The Thesis Committee for Katy Lynn Hanson
Certifies that this is the approved version of the following Thesis:

Isogeometric Analysis: Applications for torque and drag models, drillstring and bottom-hole assembly design

APPROVED BY
SUPERVISING COMMITTEE:

Eric van Oort, Supervisor

John T. Foster, Co-Supervisor

Pradeepkumar Ashok

Isogeometric Analysis: Applications for torque and drag models, drillstring and bottom-hole assembly design

by

Katy Lynn Hanson

Thesis

Presented to the Faculty of the Graduate School of

The University of Texas at Austin

in Partial Fulfillment

of the Requirements

for the Degree of

Master of Science in Engineering

The University of Texas at Austin
May 2018

Acknowledgements

I would like to highly thank Dr. Eric van Oort and Dr. John T. Foster for their supervision and guidance through this research endeavor. I also thank my fellow members of the UT RAPID research group. Foremost, I thank God for the opportunities set before me, and for the blessing of parents, family, and friends, whose support makes my success possible.

Abstract

Isogeometric Analysis: Applications for torque and drag models, drillstring and bottom-hole assembly design

Katy Lynn Hanson, MSE

The University of Texas at Austin, 2018

Supervisor: Dr. Eric van Oort, Dr. John T. Foster

The drilling industry today relies on torque and drag models to analyze and ensure success during all phases of well construction and operations, including planning, drilling, and completion. Analytical models are based on equations that are undergoing constant development and improvement. The finite element method is an alternative to complex analytical calculations that is used often to determine torque and drag forces that are present when a drillstring is lowered, raised, and rotated in a wellbore. Traditional finite element analysis (FEA), however, is not time efficient or computationally able to simulate the complexities of a real wellbore. Thus, we introduce an alternative to the traditional finite element approach: isogeometric analysis. Isogeometric analysis is similar to finite element analysis except that it uses NURBS (Non-Uniform Rational B-Splines), as opposed to interpolatory polynomials used in traditional FEA, as the basis functions. NURBS functions are the same as those used in CAD programs, and they are able to construct exact conic shapes, such as circles and ellipses. Adopting NURBS basis functions allows finite element analysis to be performed directly on the exact geometrical surface – not on an approximate geometric surface mesh, as in traditional FEA. IGA yields a significantly faster and more accurate simulation. This thesis presents a real-world application of IGA to a drag force model to determine the resultant surface hook load during run-in-hole (RIH) operations. Real well data is used, and IGA results are compared to a similar FEA analysis. The outcome shows that IGA is indeed a superior finite element method that has immense potential for further application in the industry.

Table of Contents

List of Tables

List of Figures

Chapter 1: Introduction

1.1 The Drilling Process and Drillstring Design

Oil and gas hydrocarbons are found in organic rich (source) rocks buried deep beneath the surface of the earth, at both onshore and offshore locations. Organic matter in these source rocks has been compacted over millions of years and exposed to high temperatures and pressures, causing thermogenic breakdown and creation of oil and gas. After formation, the oil and gas may migrate upwards through permeable rock until it reaches an impermeable seal and collects to form a conventional reservoir. Or, it may remain in place in the source rock to form an unconventional reservoir (Figure 1.1). Hydrocarbon reservoirs ar e typically found at true vertical depths (TVD) of 2,000 to 20,000ft. Reservoirs at extreme depths, nearly 30,000ft, are possible in the Gulf of Mexico.

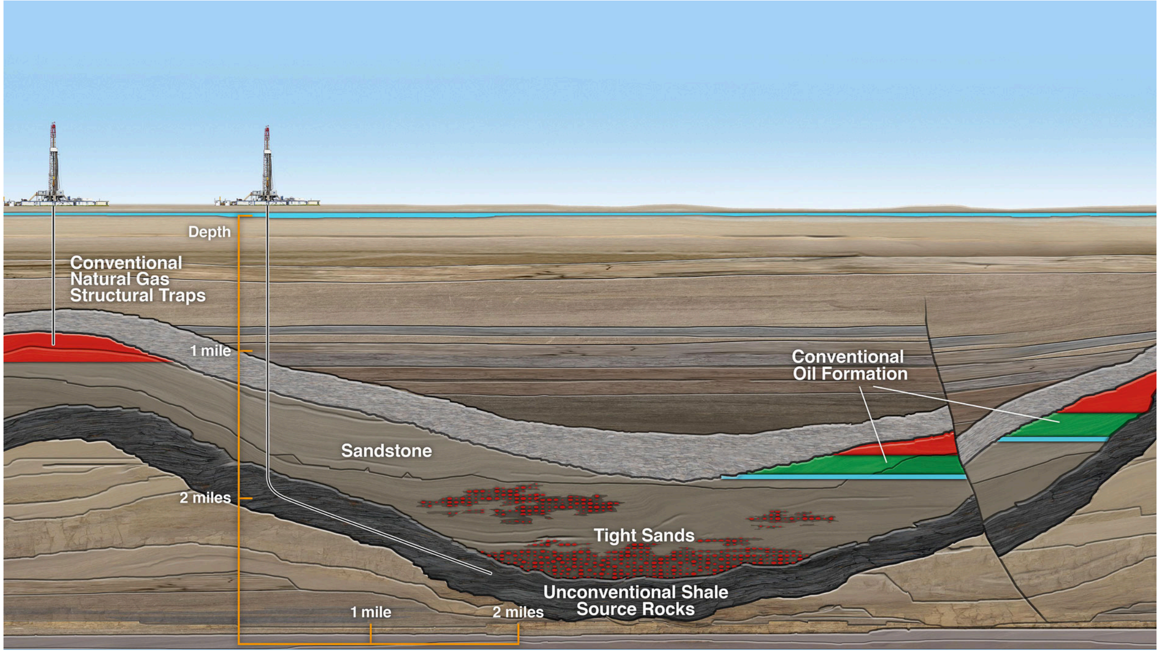

Depth
Conventional Natural Gas Structural Traps
1 mile
Conventional Oil Formation
Sandstone
2 miles
Tight Sands
Unconventional Shale Source Rocks
1 mile
2 miles

Figure 1.1 Conventional and unconventional reservoirs, vertical and deviated wells (xtoenergy.com)

An unwavering demand for hydrocarbons in today's economies has created a need to discover a means of producing them more efficiently and from new sources. To access these resources, wells are drilled to penetrate the reservoirs and provide a flow path to the surface. Expansive research and technological evolution has made it possible to drill complex ERD (extended-reach drilling) wells and produce hydrocarbons that were once unreachable. These wells often require extensive measured depths (MD), including a significant length of deviated or horizontal section, to maximize exposure to the reservoir layer. The hole bending section and horizontal lateral can add up to 30,000 feet of MD to the TVD that must be drilled.

A major concern in the construction of ERD wells is drilling torque and drag, which can be limiting factors in well design, drilling, and completion. Drilling torque and drag are the moment and pulling force, respectively, required to overcome friction as the pipe is rotated and pulled or lowered through the hole. High forces can be caused by poor, tight-hole conditions, wellbore tortuosity, sloughing shale, key seats, differential sticking, cuttings build-up, and sliding wellbore friction. In long, horizontal wells – with the assumption of acceptable hole conditions – sliding friction is of greatest concern and is the only drag effect considered in this study.

Drilling is conducted with a drill bit that is connected to a length of steel drillstring and hoisted by a drilling rig on the surface. A drillstring consists of two parts: (1) the drillpipe and (2) the bottom-hole assembly (BHA). Most of the drill string is drillpipe of uniform diameter which connects the BHA to surface. The BHA may include a variety of specialty pipe and tools that will differ in length, weight, and diameter. For example, heavy weight drill pipe, drill collars, measurement while drilling (MWD) tools, logging while drilling (LWD) tools, down hole motors, steering tools, and plenty more. The drillstring is rotated by the rig's topdrive and transmits the rock cutting force from surface to the bit. The drillstring can be miles long, and the drilled borehole is often tortuous.

To improve drilling efficiency, it is crucial to understand the dynamics of the moving drillstring and the forces that act upon it. We must also understand the drilling control parameters and their limits, including surface rotational speed, rotary torque, axial loading of the string, and mud flow rate. Then, we can create models to help understand and predict the behavior of the drillstring as certain conditions are encountered. An important modeling application, among many, is determining the amount of applied surface weight and torque that is lost to friction, and the resulting hook load.

This paper will explore the effects of torque and drag forces experienced in well construction and demonstrate how novel ***isogeometric analysis*** models, in comparison to traditional finite element models, can run faster, more efficient simulations and give improved insight to better selection of surface control inputs for improved drilling operations.

1.2 Torque and Drag Analysis

1.2.1 Application

Torque and drag analysis has become an essential practice in drilling directional wells. It is important to have an accurate idea of the torque and drag forces that will be encountered in multiple phases of well construction: (1) planning, (2) drilling, and (3) completion / post-drilling.

(1) During the well design phase, a planned well trajectory should be analyzed to determine if the drillstring and equipment intended for use will be sufficient to overcome torque and drag forces. Analysis can aid optimization of the drillstring, bottom-hole assembly, and well path. Planned trajectory data is smooth and will not be representative of the real well survey data, which will be tortuous, so it is important to account for this error when considering analysis. Despite this, models are still useful in determining if a well will be within reasonable and safe limits of torque and drag to drill.

(2) As the well is being drilled, torque and drag models can be used to analyze forces in the well. A model can help to predict and prevent drilling problems with hole cleaning, and in severe dog leg sections, before they are encountered (Aadnoy, B.S. et al., 2010). Simulation is also a preferred method for monitoring weight-on-bit (WOB) and torque-on-bit (TOB) transmission from surface to the bit because downhole measuring tools are expensive.

(3) After the well has been drilled, torque and drag models can be used for completions operations, such as running casing or coiled tubing. Analysis using the complete, real drilling data can give a better estimate of the real friction factors, which will aid in better modeling and predictions in the next well planned for drilling.

1.2.2 Modeling Approach

There are two primary modeling methods applied to drillstring analysis, and subcategories of each:

1. Analytical Method
   - Soft-String Model
   - Stiff-String Model
2. Finite Element Analysis Method
   - Traditional Finite Element Analysis (FEA)
   - Isogeometric Analysis (IGA)

1. First, ***analytical methods*** (detailed in Chapter 3) involve solving an entire system in one operation. Current drillstring analysis uses either a *soft-string* or *stiff-string* modeling approach.

In brief, a soft-string model assumes the drillstring to be like a long, loose cable that is in constant contact with the wellbore well. The drillstring is considered to be made of many connected segments for which friction calculations are made and summed from bottom to surface. The soft-string model is notorious for underestimating torque and drag, overestimating the friction coefficient, and not properly estimating side contact forces (stiff model patent).

On the other hand, a stiff-string model assumes pointwise contact with the wellbore and calculates the resulting bending stresses that the soft-string model neglects. It is considered the superior, more realistic analytical model, but the calculation is complex and time intensive.

2. Second, the ***finite element method*** (detailed in Chapter 4) is a numerical approach to solving engineering problems that is computer-based and well suited for computational automation. It is applicable to problems with complex geometries and loadings in which the analytical solution is not easily obtained. Thus, it is a preferred

method for analysis of contact, torque and drag forces in long, tortuous wellbores. In addition, the finite element approach considers realistic drillstring stiffness and other drilling complexities, and pointwise contact forces and bending stresses can be calculated. This is not easily possible with analytical methods.

Finite element analysis divides a system into smaller *elements* joined at *nodal points* (degrees of freedom) that constitute an approximate *mesh* geometry and corresponding solution space. A finite element mesh is approximated from an exact geometrical model constructed with Computer Aided Design (CAD) technology. Equations are defined and solved over each element, and combined to obtain an entire system solution. The choice of element shape and equation, which affects the resulting solution, varies. Traditional finite elements is based on interpolatory polynomials and employs the *isoparametric* c*oncept*, meaning that the solution space for dependent variables is represented in terms of the same functions which represent the geometry.

1.2.3 Current Modeling Challenges

Due to the complexity of analysis for a full-length wellbore, the finite element approach is preferred to analytical methods for drillstring torque and drag models in the field. There is, however, need to improve efficiency, accuracy, and automation of FEA models. Improvement is needed in the areas of (1) analysis / mesh generation time, (2) geometrical exactness, (3) mesh refinement, and (4) computational requirement.

(1) First, a major issue of the traditional polynomial-based FEA approach is the construction of finite element geometry (the mesh). Mesh generation is very expensive and time consuming, taking nearly 80% of overall analysis time in automotive, aerospace, and ship building industries, for example (Hughes Paper). After the initial mesh is created for analysis, any later changes made to the design will only be possible if a new mesh can be created in the available time span.

(2) Second, geometrical representations of FEA models are not exact representations of the exact CAD. This can create inaccuracies in solution. It is crucial to

acknowledge that the FEA solution is approximated and is only as accurate as the geometrical element mesh.

(3) Third, to increase accuracy and geometrical exactness, the FEA mesh can be more coarsely discretized, or refined; but, each iteration of refinement requires interaction with the original CAD. This is a hurdle to automating the modeling process. Further, the original CAD may not be available, making refinement impossible altogether.

(4) As the FEA mesh is refined and discretized into smaller elements, there are more nodal points. In other words, there are more degrees of freedom that must be solved for, which increases the computational power required. Again, this costs more time.

Work has been done to create models that have inputs for drilling control parameters and can be used for comparison to real drilling data. If the real data does not desirably match the model prediction, this is an indication that real parameters may need adjustment to achieve optimized or predicted performance (the model output). This is an important aspect of the work towards drilling automation. Ideally, we would like to develop a model that is able to, in real-time, quickly and efficiently calculate expected torque and drag under specified conditions.

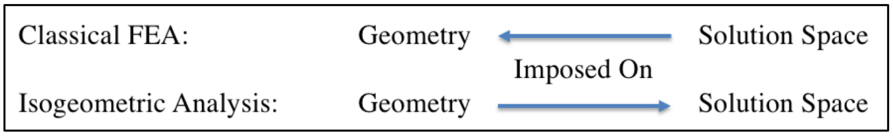
Because finite element analysis is computationally and time intensive, actual real-time models are not entirely feasible. It is feasible, however, to update a model with real-time well survey and drilling data as it is obtained, and then update predictions of torque, drag, WOB, TOB, etc., as quickly as a simulation can run. If the real measurements of these quantities differ significantly from predicted values, this indicates potential downhole problems (i.e. tight-hole or poor cleaning conditions) and operational changes may be required. Clearly, a model simulation must output results fast enough for problems to be recognized, and changes implemented, within the opportune time frame of the operation.

With sophisticated finite element analysis, it has become possible to analyze models that account for complexities of real drilling, but such detailed mesh formulations, refinement, and calculation requires an extensive amount of time, as well as human interaction with the system.

These challenges present a clear case to replace traditional FEA with an analysis technique that is more CAD-like (Hughes et al., 2009). It will be shown in the research of this thesis that the ***isogeometric*** approach to finite element analysis improves calculation efficiency, accuracy, and provides many new advantages to the endeavor towards automation.

1.3 Isogeometric Analysis

Isogeometric analysis is the concept of performing engineering analysis on exact CAD geometrical models, and it is built upon existing ideas and isoparametric concept of finite element analysis (FEA). Traditional FEA imposes its chosen basis for the solution space (i.e. interpolatory polynomials) onto the description of a known geometry, creating the approximate geometry mesh. Oppositely, IGA chooses a basis capable of representing the exact geometry (i.e. NURBS, Non-Uniform Rational B-Splines) and imposes this basis on the solution space (Figure 1.2).



Classical FEA: Geometry ← Solution Space

Imposed On

Isogeometric Analysis: Geometry → Solution Space

Figure 1.2 IGA reverses the "isoparametric arrow" (Hughes et al., 2009)

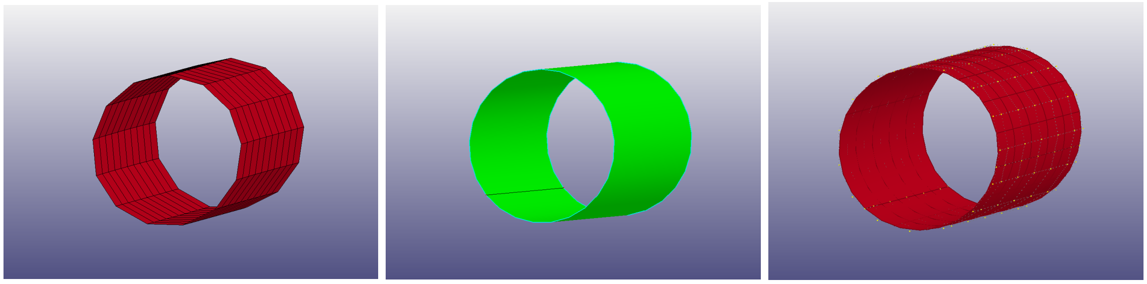


Figure 1.3 Exact geometry (center), FEA mesh (left), and IGA mesh (right)

NURBS are the same functions used to construct CAD geometries, and they can exactly represent certain geometries, such as conic shapes, with very few control points (degrees of freedom). In addition, model refinement is significantly easier because communication with the original CAD is not required, and geometrical exactness is always maintained. Contrarily, an FEA mesh requires many nodal degrees of freedom, and multiple refinements, to approach the exact geometrical shape; yet, it will never be truly exact. These concepts are demonstrated by Figures 1.3 and 1.4, which show that the IGA exact geometry has many less DOFs (yellow control points) than the FEA approximate geometry (points of element intersection).

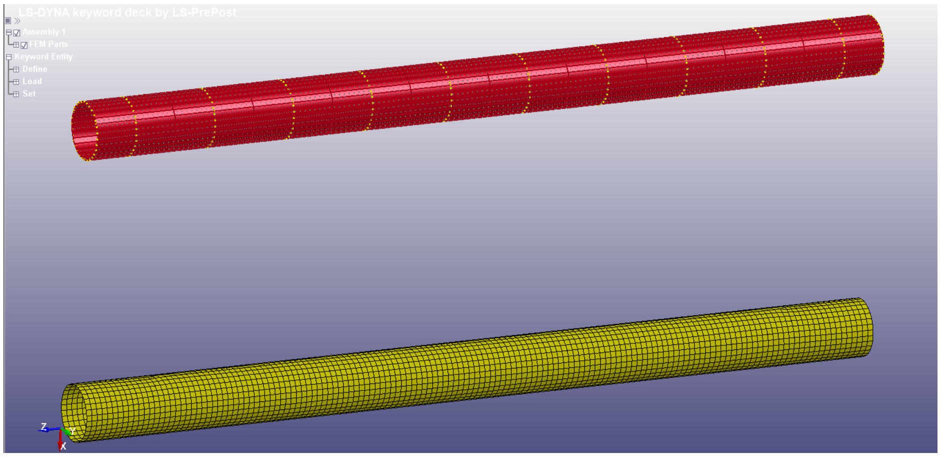
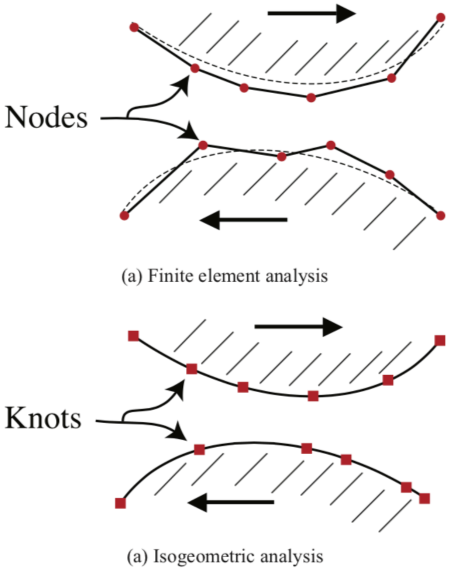


Figure 1.4 FEA mesh (bottom) requires many nodal points to approach exactness given by IGA mesh (top).

By using NURBS as the basis of analysis in IGA, analysis can be performed directly on CAD surfaces, eliminating the need for costly generation of an approximate finite element geometry (or mesh). This saves an enormous amount of meshing time and cost, and makes the overall analysis process significantly faster. The development of IGA is heralded as the missing link between FEA and CAD that will enhance analysis capabilities and allow for more automated and efficient models.



(a) Finite element analysis

(a) Isogeometric analysis

Figure 1.5 Sliding contact modeled with (a) finite element analysis and (b) isogeometric analysis.

Figure 1.5 illustrates how the geometrical inexactness of faceted polynomial finite elements can create problems in sliding contact models. NURBS geometries of isogeometric analysis can avoid this problem by exactly representing the smoothness of real bodies (Hughes et al., 2009).

This study strives to highlight the significance of IGA to the drilling industry. Section 4.3 provides a discussion of the mathematical theory of IGA. Chapter 5 explains the software program, LS-DYNA, which is used for computational modeling and implementation of IGA. Chapter 6 develops a real well example to exhibit calculation in LS-DYNA of torque and drag using both IGA and FEA methods. The results are compared to calculations from an analytical soft-string model. The discussion will transition into an introduction of the possible future applications and work to be done for IGA.

Chapter 2: Torque and Drag Theory

Torque and drag forces are always present when the drillstring is moving and/or rotating in the wellbore. Drag is the cumulative force that is needed to raise or lower the pipe, and torque is the moment needed to rotate the pipe. For each, the amount of force required is generally proportional to the amount of friction that must be overcome to initiate pipe movement. Additionally, there is a strong correlation for high drag force with high torque.

Some of the many factors that contribute to torque and drag include (Tveitdal, 2011):

- Mud type
- Hydrodynamic viscous forces
- Formation properties
- Wellbore instability, i.e. swelling shale, tight hole, sloughing hole
- Poor hole cleaning and cuttings accumulation
- Contact surface and roughness, i.e. pipe to casing or pipe to open hole)
- Tortuosity of well path
- Doglegs
- Key seating, i.e. pockets that are worn by the drillstring into the formation wall on a bend
- Differential sticking, i.e. higher pressure in the wellbore than in formation pores
- BHA and pipe stiffness
- Drillstring weight

2.1 Friction and Friction Factor

This study will consider only the torque and drag effects from sliding (dynamic) friction that occurs when the drillstring is moving while in contact with the wellbore. The dimensionless friction factor (μ) is a function of the surface-to-surface interaction in the model and will vary depending on whether the drillstring is in contact with steel casing or the formation of the open hole. The friction factor is notably different than the "friction coefficient" in pure kinetic sliding friction (Tveitdal, 2011) and is accepted as a "fudge factor" that also lumps together other contributing factors to torque and drag forces. It is an unknown, estimated value typically ranging from 0.2 to 0.4, and several iterations may be necessary to arrive at an appropriate value.

2.2 Drag

Drag force is primarily associated with deviated and tortuous wellbores, where the drillstring rests on the side of the wellbore (Figure 2.1). In vertical wells, where there is assumed to be minimal or no contact with the wellbore, drag is neglected. Drag is a function of normal force (N), friction factor (μ), and deviation angle (φ).

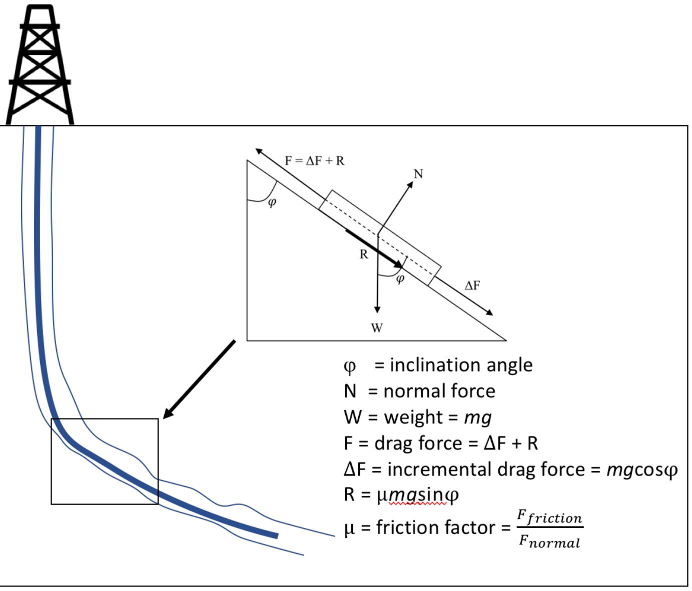


Figure 2.1 Drag in a deviated wellbore.

2.3 Torque

Torque is the moment force required to rotate the pipe and overcome friction in the rotational direction (Figure 2.2). The torque applied downhole at the bit is less than that applied at the surface because of torque loss to friction. Similar to drag, torque is associated with deviated and horizontal wellbores, and there is also assumed to be zero torque loss, except for a small amount lost to viscous mud forces, in perfectly vertical wells. Torque is a function of the coefficient of friction (μ), normal force (N), and the radius of rotation.

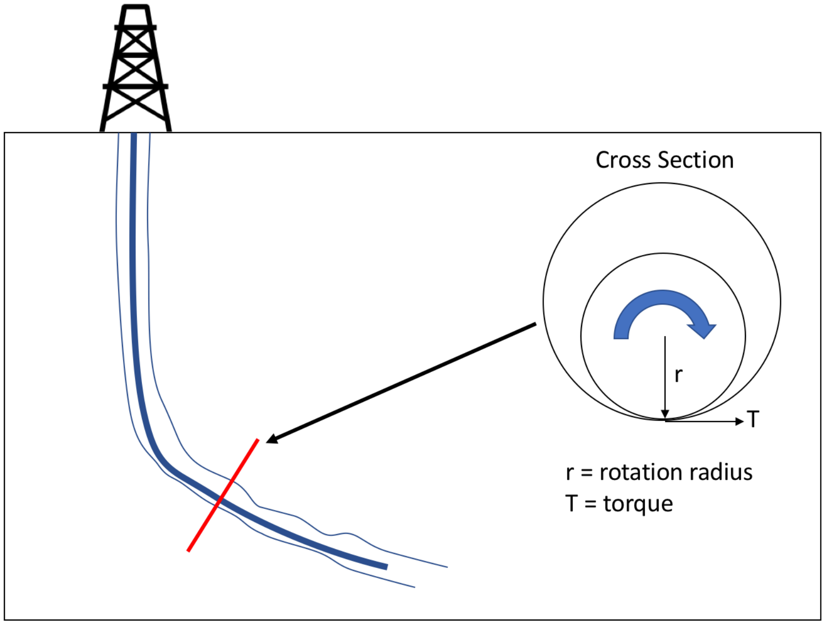


Figure 2.2 Torque in a deviated wellbore.

2.4 Resultant Force

Knowledge of the resultant force will be important for interpreting simulation output files, discussed later. The resultant force is the sum of all forces with various magnitude and direction acting on an object. This single force has the same effect on the object as all the individual forces acting together.

2.5 Buoyancy Factor

Buoyancy is the upward force exerted by a fluid that opposes the weight of an immersed object. The buoyancy factor is used to compensate for loss of drillstring weight due to immersion in drilling fluid, and it is calculated as follows:

$$BF = \frac{(65.5\ [\text{ppg}] -\ \rho_{mud}\ [\text{ppg}])}{65.5\ [\text{ppg}]}$$

Where 65.5 ppg is the density of steel.

Chapter 3: Analytical Modeling Methods

This chapter gives a condensed overview of the theory behind two analytical drillstring models: soft-string and stiff-string. Because the focus of this study is computational modeling, the intimate mathematical details are kept brief. For a full discussion of the mathematics and analytical theory, the reader is encouraged to further review the cited papers and previous works.

3.1 The Soft-String Model

The soft-string drillstring model was introduced in 1983 by Johancsik et al. to predict torque and drag in directional wells. The drillstring is assumed to be in constant contact with the wellbore, and radial clearance between the drillstring and borehole is neglected, as in Figure 3.1 It ignores the bending stresses that are realistically caused by pointwise wellbore contact, which generally leads to an underestimation of torque and drag, especially in the stiff BHA section and in tortuous or deviated hole sections.

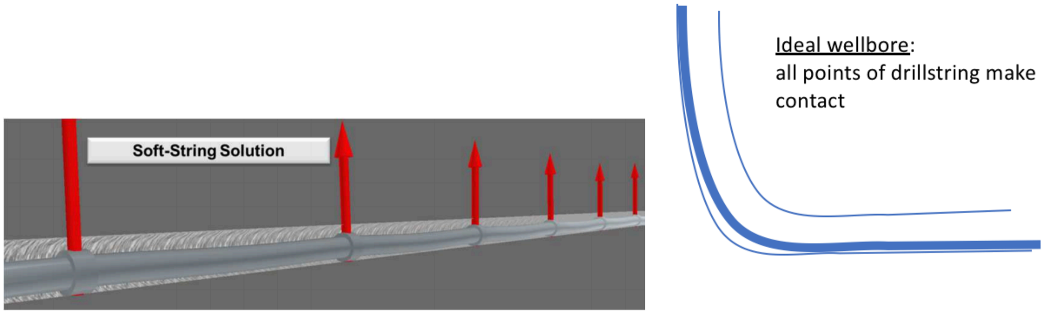


Figure 3.1 Soft-string model solution (DrillScan, 2013)

Johancsik's soft-string model assumes torque and drag to be only caused by sliding friction, which is calculated by multiplying the sidewall contact force by a friction factor. The normal sidewall contact force between the pipe and hole depends on gravity and tension acting through curvature of the wellbore.

The model divides the drillstring into many short, joined segments that transmit tension, compression and torsion. Beginning at the bottom of the string, where tension is assume to be zero when tripping in and out, the friction equation is applied to each finite section and summed towards the top of the drillstring. Each individual section contributes an increment of torque, weight, and axial drag. The maximum tension is felt at the top ofthe drillstring – at the surface.

The soft-string model solutions referenced in this study are based on the adaptation of Johancsik's original model by Bernt S. Aadnoy.

3.2 The Stiff-String Model

The stiff-string theory has been sought after by many who attempt to create a torque and drag model that accounts for bending stiffness of the drillstring, pointwise contact with the wellbore, and radial clearance. A stiff-string model is highly preferred for tortuous and deviated wellbore trajectories. It is also useful for when the drillstring is in compression and there is a risk of buckling because it can help to identify points of drillstring-wellbore contact, as in Figure 3.2 There are many approaches and theories for stiff-string modeling.

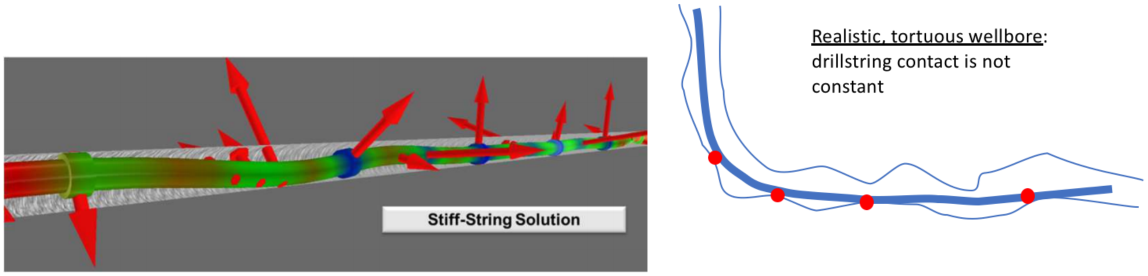


Figure 3.2 Stiff-string model solution (DrillScan, 2013)

In 1990, Hwa-shan Ho developed the stiff-string model for better analysis at the collar and BHA portions of the drillstring. The stiff-string model more appropriately considers stiffness of the drillstring, bending stresses, and can calculate pointwise contact locations and forces. The drawback, however, is that the science of the stiff-string model algorithm is much more complex, and computation time is great. Time is generally an issue with any analytical computation (DrillScan, 2013). Due to the complexities of stiff-string analytical calculations, the remainder of this study will only consider the analytical solutions of the soft-string model.

Chapter 4: Finite Element Modeling Methods (FEA and IGA)

4.1 Finite Element Analysis

Analysis is the heart of engineering design. It involves breaking down a system, object, or problem in order to understand the essential features and elements, and their relationships to each other. Finite element analysis (FEA) is a standard numerical method for solving engineering problems whose analytical solutions require complex boundary value problems for partial differential equations. FEA formulation of the problem gives a system of algebraic equations that yield approximate values of the unknowns at discrete nodal points over the domain, which has been subdivided into smaller finite elements.

FEA was first developed in the 1950s and 1960s and is a method for obtaining approximate solutions to boundary value problems that are governed by partial differential equations (Hughes et al., 2009). The finite element method divides the domain of a physical problem into a mesh of smaller sections for which analysis can be performed to determine a number of outcomes, for example, deformations, internal stresses/strains, temperature and heat transfer, or fluid flows. Interpolation of the local element analyses provides a global solution to the entire physical problem. It is important to remember that the obtained solutions to these problems are approximate, and not exact, due to the inexactness of the geometry caused by the element discretization.

The finite element mesh used in analysis is created through discretization of a CAD geometrical model, which is considered the exact geometry. The discretization elements are non-overlapping, simple shapes such as triangles, quadrilaterals, tetrahedra, etc. connected at nodal points, or the dependent degrees of freedom (DOF) at which the set of equations is solved. The equations approximate the governing equation of interest via a set of polynomial functions, also called basis functions, defined over each element (Comsol). Piecewise Lagrangian or Hermite functions are common choices for the FEA basis. The concept of using the same basis for approximation of geometry and analysis in the solution space is the "isoparametric concept" and is quite standard in classical FEA.

4.1.1 FEA Mesh Refinement Techniques

There are two notable methods of mesh refinement for finite element analysis: (1) decrease of mesh element size: *h*-refinement, and (2) increase of element order: *p*-refinement.

(1) *h*-refinement: Reducing the size of the elements is intuitively simple and can be done with a FEA meshing software. The downside, however, it takes the software an extensive amount of time to re-mesh the shape and create more, smaller elements. In addition, extensive human interaction with the software is required, as the original CAD file must be revisited, and the entire process is essentially restarted from the beginning. There are cases when the original CAD file for the model is not even available to access and there is no opportunity to refine the element mesh at all.

(2) *p*-refinement: Increasing the order of the element polynomial used in analysis does not require any alteration of the mesh in use. This method is equally disadvantageous, however, because it rapidly increases the computational expense of the analysis.

4.1.2 Convergence

Each time the mesh is refined, the solution at a point can be compared to that of the previous trial. After a number of mesh refinements, the solution will cease to show significant change, and the model can be considered converged. Of course, the user should be aware that a *converged* solution is not necessarily a *correct* solution. The use of a model is only as good as the certainty of model assumptions and inputs.

4.1.3 FEA Summary

For a more detailed review of the finite element method, one should explore the cited references. It is most important for the purpose of this study to understand the following list of FEA fundamentals.

Key Ideas of FEA:

- *Approximate* FEA mesh geometry used for analysis is generated from *exact* CAD geometry

- Mesh generation takes the majority of analysis time, a major bottleneck in the analysis process
- Mesh refinement requires interaction with the exact CAD geometry and extensive human effort
- Interpolation of basis functions finds *approximate* solution at element nodes

4.2 Isogeometric Analysis

4.2.1 Isogeometric Analysis: Integration of CAD and FEA

We begin with a brief history and intent of IGA development, and relate the significance of its abilities to drilling research. It is shown that the IGA approach can increase accuracy, save time and cost, reduce human effort, and be a more efficient, automated method for modeling torque and drag. Future work may demonstrate that IGA is also efficient in analyzing drillstring dynamics.

Computer-aided design (CAD) models and finite element analysis (FEA) consider the same engineering designs and objects, yet they have entirely independent backgrounds. They each adopt the isoparametric concept, but they also each represent objects with different geometrical constructs (Hughes et al., 2009). The pioneer of isogeometric analysis, Tom Hughes, believed that it would be possible to defeat this barrier by finding a single geometrical description for analysis. In other words, he thought that it might be possible to perform analysis directly on the exact geometric framework of CAD technologies, removing the need for generation and use of an approximate finite element geometry.

Hughes strived to find a method which could maintain geometric exactness however coarse the discretization (hence, the term "isogeometric") and eliminate need for communication with CAD geometry once an initial mesh has been constructed. This is achieved by making several changes to classical finite element analysis, particularly to the basis functions. The choice of basis function used in IGA can vary, but NURBS (Non-Uniform Rational B-Splines) are selected because they are the same basis of CAD. Table 4.1 gives a comparison of several features inherent to traditional FEA and IGA.

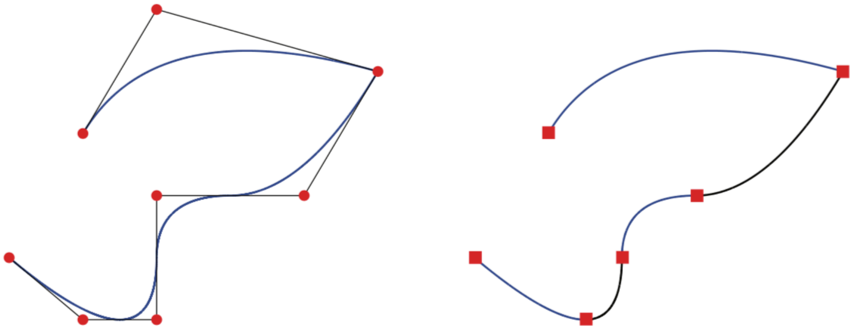
FEA	Both	IGA
Nodal points		Control points
Nodal variables		Control variables
Mesh		Knots
Elements		Patches
Basis interpolates nodal points and variables		Basis does NOT interpolate control points and variables
Approximate geometry		Exact geometry
	Compact support	
	Partition of unity	
	Isoparametric concept	

Table 4.1 Comparison of FEA and IGA Features (Hughes et al., 2009).

Again, because this study utilizes computational implementation of IGA methods, a complete understanding of mathematical theory is not necessary, but a condensed overview is highly useful. The following sections will explain the fundamentals and brief mathematics of the basis functions used in isogeometric analysis, progressing from B-splines to NURBS, and highlight the benefits they provide for better integration of CAD and FEA. For an in-depth discussion of mathematics and numerical methods with NURBS, please reference the complete work of Hughes and his colleges: "Isogeometric Analysis: Toward Integration of CAD and FEA." The following summary of IGA mathematics and included figures are borrowed from his quality works.

4.2.2 B-Splines

NURBS are built from B-splines ("basis-splines"). A "spline" in mathematics is a piecewise polynomial parametric curve. Any spline function of a certain degree can be represented by a linear combination of B-splines of that same degree. Figure 4.1 illustrates an example B-spline curve in $\mathbb{R}^2$ built from quadratic basis functions.



(a) Curve and control points

(b) Curve and mesh denoted by knot locations

Figure 4.1 Example B-spline cure (Hughes et al., 2009)

4.2.3 Knot Vectors

B-spline parametric space is local to *patches*, rather than *elements* as in FEA. Patches can be considered *subdomains* that are comprised of many "elements", referred to in IGA as "knot spans." Knot spans are defined by a *knot vector* that in one-dimension is a set of non-deceasing coordinates that partition the parameter space and are given as:

$$\Xi = \{\xi_1, \xi_2, \dots, \xi_{n+p+1}\}$$

Where

$\xi_i \in \mathbb{R}$ is the i^{th} *knot*

i is the knot index, $i = 1, 2, \dots, n+p+1$

p is the polynomial order

(note: $p = 0, 1, 2, 3,$ etc. refers respectively to constant, linear, quadratic, cubic, etc., piecewise polynomials)

n is the number of basis functions (and corresponding control points) used to construct the B-spline curve

$k = n + p + 1$ is the number of knots

A knot vector is *uniform* if the knots are equally-spaced, or *non-uniform* if unequally spaced in the parameter space. Knots may be *repeated*, and the multiplicities of knot values have meaning for the properties of the basis. Lastly, a knot vector is *open* (standard in CAD) if its first and last knots appear $p+1$ times. Basis functions formed from open knot vectors in one-dimension are interpolatory at the ends of the parametric space interval, $[\xi_1, \xi_{n+p+1}]$, and at the corners of patches in multiple dimensions; but, they are **not** interpolatory at interior knots.

4.2.4 B-Spline Basis Functions

Definitions

For piecewise constants ($p = 0$), the B-spline basis functions are defined recursively as:

$$N_{i,0}(\xi) = \begin{cases} 1 & \text{if } \xi_i \leq \xi < \xi_{i+1} \\ 0 \text{ otherwise} \end{cases} \tag{1}$$

For greater degrees $p = 1, 2, 3, \ldots$, they are defined by the *Cox-de Boor recursion formula* (Cox, 1971; de Boor, 1972):

$$N_{i,p}(\xi) = \frac{\xi - \xi_i}{\xi_{i+p} - \xi_i} N_{i,p-1}(\xi) + \frac{\xi_{i+p+1} - \xi}{\xi_{i+p+1} - \xi_{i+1}} N_{i+1,p-1}(\xi) \tag{2}$$

Figure 4.2 is a visual example of applying equations (1) and (2) to a uniform knot vector. For $p = 0$ and 1, the basis functions are exactly the same as those for FEA. There is a difference, however, for $p \geq 2$. Reiterated in Figure 4.3, $p \geq 2$ B-spline basis functions are identical and shifted ("homogenous"), unlike the FEA functions, which differ at internal and end nodes ("heterogeneous"). This is an advantage of B-splines over FEA functions in equation solving as the order is increased.

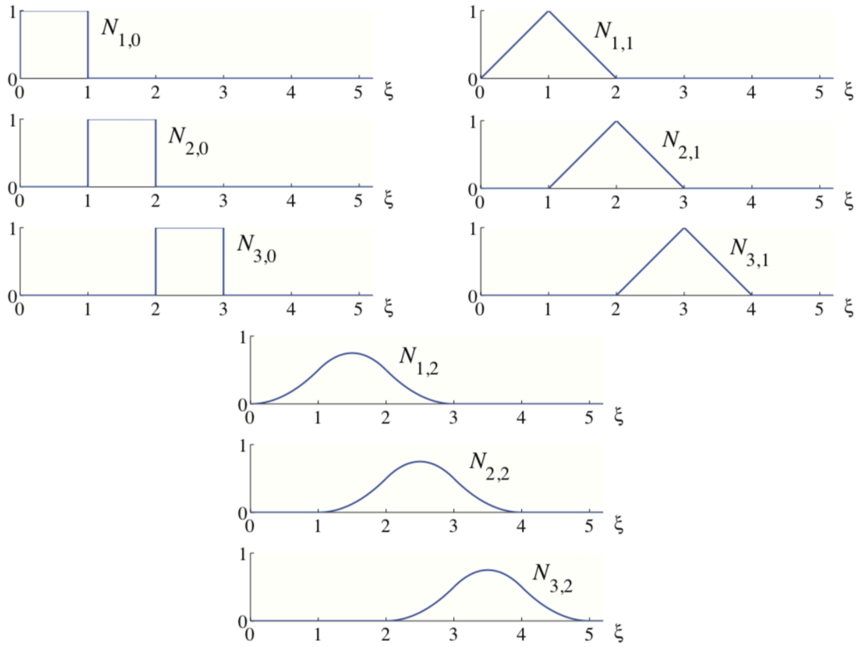
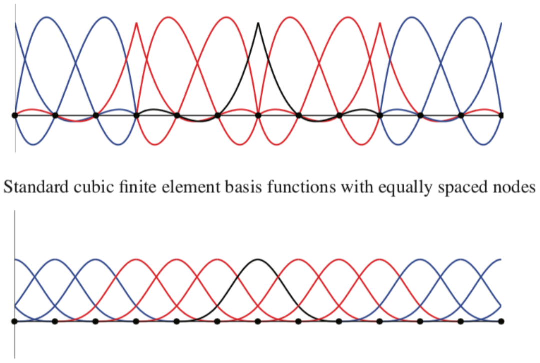


Figure 4.2 Basis functions of order 0, 1, and 2 for uniform knot vector

$\Xi = \{0, 1, 2, 3, 4, \dots\}$.



(a) Standard cubic finite element basis functions with equally spaced nodes

(b) Cubic B-spline basis functions with equally spaced knots

Figure 4.3 FEA and B-spline functions bandwidth comparison. For both, function in black has overlapping support with each function in red, and with itself.

Knot Multiplicity

Figure 4.4 is an example of quadratic basis functions for an open, non-uniform knot vector. It can be seen that the functions are interpolatory at the ends of the interval and at repeated knot ξ= 4 where continuity is C^0. Functions are C^1elsewhere.

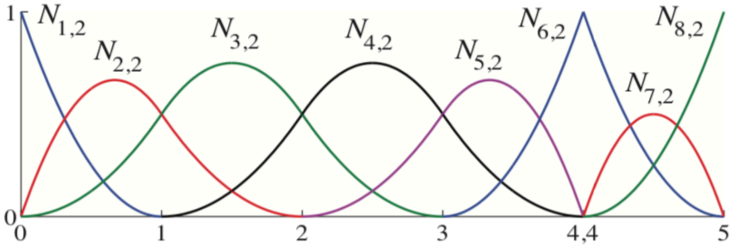


Figure 4.4 Quadratic basis functions for open, non-uniform knot vector $\Xi = \{0, 0, 0, 1, 2, 3, 4, 4, 5, 5, 5\}$.

Generally, basis functions of order p have p – 1 continuous derivatives. If a knot is repeated m times, the number of continuous derivatives is decreased by m. If the multiplicity m is equal to p, the basis function is interpolatory.

Important Properties of B-Splines

1. They constitute a partition of unity,

$$\sum_{i=1}^{n} N_{i,p}(\xi) = 1 \, , \ \forall \xi \, .$$

2. Each p^{th} order function has p - m_i continuous derivatives across element boundaries (i.e. across the knots ξ_i)

   Note: m_i is the multiplicity of the value of ξ_i in the knot vector.

   If $m_i = p$, basis is interpolatory at that knot.

   If $m_i = p + 1$, basis is discontinuous and patch boundary is formed.

3. Each basis function is non-negative, meaning that $N_{i,p}(\xi) \geq 0, \ \forall \ \xi$.

   Thus, all the coefficients of a mass matrix computed from a basis are ≥ 0.

4. Support of $N_{i,p}$ is compact and contained in $[\xi_i, \xi_{i+p+1}]$,

4.2.5 B-Spline Curves

Definitions

In a certain $\mathbb{R}^d$ (dimension space), B-spline curves are constructed from a linear combination of B-spline basis functions. *Control points*, the coefficients of the basis functions are similar to nodal coordinates in FEA. The *control polygon* is created through piecewise linear interpolation of the control points.

The piecewise-polynomial B-spline curve is given as:

$$\mathbf{C}(\xi) = \sum_{i=1}^{n} N_{i,p}(\xi)\mathbf{B}_i$$

Where

$N_{i,p}$ are *n* basis functions (i = 1, 2, . . . , *n*) of order *p*

$\mathbf{B}_i$ are control points ∈ $\mathbb{R}^d$ (i = 1, 2, . . . , *n*)

Important Properties of B-Spline Curves

1. B-spline curves have continuous derivatives of order $p - 1$ in the absence of repeated knots or control points.
2. Repeating a knot or control point by multiplicity *m* decreases the number of continuous derivatives by *k*.
3. The property of *affine covariance*, meaning that an affine transformation of a B-spline curve is obtained by applying the transformation to the control points.
4. The shape of a B-spline curve can be intuitively changed by adjusting the control points, which is also possible with NURBS (Hughes et al., 2009).

4.2.6 B-Spline Surfaces

Definitions

B-spline surfaces are similar to B-spline curves, but instead use a *control net* $\{\mathbf{B}_{i,j}\}$, i = 1, 2, . . . , n, j = 1, 2, . . . , m, and knot vectors $\Xi = \{\xi_1, \xi_2, \ldots, \xi_{n+p+1}\}$ and $\mathcal{H} = \{\eta_1, \eta_2, \ldots, \eta_{m+q+1}\}$. The B-spline surface is a tensor product defined as:

$$\mathbf{S}(\xi, \eta) = \sum_{i=1}^{n} \sum_{j=1}^{m} N_{i,p}(\xi) M_{j,q}(\eta) \mathbf{B}_{i,j,}$$

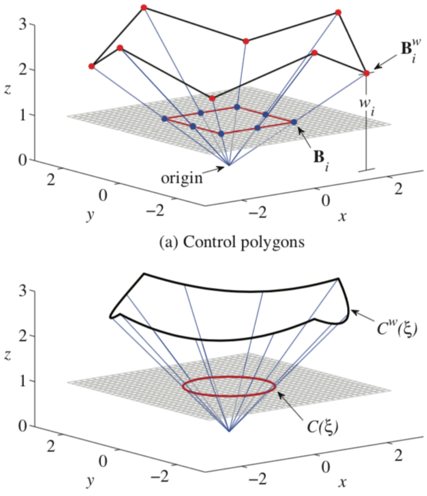
where $N_{i,p}$ and $M_{j,q}$ are the basis functions of B-spline curves.

4.2.7 NURBS: Rational B-Splines

Now that we have an understanding of the non-rational B-splines, we can discuss the construction of Non-Uniform Rational B-Splines (NURBS).

Definitions

Projective transformations of B-spline entities in $\mathbb{R}^{d+1}$ can be made to obtain geometric entities in $\mathbb{R}^d$. Conic sections (i.e. circles and ellipses) can be built exactly from projective transformations of piecewise linear quadratic curves. For instance, a circle in $\mathbb{R}^2$ can be constructed from a piecewise quadratic B-spline curve in $\mathbb{R}^3$.



(a) Control polygons

(b) Curves

Figure 4.5 Construction of a circle in $\mathbb{R}^2$ by projective transformation of a piecewise quadratic B-spline in $\mathbb{R}^3$.

As shown in Figure 4.5, the B-spline curve, $\mathbf{C}^{\boldsymbol{w}}(\xi)$, is the "projective curve" with "projective control points", $\mathbf{B}_i^w$, and the NURBS curve $\mathbf{C}(\xi)$ has "control points" $\mathbf{B}_{\boldsymbol{i}}$. The projective transformation is made by dividing the projective control points by the weights, w_i, which are the z-components of B_i^w.

Control points for the NURBS curve are obtained by the following:

$$(\mathbf{B}_i)_j = \frac{(\mathbf{B}_i^w)_j}{w_i} \qquad j = 1, \ldots, d$$

$$w_i = (\mathbf{B}_i^w)_{d+1}$$

where $(\mathbf{B}_i)_j$ is the j^{th} component of the vector $\mathbf{B}_i$ and w_i is the i^{th} weight. The weight values are generally positive and are the $d + 1$ components of projective control points in $\mathbb{R}^{d+1}$.

The *weighting function* applies the same transformation to every point in the curve:

$$W(\xi) = \sum_{i=1}^{n} N_{i,p}(\xi) w_i$$

The NURBS curve is then defined as:

$$(\mathbf{C}(\xi))_j = \frac{(\mathbf{C}^w(\xi))_j}{W(\xi)} \qquad j = 1, \ldots, d.$$

The NURBS rational basis function is given as:

$$R_i^p(\xi) = \frac{N_{i,p}(\xi) w_i}{W(\xi)} = \frac{N_{i,p}(\xi) w_i}{\sum_{\hat{i}=1}^{n} N_{\hat{i},p}(\xi) w_{\hat{i}}}.$$

Combing the above with the definition of the control points gives the equation for the NURBS curve, which has a form identical to that of B-spline curves:

$$\mathbf{C}(\xi) = \sum_{i=1}^{n} R_i^p(\xi) \mathbf{B}_i$$

The NURBS surface is then:

$$R_{i,j}^{p,q}(\xi, \eta) = \frac{N_{i,p}(\xi) M_{j,q}(\eta) w_i}{\sum_{\hat{i}=1}^{n} \sum_{\hat{j}=1}^{m} N_{\hat{i},p}(\xi) M_{\hat{j},q}(\eta) w_{\hat{i},\hat{j}}}.$$

Important Properties of NURBS

1. The NURBS basis functions form a partition of unity.
2. The continuity and support of NURBS basis functions are the same as for B-splines.
3. NURBS also possess the property of *affine covariance*, similar to B-splines.

4.2.8 IGA Refinement Techniques

The three refinement techniques used in IGA are (1) *knot insertion*, (2) *order elevation*, and (3) *k-refinement*.

(1) *Knot insertion* is a subdivision strategy analogous to *h*-refinement in FEA. The advantage of IGA knot insertion is that a curve is always geometrically and parametrically maintained. Knot values can be repeated if already present in the knot vector, but no more than *p* times, and continuity of the basis will be reduced as described in Section 4.3.4.

Given a knot vector $\Xi = \{\xi_1, \xi_2, \dots, \xi_{n+p+1}\}$ and new knot $\bar{\xi} \in [\xi_k, \xi_{k+1}]$, new $n + 1$ basis functions are generated using recursive equations (1) and (2) and new knot vector $\Xi = \left\{\xi_1, \xi_2, \dots, \xi_k, \bar{\xi}, \xi_{k+1}, \dots, \xi_{n+p+1}\right\}$.

Figure 4.6 shows a simple example of knot insertion, with original knot vector $\Xi = \{0, 0, 0, 1, 1, 1\}$. A new knot is inserted at $\bar{\xi} = 0.5$. Clearly, the refined curve is geometrically and parametrically identical to the original, but the basis functions and control points have changed. Repeating this process will add more basis functions of the same order and leave the original curve unchanged.

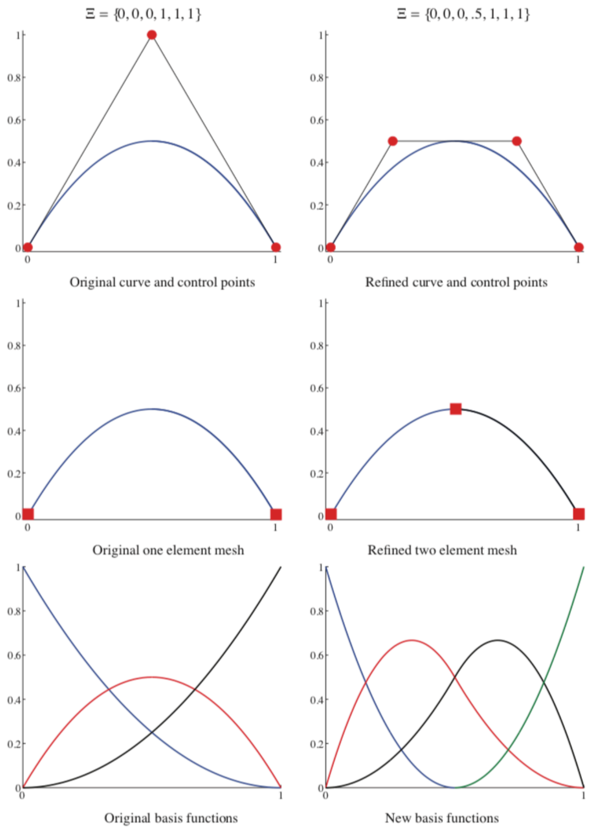


Figure 4.6 Example of knot insertion IGA refinement

(2) *Order elevation* is analogous to p-refinement in FEA. The polynomial order of the basis functions can be increased while again maintaining the geometry and parameterization of the original curve. Each knot value in Ξ must be repeated to preserve discontinuities in the p^{th} derivative of the elevated curve. The multiplicities of existing knots determines the number of new control points. The mathematical details of this process are extensive, and one is encouraged to review the work of Piegl and Tiller (21 in paper).

Figure 4.7 shows an example of order elevation, beginning with the same curve and basis functions in the example of knot insertion. Here, multiplicity of the knots is increased by one, and the numbers of control points and basis functions also increases by one. Notice again that the refined curve is geometrically and parametrically identical to the original.

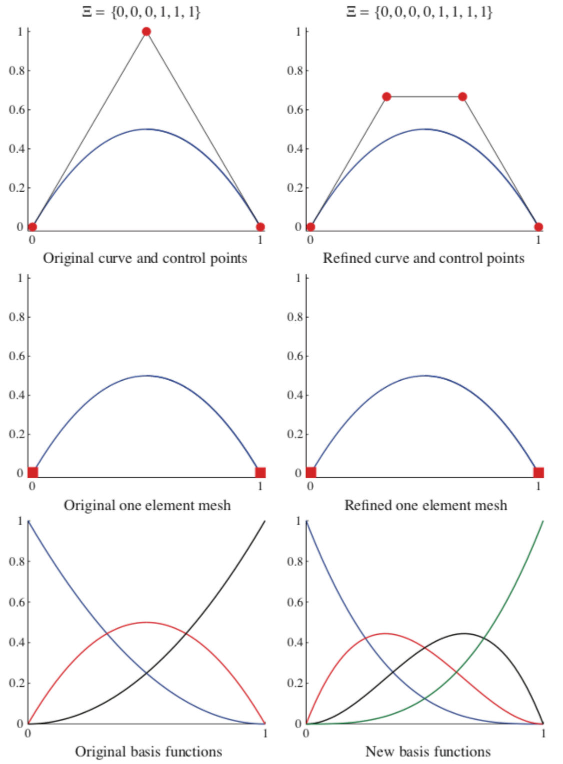


Figure 4.7 Example of order elevation IGA refinement

(3) There is no FEA analogue for *k-refinement*, which arises from the fact that processes of knot insertion and order elevation do not commute. As previously stated, a new knot value $\bar{\xi}$ may be inserted into an existing knot vector for a p order curve, and the number of continuous derivatives of the basis functions at $\bar{\xi}$ is $p - 1$. Then, if the order is elevated to q, the multiplicity of every distinct knot value in the new knot vector is increased so that discontinuities in the pth derivative of the basis are preserved (i.e. basis still has $p - 1$ continuous derivatives at $\bar{\xi}$.

Instead, if the order of the original curve was first elevated to q, and a unique know value was then inserted, the basis would have $q - 1$ continuous derivatives at $\bar{\xi}$. This process is *k-refinement*.

4.2.9 Analysis with NURBS

In his work, Hughes delivers a comprehensive list of the concepts and features that enable NURBS to serve as a base for analysis:

1. A mesh for a NURBS patch is defined by the product of knot vectors.
2. Knot spans subdivide the domain into “elements”.
3. The support of each basis function consists of a small number of “elements”.
4. The control points associated with the basis functions define the geometry.
5. The isoparametric concept is invoked, that is, the fields in question (e.g. displacement, velocity, temperature, etc.) are represented in terms of the same basis functions as the geometry. The coefficients of the basis functions are the degrees-of-freedom, or *control variables*.
6. Mesh refinement strategies are developed from a combination of knot insertion and order elevation techniques. These enable analogues of classical *h*-refinement and *p*-refinement methods, as well as the new possibility of *k*-refinement.
7. Dirichlet boundary conditions should be applied to the control variables. Homogenous Dirichlet conditions result in exact, pointwise satisfaction. For the case of inhomogeneous Dirichlet boundary conditions, the boundary values must be approximated by functions lying within the NURBS space.

4.2.10 IGA Summary

This concludes the discussion of background mathematics for isogeometric analysis, and we proceed next to discuss the computational employment of IGA for experiments with drillstring analysis.

Key Ideas of IGA:

- *Exact* CAD geometry used as analysis solution space
- No need to spend time generating FEA mesh from CAD geometry
- Use of *exact* CAD solution space increases accuracy of solution

- Refinement is simply effective and does not require interaction with original CAD geometry model or extensive human effort
- Eliminating the mesh generation time bottleneck and need for redundant human oversight provides opportunity to automate the torque and drag modeling process

Chapter 5: LS-DYNA Simulation

5.1 About LS-DYNA and LSPrePost Programs

To directly compare results of the traditional FEA and IGA methods, we employ the LS-DYNA general-purpose finite element program, developed by Livermore Software Technology Corporation (LSTC). LS-DYNA is unique in that it has been developed for use of isogeometric analysis. Therefore, the same physical problem can be built in LS-DYNA and modeled with both FEA and IGA. This leads to a very straightforward comparison of the methods in simulation convergence, accuracy, and run time.

In addition to monitoring result convergence, accuracy of the computational solution will be reference checked with the analytical model solution. Several adjustments, discussed later, are made to the LS-DYNA model to account for soft-string model assumptions and allow for better comparison of results.

Hypothetically, the contact feature of LS-DYNA simulation, and inclusion of material part stiffness, will naturally account for the effects of bending stiffness, similar to a stiff-string model. This is why input adjustments are made to "trick" LS-DYNA into acting more like a soft-string model. Contact in LS-DYNA searches for penetrations, using any of a number of different assigned algorithms, of a defined "slave" part or node through a defined "master" part or segment. The resulting contact force and resultant forces of the drillstring are calculated.

LSPrePost is the pre/post-processing program for LS-DYNA. Its user interface is designed to be efficient and intuitive. In pre-processing, LSPrePost aids geometry creation, meshing, and model visualization. It provides an interactive, convenient way of creating the LS-DYNA keyword input file. In post-processing, LSPrePost can display output files, plots, and animated visual results. A screenshot example of the LSPrePost interface is provided for reference in Appendix A.

5.2 Modeling Workflow

Prior to developing a torque and drag model, one should at least have knowledge of the following:

- Drillstring / BHA specifications
- Wellbore geometry / survey data
- Set casing depths and diameters
- Control parameters
- Drilling operating parameters (i.e. tripping and rotating speeds)
- A consistent units system

5.2.1 Building CAD Geometry Models

First, CAD geometry models must be created for all analysis components: the wellbore and drillstring.

Wellbore Geometry

- Construct 3D wellbore geometry model using a CAD software (Autodesk Inventor or LS-DYNA geometry creation tools were both used)
  - Import well survey data points, connect points to build a spline curve
  - Create a circle of desired wellbore radius at the top survey point, in a plane perpendicular to the z-axis (or axial direction of wellbore)
  - Use a "sweep" tool to sweep the circle along the spline curve and create a cylindrical 3D wellbore geometry
- If casing has been installed in portions of the wellbore, multiple wellbore diameters may be used following the same "sweep" procedure along each hole portion
- Save and export geometry as an IGES file

Drillstring Geometry

The drillstring geometry is more sophisticated and complex to construct. It will have a length of uniform diameter, regular drillpipe from the surface down to the BHA, which may be any combination of various length and diameter drill pipe, heavy weight drill pipe (HWDP), drill collars, various drilling tools, and a bit on bottom. Figure 5.1 is an example BHA and bit CAD geometry model of the small-scale “nano-rig” created by UT Austin PGE.

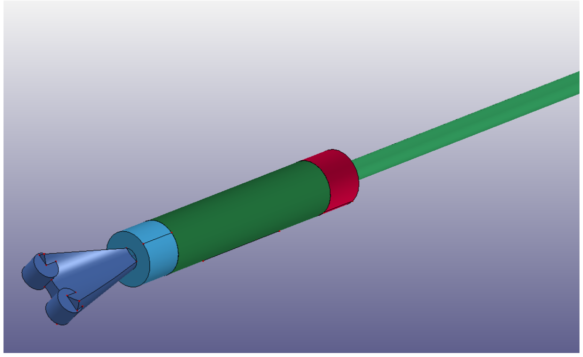


Figure 5.1 Example CAD model of UT Austin PGE nano-rig.

5.2.2 Creating LS-DYNA Keyword Files

The LS-DYNA simulation is run from a keyword file input, which organizes the database by grouping similar functions under the same “keyword” card. An example abbreviated keyword file is included in Appendix C. To begin, open a new LSPrePost window and save as a *.k file. The following steps describe the general procedure for using the LSPrePost interface to build a keyword input file.

STEP 1: Import IGES (preferred) or STEP CAD geometry files

STEP 2: Create and Refine Finite Elements

Here, choose whether to conduct a standard FEA or IGA simulation and continue with the following relevant procedure.

FEA Standard Elements

- *To Create:* There are several methods for creating standard FEA elements in LS-DYNA. The simplest, used in this study, is the “Auto Mesher”. Choose a geometry object and use the dialogue box for the Auto Mesher feature to select mesh mode, mesh type, element size, tolerance, and other options. Once the mesh has been generated for the object, you may click to “accept,” or “reject” to retry with different mesh parameters.

- *To Refine:* The Auto Mesher has a mesh mode option to "remesh", which can be used to create a new mesh of different element size. The *ELEMENT_ and *SECTION_ keyword cards can also be edited directly to change the polynomial order of the element. IMPORTANT: To create a new mesh, the original geometry IGES file must be re-imported.

IGA Elements

- *To Create:* To create IGA NURBS elements in LS-DYNA, select the "Create" option in the "NURBS Editor" window. Choose a geometry object, set tolerance and other preferences, and apply. You may "accept" the generated elements or "reject" to try again.

- *To Refine:* Select the "Refine" option from the same NURBS editor window. Here, you can directly select the method and parameters of refinement (h-, p-, or k-refinement). IMPORTANT: The original geometry IGES file is *not* needed for IGA mesh refinement.

STEP 3: Assign Part Materials and Properties

Each geometry entity is assigned a "part" keyword identification once meshed. The part keyword is used to assign "section" and "material" keyword properties.

- Section: Defines element formulation, integration rule, nodal thickness, cross sectional properties.
  - *SECTION_SHELL keyword is used to specify IGA element formulation (ELFORM = 201). Many different element formulations can be selected for FEA. Default is Belytschko-Tsay (ELFORM = 2).

- Material: Defines the material properties.
  - *MAT_ELASTIC keyword (available for beam, shell, and solid LS-DYNA elements) is used to model the steel material for the drillstring and

outer hole wall, assumed to be steel casing. Relevant variable inputs are mass density, Young's modulus, and Poisson's ratio.

STEP 4: Define Solution Control and Output Parameters

- Control: Used to change defaults and activate solution options, such as implicit solution.
  - *CONTROL_TERMINATION must be included to define termination time of the simulation job.
  - *CONTROL_IMPLICIT_AUTO, *CONTROL_IMPLICIT_GENERAL, and *CONTROL_IMPLICIT_SOLUTION are used to invoke and define parameters for an implicit simulation.

- Database: Necessary to define output files containing results information.
  - *DATABASE_ASCII_option defines the following output files:
    - GLSTAT: Global data.
    - NCFORC: Nodal interface forces.
    - NODFOR: Nodal force groups.
    - NODOUT: Nodal point data.
    - RCFORC: Resultant interface forces.
    - SLEOUT: Sliding interface energy.
  - *DATABASE_BINARY_D3PLOT adds the part, material, section, and hourglass data to the first d3plot file, which is a database for the entire model that can be viewed in post-processing.
  - *DATABASE_BINARY_INTFOR is the contact interface database. It produces post-processing visuals of contact results.
  - *DATABASE_HISTORY_NODE_SET defines the set of nodes to be included in the output files.

STEP 5: Define Loads, Boundary Conditions, and Contact Parameters

- Set Data: Defines groups of nodes, parts, elements, etc.
    - *SET_NODE_LIST allows user to define sets of nodes to which boundary conditions can be applied.

- Define: This keyword provides a way to define boxes, coordinate systems, load curves, tables, and orientation vectors for use in the model.
    - *DEFINE_BOX can define a box-shaped volume that can be used for various specifications for a variety of inputs, such as velocities, contact etc.
    - *DEFINE_COORDINATE_SYSTEM can define a local coordinate system.
    - *DEFINE_CURVE defines a load curve (i.e. load versus time).
    - *DEFINE_VECTOR can define a vector that can be used to assign direction to a prescribed motion.

- Boundary: Defines imposed motions on boundary nodes or sets of nodes.
    - *BOUNDARY_PRESCRIBED_MOTION_SET_BOX is used to prescribe motion to a part, node, or set of parts or nodes within a defined box volume.
    - *BOUNDARY_SPC is used to apply translational constraints to parts or nodes.

- Contact: Defined by identifying (via parts, part sets, segment sets, and/or node sets) locations for potential penetration of a slave node through a master segment.
    - *CONTACT_AUTOMATIC_NODES_TO_SURFACE
    - *CONTACT_AUTOMATIC_ONE_WAY_SURFACE_TO_SURFACE defines surfaces of contact and contact parameters, such as friction factors.

- Load (Gravity)
    - For FEA: *LOAD_GRAVITY_PART
    - For IGA: *LOAD_BODY_PARTS, *LOAD_BODY_Z

STEP 6: Run Simulation

When the keyword file has been completely and properly formatted, it can be input to run the LS-DYNA simulation. LS-DYNA can be invoked directly from LSPrePost with File → Run LS-DYNA. Results will be saved to the location specified in the pop-up window. The amount of time required to complete the simulation will depend on the problem's level of refinement and corresponding number of degrees of freedom. If any errors are encountered in the keyword or model set-up, you will be informed in the run window and the job will be terminated.

STEP 7: Post-Processing Results

When normal termination is reached for the simulation, two important output files can be opened for viewing in LSPrePost: the "d3plot" and "intfor". These files show animated visuals of the model results and allow for plotting of various output data, i.e. stresses, strain, pressure, displacement, forces, etc. In the LSPrePost window for d3plot and intfor files, the user can also call other output files (i.e. "rcforc" and "nodout") to view results data and create plots. These files can also be opened in a text editor and imported to Microsoft Excel for additional processing.

5.3 Consistent Units

The LS-DYNA user is advised to maintain a consistent set of units when defining a model. There is no function in the program to automatically define a units system.

MASS	LENGTH	TIME	FORCE	STRESS	ENERGY	DENSITY	YOUNG's	35MPH 56.33KMPH	GRAVITY
Kg	M	s	N	Pa	J	7.83e+03	2.07e+11	15.65	9.806

Table 5.1 LS-DYNA consistent units

Chapter 6: Model Problem

6.1 Problem Statement

In this chapter, the following theory and procedures are implemented to develop a real well example and prove the superiority of IGA over FEA in drilling industry models. This example will model the drag force, also referred to as frictional sliding energy, experienced by a drillstring of realistic length and diameter that is lowered into the build section of a deviated wellbore. The simulation will output the resultant force experienced by the drillstring, which is equivalent to the hook load felt by the surface equipment.

In realistic practice, this type of model would be useful in every phase of well construction:

1. Planning: Ensure that the drillstring and BHA equipment planned for use can overcome drag to drill, lower in, and pull out of the extended lateral section.
2. Drilling: Monitor drag forces as drilling progress, with updated survey data, to determine if actual well conditions are similar to those predicted in the planning stage. If not, analyze to see if there is a problem or if operation parameters can be changed to improve drilling performance. Also, ensure ability to install casing.
3. Completion: Ensure that any casing or completion tools, such as coiled tubing, can be run as far as necessary in a lateral section.

Assumptions: Good hole conditions are assumed, and gravity is the only other load considered to act on the drillstring. Initially, the pipe is not rotated and torque is not considered.

6.2 LS-DYNA Licensing and Computer Processing

The single computer used in this study is a Dell Latitude with Intel Core i7. There are several software products and licenses available to LS-DYNA users. For this study, LSTC provided a Node-Locked License, which permits the use of LS-DYNA on a single machine with use of up to 50,000 elements in simulation. The limited element number and use of a single computer processing unit (CPU) in the study leaves room for simulation improvement, discussed in the following.

6.3 Well Survey Data

Apache Well 2408

Survey data from real, deviated wells (both planned and actual) was obtained from Apache Corporation. Apache is an oil and gas exploration and production company with operations in the U.S., Egypt, and the United Kingdom North Sea.

This study focused on "Well 2408" (Figure 6.1) because it is a typical "build-and-hold" well, meaning that it deviates steadily from 0 to 90 degrees and remains at 90 degrees for the entire lateral section. It is located in the Permian Basin of the United States. Comparison of the planned and actual survey data clearly shows that the real wellbore encounters significant tortuosity and deviates from the perfect route that is intended.

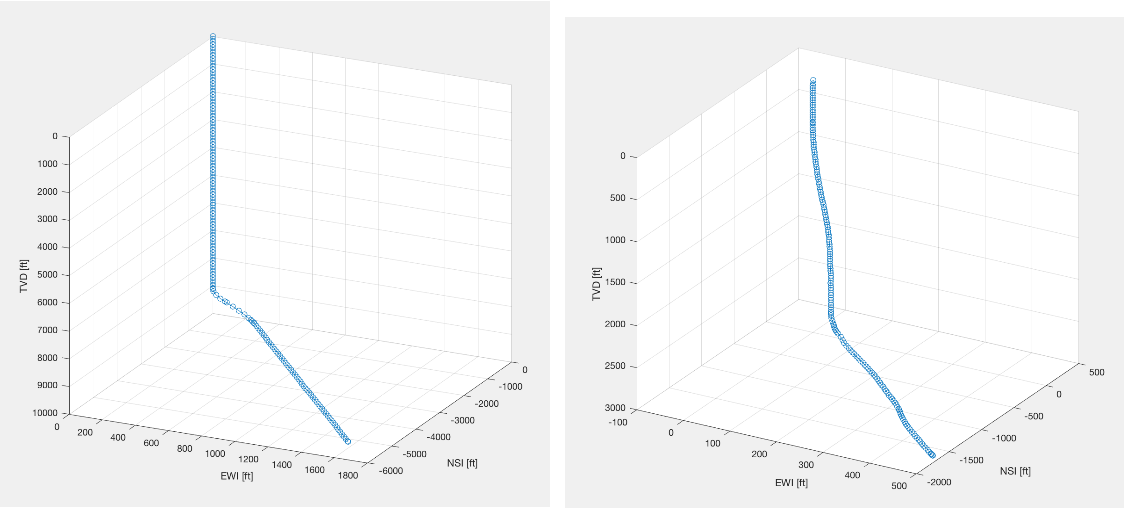


Figure 6.1 Apache well 2408 planned (left) vs. actual (right) complete survey data.

Apache Well 2408	
MD	15,202 ft
TVD	9,738 ft

Table 6.1 Apache well 2408 data.

The actual depths and diameters of set casing were altered to fit the purpose of this analysis, explained later, and are not representative of the real well. A uniform wellbore diameter of 10” (0.254 m radius) is used.

Apache Well 2408 Build Section

Clearly, it is desirable to study the results of a torque and drag analysis for an entire well; however, full well analysis is computationally intensive. The single CPU made accessible to this study was not ideally suited to handle such a complex computation (memory was limited and run time was excessive). Instead, a portion of the build section of the well, where torque and drag forces are most significant, was chosen for analysis. The results of a shorter well segment analysis are sufficient to demonstrate the difference and superiority of IGA to the FEA and analytical solutions.

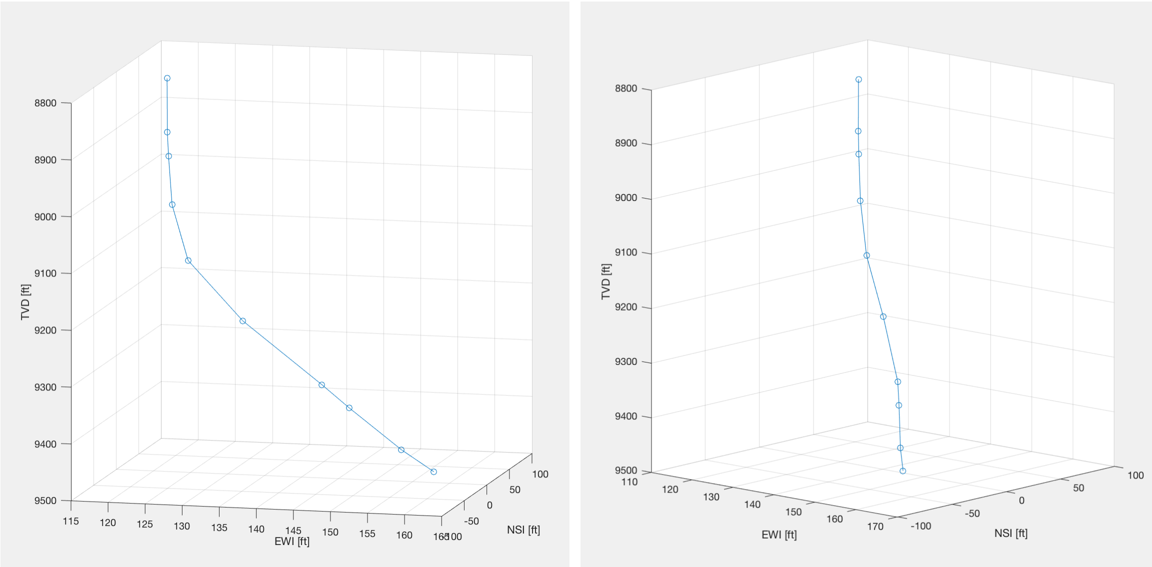


Figure 6.2 Apache well 2408 build section (two views)

Apache Well 2408 (Build Section)	
MD	8,845 – 9,478 ft
TVD	8,845 – 9,444 ft
Radius	0.254 m

Table 6.2 Apache well 2408 data used for simulation.

6.4 Drillstring Data

Apache also provided real drillstring and BHA data, including the lengths and properties of dill pipe, drill collars, and heavy-weight drill pipe used. Again, alterations we made to the drillstring input data in order for this study to clearly compare IGA and classic FEA computational analyses to the analytical analysis.

Recall that the soft-string model assumes constant contact of the drillstring and wellbore wall. To emulate this assumption in LS-DYNA, a uniform diameter drillstring is used with intentionally decreased stiffness properties, as shown in Table 6.3.

Steel Drillstring Properties	
Radius	0.2032 m
Density	7,850 kg/m^3
Young's Modulus	1 x 10^9 Pa
Poisson's Ratio	0.33

Table 6.3 Drillstring inputs used in simulation

6.5 Other Parameter Inputs

Listed below are other miscellaneous parameter inputs used to construct the drag model.

- **Friction Factor:** 0.3 (Steel to steel surface)
- **Termination Time:** 600 seconds
- **Tripping Speed (into hole):** 0.3 m/s
- **Gravity Acceleration:** 9.8 m/s^2

Chapter 7: Results

This chapter presents the details and results for each of 6 trial simulations run in LS-DYNA. Three trials of increased refinement (increased DOFs) were completed for both the FEA and IGA methods. Trials we completed over the build section of the well, described in Chapter 6. A segment of approximately 25m, which had the least output noise, was selected and used for best comparison of all trial results.

Remember, the stiffness parameters in the LS-DYNA model were lessened to achieve a better comparison and representation of the assumptions made by the analytical soft-string model. Also remember that the soft-string analytical model itself is not necessary a real, exact solution, because all models are an approximation. Despite this, it is widely accepted by the industry as an accurate model for torque and drag calculation and is therefore accepted for use here as a reference solution for the results of IGA and FEA simulations.

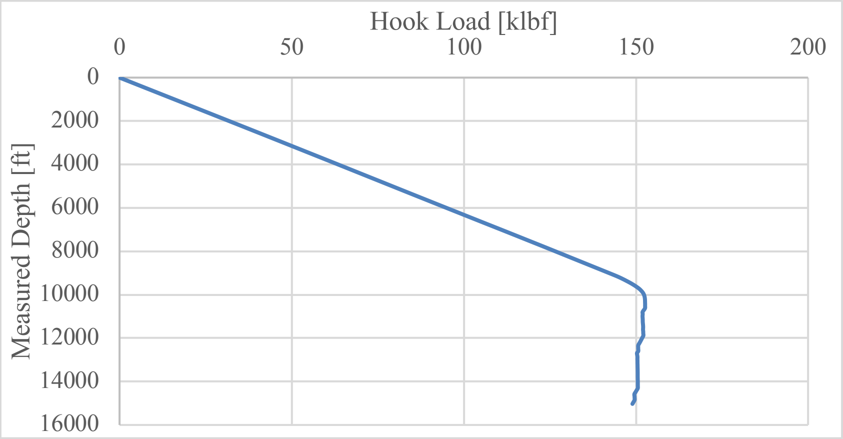


Figure 7.1 Soft-string analytical model results for entire Well 2408 (DeAngelo, 2017).

In LS-DYNA, results are output in the designated "rcforc" file, which gives data of the resultant force (in Newtons) versus time (in seconds), or pipe displacement (in meters). The resultant force is equivalent to the hook load force (drag force), which is the force needed to overcome all friction and gravity forces and move the pipe. To compare to the analytical results, the LS-DYNA output is converted to resultant force (in klbf) versus measured depth (in feet). See Appendix B for a sample plot of hook load (resultant force) versus MD. The average hook load force was taken along the analyzed segment for both the analytical and LS-DYNA experimental results, and the apparent error was found as follows:

$$\text{Apparent Error} = (\text{Experimental} - \text{Analytical})/\text{Analytical}$$

Trial	**FEA 1**	**FEA 2**	**FEA 3**
# Elements	9706	12190	27082
# Nodes	9718	12204	27103
Memory required	6321K	9407K	11000K
Run time	34743 sec	53656 sec	132127 sec
Polynomial Degree	2	2	2
Element size	0.3	0.2	0.1
Error	0.16364	0.11576	.02032

Table 7.1 FEA trial results

Trial	**IGA 1**	**IGA 2**	**IGA 3**
# Elements	2	2	2
# Nodes	1568	5068	8288
Subdivision	3R, 200S	3R, 700S	3R, 300S
Memory required	3831K	6579K	9307K
Run time	8051 sec	12359 sec	15053 sec
NURBS Degree	2	2	3
Error	0.36384	0.19348	0.04075

Table 7.2 IGA trial results

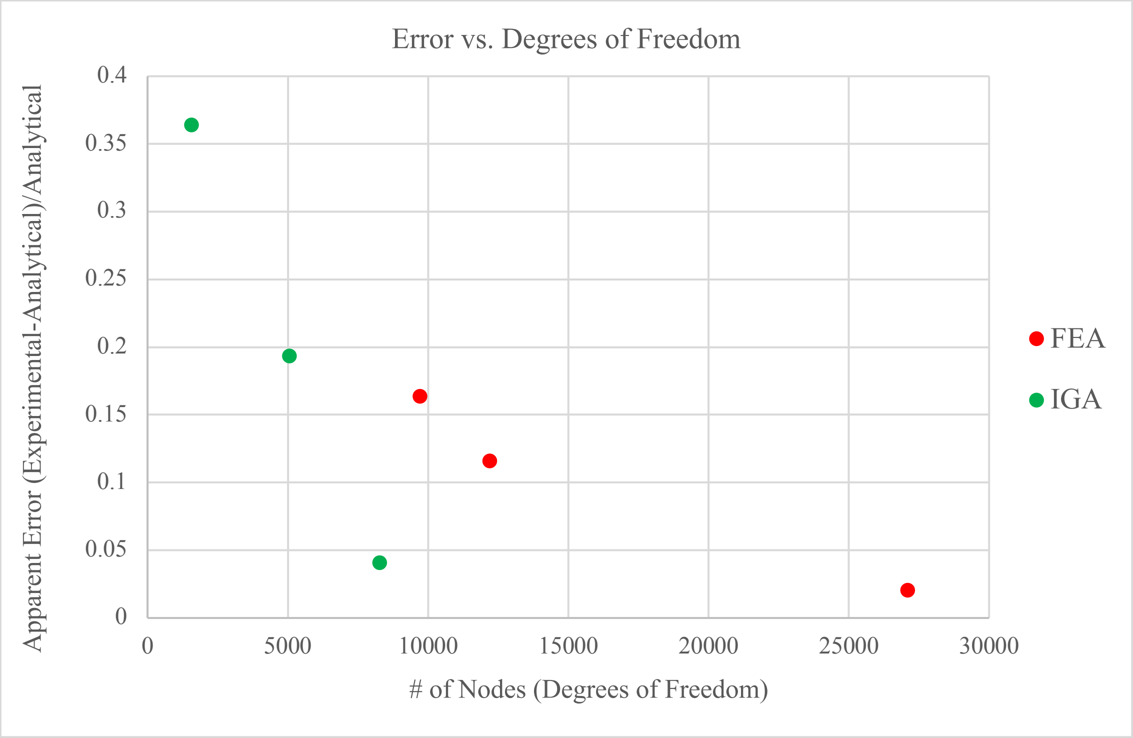


Figure 7.2 Apparent error for FEA vs. IGA with increased DOFs.

Immediately, several conclusions can be intuitively drawn from the given model data and results:

- **Convergence**: IGA convergence (arrival at correct solution) is significantly better than FEA in simulation run time, required memory and degrees of freedom.
- **Geometrical solution:** It is important to remember that each IGA trial maintains the *exact* original CAD geometry, while each FEA trial only *approximates* the CAD geometry, and requires many more nodes (DOFs) to do so.
- **Number of elements:** Note also that while the number of elements for FEA increases with each mesh refinement, the IGA method maintains only 2 elements. This is because mesh refinement for IGA is done instead through a subdivision process with knot insertion, as described in Chapter 4, which does not physically create new elements, as in FEA.
- **Finite element order:** The polynomial element order is maintained at the LS-DYNA default for the FEA method because any increase caused the simulation to exceed the computational and time limits of this study.

Chapter 8: Discussion of Results, IGA Applications, and Future Work

8.1 Implementing IGA into Current Drilling Models

As reiterated throughout this study and shown by the experimental results, IGA has immediate potential to replace traditional FEA in torque and drag modeling because IGA simulations require significantly less time and computational memory. Further, the refinement process for each IGA trial was much simpler and quicker than each FEA refinement. In order to refine the FEA model, the original CAD geometry file had to be imported again into LS-DYNA so that a new mesh could be generated. This is not necessary in the IGA models, as the original geometry is always maintained with refinement.

It is possible that IGA can be implemented into already existing FEA codes, such as those that are designed to identify points of drillstring contact with the wellbore wall. As discussed, conversion from FEA to IGA can be done by changing the basis functions used to solve for the finite element solution. This is rather simple, but certainly does require much work and skill from the finite element code programmer. However, once this change is made, great opportunity is introduced to improve efficiency and accuracy of the program solution.

8.2 Future Applications of IGA Models

Improved Finite Element Programs

The LS-DYNA application of IGA shown in this study can also be expanded for greater simulations of both torque and drag forces for entire wellbores. To do this, a full license must be obtained so that the LS-DYNA simulation can be run in parallel on multiple CPUs (computer processing units). Running in parallel lessens the computational demand assigned to each CPU, and bigger simulations can be run more efficiently. Again, this study was limited to a single CPU, which limited demonstrational potential. Torque forces can be analyzed by applying a rotational movement to the drillstring and reading the resultant force measured in the radial direction.

It is notable to mention that the use of LS-DYNA in this study was primarily for direct comparison of IGA and FEA methods. LS-DYNA is a useful finite element program for many engineering industries, but the interface is not very well suited for modeling the drilling process. A clever approach was needed to create a model of the wellbore and drillstring, and to prescribe motion to the drillstring in a way that properly simulated tripping into the hole.

Advancing the simulation to include other drilling complexities will prove to be difficult, as well. Thus, this realizes a need for a similar IGA-based finite element program that has an interface suited for easily modeling standard drilling procedures and equipment. In addition, developing a highly efficient IGA-based drilling model program will be significant in the endeavor towards automated drilling processes. Time efficient IGA models will also make it viable for torque and drag analysis to become a standard practice and duty of engineers during the routine well design and drilling process.

Drillstring Dynamics Analysis

There is great potential for applying IGA to other areas of study in the drilling industry. Advancement in data transmission from mud pulse telemetry to wired drill pipe has led to a new era of "big data", and there is ongoing study on how to best harness this newly available wealth of drilling information. A particular topic of interest is the use of this high-frequency data to detect drilling dysfunctions caused by oscillatory movements, or vibrations, of the drillstring (i.e. whirl, stick-slip, and bit bounce). These movements are associated with a host of drilling problems including ROP (rate of penetration) reduction and tool damage.

It would be ideal to have a model that is able to identify drilling problems in real-time and alter parameters as necessary to mitigate the issue. Eventually, this type of model could also be automated and able to perform immediate mitigation efforts without human supervision. This would be an incredible advantage to the safety and smoothness of the drilling process. With FEA, efficient, automated, and real-time simulation is not possible; but with IGA, it may become possible in the near future.

Chapter 9: Conclusion

The work completed in this study provides strong evidence that isogeometric analysis is a viable and improved alternative to traditional finite element methods. There is immense potential for introducing IGA into existing and developing drilling models. The real well example LS-DYNA model in this study greatly demonstrates IGA application to drag force analysis. Although the example model was run for only a portion of the entire real wellbore, the results are certainly representative and can be expanded with access to parallel CPU simulation.

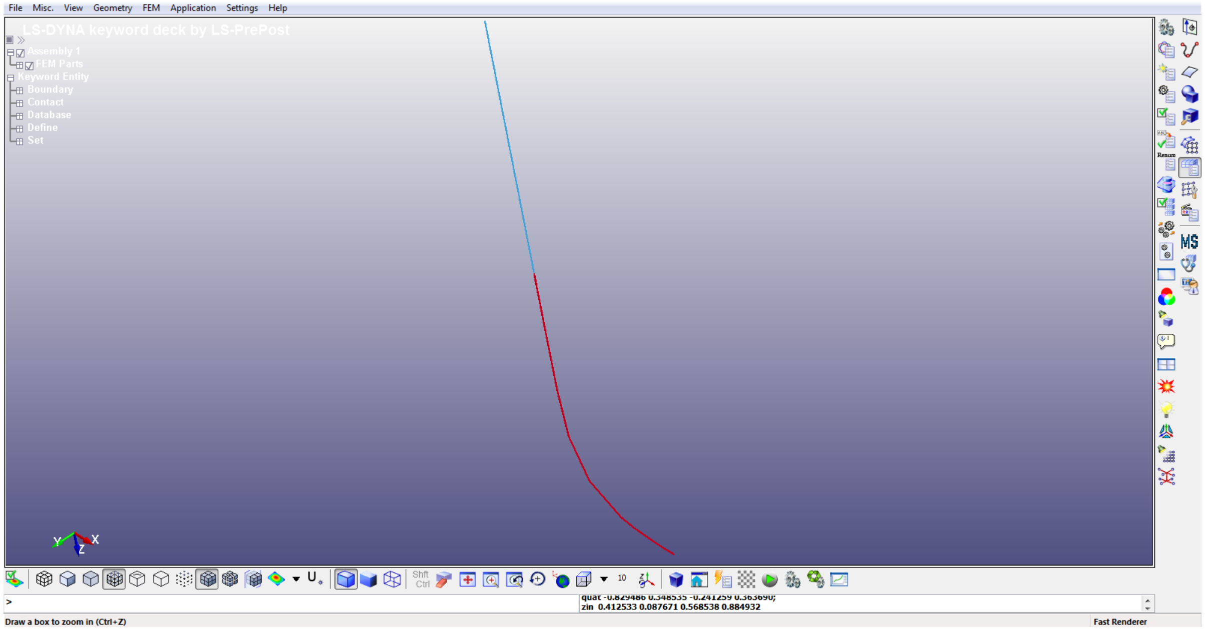
IGA methods are proven to be significantly faster and more computationally efficient than FEA methods. Saving time saves money, which is crucial in times of low oil price, when efficiency is needed the most. Even more important, the potential for IGA to improve automated drilling is an invaluable success for safety improvement in the industry.

Appendix

A. LS-DYNA LSPrePost Graphic User Interface

The following image is a screenshot of the LSPrePost modeling program. This user interface provides a visually interactive way to build the LS-DYNA keyword file that is used to run the simulation. The sidebars contain various tools that can be used for geometry and mesh creation, and to define all other parameters of the model. For greater detail, please visit the well-developed manuals and tutorials provided by LSTC.

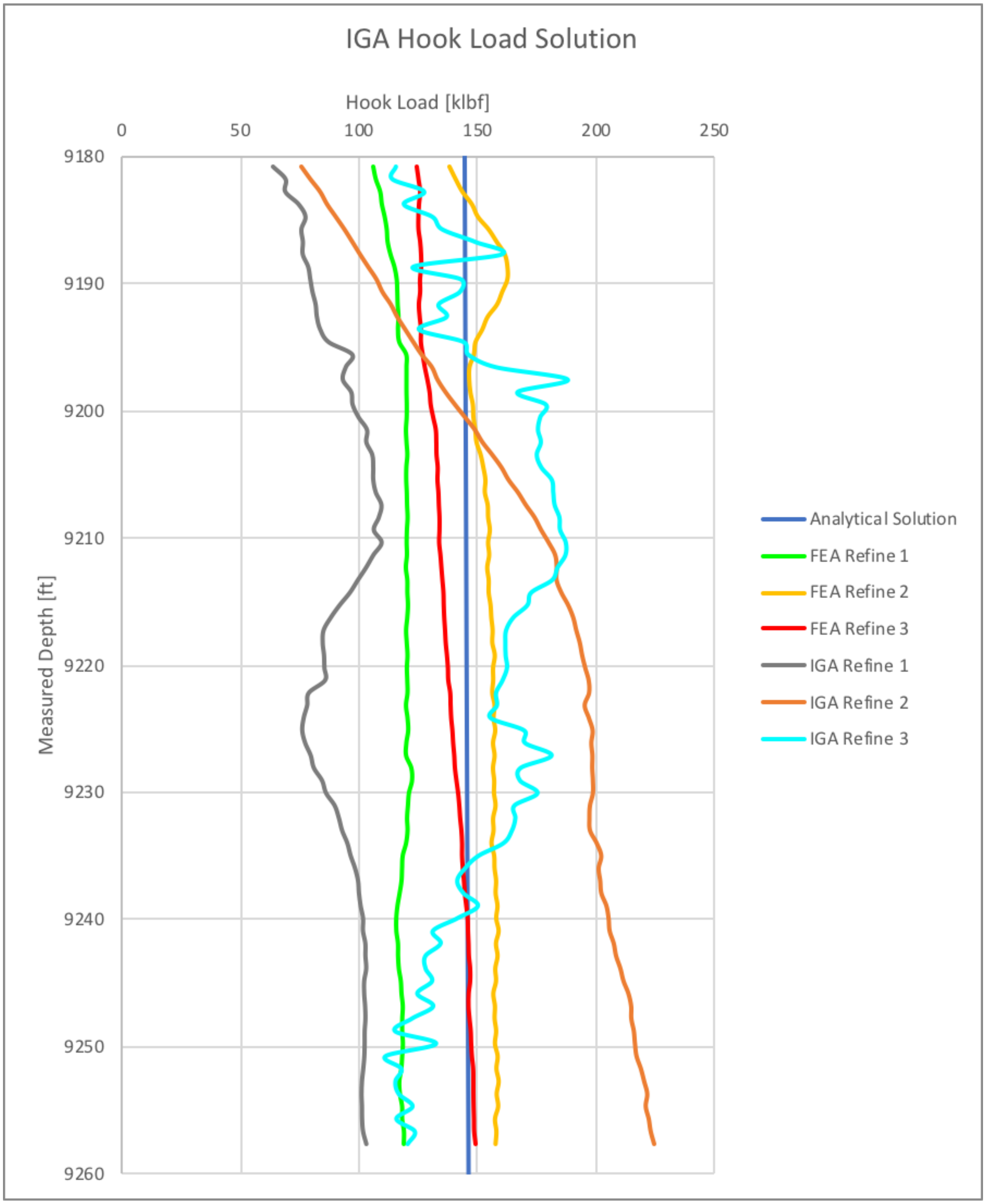
The red section is a finite element mesh of the wellbore, and the blue section is a finite element mesh of the drillstring. For each trial of this study, this LSPrePost visual is generally the same, the only difference being in the type of element formulation applied (FEA or IGA).



```
>
quat -0.829486 0.348535 -0.241259 0.363690;
zin  0.412533 0.087671 0.568538 0.884932
```

Draw a box to zoom in (Ctrl+Z)

B. Sample Resultant Force vs. MD Plot



C. Example LS-DYNA Keyword File

```
$# LS-DYNA Keyword file created by LS-PrePost(R) V4.5.5 - 14Nov2017
$# Created on Apr-12-2018 (19:13:51)
*KEYWORD
*TITLE
$#                                                                         title
LS-DYNA keyword deck by LS-PrePost
*CONTROL_IMPLICIT_AUTO
$#   iauto    iteopt    itewin     dtmin     dtmax     dtexp     kfail    kcycle
         0        11         5       0.0       0.0       0.0         0         0
*CONTROL_IMPLICIT_GENERAL
$#  imflag       dt0    imform      nsbs       igs     cnstn      form    zero_v
         0       0.0         2         1         2         0         0         0
*CONTROL_IMPLICIT_SOLUTION
$#  nsolvr    ilimit    maxref     dctol     ectol     rctol     lstol    abstol
        12        11        15     0.001      0.011.00000E10       0.91.0000E-10
$#   dnorm    diverg     istif   nlprint    nlnorm   d3itctl     cpchk
         2         1         1         0         2         0         0
$#  arcctl    arcdir    arclen    arcmth    arcdmp    arcpsi    arcalf    arctim
         0         0       0.0         1         2         0         0         0
$#   lsmtd     lsdir      irad      srad      awgt      sred
         4         2       0.0       0.0       0.0       0.0
*CONTROL_TERMINATION
$#  endtim    endcyc     dtmin    endeng    endmas     nosol
     600.0         0       0.0       0.01.000000E8         0
*DATABASE_GLSTAT
$#      dt    binary      lcur     ioopt
       1.0         1         0         1
*DATABASE_NCFORC
$#      dt    binary      lcur     ioopt
       1.0         1         0         1
*DATABASE_NODFOR
$#      dt    binary      lcur     ioopt
       1.0         1         0         1
*DATABASE_NODOUT
$#      dt    binary      lcur     ioopt   option1   option2
       1.0         1         0         1       0.0         0
*DATABASE_RCFORC
$#      dt    binary      lcur     ioopt
       1.0         1         0         1
*DATABASE_SLEOUT
$#      dt    binary      lcur     ioopt
       1.0         1         0         1
*DATABASE_BINARY_D3PLOT
$#      dt      lcdt      beam     npltc    psetid
       1.0         0         0         0         0
$#   ioopt
         0
*DATABASE_BINARY_INTFOR
$#      dt      lcdt      beam     npltc    psetid
       1.0         0         0         0         0
$#   ioopt
         0
*DATABASE_NODAL_FORCE_GROUP
$#    nsid       cid
         2         0
*DATABASE_HISTORY_NODE_SET
$#     id1       id2       id3       id4       id5       id6       id7       id8
         1         2         0         0         0         0         0         0
*BOUNDARY_PRESCRIBED_MOTION_SET_BOX
$#  typeid       dof       vad      lcid        sf       vid     death     birth
         2         3         0         1       1.0         01.00000E28       0.0
$#   boxid   toffset    lcbchk
         0         0                   0
*BOUNDARY_SPC_SET
```

```
$#    nsid       cid      dofx      dofy      dofz     dofrx     dofry     dofrz
         1         0         1         1         1         1         1         1
*SET_NODE_LIST_TITLE
Outer
$#     sid       da1       da2       da3       da4    solver
         1       0.0       0.0       0.0       0.0MECH
$#    nid1      nid2      nid3      nid4      nid5      nid6      nid7      nid8

*LOAD_BODY_PARTS
$#    psid
         1
*LOAD_BODY_Z
$#    lcid        sf    lciddr        xc        yc        zc       cid
         4       9.8         0       0.0       0.0       0.0         1
*CONTACT_AUTOMATIC_NODES_TO_SURFACE
$#     cid                                                                 title
$#    ssid      msid     sstyp     mstyp    sboxid    mboxid       spr       mpr
         2         1         3         3         0         0         1         1
$#      fs        fd        dc        vc       vdc    penchk        bt        dt
       0.3       0.3       0.0       0.0       0.0         0       0.01.00000E20
$#     sfs       sfm       sst       mst      sfst      sfmt       fsf       vsf
       1.0       1.0       0.0       0.0       1.0       1.0       1.0       1.0
*CONTACT_AUTOMATIC_ONE_WAY_SURFACE_TO_SURFACE
$#     cid                                                                 title
$#    ssid      msid     sstyp     mstyp    sboxid    mboxid       spr       mpr
         2         1         3         3         0         0         1         0
$#      fs        fd        dc        vc       vdc    penchk        bt        dt
       0.3       0.3       0.0       0.0       0.0         0       0.01.00000E20
$#     sfs       sfm       sst       mst      sfst      sfmt       fsf       vsf
       1.0       1.0       0.0       0.0       1.0       1.0       1.0       1.0
*PART
$#                                                                         title
Nurbs1
$#     pid     secid       mid     eosid      hgid      grav    adpopt      tmid
         1         1         1         0         0         0         0         0
*SECTION_SHELL
$#   secid    elform      shrf       nip     propt   qr/irid     icomp     setyp
         1       201       1.0         2       1.0         0         0         1
$#      t1        t2        t3        t4      nloc     marea      idof    edgset
      0.02      0.02      0.02      0.02       0.0       0.0       0.0         0
*MAT_ELASTIC
$#     mid        ro         e        pr        da        db  not used
         1    7850.01.000000E9      0.33       0.0       0.0         0
*PART
$#                                                                         title
Nurbs2
$#     pid     secid       mid     eosid      hgid      grav    adpopt      tmid
         2         2         1         0         0         0         0         0
*SECTION_SHELL
$#   secid    elform      shrf       nip     propt   qr/irid     icomp     setyp
         2       201       1.0         2       1.0         0         0         1
$#      t1        t2        t3        t4      nloc     marea      idof    edgset
      0.02      0.02      0.02      0.02       0.0       0.0       0.0         0
*DEFINE_BOX
$#   boxid       xmn       xmx       ymn       ymx       zmn       zmx
         1   35.3321   37.2519    17.374   19.9175   2540.39   2694.94
*DEFINE_CURVE_TITLE
Velocity
$#    lcid      sidr       sfa       sfo      offa      offo    dattyp     lcint
         1         0       1.0       1.0       0.0       0.0         0         0
$#                a1                  o1
                 0.0                 0.0
                 5.0                 0.3
             1000000                 0.3
*DEFINE_CURVE_TITLE
Gravity SF
$#    lcid      sidr       sfa       sfo      offa      offo    dattyp     lcint
         4         0       1.0       1.0       0.0       0.0         0         0
```

```
$#                a1                  o1
                 0.0                 1.0
              1000.0                 1.0
*SET_NODE_LIST_TITLE
Inner
$#     sid       da1       da2       da3       da4    solver
         2       0.0       0.0       0.0       0.0MECH
$#    nid1      nid2      nid3      nid4      nid5      nid6      nid7      nid8
     43706     43707     43708     43709     43710     43711     43712     43713
$#      w1        w2        w3        w4        w5        w6        w7        w8
       1.0       0.8       0.7       0.7       0.8       1.0       0.8       0.7
            ....
            1.0       0.8       0.7       0.7       0.8       1.0       0.8       0.7
*NODE
$#   nid               x               y               z      tc      rc
       1        83.45972       -120.3813        2957.559       0       0
            ....
   21832        35.26573        19.46165        2566.716       0       0
*END
```

References

Aadnoy, B.S., Fazaelizadeh, M., Kaarstad, E., and Mirhaj, S.A. (2010). New Aspects of Torque-and-Drag Modeling in Extended-Reach Wells. (SPE 135719). *SPE Annual Technical Conference and Exhibition.*

Aadnoy, B.S., Kaarstad, E., and Mirhaj, S.A. (2016). Torque and Drag Modeling; Soft-string versus Stiff-string Models. (SPE/IADC-178197-MS). *SPE/IADC Middle East Drilling Technology Conference and Exhibition.*

Baumgartner, T., van Oort, E. (2014). Pure and Coupled Drill String Vibration Pattern Recognition in High Frequency Downhole Data. (SPE 170955). *SPE Annual Technical Conference and Exhibition held in Amsterdam, The Netherlands.*

Baumgartner, T., van Oort, E. (2015). Maximizing Drilling Sensor Value through Optimized Frequency Selection and Data Processing. (SPE-174986-MS). *SPE Annual Technical Conference and Exhibition held in Houston, Texas.*

DeAngelo, John (2017). Torque and Drag Analycial Soft-String Matlab Model. The University of Texas at Austin, United States.

DrillScan (2013). Torque & Drag Buckling, Soft-string vs. Stiff-string Models. *DrillScan, France.*

Fazaelizadeh, M. (2013). Real Time Torque and Drag Analysis during Directional Drilling. Department of Chemical and Petroleum Engineering. Calgary, Alberta.

Hughes, T.J.R., Cottrell, J.A., Bazilevs, Y. (2009). Isogeometric Analysis: Toward Integration of CAD and FEA. John Wiley & Sons, Ltd., United Kingdom.

Hughes, T.J.R., Cottrell, J.A., Bazilevs, Y. (2004). Isogeometric analysis: CAD, finite elements, NURBS, exact geometry and mesh refinement. Elsevier B.V., United States.

Johancsik, C.A., Friesen, D.B., and Dawson, R. (1983). Torque and Drag in Directional Wells – Prediction and Measurement. (SPE 11380). *IADC/SPE Drilling Conference held in New Orleans.*

Tveitdal, T. (2011). Torque and Drag Analyses of North Sea Wells Using New 3D Model. University of Stavanger. Stavanger, Norway.

Owens, C.B. (2009). Implementation of B-Splines in a Conventional Finite Element Framework. Texas A&M University. Texas, United States.

Wu, A., Hareland, G., Fazaelizadeh, M. (2011). Torque & Drag Analysis Using Finite Element Method. *Modern Applied Science. Vol. 5, No. 6*. Canadian Center of Science and Education.

XTO Energy. (2018). Unconventional Resource Development, Technology and Process, Horizontal Drilling. Retrieved from http://www.xtoenergy.com/en-us/energy/unconventional-resource-development/technology-and-process/horizontal-drilling

University of Wisconsin, University of Engineering. (2006). Geometric Modeling for Engineering Applications Lecture Notes. Retrieved from http://homepages.cae.wisc.edu/~me232/