FACULTY OF SCIENCE AND TECHNOLOGY

MASTER THESIS

Study programme / specialisation: MsC in Petroleum Engineering/ Drilling and Well Engineering

The spring semester, 2022

Confidential

Author: Jairo Alexander Diaz Lopez

.....
(signature author)

Course coordinator: Hans Joakim Skadsem

Supervisor(s): Edvard Omdal

Thesis title: Wellbore stability analysis for an extended-reach well on Ekofisk

Credits (ECTS): 30

Keywords: Wellbore stability, extended-reach, Ekofisk, shear failure, tensile failure, plane of weakness, anisotropy, Monte Carlo simulations

Pages: 67

+ appendix: 55

Stavanger, 14/06/2022

Abstract

The drilling of extended-reach wells is an increasingly common practice to reduce costs in exploration and development of mature fields. Extended-reach wells are technically challenge, as high inclination trajectories may predispose to borehole instability.

This thesis studies an extended-reach well drilled in the Ekofisk field in the North Sea where borehole stability issues were observed and eventually resulted in the loss of the well. A wellbore stability assessment is performed with well-specific stress and formation strength data that explores the possible failures the well may have suffered from. Uncertainty is propagated both in rock strength and anisotropy to generate a mud window that acknowledges the variability present in geomechanical data using a Monte Carlo approach.

It is observed that three parameters from the plane of weakness present a high degree of uncertainty: cohesion, friction factor and orientation. The impact this parameters have in the failure along the planes of weakness pressure is studied. A safe mud window generated with a stochastic approach is presented. Acknowledging for uncertainty and the failure along the weakness planes in extended-reach wells to be drilled in Ekofisk may generate safer mud windows that in turn reduce the occurrence of wellbore instability in the field.

Acknowledgements

I would like to thank Professor Hans Joakim Skadsem for his support and contributions during this thesis. By being coauthors in our paper I learnt the importance of wellbore stability in the industry and I found a topic I would like to develop more.

Besides, I want to thank Edvard Omdal from ConocoPhillips. I will always be grateful for opening the doors of ConocoPhillips to me and having the role of a mentor, giving me support during the thesis and advice for both personal and working life. Also, thanks to James Rutherford, from ConocoPhillips, his advice and comments made me link better wellbore stability with drilling engineering, my true passion.

Special thanks to the friends I have made in Norway during this two years, they made the process easier and way more fun.

Finally, I want to say thanks to my family. Even though the distance is big, I have always felt their support and love.

Contents

List of Figures

List of Tables

List of Equations

List of Symbols and Abbreviations

Nomenclature

ERD	Extended-reach drilling
σ_v	Overburden stress
σ_H	Maximum horizontal stress
σ_h	Minimum horizontal stress
P_f, P_o	Pore pressure
v	Poisson ratio
T_0	Tensile rock strength
σ	Stress, stress tensor
F	Force
A	Area
σ_x	Normal stress in x direction
σ_y	Normal stress in y direction
σ_v	Normal stress in z direction
τ_{xy}	Shear stress in the plane xy
τ_{xz}	Shear stress in the plane xz
τ_{yx}	Shear stress in the plane yx
τ_{yz}	Shear stress in the plane yz
τ_{zx}	Shear stress in the plane zx
τ_{zy}	Shear stress in the plane zy
θ	Direction
τ	Shear stress
σ_1	Maximum principal stress
σ_2	Intermediate principal stress
$\sigma_3, S3$	Minimum principal stress
α	Biot coefficient
ρ	Density
g	Gravity constant

D, z Depth

p_{fn} Normal pore pressure

MPa Megapascal

LOT Leak-off test

XLOT Extended leak-off test

V_{cin} Volume change due to compression

V_{frac} Volume pumped into the fracture

p_{shut} Closing pressure

p_{fsip} Biot coefficient

V_{lost} Volume lost to the formation

V_{cout} Compresibility of the fluid in the borehole

V_{return} Volume return

ρ_w Fluid density

Pp^{hydro} Fluid density

DST Drillstem test tools

ROP Rate of penetration

Φ Weight on bit

Θ Weight on bit

WOB Weight on bit

S_0 Matrix cohesion

μ_0 Matrix friction factor

β Friction angle

I_1, I_2, I_3 Invariants 1, 2, 3

S_w Plane of weakness cohesion

μ_w Plane of weakness friction factor

QRA Quantitative risk assessment

MD Measured Depth

TVD Total Vertical Depth

POOH Pull out of the hole

RIH Run in hole

NCS	Norwegian Continental Shelf
RCA	Root-cause analysis
SD	Standard deviation
CDF	Cumulative density function
Q1	Quartile 1
Q2	Quartile 2
Q3	Quartile 3
P25	Percentile 25
P75	Percentile 75
P90	Percentile 90

1 INTRODUCTION

Humanity's energy use has undergone three main energy transitions: from firewood to coal, from coal to petroleum and from petroleum to renewable energy [1]. With the transition to renewable energies oil consumption will decrease remarkably, but still oil and gas are expected to be the main energy sources, as shown in Figure 1.

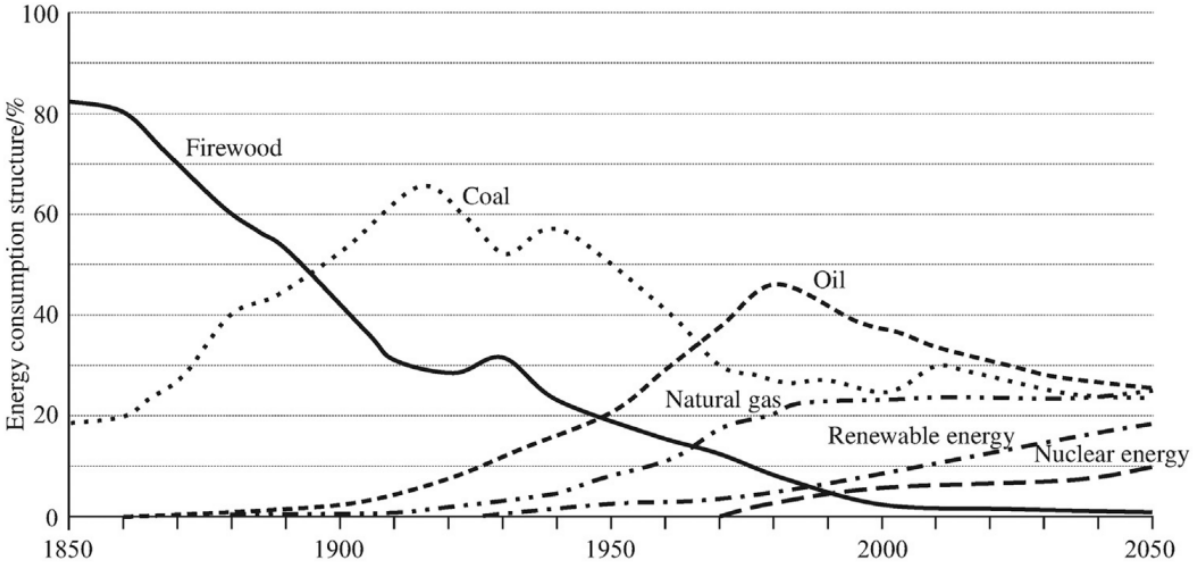


Figure 1: History and trends of global energy consumption
[1]

In order to remain competitive in the future, the oil industry has developed ways to produce unconventional oil and gas cheaper, like highly deviated wells, horizontal wells and extended-reach (ERD) wells. [2]. Deviated wells main advantage is covering larger draining areas of a reservoir, thus reducing the number of wells to be drilled and the overall cost of field development. Nevertheless, non-vertical wells require more planning, due to being less stable than vertical wells, as wellbore stability tends to decrease with the increase of inclination [3].

Wellbore stability is a problem that has been present during all the history of drilling. It is caused by the rock removal during drilling and the creation of a stress concentration in the borehole wall, which do not allow the hole to maintain its structural integrity. Wellbore instability often involves an increase in operational costs in terms of increase on tripping and reaming costs, reduced drilling performance, loss of equipment and, in the worst scenario, can suppose the lose of the well. New developments like deviated wells cause instability issues more difficult to handle, but in the same time more important to solve [4].

The present project has as main objective to find the cause of failure of the well COP-16, which is an extended-reach well which had to be abandoned due to a collapse even though the employed mud followed the plan. Minor wellbore stability issues were present while drilling the 13 1/2" section, but the liner stuck while running it in hole (RIH). After

this event, the well had to be sidetracked and abandoned. Mud windows including shear failure, tensile failure and failure along the bedding planes will be studied, with both a deterministic and a stochastic approach.

1.1 Wellbore Stability Assessment of an Anisotropic Shale Formation in the North Sea. SPE-208704-MS.

The paper "Wellbore Stability Assessment of an Anisotropic Shale Formation in the North Sea. SPE-208704-MS." was presented by Diaz and Skadsem the 8th of March 2022 at the IADC/SPE International Drilling Conference and Exhibition, Galveston, Texas, USA.

This paper is the base for the present project and the presented research will be continued and expanded in this thesis. It presents a wellbore stability analysis at a fixed depth with varying inclination while accounting for shear failure, tensile failure and plane of weakness failure.

The core employed in "Wellbore Stability Assessment of an Anisotropic Shale Formation in the North Sea" corresponds to an area near to the well COP-16. The core's datais presented in Table 1. As it can be observed, the friction factor in the plane of weakness is 60% of the matrix's friction factor, while the cohesion in the bedding plane and matrix are the same.

Table 1: Summary of the core's data for the mechanical stability analysis.

Variable	Value
True vertical depth	3020 m
Overburden stress, σ_v	60 MPa
Maximum horizontal stress, σ_H	55, 56 MPa
Minimum horizontal stress, σ_h	55 MPa
Pore pressure, p_f	49 MPa
Formation Poisson ratio, ν	0.25
Tensile rock strength, T_0	0 MPa
Shear strength of rock matrix	$\lvert\tau\rvert = (1.8 + 1.0\sigma'_n)$ MPa
Shear strength of bedding plane	$\lvert\tau\rvert = (1.8 + 0.4\sigma'_n)$ MPa
Orientation of bedding plane	Parallel to horizontal, $\Theta_w = 0$

With the information from Table 1, Figure 2 is generated. This mud window shows how the limiting mud weights and failure modes are affected by inclination, as it is considered only for TVD = 3020 m.

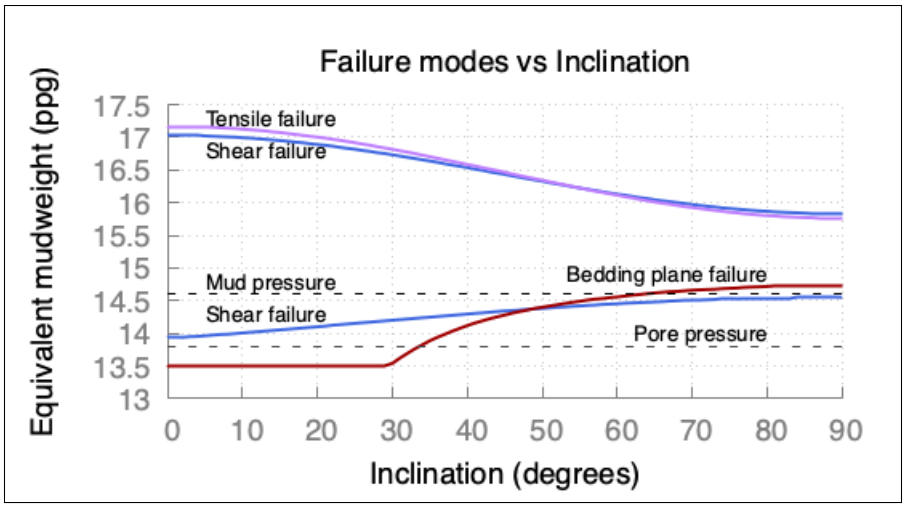


Figure 2: Failure modes as function of inclination [5].

As expected, the operational mud window narrows with the increase of inclination. For the lower bound, shear failure is the limiting failure mode until 48°, meaning that plane of weakness failure is not a concern, as shear failure would happen before. After 48°, the bedding plane failure predominates over shear failure. As it is observed, the failure pressure increases with the increase of inclination, as failure along the plane of weakness is highly dependant on the angle between the well and the bedding plane orientation. Higher pressures occur when the trajectory is more parallel to the orientation of the bedding plane, which is parallel to horizontal, as seen in Table 1. The constant bedding plane failure between 0° and 30° is caused by the considerations in the algorithm. The bedding plane failure at those angles is less than the minimum pressure at the code, therefore the code selects the minimum available pressures as the failure pressures. This is not a concern, as the mud weight will not be further reduced after observing a shear failure. For the upper bound, it is observed that there exists a shear matrix failure that predominates over the tensile failure up to 40°. After this inclination, they happen at almost the same pressures. This upper shear failure is not noticed in the field due to the high pressures in the wellbore, which do not allow the debris to fall to the bottom.

In the paper it is shown how high inclination wells, like extended-reached wells, are prone to suffer a shear failure along the bedding planes.

2 OBJECTIVES

2.1 General Objective

Define the cause of failure in well COP-16 of the Ekofisk Field.

2.2 Specific Objectives

- Describe the geology and generalities of the Ekofisk Field.
- Present a wellbore stability assessment.
- Analyze the impact of different failure mechanisms in well COP-16.
- Generate a mud window applying a stochastic analysis for well COP-16.

3 THE EKOFISK FIELD

The following chapter gives a brief description of the location, production and geology of the Ekofisk Field. At the end of the chapter, the whole geological sequence of the Field will be described.

3.1 The Ekofisk Field

Ekofisk is the first producing field in Norway and is operated by ConocoPhillips. It is located in the Great Ekofisk Area, in the southern part of the Norwegian North Sea, 300 kilometers south-west from Stavanger, close to the British and Danish sectors, as it can be observed in Figure 3. Test production started the 15th of June 1971 and ordinary production started in 1972 [6].

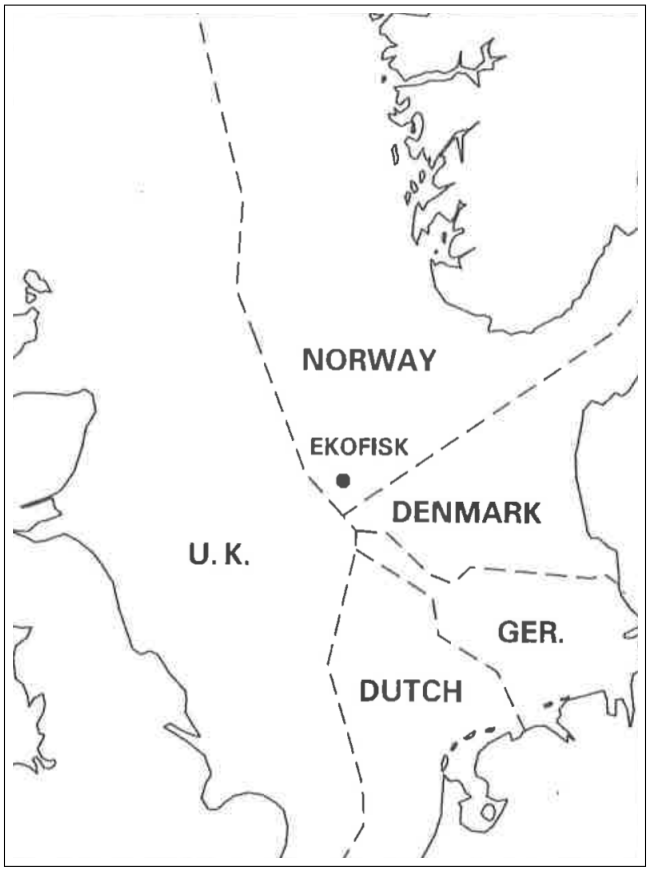


Figure 3: Ekofisk's location in the North Sea [7]

The Ekofisk reservoir is an elliptical anticline approximately 10.5 kilometers in length along N-S and 4.8 kilometers along E-W [8]. The top reservoir horizon is situated at about 3000 m depth at the crest of the structure, where the oil column is approximately 300 m thick [9]. Figure 4 shows the reservoir's structure.

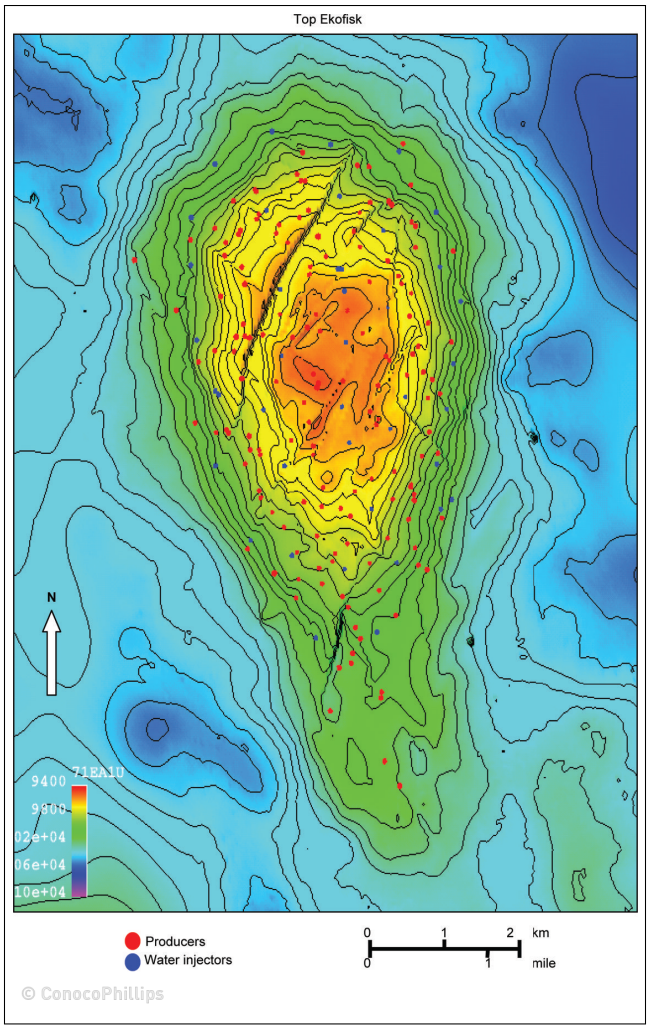


Figure 4: The Ekofisk field structure. [10]

The oil-bearing formations consists of fractured chalk with high porosity and low permeability. The original drive mechanisms were oil expansion, solution gas drive, reservoir compaction and limited gas injection [11]. Large-scale water injection started in 1987, which combined with the compaction of soft chalk has increased the recovery factor from 17% before injection to more than 50% after injection [6]. Figure 5 shows the impact water injection has had on the Field. Before water injection started production was steadily declining and the Field was expected to shut down in 1997, but water injection has allowed production for more than 50 years. By 2018, the total Field production was 4.2 billion equivalent oil barrels, with a daily production of 125000 bbls [12].

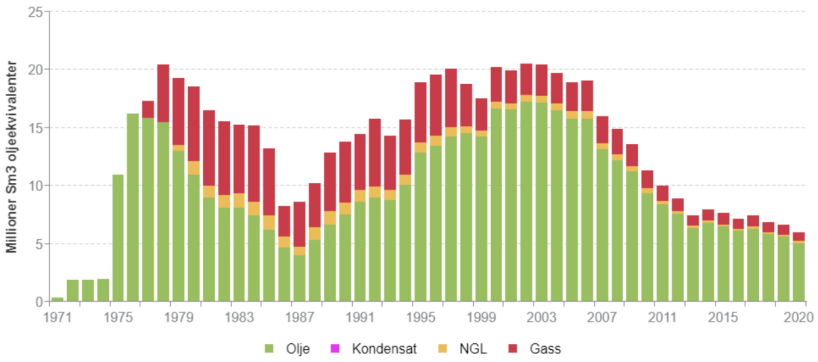


Figure 5: Ekofisk's production.

Source: https://www.npd.no/en/facts/news/general-news/2021/exceptional-ekofisk/

3.2 The Ekofisk Complex

The Ekofisk Complex is the field center and a hub for the production from Ekofisk and the other fields in the Greater Ekofisk Area, Elfisk and Embla. From the Ekofisk Field, the production from the Greater Ekofisk Area is sent to the receiving terminals, either Emden, Germany, for gas production, or to Teeside, UK, for oil production. The Ekofisk Complex is currently composed by 8 platforms and 3 seabed units [13]. Figure 6 displays an aerial picture of the Ekofisk Complex.

Figure 6: The Ekofisk Complex.

Source: ConocoPhillips.

3.3 Geology

The Ekofisk Field net area is approximately 49 km2. In well COP-16, the water depth is 260.3 ft (80 m).

3.3.1 Reservoir

The reservoir consists of two fine-graded limestone formations, the Ekofisk Formation (Danian Age), which can be divided into Upper and Lower Ekofisk, and the Tor Formation (Maastrichtian Age) [14]. Upper Ekofisk is neutral to preferentially oil-wet, Lower Ekofisk is neutral to low water wetness and Tor Formation is preferentially water-wet. The matrix permeability ranges between 0.1 and 10 mD, porosity is high, between 25% and 48%, the reservoir temperature is 130 ºC and the initial water saturation is between 15 and 20% [15]. Ekofisk and Tor are separated by a thin tight zone, as it can be seen in Figure 7.

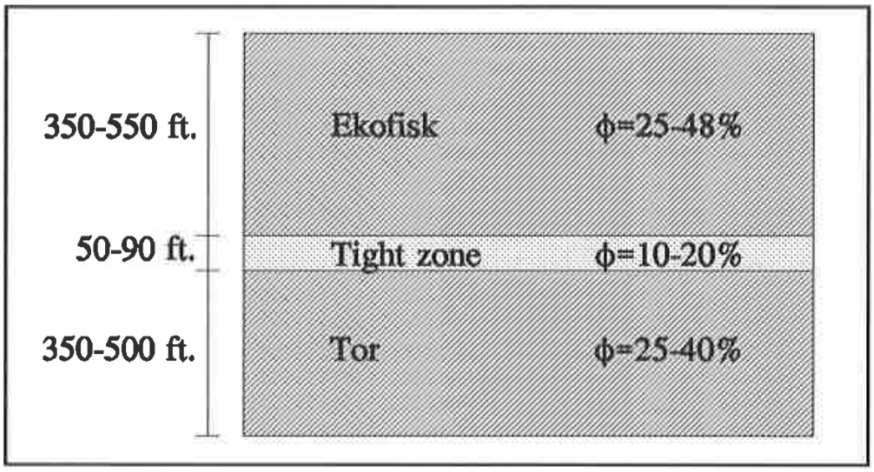


Figure 7: Ekofisk's Reservoir Formations

Source: Petersen, T.

3.3.2 Overburden

The overburden in Ekofisk is mostly composed by undercompacted shales and clays from Tertiary age. It is overpressured from 4000 to 5000 ft, but there is no indication of communication between the reservoir and the overburden [7]. An illustration of the stratigraphical sequence can be observed in Figure 8.

3.3.3 Geological Sequence

As it appears in Figure 8, ConocoPhillips differentiates between five groups in Ekofisk's overburden: Nordland, Hordaland, Rogaland, Chalk and Shetland.

- Nordland Group. The Nordland Group is the shallowest Group in Ekofisk. It is developed from the southern North Sea up to the western Barents Sea. Its thickness varies from more than 1700 m in the Central Through to less than 100 m in the Hammerfest Basin [16]. It is mainly composed by grey to brown-grey poorly bedded soft mudstones and siltstones [17]. In well COP-16, the 24", 17" and 13-5/8" casings were set in this Group.

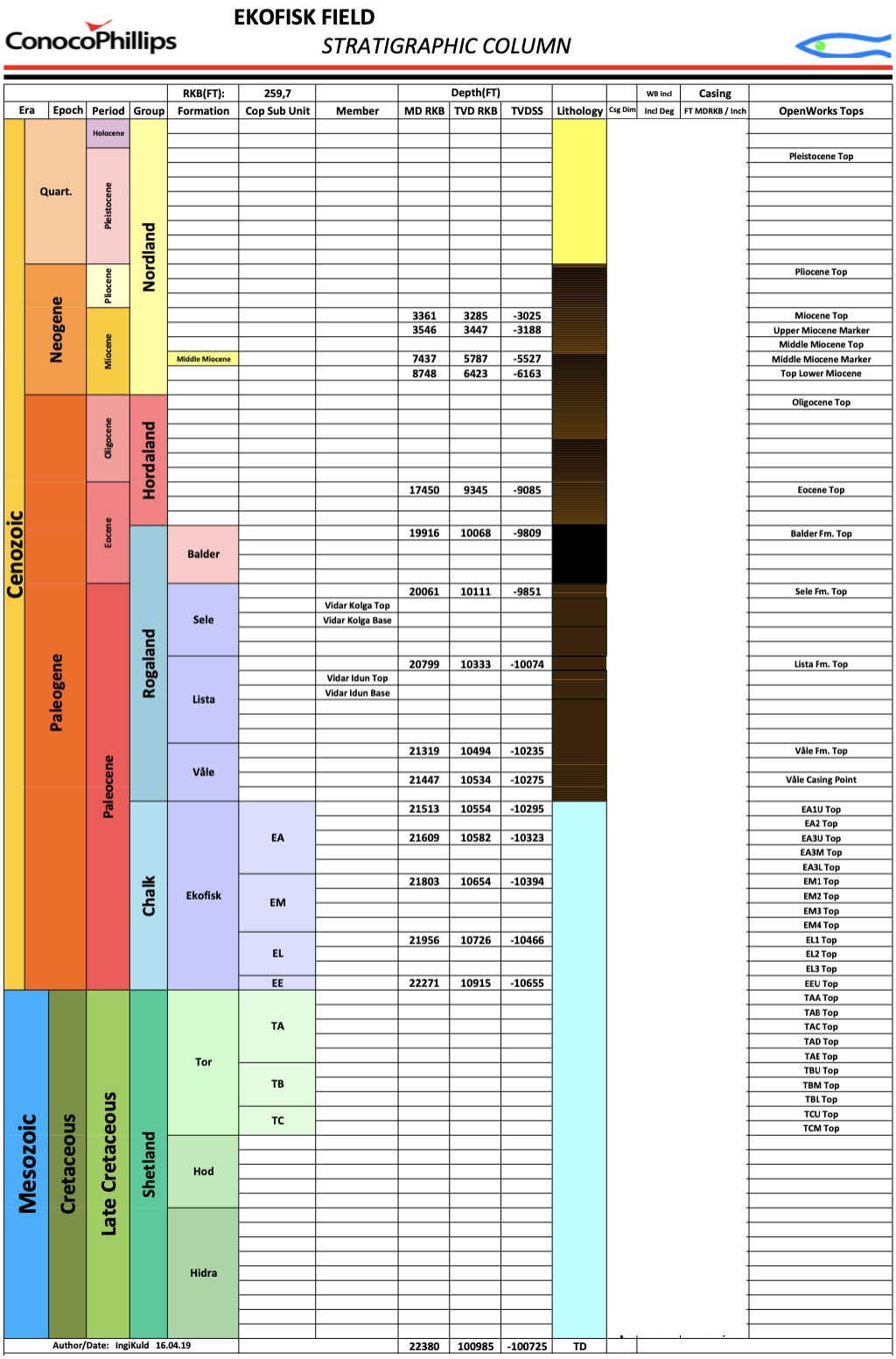


Figure 8: Stratigraphic sequence of Ekofisk Field. Obtained from ConocoPhillips.

- Hordaland Group. The Hordaland Group is present over most of the North Sea Basin. In consists of marine claystones with intersections of very fine to medium-graded sandstones. The Group's thickness decreases towards the basin margins, in general, it ranges between 1060 m to 1400 m [18]. In Ekofisk, it can be divided in Upper, Middle and Lower Hordaland [7].
- Rogaland Group. The Rogaland group is widely defined in the central and northern North Sea. It is thicker in the UK Sector, about 700 m thick, and thins eastwards and southwards, to about 100 m thick [19]. It is characterised by basin-wide mudstones and shales with intercalations of sandstones [20]. In Ekofisk, the thickness varies between 130 m to 500 m. It is on top of the reservoir, and consists of 4 formations: Balder, Sele, Lista and Våle [7]. The Lista Formation is the seal and the final overburden shoe is set in Våle.
- Chalk Group, composed by the Ekofisk Formation. The Ekofisk Formation extends along the Central Graben, in Ekofisk, where the most complete section is found, its thickness is around 200 m, while it is locally absent towards the east and northeast flanks of the Central Graben. It is composed mainly by reworked chalk, and the depositional environment is open marine with deposition of calcareous debris flows, turbidites and autochthonous periodites [21].
- Shetland Group, which most important formation is Tor Formation. The Tor Formation consists of pelagious and allochthonous chalks. The Ekofisk Formation and Tor Formation share the same sepositional environment: open marine with deposition of calcareous debris flows, turbidites and autochthonous periodites [22].

3.4 Subsidence in Ekofisk

Production of the high-porosity chalk reservoir causes subsidence in Ekofisk. Fluid-removal reduces the pore pressure, thus increasing the effective stress in the rock, which provokes compaction in the reservoir. The original reservoir pressure was approximately 500 bar, and was reduced to 250 by the mid-90's [8]. As described in Subsection 3.3.2, the overburden is predominantly composed by weak shales and mud rocks, which causes that the compaction in the reservoir is noticed in the sea-floor as subsidence [23] [24]. This problem was discovered in 1984, when the depth of the subsidence bowl was about 3 m with a subsidence rate between 25 cm/year and 40 cm/year. Furthermore, the subsidence created a "hinge" zone, in which the bending stresses were maximum [25]. Figure 9 shows the relationship between compaction and subsidence from 1986 to 1998. The "instantaneous" compaction to subsidence ratio (C/S) varies from 0.6 to 2.1, while the cumulative ratio trends towards 1.1 to 1.2 [8].

A water injection program was started in order to increase the recovery and decrease the subsidence rate. By August 2002, the total subsidence at the field's crest reached 8.26 meters [26]. The biggest impact of subsidence in the field is the effect on the in-situ stresses in the field, causing the need of updating the stress models every year. Furthermore, water injection has an impact on the stresses too. In 2001, a moderate

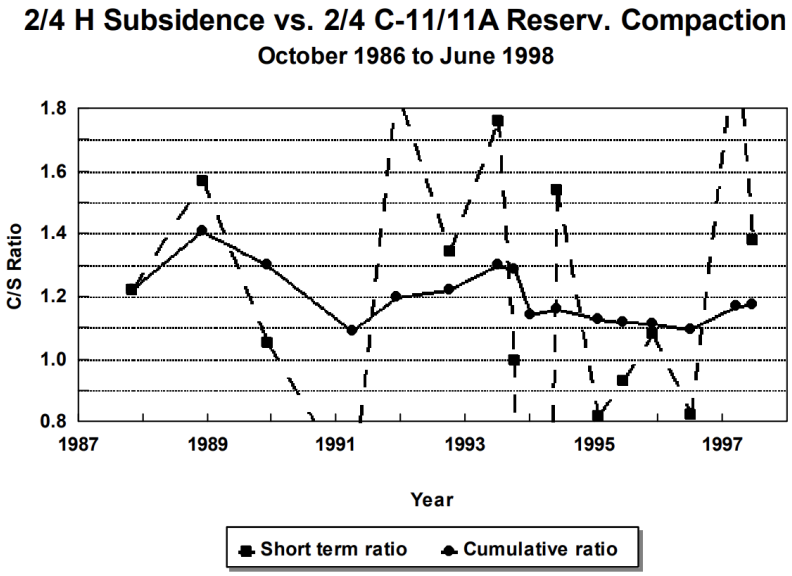


Figure 9: Instantaneous and cumulative compaction to subsidence ratios [8].

earthquake occurred in the southern North Sea with epicenter inside Ekofisk. The seismic event was induced by unintentional water injection in the overburden due to leakage of an injector well in the north flank of the field. This could be noticed due to uplifting in the north flank and overpressure in the area. Water injection increased the rock's pore pressure, which provoked fracture and changed the stress field. It seems that some strain in poorly consolidated and overpressured mud and shale rocks may be seismogenic [26].

4 WELLBORE STABILITY ASSESSMENT

In the following chapter the methodology to perform a wellbore stability assessment that will be followed in this project will be explained. A brief explanation of the necessary concepts needed to derive it will be given. At the end of the chapter, the importance of considering uncertainty in wellbore stability analysis will be stated.

4.1 Stresses and Calculation

4.1.1 Stress Definition and Principal Stresses

In general, the stress is the average force acting over an area, as it can be seen in Equation 1 [3].

$$\sigma = \frac{F}{A} \tag{1}$$

where σ is the stress (Pa or psi), F is the force (N or lbf) and A is the surface area (m^2 or in^2).

Stresses may result into two different stresses, σ, normal stresses, which act normal to the plane and τ, shear stresses, which act along the plane. Normal stresses can result in compaction or pore collapse, or tensile failure, while shear stresses may lead to shear failure.

Figure 10 shows a three dimensional stress distribution, it shows 9 different stresses: 3 normal stresses, σ_x, σ_y, σ_z and 6 shear stresses, τ_{xy}, τ_{xz}, , τ_{yx}, τ_{yz}, τ_{zx} and τ_{zy}.

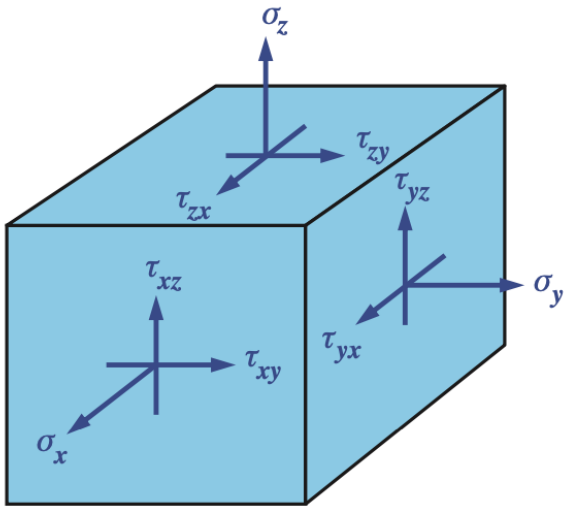


Figure 10: Three-dimensional state of a cube [3].

Assuming the body is at rest, meaning that all the forces on the body cancel, the stresses with the same subindexes in different order are equal, therefore the stress state can be reduced to 3 normal stresses and 3 shear stresses, which is shown in Equation 2.

$$\boldsymbol{\sigma} = \begin{bmatrix} \sigma_x & \tau_{xy} & \tau_{xz} \\ \tau_{xy} & \sigma_y & \tau_{yz} \\ \tau_{xz} & \tau_{yz} & \sigma_z \end{bmatrix} \tag{2}$$

In an arbitrary coordinate system, the full stress state within a three-dimensional object is determined by the 6 tensor components in Equation 2. It is, however, always possible to find a coordinate system in which the shear stresses vanish, assuming a body at rest where the forces are balanced. To illustrate this process, a 2D system will be assumed, where a triangle is at rest and no net forces act on it, as seen in Figure 11 [4]. With this assumptions, Equation 3 and Equation 4 are obtained.

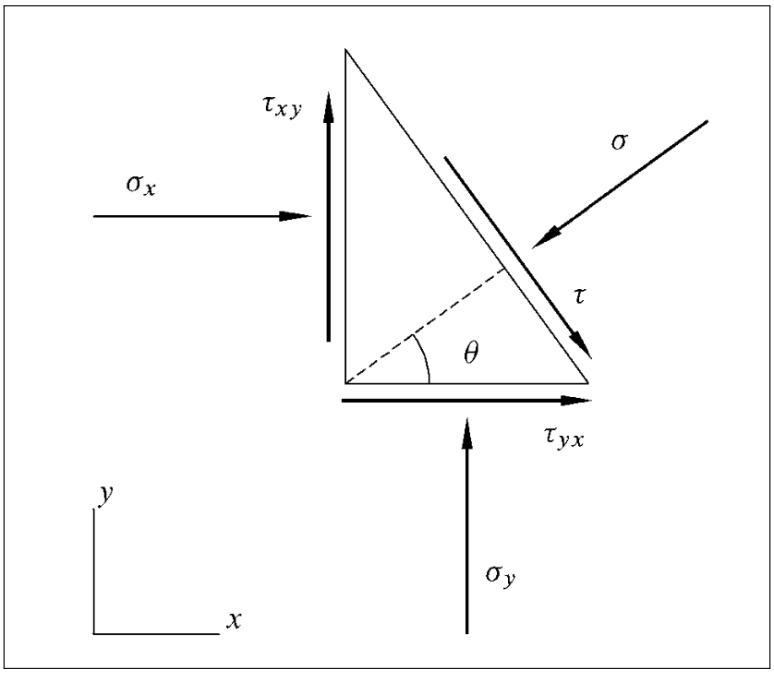


Figure 11: Force equilibrium on a triangle [4]

$$\begin{aligned} \sigma &= \sigma_x \cos^2\theta + \sigma_y \sin^2\theta + 2\tau_{xy}\sin\theta\cos\theta \\ &= \frac{1}{2}(\sigma_x + \sigma_y) + \frac{1}{2}(\sigma_x - \sigma_y)\cos 2\theta + \tau_{xy}\sin 2\theta \end{aligned} \tag{3}$$

$$\begin{aligned} \tau &= \sigma_y \sin\theta\cos\theta - \sigma_x\cos\theta\sin\theta + \tau_{xy}\cos\theta\cos\theta - \tau_{xy}\sin\theta\sin\theta \\ &= \frac{1}{2}(\sigma_y - \sigma_x)\sin 2\theta + \tau_{xy}\cos 2\theta \end{aligned} \tag{4}$$

From the previous equations, it can be found that there is a direction (θ) where the shear stress (τ) is equal to 0 , which is defined by Equation 5.

$$\tau = \frac{2\tau_{xy}}{\sigma_x - \sigma_y} \tag{5}$$

Equation 5 has two solutions, θ_1 and θ_2. These two directions are the principal axes of stress, for which the corresponding stresses are σ_1 and σ_2. These stresses are known as principal stresses, and they can be calculated by using the result from Equation 5 in Equation 3, as shown in Equation 6 and Equation 7.

$$\sigma_1 = \frac{1}{2}(\sigma_x + \sigma_y) + \sqrt{\tau_{xy}^2 + \frac{1}{4}(\sigma_x + \sigma_y)^2} \tag{6}$$

$$\sigma_2 = \frac{1}{2}(\sigma_x + \sigma_y) - \sqrt{\tau_{xy}^2 + \frac{1}{4}(\sigma_x + \sigma_y)^2} \tag{7}$$

This generates a new stress state conformed only by 2 principal stresses: σ_1 and σ_2, as seen in Equation 8.

$$\boldsymbol{\sigma} = \begin{bmatrix} \sigma_1 & 0 \\ 0 & \sigma_2 \end{bmatrix} \tag{8}$$

3D systems can also be reoriented to obtain 3 principal stresses: σ_1, σ_2 and σ_3, which can be observed in Equation 9. By definition, $\sigma_1 > \sigma_2 > \sigma_3$.

$$\boldsymbol{\sigma} = \begin{bmatrix} \sigma_1 & 0 & 0 \\ 0 & \sigma_2 & 0 \\ 0 & 0 & \sigma_3 \end{bmatrix} \tag{9}$$

In 3D systems, there are 3 possible geometric descriptions of the principal stresses according to the relation of the stresses between each other.

- **All principal stresses are equal** This case is known as hydrostatic state of stress and can be displayed as an sphere, as in Figure 12.

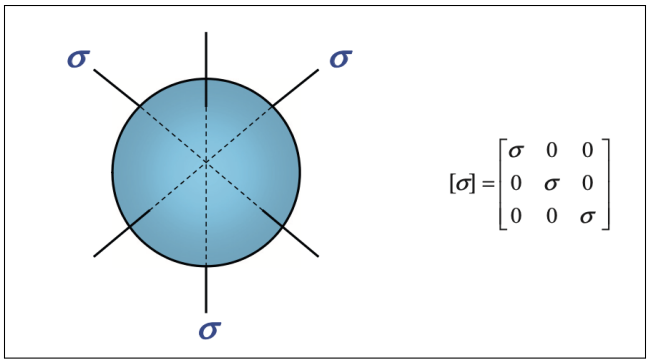


Figure 12: Hydrostatic stress loading representation [3].

- **Two principal stresses are equal.** When two principal stresses are equal there will by symmetry in the plane orthogonal to the unequal principal stress. It can be represented as a cylinder, like in Figure 13.

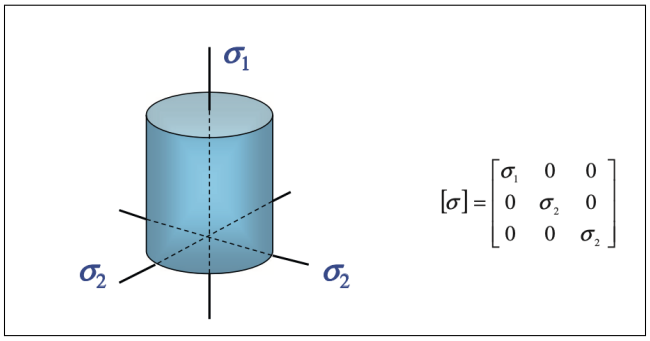


Figure 13: Load of two equal principal stresses representation [3].

- **All principal stresses are different.** All principal stresses have different magnitudes, it is also known as triaxial stress state, and the cubic stress geometric presentation is shown in Figure 14.

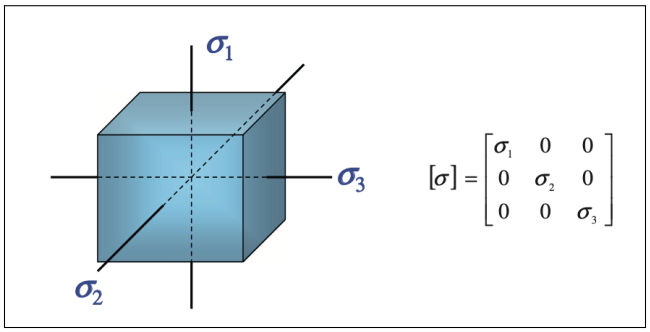


Figure 14: Triaxial stress loading representation [3].

4.1.2 Effective Stresses

Rocks are different from artificial materials, as they contain pores that may be saturated with fluids, therefore, the total stress applied to a rock is supported by the pore pressure and by the rock matrix. As shown in Equation 10, the total stress is the sum between the pore pressure(P_o) and the effective stress of the rock(σ'). α is the Biot coefficient, which refers to the fluid change induced by bulk volume changes in the drained conditions. The Biot coefficient varies between 0 and 1, but is conservatively set to 1 when studying failure, therefore, it will be omitted in the next equations.

$$\sigma = \sigma^{'} + \alpha Po \tag{10}$$

Rock mechanics studies the failure of the rock matrix, so when performing a wellbore stability analysis effective stresses must be employed.

$$\sigma^{'} = \sigma - Po \tag{11}$$

It is important to take into account that effective stresses are only applicable to normal stresses, as fluids at rest cannot transmit shear stresses [27].

4.2 In Situ Principal Stresses

In situ stresses affect all the operations that may be affected by rock failure, like drilling, completion, well service, production, and reinjection. Consequently, full knowledge of the in situ stresses is vital to perform stability analysis.

The stress state at any point of the rock can be expressed with three principal stresses: σ_v, σ_H and σ_h, which are respectively the vertical stress, the maximum horizontal stress and the minimum horizontal stress. The stress field determines the relationship between stresses. In normal stress regime, $\sigma_v = \sigma_1$, $\sigma_H = \sigma_2$ and $\sigma_h = \sigma_3$; in reverse stress regime, $\sigma_H = \sigma_1$, $\sigma_h = \sigma_2$ and $\sigma_v = \sigma_3$; and, in strike-slip stress regime, $\sigma_H = \sigma_1$, $\sigma_v = \sigma_2$ and $\sigma_h = \sigma_3$. Figure 15 shows the rock formation stresses for a normal stress fault system.

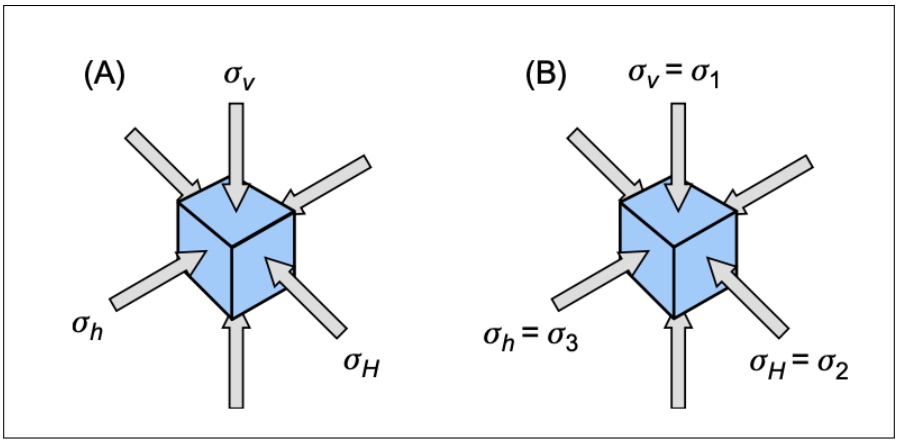


Figure 15: (A) Rock formation in situ stresses. (B) Rock formation in situ principal stresses for a drilled vertical well [3].

When considering the stress field, aside from the principal stresses, the pore pressure, Po, is also considered [28].

The vertical stress is assumed to be the overburden stress, and can be calculated with Equation 12. The overburden stress is directed downwards to the center of the Earth, increases with depth and depends on the superjacent rock's average density, depth and gravity.

$$\sigma_v = \int_0^D \rho(z) g dz, \tag{12}$$

Where D is depth, ρ the rock's density and g the gravitational constant.

In normal faulted stress regimes, $\sigma_v > \sigma_H > \sigma_h$, as seen in Figure 15(B). This relation between stresses changes with the increase of inclination. The change in stresses is reflected on the failure pressures not being constant with the increase of inclination, as it was seen in Figure 2.

In situ stresses are related to one another. As the overburden acts vertically it also has a horizontal impact, which affects the horizontal stresses, that may be constrained by adjacent rocks [3]. In situ stresses are orthogonal, and while the vertical stress is usually assumed to be a principal stress, it is not always the case. There are cases where the principal stresses do not follow the vertical-horizontal orientation, like in strongly sloped surfaces, near inclusions or faults, near underground openings like boreholes or near depleting reservoirs [4]. While vertical stress depends mainly in the depth and density of the superjacent rocks, it has been observed that horizontal stresses are sensitive to the rock's Poisson ratio, porosity and effective stresses.

Horizontal stresses are equal when they are only generated by the overburden, but the presence of active tectonic areas may generate additional stress components, which

cases σ_H and σ_h to be different. The horizontal stresses involve more uncertainty than the vertical stress, specially in new regions, where measurements are not available. Empirical equations have been developed to estimate σ_h.

Equation 13 shows an empirical equation presented by Avasthi et al., which relates the horizontal stress with the Biot coefficient, Poisson ratio (ν) overburden stress and the horizontal component of the pore pressure.

$$\sigma_h = \frac{\nu}{1-\nu}(\sigma_v - \alpha P_o) + \alpha P_o, \tag{13}$$

Breckels and van Eekelen developed empirical correlations for estimation of σ_h as a function of depth. The correlations for the US Gulf Coast are shown in Equation 14 and Equation 15.

$$\sigma_h = 0.0053D^{1.145} + 0.46(p_f - p_f n)(D < 3500m), \tag{14}$$

$$\sigma_h = 0.0264D + 0.46(p_f - p_f n)(D > 3500m), \tag{15}$$

Where D is the depth in meters, p_f the pore pressure in MPa and $p_f n$ the normal pore pressure.

Another way to estimate the minimum horizontal stress is the use of extended leak-off test (XLOT), which is a modification of the leak-off test (LOT). The main difference between the two tests is than in XLOT the pumping continues past the leak-off point and the breakdown pressure [4]. Several cycles are performed in order to obtain repeatable test results, as seen in Figure 16.

The bold line represents the pressure-volume relation while pump-in and the broken line, the shut-in and flowback phase. V_{cin} is the volume change due to compresion of the fluid, V_{frac} represents the volume pumped into the fracture. When pumping stops and the well is shut-in, the pressure decreases from p_{shut} to p_{fsip}. The broken line between p_{shut} to p_{fsip} represents the flowback phase, which can provide an estimate of σ_h. V_lost is the volume of fluid lost to the formation, V_{cout} reflects the compressibility of the fluid in the borehole and V_{return} is the actual volume returned from the fracture. By employing databases with regional data from XLOTs, models can be developed, which are more accurate than the ones presented in Equation 14 and Equation 15 [29].

The concentration of stresses in the wellbore wall is different than the in situ stresses due to the effect of inclination, azimuth and the removal of rocks during drilling. Failures, either shear failure or tensile failure, may appear. Shear failures are often referred to as breakouts, and tensile failures as drilling induced fractures. Drilling induced fractures are different from hydraulic fractures, as they are caused by high wellbore pressures and

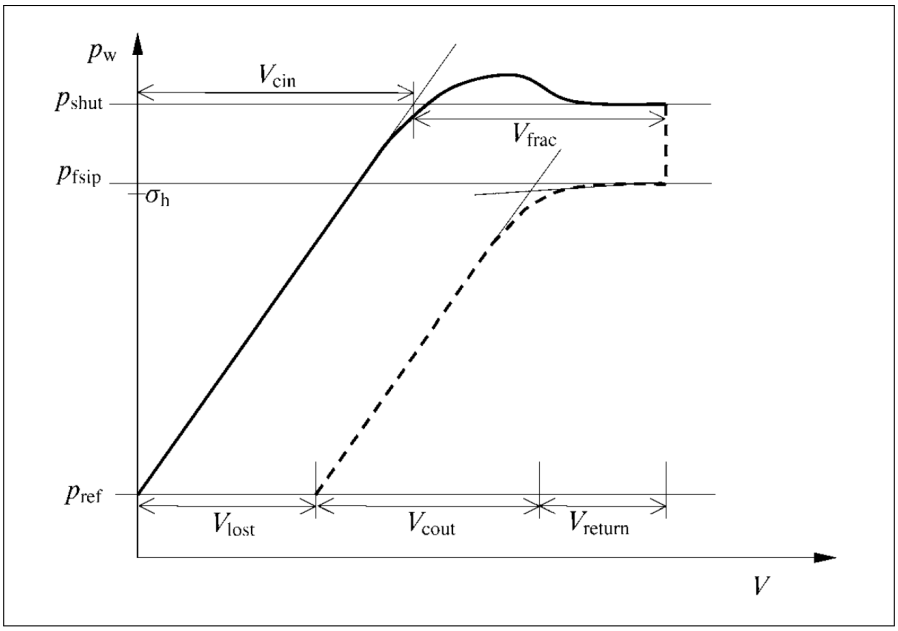


Figure 16: Schematic XLOT test [4].

propagate far from the stress concentration. Both breakout and hydraulic fractures have importance on determining the principal stresses' magnitude and orientation.

4.2.1 Pore pressure calculation

Pore pressure may be measured directly or estimated by employing logs or seismic data. As Figure 17 shows, pore pressure is defined as a scalar hydraulic potential acting within an interconnected pore space at depth (z), which is described in relation to hydrostatic pressure (ρ_w), as shown in Equation 16. A hydrostatic pore pressure implies the existence of a well interconnected and open pore and fracture network from surface to the depth of measurement. As rocks have a negligible small tensile strength, pore pressure will always be smaller that the least principal stress (σ_3) [28].

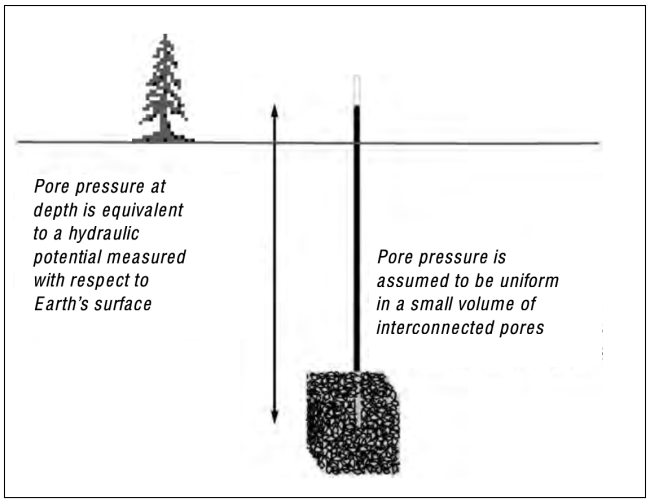


Figure 17: Schematic of pore pressure concept [28].

$$Pp^{hydro} = \int_0^z \rho_w(z) g dz, \tag{16}$$

Equation 16 accounts for hydrostatic pore pressures, but geological processes like rapid sedimentation, slides, erosion and fluid seeps may result in non-hydrostatic pore pressures, which are hazard areas due to strength reduction and decrease in stability [30]. Pore pressure may be affected by underpressure, which occurs rarely, or overpressure. Some mechanisms that lead to overpressure are: undercompaction (rapid sedimentation), tectonic compression, hydrocarbon column effects, aquathermal pressurization, dehydration reactions and hydrocarbon generation [4].

There are commercial tools available that allow measuring the pore pressure directly, like drillstem test tools (DST). This type of tools include a surface-actuated packer that isolates the formation from the annulus, thus forcing the produced fluids to enter the drillstring [31].

Pore pressure in shales is challenging to measure, as their low permeability increases the difficulty of direct measuring. Traditionally, geophysical logging data is employed to estimate it by relating the measured velocity of a formation with its pore pressure, but there are new solutions which allow more accurate measures, like prediction models employing machine learning and direct measuring by using new tools:

- Machine learning allows to solve complex problems and deal with the big data, and is widely employed in Petroleum Engineering, mainly in ROP prediction and optimization and estimation of the recovery factor. Machine learning based models require a filtering and cleaning stage for the selected data, but in return offer accurate predictions. There are different machine learning models for pore pressure estimation, like the one presented by Ahmed Abdelaal et al., which includes as parameters hydraulic data (pump rate and standpipe pressure) and mechanical measurements (rotary speed, ROP, torque and WOB) [32]. Other example is the model presented

by Booncharoen et al., employs as parameters pay thickness, porosity, water saturation, original pressure and total gas show [33].

- Some examples of new technologies developed to allow in-situ measuring of pore pressure are piezometers and wireless downhole sensors. Piezometers are designed for autonomous subsea deployment and also allow long-term monitoring campains [30]. MESHPOSH sensor is a system utilized for first time in the Grane Field in a water injection well , and it is based in wireless communication between sensor located inside and outside the casing, the outside sensor is cemented, and makes permanent monitoring feasible. As the system is wireless, the data and power transmission occurs through the casing.

4.2.2 Overburden calculation

The overburden can be calculated from the integration of the density of the suprajacent rocks, as seen in Equation 12. Overburden is the principal vertical stress in vertical wells, this can be tested with drilling induced tensile fractures.

4.2.3 Minimum horizontal stress calculation

The minimum horizontal stress magnitude and orientation can be obtained from information retrieved from hydraulic fractures, for example, by performing mini-fracks and leak-off-tests, as hydraulic fractures will always propagate perpendicular to the minimum horizontal stress [34]. In vertical wells, breakouts always form in the azimuth of σ_h, as long as the principal stresses are vertical and horizontal [28].

4.2.4 Maximum horizontal stress calculation

The maximum horizontal stress is the only in situ stress that cannot be measured directly. It requires to be estimated through constrains, the frictional strength of the crust provides general bounds and observations of wellbore failures, like breakouts and drilling-induced tensile fractures increase the accuracy on the estimations.

4.2.5 Stress direction measurement

The stresses direction is measured through the observation of faults and fractures by employing wellbore imaging tools, as shown in Figure 18.

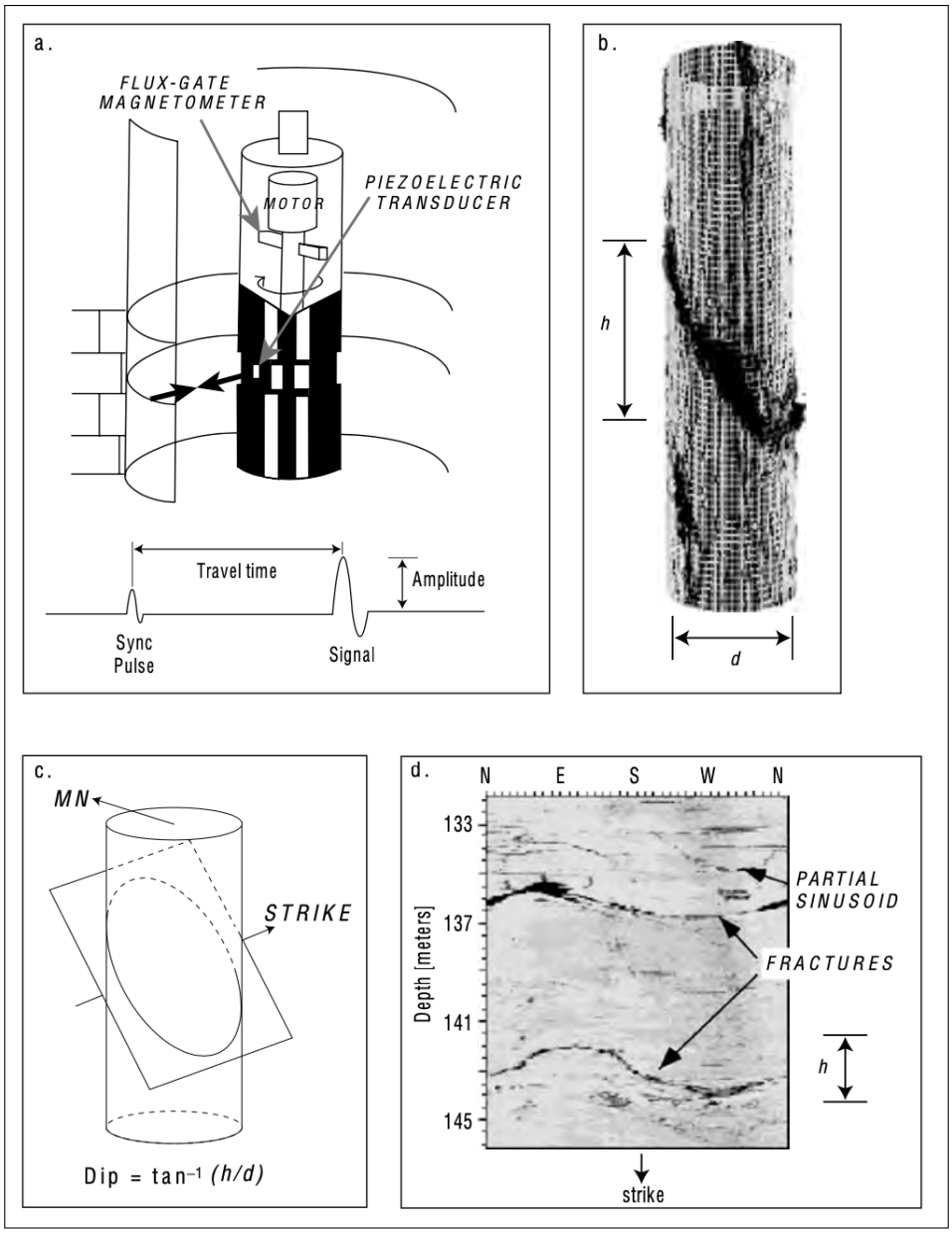


Figure 18: a. Ultrasonic transducer principle of operation. b. 3D amplitude data display. c. schematic view of a plane cutting through a wellbore. d. Unwrapped view of a wellbore view with depth on the ordinate and azimuth on the coordinate [28].

4.3 Types of Rock Failure

Rocks may fail in two different ways, either a tensile failure or a shear failure. In boreholes, tensile failure is associated with fracture, which is represented in the mud window by the fracture pressure, the upper pressure limit of the mud operational window. Shear failure in boreholes is traditionally assumed to be a collapse of the rock matrix and to be the lower pressure limit of the operational mud window.

4.3.1 Tensile Failure

Tensile failure occurs when the effective tensile stress exceeds the tensile strength of the rock, T_0 [4]. The tensile strength value is low, just a few MPa, and is conservatively assumed as 0, as most rocks have pre-existing flaws. It is usually assumed that this type of failure occurs at high pressures, but it may happen when drilling underbalanced, where the wellbore pressure is less than the pore pressure. Tensile failures generate big amounts of cavings.

Figure 19 shows how samples split along one fracture plane. This fracture plane originates from preexisting cracks and is oriented perpendicular to the tensile stress.

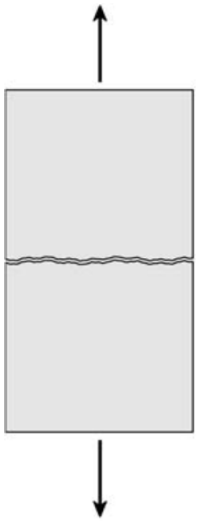


Figure 19: Tensile failure in a rock sample [4] .

4.3.2 Shear Failure

Shear failure is caused when the shear stress exceeds the rock's shear strength. When the fault zone is developed along the failure plane the two sides of the plane will move against each other in a frictional process [4], as it can be observed in Figure 20.

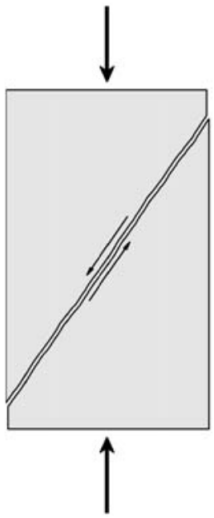


Figure 20: Tensile failure in a rock sample [4].

Researchers like Ding et al. [35] have found that shear failures not only occur at low pressures, but also happen at high well pressures. The reason why shear failures at high pressures are not observed at field is because the high mud pressures do not allow the generated debris into the borehole.

Shear failure may also occur along weak bedding planes in laminated rocks, like shales or finely laminated sandstones. The influence of weak bedding planes on rock strength is called strength anisotropy [28]. This type of shear failure depends heavily on the orientation of the well respect to the orientation of the bedding planes of the rock. Even though failure along the planes of weakness is not included in mud windows, the industry gives a recommendation to drill shales perpendicular to the orientation of the shales' planes.

4.4 Cavings

Cavings are pieces of rock which were not removed directly by the drill bit [36]. Cavings allow rapid interpretation of downhole conditions, including the nature of instability and locations [37]. Table 2 summarizes the main caving types, their shape and how they are produced.

Table 2: Types of cavings, morphology and causes. Adapted from Skea et al [37].

Shape	Description	Cause
Angular	Triangular/arrowhead shape	Low mud weight pressures are not able to support the wellbore wall
Tabular	Flat parallel faces	Caused by the invasion of drilling fluids in the planes of weakness. They are difficult to circulate.
Splintery	Flat, thin and planar structures	Underbalanced drilling in hard rock or high tectonic areas
Blocky	Cubic	The invasion of drilling fluid into pre-existing fractures destabilizes the formation

4.5 Wellbore Stability Assessment

To perform any wellbore stability assessment the following steps must be followed: first, the in-situ stresses have to be transformed into the direction of the wellbore. Second, the Kirsch Equations are employed to calculate the principal stresses in the borehole wall. The last step is to apply the chosen criteria for the different types of failure. Each step will be explained in the following subsections.

4.5.1 Coordinate System Transformations

The in-situ stresses are assumed to be oriented orthogonally, where the overburden σ_v is parallel to the vertical direction, z'. The maximum and minimum horizontal stresses, σ_H and σ_h, are parallel to x' and y', respectively.

The borehole system follows a different coordinate system, (x, y, z) where z is parallel to the borehole axis and x follows the low side of the borehole's direction, as shown in Fig. 21. The transformation from (x', y', z') is achieved by a counter-clockwise rotation through the azimuth angle, Φ, about z', followed by a counter-clockwise rotation through the inclination angle Θ about the new y axis [4].

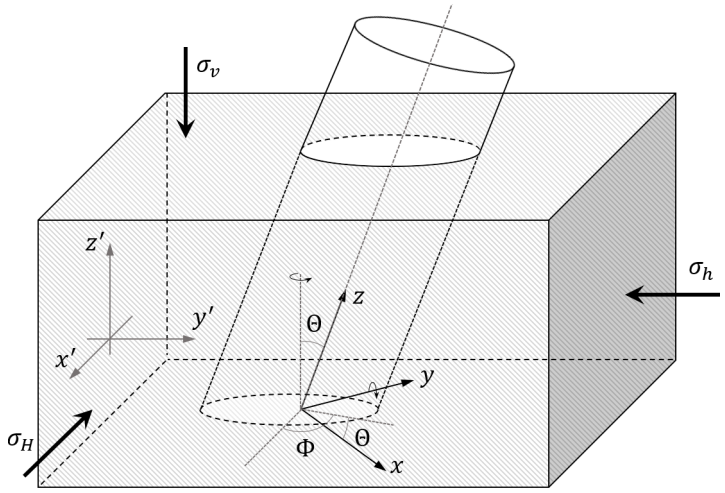


Figure 21: Coordinate systems aligned with the in-situ stresses and with the borehole [5].

To address failure along the planes of weakness another coordinate system transformation has to be performed. The stresses must be transformed into the orientation of the planes of the rock, as shown in Fig 22.

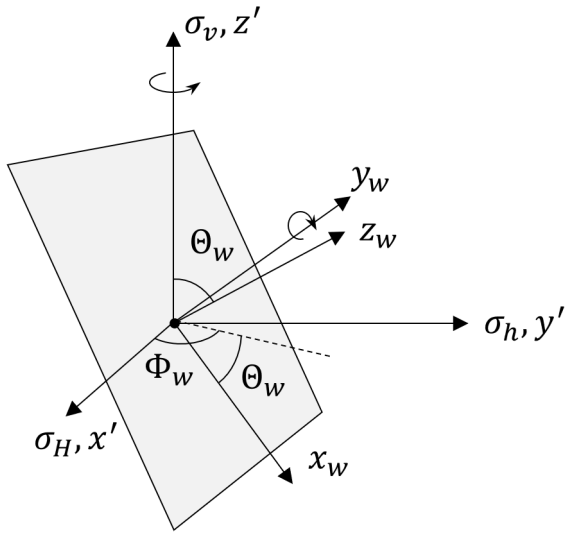


Figure 22: Orientation of the planes of weakness relative to the in-situ stresses [5].

Transformations between these coordinate systems are achieved by combining rotation matrices about the y and z axes.The individual rotation matrices are [4]:

$$\boldsymbol{R}_y(\theta) = \begin{pmatrix} \cos\theta & 0 & -\sin\theta \\ 0 & 1 & 0 \\ \sin\theta & 0 & \cos\theta \end{pmatrix}, \quad \text{and} \quad \boldsymbol{R}_z(\phi) = \begin{pmatrix} \cos\phi & \sin\phi & 0 \\ -\sin\phi & \cos\phi & 0 \\ 0 & 0 & 1 \end{pmatrix}. \tag{17}$$

The transformations described above, are defined by

$$\boldsymbol{R}_1 = \boldsymbol{R}_y(\Theta)\boldsymbol{R}_z(\phi), \quad \text{and} \quad \boldsymbol{R}_2 = \boldsymbol{R}_y(\Theta_w)\boldsymbol{R}_z(\phi_w). \tag{18}$$

With these definitions, a stress tensor $\boldsymbol{\sigma}_i$ in the (x', y', z') coordinate system is transformed to the borehole coordinate system (x, y, z) by $\boldsymbol{\sigma} = \boldsymbol{R}_1\boldsymbol{\sigma}_i\boldsymbol{R}_1^T$. The transformation to the plane of weakness is achieved by $\boldsymbol{\sigma}_w = \boldsymbol{R}_2\boldsymbol{\sigma}_i\boldsymbol{R}_2^T$.

4.5.2 The Kirsch Equations

In 1898, Kirsch published a paper about the stress distribution around a circular hole in one-dimensional system, which can be generalized to a vertical borehole with unequal far field stress [4]. All the obtained stresses are in the borehole wall. p_w is the wellbore pressure, σ_r the radial stress, σ_θ the hoop stress, σ_v the vertical stress and $\tau_{r\theta}$, $\tau_{\theta z}$ and τ_{rz} the shear stresses. The input stresses: σ_x, σ_y, σ_z, τ_{xy}, τ_{xz} and τ_{yz}, are oriented in the borehole coordinate system.

$$\sigma_r = p_w, \tag{19a}$$
$$\sigma_\theta = \sigma_x + \sigma_y - 2(\sigma_x - \sigma_y)\cos 2\theta - 4\tau_{xy}\sin 2\theta - p_w, \tag{19b}$$
$$\sigma_z = \sigma_z - \nu\left[2(\sigma_x - \sigma_y)\cos 2\theta + 4\tau_{xy}\sin 2\theta\right], \tag{19c}$$
$$\tau_{r\theta} = 0, \tag{19d}$$
$$\tau_{\theta z} = 2(-\tau_{xz}\sin\theta + \tau_{yz}\cos\theta), \tag{19e}$$
$$\tau_{rz} = 0, \tag{19f}$$

Where θ is a polar angle measured from x and ν is the Poisson ratio of the elastically isotropic formation. The stress components are illustrated in Fig.23.

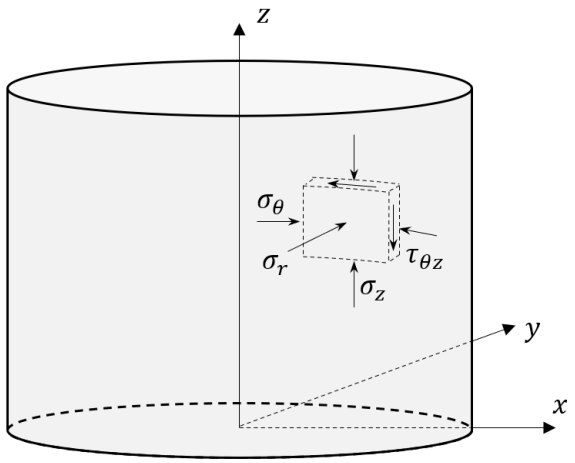


Figure 23: Principal stresses at the borehole wall [5].

The principal stresses in the borehole coordinate system are:

$$\sigma_i' = \sigma_r', \tag{20a}$$

$$\sigma_j' = \frac{\sigma_\theta' + \sigma_z'}{2} + \frac{1}{2}\sqrt{(\sigma_\theta' - \sigma_z')^2 + 4\tau_{\theta z}^2}, \tag{20b}$$

$$\sigma_k' = \frac{\sigma_\theta' + \sigma_z'}{2} - \frac{1}{2}\sqrt{(\sigma_\theta' - \sigma_z')^2 + 4\tau_{\theta z}^2}. \tag{20c}$$

It is important to note that all stresses are effective. As shown in Equation 11, effective stresses are obtained by subtracting the pore pressure from the total stress.

4.5.3 Failure Criteria

The last step when performing wellbore stability assessments is to apply the comparisons with the elected criteria. This thesis will employ four failure criteria: the common tensile failure criteria for tensile failure, Mohr-Coulomb criterion and Modified Lade for shear failure and Mohr-Coulomb for shear failure in the plane of weakness.

The tensile failure criterion employed in this project is:

$$\sigma_3{}' = -T_0 \tag{21}$$

where σ_3' denotes the minimum effective principal stress and T_0 is the tensile strength.

For shear failure there are many available criteria, which can be classified by linearity of the governing equation and consideration of the intermediate principal stress [38]. Two criteria will be considered: the Mohr-Coulomb criterion and the Modified Lade Criterion.

The Mohr-Coulomb criterion is the most common criterion in wellbore stability assessments. It is a linear criterion that neglects the intermediate stress:

$$|\tau| = S_0 + \mu_0 \sigma', \tag{22}$$

where S_0 is the rock cohesion and μ_0 is the coefficient of internal friction, or friction factor. The criterion can be expressed in terms of the effective maximum and minimum principal stresses at the borehole wall as follows [39]:

$$\sigma_1' = 2S_0 \tan\beta + \sigma_3' \tan^2\beta, \tag{23}$$

where $\beta = \pi/4 + \phi/2$ and $\tan\phi = \mu$, [39].

The Modified-Lade criterion is a non-linear criterion which acknowledges the impact of the intermediate principal stress [40]:

$$\frac{(I_1'')^3}{I_3''} = 27 + \eta, \tag{24}$$

Where I_1'' and I_3'' are the invariants 1 and 3, respectively and η is a material constant that can be derived from the friction angle ϕ .

$$(I_1'') = (\sigma_1' + S_L) + (\sigma_2' + S_L) + (\sigma_3' + S_L) \tag{25}$$

$$(I_3'') = (\sigma_1' + S_L)(\sigma_2' + S_L)(\sigma_3' + S_L) \tag{26}$$

$$\eta = 4tan^2\phi(9 - 7sin\phi)/(1 - sin\phi) \tag{27}$$

S_L is a material constant and can be derived directly from the Mohr-Coulomb cohesion, S_0, and friction angle ϕ .

$$SL = \frac{S_0}{tan(\phi)} \tag{28}$$

To assess plane of weakness failure, a Mohr-Coulomb criteria will be employed, [41]. The main difference between the assessment of matrix failure and failure along the bedding planes is the consideration of the parameters of the rock or the bedding plane. For the plane of weakness, this criterion is expressed as:

$$|\tau| = S_w + \mu_w \sigma', \tag{29}$$

where σ' denotes the effective normal stress on the plane of weakness, and $|\tau|$ the shear stress along the plane. S_w and μ_w correspond to the cohesion and the coefficient of internal friction for the plane of weakness.

As performed by Lee et al., [42], the effective normal stress that acts perpendicular to the plane of weakness is the z_w component of the stress tensor, $\sigma' = \sigma'_{z,w}$. The shear stress acting on the plane of weakness is expressed as:

$$|\tau| = \sqrt{\tau_{zx,w}^2 + \tau_{zy,w}^2}, \tag{30}$$

4.6 Uncertainty

The data employed in this thesis comes from core measurements and stress predictions. Cores are the best way to measure the rock's properties, but they are expensive to retrieve. Typically, only a few cores are taken for a field, and the data obtained from them is extrapolated for the whole field, which does not acknowledge the anisotropy and geological heterogeneity of rocks. Also, their properties may be affected during the retrieving process in an unknown way. Wellbore stability assessments assume the properties of the rockes and in situ conditions are precisely known, but the lack of data generates uncertainty [43].

One way to assess this uncertainty is performing a stochastic analysis, which would give ranges of values instead of a exact value. Ottessen et al. proposed the use of QRA (Quantitative Risk Assessment), which is used to account possible risk scenarios [44]. In this thesis, the approach presented by Diaz and Skadsem [5] will be applied, where Monte Carlo simulations for uncertainty propagation will be used.

First, each uncertain input parameter is associated with a probability distribution, either a normal or triangular distribution. Most events in nature follow a normal distribution, but triangular distributions have the advantage to set the limits and center the data around certain value. Next, multiple realizations are generated by sampling the probability distributions and the failure pressures are calculated for each realization. Python will be used to realize the whole wellbore stability assessment.

5 STUDY CASE, HYPOTHESIS AND METHODOLOGY

This chapter presents the study case followed by the formulation of the hypothesis. Then, the methodology and assumptions used to explore the former problem is explained.

5.1 Study Case

The well COP-16 was planned as a horizontal producer on the south-eastern flank of the Ekofisk structure. The conductor was set during May 2018 and the main drilling started in May 2019. Figure 24 shows COP 16's wellbore schematic.

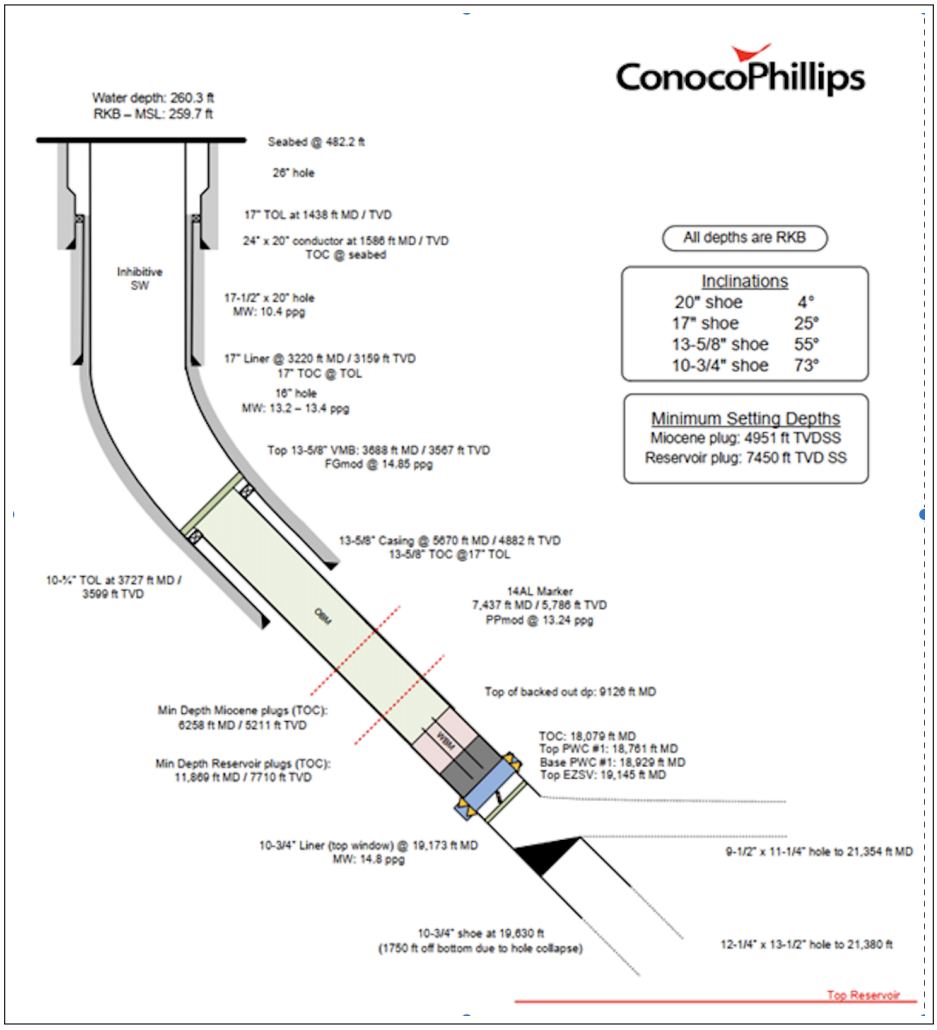


Figure 24: Well COP 16's wellbore schematic.

The 20" and 16" sections were drilled without any problems. A 17" liner and 13 5/8" casing where set and cemented without any issues and guaranteed a green well status from a well integrity point of view.

The 13 1/2" section was drilled without any major problems to the total depth (TD) of the section. Past 9360 ft MD (about 6700 ft TVD) cavings were observed on the shakers, indicating wellbore stability issues. A clear run out was performed and the TD of the section, 21380 ft MD (10050 ft TVD), was reached. When RIH during the liner installation, the liner stood up at 19845 ft MD, a depth correlated with the Balder Formation. After this, the well was abandoned and sidetracked. The 12 1/4 " x 13 1/2 " BHA required 8 days to pull out of the hole (POOH). After successfully retrieving it, a 12 1/4" cleanout assembly was run into the OH section. Through the resistivity log, significant hole enlargements were observed through all the section, which were more critical in the high angle interval of the section from 10900 ft MD (7200 ft TVD). Table 3 shows information regarding the FIT/LOT pressures and the mud density for each drilled section of COP-16.

Table 3: FIT/LOT Data.

Hole Size (in)	Casing Size (in)	Shoe Depth ft MD/TVD	Pressure (psi)	Mud Weight (ppg)	FIT/LOT ppg EMW
26	24 x 20	1586/1586	150	10.3	LOT = 12.1
17 1/2	17	3220/3159	295	13.1	LOT = 14.9
16	13 5/8	5671/4882	385	14.3	FIT = 15.8

During the drilling of the 13 1/2" section the mud window was followed and the mud weight was controlled, but still wellbore instability caused the well to be abandoned.

During the post-examination of well COP-16 it was determined that the well suffered a matrix shear failure between 8500 ft to 9200 ft due to low mudweight. Nevertheless, that shear failure does not explain all the wellbore stability issues in the 13 1/2" section, so other types of failure are suspected.

5.1.1 Cavings Report

There are three cavings reports: while drilling, while running clean and while pulling out of the hole. All reports show the depth at which the cavings were retrieved and a characterization by shape and type.

Table 4 shows a summary of the cavings report while drilling. In total, it includes 35 caving records, but for readability purposes 12 samples have been included in the table. From 15350 ft MD to 18900 ft MD all the samples were tabular platy and sub-angular cavings. From 18900 on, all the cavings showed planes of weakness. Between 19200 ft MD and 19900 ft MD all cavings were mechanical, angular and slightly rounded, same as between 20900 ft MD to 21380 ft MD (when RIH the liner).

Figure 25(a), 25(b) and 25(c) show some retrieved cavings. The slightly rounded cavings found in the shakers indicate they have been worked, probably they took longer time to circulate than non-rounded cavings. Planes of weakness were evident during the construction of the well. Also, tabular and angular cavings were shown, which indicate different types of rock failure.

Table 4: Summary of the cavings report.

Depth(ft MD/ft TVD)	Shape	Type
9360/6700	Angular	Mechanical
9360-15350/6700-8727	Angular	Mechanical
15350/8727	Platy, sub-angular	Tabular
18900/9796	Platty, sub-blocky	Weak bedding planes
19200/9862	Angular, slightly rounded	Mechanical, weak bedding planes
19900/10057	Angular, slightly rounded	Mechanical, weak bedding planes
21000/10385	Platty, sub-angular, slightly rounded	Mechanical, weak bedding planes
23000/	Platty, sub-angular, slightly rounded	Mechanical, weak bedding planes
26000/	Angular, slightly rounded	Mechanical, weak bedding planes
27000/	Angular, slightly rounded	Mechanical, weak bedding planes
20900/10344	Angular, slightly rounded	Mechanical, weak bedding planes
21380/10500	Angular, slightly rounded	Mechanical, weak bedding planes

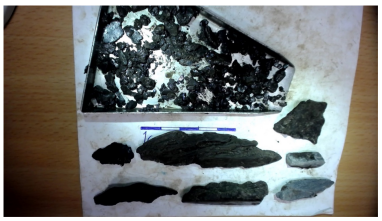
(a) Angular, platy, rounded, sub-angular cavings found at 15350 ft MD

(b) Mechanical, platy, sub-angular cavings showing weak bedding planes found at 18450 ft MD

(c) Mechanical, angular slightly rounded cavings showing weak bedding planes found at 21380 ft MD

Figure 25: Illustration of some retrieved cavings while drilling.

When circulating clean the hole three caving samples were recorded, two at 13100 ft MD and one at 13709 ft MD. The retrieved cavings were small, blocky to tabular, rounded to semi rounded and mechanical with weak bedding planes. At 13100 ft MD some big splintery cavings were retrieved. They are shown in Figure 26.



(a) Splintery, blocky and tabular cavings found at 13100 ft MD

(b) Blocky and tabular cavings found at 13100 ft MD

(c) Blocky and tabular cavings found at 13709 ft MD

Figure 26: Illustration of some retrieved cavings while circulating the hole clean.

When POOH, the amount of cavings increased drastically. Between 20566 ft MD and 8756 ft MD, 95 caving samples were recorded. Figure 27 shows some examples. Most of cavings were tabular, small to medium, but big angular cavings were also present.

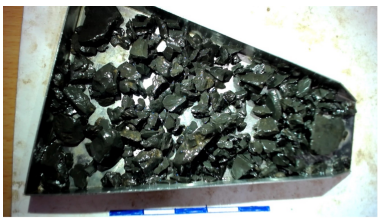
(a) Angular, slightly rounded cavings found at 20463 ft MD

(b) Angular and tabular cavings found at 20156 ft MD

(c) Blocky and tabular cavings with occasional angular cavings found at 19248 ft MD

(d) Tabular, sub-blocky cavings found at 17660 ft MD

(e) Blocky and tabular, rounded to sub-rounded cavings found at 15632 ft MD



(f) Small tabular and sub-blocky cavings found at 9568 ft MD

Figure 27: Illustration of some retrieved cavings while POOH .

Figure 27 shows the progression of cavings by depth. As seen in Figure 27(a), at deeper depths the cavings were angular in a concentration around 5%, the same as when drilling. In shallower depths, most cavings were tabular and blocky, with occasional angular cavings. Also, caving concentration increased.

As seen in Table 2, most cavings circulated while drilling were angular, suggesting a shear failure. Furthermore, many cavings showed weak bedding planes, so a failure along the bedding planes may have happened. While cleaning the hole and POOH, the

amount of cavings increased drastically, being most of them tabular, which suggests that a failure due to mud infiltration occurred [37]. Tensile failure is not considered as a possible caving source, as the bottom-hole pressure was neither lower than the pore pressure (underbalanced) nor higher than the minimum principal stress.

5.2 Summary of previous experiences

Before presenting the hypothesis to explain what happened in Well COP-16, five wellbore instability experiences in high inclination wells will be summarized in order to find similarities and try to explain what happened.

5.2.1 NCS, Oseberg field, Norway. 1998.

The first experience relates wellbore stability issues in Oseberg field in the paper "Bedding-related Borehole Instability in High-angle Wells", by Okland and Cook [45].

Well 30/9-B 30 was a extended-reach well drilled in 168 days. From the total drilling time, 71% corresponded to downtime, mainly caused by wellbore stability issues. The well was abandoned after its fifth side-track, and was renamed to 30/9-B-30X. Experiences from wells 30/9-B-48 and 30/6-C-26 / C-26A where employed to change the strategy and drill successfully well 30/9-B-34, which did not present stability issues.

The authors concluded that the borehole instability was produced due the pronounced bedding in the shale formation Draupne. The solution they implemented in well 30/9-B-34 was increasing the angle of attack, with a goal of it being always higher than 20°. They observed that the mud weight does not need to be increased if the angle of attack is optimized.

5.2.2 Pedernales field and Cusiana field, Venezuela and Colombia. 1999.

"Drilling in South America : A Wellbore Stability Approach for Complex Geologic Conditions" by Wilson et al., narrates two experiences in South America: in Pedernales field in Venezuela and in the Cusiana field in Colombia. This is one of the first papers about shear failure along the bedding planes. Both fields presented wellbore stability issues due to their high dip in case of the Pedernales field, where sediments dip up to 45°, or to small changes in the bedding dip angle due to the effect of faults in Cusiana field.

For both fields, adjustments on the mud plan where performed to guarantee proper wellbore stability. The authors acknowledge that the complex geology of the Andes foothills is prone to present shear failure along the bedding planes, but the findings of the paper are still applicable to other hydrocarbon provinces in the world. They note that the bedding does not be steeply dipping for the failure to occur, and that it may occur also in high-angle wells, specially when changes of azimuth are made.

5.2.3 Ordos Basin, China. 2019.

The third experience is narrated in Tong et al. paper, "New Modified Plane of Weakness Method Enables Drilling Horizontal Wells Successfully in Ordos Basin, China" [46].

Most horizontal wells had stability problems in the 8 1/2" section, with an inclination between 60° and 85°, including stuck pipe, overpull, pack-offs, hole collapse and large cavings, suggesting different types of rock failure. The cavings were angular and tabular. In comparison, vertical wells in the area did not present stability issues.

In a critical well, the mud weight employed was 1.20-1.25 g/cc. After applying a plane of weakness criterion, the appropriate mud for that section should be 1.40-1.45 g/cc. After the analysis, three wells were drilled with the mudweight that takes into account the plane of weakness failure. These three wells did not have any stability issues and reached TD successfully.

5.2.4 Marcellus Shale, Pennsylvania, U.S. 2021.

The fourth experience comes from the paper "Conclusive Proof of Weak Bedding Planes in the Marcellus Shale and Proposed Mitigation Strategies" from Kowan et al [47].

Wellbore instability problems were recurrent in lateral development wells in the Marcellus Shale in Greene and Washington counties in Pennsylvania. Two wells, a vertical one and a lateral were considered for the experiment. The vertical well did not show stability problems, while the lateral one showed isolated tight spots while drilling and several tight spots while tripping out, which required back-reaming and additional clean-up cycles. There is no record of cavings retrieved from the lateral well, but nearby lateral wells experienced tabular cavings, an indication of weak bedding planes.

The authors conclude that the wellbore instability problems in the Marcellus Shale are caused by weak planes. As mitigation, they propose increasing the mud weight to avoid shear failure, while avoiding mud infiltration in the bedding planes, which could also cause instability.

5.2.5 NCS, southwest to Bergen, Norway. 2022.

The fifth experience is explained by Kristiansen et al. in "A Troublesome Well Section: The Rock Mechanics Analysis" [48].

The problematic well was a multilateral drilled by AkerBP. Multilateral wells have been drilled before in the area, but this one had higher inclination and sail angle in the overburden. The problems happened in the 12 1/2" x 14 1/4", hole cleaning was challenging and cavings appeared during the problematic trip outs. Platy cavings indicating potential failure along the bedding planes were present among them. A caliper was not run, but time lapse resistivity logs indicate hole enlargement. Angular cavings were not observed. Deeper, blocky cavings were observed.

The authors did not find their wellbore stability analysis conclusive. They acknowledge including the plane of weakness failure into the stability analysis fits better the drilling data in the area, but arises concerns with the Balder formation. Another possible explanation offered to explain the stuck pipe and pack-offs is a possible avalanche of sediments caused by poor cleaning. The hypothesis is that the blocky cavings liberated with time from the weak planes, and this was exacerbated by the back reaming.

From the comparison between the fifth experiences some conclusions can be taken:

- Wellbore stability issues are common around the world in high angle wells while drilling through shale formations.
- Tabular cavings are present in the third, fourth and fifth cases, but angular cavings may be present too. The two different types of cavings indicate distinct rock failure mechanisms. In the two first cases the cavings were not considered in the analysis.
- The three last experiences indicate a time effect in the failure, as stability issues tend to increase when pulling out of the hole, this is attributed to an avalanche of sediments in the third case.
- Including weak bedding planes in the stability model increases the required mud weights. This must take into account the risk of infiltrating mud into the bedding planes, which may cause more stability issues. The first case is the only one that contemplates improving the design of the angle of attack, which may be a good solution to improve wellbore stability without increasing the mud weight.

5.3 Hypothesis

There are two main classes of mechanical wellbore instability mechanisms: isotropic rock failure, caused by failure of the intact rock, and anisotropic rock failure, caused by infiltration of fluid in the pre-existing planes of weakness (bedding planes, fractures) [37].

By the cavings report of Well COP-16, angular and tabular cavings were circulated from the wellbore. By the cavings definition presented in Chapter 2, angular cavings are caused by the mud's inability to support the wellbore wall, and by Edwards et al. definition ([37]), an isotropic failure. Tabular cavings are caused by the weakening effect of the drilling fluid on shales when it infiltrates the bedding planes, by Edwards et al. definition, an anisotropic failure.

Considering the end of well report, the stability issues were bigger in the deeper section of the well. By the cavings report, the presence of weak bedding planes is obvious while drilling, cleaning the hole and POOH, as in the type of cavings it was stated by the mudlogger, as it can be seen in Table 2 and Figure 25. As in the previous experiences, there seems to be a time effect in the stability issues, as the liner stood up, even though the well had been run clean previously. Also, the amount of tabular cavings increased with time. While drilling they were barely present, as angular cavings were predominant, as seen in Table 2, but the concentration of tabular cavings increased drastically while cleaning the hole and POOH, as showed in Figure 26 and Figure 27.

A possible explanation is that a shear failure along the weak planes occurred through all the 13 1/2" section, but was more serious when the critical hole enlargements started to occur, below 7200 ft TVD. The first collapse was caused by a shear failure along the bedding planes, which was observed at surface as angular cavings. After the failure of the first shales, the contact between drilling fluid and the bedding planes increased, allowing the mud infiltration to start, causing an anisotropic rock failure. This caused a progressive avalanche of tabular cavings, which went mostly unnoticed due to the time effect and the difficulty of circulating this type of cavings due to their small size and flat shape [37]. When the well was cleaned out the angular cavings were circulated, but an amount of tabular cavings stayed at bottom, while the mud kept infiltrating the weak planes. Also, back-reaming produced a mechanical stress on the wellbore, helping cavings to get loose. In the end, the liner stuck due to the avalanche of tabular cavings that could not be circulated. The mechanical stress of POOH contributed to the avalanche, thus increasing the amount of cavings at shallower depths.

In this hypothesis, the failure is caused by a shear failure of the rock along the bedding planes, which allows the drilling fluid to infiltrate the cracks, propagating the shear failure, as described by Skea et al. [37]. The propagation of the shear failure caused a progressive avalanche of tabular cavings that may have provoked the stuck pipe incident. Then, to avoid a wellbore collapse, a mud window which includes shear failure due to failure along the bedding planes must be generated, as it is the failure that provokes the consecutive failures.

5.4 Root Cause Analysis and Methodology

The hypothesis states that the failure occurred due to shear failure along the bedding planes, and that the mud window must be updated to include shear failure due to weakness planes. A root cause analysis (RCA) will be presented to highlight the challenges in wellbore stability during the drilling phase of ERD wells in Ekofisk.

5.4.1 RCA

Three possible causes of wellbore instability issues have been identified, as shown in the root cause analysis(RCA) in Figure 28.

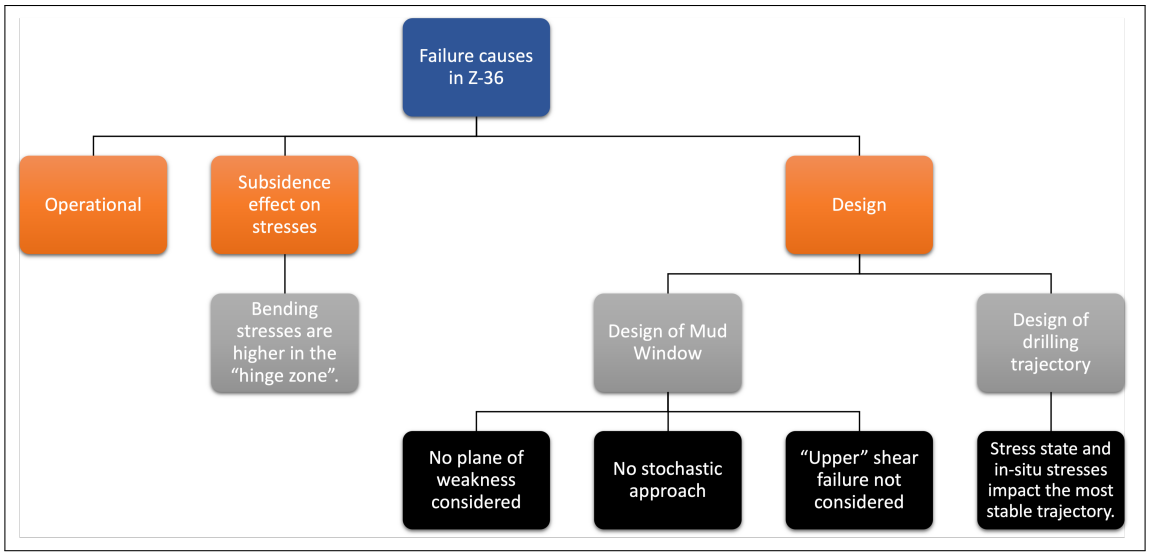


Figure 28: Root cause analysis.

The first possible cause is related to operational failure due to human errors. Wellbore stability is affected by the ECD, which is controlled from surface with the mud weight.

The second possible cause is subsidence effect on stresses. As explained on Chapter 1, Ekofisk is a field greatly affected by subsidence, and it has altered both the reservoir and the overburden. Subsidence is a dynamic process that affects constantly the in-situ stresses, which needs to be updated in the model periodically. The dynamic change in the stress field causes uncertainty in the inputs for the failure models, which may affect the wellbore stability assessment.

The third possible cause is the design, which can be the design of the mud window or the design of the drilling trajectory.

- Design of mud window. Due to the lack of data the failure along the bedding planes is usually not accounted for during well planning [49]. The plane of weakness may be predominant over the shear failure of the rock matrix, specially in high inclinations, as the failure of the plane of weakness is highly dependant on the interaction between the wellbore's and bedding planes' interaction [50]. There is a high uncertainty in the bedding plane orientation caused by the presence of faults, as it may change drastically between the foot-wall and the hanging-wall [50]. Also, although stochastic methods like the QRA have been proposed [43], the industry approach towards the construction of the mud window is mainly deterministic. Another consideration is the occurrence of a shear failure in the upper bound of the mud window [35], which has not been widely documented, and probably does not involve wellbore deformation, as the high pressures do not allow the debris to fall into the wellbore.
- Design of drilling trajectory. Drilling trajectories can be optimized by taking into account the in-situ stresses [51]. It has been shown that the fault stress field determines the most stable trajectories [52]. Also, the angle of attack can be designed

during the planning phase to decrease the risk of having wellbore stability issues [53].

5.4.2 Methodology

As shown in the root cause analysis, three different causes may have provoked the failure in the well COP-16. The proposed ways to solve this problems are:

- Operational errors. According to well reports there were no human errors during the drilling process. This possible cause will not be further examined in this project.
- Subsidence effect on stresses. As subsidence changes the stresses in an unknown way a stochastic analysis by employing Monte Carlo simulations will be applied. Ranges will be presented instead of deterministic values to account for the uncertainty.
- Design. A mud window that accounts for failure along the bedding plane and shear failure at high wellbore pressures will be presented. Wellbore stability assessments for all the possible bedding plane orientations in Ekofisk will be performed. Cases will be constructed with the optimistic and worst possible scenarios. Uncertainty will be considered.

The ordered methodology to perform the wellbore stability assessment will be the following: First, the shear failure criteria Mohr-Coulomb and Modified Lade will be compared to select the one that will be used for the column construction. Second, a sensitivity analysis will be performed on the plane of weakness failure to understand the impact of its friction factor and cohesion. It is known that there is a reduction in those parameters compared to the ones in the rock matrix. As there is only one available measure from a core, different scenarios will be studied. Then, a mud window showing the failures as function of depth will be shown. This mud window will include the mud weight employed during the drilling of well COP-16, which will show graphically if a failure was produced. After this, all possible orientations for the bedding planes will be considered, as it is a parameter with high uncertainty, and the optimistic and worst scenarios versus depth will be plotted in the mud window. Finally, a stochastic analysis will be applied to give ranges for the failures in order to produce a safer mud window.

6 RESULTS

In the following chapter the results of the applied methodology explained in the Chapter 3 will be presented. All the Python codes employed to generate the data for the Figures are present in Appendix 1. Appendix 2 contains the Gnuplot codes that plotted all the Figures.

Before generating the mud windows, the stress and strength data employed in this project will be shown.

Figure 29 shows the stresses σ_x, σ_y, σ_z, $\tau_x y$, $\tau_x z$ and $\tau_y z$ as function of depth.

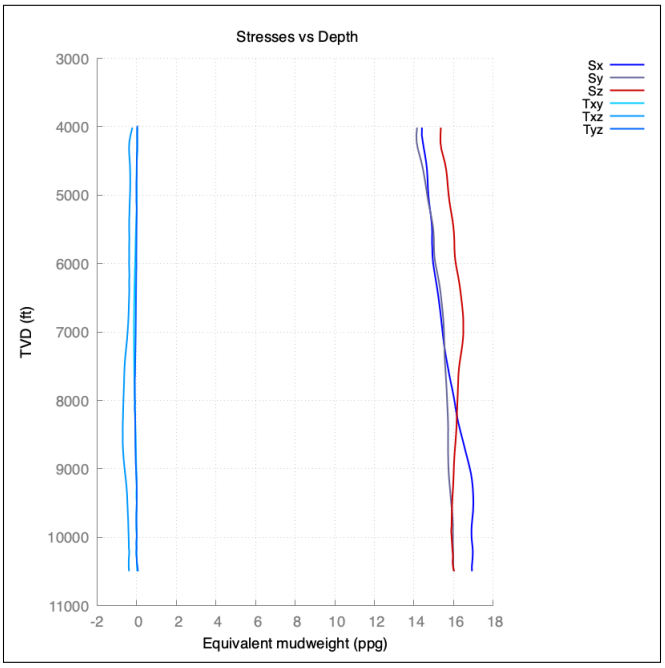


Figure 29: Stresses in the borehole coordinate system. Generated with data from ConocoPhillips.

In wellbore stability, strength is based on the cohesion and tensile strength. Figure 30 shows this two properties as function of depth.

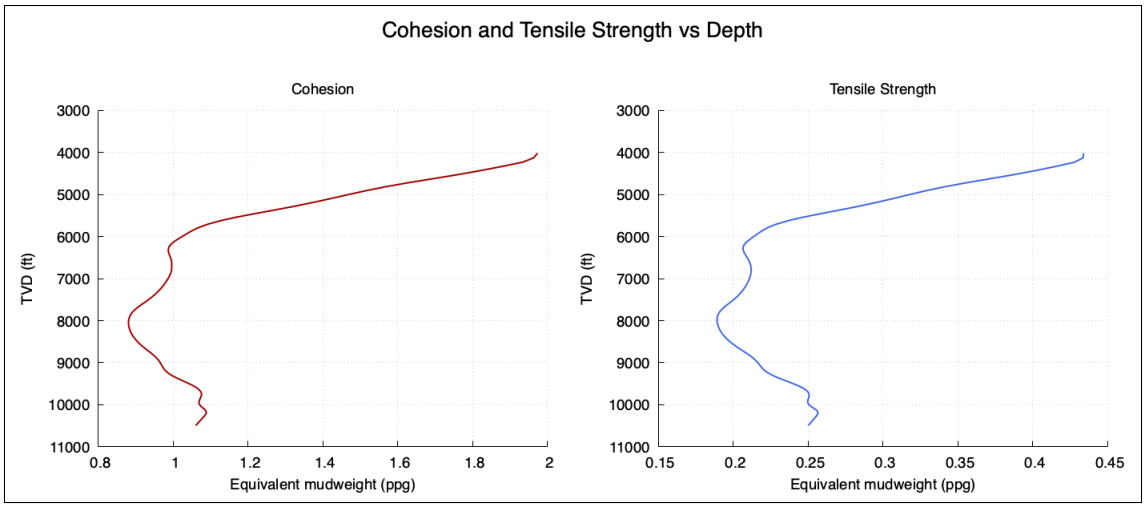


Figure 30: Cohesion and tensile strength as function of depth. Generated with data from ConocoPhillips.

In Figure 30 it can be observed that the cohesion and tensile strength follow the same behaviour. There is a reduction in both between 6000 ft and 9500 ft, which is likely caused by undercompaction that leads to overpressure. Overpressured areas are expected to present narrower mud windows [28].

6.1 Comparison between Mohr-Coulomb and Modified Lade Criteria

In "Wellbore Stability Assessment of an Anisotropic Shale Formation in the North Sea" [5] a Mohr-Coulomb criterion was employed, which is the most common failure criterion. The company suggested employing a Modified Lade criterion, due to past experiences in Ekofisk. In this section, the two criteria will be compared.

The main difference between the two criteria is that Modified Lade criterion takes into account the strengthening effect of the principal intermediate stress and is non-linear, as seen in Equation 24. It is expected that the Modified Lade criterion will give lower shear failure pressures, as the strengthening effect of the principal intermediate stress is considered. Failure criteria that do not considerate the principal intermediate stress, like the Mohr-Coulomb criterion, are considered as conservative, as they tend to present higher shear failure pressures, thus narrowing the operational mud window.

The plots presented in this section will show the shear failure as function of depth, both in the upper bound and lower bound. In them, the pore pressure will be shown in gray, the minimum stress in dark-gold, the employed mud weight in COP-16 in black dots, the tensile failure in purple and the shear failure in blue.

Figure 31 shows the failure modes as function of depth with two different shear failure curves for the upper and lower bound. The stronger shade of blue corresponds to the pressures calculated with a Modified Lade criterion ("M. L. Criterion"), while the lighter ones to the pressures calculated with the Mohr-Coulomb criterion ("M. C. Criterion").

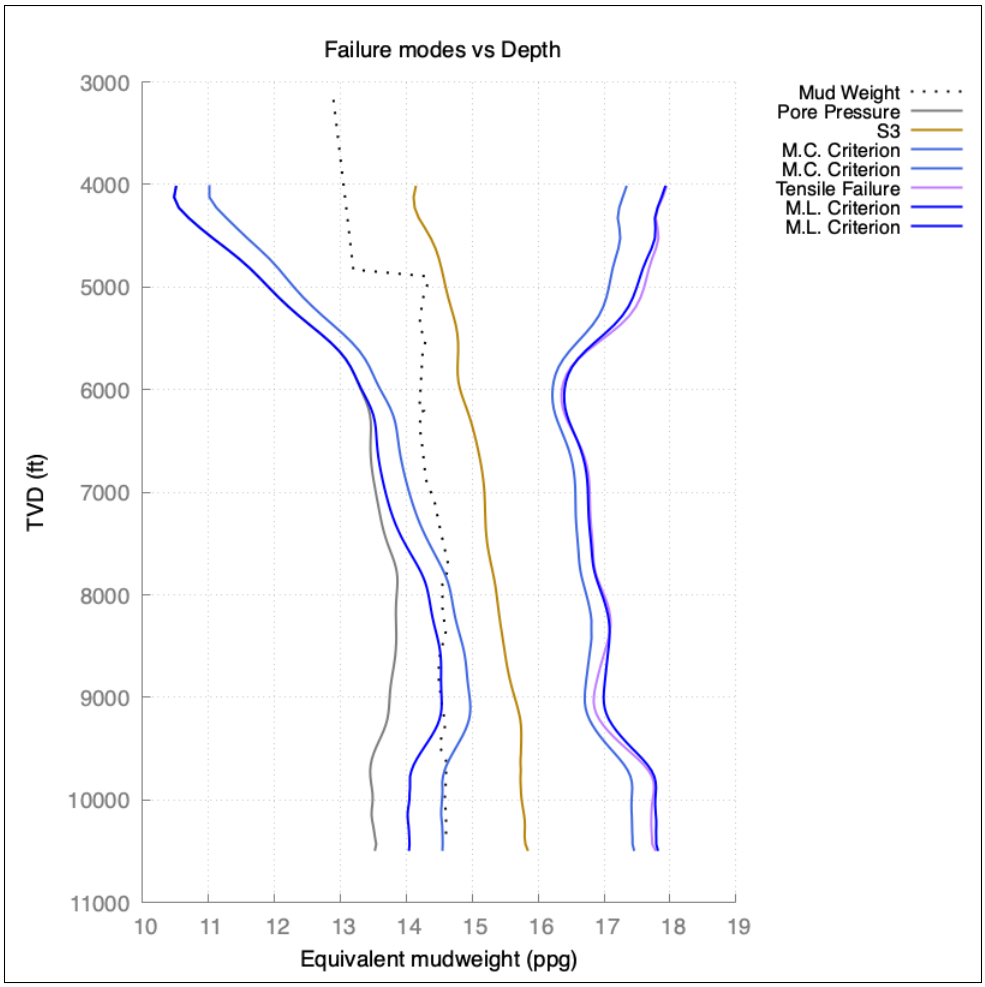


Figure 31: Shear failure criteria as function of depth

As expected, the Modified Lade criterion predicts lower shear failure pressures in the lower bound, as it takes into account the principal intermediate stress. In the upper bound, the shear failure pressures are similar to the tensile failure pressure. By comparing the employed mud weight, the shown in Figure 31 suggest a shear failure between approximately 7700 ft TVD and 9700 ft TVD. ConocoPhillips does not employ this criterion for Ekofisk due to it not considering the strengthening effect of S2. Figure 32 shows the principal stresses at the borehole wall that provoke the shear failure. Note that the S3 that appears in the plot is the minimum principal stress at the borehole wall at the moment of failure, it is different than the one plotted in the mud windows, which is local.

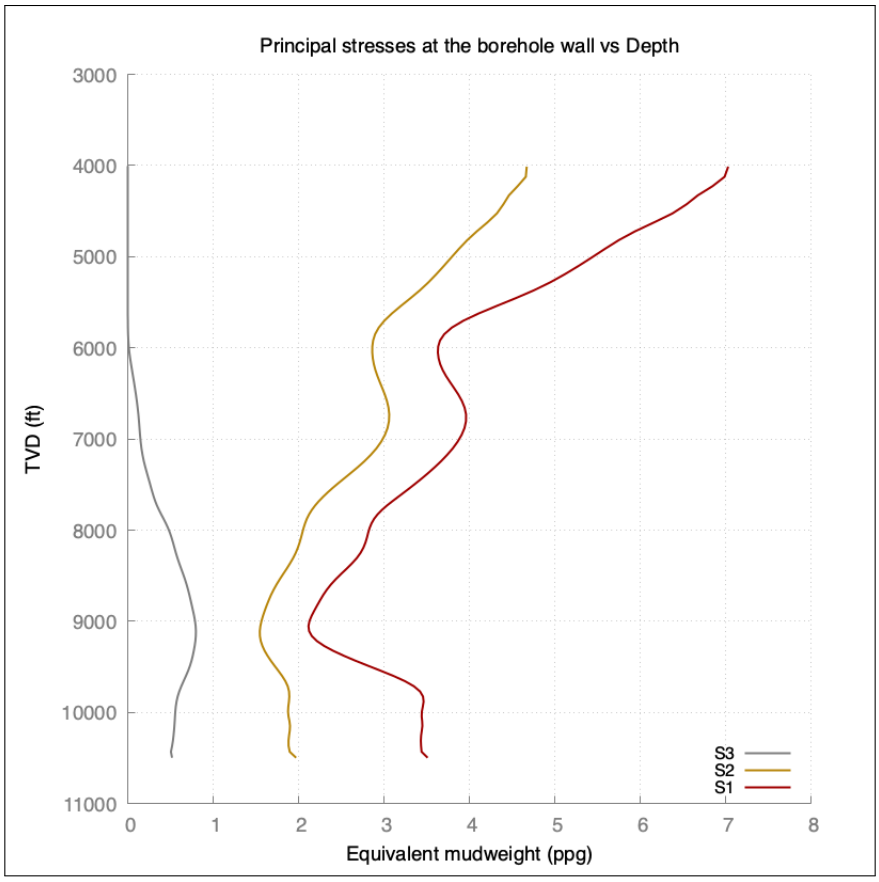


Figure 32: Shear failure generating stresses as function of depth.

As it can be seen, the intermediate principal stress (S2) has a similar behaviour to the maximum principal stress (S1), and it is closer to it than to the minimum principal stress (S3). Due to its relatively high values it cannot be excluded from the analysis, thus the Modified Lade criterion describes better the shear failure criterion than the Mohr-Coulomb criterion. The Mohr-Coulomb criterion will not be considered for the rest of this analysis and Modified Lade will be preferred over it. The shear failure limits by Modified Lade criterion predict a failure between 8500 ft and 9200 ft approximately, as it was observed during the COP-16's posterior evaluation.

Figure 33 shows a version of the plot employed in "Wellbore Stability Assessment of an Anisotropic Shale Formation in the North Sea" that is consistent with the observation of the Modified Lade criterion being more accurate for this area than the Mohr-Coulomb criterion.

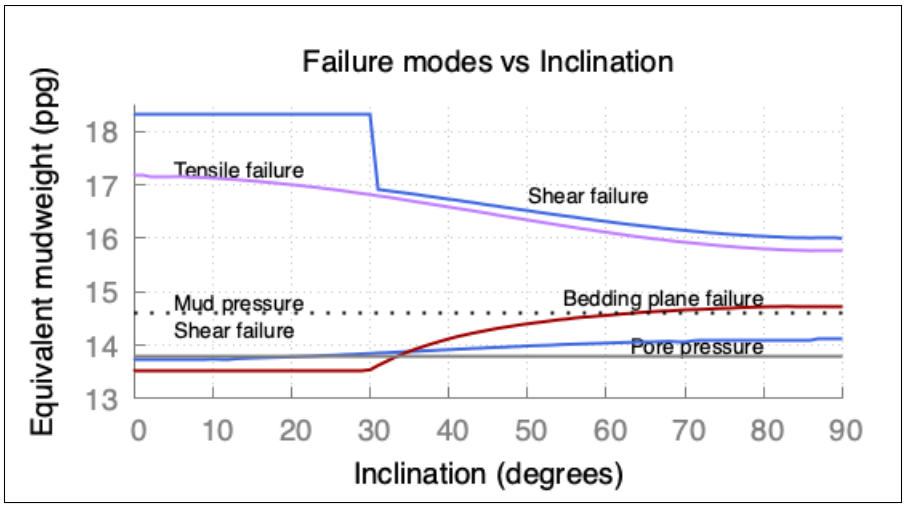


Figure 33: Failure modes as function of inclination with Modified Lade criterion.

After changing the criterion for shear failure to Modified Lade it is observed that the bedding plane failure is predominant over the pore pressure from 35°, 10° lower than when applying a Mohr-Coulomb criterion (Figure 2). In the upper bound, the shear failure is no longer predominant over the tensile failure, as it happens at higher pressures. For less than 30° the shear failure is higher than the upper bound considered in the code, causing it to behave as a straight line.

Before generating a mud weight window that shows the failure modes as function of TVD, the cohesion (S_w) and friction factor of the plane of weakness (μ_w) must be considered. These two parameters affect directly the integrity of the plane of weakness, as shown in Equation 29, but only the measurement from the core shown in Table 1 is available. As there are no measurements available, the cohesion (S_w) and friction factor for the plane of weakness are assumed to be reduced values compared to the cohesion and friction factor from the rock's matrix (μ_0), which are available. Therefore, a sensitivity analysis on these two properties will be performed in the next section.

6.2 Sensitivity Analysis on Bedding Plane's Cohesion and Friction Factor

Three scenarios will be considered: Two where only one property varies and a last one where both properties change at the same time.

All the plots presented in this section will show the plane of weakness failure as function of depth. In them, the pore pressure will be shown in gray, the minimum stress in dark-gold and the employed mud weight in COP-16 in black dots.

Figure 34 shows the variation of the plane of weakness failure with the varying μ_w. Three possible scenarios are considered, a 30% reduction in μ_0, a 50% reduction and a 60% reduction (respectively 0.7 μ_0, 0.5 μ_0 and 0.4 μ_0).

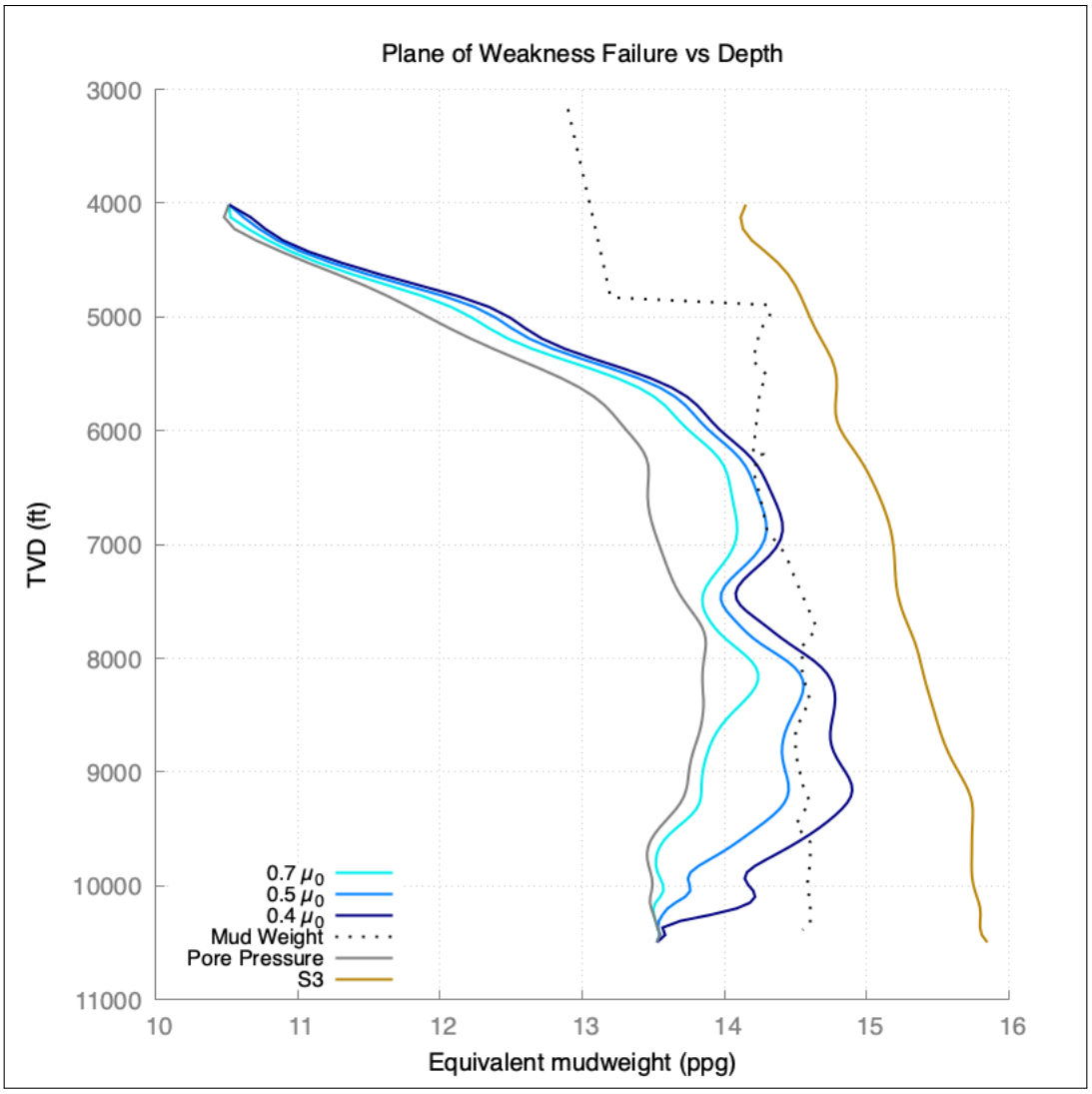


Figure 34: μ_w effect on plane of weakness failure.

It is observed that a higher reduction produces higher plane of weakness failure pressures, being the 60% reduction the most critical one. By comparing the failure curves with the employed mud weight, it can be indicated that 40% and 50% reductions in μ_0 would produce failures in the well.

Figure 35 presents the variation of the plane of weakness failure with the varying S_w. Three possible scenarios are considered, a 10% reduction in S_o, a 30% reduction and a 50% reduction (respectively 0.9 S_o, 0.7 S_o and 0.5 S_o).

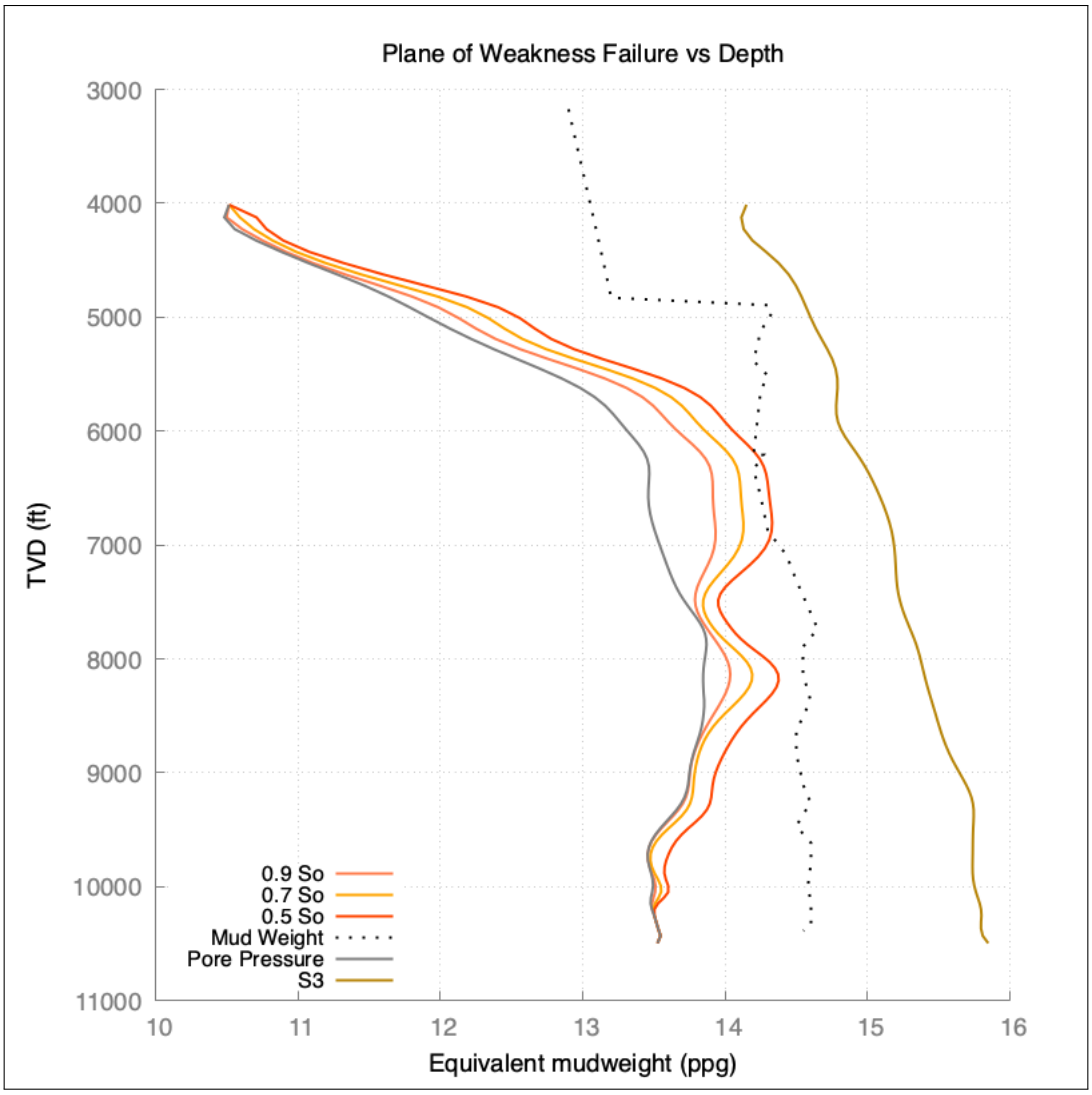


Figure 35: S_w effect on plane of weakness failure.

Between the three cases, only a 50% reduction in S_o would provoke bedding plane failure. Comparatively, it seems that the effect of the cohesion is lower at higher depths. This occurs because cohesion increases with depth, causing lower failure pressures.

For the last scenario, a simultaneous reduction is applied with a 10% reduction in S_o and 50% in μ_0. Figure 36 shows the effect of a 50% reduction in μ_0 and a 10% reduction in S_o simultaneously. At the right part of the Figure, the effect of the two reductions separately is presented.

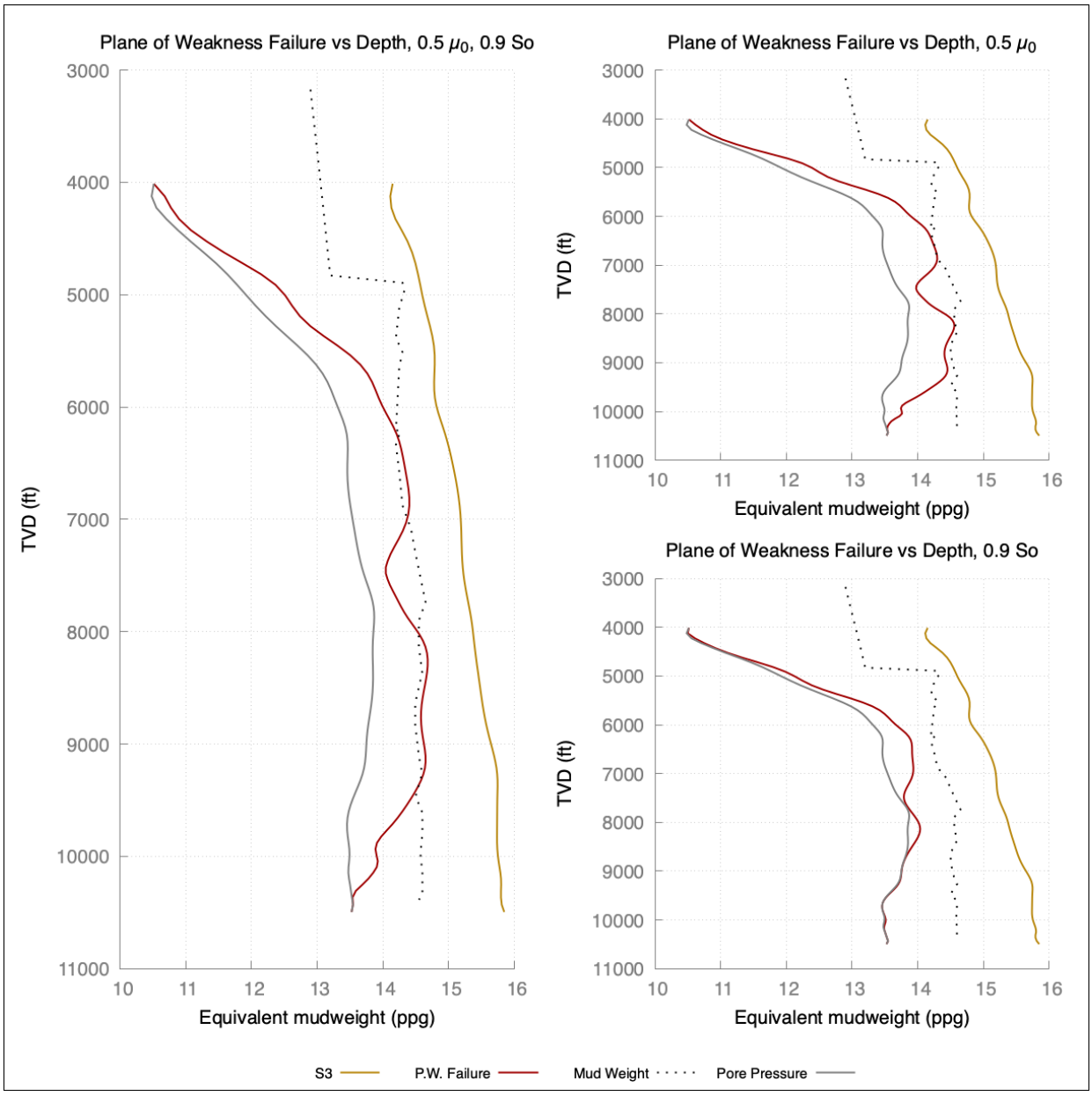


Figure 36: μ_w and S_w effect on plane of weakness failure.

The effect of both reductions simultaneously is particularly noticeable in the lower part. When there is a reduction of 10% in S_o, the failure pressures below 7500 ft TVD are low, in many depths they are the same as the pore pressure, but when combined with the reduction in μ_0 the failure is noticeable.

Combining the reduction in 50% reduction of μ_0 and 10% in S_o gives two areas of failure. This behaviour is in line with the cavings retrieved while drilling, therefore it will be assumed as valid for the rest of the thesis. Figure 37 shows the mud window that shows all the failure modes as function of TVD.

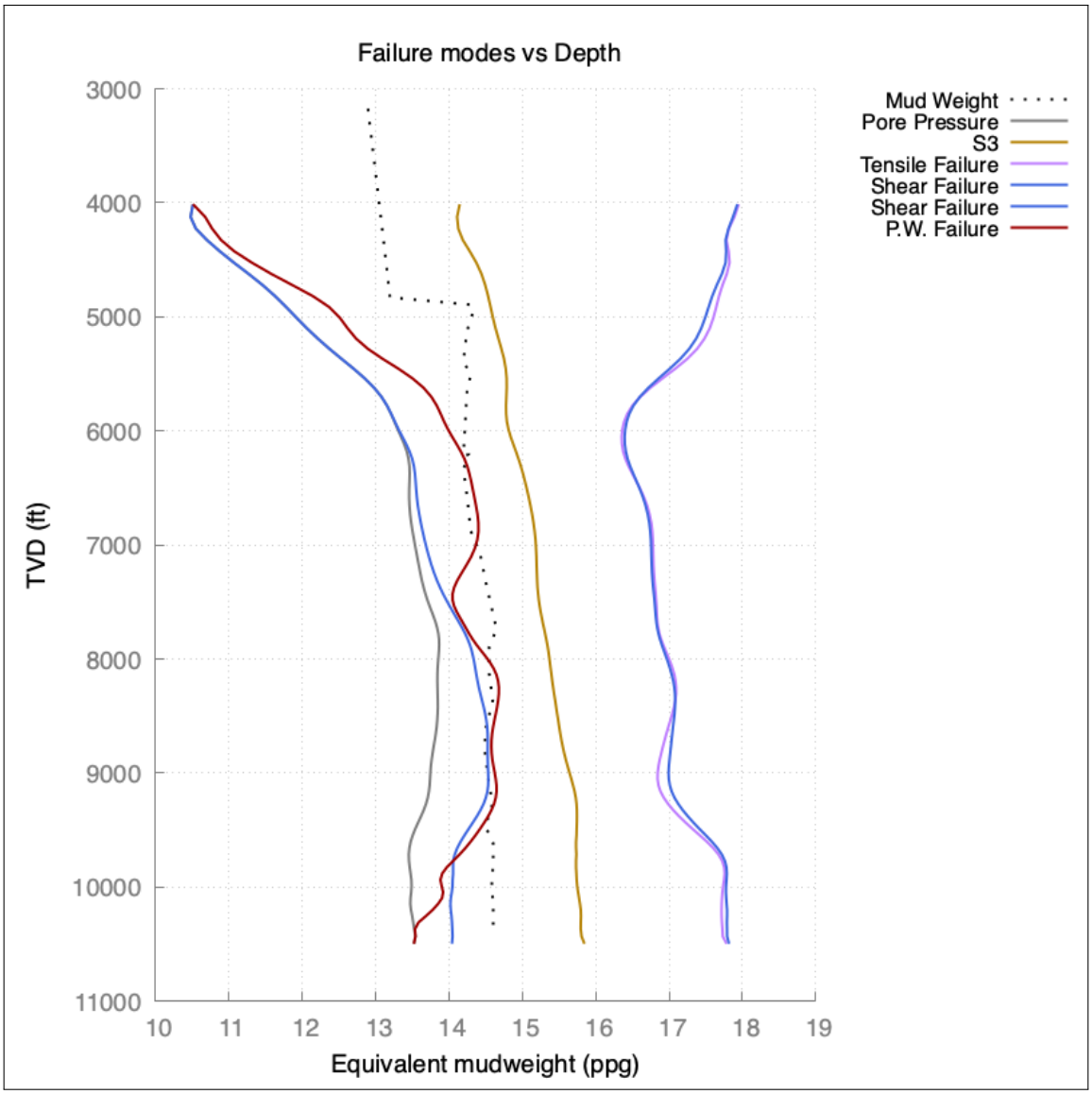


Figure 37: Failure modes as function of depth.

As it can be observed, there are two different sections where shear failure was produced. The shallowest one, between 6100 ft TVD and 7100 ft TVD and the deepest one, between 7900 ft TVD and 9500 ft TVD. In the deepest section, both the bedding plane failure criterion and the matrix failure criterion are violated.

As stated, this mud window is considered as valid due to it following the cavings reports, but still involves uncertainty. The input data comes from the planning phase, which may have small variations in reality. μ_w and S_w were tuned and assumed as having a constant reduction for all depths, which is an assumption made due to missed data. This reductions will be assumed to be the mean for all the column in the stochastic analysis.

The bedding plane orientation was assumed to be the same as the measured one in the core and to be constant for all depths. Before performing the stochastic analysis, the effect

of the bedding plane orientation on wellbore stability will be studied in the next section.

6.3 Variation of the Plane of Weakness Orientation

One of the parameters that affects the bedding plane failure is the angle of attack. The industry recommendation is to drill perpendicular to the bedding plane orientation, and various authors have shown the importance of the trajectory in wellbore stability [52] [45] [5]. As this is a study case for Well COP-16 different trajectories will not be considered. It is more relevant to consider the different bedding plane orientations and the failure pressure that each of them will produce. Therefore, the bedding plane orientation will be considered as uncertain in this subsection. In order to propagate uncertainty, the inclination of the bedding plane, θ_w, and the azimuth of the bedding plane, ϕ_w will be considered as uniform distributed variables, as it is assumed that each orientation has the same probability of occurrence. The bedding plane orientation will vary between 0° and 15° of inclination and 0° and 360° azimuth. 5000 repetitions are considered. Figure 38 shows the behavior of the failure along the bedding planes pressure with depth after applying uncertainty in the orientation of the plane of weakness. Its mean is plotted in a red hard line. A safety range of 1 standard deviation (SD) is presented in a clearer shade of red. Note that the mud weight is not plotted, as the objective of this subsection is to observe the variability generated by an uncertain bedding plane orientation.

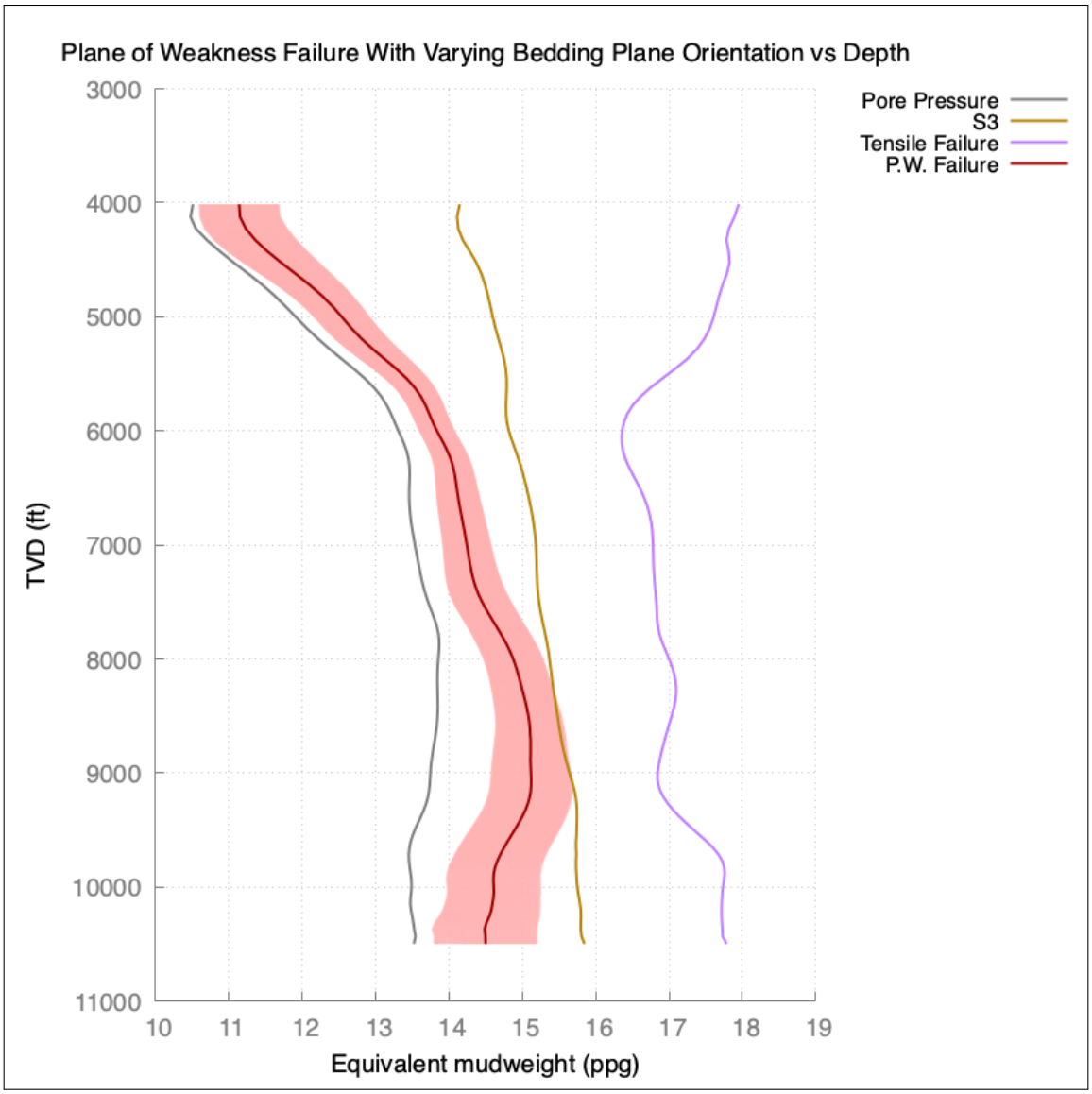


Figure 38: Failure along the planes of weakness as function of depth. The figure includes the mean failure with 1 SD of safety factor.

Figure 38 shows that the safety factor of 1 SD has a high variance in width, being narrow around 6000 ft and wider towards the bottom of the window. This indicates that in deeper depths the data is more spread, and that an uniform variation by depth cannot be expected. Therefore, five different depths will be studied in order to check if there is any kind of range that can be expected.

The bedding plane orientation will vary between 0° and 15° of inclination and 0° and 360° azimuth, with a step of 1° for both inclination and azimuth. All possible orientations are considered. Table 5 shows a summary of the obtained statistical parameters by depth.

Table 5: Summary of some statistical parameters for the bedding plane variation on the plane of weakness failure.

TVD(ft)	Maximum(ppg)	Mean(ppg)	Minimum(ppg)	SD(ppg)
6462	14.41	14.11	13.46	0.26
6939	14.56	14.24	13.51	0.31
8517	15.73	15.15	13.88	0.45
9501	15.67	14.85	13.52	0.64
10495	15.45	14.45	13.53	0.70

The minimum and maximum values, mean and standard deviation (SD) are presented. It is observed that the standard deviation ranges from 0.26 to 0.70 ppg, reflecting the behaviour observed in Figure 38. As this information in not enough to establish ranges, a deeper analysis will be performed.

In this subsection, Figures 39-43 will show a histogram and the Cumulative Density Function (CDF) for the failure pressures due to bedding plane failure. In the CDF, the maximum and minimum values, the mean, the SD, Q1 and Q3 are shown. The quartiles Q1 and Q3 correspond to the points that are equal or higher to 25% and 75%, respectively. The dashed vertical lines represent the area covered by 1 standard deviation around the mean value.

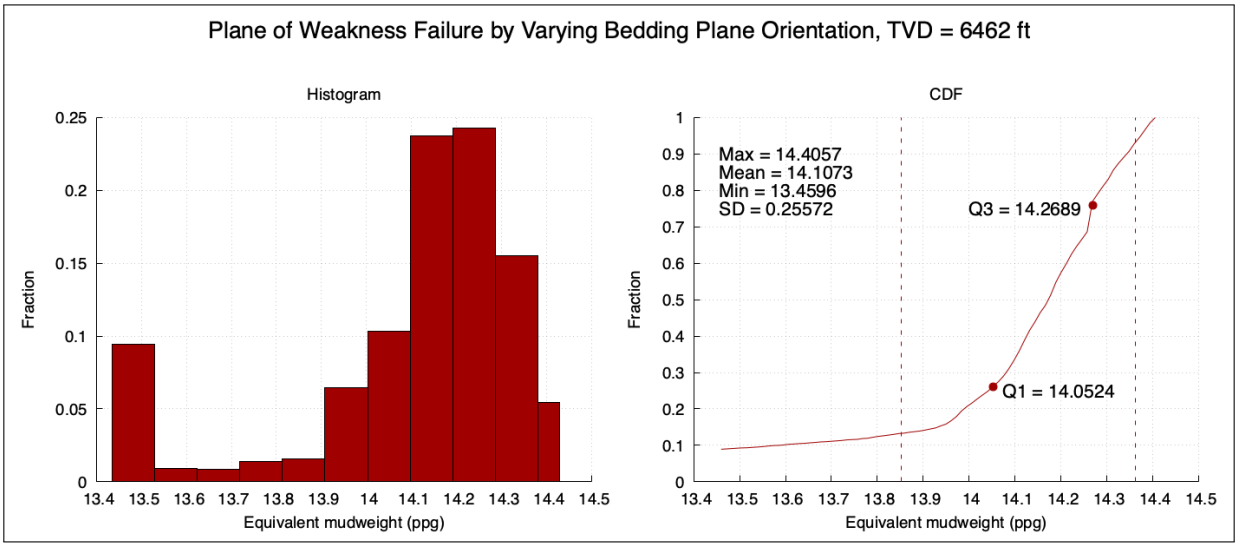


Figure 39: Effect of Bedding Plane Orientation, TVD = 6462 ft

The histogram in Figure 39 presents bins of data and their fraction of occurrence. It is clear that the data distribution is not normal. Also, approximately 50% of the data ranges from 14.1 to 14.3 ppg, and 65% between 14.1 to 14.4 ppg, in a 0.3 ppg range. The CDF shows the cumulative distribution, showing how the probability of getting a specific value or less. Q1 and Q3 are presented, which are 14.05 and 14.27 ppg, respectively, meaning

that 50% of the data is spread in a 0.22 ppg range. 1 SD centered around the mean covers from 13.85 to 14.7, approximately 82 % of the data.

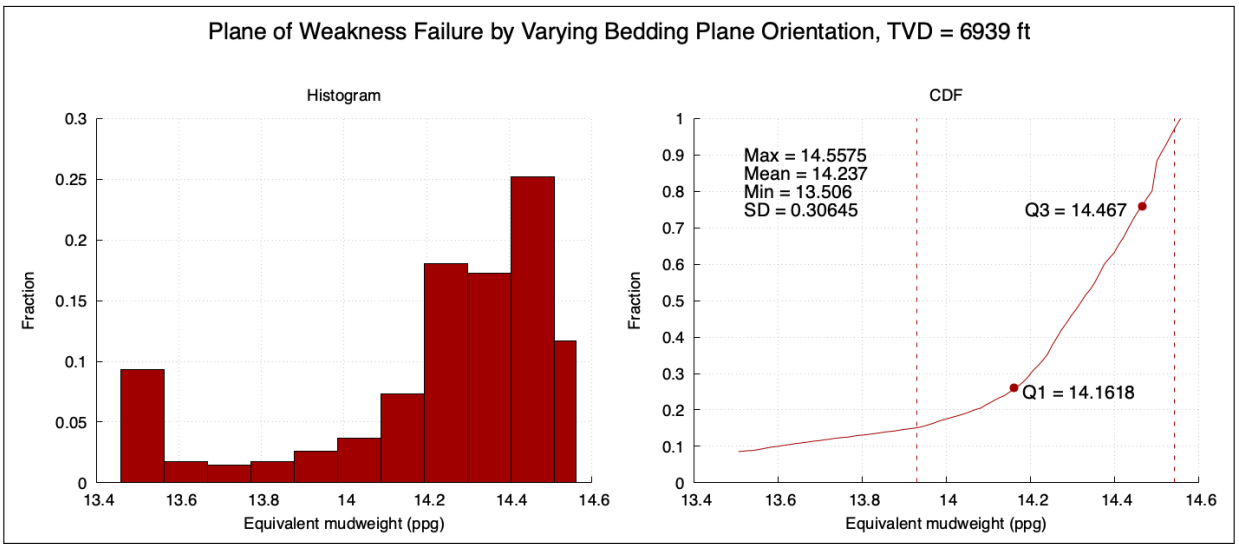


Figure 40: Effect of Bedding Plane Orientation, TVD = 6939 ft

The histogram in Figure 40 shows that approximately 60% of the data ranges from 14.2 to 14.5 ppg, meaning that the mean is in the lower part of this range, in a 0.3 ppg range. In the CDF, Q1 and Q3 have a value of 14.16 and 14.47 ppg, 50% of the data spreads in a range of 0.31 ppg. 1 standard deviation centered around the mean covers from 13.85 to 14.45, approximately 82 % of the data.

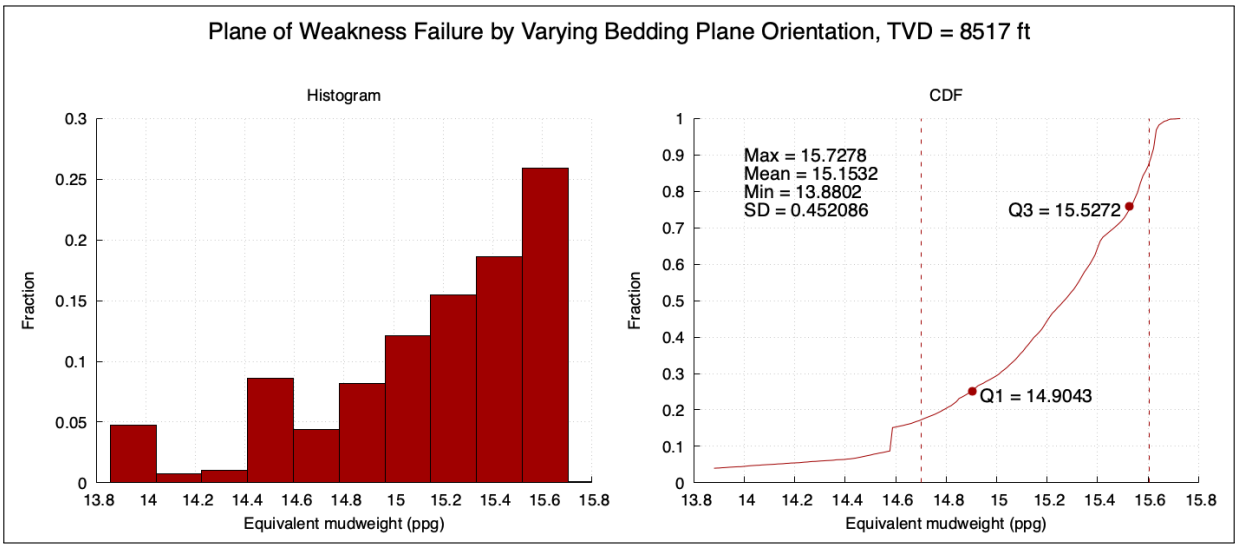


Figure 41: Effect of Bedding Plane Orientation TVD = 8517 ft

In Figure 41, 25% of the data corresponds to the bin 15.5-15.7 ppg. In the CDF, 25% of the data is spread between the minimum and 14.90, meaning that only 25% of the data

is spread in a 1,02 ppg span, while 50% ranges is concentrated in a 0.62 ppg range. 1 standard deviation centered around the mean covers from 14.7 to 15.6, approximately 71 % of the data.

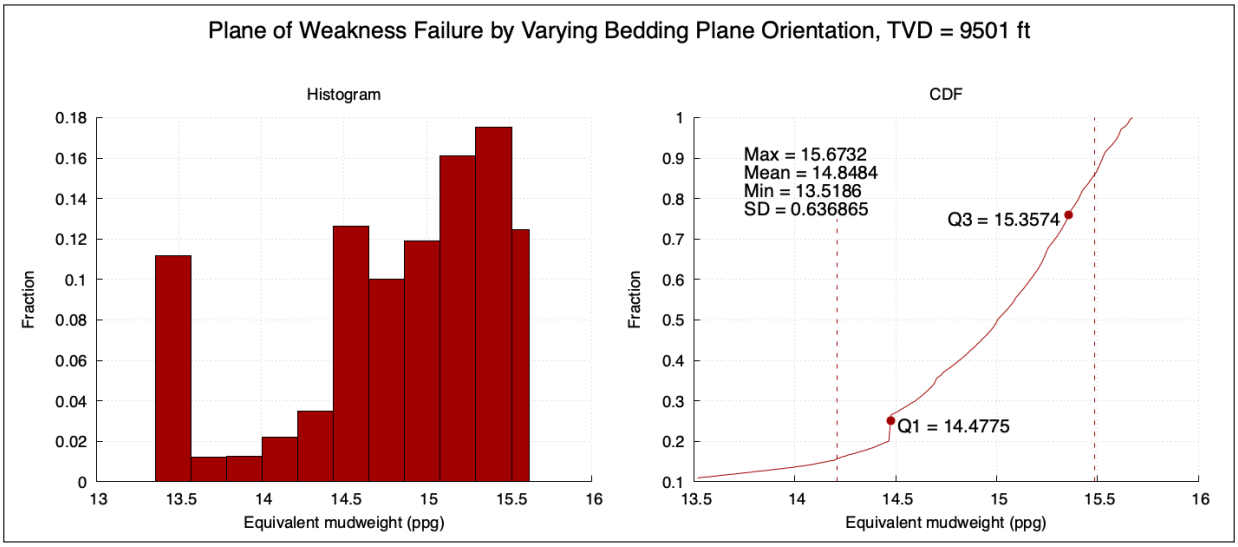


Figure 42: Effect of Bedding Plane Orientation TVD = 9501 ft

In Figure 42 most of the data is concentrated between 14.4 ppg and the maximum value, but 11% of the data is located in the bin 13.4-13.52, meaning that all the data in this bin corresponds to the minimum value, 13.51. In the CDF, 25% of the data is spread between the minimum and 14.48, while 50% ranges between 14.48 and 15.36 ppg, in a 0.88 ppg span. 1 standard deviation centered around the mean covers from 14.2 to 15.45, approximately 70 % of the data.

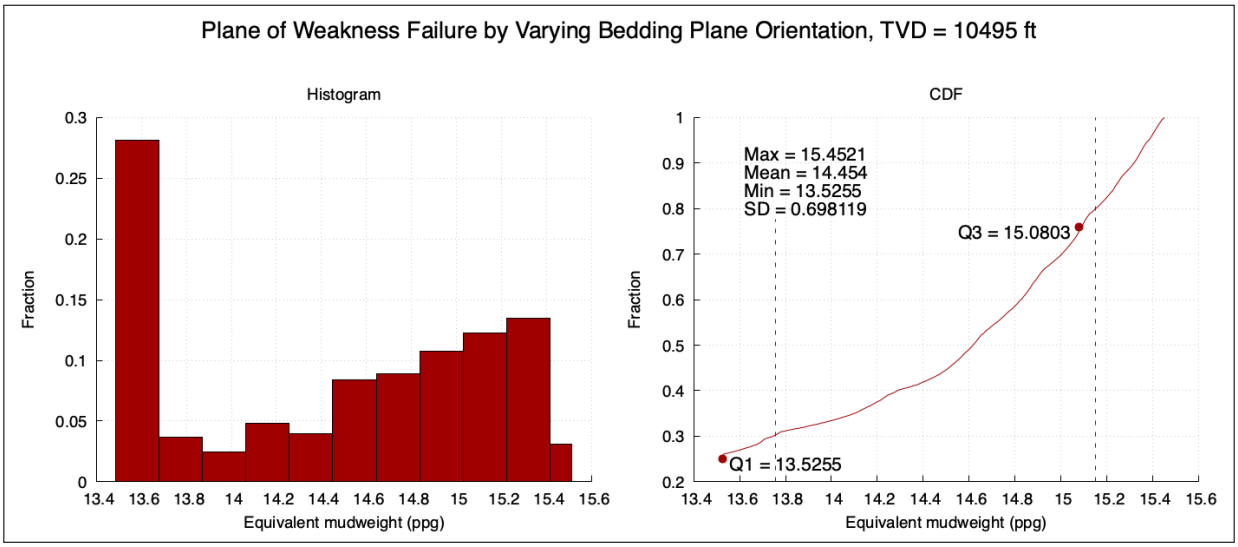


Figure 43: Effect of Bedding Plane Orientation TVD = 10495 ft

In Figure 43 the data is widely spread, with almost 30% of the data is accumulated in the first bin, corresponding to the minimum value, 13.53 ppg. This is reflected in the CDF, as Q1, the 25% of the data is located just before the curve starts. 50% ranges between 13.53 and 15.09 ppg, in a 1.56 ppg range. 1 standard deviation centered around the mean covers from 13.75 to 15.16, approximately 50 % of the data.

As observed in the 5 past cases, the different orientations do not generate similar distributions. Considering the mean with 1 SD also generates ranges with different proportions of data, which ranges from 50% to 82%. Furthermore, the data does not seem to accumulate in the same area, as seen in the last case, where the probability of obtaining the minimum value was more than 25%. The spread of data is varying too, which is reflected in the SD, and it is specially high at higher depths. Table 6 shows a summary of the ranges between different points in the distributions. Max - Min refers to the full range of the data, Q1 - Min is the 25% of the data, Max - Q3 corresponds to the data points between the 75% and 100% of the data and Q3, to the 75% of the data.

Table 6: Summary of some range lengths for the bedding plane variation on the plane of weakness failure.

TVD(ft)	Max-Min	Q1-Min	Max-Q3	Q3-Q1	Q3-Min	Mean,1 SD	% Mean, 1 SD
6462	0.95	0.59	0.14	0.22	0.81	0.51	82 %
6939	1.05	0.66	0.09	0.31	0.96	0.61	82 %
8517	1.85	1.02	0.20	0.62	1.65	0.90	71 %
9501	2.15	0.96	0.32	0.88	1.84	1.28	70 %
10495	1.9	0	0.37	1.55	1.58	1.4	50 %

The summary presented in Table 6 does not conclude any statistical range as better than the others, as the data range length has a big variance between the considered depths. 100 % of the data is accumulated in a range that varies between 0.95 and 2.15 ppg. The lower 25% (Q1 - Min), between 0 and 1.02 ppg. The upper 25% (Max - Q3), from 0.09 to 0.37. The middle 50% of the data (Q3-Q1), which excludes the most optimistic cases (lower 25%) and the worst cases (upper 25%), may have a length varying between 0.22 and 1.55 ppg. Taking into account the mean and 1 standard deviation would give a range that covers around 68% of the data for normal distributions, but the distributions are not normal, thus why it covers from 50% to 82% of the data. Therefore, the orientation of the plane of weakness has a big impact in the failure pressure, but cannot be predicted nor has a expected range, as the data does not follow a pattern.

As there is not an expected range in the failure pressures due to the orientation of the plane of weakness, and the distributions have a high variance, the stochastic analysis will consider the plane of weakness orientation as a uniform distributed parameter, where every value has the same probability of occurrence. Additionally, different scenarios will

be provided in the stochastic mud window in order to let the Operator decide based on the risk they are willing to take.

6.4 Stochastic Analysis

In this subsection, the stochastic analysis will be performed. The stochastic analysis will only be performed on the lower shear failure and on the shear failure along the bedding planes, as there is certainty on the pore pressure and minimum principal stress plots. 10000 repetitions are considered for each depth. As the Operator will not drill with mud weights higher than the minimum principal stress, uncertainty will not be considered for the tensile failure and upper matrix shear failure. The parameters and distributions for which the uncertainty will be considered are shown in Table 7.

Table 7: Uncertain variables and distributions.

Variable	Distribution	Description
σ_x	Normal	SD of 5 Pa, mean equal to the used value for the deterministic analysis
σ_y	Normal	SD of 5 Pa, mean equal to the used value for the deterministic analysis
σ_z	Normal	SD of 5 Pa, mean equal to the used value for the deterministic analysis
τ_{xy}	Normal	SD of 5 Pa, mean equal to the used value for the deterministic analysis
τ_{xz}	Normal	SD of 5 Pa, mean equal to the used value for the deterministic analysis
τ_{yz}	Normal	SD of 5 Pa, mean equal to the used value for the deterministic analysis
υ	Normal	SD of 0.08, mean equal to the used value for the deterministic analysis
ϕ	Triangular	Centered around the used value for the deterministic analysis with a variation of 5°
So	Normal	SD of 2 Pa, mean equal to the used value for the deterministic analysis
θ_w	Uniform	Range between 0° and 360°
ϕ_w	Uniform	Range between 0° and 15°

Most variables are assumed to follow a normal distribution, as most phenomena in nature follow this behaviour. A triangular distribution is assumed for the friction factor, ϕ, as the triangular distribution has the benefit of limiting the obtained values and centering the data around certain point. The orientation of the plane of weakness, given by θ_w and ϕ_w, are assumed to follow an uniform distribution, as it will be assumed that each orientation has the same probability of happening.

The standard deviation for normal variables has been considered to be low, as Ekofisk is a mature field that is properly documented. The biggest uncertainty is related to the orientation of the plane of weakness, as it is not well documented, and small changes in θ_w and ϕ_w can affect strongly the failure along the bedding planes pressure, as seen in the previous subsection. Also, faults may change the orientation of the bedding planes [50], increasing the uncertainty in these parameters. In this analysis μ_w and S_w are assumed to be reductions of μ_0 and S_0 as in Subsection 4.2. The reductions will be considered to be constant, so uncertainty will be considered for μ_0 and S_0.

All the variables employed to analyze the matrix shear failure, except $S0$, follow normal distributions, so a range of 1 SD centered around the mean would cover approximately 68% of the data [54]. For the failure along the plane of weakness, four variables do not follow normal distributions: ϕ, So, θ_w and ϕ_w, so it is expected for this range to include close to 68% of the variables too. Figure 44 shows the failures modes as function of depth with uncertainty applied over the matrix shear failure and the failure along the bedding planes. The means of this curves are plotted in hard lines. A safety range of 1 SD is presented in in clearer colors.

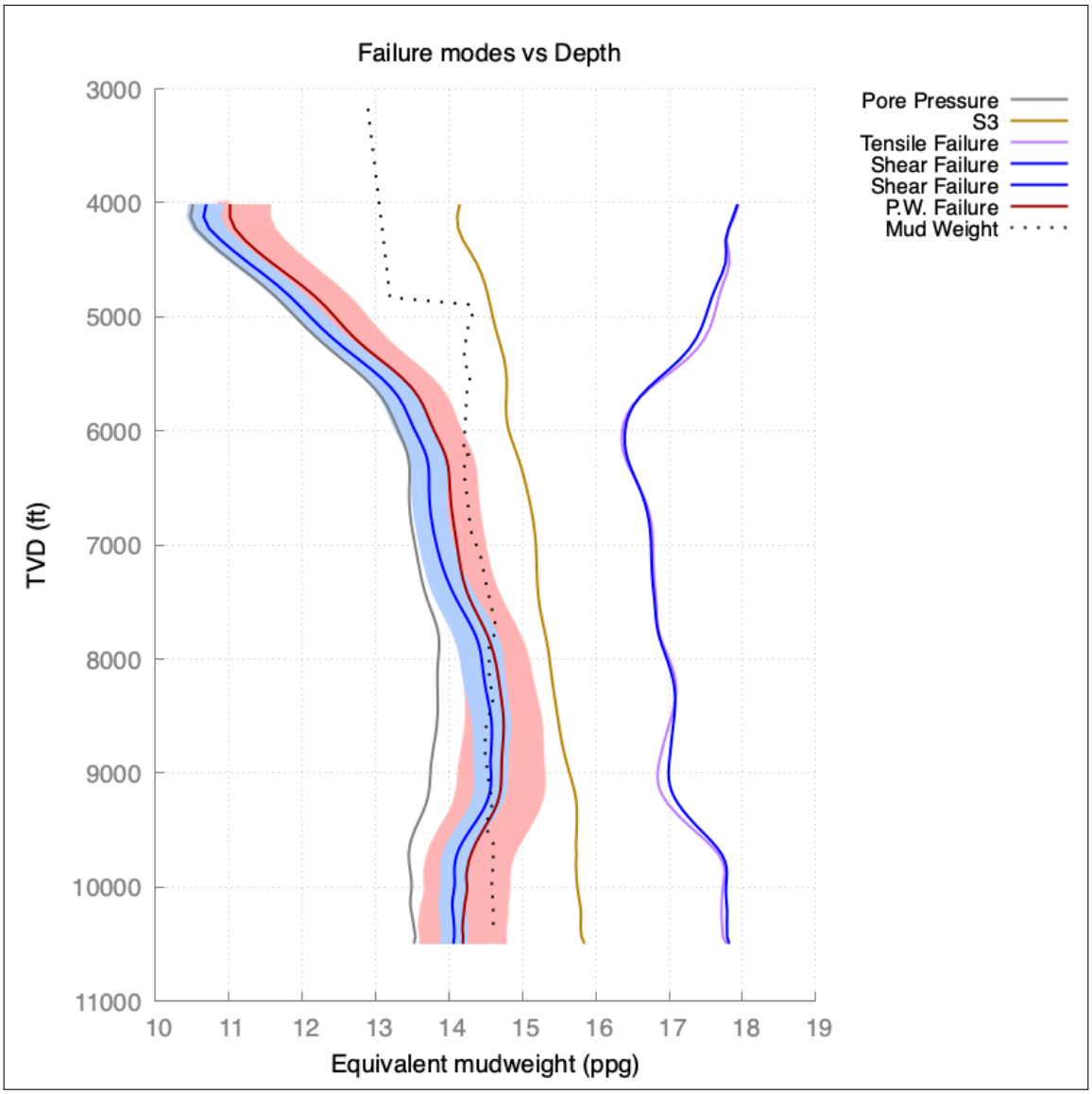


Figure 44: Failure modes as function of depth. Safety factors of 1 SD are applied to the matrix shear failure and shear failure along the bedding planes.

After applying the stochastic analysis, it is observed that the Figure 44 has a similar behavior to Figure 37, specially for the matrix shear failure. When applying uncertainty, the safety range predicts a possible constant failure, instead of two differentiated areas of failure. Also, it is noticeable that the mean of the failure along the bedding planes shows failure from 8000 ft, not at shallower depths, although the safety range of 1 SD considers it from 6000 ft.

As shown in Subsection 6.3, the the bedding plane orientation affects greatly the variation of the obtained failure pressures. Figure 45 presents the mean value with a safety range of 1 SD and three cases: P25, P75 and P90. As P25 includes 25% of the data, it is the same as Q1, it is a high risk scenario. P75 includes 75%, which is a safer scenario, and it

is the same value as Q3. Finally, P90 is a low risk scenario, with the downside of it limiting too much the operational window.

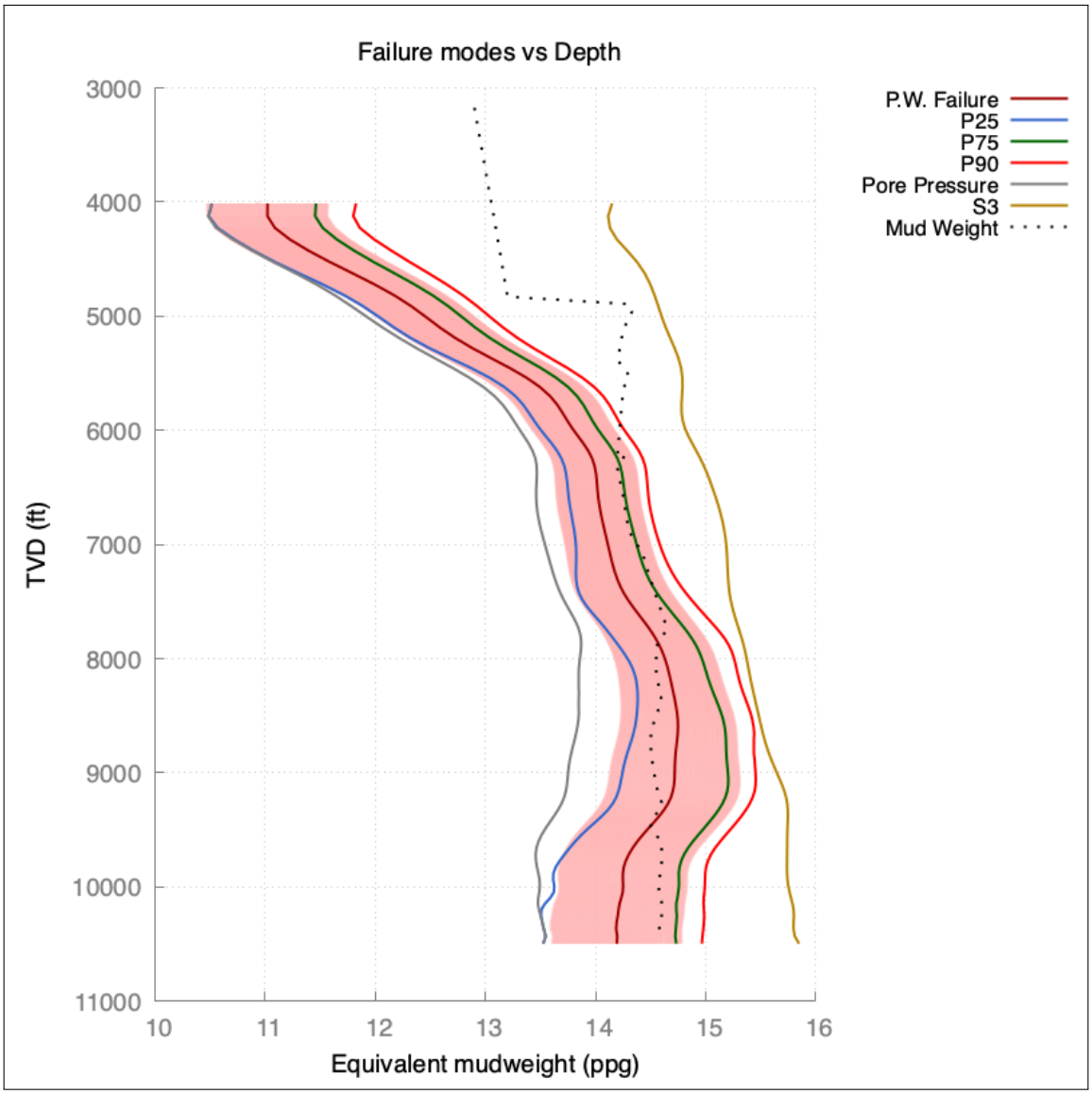


Figure 45: Failure along the planes of weakness as function of depth. The figure includes the mean failure with 1 SD of safety factor and the cases P25, P75 and P90.

Figure 45 shows that the safety factor is almost contained between P25 and P75, meaning that 1 SD for the generated case covers around 50% of the data. As expected, P90 limits the operational window too much. P75 and the mean plus 1 SD include 75% of the data or more, while offering a operation margin. As in Figure 44, the failure is expected from 6000 ft.

7 DISCUSSION

The results obtained in Section 6 indicate that the main cause of the wellbore stability issues found on well COP-16 was a shear failure along the bedding planes, which was expected after analyzing the caving reports and supposed by the hypothesis. Between 8500 and 9000 ft TVD, matrix shear failure was also provoked. Cavings are a good indication of wellbore instability, and their observation allows to differentiate different types of failure.

One of the biggest concerns while analyzing the failure along the bedding planes was the values of Sw and μ_w. As there is not information available a percentage of reduction compared to the same variables in the rock matrix was assumed. This uncertainty was dealt with by applying the stochastic analysis, but it is far-reached assuming that the same reduction is expected in the whole overburden, so it is mandatory to acquire more data about these variables. The orientation of the plane of weakness was also a cause of concern, but it was reduced to a range from 0° to 15° in inclination and from 0° to 360° in azimuth. The orientation of the plane of weakness has a high degree of uncertainty and has a big impact in the failure pressure's variability.

Both the deterministic and stochastic mud windows, Figure 37 and Figure 44, show that the mud weight employed was not high enough from 6000 ft on. As shown in this thesis, the failure pressures depend on several factors, but one of the most important ones is the rock strength, which in wellbore stability is based on the cohesion and tensile strength. As it is observed in Figure 30, from 6000 ft to 9500 ft there is a reduction in both parameters, which is caused by undercompaction. Figure 37 and Figure 44 reflect these properties' behavior in the operational mud windows. By comparing the strength behavior with the generated mud windows it is observed that rock strength and pressure failures are strongly correlated. Undercompacted formations are a high risk area, as the loss of strength increases the risk of failure.

By using the ranges presented in Figure 45, different cases can be established. While drilling, it is important to keep enough operational range, so employing the range of P90 is not viable. Also, it is important to consider formations like Balder, which present a high risk of losing circulation. By the law of large numbers, the mean value is close to the expected values when performing many repetitions. As 10000 repetitions where performed for each depth, it can be assumed that the means plotted in Figure 44 and Figure 45 are the expected values. These values imply a higher risk than P75, but their probability of happening is higher and offer a good mud operational window. Depending on the certainty and the risk the operator is willing to take, the safe operational mud weight can be restricted between the expected value and P75, in the area covered by the 1 SD safe margin.

Through the experiences presented in Subsection 5.2 and the analysis in Section 6, it can be concluded that extended-reach wells like COP-16 present a narrow mud window due to the shear failure along the bedding planes. If the high inclination wants to be kept, including the failure along the bedding planes is mandatory. A possibility to guarantee

wellbore stability would be employing Measured Pressure Drilling (MPD) systems.

Other possible solutions to avoid wellbore stability issues in high inclination wells depend on the drilling planning. One solution could be decreasing the maximum inclination. As shown in Figure 2 and Figure 33, the failure along the weakness planes pressure increase with inclination, so by reducing the inclination the operational range would increase. This is not desirable, as there are many situations where high inclinations are required. Another solution summarized in Subsection 5.2 was from the paper "Bedding-related Borehole Instability in High-angle Wells", by Okland and Cook [45]. According to the authors, the angle of attack can be optimized in order to maintain a high inclination and low mud-weight. This method requires a good comprehension of the orientation of the bedding planes in the area, and thus, more geological data.

8 CONCLUSIONS

- The well COP-16 presented two type of failures: shear failure along the bedding planes and matrix shear failure.
- Extended-reach wells and high inclination wells are prone to present shear failure along the bedding planes due to their high inclination. If they are drilled through laminated rocks like shales the risk increases. It is mandatory to include this type of failure in their operational mud window.
- Shear failure along the bedding planes is difficult to control. Well COP-16 and other experiences suggest that this failure may progress into mud infiltration into the bedding planes, generating big amounts of tabular cavings.
- Aside from designing mud windows that include the shear failure along the bedding planes, the improved design of the angle of attack may prevent wellbore stability issues.
- The rock strength and failure pressure risk are highly correlated. Undercompacted formations suppose a high risk of presenting wellbore instability.
- More information on the cohesion (Sw) and friction factor (μ_w) is required in order to increase the certainty on the failure along the bedding planes pressures.
- The orientation of the bedding planes is a parameter with high uncertainty that causes high variation in the failure pressures.
- Stochastic approaches like the Monte Carlo simulations employed in this thesis may generate safer mud windows. It can be applied to reduce the uncertainty in areas where there is lack of information. Another implementation may be the planning of exploration wells in areas where data has a high degree of uncertainty.
- In order to reduce the uncertainty in Ekofisk, it is recommended to retrieve more cores and record the cohesion, friction factor and orientation of the bedding planes.

9 FURTHER WORK

- Generate correlations between the bedding plane' cohesion and friction factor and the matrix's cohesion and friction factor.
- Design improved angle of attacks and measure their failure pressures for well COP-16.
- Analyze the failure along the bedding planes in different trajectories.

References

[1] Caineng Zou. *Unconventional petroleum geology*. Elsevier, 2017.

[2] Tianshou Ma, Ping Chen, Chunhe Yang, and Jian Zhao. “Wellbore stability analysis and well path optimization based on the breakout width model and Mogi–Coulomb criterion”. In: *Journal of Petroleum Science and Engineering* 135 (2015), pp. 678–701.

[3] Bernt Aadnoy and Reza Looyeh. *Petroleum rock mechanics: drilling operations and well design*. Gulf professional publishing, 2019.

[4] Erling Fjær, Rune Martin Holt, Per Horsrud, and Arne Marius Raaen. *Petroleum related rock mechanics*. Elsevier, 2008.

[5] Jairo Alexander Diaz Lopez and Hans Joakim Skadsem. “Wellbore Stability Assessment of an Anisotropic Shale Formation in the North Sea”. In: *IADC/SPE International Drilling Conference and Exhibition*. OnePetro. 2022.

[6] *Field: Ekofisk*. https://www.norskpetroleum.no/en/facts/field/ekofisk/.

[7] T Petersen. “An Evaluation of Leak-Off Prediction methods and an Estimation of In-Situ Stresses for the Overburden at the Ekofisk Field”. PhD thesis. M. Sc. thesis, Høgskolen i Stavanger, 1995.

[8] NB Nagel. “Ekofisk field overburden modelling”. In: *SPE/ISRM rock mechanics in petroleum engineering*. OnePetro. 1998.

[9] *Exceptional Ekofisk*. Accessed: 2022-01-25. URL: https://www.npd.no/en/facts/news/general-news/2021/exceptional-ekofisk/.

[10] *Ekofisk field*. URL: https://petrowiki.spe.org/Ekofisk_field (visited on 05/23/2022).

[11] Merete Vadla Madland. “Water weakening of chalk: A mechanistic study”. In: (2005).

[12] *Greater Ekofisk Area Facts*. URL: https://static.conocophillips.com/files/resources/20190712ekofisk-facts-final.pdf.

[13] *Ekofisk*. URL: https://www.conocophillips.no/our-norway-operations/greater-ekofisk-area/ekofisk/ (visited on 01/25/2022).

[14] B Agarwal, H Hermansen, JE Sylte, and LK Thomas. “Reservoir characterization of Ekofisk field: a giant, fractured chalk reservoir in the Norwegian North sea—history match”. In: *SPE Reservoir Evaluation & Engineering* 3.06 (2000), pp. 534–543.

[15] Tina Puntervold, Skule Strand, Raed Ellouz, and Tor Austad. “Why is it Possible to Produce Oil from the Ekofisk Field for Another 40 Years?” In: *International Petroleum Technology Conference*. OnePetro. 2014.

[16] University of Oslo. *Norland Group*. https://nhm2.uio.no/norges/litho/nordland.php.

[17] British Geological Survey. *Norland Group*. URL: https://webapps.bgs.ac.uk/lexicon/lexicon.cfm?pub=NORD.

[18] University of Oslo. *Hordaland Group*. https://nhm2.uio.no/norges/litho/hordaland.php.

[19] NPD. *CO2 storage Atlas Norwegian North Sea*. The Norwegian Petroleum Directorate.

[20] University of Oslo. *Rogaland Group*. https://nhm2.uio.no/norges/litho/rogaland.php.

[21] University of Oslo. *Ekofisk Formation*. https://nhm2.uio.no/norges/litho/ekofisk.php.

[22] University of Oslo. *Tor Formation*. https://nhm2.uio.no/norges/litho/tor.php.

[23] NB Nagel. “Compaction and subsidence issues within the petroleum industry: From Wilmington to Ekofisk and beyond”. In: *Physics and Chemistry of the Earth, Part A: Solid Earth and Geodesy* 26.1-2 (2001), pp. 3–14.
[24] LY Chin and NB Nagel. “Modeling of subsidence and reservoir compaction under waterflood operations”. In: *International Journal of Geomechanics* 4.1 (2004), pp. 28–34.
[25] LY Chin, RR Boade, B Nagel, and GH Landa. “Numerical simulation of Ekofisk reservoir compaction and subsidence: Treating the mechanical behavior of the overburden and reservoir”. In: *Rock Mechanics in Petroleum Engineering*. OnePetro. 1994.
[26] Lars Ottemöller, HH Nielsen, K Atakan, Jochen Braunmiller, and J Havskov. “The 7 May 2001 induced seismic event in the Ekofisk oil field, North Sea”. In: *Journal of Geophysical Research: Solid Earth* 110.B10 (2005).
[27] Karl Terzaghi and RALPH B Peck. “Soil Mechanics”. In: *Engineering Practice. John Wiley and Sons, Inc., New York* (1948).
[28] Mark D Zoback. *Reservoir geomechanics*. Cambridge university press, 2010.
[29] Nicholas Thompson, Jamie Stuart Andrews, Håvard Reitan, and Nuno Eládio Teixeira Rodrigues. “Data Mining of In-Situ Stress Database Towards Development of Regional and Global Stress Trends and Pore Pressure Relationships”. In: *SPE Norway Subsurface Conference*. OnePetro. 2022.
[30] JM Strout and TI Tjelta. “Excess pore pressure measurement and monitoring for offshore instability problems”. In: *Offshore Technology Conference*. OnePetro. 2007.
[31] *Drillstem test*. URL: https://glossary.oilfield.slb.com/en/terms/d/drillstem_test (visited on 02/11/2022).
[32] AA Ahmed Abdelaal, SE Salaheldin Elkatatny, and AA Abdulazeez Abdulraheem. “Pore Pressure Estimation While Drilling Using Machine Learning”. In: *ARMA/DGS/SEG 2nd International Geomechanics Symposium*. OnePetro. 2021.
[33] Pichita Booncharoen, Thananya Rinsiri, Pakawat Paiboon, Supaporn Karnbanjob, Sonchawan Ackagosol, Prateep Chaiwan, and Ouraiwan Sapsomboon. “Pore pressure estimation by using machine learning model”. In: *International Petroleum Technology Conference*. OnePetro. 2021.
[34] M King Hubbert and David G Willis. “Mechanics of hydraulic fracturing”. In: *Transactions of the AIME* 210.01 (1957), pp. 153–168.
[35] Liqin Ding, Jianguo Lv, Zhiqiao Wang, and Baolin Liu. “Borehole stability analysis: Considering the upper limit of shear failure criteria to determine the safe borehole pressure window”. In: *Journal of Petroleum Science and Engineering* (2022), p. 110219.
[36] *cavings*. URL: https://glossary.oilfield.slb.com/en/terms/c/cavings (visited on 05/05/2022).
[37] Christopher Skea, Alireza Rezagholilou, Pouria Behnoud Far, Raoof Gholami, and Mohammad Sarmadivleh. “An approach for wellbore failure analysis using rock cavings and image processing”. In: *Journal of Rock Mechanics and Geotechnical Engineering* 10.5 (2018), pp. 865–878.

[38] Reza Rahimi and Runar Nygaard. “Comparison of rock failure criteria in predicting borehole shear failure”. In: *International Journal of Rock Mechanics and Mining Sciences* 79 (2015), pp. 29–40.
[39] John Conrad Jaeger, Neville GW Cook, and Robert Zimmerman. *Fundamentals of rock mechanics*. John Wiley & Sons, 2009.
[40] Russell T Ewy. “Wellbore-stability predictions by use of a modified Lade criterion”. In: *SPE Drilling & Completion* 14.02 (1999), pp. 85–91.
[41] JC Jaeger. “Shear failure of anistropic rocks”. In: *Geological magazine* 97.1 (1960), pp. 65–72.
[42] Hikweon Lee, See Hong Ong, Mohammed Azeemuddin, and Harvey Goodman. “A wellbore stability model for formations with anisotropic rock strengths”. In: *Journal of Petroleum Science and Engineering* 96 (2012), pp. 109–119.
[43] Daniel Moos, Pavel Peska, Thomas Finkbeiner, and Mark Zoback. “Comprehensive wellbore stability analysis utilizing quantitative risk assessment”. In: *Journal of Petroleum Science and Engineering* 38.3-4 (2003), pp. 97–109.
[44] S Ottesen, RH Zheng, and RC McCann. “Borehole stability assessment using quantitative risk analysis”. In: *SPE/IADC drilling conference*. OnePetro. 1999.
[45] D Okland and JM Cook. “Bedding-related borehole instability in high-angle wells”. In: *SPE/ISRM rock mechanics in petroleum engineering*. OnePetro. 1998. DOI: https://doi.org/10.2118/47285-MS.
[46] Fangchao Tong, Mingming Tang, Gang Chen, Ningbo Wang, Peng Liu, Gongrui Yan, and Wei Lin. “New Modified Plane of Weakness Method Enables Drilling Horizontal Wells Successfully in Ordos Basin, China”. In: *International Petroleum Technology Conference*. OnePetro. 2019.
[47] Julie Kowan, Luke Schanken, and Robert Jacobi. “Conclusive Proof of Weak Bedding Planes in the Marcellus Shale and Proposed Mitigation Strategies”. In: *Petrophysics-The SPWLA Journal of Formation Evaluation and Reservoir Description* 62.01 (2021), pp. 31–44.
[48] Tron Golder Kristiansen, Andreas Bauer, Assia Guida, and Claudia Bonin. “A Troublesome Well Section: The Rock Mechanics Analysis”. In: *SPE Norway Subsurface Conference*. OnePetro. 2022.
[49] Chris Gallant, Jianguo Zhang, Christopher Allen Wolfe, John Freeman, Talal M Al-Bazali, and Mike Reese. “Wellbore stability considerations for drilling high-angle wells through finely laminated shale: a case study from Terra Nova”. In: *SPE annual technical conference and exhibition*. OnePetro. 2007.
[50] SM Willson, NC Last, MD Zoback, and D Moos. “Drilling in South America: a wellbore stability approach for complex geologic conditions”. In: *Latin American and Caribbean petroleum engineering conference*. OnePetro. 1999.
[51] Saeed Rafieepour and Hossein Jalalifar. “Drilling optimization based on a geomechanical analysis using probabilistic risk assessment, a case study from offshore Iran”. In: *ISRM Regional Symposium-EUROCK 2014*. OnePetro. 2014.
[52] MR Zare-Reisabadi, A Kaffash, and SR Shadizadeh. “Determination of optimal well trajectory during drilling and production based on borehole stability”. In: *International Journal of Rock Mechanics and Mining Sciences* 56 (2012), pp. 77–87.

[53] Paul Fekete, Adewale Dosunmu, Chimaroke Anyanwu, Samuel B Odagme, and Ekeinde Evelyn. “Wellbore stability management in weak bedding planes and angle of attack in well planing”. In: *SPE Nigeria Annual International Conference and Exhibition*. OnePetro. 2014.

[54] Dennis Wackerly, William Mendenhall, and Richard L Scheaffer. *Mathematical statistics with applications*. Cengage Learning, 2014.

APPENDIX 1. PYTHON CODES

```
#!/usr/bin/env python
# coding: utf-8

# # Wellbore Stability

# In[1]:


import numpy as np
import plotly.express as px
import pandas as pd
import math
import plotly.graph_objects as go


# The considered failures are tensile failure and shear failure. Shear failure is caused by failure of the shale matrix and failure in the plane of weakness. Isotropic scenario is considered.

# ### Useful Functions:

# In[2]:


# Sinus in degrees
def sind(q):
    if q == 0 or q == 360 or q == 180:
        f = 0
    else:
        f = np.sin(np.deg2rad(q))
    return f

# Cosinus in degrees
def cosd(q):
    if q == 90 or q == 270:
        f = 0
    else:
        f = np.cos(np.deg2rad(q))
    return f

# Tangent in degrees
def tand(q):
    if q == 0 or q == 360 or q == 180:
        f = 0
    else:
        f = np.tan(np.deg2rad(q))
    return f


# ## No Uncertainty

# ### Case A. Variation in inclination.
```

```
51
52 # ### Data:
53
54 # In[3]:
55
56
57 # Variables
58
59 n = 10**4 #  Samples
60 TVD = 3020 # Depth  , meters
61 overburden = 600 # P, bars
62 v = 0.25 #  Poisson  ratio
63 a = 0 #  Azimuth  , degrees
64
65 fw = 0.4 # fw , Friction of plane of weakness.
66
67 fo = 1 # fo, Friction of rock matrix.
68 fi = np.degrees(np.arctan(fo))
69 beta = 45 + 0.5* fi
70
71 Sw = 18 # Sw , Cohesion of the plane of weakness.
72
73 So = 18 # So, Cohesion of the rock matrix.
74 Co = 2 * So * tand(beta)
75
76 # min_h and max_h, Horizontal stresses. Bar.
77 min_h = 550
78 max_h = min_h
79
80 po = 490 # Po , Pore pressure, Bar.
81
82 iw = 0 # iw, Angle of the plane of weakness.
83 aw = 0 # aw, Angle of the plane of weakness.
84 fi = np.deg2rad(fi)
85 SL = So/np.tan(fi)
86 nL = 4 * (np.tan(fi))**2 * (9 - 7 * np.sin(fi))/(1 - np.sin(fi))
87
88 tr = 0 # tr, . Triangular distribution.
89
90 # Loop variables
91 P = -1
92 max_d = 360 #  Degrees  , 360 degrees in the wellbore
93 min_pw = (po - 10) / (0.098 * TVD) # Minimum  Well Pressure considered,
      pore pressure
94 max_pw = 2.2 # Maximum  Well Pressure considered
95 n = 1000 #  Iterations  between the minimum wellbore pressure and the
      maximum well pressure
96 step = (max_pw - min_pw)//n # Step for the given iterations between the
      given well pressures
97 n_d = max_d//4 #  Iterations  for the wellbore directions, there will be a
      measure every 4 degrees
98 p_col = np.zeros(n_d + 1)
99 p_frac = np.zeros(n_d + 1)
100
```

```
101
102 # In[4]:
103
104
105 # Matrix collapse
106
107 def m_collapse (TVD, min_h, max_h, overburden, v, Co, beta, po, P, a):
108     matrix_collapse = []
109
110     for i in range(0, 91):
111         p_col[n_d] = 0 # In  the last measure of the wellbore (90 if done
    every 4 degrees), the initial value is set to 0
112
113         R = np.array([[cosd(a) * cosd(i), sind(a) * cosd(i), -sind(i)]
114                   , [-sind(a), cosd(a), 0],
115                   [cosd(a) * sind(i), sind(a) * sind(i), cosd(i)]])
116
117         strs = np.array([[max_h, 0, 0], [0, min_h, 0], [0, 0, overburden
    ]]) # Stress tensor in x', y', z' directions
118
119         stress = np.linalg.multi_dot([R, strs, np.transpose(R)])
120
121         sx = stress[0, 0]
122         sy = stress[1, 1]
123         szz = stress[2, 2]
124         tau_xy = stress[1, 0]
125         tau_xz = stress[2, 0]
126         tau_yz = stress[1, 2]
127
128         for j in range(0, n + 1):
129
130             if p_col[n_d] == 0:
131
132                 # Python considers a decimal to be 0, round and 10**5 are
    used to avoid this issue
133                 pw = 10**5 * min_pw + (j) * (round(max_pw - min_pw, 2) *
    10**5//n)
134                 pw = 0.098 * TVD * pw/10**5
135
136                 for k in range(0, n_d + 1):
137                     theta = k * 4
138
139                     # Applying the Kirsch Equations
140                     sr = pw ;
141                     stan = sx + sy - 2*(sx - sy)* cosd(2*theta) - 4 *
    tau_xy *sind(2*theta) - pw ;
142                     sz = szz - v*(2*(sx - sy)* cosd(2*theta) + 4 * tau_xy
    * sind(2*theta));
143                     tau_rtan = 0;
144                     tau_tanz = 2*(-tau_xz * sind(theta) + tau_yz * cosd(
    theta));
145                     tau_rz = 0;
146
147                     str_wb = np.array([[sr, tau_rtan, tau_rz], [tau_rtan,
```

```
    stan, tau_tanz], [tau_rz, tau_tanz, sz]])

                    s1 = np.amax(np.linalg.eigvals(str_wb)) - po
                    s3 = np.amin(np.linalg.eigvals(str_wb)) - po

                    P = s1 - Co - s3*(tand(beta)**2)
                    p_col[k] = pw

                    if P > 0: #Whenever a wellbore pressure produces collapse the process stops and Pw increases
                        break

            else:
                break # When the 360 degrees of a wellbore have been analyzed and there is no collapse,

        #the inclination increases
        matrix_collapse = np.append(matrix_collapse, pw)
    return(matrix_collapse)

# Plane of weakness collapse

def w_collapse (TVD, min_h, max_h, overburden, v, Sw, fw, iw, aw, po, P, a):
    pw_collapse = []

    for i in range(0, 91):
        p_col[n_d] = 0 # In  the last measure of the wellbore (90 if done every 4 degrees), the initial value is set to 0

        R1 = np.array([[cosd(a) * cosd(i), sind(a) * cosd(i), -sind(i)]
                  , [-sind(a), cosd(a), 0],
                  [cosd(a) * sind(i), sind(a) * sind(i), cosd(i)]])

        R2 = np.array([[cosd(aw) * cosd(iw), sind(aw) * cosd(iw), -sind(iw)]
                  , [-sind(aw), cosd(aw), 0],
                  [cosd(aw) * sind(iw), sind(aw) * sind(iw), cosd(iw)]])

        strs = np.array([[max_h, 0, 0], [0, min_h, 0], [0, 0, overburden]]) # Stress tensor in x', y', z' directions

        stress = np.linalg.multi_dot([R1, strs, np.transpose(R1)])

        sx = stress[0, 0]
        sy = stress[1, 1]
        szz = stress[2, 2]
        tau_xy = stress[1, 0]
        tau_xz = stress[2, 0]
        tau_yz = stress[1, 2]

        for j in range(0, n + 1):

            if p_col[n_d] == 0:
```

```
194
195                 # Python considers a decimal to be 0, round and 10**5 are
    used to avoid this issue
196                 pw = 10**5 * min_pw + (j) * (round(max_pw - min_pw, 2) *
    10**5//n)
197                 pw = 0.098 * TVD * pw/10**5
198
199                 for k in range(0, n_d + 1):
200                     theta = k * 4
201
202                     Rz = np.array([[cosd(theta), sind(theta), 0], [-sind(
    theta), cosd(theta), 0], [0, 0, 1]])
203                     # Applying the Kirsch Equations
204                     sr = pw ;
205                     stan = sx + sy - 2*(sx - sy)* cosd(2*theta) - 4 *
    tau_xy *sind(2*theta) - pw ;
206                     sz = szz - v*(2*(sx - sy)* cosd(2*theta) + 4 * tau_xy
    * sind(2*theta));
207                     tau_rtan = 0;
208                     tau_tanz = 2*(-tau_xz * sind(theta) + tau_yz * cosd(
    theta));
209                     tau_rz = 0;
210
211                     str_wb = np.array([[sr, tau_rtan, tau_rz], [tau_rtan,
    stan, tau_tanz], [tau_rz, tau_tanz, sz]])
212                     str_wb = np.linalg.multi_dot([R2, np.transpose(R1), np
    .transpose(Rz), str_wb, Rz, R1, np.transpose(R2)])
213
214                     tau = np.sqrt(str_wb[2, 0]**2 + str_wb[2, 1]**2)
215                     szw = str_wb[2, 2] - po
216                     P = tau - Sw - fw * szw
217
218                     p_col[k] = pw
219
220                     if P > 0: # Whenever a wellbore pressure produces
    collapse the process stops and Pw increases
221                         break
222
223             else:
224                 break # When the 360 degrees of a wellbore have been
    analyzed and there is no collapse,
225
226         #the inclination increases
227          pw_collapse = np.append(pw_collapse, pw)
228     return(pw_collapse)
229
230 # Plane of weakness collapse upper
231
232 def upw_collapse (TVD, min_h, max_h, overburden, v, Sw, fw, iw, aw, po, P,
        a):
233     upw_collapse = []
234
235     for i in range(0, 91):
236         p_col[n_d] = 0 # In  the last measure of the wellbore (90 if done
```

```
    every 4 degrees), the initial value is set to 0

        R1 = np.array([[cosd(a) * cosd(i), sind(a) * cosd(i), -sind(i)]
                  , [-sind(a), cosd(a), 0],
                  [cosd(a) * sind(i), sind(a) * sind(i), cosd(i)]])

        R2 = np.array([[cosd(aw) * cosd(iw), sind(aw) * cosd(iw), -sind(iw
    )]
                  , [-sind(aw), cosd(aw), 0],
                  [cosd(aw) * sind(iw), sind(aw) * sind(iw), cosd(iw)]])

        strs = np.array([[max_h, 0, 0], [0, min_h, 0], [0, 0, overburden
    ]]) # Stress tensor in x', y', z' directions

        stress = np.linalg.multi_dot([R1, strs, np.transpose(R1)])

        sx = stress[0, 0]
        sy = stress[1, 1]
        szz = stress[2, 2]
        tau_xy = stress[1, 0]
        tau_xz = stress[2, 0]
        tau_yz = stress[1, 2]

        for j in range(0, n + 1):

            if p_col[n_d] == 0:

                # Python considers a decimal to be 0, round and 10**5 are
    used to avoid this issue
                pw = 10**5 * max_pw - (j) * (round(max_pw - min_pw, 2) *
    10**5//n)
                pw = 0.098 * TVD * pw/10**5

                for k in range(0, n_d + 1):
                    theta = k * 4

                    Rz = np.array([[cosd(theta), sind(theta), 0], [-sind(
    theta), cosd(theta), 0], [0, 0, 1]])
                    # Applying the Kirsch Equations
                    sr = pw ;
                    stan = sx + sy - 2*(sx - sy)* cosd(2*theta) - 4 *
    tau_xy *sind(2*theta) - pw ;
                    sz = szz - v*(2*(sx - sy)* cosd(2*theta) + 4 * tau_xy
    * sind(2*theta));
                    tau_rtan = 0;
                    tau_tanz = 2*(-tau_xz * sind(theta) + tau_yz * cosd(
    theta));
                    tau_rz = 0;

                    str_wb = np.array([[sr, tau_rtan, tau_rz], [tau_rtan,
    stan, tau_tanz], [tau_rz, tau_tanz, sz]])
                    str_wb = np.linalg.multi_dot([R2, np.transpose(R1), np
    .transpose(Rz), str_wb, Rz, R1, np.transpose(R2)])
```

```
                    tau = np.sqrt(str_wb[2, 0]**2 + str_wb[2, 1]**2)
                    szw = str_wb[2, 2] - po
                    P = tau - Sw - fw * szw

                    p_col[k] = pw

                    if P > 0: # Whenever a wellbore pressure produces collapse the process stops and Pw increases
                        break

            else:
                break # When the 360 degrees of a wellbore have been analyzed and there is no collapse,

        #the inclination increases
        upw_collapse = np.append(upw_collapse, pw)
    return(upw_collapse)

# Matrix collapse in high well pressures

def um_collapse (TVD, min_h, max_h, overburden, v, Co, beta, po, P, a):
    up_matrix_collapse = []

    for i in range(0, 91):
        p_col[n_d] = 0 # In  the last measure of the wellbore (90 if done every 4 degrees), the initial value is set to 0

        R = np.array([[cosd(a) * cosd(i), sind(a) * cosd(i), -sind(i)]
                  , [-sind(a), cosd(a), 0],
                  [cosd(a) * sind(i), sind(a) * sind(i), cosd(i)]])

        strs = np.array([[max_h, 0, 0], [0, min_h, 0], [0, 0, overburden]]) # Stress tensor in x', y', z' directions

        stress = np.linalg.multi_dot([R, strs, np.transpose(R)])

        sx = stress[0, 0]
        sy = stress[1, 1]
        szz = stress[2, 2]
        tau_xy = stress[1, 0]
        tau_xz = stress[2, 0]
        tau_yz = stress[1, 2]

        for j in range(0, n + 1):

            if p_col[n_d] == 0:

                pw = 10**5 * max_pw - (j) * (round(max_pw - min_pw, 2) * 10**5//n)
                pw = 0.098 * TVD * pw/10**5

                for k in range(0, n_d + 1):
                    theta = k * 4

```

```
                    # Applying the Kirsch Equations
                    sr = pw ;
                    stan = sx + sy - 2*(sx - sy)* cosd(2*theta) - 4 *
    tau_xy *sind(2*theta) - pw ;
                    sz = szz - v*(2*(sx - sy)* cosd(2*theta) + 4 * tau_xy
    * sind(2*theta));
                    tau_rtan = 0;
                    tau_tanz = 2*(-tau_xz * sind(theta) + tau_yz * cosd(
    theta));
                    tau_rz = 0;

                    str_wb = np.array([[sr, tau_rtan, tau_rz], [tau_rtan,
    stan, tau_tanz], [tau_rz, tau_tanz, sz]])

                    s1 = np.amax(np.linalg.eigvals(str_wb)) - po
                    s3 = np.amin(np.linalg.eigvals(str_wb)) - po
                    P = s1 - Co - s3*(tand(beta)**2)

                    p_col[k] = pw

                    if P > 0: #Whenever a wellbore pressure produces
    collapse the process stops and Pw increases
                        break

            else:
                break # When the 360 degrees of a wellbore have been
    analyzed and there is no collapse,

        #the inclination increases
         up_matrix_collapse = np.append(up_matrix_collapse, pw)
     return(up_matrix_collapse)

# Tensile failure (fracture)

def m_tensile (TVD, min_h, max_h, overburden, v, po, tr, a):
    matrix_ten = []

    for i in range(0, 91):
        p_frac[n_d] = 0 # In  the last measure of the wellbore (90 if done
     every 4 degrees), the initial value is set to 0

        R = np.array([[cosd(a) * cosd(i), sind(a) * cosd(i), -sind(i)]
                  , [-sind(a), cosd(a), 0],
                  [cosd(a) * sind(i), sind(a) * sind(i), cosd(i)]])

        strs = np.array([[max_h, 0, 0], [0, min_h, 0], [0, 0, overburden
    ]]) # Stress tensor in x', y', z' directions

        stress = np.linalg.multi_dot([R, strs, np.transpose(R)])

        sx = stress[0, 0]
        sy = stress[1, 1]
        szz = stress[2, 2]
        tau_xy = stress[1, 0]
```

```
        tau_xz = stress[2, 0]
        tau_yz = stress[1, 2]

        for j in range(0, n + 1):

            if p_frac[n_d] == 0:

                # Python considers a decimal to be 0, round and 10**5 are used to avoid this issue
                pw = 10**5 * max_pw - (j) * (round(max_pw - min_pw, 2) * 10**5//n)
                pw = 0.098 * TVD * pw/10**5

                for k in range(0, n_d + 1):
                    theta = k * 4

                    # Applying the Kirsch Equations
                    sr = pw ;
                    stan = sx + sy - 2*(sx - sy)* cosd(2*theta) - 4 * tau_xy *sind(2*theta) - pw ;
                    sz = szz - v*(2*(sx - sy)* cosd(2*theta) + 4 * tau_xy * sind(2*theta));
                    tau_rtan = 0;
                    tau_tanz = 2*(-tau_xz * sind(theta) + tau_yz * cosd(theta));
                    tau_rz = 0;

                    str_wb = np.array([[sr, tau_rtan, tau_rz], [tau_rtan, stan, tau_tanz], [tau_rz, tau_tanz, sz]])

                    s3 = np.amin(np.linalg.eigvals(str_wb)) - po

                    p_frac[k] = pw

                    if s3 < -tr : #Whenever a wellbore pressure produces fracture the process stops and Pw increases
                        break

            else:
                break # When the 360 degrees of a wellbore have been analyzed and there is no collapse,

        #the inclination increases
         matrix_ten = np.append(matrix_ten, pw)
     return(matrix_ten)


# In[5]:


# Modified Lade
def lade_collapse_d (TVD, min_h, max_h, overburden, v, Co, beta, po, P, a):
     matrix_collapse = []
```

```
 for i in range(0, 91):
     p_col[n_d] = 0 # In  the last measure of the wellbore (90 if done
every 4 degrees), the initial value is set to 0

     R = np.array([[cosd(a) * cosd(i), sind(a) * cosd(i), -sind(i)]
               , [-sind(a), cosd(a), 0],
               [cosd(a) * sind(i), sind(a) * sind(i), cosd(i)]])

     strs = np.array([[max_h, 0, 0], [0, min_h, 0], [0, 0, overburden
]]) # Stress tensor in x', y', z' directions

     stress = np.linalg.multi_dot([R, strs, np.transpose(R)])

     sx = stress[0, 0]
     sy = stress[1, 1]
     szz = stress[2, 2]
     tau_xy = stress[1, 0]
     tau_xz = stress[2, 0]
     tau_yz = stress[1, 2]

     for j in range(0, n + 1):

         if p_col[n_d] == 0:

             # Python considers a decimal to be 0, round and 10**5 are
used to avoid this issue
             pw = 10**5 * min_pw + (j) * (round(max_pw - min_pw, 2) *
10**5//n)
             pw = 0.098 * TVD * pw/10**5

             for k in range(0, n_d + 1):
                 theta = k * 4

                 # Applying the Kirsch Equations
                 sr = pw ;
                 stan = sx + sy - 2*(sx - sy)* cosd(2*theta) - 4 *
tau_xy *sind(2*theta) - pw ;
                 sz = szz - v*(2*(sx - sy)* cosd(2*theta) + 4 * tau_xy
* sind(2*theta));
                 tau_rtan = 0;
                 tau_tanz = 2*(-tau_xz * sind(theta) + tau_yz * cosd(
theta));
                 tau_rz = 0;

                 str_wb = np.array([[sr, tau_rtan, tau_rz], [tau_rtan,
stan, tau_tanz], [tau_rz, tau_tanz, sz]])
                 eig = np.sort(np.linalg.eigvals(str_wb))
                 s1, s2, s3  = eig[2] - po, eig[1] - po, eig[0] - po

                 I1, I3 = (s1 + SL) + (s2 + SL) + (s3 + SL), (s1 + SL)
* (s2 + SL) * (s3 + SL)

                 p_col[k] = pw
```

```
461
462                     if I1**3/I3 - 27 > nL: #Whenever a wellbore pressure
    produces collapse the process stops and Pw increases
463                         break
464
465             else:
466                 break # When the 360 degrees of a wellbore have been
    analyzed and there is no collapse,
467
468         #the inclination increases
469          matrix_collapse = np.append(matrix_collapse, pw)
470      return(matrix_collapse)
471
472 # Modified upper
473 def lade_collapse_u (TVD, min_h, max_h, overburden, v, Co, beta, po, P, a)
    :
474      matrix_collapse = []
475
476      for i in range(0, 91):
477          p_col[n_d] = 0 # In  the last measure of the wellbore (90 if done
    every 4 degrees), the initial value is set to 0
478
479          R = np.array([[cosd(a) * cosd(i), sind(a) * cosd(i), -sind(i)]
480                    , [-sind(a), cosd(a), 0],
481                    [cosd(a) * sind(i), sind(a) * sind(i), cosd(i)]])
482
483          strs = np.array([[max_h, 0, 0], [0, min_h, 0], [0, 0, overburden
    ]]) # Stress tensor in x', y', z' directions
484
485          stress = np.linalg.multi_dot([R, strs, np.transpose(R)])
486
487          sx = stress[0, 0]
488          sy = stress[1, 1]
489          szz = stress[2, 2]
490          tau_xy = stress[1, 0]
491          tau_xz = stress[2, 0]
492          tau_yz = stress[1, 2]
493
494          for j in range(0, n + 1):
495
496              if p_col[n_d] == 0:
497
498                  # Python considers a decimal to be 0, round and 10**5 are
    used to avoid this issue
499                  pw = 10**5 * max_pw - (j) * (round(max_pw - min_pw, 2) *
    10**5//n)
500                  pw = 0.098 * TVD * pw/10**5
501
502                  for k in range(0, n_d + 1):
503                      theta = k * 4
504
505                      # Applying the Kirsch Equations
506                      sr = pw ;
                         stan = sx + sy - 2*(sx - sy)* cosd(2*theta) - 4 *
```

```
    tau_xy *sind(2*theta) - pw ;
                    sz = szz - v*(2*(sx - sy)* cosd(2*theta) + 4 * tau_xy
    * sind(2*theta));
                    tau_rtan = 0;
                    tau_tanz = 2*(-tau_xz * sind(theta) + tau_yz * cosd(
    theta));
                    tau_rz = 0;

                    str_wb = np.array([[sr, tau_rtan, tau_rz], [tau_rtan,
    stan, tau_tanz], [tau_rz, tau_tanz, sz]])
                    eig = np.sort(np.linalg.eigvals(str_wb))
                    s1, s2, s3  = eig[2] - po, eig[1] - po, eig[0] - po

                    I1, I3 = (s1 + SL) + (s2 + SL) + (s3 + SL), (s1 + SL)
    * (s2 + SL) * (s3 + SL)

                    p_col[k] = pw

                    if I1**3/I3 - 27 > nL: #Whenever a wellbore pressure
    produces collapse the process stops and Pw increases
                        break

            else:
                break # When the 360 degrees of a wellbore have been
    analyzed and there is no collapse,

        #the inclination increases
         matrix_collapse = np.append(matrix_collapse, pw)
     return(matrix_collapse)


# In[6]:


# All pressures are given in bars
matrix_collapse = m_collapse (TVD, min_h, max_h, overburden, v, Co, beta,
    po, P, a)
pw_collapse = w_collapse (TVD, min_h, max_h, overburden, v, Sw, fw, iw, aw
    , po, P, a)
upw_collapse = upw_collapse (TVD, min_h, max_h, overburden, v, Sw, fw, iw,
     aw, po, P, a)
upm_collapse = um_collapse (TVD, min_h, max_h, overburden, v, Co, beta, po
    , P, a)
matrix_tensile = m_tensile (TVD, min_h, max_h, overburden, v, po, tr, a)
lade_collapse = lade_collapse_d (TVD, min_h, max_h, overburden, v, Co,
    beta, po, P, a)
lade_collapse_u = lade_collapse_u (TVD, min_h, max_h, overburden, v, Co,
    beta, po, P, a)

inc = np.arange(0, 91, 1)


# In[7]:

```

```
547
548 # Pressures in sg
549 # matrix_collapse = matrix_collapse/(0.098 * TVD)
550 # pw_collapse = pw_collapse/(0.098 * TVD)
551 # upm_collapse = upm_collapse/(0.098 * TVD)
552 # matrix_tensile = matrix_tensile/(0.098 * TVD)
553 # pore = np.ones(91) * (po/(0.098 * TVD))
554
555 # Pressures in ppg
556 matrix_collapse = matrix_collapse * 8.33/(0.098 * TVD)
557 pw_collapse = pw_collapse * 8.33/(0.098 * TVD)
558 upw_collapse = upw_collapse * 8.33/(0.098 * TVD)
559 upm_collapse = upm_collapse * 8.33/(0.098 * TVD)
560 matrix_tensile = matrix_tensile * 8.33/(0.098 * TVD)
561 pore = np.ones(91) * (po * 8.33/(0.098 * TVD))
562 mud = np.ones(91) * 14.6
563 lade_collapse = lade_collapse * 8.33/(0.098 * TVD)
564 lade_collapse_u = lade_collapse_u * 8.33/(0.098 * TVD)
565
566
567 # In[8]:
568
569
570 mw_window = go.Figure()
571 mw_window.add_trace(go.Scatter(y = matrix_collapse, x = inc, line=dict(
    color='royalblue'), name='Shear failure'))
572 mw_window.add_trace(go.Scatter(y = pw_collapse, x = inc, line=dict(color='
    red'), name='Bedding plane failure'))
573 mw_window.add_trace(go.Scatter(y = upw_collapse, x = inc, line=dict(color=
    'red'), name='Bedding plane failure'))
574 mw_window.add_trace(go.Scatter(y = upm_collapse, x = inc, line=dict(color=
    'royalblue'), name='Shear failure'))
575 mw_window.add_trace(go.Scatter(y = matrix_tensile, x = inc, line=dict(
    color='purple'), name='Tensile failure'))
576 mw_window.add_trace(go.Scatter(y = mud, x = inc, line=dict(color='black',
    width=2, dash='dot'), name='Mud Weight (ppg)'))
577 mw_window.add_trace(go.Scatter(y = pore, x = inc, line=dict(color='olive',
     width=3, dash='dash'), name='Pore pressure'))
578 mw_window.update_layout(title={'text':'Mud Window by inclination','x'
    :0.45,'y':0.85}, xaxis_title='Inclination', yaxis_title='Pressure (ppg)
    ')
579 mw_window.add_trace(go.Scatter(y = lade_collapse, x = inc, line=dict(color
    ='blue'), name='Lade failure'))
580 mw_window.add_trace(go.Scatter(y = lade_collapse_u, x = inc, line=dict(
    color='blue'), name='Lade failure'))
581
582
583 # In[9]:
584
585
586 np.savetxt('failure.txt',
587            np.c_[inc, matrix_collapse, pw_collapse,upw_collapse,
    upm_collapse, matrix_tensile,\
588                  pore, mud, lade_collapse, lade_collapse_u], header = 'inc
```

```
    , Shear failure, Bedding plane failure,\
589                 Bedding plane failure, Shear failure, Tensile failure,
    Pore pressure, Mud pressure, lade_d, lade_u')
590
591
592 # In[2]:
593
594
595
596
597
598 # In[ ]:
```

```
1  #!/usr/bin/env python
2  # coding: utf-8
3
4  # # Plane of Weakness Sensitivity Analysis
5
6  # The effect of the friction coefficient and cohesion of the plane of
       weakness will be studied in the failure of the plane of weakness.
       Reductions respect the matrix values will be performed.
7
8  # In[1]:
9
10
11 import numpy as np
12 import plotly.express as px
13 import pandas as pd
14 import math
15 import plotly.graph_objects as go
16
17
18 # ### Data:
19
20 # In[2]:
21
22
23 stress =  pd.read_excel(r'z36_data.xlsx', sheet_name = '1') #Import of
       Excel Sheet
24 mud =  pd.read_excel(r'z36_data.xlsx', sheet_name = '2') #Import of Excel
       Sheet
25 TVD = np.asarray(stress.loc[:,"TVD_KB"]/3.28084) # TVD in m
26 TVD_ft = np.asarray(stress.loc[:,"TVD_KB"]) # TVD in ft
27 TVD_mud = np.asarray(mud.loc[:,"TVD"]) # TVD in m for the 2nd sheet
28 ppg = np.asarray(mud.loc[:,"Density"]) # Mud density in ppg for the well
29 sx = np.asarray(stress.loc[:,"BSx"]) # Pa
30 sy = np.asarray(stress.loc[:,"BSy"]) # Pa
31 szz = np.asarray(stress.loc[:,"BSv"]) # Pa
32 tau_xy = np.asarray(stress.loc[:,"BTxy"]) # Pa
33 tau_xz = np.asarray(stress.loc[:,"BTzx"]) # Pa
34 tau_yz = np.asarray(stress.loc[:,"BTyz"]) # Pa
35 po = np.asarray(stress.loc[:,"Po"]) # sg
36 a = np.asarray(stress.loc[:,"Az"]) # Azimuth
37 inc = np.asarray(stress.loc[:,"Incl"]) # Inclination
```

```
S3 = np.asarray(stress.loc[:,"S3"]/(10**5)) # S3  in bar
v = np.asarray(stress.loc[:,"Poisson"]) # Poisson ratio
fi = np.deg2rad(np.asarray(stress.loc[:,"FrictionAngle"])) # Friction angle of rock matrix


# In[3]:


# Constants
iw = 0 # iw, Angle of the plane of weakness.
aw = 0 # aw, Angle of the plane of weakness.

# Loop Constants
P = -1

# Estimation for min_pw and max_pw
min_pw = po  # Minimum  Well Pressure considered, bar
max_pw = 2.3 * 0.098 * TVD # Maximum  Well Pressure considered, bar
n = 500 #  Iterations  between the minimum wellbore pressure and the maximum well pressure
n_d = 360//4 #  Iterations  for the wellbore directions, there will be a measure every 4 degrees
p_col = np.zeros(n_d + 1)
p_frac = np.zeros(n_d + 1)


# ### Mohr-Coulomb Plane of Weakness failure

# In[4]:


# Plane of weakness

def w_collapse_d (TVD):
    pw_collapse, max_s, min_s = np.zeros(TVD.size), np.zeros(TVD.size), np.zeros(TVD.size)

    for i in range(0, TVD.size):
        p_col[n_d] = 0 # In  the last measure of the wellbore (90 if done every 4 degrees), the initial value is set to 0

        A, Inc = a[i], inc[i]
        Min_pw, Max_pw, Sx, Sy, Szz, Tau_xy, Tau_xz, Tau_yz, Po, V = min_pw[i] , max_pw[i], sx[i], sy[i],
                                    szz[i], tau_xy[i], tau_xz[i], tau_yz[i], po[i],
            v[i]
        Sw, fw = sw[i], uw[i]
        R1 = np.array([[np.cos(A) * np.cos(Inc), np.sin(A) * np.cos(Inc), -np.sin(Inc)]
                , [-np.sin(A), np.cos(A), 0],
                [np.cos(A) * np.sin(Inc), np.sin(A) * np.sin(Inc), np.cos(Inc)]])
```

```
81
82      R2 = np.array([[np.cos(aw) * np.cos(iw), np.sin(aw) * np.cos(iw),
    -np.sin(iw)]
83                 , [-np.sin(aw), np.cos(aw), 0],
84                 [np.cos(aw) * np.sin(iw), np.sin(aw) * np.sin(iw), np.
    cos(iw)]])
85
86      for j in range(0, n + 1):
87
88          if p_col[n_d] == 0:
89
90              pw = Min_pw + (j) * (Max_pw - Min_pw)/n
91              for k in range(0, n_d + 1):
92                  theta = np.deg2rad(k * 4)
93
94                  Rz = np.array([[np.cos(theta), np.sin(theta), 0], [-np
    .sin(theta), np.cos(theta), 0], [0, 0, 1]])
95                  # Applying the Kirsch Equations
96                  sr = pw
97                  stan = Sx + Sy - 2*(Sx - Sy)* np.cos(2*theta) - 4 *
    Tau_xy * np.sin(2*theta) - pw
98                  sz = Szz - V*(2*(Sx - Sy)* np.cos(2*theta) + 4 *
    Tau_xy * np.sin(2*theta))
99                  tau_rtan = 0
100                 tau_tanz = 2*(-Tau_xz * np.sin(theta) + Tau_yz * np.
    cos(theta))
101                 tau_rz = 0
102
103                 str_wb = np.array([[sr, tau_rtan, tau_rz], [tau_rtan,
    stan, tau_tanz], [tau_rz, tau_tanz, sz]])
104                 s1, s3  = np.amax(np.linalg.eigvals(str_wb)) - Po ,
    np.amin(np.linalg.eigvals(str_wb)) - Po
105                 str_wb = np.linalg.multi_dot([R2, np.transpose(R1), np
    .transpose(Rz), str_wb, Rz, R1, np.transpose(R2)])
106
107                 tau = np.sqrt(str_wb[2, 0]**2 + str_wb[2, 1]**2)
108                 szw = str_wb[2, 2] - Po
109                 P = tau - Sw - fw * szw
110
111                 p_col[k] = pw
112
113                 if P >= 0: #Whenever a wellbore pressure produces
    collapse the process stops and Pw increases
114                     break
115
116         else:
117             break # When the 360 degrees of a wellbore have been
    analyzed and there is no collapse,
118
119     #the inclination increases
120      pw_collapse[i] = pw
121
122  return(pw_collapse)
```

```
# In[5]:


# CASE 1, Same Cohesion, 0.7 friction coefficient
sw = np.asarray(stress.loc[:,"So"]/(10**5)) # Cohesion plane of weakness
uw = np.tan(fi) * 0.7 # fw , Friction coefficient of plane of weakness.
pw_collapse_1 = w_collapse_d (TVD) * 8.33/(0.098 * TVD)

# CASE 2, Same Cohesion, 0.5 friction coefficient
uw = np.tan(fi) * 0.5 # fw , Friction coefficient of plane of weakness.
pw_collapse_2 = w_collapse_d (TVD) * 8.33/(0.098 * TVD)

# CASE 3, Same Cohesion, 0.4 friction coefficient
uw = np.tan(fi) * 0.4 # fw , Friction coefficient of plane of weakness.
pw_collapse_3 = w_collapse_d (TVD) * 8.33/(0.098 * TVD)

# CASE 4, 0.9 Cohesion, same friction coefficient
sw = np.asarray(stress.loc[:,"So"]/(10**5)) * 0.9 # Cohesion plane of
    weakness
uw = np.tan(fi)  # fw , Friction coefficient of plane of weakness.
pw_collapse_4 = w_collapse_d (TVD) * 8.33/(0.098 * TVD)

# CASE 5, 0.7 Cohesion, same friction coefficient
sw = np.asarray(stress.loc[:,"So"]/(10**5)) * 0.7 # Cohesion plane of
    weakness
pw_collapse_5 = w_collapse_d (TVD) * 8.33/(0.098 * TVD)

# CASE 6, 0.5 Cohesion, same friction coefficient
sw = np.asarray(stress.loc[:,"So"]/(10**5)) * 0.5 # Cohesion plane of
    weakness
pw_collapse_6 = w_collapse_d (TVD) * 8.33/(0.098 * TVD)

# CASE 7, 0.9 Cohesion, 0.5 friction coefficient
sw = np.asarray(stress.loc[:,"So"]/(10**5)) * 0.9 # Cohesion plane of
    weakness
uw = np.tan(fi) * 0.5  # fw , Friction coefficient of plane of weakness.
pw_collapse_7 = w_collapse_d (TVD) * 8.33/(0.098 * TVD)


# In[6]:


# In imperial system
po_ppg = po * 8.33/(0.098 * TVD)
TVD_mud = TVD_mud/0.3048
S3 = S3 * 8.33/(0.098 * TVD)


# In[16]:


np.savetxt('case_1.txt', np.c_[TVD_ft, pw_collapse_1], header = 'TVD, pw')
np.savetxt('case_2.txt', np.c_[TVD_ft, pw_collapse_2], header = 'TVD, pw')
```

```
np.savetxt('case_3.txt', np.c_[TVD_ft, pw_collapse_3], header = 'TVD, pw')
np.savetxt('case_4.txt', np.c_[TVD_ft, pw_collapse_4], header = 'TVD, pw')
np.savetxt('case_5.txt', np.c_[TVD_ft, pw_collapse_5], header = 'TVD, pw')
np.savetxt('case_6.txt', np.c_[TVD_ft, pw_collapse_6], header = 'TVD, pw')
np.savetxt('case_7.txt', np.c_[TVD_ft, pw_collapse_7], header = 'TVD, pw')
np.savetxt('mud_weight.txt', np.c_[TVD_mud, ppg], header = 'TVD, mw')
np.savetxt('po_s3.txt', np.c_[TVD_ft, po_ppg, S3], header = 'TVD, po, S3')


# In[15]:


np.savetxt('po_s3.txt', np.c_[TVD_ft, po_ppg, S3], header = 'TVD, po, S3')


# In[ ]:
```

```
#!/usr/bin/env python
# coding: utf-8

# # Wellbore Stability - Deterministic ppg vs ft

# In[4]:


import numpy as np
import pandas as pd
import math
import plotly.graph_objects as go


# The considered failures are tensile failure and shear failure. Shear
    failure is caused by failure of the shale matrix and failure in the
    plane of weakness.
# For mud window, consider data from 4000 ft TVD on.

# ### Data:

# In[5]:


stress =  pd.read_excel(r'z36_data.xlsx', sheet_name = '1') #Import of
    Excel Sheet
mud =  pd.read_excel(r'z36_data.xlsx', sheet_name = '2') #Import of Excel
    Sheet
TVD = np.asarray(stress.loc[:,"TVD_KB"]/3.28084) # TVD in m
TVD_ft = np.asarray(stress.loc[:,"TVD_KB"]) # TVD in ft
TVD_mud = np.asarray(mud.loc[:,"TVD"]) # TVD in m for the 2nd sheet
ppg = np.asarray(mud.loc[:,"Density"]) # Mud density in ppg for the well
sx = np.asarray(stress.loc[:,"BSx"]) # Pa
sy = np.asarray(stress.loc[:,"BSy"]) # Pa
szz = np.asarray(stress.loc[:,"BSv"]) # Pa
tau_xy = np.asarray(stress.loc[:,"BTxy"]) # Pa
tau_xz = np.asarray(stress.loc[:,"BTzx"]) # Pa
```

```
tau_yz = np.asarray(stress.loc[:,"BTyz"]) # Pa
po = np.asarray(stress.loc[:,"Po"]) # sg
a = np.asarray(stress.loc[:,"Az"]) # Azimuth
inc = np.asarray(stress.loc[:,"Incl"]) # Inclination
S3 = np.asarray(stress.loc[:,"S3"]/(10**5)) # S3  in bar
v = np.asarray(stress.loc[:,"Poisson"]) # Poisson ratio
fi = np.deg2rad(np.asarray(stress.loc[:,"FrictionAngle"])) # Friction
    angle of rock matrix
beta = fi * 0.5 + np.pi/4 # beta in radians
Co = 2 * np.asarray(stress.loc[:,"So"]/(10**5)) * np.tan(beta) # Cohesion
    in bar
tr = np.asarray(stress.loc[:,"TensileS"]/(10**5)) # Tensile Strength in
    bar
sw = np.asarray(stress.loc[:,"So"]/(10**5)) * 0.9 # Cohesion plane of
    weakness
sl = np.asarray(stress.loc[:,"So"]/(10**5))/np.tan(fi)
nl = 4 * (np.tan(fi))**2 * (9 - 7 * np.sin(fi))/(1 - np.sin(fi))
uw = np.tan(fi) * 0.5 # fw , Friction coefficient of plane of weakness.

# To save some variables as txt file
#np.savetxt('strength.txt', np.c_[TVD_ft, tr_f* 8.33/(0.098 * TVD), so_f
    /(10**5))* 8.33/(0.098 * TVD)], header = 'TVD, tens_strength, Cohesion
    ')
#np.savetxt('stresses_b.txt', np.c_[TVD_ft, sx* 8.33/(0.098 * TVD), sy*
    8.33/(0.098 * TVD), szz* 8.33/(0.098 * TVD), \
                                #tau_xy* 8.33/(0.098 * TVD), tau_xz*
    8.33/(0.098 * TVD), tau_yz* 8.33/(0.098 * TVD)])


# In[ ]:


# Constants
iw = 0 # iw, Angle of the plane of weakness.
aw = 0 # aw, Angle of the plane of weakness.

# Estimation for min_pw and max_pw
min_pw = po  # Minimum  Well Pressure considered, bar
max_pw = 2.3 * 0.098 * TVD # Maximum  Well Pressure considered, bar
n = 500 #  Iterations  between the minimum wellbore pressure and the
    maximum well pressure
n_d = 360//4 #  Iterations  for the wellbore directions, there will be a
    measure every 4 degrees
p_col = np.zeros(n_d + 1)
p_frac = np.zeros(n_d + 1)


# ### Failure Functions:

# ### Mohr-Coulomb Matrix Collapse lower bound

# In[ ]:


```

```
def m_collapse_d (TVD):
    matrix_collapse = np.zeros(TVD.size)
    for i in range(0, TVD.size):
        p_col[n_d] = 0 # In  the last measure of the wellbore (90 if done every 4 degrees), the initial value is set to 0

        Min_pw, Max_pw, Sx, Sy, Szz, Tau_xy, Tau_xz, Tau_yz, Po, co, Beta, V = min_pw[i] , max_pw[i], sx[i], sy[i],
                                                szz[i], tau_xy[i], tau_xz[i], tau_yz[i], po[i],
                Co[i], beta[i], v[i]

        for j in range(0, n + 1):

            if p_col[n_d] == 0:

                pw = Min_pw + (j) * (Max_pw - Min_pw)/n

                for k in range(0, n_d + 1):
                    theta = np.deg2rad(k * 4)
                    # Applying the Kirsch Equations
                    sr = pw
                    stan = Sx + Sy - 2*(Sx - Sy)* np.cos(2*theta) - 4 * Tau_xy * np.sin(2*theta) - pw
                    sz = Szz - V*(2*(Sx - Sy)* np.cos(2*theta) + 4 * Tau_xy * np.sin(2*theta))
                    tau_rtan = 0
                    tau_tanz = 2*(-Tau_xz * np.sin(theta) + Tau_yz * np.cos(theta))
                    tau_rz = 0

                    str_wb = np.array([[sr, tau_rtan, tau_rz], [tau_rtan, stan, tau_tanz], [tau_rz, tau_tanz, sz]])
                    s1 = np.amax(np.linalg.eigvals(str_wb)) - Po
                    s3 = np.amin(np.linalg.eigvals(str_wb)) - Po

                    p_col[k] = pw

                    if s1 > co + s3 * np.tan(Beta)**2: # Whenever a wellbore pressure produces collapse the process stops and Pw increases
                        break

            else:
                break # When the 360 degrees of a wellbore have been analyzed and there is no collapse,

        #the inclination increases
        matrix_collapse[i] = pw
    return(matrix_collapse)


# ### Mohr-Coulomb Plane of Weakness failure

# In[ ]:
```

```
120
121
122 # Plane of weakness
123
124 def w_collapse_d (TVD):
125     pw_collapse = np.zeros(TVD.size)
126
127     for i in range(0, TVD.size):
128         p_col[n_d] = 0 # In  the last measure of the wellbore (90 if done
    every 4 degrees), the initial value is set to 0
129
130         A, Inc = a[i], inc[i]
131         Min_pw, Max_pw, Sx, Sy, Szz, Tau_xy, Tau_xz, Tau_yz, Po, V =
    min_pw[i] , max_pw[i], sx[i], sy[i],
                                        szz[i], tau_xy[i], tau_xz[i], tau_yz[
    i], po[i],
             v[i]
132         Sw, fw = sw[i], uw[i]
133
134         R1 = np.array([[np.cos(A) * np.cos(Inc), np.sin(A) * np.cos(Inc),
    -np.sin(Inc)]
135                        , [-np.sin(A), np.cos(A), 0],
136                        [np.cos(A) * np.sin(Inc), np.sin(A) * np.sin(Inc), np.
    cos(Inc)]])
137
138         R2 = np.array([[np.cos(aw) * np.cos(iw), np.sin(aw) * np.cos(iw),
    -np.sin(iw)]
139                        , [-np.sin(aw), np.cos(aw), 0],
140                        [np.cos(aw) * np.sin(iw), np.sin(aw) * np.sin(iw), np.
    cos(iw)]])
141
142         for j in range(0, n + 1):
143
144             if p_col[n_d] == 0:
145
146                 pw = Min_pw + (j) * (Max_pw - Min_pw)/n
147                 for k in range(0, n_d + 1):
148                     theta = np.deg2rad(k * 4)
149
150                     Rz = np.array([[np.cos(theta), np.sin(theta), 0], [-np
    .sin(theta), np.cos(theta), 0], [0, 0, 1]])
151                     # Applying the Kirsch Equations
152                     sr = pw
153                     stan = Sx + Sy - 2*(Sx - Sy)* np.cos(2*theta) - 4 *
    Tau_xy * np.sin(2*theta) - pw
154                     sz = Szz - V*(2*(Sx - Sy)* np.cos(2*theta) + 4 *
    Tau_xy * np.sin(2*theta))
155                     tau_rtan = 0
156                     tau_tanz = 2*(-Tau_xz * np.sin(theta) + Tau_yz * np.
    cos(theta))
157                     tau_rz = 0
158
159                     str_wb = np.array([[sr, tau_rtan, tau_rz], [tau_rtan,
    stan, tau_tanz], [tau_rz, tau_tanz, sz]])
```

```
                    str_wb = np.linalg.multi_dot([R2, np.transpose(R1), np
    .transpose(Rz), str_wb, Rz, R1, np.transpose(R2)])

                    tau = np.sqrt(str_wb[2, 0]**2 + str_wb[2, 1]**2)
                    szw = str_wb[2, 2] - Po

                    p_col[k] = pw

                    if tau > Sw + fw * szw: #Whenever a wellbore pressure
    produces collapse the process stops and Pw increases
                        break

            else:
                break # When the 360 degrees of a wellbore have been
    analyzed and there is no collapse,

        #the inclination increases
         pw_collapse[i] = pw
     return(pw_collapse)



# ### Navier Coulomb matrix collapse at high pressures

# In[ ]:


def um_collapse (TVD):
    up_matrix_collapse = np.zeros(TVD.size)

    for i in range(0, TVD.size):
        p_col[n_d] = 0 # In  the last measure of the wellbore (90 if done
    every 4 degrees), the initial value is set to 0

        Min_pw, Max_pw, Sx, Sy, Szz, Tau_xy, Tau_xz, Tau_yz, Po, co, Beta,
     V = min_pw[i] , max_pw[i], sx[i], sy[i],
                                            szz[i], tau_xy[i], tau_xz[i],
    tau_yz[i], po[i],
                  Co[i], beta[i], v[i]

        for j in range(0, n + 1):

            if p_col[n_d] == 0:

                # Python considers a decimal to be 0, round and 10**5 are
    used to avoid this issue
                pw = Max_pw - (j) * (Max_pw - Min_pw)/n

                for k in range(0, n_d + 1):
                    theta = np.deg2rad(k * 4)

                    # Applying the Kirsch Equations
                    sr = pw
                    stan = Sx + Sy - 2*(Sx - Sy)* np.cos(2*theta) - 4 *
```

```
    Tau_xy * np.sin(2*theta) - pw
                    sz = Szz - V*(2*(Sx - Sy)* np.cos(2*theta) + 4 *
    Tau_xy * np.sin(2*theta))
                    tau_rtan = 0
                    tau_tanz = 2*(-Tau_xz * np.sin(theta) + Tau_yz * np.
    cos(theta))
                    tau_rz = 0

                    str_wb = np.array([[sr, tau_rtan, tau_rz], [tau_rtan,
    stan, tau_tanz], [tau_rz, tau_tanz, sz]])
                    s1 = np.amax(np.linalg.eigvals(str_wb)) - Po
                    s3 = np.amin(np.linalg.eigvals(str_wb)) - Po

                    p_col[k] = pw

                    if s1 > co + s3 * np.tan(Beta)**2: #Whenever a
    wellbore pressure produces collapse the process stops and Pw increases
                        break

            else:
                break # When the 360 degrees of a wellbore have been
    analyzed and there is no collapse,

        #the inclination increases
         up_matrix_collapse[i] = pw
     return(up_matrix_collapse)


# ### Tensile failure criterion

# In[ ]:


# Tensile failure (fracture)

def m_tensile (TVD):
    matrix_ten = np.zeros(TVD.size)

    for i in range(0, TVD.size):
        p_frac[n_d] = 0

        Min_pw, Max_pw, Sx, Sy, Szz, Tau_xy, Tau_xz, Tau_yz, Po, Tr, V =
    min_pw[i] , max_pw[i], sx[i], sy[i], szz[i],
                                        tau_xy[i], tau_xz[i], tau_yz[i],
    po[i], tr[i], v[i]

        for j in range(0, n + 1):

            if p_frac[n_d] == 0:
                pw = Max_pw - (j) * (Max_pw - Min_pw)/n

                for k in range(0, n_d + 1):
                    theta = np.deg2rad(k * 4)

```

```
                    # Applying the Kirsch Equations
                    sr = pw
                    stan = Sx + Sy - 2*(Sx - Sy)* np.cos(2*theta) - 4 *
    Tau_xy * np.sin(2*theta) - pw
                    sz = Szz - V* (2*(Sx - Sy)* np.cos(2*theta) + 4 *
    Tau_xy * np.sin(2*theta))
                    tau_rtan = 0
                    tau_tanz = 2*(-Tau_xz * np.sin(theta) + Tau_yz * np.
    cos(theta))
                    tau_rz = 0

                    str_wb = np.array([[sr, tau_rtan, tau_rz], [tau_rtan,
    stan, tau_tanz], [tau_rz, tau_tanz, sz]])

                    s3 = np.amin(np.linalg.eigvals(str_wb)) - Po

                    p_frac[k] = pw

                    if s3 < -Tr : #Whenever a wellbore pressure produces
    fracture the process stops and Pw increases
                        break

            else:
                break # When the 360 degrees of a wellbore have been
    analyzed and there is no collapse,

        #the inclination increases
         matrix_ten[i] = pw
     return(matrix_ten)


# ### Modified Lade Criterion for matrix collapse lower bound

# In[ ]:


def lade_collapse_d (TVD):
    matrix_collapse = np.zeros(TVD.size)
    for i in range(0, TVD.size):
        p_col[n_d] = 0 # In  the last measure of the wellbore (90 if done
    every 4 degrees), the initial value is set to 0

        Min_pw, Max_pw, Po, co, Beta, V = min_pw[i] , max_pw[i], po[i], Co
    [i], beta[i], v[i]

        Sx, Sy, Szz, Tau_xy, Tau_xz, Tau_yz = sx[i], sy[i], szz[i], tau_xy
    [i], tau_xz[i], tau_yz[i]

        SL, nL = sl[i], nl[i]

        for j in range(0, n + 1):

            if p_col[n_d] == 0:

```

```
                pw = Min_pw + (j) * (Max_pw - Min_pw)/n

                for k in range(0, n_d + 1):
                    theta = np.deg2rad(k * 4)
                    # Applying the Kirsch Equations
                    sr = pw
                    stan = Sx + Sy - 2*(Sx - Sy)* np.cos(2*theta) - 4 *
    Tau_xy * np.sin(2*theta) - pw
                    sz = Szz - V*(2*(Sx - Sy)* np.cos(2*theta) + 4 *
    Tau_xy * np.sin(2*theta))
                    tau_rtan = 0
                    tau_tanz = 2*(-Tau_xz * np.sin(theta) + Tau_yz * np.
    cos(theta))
                    tau_rz = 0

                    str_wb = np.array([[sr, tau_rtan, tau_rz], [tau_rtan,
    stan, tau_tanz], [tau_rz, tau_tanz, sz]])
                    eig = np.sort(np.linalg.eigvals(str_wb))
                    s1, s2, s3  = eig[2] - Po, eig[1] - Po, eig[0] - Po

                    I1, I3 = (s1 + SL) + (s2 + SL) + (s3 + SL), (s1 + SL)
    * (s2 + SL) * (s3 + SL)

                    p_col[k] = pw

                    if I1**3/I3 - 27 > nL: #Whenever a wellbore pressure
    produces collapse the process stops and Pw increases
                        break

            else:
                break # When the 360 degrees of a wellbore have been
    analyzed and there is no collapse,

        #the inclination increases
         matrix_collapse[i] = pw
     return(matrix_collapse)


# In[ ]:


### Modified Lade Criterion for matrix collapse upper bound


# In[ ]:


def lade_collapse_u (TVD):
    matrix_collapse = np.zeros(TVD.size)
    for i in range(0, TVD.size):
        p_col[n_d] = 0 # In  the last measure of the wellbore (90 if done
    every 4 degrees), the initial value is set to 0

        Min_pw, Max_pw, Po, co, Beta, V = min_pw[i] , max_pw[i], po[i], Co
```

```
    [i], beta[i], v[i]

        Sx, Sy, Szz, Tau_xy, Tau_xz, Tau_yz = sx[i], sy[i], szz[i], tau_xy
    [i], tau_xz[i], tau_yz[i]

        SL, nL = sl[i], nl[i]

        for j in range(0, n + 1):

            if p_col[n_d] == 0:

                pw = Max_pw - (j) * (Max_pw - Min_pw)/n

                for k in range(0, n_d + 1):
                    theta = np.deg2rad(k * 4)
                    # Applying the Kirsch Equations
                    sr = pw
                    stan = Sx + Sy - 2*(Sx - Sy)* np.cos(2*theta) - 4 *
    Tau_xy * np.sin(2*theta) - pw
                    sz = Szz - V*(2*(Sx - Sy)* np.cos(2*theta) + 4 *
    Tau_xy * np.sin(2*theta))
                    tau_rtan = 0
                    tau_tanz = 2*(-Tau_xz * np.sin(theta) + Tau_yz * np.
    cos(theta))
                    tau_rz = 0

                    str_wb = np.array([[sr, tau_rtan, tau_rz], [tau_rtan,
    stan, tau_tanz], [tau_rz, tau_tanz, sz]])
                    eig = np.sort(np.linalg.eigvals(str_wb))
                    s1, s2, s3  = eig[2] - Po, eig[1] - Po, eig[0] - Po

                    I1, I3 = (s1 + SL) + (s2 + SL) + (s3 + SL), (s1 + SL)
    * (s2 + SL) * (s3 + SL)

                    p_col[k] = pw

                    if I1**3/I3 - 27 > nL: #Whenever a wellbore pressure
    produces collapse the process stops and Pw increases
                        break

            else:
                break # When the 360 degrees of a wellbore have been
    analyzed and there is no collapse,

        #the inclination increases
         matrix_collapse[i] = pw
     return(matrix_collapse)


# In[ ]:


# In imperial system
matrix_collapse = m_collapse_d (TVD) * 8.33/(0.098 * TVD)
```

```
pw_collapse = w_collapse_d (TVD) * 8.33/(0.098 * TVD)
upm_collapse = um_collapse (TVD) * 8.33/(0.098 * TVD)
matrix_tensile = m_tensile (TVD) * 8.33/(0.098 * TVD)
lade_collapse = lade_collapse_d (TVD) * 8.33/(0.098 * TVD)
lade_upper = lade_collapse_u (TVD) * 8.33/(0.098 * TVD)


# In[ ]:


# Units in imperial system
po_ppg = po * 8.33/(0.098 * TVD)
TVD_mud = TVD_mud/0.3048
S3 = S3 * 8.33/(0.098 * TVD)


# In[ ]:


mw_window_d = go.Figure()
mw_window_d.add_trace(go.Scatter(y = TVD_ft, x = matrix_collapse, line=
    dict(color='royalblue'), name='Shear failure'))
mw_window_d.add_trace(go.Scatter(x = upm_collapse, y = TVD_ft, line=dict(
    color='royalblue'), name='NC Shear failure'))
mw_window_d.add_trace(go.Scatter(x = lade_collapse, y = TVD_ft, line=dict(
    color='blue'), name='Lade Shear failure'))
mw_window_d.add_trace(go.Scatter(x = lade_upper, y = TVD_ft, line=dict(
    color='blue'), name='Lade Shear failure'))
mw_window_d.add_trace(go.Scatter(x = pw_collapse, y = TVD_ft, line=dict(
    color='red'), name='Bedding plane failure'))
mw_window_d.add_trace(go.Scatter(x = matrix_tensile, y = TVD_ft, line=dict
    (color='purple'), name='Tensile failure'))
mw_window_d.add_trace(go.Scatter(x = po_ppg, y = TVD_ft, line=dict(color='
    olive') , name='Pore pressure'))
mw_window_d.add_trace(go.Scatter(x = S3, y = TVD_ft, line=dict(color='
    orange') , name='S3'))
mw_window_d.add_trace(go.Scatter(y = TVD_mud, x = ppg, line=dict(color='
    black', width=2, dash='dot'), name='Mud Weight (ppg)'))
mw_window_d.update_yaxes(autorange="reversed")
mw_window_d.update_layout(title={'text':'Mud Window by depth','x':0.45,'y'
    :0.85}, xaxis_title='Pressure (ppg)', yaxis_title='TVD(ft)')
mw_window_d


# In[ ]:


np.savetxt('deterministic.txt',            np.c_[TVD_ft, matrix_collapse,
    pw_collapse, upm_collapse, matrix_tensile, lade_collapse, lade_upper],
               header = 'TVD, NCoulomb_L, PW, NCoulomb_U, Tensile, Lade_L,
    Lade_U')


# # Principal stresses calculation
```

```
# In[ ]:


def lade_collapse_d (TVD):
    S1 = np.zeros(TVD.size)
    S2 = np.zeros(TVD.size)
    S3 = np.zeros(TVD.size)

    for i in range(0, TVD.size):
        p_col[n_d] = 0 # In  the last measure of the wellbore (90 if done every 4 degrees), the initial value is set to 0

        Min_pw, Max_pw, Po, co, Beta, V = min_pw[i] , max_pw[i], po[i], Co[i], beta[i], v[i]

        Sx, Sy, Szz, Tau_xy, Tau_xz, Tau_yz = sx[i], sy[i], szz[i], tau_xy[i], tau_xz[i], tau_yz[i]

        SL, nL = sl[i], nl[i]

        for j in range(0, n + 1):

            if p_col[n_d] == 0:

                pw = Min_pw + (j) * (Max_pw - Min_pw)/n

                for k in range(0, n_d + 1):
                    theta = np.deg2rad(k * 4)
                    # Applying the Kirsch Equations
                    sr = pw
                    stan = Sx + Sy - 2*(Sx - Sy)* np.cos(2*theta) - 4 * Tau_xy * np.sin(2*theta) - pw
                    sz = Szz - V*(2*(Sx - Sy)* np.cos(2*theta) + 4 * Tau_xy * np.sin(2*theta))
                    tau_rtan = 0
                    tau_tanz = 2*(-Tau_xz * np.sin(theta) + Tau_yz * np.cos(theta))
                    tau_rz = 0

                    str_wb = np.array([[sr, tau_rtan, tau_rz], [tau_rtan, stan, tau_tanz], [tau_rz, tau_tanz, sz]])
                    eig = np.sort(np.linalg.eigvals(str_wb))
                    s1, s2, s3  = eig[2] - Po, eig[1] - Po, eig[0] - Po

                    I1, I3 = (s1 + SL) + (s2 + SL) + (s3 + SL), (s1 + SL) * (s2 + SL) * (s3 + SL)

                    p_col[k] = pw

                    if I1**3/I3 - 27 > nL: #Whenever a wellbore pressure produces collapse the process stops and Pw increases
                        break

```

```
            else:
                break # When the 360 degrees of a wellbore have been analyzed and there is no collapse,

        #the inclination increases
         S1[i] = s1
         S2[i] = s2
         S3[i] = s3

    return{'S1': S1, 'S2': S2, 'S3': S3}


# In[ ]:


stress = lade_collapse_d(TVD)
S1, S2, S3  = stress.get('S1')* 8.33/(0.098 * TVD), stress.get('S2') * 8.33/(0.098 * TVD),                                stress.get('S3') * 8.33/(0.098 * TVD)
np.savetxt('stress_bh.txt', np.c_[TVD_ft, S1, S2, S3], header = 'TVD, S1, S2, S3')


# In[ ]:







# In[ ]:


sl[10]


# In[ ]:
```

```
#!/usr/bin/env python
# coding: utf-8

# # Wellbore Stability - Plane of weakness failures

# In[1]:


import numpy as np
import pandas as pd
import math


# ## Data:

# In[2]:

```

```
18
19 stress =  pd.read_excel(r'z36_data.xlsx', sheet_name = '1') #Import of
      Excel Sheet
20 mud =  pd.read_excel(r'z36_data.xlsx', sheet_name = '2') #Import of Excel
      Sheet
21 TVD = np.asarray(stress.loc[:,"TVD_KB"]/3.28084) # TVD in m
22 TVD_ft = np.asarray(stress.loc[:,"TVD_KB"]) # TVD in ft
23 TVD_mud = np.asarray(mud.loc[:,"TVD"]) # TVD in m for the 2nd sheet
24 ppg = np.asarray(mud.loc[:,"Density"]) # Mud density in ppg for the well
25 sx = np.asarray(stress.loc[:,"BSx"]) # Pa
26 sy = np.asarray(stress.loc[:,"BSy"]) # Pa
27 szz = np.asarray(stress.loc[:,"BSv"]) # Pa
28 tau_xy = np.asarray(stress.loc[:,"BTxy"]) # Pa
29 tau_xz = np.asarray(stress.loc[:,"BTzx"]) # Pa
30 tau_yz = np.asarray(stress.loc[:,"BTyz"]) # Pa
31 po = np.asarray(stress.loc[:,"Po"]) # sg
32 a = np.asarray(stress.loc[:,"Az"]) # Azimuth
33 inc = np.asarray(stress.loc[:,"Incl"]) # Inclination
34 S3 = np.asarray(stress.loc[:,"S3"]/(10**5)) # S3  in bar
35 v = np.asarray(stress.loc[:,"Poisson"]) # Poisson ratio
36 fi = np.deg2rad(np.asarray(stress.loc[:,"FrictionAngle"])) # Friction
      angle of rock matrix
37 sw = np.asarray(stress.loc[:,"So"]/(10**5)) * 0.9 # Cohesion plane of
      weakness
38 uw = np.tan(fi) * 0.5 # fw , Friction coefficient of plane of weakness.
39
40
41 # In[3]:
42
43
44 # Estimation for min_pw and max_pw
45 min_pw = po  # Minimum  Well Pressure considered, bar
46 max_pw = 2.3 * 0.098 * TVD # Maximum  Well Pressure considered, bar
47 n = 500 #  Iterations  between the minimum wellbore pressure and the
      maximum well pressure
48 n_d = 360//4 #  Iterations  for the wellbore directions, there will be a
      measure every 4 degrees
49 p_col = np.zeros(n_d + 1)
50 p_frac = np.zeros(n_d + 1)
51 max_iw = 15 + 1 # In  the field, max 15 degrees inclination
52 max_aw = 360 + 1 # 360  degrees of azimuth
53
54
55 # # Variation of the plane of weakness orientation
56
57 # 10 points have been selected from the intersection of the mud window
      employed and the plane of weakness collapse. The objective is observing
       which are the worst and best case scenarios in terms of possible
      orientations of the plane of weakness failure with the employed
      inclinations and azimuths. Iteration around 360 aw and 90 iw (azimuth
      and inclination of pw).
58
59 # In[4]:
60
```

```
61
62 # Plane of weakness selected
63
64 def w_collapse_s (x):
65     pw_collapse = np.zeros((max_iw) * (max_aw))
66     sq_iw = np.zeros((max_iw) * (max_aw))
67     sq_aw = np.zeros((max_iw) * (max_aw))
68     i = 0
69
70     A, Inc = a[x], inc[x]
71     Min_pw, Max_pw, Sx, Sy, Szz, Tau_xy, Tau_xz, Tau_yz, Po, V = min_pw[x]
       , max_pw[x], sx[x], sy[x],
                                szz[x], tau_xy[x], tau_xz[x], tau_yz[x], po[x
       ],
       v[x]
72     Sw, fw = sw[x], uw[x]
73
74     R1 = np.array([[np.cos(A) * np.cos(Inc), np.sin(A) * np.cos(Inc), -np.
       sin(Inc)]
75                    , [-np.sin(A), np.cos(A), 0],
76                    [np.cos(A) * np.sin(Inc), np.sin(A) * np.sin(Inc), np.
       cos(Inc)]])
77
78     for iw in range(0, max_iw):
79
80         for aw in range(0, max_aw):
81
82             p_col[n_d] = 0 # In  the last measure of the wellbore (90 if
       done every 4 degrees), the initial value is set to 0
83
84             R2 = np.array([[np.cos(aw) * np.cos(iw), np.sin(aw) * np.cos(
       iw), -np.sin(iw)]
85                    , [-np.sin(aw), np.cos(aw), 0],
86                    [np.cos(aw) * np.sin(iw), np.sin(aw) * np.sin(iw), np.
       cos(iw)]])
87
88             for j in range(0, n + 1):
89
90                 if p_col[n_d] == 0:
91
92                     pw = Min_pw + (j) * (Max_pw - Min_pw)/n
93                     for k in range(0, n_d + 1):
94                         theta = np.deg2rad(k * 4)
95
96                         Rz = np.array([[np.cos(theta), np.sin(theta), 0],
       [-np.sin(theta), np.cos(theta), 0], [0, 0, 1]])
97                         # Applying the Kirsch Equations
98                         sr = pw
99                         stan = Sx + Sy - 2*(Sx - Sy)* np.cos(2*theta) - 4
       * Tau_xy * np.sin(2*theta) - pw
100                        sz = Szz - V*(2*(Sx - Sy)* np.cos(2*theta) + 4 *
       Tau_xy * np.sin(2*theta))
101                        tau_rtan = 0
102                        tau_tanz = 2*(-Tau_xz * np.sin(theta) + Tau_yz *
```

```
    np.cos(theta))
                        tau_rz = 0

                        str_wb = np.array([[sr, tau_rtan, tau_rz], [
    tau_rtan, stan, tau_tanz], [tau_rz, tau_tanz, sz]])
                        str_wb = np.linalg.multi_dot([R2, np.transpose(R1)
    , np.transpose(Rz), str_wb, Rz, R1, np.transpose(R2)])

                        tau = np.sqrt(str_wb[2, 0]**2 + str_wb[2, 1]**2)
                        szw = str_wb[2, 2] - Po

                        p_col[k] = pw

                        if tau > Sw + fw * szw: #Whenever a wellbore
    pressure produces collapse the process stops and Pw increases
                            break

                else:
                    break # When the 360 degrees of a wellbore have been
    analyzed and there is no collapse,

                #the pw increases
            pw_collapse[i] = pw
            sq_iw[i] = iw
            sq_aw[i] = aw
            i += 1

     return{'pw_collapse': pw_collapse,'sq_iw': sq_iw, 'sq_aw':sq_aw}


# In[ ]:


# selected indeces [68, 83, 100, 126, 147, 167, 188, 208, 224, 229]
pw_selected_1 = w_collapse_s (68)
pw_selected_2 = w_collapse_s (83)
pw_selected_3 = w_collapse_s (100)
pw_selected_4 = w_collapse_s (126)
pw_selected_5 = w_collapse_s (147)
pw_selected_6 = w_collapse_s (167)
pw_selected_7 = w_collapse_s (188)
pw_selected_8 = w_collapse_s (208)
pw_selected_9 = w_collapse_s (224)
pw_selected_10 = w_collapse_s (229)


# In[11]:


# TXT FILES
# 1
pw_collapse, inc_w, az_w  = pw_selected_1.get('pw_collapse')* 8.33/(0.098
    * TVD[68]), pw_selected_1.get('sq_iw'),
    pw_selected_1.get('sq_aw')
```

```
np.savetxt('tvd_1.txt', np.c_[inc_w, az_w, pw_collapse], header = 'iw, aw, pw ')
# 2
pw_collapse, inc_w, az_w  = pw_selected_2.get('pw_collapse')* 8.33/(0.098 * TVD[83]), pw_selected_2.get('sq_iw'), pw_selected_2.get('sq_aw')
np.savetxt('tvd_2.txt', np.c_[inc_w, az_w, pw_collapse], header = 'iw, aw, pw ')
# 3
pw_collapse, inc_w, az_w  = pw_selected_3.get('pw_collapse')* 8.33/(0.098 * TVD[100]), pw_selected_3.get('sq_iw'), pw_selected_3.get('sq_aw')
np.savetxt('tvd_3.txt', np.c_[inc_w, az_w, pw_collapse], header = 'iw, aw, pw ')
# 4
pw_collapse, inc_w, az_w  = pw_selected_4.get('pw_collapse')* 8.33/(0.098 * TVD[126]), pw_selected_4.get('sq_iw'), pw_selected_4.get('sq_aw')
np.savetxt('tvd_4.txt', np.c_[inc_w, az_w, pw_collapse], header = 'iw, aw, pw ')
# 5
pw_collapse, inc_w, az_w  = pw_selected_5.get('pw_collapse')* 8.33/(0.098 * TVD[147]), pw_selected_5.get('sq_iw'), pw_selected_5.get('sq_aw')
np.savetxt('tvd_5.txt', np.c_[inc_w, az_w, pw_collapse], header = 'iw, aw, pw ')
# 6
pw_collapse, inc_w, az_w  = pw_selected_6.get('pw_collapse')* 8.33/(0.098 * TVD[167]), pw_selected_6.get('sq_iw'), pw_selected_6.get('sq_aw')
np.savetxt('tvd_6.txt', np.c_[inc_w, az_w, pw_collapse], header = 'iw, aw, pw ')
# 7
pw_collapse, inc_w, az_w  = pw_selected_7.get('pw_collapse')* 8.33/(0.098 * TVD[188]), pw_selected_7.get('sq_iw'), pw_selected_7.get('sq_aw')
np.savetxt('tvd_7.txt', np.c_[inc_w, az_w, pw_collapse], header = 'iw, aw, pw ')
# 8
pw_collapse, inc_w, az_w  = pw_selected_8.get('pw_collapse')* 8.33/(0.098 * TVD[208]), pw_selected_8.get('sq_iw'), pw_selected_8.get('sq_aw')
np.savetxt('tvd_8.txt', np.c_[inc_w, az_w, pw_collapse], header = 'iw, aw, pw ')
# 9
pw_collapse, inc_w, az_w  = pw_selected_9.get('pw_collapse')* 8.33/(0.098 * TVD[224]), pw_selected_9.get('sq_iw'), pw_selected_9.get('sq_aw')
np.savetxt('tvd_9.txt', np.c_[inc_w, az_w, pw_collapse], header = 'iw, aw, pw ')
# 10
pw_collapse, inc_w, az_w  = pw_selected_10.get('pw_collapse')* 8.33/(0.098 * TVD[229]), pw_selected_10.get('sq_iw'), pw_selected_10.get('sq_aw')
```

```
176 np.savetxt('tvd_10.txt', np.c_[inc_w, az_w, pw_collapse], header = 'iw, aw
    , pw ')
177
178
179 # In[ ]:
180
181
182 pw_1 = pw_selected_1.get('pw_collapse')* 8.33/(0.098 * TVD[68])
183 pw_2 = pw_selected_2.get('pw_collapse')* 8.33/(0.098 * TVD[83])
184 pw_3 = pw_selected_3.get('pw_collapse')* 8.33/(0.098 * TVD[100])
185 pw_4 = pw_selected_4.get('pw_collapse')* 8.33/(0.098 * TVD[126])
186 pw_5 = pw_selected_5.get('pw_collapse')* 8.33/(0.098 * TVD[147])
187 pw_6 = pw_selected_6.get('pw_collapse')* 8.33/(0.098 * TVD[167])
188 pw_7 = pw_selected_7.get('pw_collapse')* 8.33/(0.098 * TVD[188])
189 pw_8 = pw_selected_8.get('pw_collapse')* 8.33/(0.098 * TVD[208])
190 pw_9 = pw_selected_9.get('pw_collapse')* 8.33/(0.098 * TVD[224])
191 pw_10 = pw_selected_10.get('pw_collapse')* 8.33/(0.098 * TVD[229])
192
193
194 # # Stochastic Analysis (only planes of weakness)
195
196 # In[12]:
197
198
199 # Parameters with uncertainty
200
201 np.random.seed(16)
202
203 ms = 5000 # Amount  of repetitions
204 iw_m = np.random.uniform(0, 15, size= ms) # iw, Inclination of the plane
    of weakness. Uniform distribution. 5k
205 aw_m = np.random.uniform(0, 360, size= ms) # aw, Azimuth of the plane of
    weakness. Uniform distribution.5k
206 # Estimation for min_pw and max_pw
207 min_pw = po  # Minimum  Well Pressure considered, bar
208 max_pw = S3 # Maximum  Well Pressure considered, bar
209 n = 100 # Iterations  between the minimum wellbore pressure and the
    maximum well pressure
210 n_d = 360//4 # Iterations  for the wellbore directions, there will be a
    measure every 4 degrees
211 p_col = np.zeros(n_d + 1)
212
213
214 # In[13]:
215
216
217 def w_collapse_d (x):
218     pw_collapse = np.zeros(ms)
219     # Code selects the row in the matrices according to x
220
221     p_col[n_d] = 0 # In  the last measure of the wellbore (90 if done
    every 4 degrees), the initial value is set to 0
222
223     A, Inc = a[x], inc[x]
```

```
224     Min_pw, Max_pw, Sx, Sy, Szz, Tau_xy, Tau_xz, Tau_yz, Po, V = min_pw[x]
     , max_pw[x], sx[x], sy[x],
                              szz[x], tau_xy[x], tau_xz[x], tau_yz[x], po[x
    ],
    v[x]
225     Sw, fw = sw[x], uw[x]
226
227     R1 = np.array([[np.cos(A) * np.cos(Inc), np.sin(A) * np.cos(Inc), -np.
    sin(Inc)]
228                    , [-np.sin(A), np.cos(A), 0],
229                    [np.cos(A) * np.sin(Inc), np.sin(A) * np.sin(Inc), np.
    cos(Inc)]])
230
231     for i in range(0, ms): # Collapse pressure for every parameter. Code
    selects specific value in the row
232
233         p_col[n_d] = 0 # In  the last measure of the wellbore (90 if done
    every 4 degrees), the initial value is set to 0
234
235         iw, aw = iw_m[i], aw_m[i]
236
237         R2 = np.array([[np.cos(aw) * np.cos(iw), np.sin(aw) * np.cos(iw),
    -np.sin(iw)]
238                    , [-np.sin(aw), np.cos(aw), 0],
239                    [np.cos(aw) * np.sin(iw), np.sin(aw) * np.sin(iw), np.
    cos(iw)]])
240
241         for j in range(0, n + 1):
242
243             if p_col[n_d] == 0:
244
245                 pw = Min_pw + (j) * (Max_pw - Min_pw)/n
246
247                 for k in range(0, n_d + 1):
248
249                     theta = np.deg2rad(k * 4)
250
251                     Rz = np.array([[np.cos(theta), np.sin(theta), 0], [-np
    .sin(theta), np.cos(theta), 0], [0, 0, 1]])
252                     # Applying the Kirsch Equations
253                     sr = pw
254                     stan = Sx + Sy - 2*(Sx - Sy)* np.cos(2*theta) - 4 *
    Tau_xy * np.sin(2*theta) - pw
255                     sz = Szz - V*(2*(Sx - Sy)* np.cos(2*theta) + 4 *
    Tau_xy * np.sin(2*theta))
256                     tau_rtan = 0
257                     tau_tanz = 2*(-Tau_xz * np.sin(theta) + Tau_yz * np.
    cos(theta))
258                     tau_rz = 0
259
260                     str_wb = np.array([[sr, tau_rtan, tau_rz], [tau_rtan,
    stan, tau_tanz], [tau_rz, tau_tanz, sz]])
261                     str_wb = np.linalg.multi_dot([R2, np.transpose(R1), np
    .transpose(Rz), str_wb, Rz, R1, np.transpose(R2)])
```

```
262
263                     tau = np.sqrt(str_wb[2, 0]**2 + str_wb[2, 1]**2)
264                     szw = str_wb[2, 2] - Po
265                     p_col[k] = pw
266
267                     if tau > Sw + fw * szw: #Whenever a wellbore pressure
    produces collapse the process stops and Pw increases
268                         break
269
270             else:
271                 break # When the 360 degrees of a wellbore have been
    analyzed and there is no collapse,
272
            #the inclination increases
273         pw_collapse[i] = pw * 8.33/(0.098 * TVD[x]) # Value in ppg
274
275     return(np.mean(pw_collapse), np.std(pw_collapse))
276
277
278 # In[14]:
279
280
281 import time
282 start_time = time.time()
283 e = {}
284 for x in range(0, TVD.size):
285     print(x)
286     e[TVD_ft[x]] = w_collapse_d(x)
287
288 print ("My program took", time.time() - start_time, "to run")
289
290
291 # In[15]:
292
293
294 # Creates txt file ready to use without brackets and multiple spaces
295 file = open("pw_orientation.txt","w")
296
297 for key in e.keys():
298
299     file.write(str(key)+" "+str(e[key]))
300     file.write("\n")
301
302 file.close()
303
304 import re
305
306 with open('pw_orientation.txt', 'r') as f: # deletes brackets
307     text = f.read()
308     patn = re.sub(r"[\([{})\]]", "", text) # regex pattern to detect
    brackets
309
310 with open('pw_orientation.txt', 'w') as my_file: # saves the changes
311     my_file.write(patn)
```

```
312
313 with open('pw_orientation.txt', 'r') as f: # deletes extra spaces
314     text = f.read()
315     subbed = re.sub(r'\s{2,}',' ',text)
316
317 with open('pw_orientation.txt', 'w') as my_file: # saves the changes
318     my_file.write(subbed)
319
320
321 # In[ ]:
```

```
1  #!/usr/bin/env python
2  # coding: utf-8
3
4  # # Wellbore Stability - Stochastic ppg vs ft
5
6  # For each depth a stochastic analysis for the failure modes will be
       performed. 10000 iterations will be performed, means will be plotted
       and a SF of 1 SD will be taken.
7
8  # In[15]:
9
10
11 import numpy as np
12 import pandas as pd
13 import math
14
15
16 # In[16]:
17
18
19 ### Data:
20 stress =  pd.read_excel(r'z36_data.xlsx', sheet_name = '1') #Import of
       Excel Sheet
21 mud =  pd.read_excel(r'z36_data.xlsx', sheet_name = '2') #Import of Excel
       Sheet
22 TVD = np.asarray(stress.loc[:,"TVD_KB"]/3.28084) # TVD in m
23 TVD_ft = np.asarray(stress.loc[:,"TVD_KB"]) # TVD in ft
24 TVD_mud = np.asarray(mud.loc[:,"TVD"]) # TVD in m for the 2nd sheet
25 ppg = np.asarray(mud.loc[:,"Density"]) # Mud density in ppg for the well
26 a = np.asarray(stress.loc[:,"Az"]) # Azimuth
27 inc = np.asarray(stress.loc[:,"Incl"]) # Inclination
28 S3 = np.asarray(stress.loc[:,"S3"]/(10**5)) # S3  in bar
29
30 sx = np.asarray(stress.loc[:,"BSx"]) # Pa
31 sy = np.asarray(stress.loc[:,"BSy"]) # Pa
32 szz = np.asarray(stress.loc[:,"BSv"]) # Pa
33 tau_xy = np.asarray(stress.loc[:,"BTxy"]) # Pa
34 tau_xz = np.asarray(stress.loc[:,"BTzx"]) # Pa
35 tau_yz = np.asarray(stress.loc[:,"BTyz"]) # Pa
36 po = np.asarray(stress.loc[:,"Po"]) # sg, no uncertainty as it is easy to
       measure
37 tr = np.asarray(stress.loc[:,"TensileS"]/(10**5)) # Tensile Strength in
       bar
```

```
v = np.asarray(stress.loc[:,"Poisson"]) # Poisson ratio
So = np.asarray(stress.loc[:,"So"]/(10**5)) # Cohesion rock matrix in bar
fi = np.asarray(stress.loc[:,"FrictionAngle"]) # Friction angle of rock
    matrix in degrees


# In[17]:


# Parameters with uncertainty

np.random.seed(16)

ms = 10000 #  Amount   of repetitions

# Stresses in the wellbore wall:
# sx . Assume 1 sd
sx_m = np.zeros((230, ms))
for i in range (0, sx.size):
    rn = np.random.normal(sx[i], scale=5, size=ms)
    sx_m[i] = rn

# sy
sy_m = np.zeros((230, ms))
for i in range (0, sy.size):
    rn = np.random.normal(sy[i], scale = 5, size= ms)
    sy_m[i] = rn

# szz
szz_m = np.zeros((230, ms))
for i in range (0, sx.size):
    rn = np.random.normal(szz[i], scale = 5, size = ms)
    szz_m[i] = rn

# tau_xy
tau_xy_m = np.zeros((230, ms))
for i in range (0, sx.size):
    rn = np.random.normal(tau_xy[i], scale = 5, size= ms)
    tau_xy_m[i] = rn

# tau_xz
tau_xz_m = np.zeros((230, ms))
for i in range (0, sx.size):
    rn = np.random.normal(tau_xz[i], scale = 5, size = ms)
    tau_xz_m[i] = rn

# tau_yz
tau_yz_m = np.zeros((230, ms))
for i in range (0, sx.size):
    rn = np.random.normal(tau_yz[i], scale=5, size=ms)
    tau_yz_m[i] = rn

# v, V_m
V_m = np.zeros((230, ms))
```

```
for i in range (0, sx.size):
    rn = np.random.normal(v[i], scale=0.08, size=ms)
    V_m[i] = rn

# fi ASSUME 5 DEGREES
fi_m = np.zeros((230, ms))
for i in range (0, sx.size):
    rn = np.random.triangular(fi[i] - 5, fi[i], fi[i] + 5, size=ms)
    fi_m[i] = np.deg2rad(rn)

# So
So_m = np.zeros((230, ms))
for i in range (0, sx.size):
    rn = np.random.normal(So[i], scale=2, size=ms)
    So_m[i] = rn

iw_m = np.random.uniform(0, 15, size= ms) # iw, Inclination of the plane
   of weakness. Uniform distribution. 10k
aw_m = np.random.uniform(0, 360, size= ms) # aw, Azimuth of the plane of
   weakness. Uniform distribution.10k

sl = So_m/np.tan(fi_m) # SL, 230k
nl = 4 * (np.tan(fi_m))**2 * (9 - 7 * np.sin(fi_m))/(1 - np.sin(fi_m)) #
   NL, 230k

beta = fi_m * 0.5 + np.pi/4 # beta in radians # BETA , 230k
Co = 2 * np.asarray(So_m) * np.tan(beta) # Cohesion in bar, 230k
uw_m = np.tan(fi_m) * np.random.uniform(0.5, 0.8) # fw , Friction
   coefficient of plane of weakness, 230k
sw_m = So_m * np.random.uniform(0.8, 1) # Cohesion plane of weakness, 230k


# In[21]:


# Estimation for min_pw and max_pw
min_pw = po  # Minimum  Well Pressure considered, bar
max_pw = S3 # Maximum  Well Pressure considered, bar
n = 100 # Iterations  between the minimum wellbore pressure and the
   maximum well pressure
n_d = 360//4 # Iterations  for the wellbore directions, there will be a
   measure every 4 degrees
p_col = np.zeros(n_d + 1)


# In[5]:


def lade_collapse_d (x):
    matrix_collapse = np.zeros(aw_m.size)
    # Code selects the row in the matrices according to x
    Min_pw, Max_pw, Co_m, beta_m, v_m = min_pw[x] , max_pw[x], Co[x], beta
   [x], V_m[x]

```

```
 sx, sy, szz, tau_xy, tau_xz, tau_yz = sx_m[x], sy_m[x], szz_m[x], tau_xy_m[x], tau_xz_m[x], tau_yz_m[x]

 sl_m, nl_m, Po = sl[x], nl[x], po[x]

 for i in range(0, aw_m.size): # Collapse pressure for every parameter. Code selects specific value in the row

     p_col[n_d] = 0 # In  the last measure of the wellbore (90 if done every 4 degrees), the initial value is set to 0

     co, Beta, V = Co_m[i], beta_m[i], v_m[i]

     Sx, Sy, Szz, Tau_xy, Tau_xz, Tau_yz = sx[i], sy[i], szz[i], tau_xy[i], tau_xz[i], tau_yz[i]

     SL, nL = sl_m[i], nl_m[i]

     for j in range(0, n + 1):

         if p_col[n_d] == 0:

             pw = Min_pw + (j) * (Max_pw - Min_pw)/n

             for k in range(0, n_d + 1):
                 theta = np.deg2rad(k * 4)
                 # Applying the Kirsch Equations
                 sr = pw
                 stan = Sx + Sy - 2*(Sx - Sy)* np.cos(2*theta) - 4 * Tau_xy * np.sin(2*theta) - pw
                 sz = Szz - V*(2*(Sx - Sy)* np.cos(2*theta) + 4 * Tau_xy * np.sin(2*theta))
                 tau_rtan = 0
                 tau_tanz = 2*(-Tau_xz * np.sin(theta) + Tau_yz * np.cos(theta))
                 tau_rz = 0

                 str_wb = np.array([[sr, tau_rtan, tau_rz], [tau_rtan, stan, tau_tanz], [tau_rz, tau_tanz, sz]])
                 eig = np.sort(np.linalg.eigvals(str_wb))
                 s1, s2, s3  = eig[2] - Po, eig[1] - Po, eig[0] - Po

                 I1, I3 = (s1 + SL) + (s2 + SL) + (s3 + SL), (s1 + SL) * (s2 + SL) * (s3 + SL)
                 p_col[k] = pw

                 if I1**3/I3 - 27 > nL: #Whenever a wellbore pressure produces collapse the process stops and Pw increases
                     break

         else:
             break # When the 360 degrees of a wellbore have been analyzed and there is no collapse,

```

```
            #the inclination increases
        matrix_collapse[i] = pw

    return(np.mean(matrix_collapse), np.std(matrix_collapse), np.
    percentile(matrix_collapse, 25),           np.percentile(
    matrix_collapse, 75), np.percentile(matrix_collapse, 90))


# In[6]:


import time
start_time = time.time()
d = {}
for x in range(0, sx.size):
    print(x)
    d[TVD_ft[x]] = lade_collapse_d(x)

print ("My program took", time.time() - start_time, "to run")


# In[7]:


# Creates txt file ready to use without brackets and multiple spaces
file = open("stlade.txt","w")

for key in d.keys():

   file.write(str(key)+" "+str(d[key]))
   file.write("\n")

file.close()

import re

with open('stlade.txt', 'r') as f: # deletes brackets
    text = f.read()
    patn = re.sub(r"[\([{})\]]", "", text) # regex pattern to detect
    brackets

with open('stlade.txt', 'w') as my_file: # saves the changes
    my_file.write(patn)

with open('stlade.txt', 'r') as f: # deletes extra spaces
    text = f.read()
    subbed = re.sub(r'\s{2,}',' ',text)

with open('stlade.txt', 'w') as my_file: # saves the changes
    my_file.write(subbed)


# In[22]:

```

```
231
232 def w_collapse_d (x):
233     pw_collapse = np.zeros(ms)
234     # Code selects the row in the matrices according to x
235     Min_pw, Max_pw, v_m = min_pw[x] , max_pw[x], V_m[x]
236
237     sx, sy, szz, tau_xy, tau_xz, tau_yz = sx_m[x], sy_m[x], szz_m[x],
    tau_xy_m[x], tau_xz_m[x], tau_yz_m[x]
238
239     Po, A, Inc, uw, sw = po[x], a[x], inc[x], uw_m[x], sw_m[x]
240
241     R1 = np.array([[np.cos(A) * np.cos(Inc), np.sin(A) * np.cos(Inc), -np.
    sin(Inc)]
242                    , [-np.sin(A), np.cos(A), 0],
243                    [np.cos(A) * np.sin(Inc), np.sin(A) * np.sin(Inc), np.
    cos(Inc)]])
244
245     for i in range(0, ms): # Collapse pressure for every parameter. Code
    selects specific value in the row
246
247         p_col[n_d] = 0 # In  the last measure of the wellbore (90 if done
    every 4 degrees), the initial value is set to 0
248
249         Sx, Sy, Szz, Tau_xy, Tau_xz, Tau_yz = sx[i], sy[i], szz[i], tau_xy
    [i], tau_xz[i], tau_yz[i]
250
251         iw, aw, Sw, fw, V = iw_m[i], aw_m[i], sw[i], uw[i], v_m[i]
252
253         R2 = np.array([[np.cos(aw) * np.cos(iw), np.sin(aw) * np.cos(iw),
    -np.sin(iw)]
254                        , [-np.sin(aw), np.cos(aw), 0],
255                        [np.cos(aw) * np.sin(iw), np.sin(aw) * np.sin(iw), np.
    cos(iw)]])
256
257         for j in range(0, n + 1):
258
259             if p_col[n_d] == 0:
260
261                 pw = Min_pw + (j) * (Max_pw - Min_pw)/n
262
263                 for k in range(0, n_d + 1):
264
265                     theta = np.deg2rad(k * 4)
266
267                     Rz = np.array([[np.cos(theta), np.sin(theta), 0], [-np
    .sin(theta), np.cos(theta), 0], [0, 0, 1]])
268                     # Applying the Kirsch Equations
269                     sr = pw
270                     stan = Sx + Sy - 2*(Sx - Sy)* np.cos(2*theta) - 4 *
    Tau_xy * np.sin(2*theta) - pw
271                     sz = Szz - V*(2*(Sx - Sy)* np.cos(2*theta) + 4 *
    Tau_xy * np.sin(2*theta))
272                     tau_rtan = 0
273                     tau_tanz = 2*(-Tau_xz * np.sin(theta) + Tau_yz * np.
```

```
    cos(theta))
                    tau_rz = 0

                    str_wb = np.array([[sr, tau_rtan, tau_rz], [tau_rtan,
    stan, tau_tanz], [tau_rz, tau_tanz, sz]])
                    str_wb = np.linalg.multi_dot([R2, np.transpose(R1), np
    .transpose(Rz), str_wb, Rz, R1, np.transpose(R2)])

                    tau = np.sqrt(str_wb[2, 0]**2 + str_wb[2, 1]**2)
                    szw = str_wb[2, 2] - Po
                    p_col[k] = pw

                    if tau > Sw + fw * szw: #Whenever a wellbore pressure
    produces collapse the process stops and Pw increases
                        break

            else:
                break # When the 360 degrees of a wellbore have been
    analyzed and there is no collapse,

           #the inclination increases
        pw_collapse[i] = pw * 8.33/(0.098 * TVD[x])

     return(np.mean(pw_collapse), np.std(pw_collapse), np.percentile(
    pw_collapse, 25),           np.percentile(pw_collapse, 75), np.
    percentile(pw_collapse, 90))


# In[23]:


import time
start_time = time.time()
e = {}
for x in range(0, sx.size):
    print(x)
    e[TVD_ft[x]] = w_collapse_d(x)

print ("My program took", time.time() - start_time, "to run")


# In[24]:


# Creates txt file ready to use without brackets and multiple spaces
file = open("pw.txt","w")

for key in e.keys():

   file.write(str(key)+" "+str(e[key]))
   file.write("\n")

file.close()

```

```
import re

with open('pw.txt', 'r') as f: # deletes brackets
    text = f.read()
    patn = re.sub(r"[\([{})\]]", "", text) # regex pattern to detect
   brackets

with open('pw.txt', 'w') as my_file: # saves the changes
    my_file.write(patn)

with open('pw.txt', 'r') as f: # deletes extra spaces
    text = f.read()
    subbed = re.sub(r'\s{2,}',' ',text)

with open('pw.txt', 'w') as my_file: # saves the changes
    my_file.write(subbed)


# In[ ]:
```

APPENDIX 2. GNUPLOT CODES

.

```
#!/usr/bin/env python
# coding: utf-8

# In[3]:


get_ipython().run_line_magic('load_ext', 'gnuplot_kernel')


# # Failure at constant depth, varying inclination

# In[ ]:


get_ipython().run_cell_magic('gnuplot', '', '\nset output \'failure_modes.
   png\'\nset nokey\nset grid\n\nset terminal pngcairo size 550,300 font "
   Calibri,14"\n\n# Line style for axes\nset style line 80 lt rgb
   "#808080"\n\n# Line style for grid\nset style line 81 lt 0  # dashed\
   nset style line 81 lt rgb "#808080"  # grey\n\nset grid back linestyle
   81\nset border 3 back linestyle 80 # Remove border on top and right.
   These borders are useless and make it harder \n    #to see plotted
   lines near the border. Also, put it in grey; no need for so much
   emphasis on a border.\nset xtics nomirror\nset ytics nomirror\n\nset
   yrange [13 : 17.5]\nset xlabel \'Inclination (degrees)\'\nset ylabel \'
   Equivalent mudweight (ppg)\'\nset title \'Failure modes vs Inclination
   \'\nset label "Tensile failure" left at 5, 17.3 font", 10"\nset label "
   Mud pressure" left at 5, 14.8 font", 10"\nset label "Pore pressure"
   right at 80, 14 font", 10"\nset label "Bedding plane failure" right at
   80, 14.9 font", 10"\nset label "Shear failure" left at 50, 16.8 font",
   10"\nset label "Shear failure" left at 5, 14.3 font", 10"\n\nset style
   line 1 lw 2 lt 2 dt 3 lc rgb "gray0"  # mud weight\nset style line 2 lw
    2 lt 1 lc rgb "gray50"  # Pore Pressure\nset style line 3 lw 2 lt 1 lc
    rgb "dark-goldenrod" # S3\nset style line 4 lw 2 lt 1 lc rgb "#A00000"
    #\xa0Plane of Weakness Failure\nset style line 5 lw 2 lt 1 lc rgb "
   purple" #\xa0Tensile Failure\nset style line 6 lw 2 lt 1 lc rgb "
   royalblue" #\xa0Navier Coulomb Criterion\nset style line 7 lw 2 lt 1 lc
    rgb "blue" #\xa0Lade Modified Criterion\n\n\nplot \'failure.txt\' u
   1:2 title \'Shear failure\' w l lw 2 lc rgb \'royalblue\', \\\n    \'
   failure.txt\' u 1:3 title \'Bedding plane failure\' w l ls 4, \\\n
   \'failure.txt\' u 1:5 title \'Shear failure\' w l ls 6, \\\n    \'
   failure.txt\' u 1:6 title \'Tensile failure\' w l ls 5,\\\n    \'
   failure.txt\' u 1:7 title \'Pore pressure\' w l ls 2,\\\n    \'failure.
   txt\' u 1:8 title \'Mud pressure\' w l ls 1 \n\nset yrange [13 : 18.5]\
   n\nset output \'failure_modes_def.png\'\nplot \'failure.txt\' u 1:9
   title \'Shear failure\' w l lw 2 lc rgb \'royalblue\', \\\n    \'
   failure.txt\' u 1:3 title \'Bedding plane failure\' w l ls 4, \\\n
   \'failure.txt\' u 1:10 title \'Shear failure\' w l ls 6, \\\n    \'
   failure.txt\' u 1:6 title \'Tensile failure\' w l ls 5,\\\n    \'
   failure.txt\' u 1:7 title \'Pore pressure\' w l ls 2,\\\n    \'failure.
   txt\' u 1:8 title \'Mud pressure\' w l ls 1 ')
```

```
16
17
18 # # Comparison between Navier Coulomb and Lade Modified
19
20 # In[3]:
21
22
23 get_ipython().run_cell_magic('gnuplot', '', 'reset\nset terminal pngcairo
   size 800,800 font "Calibri,14"\nset termoption enhanced\n\nset grid\
   nset rmargin 19\nset key at screen 1, graph 1\nset key font ",12"\n\n#
   Line style for axes\nset style line 80 lt rgb "#808080"\n\n# Line style
    for grid\nset style line 81 lt 0  # dashed\nset style line 81 lt rgb
   "#808080"  # grey\n\nset grid back linestyle 81\nset border 3 back
   linestyle 80 # Remove border on top and right.  These borders are
   useless and make it harder \n    #to see plotted lines near the border.
    Also, put it in grey; no need for so much emphasis on a border.\nset
   xtics nomirror\nset ytics nomirror\n\nset yrange [11000 : 3000]\n\nset
   xlabel \'Equivalent mudweight (ppg)\'\nset ylabel \'TVD (ft)\'\nset
   title \' Failure modes vs Depth\'\n\nset style line 1 lw 2 lt 2 dt 3 lc
    rgb "gray0"  # mud weight\nset style line 2 lw 2 lt 1 lc rgb "gray50"
    # Pore Pressure\nset style line 3 lw 2 lt 1 lc rgb "dark-goldenrod" #
   S3\nset style line 4 lw 2 lt 1 lc rgb "#A00000" #\xa0Plane of Weakness
   Failure\nset style line 5 lw 2 lt 1 lc rgb "purple" #\xa0Tensile
   Failure\nset style line 6 lw 2 lt 1 lc rgb "royalblue" #\xa0Navier
   Coulomb Criterion\nset style line 7 lw 2 lt 1 lc rgb "blue" #\xa0Lade
   Modified Criterion\n\nset output \'nclm_comp.png\'\nplot \'mud_weight.
   txt\' u 2:1 title \'Mud Weight\' w l ls 1, \\\n     \'po_s3.txt\' u 2:1
    title \'Pore Pressure\' w l smooth bezier ls 2,  \\\n     \'po_s3.txt
   \' u 3:1 title \'S3\' w l smooth bezier ls 3, \\\n     \'deterministic.
   txt\' u 2:1 title \'M.C. Criterion\' w l smooth bezier ls 6,\\\n     \'
   deterministic.txt\' u 4:1 title \'M.C. Criterion\' w l smooth bezier ls
    6,\\\n     \'deterministic.txt\' u 5:1 title \'Tensile Failure\' w l
   smooth bezier ls 5,\\\n     \'deterministic.txt\' u 6:1 title \'M.L.
   Criterion\' w l smooth bezier ls 7,\\\n     \'deterministic.txt\' u 7:1
    title \'M.L. Criterion\' w l smooth bezier ls 7,\\\n\n\nset output \'
   determ.png\'\nplot \'mud_weight.txt\' u 2:1 title \'Mud Weight\' w l ls
    1, \\\n     \'po_s3.txt\' u 2:1 title \'Pore Pressure\' w l smooth
   bezier ls 2,  \\\n     \'po_s3.txt\' u 3:1 title \'S3\' w l smooth
   bezier ls 3, \\\n     \'deterministic.txt\' u 5:1 title \'Tensile
   Failure\' w l smooth bezier ls 5,\\\n     \'deterministic.txt\' u 6:1
   title \'Shear Failure\' w l smooth bezier ls 6,\\\n     \'deterministic
   .txt\' u 7:1 title \'Shear Failure\' w l smooth bezier ls 6,\\\n\nset
   output \'deterministic.png\'\nplot \'mud_weight.txt\' u 2:1 title \'Mud
    Weight\' w l ls 1, \\\n     \'po_s3.txt\' u 2:1 title \'Pore Pressure
   \' w l smooth bezier ls 2,  \\\n     \'po_s3.txt\' u 3:1 title \'S3\' w
    l smooth bezier ls 3, \\\n     \'deterministic.txt\' u 5:1 title \'
   Tensile Failure\' w l smooth bezier ls 5,\\\n     \'deterministic.txt\'
    u 6:1 title \'Shear Failure\' w l smooth bezier ls 6,\\\n     \'
   deterministic.txt\' u 7:1 title \'Shear Failure\' w l smooth bezier ls
   6,\\\n     \'deterministic.txt\' u 3:1 title \'P.W. Failure\' w l
   smooth bezier ls 4\n\nset title \'Principal stresses at the borehole
   wall vs Depth\'\nset output \'stresses_bh.png\'\nset key inside right
   bottom\nset rmargin 5\nplot \'stress_bh.txt\' u 2:1 title \'S3\' w l
   smooth bezier ls 2, \\\n    \'stress_bh.txt\' u 3:1 title \'S2\' w l
```

```
   smooth bezier ls 3, \\\n    \'stress_bh.txt\' u 4:1 title \'S1\' w l
   smooth bezier ls 4, \\\n                                          ')


# #  Sensibility  Analysis

# In[193]:


get_ipython().run_cell_magic('gnuplot', '', 'reset\nset terminal pngcairo
   size 800,800 font "Calibri,14"\nset termoption enhanced\n\nset grid\
   nset rmargin 5\nset key inside left bottom\nset key font ",12"\n\n#
   Line style for axes\nset style line 80 lt rgb "#808080"\n\n# Line style
    for grid\nset style line 81 lt 0  # dashed\nset style line 81 lt rgb
   "#808080"  # grey\n\nset grid back linestyle 81\nset border 3 back
   linestyle 80 # Remove border on top and right.  These borders are
   useless and make it harder \n    #to see plotted lines near the border.
    Also, put it in grey; no need for so much emphasis on a border.\nset
   xtics nomirror\nset ytics nomirror\n\nset yrange [11000 : 3000]\n\nset
   xlabel \'Equivalent mudweight (ppg)\'\nset ylabel \'TVD (ft)\'\nset
   title \'Plane of Weakness Failure vs Depth\'\n\nset style line 1 lw 2
   lt 2 dt 3 lc rgb "gray0"  # mud weight\nset style line 2 lw 2 lt 1 lc
   rgb "gray50"  # Pore Pressure\nset style line 3 lw 2 lt 1 lc rgb "dark-
   goldenrod" # S3\nset style line 4 lw 2 lt 1 lc rgb "#A00000" #\xa0Plane
    of Weakness Failure\n\nset output \'pw_fricvar.png\'\nplot \'case_1.
   txt\' u 2:1 title \'0.7 {/Symbol m}_0\' w l smooth bezier lw 2 lc rgb
   \'dark-cyan\' ,\\\n    \'case_2.txt\' u 2:1 title \'0.5 {/Symbol m}_0\'
    w l smooth bezier lw 2 lc rgb \'web-blue\', \\\n    \'case_3.txt\' u
   2:1 title \'0.4 {/Symbol m}_0\' w l smooth bezier lw 2 lc rgb \'navy
   \',\\\n    \'mud_weight.txt\' u 2:1 title \'Mud Weight\' w l ls 1 ,\\\n
       \'po_s3.txt\' u 2:1 title \'Pore Pressure\' w l smooth bezier ls 2,
    \\\n    \'po_s3.txt\' u 3:1 title \'S3\' w l smooth bezier ls 3 \n
                       \nset output \'pw_covar.png\'\nplot \'case_4.txt\'
   u 2:1 title \'0.9 So\' w l smooth bezier lw 2 lc rgb \'coral\', \\\n
    \'case_5.txt\' u 2:1 title \'0.7 So\' w l smooth bezier lw 2 lc rgb \'
   orange\',\\\n    \'case_6.txt\' u 2:1 title \'0.5 So\' w l smooth
   bezier lw 2 lc rgb \'orange-red\',\\\n    \'mud_weight.txt\' u 2:1
   title \'Mud Weight\' w l ls 1 ,\\\n    \'po_s3.txt\' u 2:1 title \'Pore
    Pressure\' w l smooth bezier ls 2, \\\n    \'po_s3.txt\' u 3:1 title
   \'S3\' w l smooth bezier ls 3 \n\n\nreset\nset terminal pngcairo size
   1200,1200 font "Calibri,16"\nset output \'pw_var.png\'\nset key at
   screen 0.5, 0.01 center vertical maxrows 1\nset key font ",12"\nset
   grid\n\n# Line style for axes\nset style line 80 lt rgb "#808080"\n\n#
   Line style for grid\nset style line 81 lt 0  # dashed\nset style line
   81 lt rgb "#808080"  # grey\n\nset grid back linestyle 81\nset border 3
    back linestyle 80 # Remove border on top and right.  These borders are
    useless and make it harder \n    #to see plotted lines near the border
   . Also, put it in grey; no need for so much emphasis on a border.\nset
   xtics nomirror\nset ytics nomirror\n\nset yrange [11000 : 3000]\nset
   xlabel \'Equivalent mudweight (ppg)\'\nset ylabel \'TVD (ft)\'\n\nset
   style line 1 lw 2 lt 2 dt 3 lc rgb "gray0"  # mud weight\nset style
   line 2 lw 2 lt 1 lc rgb "gray50"  # Pore Pressure\nset style line 3 lw
   2 lt 1 lc rgb "dark-goldenrod" # S3\nset style line 4 lw 2 lt 1 lc rgb
   "#A00000" #\xa0Plane of Weakness Failure\n\nset multiplot layout 1,2 #
```

```
   \nunset key\nset title \'Plane of Weakness Failure vs Depth, 0.5 {/
   Symbol m}_0\'\nset size 0.5, 0.475\nset origin 0.5,0.525\nplot \'case_2
   .txt\' u 2:1 notitle w l smooth bezier ls 4,\\\n    \'mud_weight.txt\'
   u 2:1 title \'Mud Weight\' w l ls 1,\\\n    \'po_s3.txt\' u 2:1 title
   \'Pore Pressure\' w l smooth bezier ls 2,\\\n    \'po_s3.txt\' u 3:1
   title \'S3\' w l smooth bezier ls 3 \nset size 0.5, 0.95 # the one to
   the left has to be larger\nset title \'Plane of Weakness Failure vs
   Depth, 0.5 {/Symbol m}_0, 0.9 So\'\nset origin 0,0.05\nset key center
   center \nplot \'po_s3.txt\' u 3:1 title \'S3\' w l smooth bezier ls 3,
   \\\n    \'case_7.txt\' u 2:1 title \'P.W. Failure\' w l smooth bezier
   ls 4,\\\n    \'mud_weight.txt\' u 2:1 title \'Mud Weight\' w l ls 1,\\\
   n    \'po_s3.txt\' u 2:1 title \'Pore Pressure\' w l smooth bezier ls
   2\n    \n    \nset size 0.5, 0.475 \nset origin 0.5,0.05\nunset key\
   nset title \'Plane of Weakness Failure vs Depth, 0.9 So\'\nplot \'
   case_4.txt\' u 2:1 title \'0.9 Co\' w l smooth bezier ls 4,\\\n    \'
   mud_weight.txt\' u 2:1 notitle w l ls 1,\\\n    \'po_s3.txt\' u 2:1
   title \'Pore Pressure\' w l smooth bezier ls 2,\\\n    \'po_s3.txt\' u
   3:1 title \'S3\' w l smooth bezier ls 3 \nunset multiplot')
32
33
34 # # Mud Window Varying Bedding Plane Orientation
35
36 # In[8]:
37
38
39 get_ipython().run_cell_magic('gnuplot', '', 'reset\nset terminal pngcairo
   size 800,800 font "Calibri,14"\nset termoption enhanced\n\nset grid\
   nset rmargin 19\nset key at screen 1, graph 1\nset key font ",12"\n\n#
   Line style for axes\nset style line 80 lt rgb "#808080"\n\n# Line style
    for grid\nset style line 81 lt 0  # dashed\nset style line 81 lt rgb
   "#808080"  # grey\n\nset grid back linestyle 81\nset border 3 back
   linestyle 80 # Remove border on top and right.  These borders are
   useless and make it harder \n    #to see plotted lines near the border.
    Also, put it in grey; no need for so much emphasis on a border.\nset
   xtics nomirror\nset ytics nomirror\n\nset yrange [11000 : 3000]\n\nset
   xlabel \'Equivalent mudweight (ppg)\'\nset ylabel \'TVD (ft)\'\nset
   title \'Plane of Weakness Failure With Varying Bedding Plane
   Orientation vs Depth\'\n\nset style line 1 lw 2 lt 2 dt 3 lc rgb "gray0
   "  # mud weight\nset style line 2 lw 2 lt 1 lc rgb "gray50"  # Pore
   Pressure\nset style line 3 lw 2 lt 1 lc rgb "dark-goldenrod" # S3\nset
   style line 4 lw 2 lt 1 lc rgb "#A00000" #\xa0Plane of Weakness Failure\
   nset style line 5 lw 2 lt 1 lc rgb "purple" #\xa0Tensile Failure\nset
   style line 6 lw 2 lt 1 lc rgb "royalblue" #\xa0Navier Coulomb Criterion
   \nset style line 7 lw 2 lt 1 lc rgb "blue" #\xa0Lade Modified Criterion
   \nset style line 8 lw 2 lt 1 lc rgb "#99b3d1ff" #transparent\nset style
    line 9 lw 2 lt 1 lc rgb "#99ffb3b3" #transparent\n\nset output \'
   bp_orientation.png\'\n                          \nplot for [i =
   -100:100:1] \'pw_orientation.txt\' u ($2 + $3*0.01*i):1  w l smooth
   bezier ls 9 notitle, \\\n    \'po_s3.txt\' u 2:1 title \'Pore Pressure
   \' w l smooth bezier ls 2, \\\n    \'po_s3.txt\' u 3:1 title \'S3\' w l
    smooth bezier ls 3 , \\\n    \'deterministic.txt\' u 5:1 title \'
   Tensile Failure\' w l smooth bezier ls 5,\\\n    \'pw_orientation.txt\'
    u 2:1 title \'P.W. Failure\' w l smooth bezier ls 4 \n
          \n                          \n')
```

```
40
41
42 # #  Histograms   and PDFs for Varying Bedding Plane Orientation
43
44 # In[37]:
45
46
47 get_ipython().run_cell_magic('gnuplot', '', 'reset\nset terminal pngcairo
   size 1400,600 font "Calibri,14"\nset style data histogram\nset style
   fill solid border -1\n\nset grid\nset style line 80 lt rgb "#808080"\
   nset style line 81 lt 0  # dashed\nset style line 81 lt rgb "#808080"
   # grey\nset grid back linestyle 81\nset border 3 back linestyle \nset
   xtics nomirror\nset ytics nomirror\nset nokey\nset tmargin 5\n\n#
                                          PLOT 1\nset output \'stats_1.
   png\'\nset multiplot layout 1, 2 title \'Plane of Weakness Failure by
   Varying Bedding Plane Orientation, TVD = 6462 ft\' font ", 20" \n\nset
   title \'Histogram\'\nset xlabel \'Equivalent mudweight (ppg)\'\nset
   ylabel \'Fraction\'\n\nstats \'tvd_1.txt\' u 3\nn= 10 #number of
   intervals\nmin= (STATS_max)\nmax= (STATS_min)\nwidth=(max-min)/n #
   interval width\n#function used to map a value to the intervals\nhist(x,
   width)=width*floor(x/width)+width/2.0\n\nplot \'tvd_1.txt\' using (hist
   ($3, width)):(1.0/STATS_records) smooth freq with boxes linecolor rgb
   "#A00000"\n\nset title "CDF"\nset ylabel \'Fraction\'\nset label 1
   gprintf("Max = %g", STATS_max) at graph 0.05, graph 0.9 font", 16"\nset
    label 2 gprintf("Mean = %g", STATS_mean) at graph 0.05, graph 0.85
   font", 16"\nset label 3 gprintf("Min = %g", STATS_min) at graph 0.05,
   graph 0.8 font", 16"\nset label 4 gprintf("Q1 = %g", STATS_lo_quartile)
    at STATS_lo_quartile + 0.02, 0.25 font", 16"\nset label 5 gprintf("Q3
   = %g", STATS_up_quartile) at STATS_up_quartile - 0.27, 0.75 font", 16"\
   nset label 6 at STATS_lo_quartile, 0.26 "" point pointtype 7 pointsize
   1.5 lc rgb "#A00000" notitle\nset label 7 at STATS_up_quartile, 0.76 ""
    point pointtype 7 pointsize 1.5 lc rgb "#A00000" notitle\nset label 8
   gprintf("SD = %g", STATS_stddev) at graph 0.05, graph 0.75 font", 16"\
   nset arrow from STATS_mean - STATS_stddev, graph 0 to STATS_mean -
   STATS_stddev, graph 1 nohead dt 2 lc rgb "#A00000"\nset arrow from
   STATS_mean + STATS_stddev, graph 0 to STATS_mean + STATS_stddev, graph
   1 nohead dt 2 lc rgb "#A00000"\n\nplot \'tvd_1.txt\' using 3:(.001)
   smooth cnorm linecolor rgb "#A00000"\n\nunset multiplot\nunset for [i
   =1:8] label i\nunset for [i=1:2] arrow i\n\n\n#
                         PLOT 2\nset output \'stats_2.png\'\nstats \'tvd_2
   .txt\' u 3\nn= 10 #number of intervals\nmin= (STATS_max)\nmax= (
   STATS_min)\nwidth=(max-min)/n #interval width\n#function used to map a
   value to the intervals\nhist(x,width)=width*floor(x/width)+width/2.0\
   nset multiplot layout 1, 2 title \'Plane of Weakness Failure by Varying
    Bedding Plane Orientation, TVD = 6939 ft\' font ", 20"\nset title \'
   Histogram\'\nset xlabel \'Equivalent mudweight (ppg)\'\n\nplot \'tvd_2.
   txt\' using (hist($3, width)):(1.0/STATS_records) smooth freq with
   boxes linecolor rgb "#A00000"\n\nset title "CDF"\nset ylabel \'Fraction
   \'\nset label 1 gprintf("Max = %g", STATS_max) at graph 0.1, graph 0.9
   font", 16"\nset label 2 gprintf("Mean = %g", STATS_mean) at graph 0.1,
   graph 0.85 font", 16"\nset label 3 gprintf("Min = %g", STATS_min) at
   graph 0.1, graph 0.8 font", 16"\nset label 4 gprintf("Q1 = %g",
   STATS_lo_quartile) at STATS_lo_quartile + 0.02, 0.25 font", 16"\nset
   label 5 gprintf("Q3 = %g", STATS_up_quartile) at STATS_up_quartile -
```

```
0.28, 0.75 font", 16"\nset label at STATS_lo_quartile, 0.26 "" point
pointtype 7 pointsize 1.5 lc rgb "#A00000" notitle\nset label at
STATS_up_quartile, 0.76 "" point pointtype 7 pointsize 1.5 lc rgb "#
A00000" notitle\nset label 8 gprintf("SD = %g", STATS_stddev) at graph
0.1, graph 0.75 font", 16"\nset arrow from STATS_mean - STATS_stddev,
graph 0 to STATS_mean - STATS_stddev, graph 1 nohead dt 2 lc rgb "#
A00000"\nset arrow from STATS_mean + STATS_stddev, graph 0 to
STATS_mean + STATS_stddev, graph 1 nohead dt 2 lc rgb "#A00000"\n\nplot
 \'tvd_2.txt\' using 3:(.001) smooth cnorm linecolor rgb "#A00000"\
nunset multiplot\nunset for [i=1:8] label i\nunset for [i=1:2] arrow i\
n\n#                                                PLOT 3\nset output \'
stats_3.png\'\nstats \'tvd_3.txt\' u 3\nn= 10 #number of intervals\nmin
= (STATS_max)\nmax= (STATS_min)\nwidth=(max-min)/n #interval width\n#
function used to map a value to the intervals\nhist(x,width)=width*
floor(x/width)+width/2.0\nset multiplot layout 1, 2 title \'Plane of
Weakness Failure by Varying Bedding Plane Orientation, TVD = 7466 ft\'
font ", 20"\nset title \'Histogram\'\nset xlabel \'Equivalent mudweight
 (ppg)\'\n\nplot \'tvd_3.txt\' using (hist($3, width)):(1.0/
STATS_records) smooth freq with boxes linecolor rgb "#A00000"\n\nset
title "CDF"\nset ylabel \'Fraction\'\nset label 1 gprintf("Max = %g",
STATS_max) at graph 0.1, graph 0.9 font", 16"\nset label 2 gprintf("
Mean = %g", STATS_mean) at graph 0.1, graph 0.85 font", 16"\nset label
3 gprintf("Min = %g", STATS_min) at graph 0.1, graph 0.8 font", 16"\
nset label 4 gprintf("Q1 = %g", STATS_lo_quartile) at STATS_lo_quartile
 + 0.03, 0.24 font", 16"\nset label 5 gprintf("Q3 = %g",
STATS_up_quartile) at STATS_up_quartile - 0.34, 0.75 font", 16"\nset
label at STATS_lo_quartile, 0.25 "" point pointtype 7 pointsize 1.5 lc
rgb "#A00000" notitle\nset label at STATS_up_quartile, 0.76 "" point
pointtype 7 pointsize 1.5 lc rgb "#A00000" notitle\nset label 8 gprintf
("SD = %g", STATS_stddev) at graph 0.1, graph 0.75 font", 16"\nset
arrow from STATS_mean - STATS_stddev, graph 0 to STATS_mean -
STATS_stddev, graph 1 nohead dt 2 lc rgb "#A00000"\nset arrow from
STATS_mean + STATS_stddev, graph 0 to STATS_mean + STATS_stddev, graph
1 nohead dt 2 lc rgb "#A00000"\n\nplot \'tvd_3.txt\' using 3:(.001)
smooth cnorm linecolor rgb "#A00000"\nunset multiplot\nunset for [i
=1:8] label i\nunset for [i=1:2] arrow i\n#
                  PLOT 4\nset output \'stats_4.png\'\nstats \'tvd_4.txt
\' u 3\nn= 10 #number of intervals\nmin= (STATS_max)\nmax= (STATS_min)\
nwidth=(max-min)/n #interval width\n#function used to map a value to
the intervals\nhist(x,width)=width*floor(x/width)+width/2.0\nset
multiplot layout 1, 2 title \'Plane of Weakness Failure by Varying
Bedding Plane Orientation, TVD = 7982 ft\' font ", 20"\nset title \'
Histogram\'\nset xlabel \'Equivalent mudweight (ppg)\'\n\nplot \'tvd_4.
txt\' using (hist($3, width)):(1.0/STATS_records) smooth freq with
boxes linecolor rgb "#A00000"\n\nset title "CDF"\nset ylabel \'Fraction
\'\nset label 1 gprintf("Max = %g", STATS_max) at graph 0.1, graph 0.9
font", 16"\nset label 2 gprintf("Mean = %g", STATS_mean) at graph 0.1,
graph 0.85 font", 16"\nset label 3 gprintf("Min = %g", STATS_min) at
graph 0.1, graph 0.8 font", 16"\nset label 4 gprintf("Q1 = %g",
STATS_lo_quartile) at STATS_lo_quartile + 0.03, 0.24 font", 16"\nset
label 5 gprintf("Q3 = %g", STATS_up_quartile) at STATS_up_quartile -
0.45, 0.75 font", 16"\nset label at STATS_lo_quartile, 0.25 "" point
pointtype 7 pointsize 1.5 lc rgb "#A00000" notitle\nset label at
STATS_up_quartile, 0.76 "" point pointtype 7 pointsize 1.5 lc rgb "#
```

```
A00000" notitle\nset label 8 gprintf("SD = %g", STATS_stddev) at graph
0.1, graph 0.75 font", 16"\nset arrow from STATS_mean - STATS_stddev,
graph 0 to STATS_mean - STATS_stddev, graph 1 nohead dt 2 lc rgb "#
A00000"\nset arrow from STATS_mean + STATS_stddev, graph 0 to
STATS_mean + STATS_stddev, graph 1 nohead dt 2 lc rgb "#A00000"\n\nplot
 \'tvd_4.txt\' using 3:(.001) smooth cnorm linecolor rgb "#A00000"\
nunset multiplot\nunset for [i=1:8] label i\nunset for [i=1:2] arrow i\
n#                                                PLOT 5\nset output \'
stats_5.png\'\nstats \'tvd_5.txt\' u 3\nn= 10 #number of intervals\nmin
= (STATS_max)\nmax= (STATS_min)\nwidth=(max-min)/n #interval width\n#
function used to map a value to the intervals\nhist(x,width)=width*
floor(x/width)+width/2.0\nset multiplot layout 1, 2 title \'Plane of
Weakness Failure by Varying Bedding Plane Orientation, TVD = 8517 ft\'
font ", 20"\nset title \'Histogram\'\nset xlabel \'Equivalent mudweight
 (ppg)\'\n\nplot \'tvd_5.txt\' using (hist($3, width)):(1.0/
STATS_records) smooth freq with boxes linecolor rgb "#A00000"\n\nset
title "CDF"\nset ylabel \'Fraction\'\nset label 1 gprintf("Max = %g",
STATS_max) at graph 0.1, graph 0.9 font", 16"\nset label 2 gprintf("
Mean = %g", STATS_mean) at graph 0.1, graph 0.85 font", 16"\nset label
3 gprintf("Min = %g", STATS_min) at graph 0.1, graph 0.8 font", 16"\
nset label 4 gprintf("Q1 = %g", STATS_lo_quartile) at STATS_lo_quartile
 + 0.03, 0.24 font", 16"\nset label 5 gprintf("Q3 = %g",
STATS_up_quartile) at STATS_up_quartile - 0.48, 0.75 font", 16"\nset
label at STATS_lo_quartile, 0.25 "" point pointtype 7 pointsize 1.5 lc
rgb "#A00000" notitle\nset label at STATS_up_quartile, 0.76 "" point
pointtype 7 pointsize 1.5 lc rgb "#A00000" notitle\nset label 8 gprintf
("SD = %g", STATS_stddev) at graph 0.1, graph 0.75 font", 16"\nset
arrow from STATS_mean - STATS_stddev, graph 0 to STATS_mean -
STATS_stddev, graph 1 nohead dt 2 lc rgb "#A00000"\nset arrow from
STATS_mean + STATS_stddev, graph 0 to STATS_mean + STATS_stddev, graph
1 nohead dt 2 lc rgb "#A00000"\n\nplot \'tvd_5.txt\' using 3:(.001)
smooth cnorm linecolor rgb "#A00000"\nunset multiplot\nunset for [i
=1:8] label i\nunset for [i=1:2] arrow i\n#
                  PLOT 6\nset output \'stats_6.png\'\nstats \'tvd_6.txt
\' u 3\nn= 10 #number of intervals\nmin= (STATS_max)\nmax= (STATS_min)\
nwidth=(max-min)/n #interval width\n#function used to map a value to
the intervals\nhist(x,width)=width*floor(x/width)+width/2.0\nset
multiplot layout 1, 2 title \'Plane of Weakness Failure by Varying
Bedding Plane Orientation, TVD = 9015 ft\' font ", 20"\nset title \'
Histogram\'\nset xlabel \'Equivalent mudweight (ppg)\'\n\nplot \'tvd_6.
txt\' using (hist($3, width)):(1.0/STATS_records) smooth freq with
boxes linecolor rgb "#A00000"\n\nset title "CDF"\nset ylabel \'Fraction
\'\nset label 1 gprintf("Max = %g", STATS_max) at graph 0.1, graph 0.9
font", 16"\nset label 2 gprintf("Mean = %g", STATS_mean) at graph 0.1,
graph 0.85 font", 16"\nset label 3 gprintf("Min = %g", STATS_min) at
graph 0.1, graph 0.8 font", 16"\nset label 4 gprintf("Q1 = %g",
STATS_lo_quartile) at STATS_lo_quartile + 0.03, 0.24 font", 16"\nset
label 5 gprintf("Q3 = %g", STATS_up_quartile) at STATS_up_quartile -
0.6, 0.75 font", 16"\nset label at STATS_lo_quartile, 0.25 "" point
pointtype 7 pointsize 1.5 lc rgb "#A00000" notitle\nset label at
STATS_up_quartile, 0.76 "" point pointtype 7 pointsize 1.5 lc rgb "#
A00000" notitle\nset label 8 gprintf("SD = %g", STATS_stddev) at graph
0.1, graph 0.75 font", 16"\nset arrow from STATS_mean - STATS_stddev,
graph 0 to STATS_mean - STATS_stddev, graph 1 nohead dt 2 lc rgb "#
```

```
A00000"\nset arrow from STATS_mean + STATS_stddev, graph 0 to
STATS_mean + STATS_stddev, graph 1 nohead dt 2 lc rgb "#A00000"\n\nplot
 \'tvd_6.txt\' using 3:(.001) smooth cnorm linecolor rgb "#A00000"\
nunset multiplot\nunset for [i=1:8] label i\nunset for [i=1:2] arrow i\
n#                                                  PLOT 7\nset output \'
stats_7.png\'\nstats \'tvd_7.txt\' u 3\nn= 10 #number of intervals\nmin
= (STATS_max)\nmax= (STATS_min)\nwidth=(max-min)/n #interval width\n#
function used to map a value to the intervals\nhist(x,width)=width*
floor(x/width)+width/2.0\nset multiplot layout 1, 2 title \'Plane of
Weakness Failure by Varying Bedding Plane Orientation, TVD = 9501 ft\'
font ", 20"\nset title \'Histogram\'\nset xlabel \'Equivalent mudweight
 (ppg)\'\n\nplot \'tvd_7.txt\' using (hist($3, width)):(1.0/
STATS_records) smooth freq with boxes linecolor rgb "#A00000"\n\nset
title "CDF"\nset ylabel \'Fraction\'\nset label 1 gprintf("Max = %g",
STATS_max) at graph 0.1, graph 0.9 font", 16"\nset label 2 gprintf("
Mean = %g", STATS_mean) at graph 0.1, graph 0.85 font", 16"\nset label
3 gprintf("Min = %g", STATS_min) at graph 0.1, graph 0.8 font", 16"\
nset label 4 gprintf("Q1 = %g", STATS_lo_quartile) at STATS_lo_quartile
 + 0.03, 0.24 font", 16"\nset label 5 gprintf("Q3 = %g",
STATS_up_quartile) at STATS_up_quartile - 0.6, 0.75 font", 16"\nset
label at STATS_lo_quartile, 0.25 "" point pointtype 7 pointsize 1.5 lc
rgb "#A00000" notitle\nset label at STATS_up_quartile, 0.76 "" point
pointtype 7 pointsize 1.5 lc rgb "#A00000" notitle\nset label 8 gprintf
("SD = %g", STATS_stddev) at graph 0.1, graph 0.75 font", 16"\nset
arrow from STATS_mean - STATS_stddev, graph 0 to STATS_mean -
STATS_stddev, graph 0.72 nohead dt 2 lc rgb "#A00000"\nset arrow from
STATS_mean + STATS_stddev, graph 0 to STATS_mean + STATS_stddev, graph
1 nohead dt 2 lc rgb "#A00000"\n\nplot \'tvd_7.txt\' using 3:(.001)
smooth cnorm linecolor rgb "#A00000"\nunset multiplot\nunset for [i
=1:8] label i\nunset for [i=1:2] arrow i\n#
                  PLOT 8\nset output \'stats_8.png\'\nstats \'tvd_8.txt
\' u 3\nn= 10 #number of intervals\nmin= (STATS_max)\nmax= (STATS_min)\
nwidth=(max-min)/n #interval width\n#function used to map a value to
the intervals\nhist(x,width)=width*floor(x/width)+width/2.0\nset
multiplot layout 1, 2 title \'Plane of Weakness Failure by Varying
Bedding Plane Orientation, TVD = 9983 ft\' font ", 20"\nset title \'
Histogram\'\nset xlabel \'Equivalent mudweight (ppg)\'\n\nplot \'tvd_8.
txt\' using (hist($3, width)):(1.0/STATS_records) smooth freq with
boxes linecolor rgb "#A00000"\n\nset title "CDF"\nset ylabel \'Fraction
\'\nset label 1 gprintf("Max = %g", STATS_max) at graph 0.1, graph 0.9
font", 16"\nset label 2 gprintf("Mean = %g", STATS_mean) at graph 0.1,
graph 0.85 font", 16"\nset label 3 gprintf("Min = %g", STATS_min) at
graph 0.1, graph 0.8 font", 16"\nset label 4 gprintf("Q1 = %g",
STATS_lo_quartile) at STATS_lo_quartile + 0.03, 0.24 font", 16"\nset
label 5 gprintf("Q3 = %g", STATS_up_quartile) at STATS_up_quartile -
0.53, 0.75 font", 16"\nset label at STATS_lo_quartile, 0.25 "" point
pointtype 7 pointsize 1.5 lc rgb "#A00000" notitle\nset label at
STATS_up_quartile, 0.76 "" point pointtype 7 pointsize 1.5 lc rgb "#
A00000" notitle\nset label 8 gprintf("SD = %g", STATS_stddev) at graph
0.1, graph 0.75 font", 16"\nset arrow from STATS_mean - STATS_stddev,
graph 0 to STATS_mean - STATS_stddev, graph 0.72 nohead dt 2 lc rgb "#
A00000"\nset arrow from STATS_mean + STATS_stddev, graph 0 to
STATS_mean + STATS_stddev, graph 1 nohead dt 2 lc rgb "#A00000"\n\nplot
 \'tvd_8.txt\' using 3:(.001) smooth cnorm linecolor rgb "#A00000"\
```

```
nunset multiplot\nunset for [i=1:8] label i\nunset for [i=1:2] arrow i\
n#                                                  PLOT 9\nset output \'
stats_9.png\'\nstats \'tvd_9.txt\' u 3\nn= 10 #number of intervals\nmin
= (STATS_max)\nmax= (STATS_min)\nwidth=(max-min)/n #interval width\n#
function used to map a value to the intervals\nhist(x,width)=width*
floor(x/width)+width/2.0\nset multiplot layout 1, 2 title \'Plane of
Weakness Failure by Varying Bedding Plane Orientation, TVD = 10344 ft\'
 font ", 20"\nset title \'Histogram\'\nset xlabel \'Equivalent
mudweight (ppg)\'\n\nplot \'tvd_9.txt\' using (hist($3, width)):(1.0/
STATS_records) smooth freq with boxes linecolor rgb "#A00000"\n\nset
title "CDF"\nset ylabel \'Fraction\'\nset label 1 gprintf("Max = %g",
STATS_max) at graph 0.1, graph 0.9 font", 16"\nset label 2 gprintf("
Mean = %g", STATS_mean) at graph 0.1, graph 0.85 font", 16"\nset label
3 gprintf("Min = %g", STATS_min) at graph 0.1, graph 0.8 font", 16"\
nset label 4 gprintf("Q1 = %g", STATS_lo_quartile) at STATS_lo_quartile
 + 0.03, 0.24 font", 16"\nset label 5 gprintf("Q3 = %g",
STATS_up_quartile) at STATS_up_quartile - 0.53, 0.75 font", 16"\nset
label at STATS_lo_quartile, 0.25 "" point pointtype 7 pointsize 1.5 lc
rgb "#A00000" notitle\nset label at STATS_up_quartile, 0.76 "" point
pointtype 7 pointsize 1.5 lc rgb "#A00000" notitle\nset label 8 gprintf
("SD = %g", STATS_stddev) at graph 0.1, graph 0.75 font", 16"\nset
arrow from STATS_mean - STATS_stddev, graph 0 to STATS_mean -
STATS_stddev, graph 0.72 nohead dt 2 lc rgb "#A00000"\nset arrow from
STATS_mean + STATS_stddev, graph 0 to STATS_mean + STATS_stddev, graph
1 nohead dt 2 lc rgb "#A00000"\n\nplot \'tvd_9.txt\' using 3:(.001)
smooth cnorm linecolor rgb "#A00000"\nunset multiplot\nunset for [i
=1:8] label i\nunset for [i=1:2] arrow i\n#
                  PLOT 10\nset output \'stats_10.png\'\nstats \'tvd_10.
txt\' u 3\nn= 10 #number of intervals\nmin= (STATS_max)\nmax= (
STATS_min)\nwidth=(max-min)/n #interval width\n#function used to map a
value to the intervals\nhist(x,width)=width*floor(x/width)+width/2.0\
nset multiplot layout 1, 2 title \'Plane of Weakness Failure by Varying
 Bedding Plane Orientation, TVD = 10495 ft\' font ", 20"\nset title \'
Histogram\'\nset xlabel \'Equivalent mudweight (ppg)\'\n\nplot \'tvd_10
.txt\' using (hist($3, width)):(1.0/STATS_records) smooth freq with
boxes linecolor rgb "#A00000"\n\nset title "CDF"\nset ylabel \'Fraction
\'\nset label 1 gprintf("Max = %g", STATS_max) at graph 0.1, graph 0.9
font", 16"\nset label 2 gprintf("Mean = %g", STATS_mean) at graph 0.1,
graph 0.85 font", 16"\nset label 3 gprintf("Min = %g", STATS_min) at
graph 0.1, graph 0.8 font", 16"\nset label 4 gprintf("Q1 = %g",
STATS_lo_quartile) at STATS_lo_quartile + 0.03, 0.24 font", 16"\nset
label 5 gprintf("Q3 = %g", STATS_up_quartile) at STATS_up_quartile -
0.53, 0.75 font", 16"\nset label at STATS_lo_quartile, 0.25 "" point
pointtype 7 pointsize 1.5 lc rgb "#A00000" notitle\nset label at
STATS_up_quartile, 0.76 "" point pointtype 7 pointsize 1.5 lc rgb "#
A00000" notitle\nset label 8 gprintf("SD = %g", STATS_stddev) at graph
0.1, graph 0.75 font", 16"\nset arrow from STATS_mean - STATS_stddev,
graph 0 to STATS_mean - STATS_stddev, graph 0.72 nohead dt 2 lc rgb "#
A00000"\nset arrow from STATS_mean + STATS_stddev, graph 0 to
STATS_mean + STATS_stddev, graph 1 nohead dt 2 lc rgb "#A00000"\n\nplot
 \'tvd_10.txt\' using 3:(.001) smooth cnorm linecolor rgb "#A00000"\
nunset multiplot\nunset for [i=1:8] label i\nunset for [i=1:2] arrow i'
)
```

```
49
50 # # Stochastic Analysis
51
52 # In[197]:
53
54
55 get_ipython().run_cell_magic('gnuplot', '', 'reset\nset terminal pngcairo
   size 800,800 font "Calibri,14"\nset termoption enhanced\n\nset grid\
   nset rmargin 19\nset key at screen 1, graph 1\nset key font ",12"\n\n#
   Line style for axes\nset style line 80 lt rgb "#808080"\n\n# Line style
    for grid\nset style line 81 lt 0  # dashed\nset style line 81 lt rgb
   "#808080"  # grey\n\nset grid back linestyle 81\nset border 3 back
   linestyle 80 # Remove border on top and right.  These borders are
   useless and make it harder \n    #to see plotted lines near the border.
    Also, put it in grey; no need for so much emphasis on a border.\nset
   xtics nomirror\nset ytics nomirror\n\nset yrange [11000 : 3000]\n\nset
   xlabel \'Equivalent mudweight (ppg)\'\nset ylabel \'TVD (ft)\'\nset
   title \' Failure modes vs Depth\'\n\nset style line 1 lw 2 lt 2 dt 3 lc
    rgb "gray0"  # mud weight\nset style line 2 lw 2 lt 1 lc rgb "gray50"
    # Pore Pressure\nset style line 3 lw 2 lt 1 lc rgb "dark-goldenrod" #
   S3\nset style line 4 lw 2 lt 1 lc rgb "#A00000" #\xa0Plane of Weakness
   Failure\nset style line 5 lw 2 lt 1 lc rgb "purple" #\xa0Tensile
   Failure\nset style line 6 lw 2 lt 1 lc rgb "royalblue" #\xa0Navier
   Coulomb Criterion\nset style line 7 lw 2 lt 1 lc rgb "blue" #\xa0Lade
   Modified Criterion\nset style line 8 lw 2 lt 1 lc rgb "#99b3d1ff" #
   transparent\nset style line 9 lw 2 lt 1 lc rgb "#99ffb3b3" #transparent
   \n\nset output \'stochastic.png\'\nplot for [i = -100:100:1] \'pw.txt\'
    u ($2 + $3*0.01*i):1  w l smooth bezier ls 9 notitle, \\\n     for [i
   = -100:100:1] \'stlade.txt\' u (($2 + $3*0.01*i)* 8.33 /(0.098 * $1 *
   0.3048)):1  w l ls 8 smooth bezier notitle, \\\n    \'po_s3.txt\' u 2:1
    title \'Pore Pressure\' w l smooth bezier ls 2, \\\n    \'po_s3.txt\'
   u 3:1 title \'S3\' w l smooth bezier ls 3 , \\\n    \'deterministic.txt
   \' u 5:1 title \'Tensile Failure\' w l smooth bezier ls 5,\\\n    \'
   stlade.txt\' u ($2* 8.33/(0.098 * $1 * 0.3048)):1:3 title \'Shear
   Failure\' w l smooth bezier ls 7,\\\n    \'deterministic.txt\' u 7:1
   title \'Shear Failure\' w l smooth bezier ls 7,\\\n    \'pw.txt\' u 2:1
    title \'P.W. Failure\' w l smooth bezier ls 4 , \\\n    \'mud_weight.
   txt\' u 2:1 title \'Mud Weight\' w l ls 1      \n    #smooth bezier\n
     #smooth sbezier\nset output \'sto_cases.png\'   \nplot for [i =
   -100:100:1] \'pw.txt\' u ($2 + $3*0.01*i):1  w l smooth bezier ls 9
   notitle, \\\n    \'pw.txt\' u 2:1 title \'P.W. Failure\' w l smooth
   bezier ls 4, \\\n    \'pw.txt\' u 4:1 title \'P25\' w l smooth bezier
   lc rgb "#3366cc" lw 2, \\\n    \'pw.txt\' u 5:1 title \'P75\' w l
   smooth bezier lc rgb "#006600"lw 2,\\\n    \'pw.txt\' u 6:1 title \'P90
   \' w l smooth bezier lc rgb "#ff0000" lw 2,\\\n    \'po_s3.txt\' u 2:1
   title \'Pore Pressure\' w l smooth bezier ls 2,  \\\n    \'po_s3.txt\'
   u 3:1 title \'S3\' w l smooth bezier ls 3, \\\n    \'mud_weight.txt\' u
    2:1 title \'Mud Weight\' w l ls 1\n        \n    ')
56
57
58 # # Strength Multiplot
59
60 # In[203]:
61
```

```
62
63 get_ipython().run_cell_magic('gnuplot', '', 'reset\nset terminal pngcairo
   size 1400,600 font "Calibri,14"\n\nset grid\nset style line 80 lt rgb
   "#808080"\nset style line 81 lt 0  # dashed\nset style line 81 lt rgb
   "#808080"  # grey\nset grid back linestyle 81\nset border 3 back
   linestyle \nset xtics nomirror\nset ytics nomirror\nset nokey\nset
   tmargin 5\nset yrange [11000 : 3000]\n\nset style line 4 lw 2 lt 1 lc
   rgb "#A00000" #\xa0Cohesion\nset style line 6 lw 2 lt 1 lc rgb "
   royalblue" #\xa0Tensile\n\nset output \'strength.png\'\nset multiplot
   layout 1, 2 title \'Cohesion and Tensile Strength vs Depth\' font ",
   20" \n\nset title \'Cohesion\'\nset xlabel \'Equivalent mudweight (ppg)
   \'\nset ylabel \'TVD (ft)\'\n\nplot \'strength.txt\' u 3:1 title \'
   Matrix Cohesion\' w l smooth bezier ls 4\n\nset title "Tensile Strength
   "\nset ylabel \'TVD (ft)\'\n\nplot \'strength.txt\' u 2:1 title \'
   Tensile Strength\' w l smooth bezier ls 6\n\nunset multiplot')
64
65
66 # # Stress Plot
67
68 # In[5]:
69
70
71 get_ipython().run_cell_magic('gnuplot', '', 'reset\nset terminal pngcairo
   size 800,800 font "Calibri,14"\nset termoption enhanced\n\nset grid\
   nset rmargin 19\nset key at screen 1, graph 1\nset key font ",12"\n\n#
   Line style for axes\nset style line 80 lt rgb "#808080"\n\n# Line style
    for grid\nset style line 81 lt 0  # dashed\nset style line 81 lt rgb
   "#808080"  # grey\n\nset grid back linestyle 81\nset border 3 back
   linestyle 80 # Remove border on top and right.  These borders are
   useless and make it harder \n    #to see plotted lines near the border.
    Also, put it in grey; no need for so much emphasis on a border.\nset
   xtics nomirror\nset ytics nomirror\n\nset yrange [11000 : 3000]\n\nset
   xlabel \'Equivalent mudweight (ppg)\'\nset ylabel \'TVD (ft)\'\nset
   title \' Stresses vs Depth\'\n\nset style line 1 lw 2 lt 1 lc rgb
   "#0000ff"  # mud weight\nset style line 2 lw 2 lt 1 lc rgb "#666699"  #
    Pore Pressure\nset style line 3 lw 2 lt 1 lc rgb "#cc0000" # S3\nset
   style line 4 lw 2 lt 1 lc rgb "#00ccff" #\xa0Plane of Weakness Failure\
   nset style line 5 lw 2 lt 1 lc rgb "#0099ff" #\xa0Tensile Failure\nset
   style line 6 lw 2 lt 1 lc rgb "#0066ff" #\xa0Navier Coulomb Criterion\n
   \nset output \'stress_field.png\'\nplot \'stresses_b.txt\' u 2:1 title
   \'Sx\' w l smooth bezier ls 1, \\\n     \'stresses_b.txt\' u 3:1 title
   \'Sy\' w l smooth bezier ls 2,  \\\n     \'stresses_b.txt\' u 4:1 title
    \'Sz\' w l smooth bezier ls 3, \\\n     \'stresses_b.txt\' u 5:1 title
    \'Txy\' w l smooth bezier ls 4,\\\n     \'stresses_b.txt\' u 6:1 title
    \'Txz\' w l smooth bezier ls 5,\\\n     \'stresses_b.txt\' u 7:1 title
    \'Tyz\' w l smooth bezier ls 6')
72
73
74 # In[ ]:
```