering forces. This suggests that the more deviated sections of Well U2's trajectory might introduce additional challenges during drilling operations.

5.3 Well U6B

The final well we delve into is Well U6B, a well that reaches 738 m AHD and presents unique features, including a high dogleg severity of up to 14 degrees and a substantial horizontal section. The horizontal section of the well commences at approximately 439 m AHD and extends all the way to the total depth (TD) of 768 m AHD.

According to Ullrig Test Centre's specifications, the well is furnished with two casing shoes [6]: a 10 3/4" casing shoe with a weight of 60.7 $\frac{lb}{ft}$ of P-110 grade, situated at 461 m MD, and a 13 3/8" casing shoe with a weight of 72 $\frac{lb}{ft}$ of N-80 grade, positioned at 37.5 m MD.

To further examine the model's versatility and performance under varying conditions, we explore scenarios with different nominal casing weights, specifically 79.2, 60.7, and 45.5 $\frac{lb}{ft}$. These values correspond to casing weights of 117.8, 90.3, and 67.7 $\frac{kg}{m}$, respectively. The evaluation of Well U6B rounds off our comprehensive study, providing a broader perspective on the algorithm's applicability across varied well architectures and conditions.

5.3.1 Results U6B

The third and final well in this case study is Well U6B, which exhibits unique features in comparison to Wells U1 and U2. Notably, Well U6B contains a substantial horizontal section, much like Well U2, but with an even more pronounced dogleg severity, as evidenced in the survey data presented in Table 5.3.

Well U6B reaches an Along Hole Depth (AHD) of 738 meters while maintaining a relatively low True Vertical Depth (TVD) of 355 meters. This disparity underscores the presence of an extensive horizontal section, a crucial aspect of the well trajectory. A three-dimensional visualization of the well path, as shown in Figure 5.7, further elucidates the well trajectory's dynamic nature.

The X and Y offsets seen in the well path show consistent growth, beginning at approximately 200 meters and extending up to the range of 300-350 meters. A rapid surge in both X and Y offsets relative to TVD within this depth interval heralds the commencement of the horizontal section.

Another salient feature of Well U6B is the notable discrepancy observed when tripping in and out in conditions of increased friction coefficient. This is presented in Figures 5.9,and 5.8, where it's evident that an increase in inclination aligns with an escalation in the friction coefficient. This observation aligns well with the theoretical expectations and further accentuates the complexity of the operations in wells with

AHD (M)	INC (DEG)	AZI (DEG)
20	0.26	216.47
55	0.44	294.46
96	0.97	18.16
132	2.38	36.66
180	4.62	147.32
200	7.12	133.81
219	11.97	131.44
228	15.04	132.99
237	17.37	133.74
266	28.38	134.15
287	36.69	134.09
324	49.7	131.18
349	59.37	130.98
381	68.31	132.91
410	75.59	133.73
439	85.98	135.58
476	86.94	133.76
490	86.68	133.29
533	88.05	133.67
572	89.35	134.23
598	89.72	135.39
628	90.34	136.57
738	90.52	135.64

Table 5.3: Survey Data for Well U6B: AHD (m), Inclination (degrees), Azimuth (degrees)

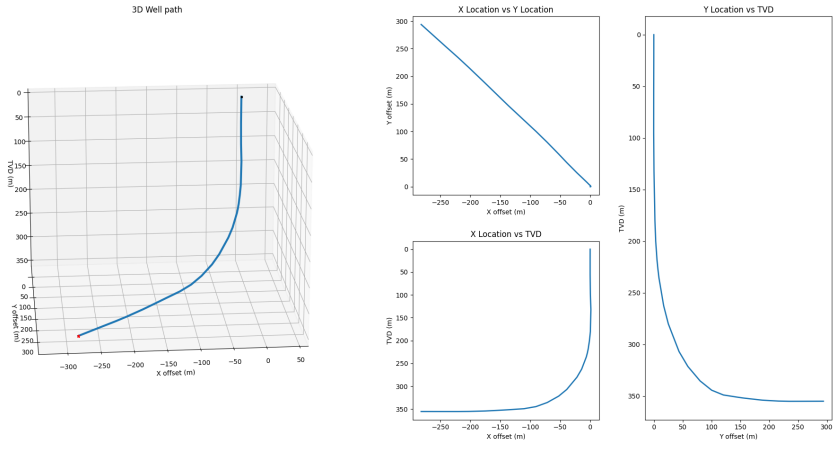

Figure 5.7: 3D well path for Well U6B

significant horizontal extensions.

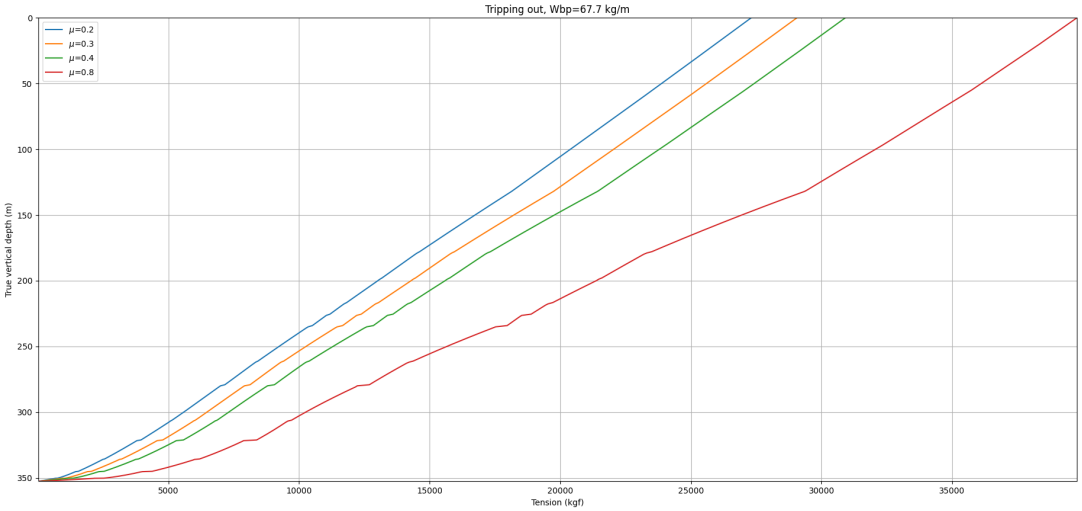

Figure 5.8: Impact of varying friction coefficients on drag forces for w_{bp} of 67.7 kg/m, tripping out, Well U6B

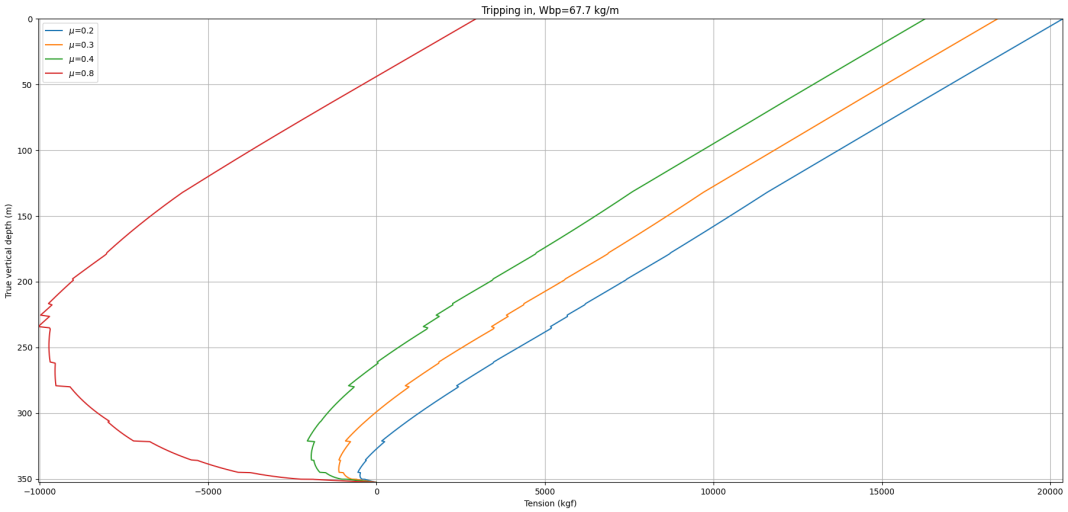

Figure 5.9: Impact of varying friction coefficients on drag forces for w_{bp} of 67.7 kg/m, tripping in, Well U6B

5.4 Discussion

A selection of plots are included in the main body of this document to illustrate and support our key findings. However, in order to provide a more comprehensive view of the analysis, the rest of the generated plots from the case study is also included in the Appendix. Refer to the appendix for a more detailed understanding of the case studies and results.

Analysis of the results obtained from the distinct well trajectories in U1, U2, and U6B has shed light on the influence of different well trajectories on the frictional forces encountered during tripping operations. Well U1, characterized by a predominantly vertical drilling path, contrasted significantly with U2 and U6B, both of which exhibited deviated trajectories with pronounced horizontal sections. Interpretation of these observations indicate a trend wherein deviated wells (specifically, U2 and U6B) tend to be associated with heightened frictional forces during tripping operations. This relationship becomes particularly pronounced at times of increased inclination, thus underscoring a clear link between well trajectory and the operational challenges arising during drilling and completion operations. A noteworthy instance of this relationship is seen in well U6B, which, given its distinct dogleg and extensive horizontal section, presented a marked divergence between the forces experienced during tripping in and out. This divergence was amplified when the friction coefficient increased, thereby expanding the gap between tripping in, and out plotted on the graph and thereby illustrating the significant influence of such horizontal sections. [6]

These findings not only validate our initial hypotheses but also highlight the considerable influence of well path geometry – especially variations in inclination and azimuth – on the magnitude of frictional forces during well operations. Such insights contribute significantly to our understanding of drilling dynamics, emphasizing the crucial role of well path design in either mitigating or exacerbating operational challenges. Moreover, these results underscore the importance of employing predictive models, such as the one developed in this thesis, in the planning, design, and execution of field operations. By incorporating these insights into future well planning and execution, it is hoped that drilling and completions operations can be optimized to minimize operational difficulties and maximize overall efficiency and safety.

While the model developed in this study presents a robust and accurate representation, it is important to acknowledge certain assumptions and limitations that may influence its output. Initially, the model is built upon the principles of minimum curvature and the soft-string pipe model as detailed in Section 3.2.2. Consequently, the model may not provide entirely accurate results in all scenarios due to the inherent assumptions of these methods. Variables influencing the drag forces during drilling and completion operations might not be fully accounted for, even though these forces primarily originate from the sliding friction between the drillstring/casing and the wellbore wall [7].

In relation to these assumptions, it is crucial to recognize that the minimum curvature method is an estimated parametrization of the wellbore path, and a real drillstring may deviate from this path due to its own tendency to follow a different curve. The model does not account for tortuosity or unevenness in the wellbore path

resulting from the drilling process or drillstring steering. Moreover, the assumptions within the torque and drag model, including soft string behavior and continuous contact with the wellbore, might not always hold true.

A related point to consider is the representation of curvature in the model. The minimum curvature method leads to a constant curvature between two survey stations, but this curvature changes abruptly as one traverses the survey list[1]. This sudden change, rather than a smooth, continuous variation with depth, reflects the limitations of the minimum curvature method and the idealization of the wellbore path, rather than a true representation of physical reality.

Secondly, although this model has been extensively tested and validated through problems outlined in 'Fundamentals of Drilling Engineering' [1], there has been no verification with official data from the Ullrig Test Centre. This is due to the unavailability of such data and represents a potential limitation in terms of the model's validity.

Lastly, despite thorough testing, potential errors in the code used to build this model may exist, thereby possibly affecting the output. Coding this model proved to be the most challenging aspect of the project and any overlooked errors could potentially impact its performance. The code, however, is readily available in the Appendix for transparency and further validation.

In spite of these limitations, the model proves to be a valuable tool for understanding the impact of well path geometries on the frictional forces during drilling operations. It reinforces the need for such models in the planning, design, and execution of drilling operations in order to mitigate potential operational challenges.

5.4.1 Future Work

In light of the findings and limitations of the present study, several directions for future work emerge that could elevate the model's performance and applicability.

To start with, a future iteration of this model could encompass a wider array of physical parameters. Although the current model has made significant strides, there remains an opportunity for further development. Variables such as different drilling fluids, temperature and pressure variations along the wellbore, tool wear and tear, and geological considerations could be incorporated to increase the model's fidelity.

Moreover, validation of this model with real-world data from drilling operations can significantly enhance its credibility. Despite rigorous testing with problems from drilling engineering literature, a comparison with actual operational data could reveal additional nuances that improve the model's predictive power.

Investigation into the impact of various drilling technologies on frictional forces also presents a promising direction for future research. Drilling techniques and tools, such as rotary steerable systems (RSS), could have an influence on frictional forces that is not currently captured by the model.

Furthermore, a comprehensive sensitivity analysis would identify which parameters most significantly impact the prediction of frictional forces. This could guide future improvements to the model and help prioritize further research efforts.

In addition, the current model's user interface could be improved for practical use. As it stands, the model requires manual data input and occasional adjustments, which could be streamlined with a more automated and user-friendly interface.

Lastly, integrating the model into broader well construction software would also

be beneficial. The implementation of a model such as the one developed in this study, specifically tailored for frictional force calculations, could provide additional value. Future iterations could include features for buckling calculations and incorporate variables like geological factors and build-up or drop-off points.

Through these enhancements and expansions, the model could potentially serve as a valuable tool in the oil and gas industry, aiding in the planning, design, and execution of drilling operations.

Chapter 6

Conclusion

This thesis presented a computational model to evaluate the impact of well path trajectory on frictional forces during well operations, with a specific focus on the process of tripping in and out. The model, based on the minimum curvature method and the soft-string model, has provided valuable insights into the dynamics of drilling and the challenges associated with various well trajectories.

The analysis of three different well paths - U1, U2, and U6B - revealed distinct differences in frictional forces due to variations in their respective trajectories. Specifically, wells with strong horizontal sections (U2 and U6B) demonstrated a clear increase in frictional forces during operations, particularly during instances of increased inclination. These findings validate the accuracy and usefulness of our model for predicting operational challenges during well drilling and completion.

Understanding the relationship between well path design and operational frictional forces can aid in the planning, design, and execution of efficient drilling operations. This study contributes to the field of drilling engineering by offering a tool to aid in such an understanding.

Despite the insightful findings, the present study has limitations. The model operates under certain assumptions, and while it captures the main dynamics of friction in well operations, there are variables not accounted for in the current version. Future work would benefit from incorporating these factors to enhance the model's accuracy and applicability in real-world scenarios.

In closing, the knowledge and insights gained from this study open up promising avenues for future research, particularly in refining the model and validating it with operational data from drilling sites. This could provide even more accurate and insightful predictions, leading to safer and more efficient well operations.

In order to provide a detailed insight into the computational process underlying my model, the complete code utilized in this study has been included in the Appendix. Readers who are interested in the specific coding details, or who wish to further explore or replicate the work, are encouraged to refer to this section. The reader is also encouraged to explore the additional plots provided in the Appendix for a more comprehensive understanding of the case studies and insights gathered in this study.

.1 Appendix 1

.1.1 Additional Figures Well U1

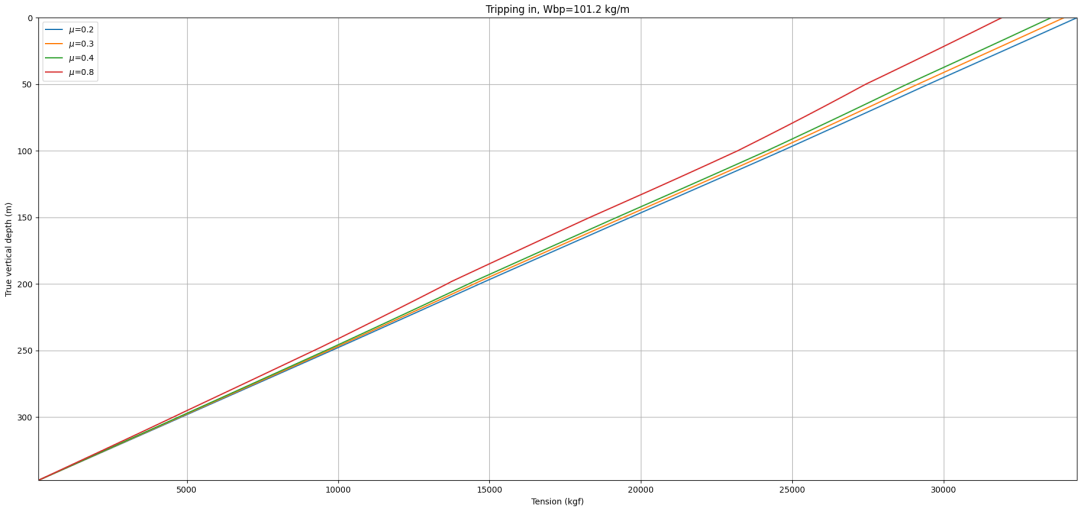

Figure 1: Impact of varying friction coefficients on drag forces for w_{bp} of 101.2 kg/m, tripping in, Well U1

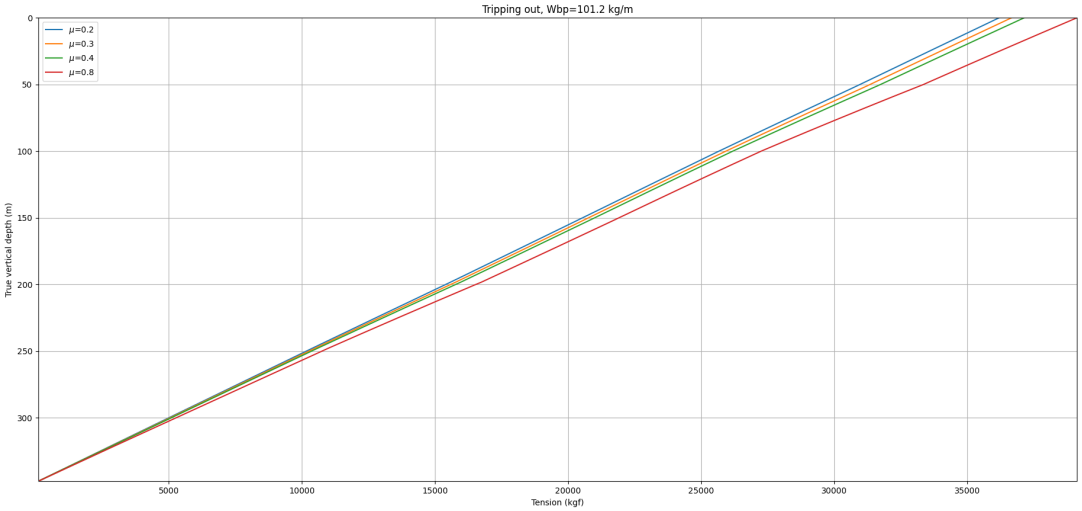

Figure 2: Impact of varying friction coefficients on drag forces for w_{bp} of 101.2 kg/m, tripping out, Well U1

.1.2 Additional Figures Well U2

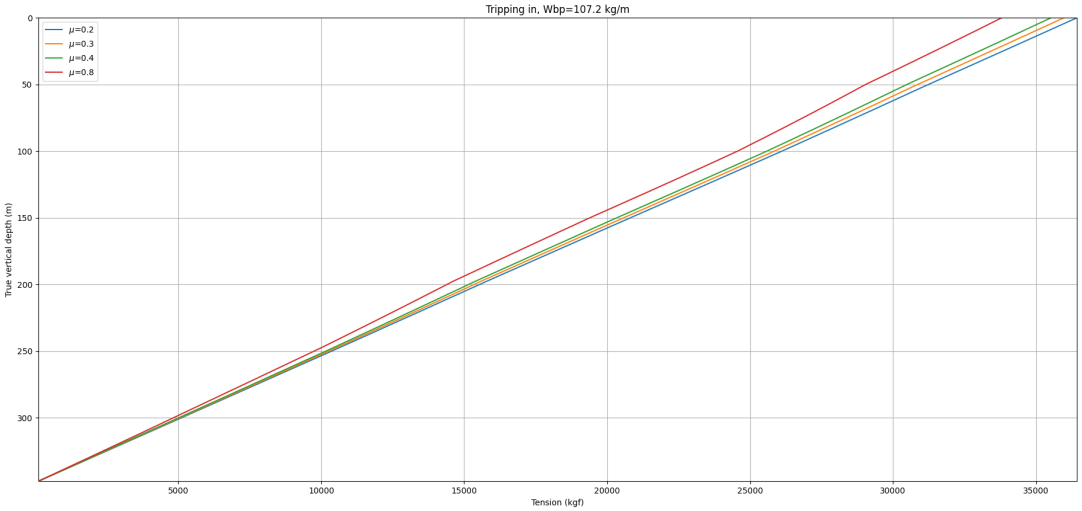

Figure 3: Impact of varying friction coefficients on drag forces for w_{bp} of 107.2 kg/m, tripping in Well U1

.1.3 Additional Figures Well U6B

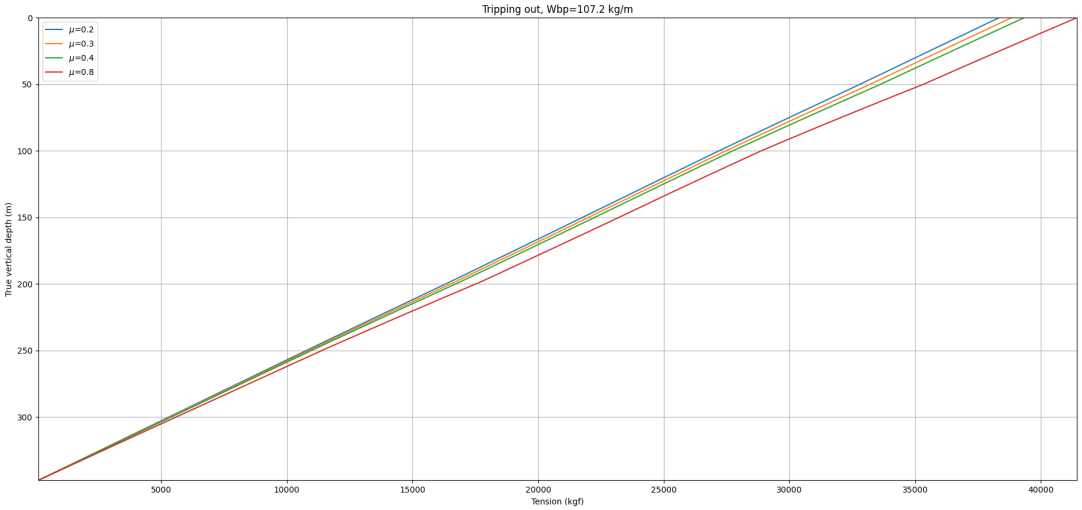

Figure 4: Impact of varying friction coefficients on drag forces for w_{bp} of 107.2 kg/m, tripping out, Well U1

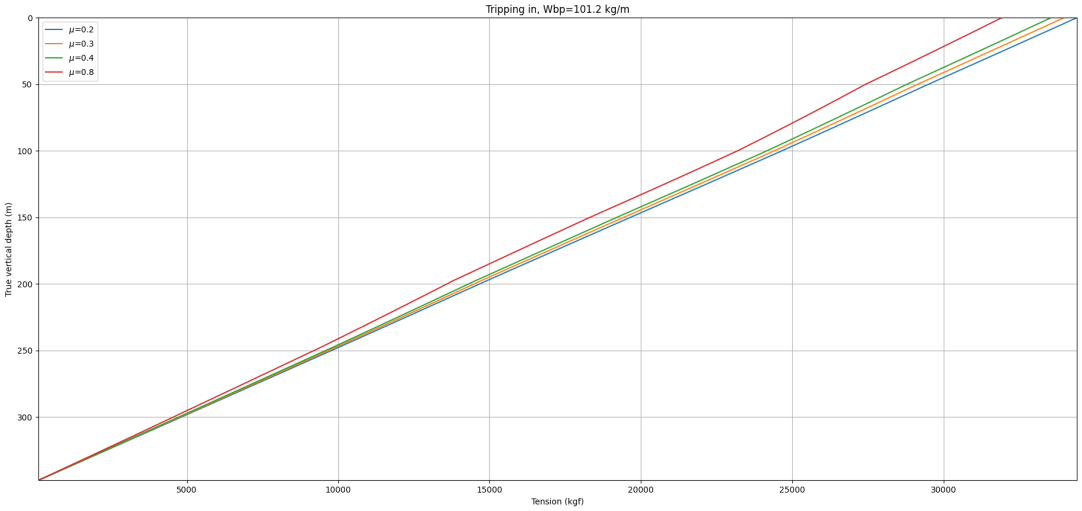

Figure 5: Impact of varying friction coefficients on drag forces for w_{bp} of 101.2 kg/m, tripping in, Well U2

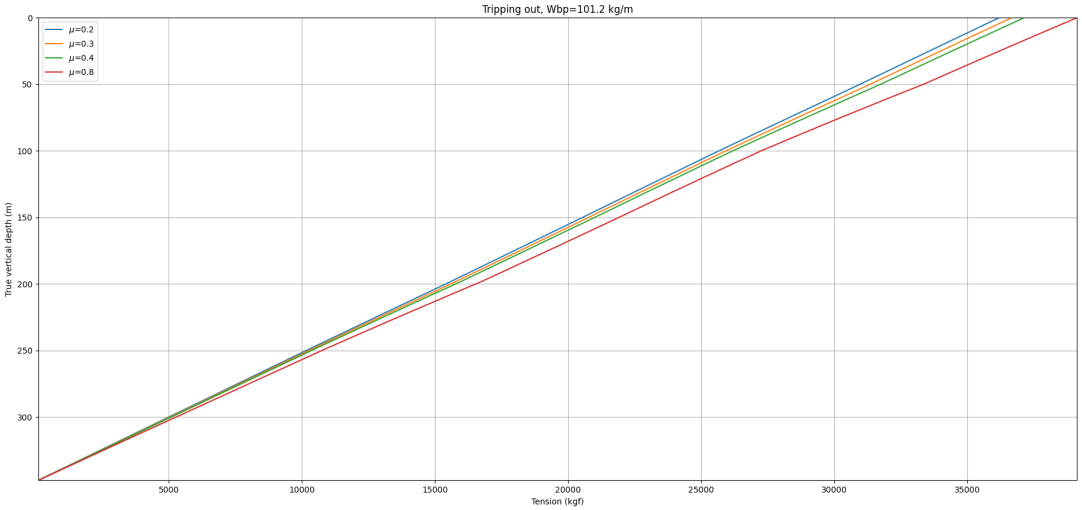

Figure 6: Impact of varying friction coefficients on drag forces for w_{bp} of 101.2 kg/m, tripping out, Well U2

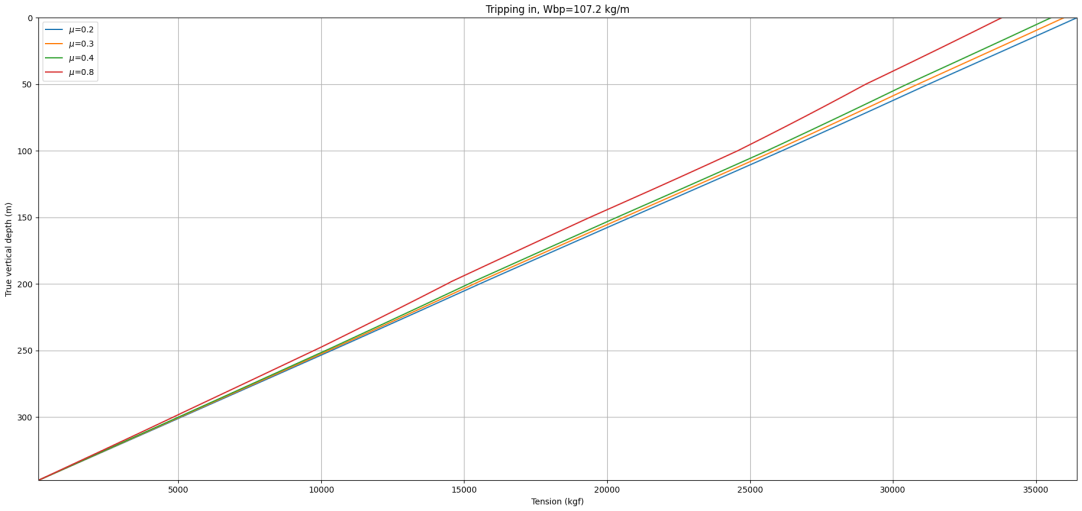

Figure 7: Impact of varying friction coefficients on drag forces for w_{bp} of 107.2 kg/m, tripping in, Well U2

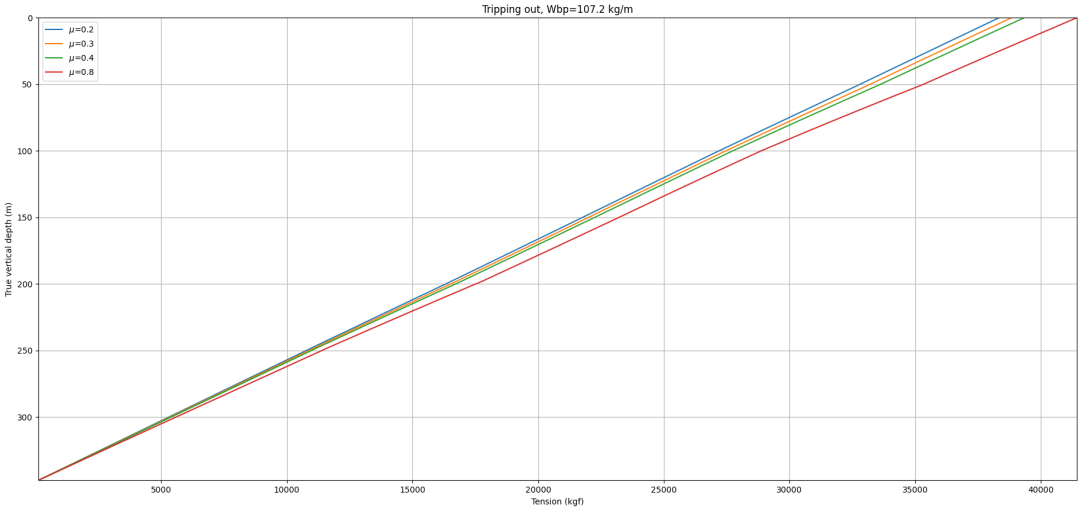

Figure 8: Impact of varying friction coefficients on drag forces for w_{bp} of 107.2 kg/m, tripping out, Well U2

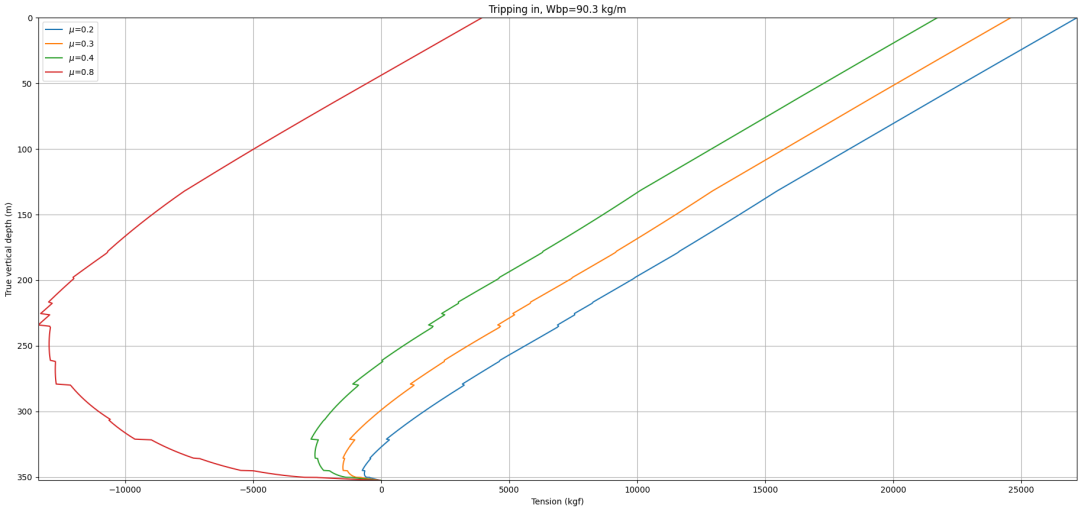

Figure 9: Impact of varying friction coefficients on drag forces for w_{bp} of 90.3 kg/m, tripping in, Well U6B

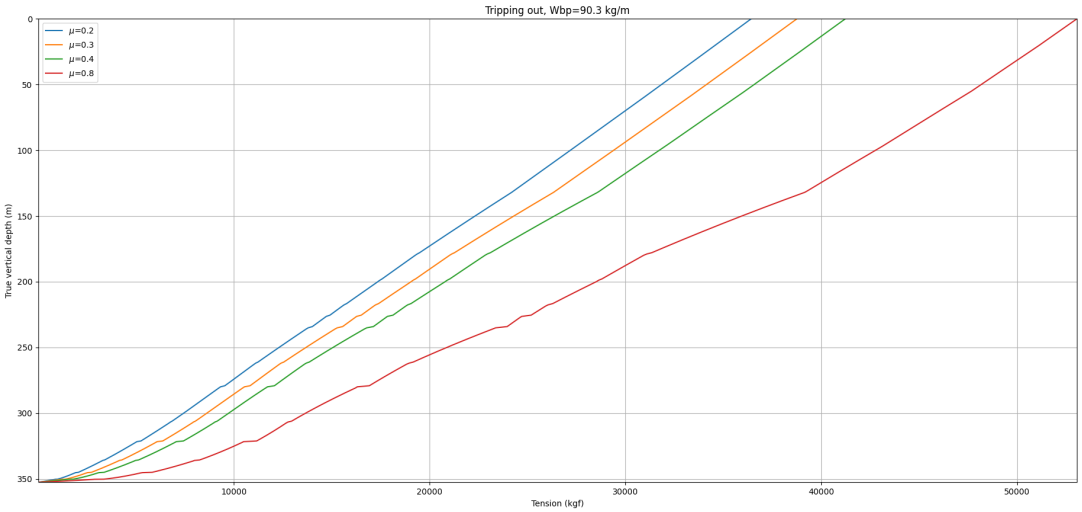

Figure 10: Impact of varying friction coefficients on drag forces for w_{bp} of 90.3 kg/m, tripping out, Well U6B

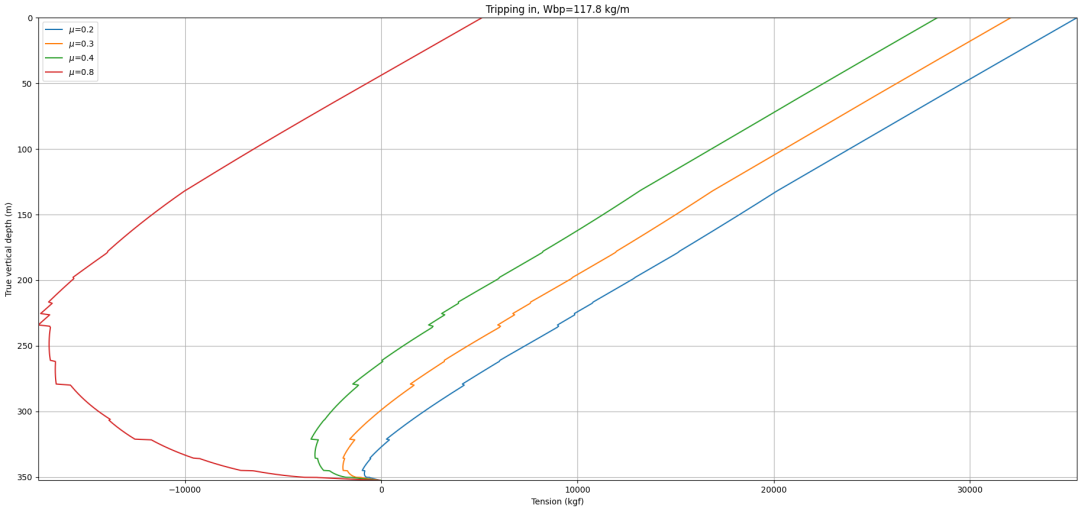

Figure 11: Impact of varying friction coefficients on drag forces for w_{bp} of 117.8 kg/m, tripping in, Well U6B

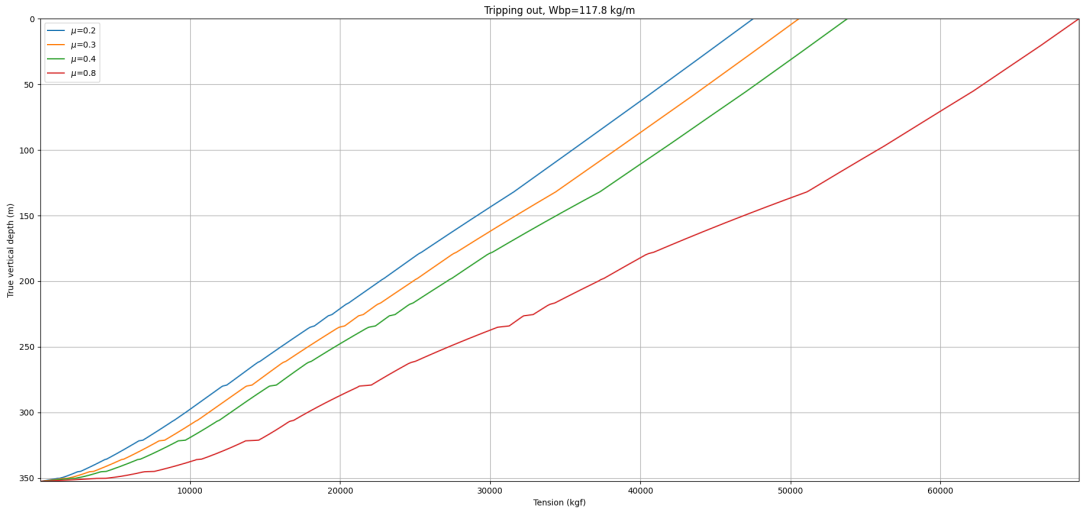

Figure 12: Impact of varying friction coefficients on drag forces for w_{bp} of 117.8 kg/m, tripping out, Well U6B

.2 Appendix 2

.2.1 Python Code

calculate_points.py

```
import math


# Loop through each survey station and calculate the coordinates
def calculate_points(rf_list, inc_list, azi_list):
    x1 = 0
    y1 = 0
    z1 = 0
    x_coords = [x1]
    y_coords = [y1]
    z_coords = [z1]

    for i in range(len(rf_list) -1):
        inc1 = math.radians(inc_list[i])
        azi1 = math.radians(azi_list[i])
        inc2 = math.radians(inc_list[i + 1])
        azi2 = math.radians(azi_list[i + 1])
        rf = rf_list[i]

        x2 = x1 + (math.sin(inc1) * math.cos(azi1) + math.sin(inc2) *
            ↪ math.cos(azi2)) * rf
        y2 = y1 + (math.sin(inc1) * math.sin(azi1) + math.sin(inc2) *
            ↪ math.sin(azi2)) * rf
        z2 = z1 + (math.cos(inc1) + math.cos(inc2)) * rf

        x_coords.append(x2)
        y_coords.append(y2)
        z_coords.append(z2)

        # Set the new starting point to the end point of the previous
            ↪ segment
        x1 = x2
        y1 = y2
        z1 = z2


    inc = math.radians(inc_list[-1])
    azi = math.radians(azi_list[-1])
    rf = rf_list[-1]
    x2 = x1 + math.sin(inc) * math.cos(azi) * rf
    y2 = y1 + math.sin(inc) * math.sin(azi) * rf
    z2 = z1 + (math.cos(inc1) + math.cos(inc2)) * rf

    x_coords.append(x2)
    y_coords.append(y2)
    z_coords.append(z2)



    return x_coords, y_coords, z_coords
```

Data - Copy.py

```
from torque_drag import nz
from torque_drag import bz
from DL_function import calculate_dl
from RF import calculate_rf
from calculate_points import calculate_points
from interpolation import interpolation
import math
import matplotlib.pyplot as plt
import numpy as np
from mpl_toolkits.mplot3d import Axes3D
from welly import Well
import lasio
import numpy as np
from mpl_toolkits.mplot3d import Axes3D
from matplotlib.gridspec import GridSpec
from prettytable import PrettyTable
from torque_drag import interpolate_every_meter
from torque_drag import tz
from torque_drag import calculate_torque_drag_l
from torque_drag import calculate_torque_drag_h
from torque_drag import calculate_rf_values
from torque_drag import calculate_wellbore_curve
from torque_drag import calculate_repeated_dl
from RF import calculate_rf_em
from torque_drag import get_wbp_changes
import matplotlib.colors as mcolors

np.set_printoptions(precision=3, suppress=True)

#Creating empty lists, fill in data from survey stations, azi, inc, md
dl_list = []
rf_list = []
points_list = []
wellbore_curve = []
md_list= []
distance_list = []
inc_list = []
azi_list = []
inc_list_radians = [math.radians(inc) for inc in inc_list]
azi_list_radians = [math.radians(azi) for azi in azi_list]

#creating distance list, used in calculations of s1-s
for i in range(len(md_list)-1):
    dist = md_list[i+1] - md_list[i]
    distance_list.append(dist)

#creating dogleg list
for i in range(len(md_list)-1):
    dl = calculate_dl(inc_list[i], azi_list[i], inc_list[i+1], azi_list
        ↪ [i+1], distance_list[i])
    dl_list.append(dl)  #Append to dl_list

#Creating list for kappa
for i in range(len(md_list)-1):
    constant_k = (dl_list[i] / distance_list[1])
    wellbore_curve.append(constant_k)
```

```
56
57 #Creating extended lists for calculations later
58 dl_list_ext = []
59 inc_list_extended = []
60 azi_list_extended = []
61 wellbore_curve_extended = []
62
63 for i in range(len(distance_list)):
64     dl_list_ext += list(np.repeat(dl_list[i], distance_list[i]))
65     inc_list_extended += list(np.repeat(inc_list_radians[i],
       ↪ distance_list[i]))
66     azi_list_extended += list(np.repeat(azi_list_radians[i],
       ↪ distance_list[i]))
67     wellbore_curve_extended += list(np.repeat(wellbore_curve[i],
       ↪ distance_list[i]))
68
69 #Creating Ratio factor list
70 rf_list = calculate_rf(dl_list, distance_list)
71
72 #Creating lists with x offset, y offset, z coordinates
73 x_coords, y_coords, z_coords = calculate_points(rf_list, inc_list,
    ↪ azi_list)
74 coordinates = list(zip(x_coords, y_coords, z_coords))
75
76 z_coords = [coord[2] for coord in coordinates]
77 z_coords = np.array(z_coords)
78
79
80 #Creating all the new lists. For later calculation
81 new_md_list, new_inc_list, new_azi_list, new_z_list = \
82     interpolate_every_meter(inc_list, azi_list, dl_list, distance_list,
       ↪  wellbore_curve, md_list, z_coords,inc_list_radians,
       ↪ azi_list_radians, dl_list_radians,wanted_depth=None)
83
84 np.set_printoptions(threshold=np.inf, edgeitems=np.inf)
85 new_values = list(zip(new_md_list, new_inc_list, new_azi_list,
    ↪ new_z_list))
86 np_new_values = np.array(new_values)
87
88
89 #Creating new lists for further calculation
90
91 new_dl_list = calculate_repeated_dl(new_inc_list, new_azi_list,
    ↪ distance_list, new_md_list)
92 new_wellbore_list = calculate_wellbore_curve(new_dl_list, new_md_list)
93
94
95 nz_list = nz(new_inc_list, wellbore_curve_extended, dl_list_ext,
    ↪ new_md_list)
96 bz_list_1 = bz(inc_list_radians, azi_list_radians, dl_list_radians,
    ↪ md_list,distance_list)
97 tz_list = tz(new_inc_list, dl_list_ext, new_wellbore_list)
98 bz_list = []
99 for i in range(len(distance_list)):
100     bz_list += list(np.repeat(bz_list_1[i], distance_list[i]))
101
102 new_rf_list = calculate_rf_values(dl_list, new_md_list,distance_list )
103
```

```
104
105 np_data2 = np.array([new_z_list, new_inc_list, new_azi_list, nz_list,
    ↪ bz_list, tz_list, dl_list_ext, wellbore_curve_extended], dtype =
    ↪ float).T
106 headers = ['TVD', 'Inclination (deg)', 'Azimuth (deg)', 'nz', 'bz', 'tz
    ↪ ', 'dl', 'k']
107 table = PrettyTable(headers)
108
109 for row in np_data2:
110     table.add_row(row)
111
112 rf_list_em = calculate_rf_em(dl_list,distance_list)
113
114 #create a numpy array
115 np_coordinates = np.array(coordinates)
116 np_data_coords = np_coordinates[:, 0:3]
117
118 np_data = np.array([md_list, inc_list, azi_list], dtype =float).T      #
    ↪ Creating NP array
119 np_data = np.hstack((np_data, np_data_coords))
120 print(np_data.shape)
121
122 headers = ['MD (m)', 'Inclination (deg)', 'Azimuth (deg)', 'X (m)', 'Y
    ↪ (m)', 'Z (m)']
123 table = PrettyTable(headers)
124
125 for row in np_data:
126     table.add_row(row)
127
128
129
130 #Create LAS file
131 las = lasio.LASFile()
132
133 # add curves to LAS file
134 las.add_curve('DEPT', unit='M', descr='Measured Depth', data =np_data
    ↪ [:,0])
135 las.add_curve('INCL', unit='DEG', descr='Inclination', data =np_data
    ↪ [:,1])
136 las.add_curve('AZIM', unit='DEG', descr='Azimuth', data = np_data[:,2])
137 las.add_curve('X', unit='NA', descr='X offset', data =np_data[:,3])
138 las.add_curve('Y', unit='NA', descr='Y offset', data =np_data[:,4])
139 las.add_curve('Z', unit='NA', descr='Z offset', data = np_data[:,5])
140
141 xs = las['X']
142 ys = las['Y']
143 zs = las['Z']
144
145 # add data to curves
146 las['DEPT'] = np_data[:, 0]
147 las['INCL'] = np_data[:, 1]
148 las['AZIM'] = np_data[:, 2]
149 las['X'] = np_data[:, 3]
150 las['Y'] = np_data[:, 4]
151 las['Z'] = np_data[:, 5]
152
153 # write LAS file
154 las.write('') #Fill in directory
```

```
155
156 # Load LAS file
157 las = Well.from_las('') #fill in directory
158
159
160 # Load LAS file
161 well = Well.from_las('C:/Users/edlar/PycharmProjects/pythonProject/
    ↪ Bachelor/well_1.las')
162
163 # Get curves
164 xs = well.data['X'].values
165 ys = well.data['Y'].values
166 zs = well.data['Z'].values
167 zs_inv = zs - zs.min() # set shallowest point as zero and let the well
    ↪ go downwards
168
169 # create 3D plot
170 fig = plt.figure(figsize=(14, 8))
171 gs = fig.add_gridspec(nrows=2, ncols=4, width_ratios=[4, 4, 4, 4])
172
173 ax1 = fig.add_subplot(gs[:, :2], projection='3d')
174 ax1.plot(xs, ys, zs_inv, linewidth=3)
175 ax1.plot(xs[0], ys[0], zs_inv[0], marker =".", color="black", ms=5)
176 ax1.plot(xs[-1], ys[-1], zs_inv[-1], marker ="*", color="red", ms=5)
177 ax1.set_title("3D Well path")
178 ax1.set_xlabel('X offset (m)')
179 ax1.set_ylabel('Y offset (m)')
180 ax1.set_zlabel('TVD (m)')
181
182 # Create cubic bounding box to simulate equal aspect ratio
183 max_range = np.array([xs.max()-xs.min(), ys.max()-ys.min(), zs_inv.max
    ↪ ()-zs_inv.min()]).max()
184 Xb = 0.5*max_range*np.mgrid[-1:2:2,-1:2:2,-1:2:2][0].flatten() + 0.5*(
    ↪ xs.max()+xs.min())
185 Yb = 0.5*max_range*np.mgrid[-1:2:2,-1:2:2,-1:2:2][1].flatten() + 0.5*(
    ↪ ys.max()+ys.min())
186 Zb = 0.5*max_range*np.mgrid[-1:2:2,-1:2:2,-1:2:2][2].flatten() + 0.5*(
    ↪ zs_inv.max()+zs_inv.min())
187
188 # Comment or uncomment following both lines to test the fake bounding
    ↪ box:
189 for xb, yb, zb in zip(Xb, Yb, Zb):
190    ax1.plot([xb], [yb], [zb], 'w')
191
192 ax2 = fig.add_subplot(gs[0, 2])
193 ax2.plot(xs, ys, lw=2)
194 ax2.set_title('X Location vs Y Location')
195 ax2.set_xlabel('X offset (m)')
196 ax2.set_ylabel('Y offset (m)')
197
198 ax3 = fig.add_subplot(gs[1, 2])
199 ax3.plot(xs, zs_inv, lw=2)
200 ax3.set_title('X Location vs TVD')
201 ax3.set_xlabel('X offset (m)')
202 ax3.set_ylabel('TVD (m)')
203
204 ax4 = fig.add_subplot(gs[:, 3])
205 ax4.plot(ys, zs_inv, lw=2)
```

```
206 ax4.set_title('Y Location vs TVD')
207 ax4.set_xlabel('Y offset (m)')
208 ax4.set_ylabel('TVD (m)')
209
210 ax3.invert_yaxis()
211 ax4.invert_yaxis()
212
213 plt.tight_layout()
214 plt.show()
215
216
217
218 # Enter test values here
219
220 ff_values = [0.2, 0.3, 0.4, 0.8] #used friction coefficeint 0.2 - 0.4
221 wbp_values = [] #Fill inn nominal weight of segment
222
223
224 def compare_plot(tvd_values):
225     for j, wbp in enumerate(wbp_values):
226         # Separate max and min forces for 'Tripping in' and 'Tripping
            ↪ out'
227         max_force_in = float('-inf')
228         min_force_in = float('inf')
229         max_force_out = float('-inf')
230         min_force_out = float('inf')
231
232         plt.figure(figsize=(10,5)) # Create a new figure for each '
            ↪ Tripping in' at given wbp
233         for i, ff in enumerate(ff_values):
234             # Calculate force_values for tripping in
235             force_values_in = calculate_torque_drag_l(wbp, ff,
                ↪ new_md_list, wellbore_curve_extended, nz_list,
                ↪ bz_list, tz_list, new_inc_list, new_azi_list)
236             plt.plot(force_values_in, tvd_values, label=f'$\mu$={ff}')
237             # Get maximum and minimum force values for setting limits
238             max_force_in = max(max_force_in, max(force_values_in))
239             min_force_in = min(min_force_in, min(force_values_in))
240
241         setup_plot(max_force_in, min_force_in, tvd_values, f'Tripping
            ↪ in, Wbp={wbp} kg/m')
242
243         plt.figure(figsize=(10,5)) # Create a new figure for each '
            ↪ Tripping out' at given wbp
244         for i, ff in enumerate(ff_values):
245             # Calculate force_values for tripping out
246             force_values_out = calculate_torque_drag_h(wbp, ff,
                ↪ new_md_list, wellbore_curve_extended, nz_list,
                ↪ bz_list, tz_list, new_inc_list, new_azi_list)
247             plt.plot(force_values_out, tvd_values, label=f'$\mu$={ff}')
248             # Get maximum and minimum force values for setting limits
249             max_force_out = max(max_force_out, max(force_values_out))
250             min_force_out = min(min_force_out, min(force_values_out))
251
252         setup_plot(max_force_out, min_force_out, tvd_values, f'Tripping
            ↪  out, Wbp={wbp} kg/m')
253
254 def setup_plot(max_force, min_force, tvd_values, title):
```

```
255     # Set up the rest of the plot
256     plt.gca().invert_yaxis()  # Invert y-axis as depth increases
            ↪ downwards
257     plt.xlabel('Tension (kgf)')
258     plt.ylabel('True vertical depth (m)')
259     plt.title(title)
260     plt.legend()  # Add a legend
261     plt.grid(True)  # Show grid
262     plt.xlim([min_force, max_force])  # Set x-axis limits
263     plt.ylim([max(tvd_values), min(tvd_values)])  # Set y-axis limits
264 
265     # Show the plot
266     plt.tight_layout()
267     plt.show()
268 
269 compare_plot(new_z_list)
```

DL_function.py

```
1  import math
2  
3  def calculate_dl(inc1, azi1, inc2, azi2, distance):
4      # Convert the inclination and azimuth angles to radians
5      inc1 = math.radians(inc1)
6      azi1 = math.radians(azi1)
7      inc2 = math.radians(inc2)
8      azi2 = math.radians(azi2)
9  
10     # Calculate the direction change between the points
11     dir_change = abs(inc2 - inc1)
12 
13     # Calculate the dogleg severity
14 
15     #dl = math.degrees(math.acos(math.sin(inc1)*math.sin(inc2)*math.cos
           ↪ (azi1-azi2)+math.cos(inc1)*math.cos(inc2)))
16     dl = 2 * math.asin(math.sqrt((math.sin((inc2 - inc1) / 2))**2 +
           ↪ math.sin(inc1) * math.sin(inc2) * (math.sin((azi2 - azi1) /
           ↪ 2))**2))
17 
18     return dl
```

interpolation.py

```
1  import math
2  
3  
4  
5  def interpolation(inc_list_radians, azi_list_radians, dl_list_radians,
       ↪ distance_list, wellbore_curve, wanted_depth, md_list):
6      if wanted_depth < min(md_list) or wanted_depth > max(md_list):
7          print("Error: Depth is outside survey range.")
8          return None
9  
10     dist_from_start = None
11     for i in range(len(md_list)-1):
12         if md_list[i] <= wanted_depth < md_list[i+1]:
13             dist_from_start = wanted_depth - md_list[i]
14             break
15 
```

```
    if dist_from_start is None:
        dist_from_start = sum(distance_list)

    cos_a = math.cos(inc_list_radians[i]) * math.cos(wellbore_curve[i]
        ↪ * dist_from_start)
    if dl_list_radians[i] == 0:
        cos_b = 0
    else:
        cos_b = (math.cos(inc_list_radians[i + 1]) - (math.cos(
            ↪ inc_list_radians[i]) * math.cos(dl_list_radians[i]))) /
            ↪ math.sin(dl_list_radians[i])
    cos_c_split = wellbore_curve[i] * dist_from_start
    cos_c = math.sin(cos_c_split)
    cos_d = cos_a + cos_b * cos_c
    des_inc = math.degrees(math.acos(cos_d))

    if math.degrees(inc_list_radians[i]) > math.degrees(
        ↪ inc_list_radians[i + 1]):
        des_azi_value += math.pi

    return "Desired inclination " + str(des_inc) , "Desired azimuth: "
        ↪ + str(des_azi_value), "Distance from start: " + str(
        ↪ dist_from_start)
```

RF.py

```
import math

from DL_function import calculate_dl  # import calculate_dl from
    ↪ DL_function module


def calculate_rf(dl_list, distance_list):
    # Calculate the ratio factor for each segment
    rf_list = []
    for i in range(len(dl_list)):
        dl_radians = math.radians(dl_list[i]) # Convert dl from degrees
            ↪  to radians
        if dl_list[i] > 0.001745:
            rf = (distance_list[i] / dl_radians) * (math.tan(dl_radians
                ↪ / 2))
        else:
            rf = 1
        rf_list.append(rf)
    return rf_list

def calculate_rf_em(dl_list, distance_list):
    # Calculate the ratio factor for each segment
    rf_list_em = []
    for i in range(len(dl_list)):
        dl_radians = math.radians(dl_list[i]) # Convert dl from degrees
            ↪  to radians
        rf = (distance_list[i] / dl_radians) * (math.tan(dl_radians/ 2)
            ↪ )
        rf_list_em.append(rf)
    return rf_list_em
```

torque_drag.py

```python
import math
from interpolation import interpolation
from RF import calculate_rf
import numpy as np
import math
from DL_function import calculate_dl
from RF import calculate_rf_em


#Function calculates change in z-coordinates
def calculate_z2(new_inc_list, new_azi_list, rf_list_em):
    z1 = 0
    new_z_coords = [z1]

    for i in range(len(new_inc_list) -1):
        inc1 = math.radians(new_inc_list[i])
        inc2 = math.radians(new_inc_list[i + 1]) if i + 1 < len(
            ↪ new_inc_list) else inc1
        rf = rf_list_em[i % len(rf_list_em)]

        z2 = z1 + (math.cos(inc1) + math.cos(inc2)) * rf
        new_z_coords.append(z2)

        z1 = z2

    return new_z_coords

#Function calculates wellbore curve with same index as max len(MD_list)
def calculate_wellbore_curve(new_inc_list, new_azi_list):
    wellbore_curve = np.empty(len(new_inc_list), dtype=float)

    for i in range(len(new_inc_list) - 1):
        azi_diff = math.radians(new_azi_list[i + 1] - new_azi_list[i])
        inc_diff = math.radians(new_inc_list[i + 1] - new_inc_list[i])
        k = math.sqrt(inc_diff**2 + math.sin(math.radians(new_inc_list[
            ↪ i]))**2 * azi_diff**2)

        wellbore_curve[i] = k
    return wellbore_curve

#Since the dogleg is constant between two points, the dl_list had to be
    ↪  extended to match len(new_md_list)
def calculate_repeated_dl(new_inc_list, new_azi_list, distance_list,
    ↪ new_md_list):
    dl_values = []
    for i in range(len(new_md_list) - 1):
        inc1 = math.radians(new_inc_list[i])
        azi1 = math.radians(new_azi_list[i])
        inc2 = math.radians(new_inc_list[i + 1])
        azi2 = math.radians(new_azi_list[i + 1])
        distance = 1

        dl = calculate_dl(inc1, azi1, inc2, azi2, distance)
        dl_values.append(dl)
    return np.array(dl_values)

#Since the RF is constant between two points, the RF had to be extended
    ↪  to match len(new_md_list)
```

```
54 def calculate_rf_values(dl_list, new_md_list,distance_list):
55     rf_values = []
56     total_distance = 0
57     next_rf_distance = new_md_list[0]
58 
59     for i in range(len(dl_list)):
60         total_distance += new_md_list[i]
61 
62         if total_distance >= next_rf_distance or i == len(dl_list) - 1:
63             rf = calculate_rf_em([dl_list[i]],distance_list)[0]
64             rf_values.append(rf)
65             next_rf_distance += 50
66         else:
67             rf_values.append(rf_values[-1] if rf_values else rf)
68     return np.array(rf_values)
69 
70 #Creates a list of len(max(md_list) so that we can calculate drag
    ↪ forces every meter.
71 def interpolate_every_meter(inc_list, azi_list, dl_list, distance_list,
    ↪  wellbore_curve, md_list, z_coords,inc_list_radians,
    ↪ azi_list_radians, dl_list_radians, wanted_depth):
72     start_depth = md_list[0] if md_list[0] != 0 else 1
73     end_depth = md_list[-1]
74     new_md_list = np.arange(start_depth, end_depth + 1)
75     new_inc_list = np.empty(new_md_list.shape, dtype=float)
76     new_azi_list = np.empty(new_md_list.shape, dtype=float)
77     new_z_list = np.empty(new_md_list.shape, dtype=float)
78     z_coords = z_coords
79     new_rf_list = np.empty(new_md_list.shape, dtype=float)
80 
81     for depth_idx, depth in np.ndenumerate(new_md_list):
82         result = interpolation(inc_list_radians, azi_list_radians,
            ↪ dl_list_radians, distance_list, wellbore_curve, depth,
            ↪ md_list)
83         if result is not None:
84             des_inc, des_azi, distance_from_start = result
85             des_inc = float(des_inc.split()[-1])
86             des_azi = float(des_azi.split()[-1])
87             new_inc_list[depth_idx[0]] = des_inc
88             new_azi_list[depth_idx[0]] = des_azi
89             i = 0
90             while i < len(md_list) - 1 and depth > md_list[i + 1]:
91                 i += 1
92             rf = calculate_rf_values([dl_list[i]], [distance_list[i]],
                ↪ new_md_list)[0]
93             new_rf_list[depth_idx[0]] = rf
94     new_z_list = calculate_z2(new_inc_list, new_azi_list, new_rf_list)
95     new_inc_list[-1] = inc_list[-1]
96     new_azi_list[-1] = azi_list[-1]
97     return new_md_list, new_inc_list, new_azi_list, new_z_list
98 
99 #Calculate nz
100 def nz(new_inc_list, wellbore_curve_extended, dl_list_ext, new_md_list)
    ↪ :
101     nz_values = np.empty(len(new_md_list), dtype=float)
102     for i in range(len(new_md_list)-1):
103         if i < len(new_md_list):
104             distance_from_start = new_md_list[i + 1] - new_md_list[i]
```

```
105             nz_values[i] = -(1/wellbore_curve_extended[i]) * math.sin(
                    ↪ math.radians(new_inc_list[i])) * math.radians(
                    ↪ new_inc_list[i+1] - new_inc_list[i])
106     nz_values[-1] = nz_values[-3 ]
107     nz_values[-2] = nz_values[-3 ]
108     return nz_values
109
110 #calculate bz
111 def bz(inc_list_radians, azi_list_radians, dl_list_radians, md_list,
        ↪ distance_list):
112     bz_values = np.empty(len(md_list)-1, dtype=float)
113     for i in range(len(md_list) -1):
114         bz = (math.sin(inc_list_radians[i]) * math.sin(inc_list_radians
                ↪ [i + 1]) * math.sin(azi_list_radians[i + 1] -
                ↪ azi_list_radians[i])) \
115                         / math.sin(dl_list_radians[i])
116         bz_values[i] = bz
117     return bz_values
118
119 #calculate tz
120 def tz(new_inc_list, new_dl_list, wellbore_curve_extended):
121     tz_values = np.empty(len(new_inc_list), dtype=float)
122     for i in range(len(new_inc_list)-1):
123         if i < len(new_inc_list):
124             tz_values[i] = math.cos(math.radians(new_inc_list[i]))
125     tz_values[-1] = tz_values[-2]
126     return tz_values
127
128 #drag calculation for hoisting
129 def calculate_torque_drag_h(wbp, ff, new_md_list,
        ↪ wellbore_curve_extended, nz_list, bz_list, tz_list, new_inc_list
        ↪ , new_azi_list):
130     force_array = np.empty(len(new_md_list), dtype=float)
131     f1 = 0
132     new_md_list_reversed = new_md_list[::-1]
133     wellbore_curve_reversed = wellbore_curve_extended[::-1]
134     nz_list_reversed = nz_list[::-1]
135     bz_list_reversed = bz_list[::-1]
136     tz_list_reversed = tz_list[::-1]
137
138     # calculate initial force
139     tz_value = tz_list_reversed[0]
140     nz_value = nz_list_reversed[0]
141     bz_value = bz_list_reversed[0]
142     wellbore_curve_value = wellbore_curve_reversed[0]
143
144     cumulative_force = 0
145
146     for idx in range(0, len(new_md_list_reversed)):
147         tz_value = tz_list_reversed[idx]
148         nz_value = nz_list_reversed[idx]
149         bz_value = bz_list_reversed[idx]
150         wellbore_curve_value = wellbore_curve_reversed[idx]
151
152         cf = np.sqrt((cumulative_force * wellbore_curve_value + wbp *
                ↪ nz_value) ** 2 + (wbp * bz_value) ** 2)
153         a = wbp * tz_value
154         delta_force = (a + ff * cf)
```

```
155         cumulative_force += delta_force
156 
157         force_array[-idx - 1] = cumulative_force
158 
159     return force_array
160 
161 #drag calculation for lowering
162 def calculate_torque_drag_l(wbp, ff, new_md_list,
    ↪ wellbore_curve_extended, nz_list, bz_list, tz_list, new_inc_list
    ↪ , new_azi_list):
163     force_array = np.empty(len(new_md_list), dtype=float)
164     f1 = 0
165     new_md_list_reversed = new_md_list[::-1]
166     wellbore_curve_reversed = wellbore_curve_extended[::-1]
167     nz_list_reversed = nz_list[::-1]
168     bz_list_reversed = bz_list[::-1]
169     tz_list_reversed = tz_list[::-1]
170 
171     # calculate initial force
172     tz_value = tz_list_reversed[0]
173     nz_value = nz_list_reversed[0]
174     bz_value = bz_list_reversed[0]
175     wellbore_curve_value = wellbore_curve_reversed[0]
176     cumulative_force = 0
177     for idx in range(0, len(new_md_list_reversed)):
178         tz_value = tz_list_reversed[idx]
179         nz_value = nz_list_reversed[idx]
180         bz_value = bz_list_reversed[idx]
181         wellbore_curve_value = wellbore_curve_reversed[idx]
182         cf = np.sqrt((cumulative_force * wellbore_curve_value + wbp *
            ↪ nz_value) ** 2 + (wbp * bz_value) ** 2)
183         a = wbp * tz_value
184         delta_force = (a - ff * cf)
185         cumulative_force += delta_force
186 
187         force_array[-idx - 1] = cumulative_force
188 
189     return force_array
190 
191 #This function was created in the case that there are different wbp
    ↪ values at different points in the wellbore,
192 # for example if used for a drill string rather than casing and there
    ↪ are different weights at different segments
193 def get_wbp_changes(new_z_list):
194     wbp_changes = []
195     min_meter = min(new_z_list)
196     max_meter = max(new_z_list)
197 
198     start_wbp = input("Enter the start value of WBP: ")
199     try:
200         start_wbp = float(start_wbp)
201     except ValueError:
202         print("Invalid input. Please enter a number for the start value
            ↪  of WBP.")
203         return wbp_changes
204 
205     wbp_changes.append((0, start_wbp))  # add the starting point wbp to
        ↪  the list
```

```
206
207     while True:
208         meter = input("Enter the meter at which wbp changes, or 'done'
                ↪ to finish: ")
209         if meter.lower() == 'done':
210             break
211         wbp = input("Enter the new wbp value at this meter: ")
212         try:
213             meter_value = int(meter)
214             if meter_value < min_meter or meter_value > max_meter:
215                 print(f"Invalid input. The meter value must be between
                        ↪ {min_meter} and {max_meter}.")
216                 continue
217             wbp_changes.append((meter_value, float(wbp)))
218         except ValueError:
219             print("Invalid input. Please enter numbers only.")
220
221     # sort the list of tuples in descending order of meter
222     wbp_changes.sort(key=lambda x: x[0], reverse=True)
223
224     return wbp_changes
225
226 def find_nearest(array, value):
227     array = np.asarray(array)
228     idx = (np.abs(array - value)).argmin()
229     return idx
```

Bibliography

[1] R. F. Mitchell and S. Z. Miska, *Fundamentals of Drilling Engineering*. Society of Petroleum Engineers. [Online]. Available: http://ebookcentral.proquest.com/lib/uisbib/detail.action?docID=3404993

[2] R. F. Mitchell, A. Bjørset, and G. Grindhaug, "Drillstring analysis with a discrete torque/drag model," vol. 30, no. 1, pp. 5–16. [Online]. Available: https://doi.org/10.2118/163477-PA

[3] I. Netichuk, A. Suvorov, A. Galimov, and RN-Shelf-Arctic, "Special features of casing running and cementing in ERD wells."

[4] E. Cayeux, H. J. Skadsem, B. Daireaux, and R. Holand, "Challenges and solutions to the correct interpretation of drilling friction tests," in *Day 2 Wed, March 15, 2017*. SPE, p. D021S010R004. [Online]. Available: https://onepetro.org/SPEDC/proceedings/17DC/2-17DC/The%20Hague,%20The%20Netherlands/194458

[5] V. P. Gupta, A. H. P. Yeap, K. M. Fischer, R. S. Mathis, and M. J. Egan, "Expanding the extended reach envelope at chayvo field, sakhalin island."

[6] Test wells - norce. [Online]. Available: https://ullrigg.norceresearch.no/test-wells

[7] C. Johancsik, D. Friesen, and R. Dawson, "Torque and drag in directional wells-prediction and measurement," vol. 36, no. 6, pp. 987–992. [Online]. Available: https://doi.org/10.2118/11380-PA

[8] (2) understanding torque & drag: Concepts and analysis | LinkedIn. [Online]. Available: https://www.linkedin.com/pulse/understanding-torque-drag-concepts-analysis-sixto-romero/

[9] "Friction," page Version ID: 1151447686. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Friction&oldid=1151447686

[10] keyseat. [Online]. Available: https://glossary.slb.com/en/terms/k/keyseat

[11] A. Ross. Casing running techniques for extended reach wells. [Online]. Available: https://citadelcasingsolutions.com/success-with-new-casing-equipment-2/

[12] ADNOC drilling delivers new world record for the longest well. [Online]. Available: https://www.adnoc.ae/en/news-and-media/press-releases/2022/adnoc-drilling-delivers-new-world-record-for-the-longest-well

[13] D. Etaje, R. Shor, and R. Lashkari, "Torque and drag calculations in multilateral wells," in *Day 3 Wed, August 03, 2022*. SPE, p. D031S014R003. [Online]. Available: https://onepetro.org/SPENAIC/proceedings/22NAIC/3-22NAIC/D031S014R003/495083

[14] H.-S. Ho, "An improved modeling program for computing the torque and drag in directional and deep wells." OnePetro. [Online]. Available: https://dx.doi.org/10.2118/18047-MS

[15] R. Samuel and W. Huang, "Dynamic torque and drag model." OnePetro. [Online]. Available: https://dx-doi-org.ezproxy.uis.no/10.2118/201629-MS

[16] S. Z. Z. Miska, Z. Zamanipour, A. Merlo, and M. N. N. Porche, "Dynamic soft string model and its practical application." OnePetro. [Online]. Available: https://dx-doi-org.ezproxy.uis.no/10.2118/173084-MS

[17] S. J. Sawaryn and J. L. Thorogood, "A compendium of directional calculations based on the minimum curvature method," vol. 20, no. 1, pp. 24–36. [Online]. Available: https://doi.org/10.2118/84246-PA

[18] Coefficient of friction | definition & formula | britannica. [Online]. Available: https://www.britannica.com/science/coefficient-of-friction

[19] B. S. Aadnoy and E. Kaarstad, "Theory and application of buoyancy in wells." OnePetro. [Online]. Available: https://dx-doi-org.ezproxy.uis.no/10.2118/101795-MS

[20] Download python. [Online]. Available: https://www.python.org/downloads/

[21] Matplotlib — visualization with python. [Online]. Available: https://matplotlib.org/

[22] NumPy. [Online]. Available: https://numpy.org/

[23] welly documentation. [Online]. Available: https://code.agilescientific.com/welly/

[24] lasio - log ASCII standard (LAS) files in python — lasio 0.31 documentation. [Online]. Available: https://lasio.readthedocs.io/en/latest/

[25] L. Maurits, "prettytable: A simple python library for easily displaying tabular data in a visually appealing ASCII table format." [Online]. Available: https://github.com/jazzband/prettytable