Methods of Pore Pressure Detection from Real-time Drilling Data

Sindre Stunes

Earth Sciences and Petroleum Engineering
Submission date: June 2012
Supervisor: Pål Skalle, IPT

Norwegian University of Science and Technology
Department of Petroleum Engineering and Applied Geophysics

Summary

The knowledge of formation pore pressure, and how it changes throughout the length of a well, is crucial in terms of maintaining control of the wellbore. Failure to recognize deviations from the expected pressures can lead to problems and instabilities, which increases drilling costs. A worst case scenario may lead to loss of an entire well section. Thus maintaining a real-time knowledge of the formation pore pressure is beneficial regarding both the cost and the safety of a drilling operation.

In this thesis multiple methods of pore pressure detection have been implemented in a Matlab program, which is used for testing with recorded real-time drilling data of a well, provided by IPT. The methods chosen were the Zamora and Eaton methods, both based on utilization of the d_c-exponent, and the Bourgoyne-Young drilling model. The program has calculated pore pressure gradients based on each of these methods. In turn these results have been compared with the pore pressure presented in a final well report provided alongside the drilling data. This forms a basis for evaluation of each methods accuracy and applicability with use of this kind of drilling data.

The results show that all three methods are able to produce a pore pressure gradient which is partly in compliance with the values provided in the final well report. However, the accuracy of the calculated results is not sufficient to be used to detect pore pressure with the desired precision. This may in part be caused by a lack of gamma ray data, which would have provided a more reliable selection of data. The addition of gamma ray as an input parameter should be of priority in any future developments. The most accurate result was calculated using the Bourgoyne-Young drilling model.

Sammendrag

Kunnskap om poretrykket i sedimentære formasjoner, og hvordan dette endres nedover i grunnen, er helt sentralt for å kunne kontrollere brønnen. Dersom ikke variasjoner forventet trykk oppdages kan det forårsake flere problemer, som igjen vil øke kostnadene knyttet til å bore brønnen. I verste fall vil dette kunne føre til tap av hele brønnseksjoner. Som følge av dette er å opprettholde sanntids kjennskap til trykket i formasjonen meget gunstig, både med tanke på kostnadene og sikkerheten knyttet til boreoperasjoner.

I denne oppgaven er flere metoder for bestemmelse av boretrykk implementert i et Matlabprogram, som igjen er benyttet for testing på lagrede sanntids boredata fra en brønnoperasjon. Metodene som ble valgt var Zamora og Eatons metoder, begge basert på bruk av d_c-eksponenten, og den matematiske boremodellen til Bourgoyne og Young. Programmet har beregnet en poretrykksgradient basert på hver av disse metodene. Disse er deretter sammenlignet med gradienten som ble presentert brønnens sluttrapport. Denne sammenligningen danner en basis for å vurdere hver av metodenes presisjon.

Resultatene viser at alle de benyttede metodene er i stand til å beregne en poretrykksgradient som til en viss grad er i samsvar med den oppgitte. Det er likevel et såpass betydelig avvik enkelte steder, at man ikke kan si at ønsket presjon for trykkberegningene er oppnådd. Dette kan til en viss grad skyldes manglende verdier fra gammastrålingslogger, som kunne ha gitt en bedre utvelgelse av data for bergegingene. Dersom programmet skal videreutvikles bør disse verdiene inkluderes. Det beste resultatet ble oppnådd ved bruk av Bourgoyne og Youngs matematiske modell for boring.

Preface

This Master's thesis was written at the Department of Petroleum Engineering and Applied Geophysics at the Norwegian University of Science and Technology, NTNU.

I would like to thank my academic supervisor, Associate Professor Pål Skalle, for his advice and guidance throughout the process. In addition I would like to thank IPT and Statoil for providing the real-time drilling data that have been used for the testing and analysis performed in this thesis.

The author of this work hereby declares that the work in this thesis is made independently and in accordance to the rules set down by Examination regulations at the Norwegian University of Science and Technology (NTNU), Trondheim.

____

Sindre Stunes, Trondheim 2012

Table of Contents

Summary i

Sammendrag iii

Preface v

Table of Contents 1

List of figures 5

List of tables 5

1 Introduction 7

2 Published material 8

2.1 Abnormal pore pressure 8

2.2 Methods of pore pressure detection 9

2.2.1 d_c-exponent 9

2.2.2 Zamora's method 10

2.2.3 Eaton's method 11

2.2.4 Bourgoyne-Young drilling model 12

2.2.5 Method for all sedimentary lithologies 14

2.3 Parameters influencing drilling performance 14

2.3.1 Lithology 14

2.3.2 Differential pressure 15

2.3.3 Drilling bits 15

3 The well and provided data 16

3.1 Lithology 17

3.1.1 Nordland Group 17

3.1.2 Hordaland Group 17

3.1.3 Rogaland Group 18

3.1.4 Shetland Group 18
3.2 Pressure gradients 18
3.2.1 Normal pressure gradient 20
4 Development of on-line tool for pore pressure detection 21
4.1 The models to be tested 21
4.1.1 d_c-exponent methods 21
4.1.2 Bourgoyne-Young drilling model 22
4.2 Importing field data 23
4.3 Flowchart 24
4.3.1 Calculation of ROP data 25
4.3.2 Removal of unwanted data points 25
4.3.3 Averaging the data and creating depth interval between data points 26
4.3.4 Calculating the standard deviation 26
5 Results from data analysis 27
5.1 Printout of input parameters 27
5.1.1 Printout of RPM, WOB and ROP 27
5.1.2 Mud weight and ECD 28
5.2 d_c-exponent trend line 30
5.3 Zamora pressure estimation 32
5.4 Eaton pressure estimation 34
5.5 Bourgoyne-Young pressure estimation 36
5.6 Standard deviations 38
6 Discussion 39
6.1 Weaknesses and limitations 39
6.1.1 General 39
6.1.2 d_c-exponent methods 40

6.1.3 Bourgoyne-Young drilling model ... 41
6.2 Future improvements ... 42
7 Conclusion ... 43
Nomenclature ... 44
References ... 45
Appendices ... 47
A Additional result plots ... 48
A.1 Field data ... 48
A.1.1 RPM ... 48
A.1.2 WOB ... 50
A.1.3 ROP ... 52
A.1.4 Mud weight and ECD ... 54
A.2 d_c-exponent ... 55
A.3 Zamora's method ... 59
A.4 Eaton's method ... 63
A.5 Bourgoyne-Young drilling model ... 67
A.5.1 Drillability ... 67
A.5.2 Pressure gradient ... 69
A.6 Standard deviation ... 70
B Matlab Code ... 73
B.1 Main script ... 73
B.2 Import functions ... 77
B.2.1 Function for importing drilling data ... 77
B.2.2 Import of excel data ... 79
B.3 Data processing functions ... 80
B.3.1 Function to align data ... 80

B.3.2 Function for ROP calculation 81

B.3.3 Function for selection of data 82

B.3.4 Function to average multiple data points 86

B.3.5 Function to create depth interval between data points 87

B.4 Calculation functions 89

B.4.1 Function to create d_c-trend line coefficients 89

B.4.2 Function to calculate d_c-exponent 90

B.4.3 Function to calculate pore pressure gradient from Eaton’s method 91

B.4.4 Function to calculate pore pressure gradient from Zamora’s method 91

B.4.5 Function to calculate factors of Bourgoyne-Young equation 92

B.4.6 Function to calculate linear drillability trend for a desired interval 93

B.4.7 Function to calculate linear drillability trend for each separate lithology 94

B.4.8 Function to calculate pressure gradients from Bourgoyne-Young equation 98

B.4.9 Function to calculate standard deviation 99

B.5 Output functions 100

B.5.1 Various plot functions 100

B.5.2 Function to plot standard deviation 101

List of figures

Figure 3.1: Wells in the same area 16
Figure 3.2: Pressure gradients from the final well report 19
Figure 3.3: Pressure gradients imported to excel from 20
Figure 4.1: Flowchart for the main program 24
Figure 5.1: Comparison of mud weight parameter of different data sources 29
Figure 5.2: d_c-exponent plot from Matlab program 31
Figure 5.3: Zamora pressure gradient 33
Figure 5.4: Eaton pressure gradient 35
Figure 5.5: Bourgoyne-Young pressure gradient 37

List of tables

Table 2-1: Average values of Bourgoyne-Young drilling coefficients 13
Table 3-1: The casing intervals of the well 16
Table 3-2: The different lithological formations of the well 17
Table 5-1: The slopes of the different d_c-exponent trend lines 30
Table 5-2: Standard deviation 38

1 Introduction

Knowledge of the pore pressure in the various zones is critical in terms of controlling the process while drilling a well. Bottom-hole pressure deviating from the expected, or normal, pressure gradients may cause various problems and instabilities. Kicks and loss of control of the well are the most critical problems that may occur, and can lead to a blowout or loss of the section if not handled properly. Even when the problems are properly handled, such events still require valuable time for restoring the situation back to normal, thus increasing the cost of drilling. Ideally, maintaining a real-time knowledge of the formation pressure may minimize the occurrence of some of the events, making drilling more efficient. Such knowledge may serve as an early kick-warning tool and will lead to avoidance or minimized occurrence of kick incidents. The efficiency of most well control actions rely on applying the proper measures as quickly as possible after the initiation of the event.

Availability of real-time data from drilling projects show increasing trend caused by new technology and better data processing capabilities. The purpose of this thesis is to analyze real-time data acquired from a previous well, trying to detect the pore pressure in the formation as drilling progresses. In order to accomplish this, a number of methods for estimation of pore pressure will be implemented in a Matlab program.

A data package containing recorded real-time data from two North Sea wells has been provided by IPT. These data will provide a foundation for testing and evaluation of the chosen methods, and their implementation in the program to be created. The results produced by each method will be compared both with respect to each other, but also compared with the results presented by the operating company in a final well report. This will yield a good foundation for identification of the most suitable method of pore pressure detection, as well as for evaluation of the accuracy of the methods.

This thesis is a continuation and expansion of a student project written in the fall of 2011. The project utilized the d_c-exponent plot in order to estimate at which depth a pore pressure increase occurred. As this is deemed relevant also in this thesis, certain parts of the previous project have been incorporated here.

2 Published material

2.1 Abnormal pore pressure

The majority of this sub chapter is copied from a previous project (Stunes, 2011).

Formation pore pressure is divided into the three categories *normal*, *abnormal* and *subnormal* formation pressure. The term normal formation pressure describes the situation where formation pressure is approximately equal to the theoretical hydrostatic pressure of a given vertical depth. Abnormal and subnormal formation pressures represent pressures of respectively higher or lower values than this normal situation (Bourgoyne et.al., 1986). In the North Sea the normal formation pressure gradient is considered to be 0.452 psi/ft, or 1.044 kg/m^3 when presented as an equivalent water density (Bourgoyne et.al., 1986).

Abnormal formation pressures are found in many sedimentary basins in the world, and can have different origins. Common to all mechanisms providing overpressure is the requirement of a seal to contain the higher pressure values. Five main mechanisms of overpressure can be listed as the following (Yassir & Bell, 1996):

- Rapid loading and undercompaction, where a seal prohibits the dissipation of pore fluids as the sediments are buried and compacted. This will result in an abnormally high pore pressure compared with the burial depth, increasing with the amount of load provided by overlying sediments, as long as the seal stay intact.
- Tectonical movements and shear deformations may create overpressures in originally normal pressured zones.
- In clay rich sediments, where a transformation of montmorillonite to illite takes place, this chemical reaction will release previous intermolecular water as pore water, providing overpressure to the sediments.
- Hydrocarbon generation can lead to overpressure, as a biochemical process in deposited organic materials is capable of producing substantial volumes of methane gas.
- If completely isolated, and the volume of the sediments are kept constant, increasing temperature with increased burial depth may also cause abnormal formation pressures.

2.2 Methods of pore pressure detection

Methods of evaluating abnormal pore pressures are separated in two categories, prediction methods and detection methods. The prediction methods normally use data obtained from seismic surveys, offset well logs and well history. Detection methods traditionally utilize drilling parameters and well log information obtained during the actual drilling of a well (Yoshida, 1996). This chapter will present some of the methods that are used for pore pressure detection.

2.2.1 d_c-exponent

The majority of this sub chapter is copied from a previous project (Stunes, 2011).

The d_c-exponent method for analyzing formation pore pressure was proposed by Jorden and Shirley in 1966 (Bourgoyne et.al., 1986). This was an attempt to normalize the rate of penetration (ROP) from the Bingham drilling model, with respect to the parameters weight on bit (WOB), rotary speed (RPM) and bit diameter (d_{bit}). The purpose was to investigate the proposed relationship between the rate of penetration, and the differential pressure existing between the formation pore pressure and the hydrostatic pressure column in the wellbore (Jorden & Shirley, 1966). The knowledge of this relationship would make it possible to predict changes in the pore pressure with respect to the obtained drilling data. Starting with the Bingham drilling model, this resulted in the calculation of a d-exponent, as shown in the equations below (Bourgoyne et.al., 1986):

$$ROP = K \times RPM \left(\frac{WOB}{d_{bit}}\right)^{d} \tag{2.1}$$

Rearranged by Jorden & Shirley (Jorden & Shirley, 1966):

$$d = \frac{\log\left(\frac{ROP}{60\,RPM}\right)}{\log\left(\frac{12\,WOB}{10^{3}\,d_{bit}}\right)} \tag{2.2}$$

In the latter equation the term K, representing the formation drillability factor of the Bingham drilling equation (2.1) has been given a constant value. This is done assuming the variations in rock properties of the formations to be drilled will be negligible (Bourgoyne et.al., 1986).

The d-exponent equation (2.2) can then be utilized to identify when entering a transition zone going from a normal pressured zone and into an abnormal pressured zone (Bourgoyne et.al., 1986). This is done by acquiring data from formations assumed to have a normal pressure gradient, thus creating a plot showing the d-exponent versus the drilling depth under such conditions. For these formations this plot will typically show an increase of the d-exponent with increasing depth. In formations with abnormal pore pressures, the increased rate of penetration would diminish the increase of the d-exponent, and in some cases also reverse the trend, making the exponent decrease with increasing depth (Bourgoyne et.al., 1986). Comparison of such data would then be used as information as to at which depth the drilling is entering formation zone containing a higher pore pressure.

To be able to also include changes of the mud density to the model, the following equation was proposed, yielding a d_c-exponent corrected with respect to the relationship between the normal pressure gradient and the hydrostatic mud column gradient (Rehm & McClendon, 1971):

$$d_c = d \frac{\rho_{normal}}{\rho_{mud}} \tag{2.3}$$

2.2.2 Zamora's method

In 1972 Zamora proposed that an empirical relation between the d_c-exponent and the pore pressure gradient would be the following (Bourgoyne et.al., 1986):

$$\rho_{pore} = \rho_{normal} \left(\frac{d_{c,normal}}{d_c}\right) \tag{2.4}$$

This was based on using overlay techniques comparing a trend line, created from drilling logs recorded in normal pressured zones, with data from over pressured zones. Zamora recommended using a semi-logarithmic plot, with logarithmic scale for the d_c-exponent, when creating the trend line. The trend lines created was reported not to vary significantly with location or geological age.

2.2.3 Eaton's method

A pressure detection method based on different well logs was presented by Eaton in 1975, where the log results of acoustic velocity, resistivity or d_c-exponent would be used to quantify the formation pore pressure. The method is an improvement of Hottman and Johnson's method of equivalent depth, proposed in 1965. The methods both rely on the widely accepted assumption that overburden pressure is dependent on pore pressure and effective vertical stress, as shown in Terzaghi's equation of 1948 (Eaton, 1975):

$$p_{pore} = p_{overburden} - \sigma_{vertical} \tag{2.5}$$

Originally based only on acoustic velocity and resistivity, it was shown that the d_c-exponent plots would correspond to the resistivity logs of shales, thus enabling the method to be applicable also for use with the d_c-exponent (Eaton, 1975). Eaton's equations are as follows:

$$\rho_{pore} = \rho_{pore} - \left((\rho_{ovb} - \rho_{normal}) \left(\frac{\Delta t_{normal}}{\Delta t}\right)^{3} \right) \tag{2.6}$$

$$\rho_{pore} = \rho_{pore} - \left((\rho_{ovb} - \rho_{normal}) \left(\frac{R}{R_{normal}}\right)^{1,2} \right) \tag{2.7}$$

$$\rho_{pore} = \rho_{pore} - \left((\rho_{ovb} - \rho_{normal}) \left(\frac{d_c}{d_{c,normal}}\right)^{1,2} \right) \tag{2.8}$$

Regardless of which log data to be used for the pressure estimation, they all rely on creating a trend line based on data from a formation with a normal pressure regime, in the addition to knowledge of the overburden pressure gradient and normal pore pressure gradients of the area.

2.2.4 Bourgoyne-Young drilling model

The Bourgoyne-Young drilling model is one of the most comprehensive models used to calculate penetration rate when using rolling cutter bits. It can be used for pore pressure detection, and also various drilling optimization calculations. It consists of eight functions each considering a different drilling variable influencing the ROP (Bourgoyne et.al., 1986):

$$ROP = (f_1)\,(f_2)\,(f_3)\,(f_4)\,\dots\,(f_8) \tag{2.9}$$

Where:

$$f_1 = e^{2{,}303\,a_1} \tag{2.9a}$$

$$f_2 = e^{2{,}303\,a_2\,(10.000-D)} \tag{2.9b}$$

$$f_3 = e^{2{,}303\,a_3\,D^{0{,}69}(\rho_{pore}-9{,}0)} \tag{2.9c}$$

$$f_4 = e^{2{,}303\,a_4\,D\,(\rho_{pore}-\,\rho_{ecd})} \tag{2.9d}$$

$$f_5 = \left(\frac{\left(\frac{WOB}{d_{bit}}\right)-\left(\frac{WOB}{d_{bit}}\right)_t}{4-\left(\frac{WOB}{d_{bit}}\right)_t}\right)^{a_5} \tag{2.9e}$$

$$f_6 = \left(\frac{ROP}{60}\right)^{a_7} \tag{2.9f}$$

$$f_7 = e^{-a_7\,h} \tag{2.9g}$$

$$f_8 = \left(\frac{F_j}{1.000}\right)^{a_8} \tag{2.9h}$$

Here, f_1, often referred to as drillability, mainly represents the effect on penetration rate that is composed from the combination formation strength and bit type. However it also takes in effects of mud type and solids content etc., effects that are not included in any of the other factors (Bourgoyne et.al., 1986).

The factors f_2 and f_3 model the effect of compaction, with f_2 taking in the rock strength increase effect from normal compaction, whilst f_3 model undercompaction in abnormally pressurized zones. The effect of overbalance within the wellbore is modeled by f_4 (Bourgoyne et.al., 1986).

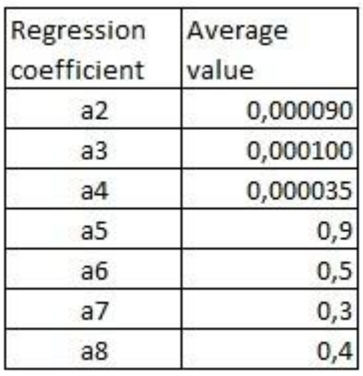
Weight on bit effects is modeled with the function f_5. This function includes a threshold weight on bit factor, i.e. the minimum weight that has to be applied to the bit in order for it to be able to produce cuttings. In soft formations this threshold factor is often neglected. The rotation speed of the drillstring is modeled with f_6. Both f_5 and f_6 are created so that their product should be close to the value 1 under normal drilling conditions (Bourgoyne et.al., 1986).

The functions of f_7 and f_8 model the effect of bit tooth wear and bit hydraulics respectively, with the latter having the jet impact force as its chosen parameter of interest. For f_7, when using tungsten carbide insert bits this effect is often negligible (Bourgoyne et.al., 1986).

2.2.4.1 Drilling constants

The various functions of the Bourgoyne-Young drilling model utilizes several constants, denoted a_1 to a_8, to adapt the model with the specific formation that is to be drilled. These constants have to be estimated from previous drilling data. Bourgoyne and Young proposed using a multiple regression analysis of detailed drilling data in order to obtain these values (Bourgoyne, 1974). The result of such an analysis is presented in table Table 2-1.

Table 2-1: Average values of Bourgoyne-Young drilling coefficients, from shale formations in the U.S. Gulf of Mexico area (Bourgoyne et.al., 1986)



Regression coefficient	Average value
a2	0,000090
a3	0,000100
a4	0,000035
a5	0,9
a6	0,5
a7	0,3
a8	0,4

2.2.5 Method for all sedimentary lithologies

The traditional models of pore pressure are limited to use in shale formations. In order to also be able to estimate pore pressure in formations of other sedimentary lithologies, a new method of quantifying the Terzaghi effective stress law (Equation 2.5) have been proposed. This method is based on use of data from gamma ray logs and porosity data, the latter obtained either from resistivity logs or from γ-γ density sensor logs (Holbrook, 1995).

The log data is used to calculate two compaction coefficients, which in turn is used to determine the maximum effective stress load that a sedimentary formation has borne. Combined with a good estimate of the overburden pressure, these are used to calculate the pore pressure by use of the effective stress law. The petrophysical data needed may be acquired from either wireline logging or MWD tools in the drillstring. When continuous log data is present, the formation pore pressure may be calculated for the complete interval of a well where multiple types of sedimentary lithologies are present. The method have been successfully tested by case studies performed in the North Sea (Holbrook, 1995).

2.3 Parameters influencing drilling performance

The majority of this sub chapter is copied from a previous project (Stunes, 2011).

The rate of penetration as a measure of drilling performance is influenced by a number of parameters. The most important factors have been recognized to include formation characteristics, differential pressure, properties of the drilling fluid, and various bit characteristics (Bourgoyne et.al., 1986).

2.3.1 Lithology

The characteristics of the formation that is being drilled into will have a significant influence on the drilling rate. The most important factor is the elastic limit, and the shear strength provided by the Mohr failure criteria. Other factors are the permeability of the formation, and whether the mineral composition of the rock consists of hard or soft minerals (Bourgoyne et.al., 1986).

2.3.2 Differential pressure

The differential pressure between the wellbore is acting on the chips formed beneath the drill bit, influencing the efficiency of their removal. In an overbalanced drilling situation, i.e. where the hydrostatic mud column of the well exceeds the formation pore pressure, this influence is observed as a chip hold down effect, making the removal of cuttings more demanding, and thus reducing the ROP (Bourgoyne et.al., 1986).

As the drilling progresses into higher pressured formation zones, this hold down effect will diminish as a result of a reduction of the overbalance of the well. This can be observed in logs as an increased ROP (Bourgoyne et.al., 1986). These effects have been verified through field studies, also stating that the sensitivity of the relationship between ROP and differential pressure is increased with the weight applied on the bit (Vidrine & Benit, 1967)

2.3.3 Drilling bits

The bit characteristics influencing ROP includes the type of bit, bit tooth wear and bit hydraulics. Also the operating conditions of the bit, i.e. the RPM and weight applied to the bit have a major influence on the ROP (Bourgoyne et.al., 1986).

The main types of drilling bits are the rolling cutter bits and diamond/PDC bits, which yield a different performance dependent on which type of formation being drilled. For rolling cone bits, the tooth length and cone offset angle are factors determining the aggressiveness of the bit. As drilling progresses wear on the bit teeth will change the bit performance, and tends to decrease the ROP. Bit hydraulics will influence the bit performance as it affects both bottom hole cleaning and cleaning of the bit itself (Bourgoyne et.al., 1986).

3 The well and provided data

IPT and Statoil have provided a data package containing real-time recorded drilling data for several wells in the North Sea. The selected well to be used for analysis was intended as an oil producer, with a horizontal wellbore within its reservoir section. Total length of the wellbore was 4399 m RKB, where the true vertical depth at the end of its horizontal section was 1982 m MSL (Statoil, 2007). Table 2-1 presents the different sections of the well.

Table 3-1: The casing intervals of the well, with corresponding depths. The RKB height above water level is 84,1 m. Water depth is 216,9 m (Statoil, 2007)

Section	Bit size	Casing diameter	Measured depth	True vertical depth
#	[in]	[in]	[m RKB]	[m RKB]
1	-	32"	301	436
2	24"	20"	1514	1308
3	17 1/2"	13 3/8"	2379	1723
4	12 1/4"	9 5/8"	2787	1906
5	8 1/2"	7"	4399	2066

The selected well is placed in an area of the North Sea where many other wells have been drilled previously. Based on data available from the Norwegian petroleum directorate, Figure 3.1 shows an overview of other wells drilled in the area.

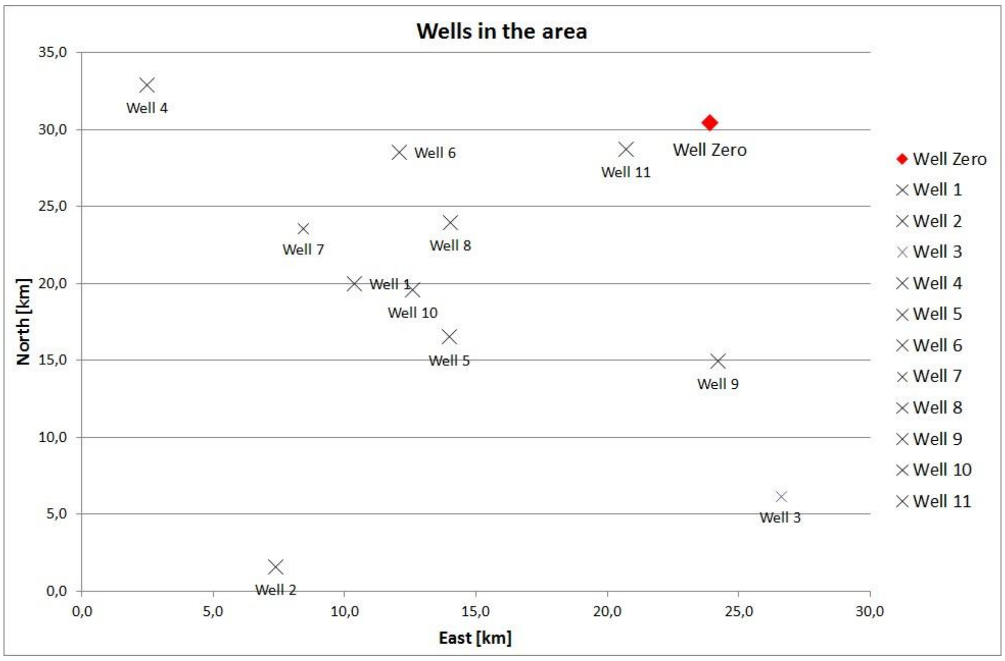
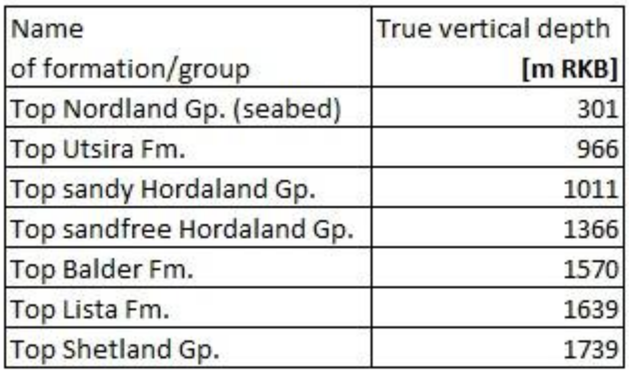


Figure 3.1: Wells in the same area, the selected well is named Well Zero (NPD, 2012).

3.1 Lithology

The wellbore section above the reservoir zone has been constructed within the following lithological formations found in the North Sea:

Table 3-2: The different lithological formations in which the well has been drilled (Statoil, 2007)



Name of formation/group	True vertical depth **[m RKB]**
Top Nordland Gp. (seabed)	301
Top Utsira Fm.	966
Top sandy Hordaland Gp.	1011
Top sandfree Hordaland Gp.	1366
Top Balder Fm.	1570
Top Lista Fm.	1639
Top Shetland Gp.	1739

3.1.1 Nordland Group

The Nordland group of the North Sea mainly consists of marine claystones. The upper part is dominated by unconsolidated clays and sands. In the Viking Graben area the lower part is assigned to the Utsira formation, which is dominated by fine grained marine sandstones Thickness of the group varies from approximately 1000 – 1700 meters (Norlex, 2012).

3.1.2 Hordaland Group

The lithology of the Hordaland Group in the North Sea consists of marine claystones, with interbedded sandstones at various levels. The sandstones are in general fine grained to medium grained. The thickness of the group varies, from a few hundred meters in the northern Viking Graben, to a maximum of 1300 meters in the southern part of the basin (Norlex, 2012).

3.1.3 Rogaland Group

Both the Balder and Lista formations are part of the Rogaland Group in the North Sea. The group generally consists of mudstones and shales, but has also layers of sandstones which may vary in geographical distribution. The thickness varies greatly, from approximately 1000 meters to below 50 meters in some locations (Norlex, 2012).

3.1.4 Shetland Group

The Shetland Group mainly consists of various chalk facies like limestones and marls, but does also have elements consisting of calcareous shales and mudstones. The group thickness ranges from 1000 – 2000 meters in graben areas (Norlex, 2012).

3.2 Pressure gradients

The final well report contains a plot of pressure gradient development throughout the length of the wellbore. Figure 3.2 presents relevant gradients of both formation pore pressure, mud weight and overburden pressure.

Pore pressure can be observed to increase more rapidly than an assumed normal pressure increase from a depth of 1400 m RKB. Also, a high pressured zone, with a reported pore pressure gradient of 1,74 SG, is seen when penetrating the top of the Shetland formation. This section is reported to have been drilled using managed pressure drilling techniques, rather than conventional overbalanced drilling (Statoil, 2007).

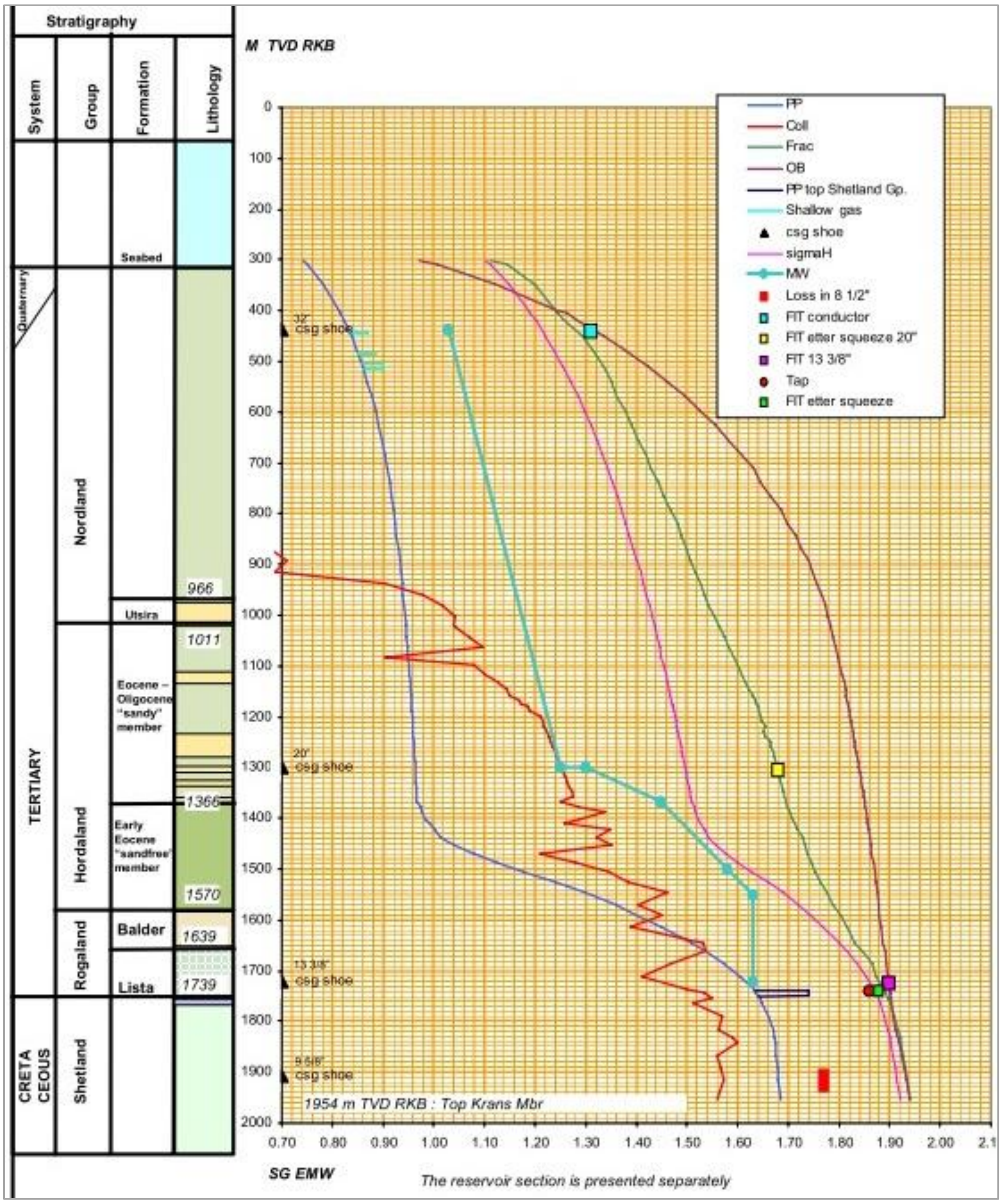


Figure 3.2: Pressure gradients from the final well report (Statoil, 2007).

3.2.1 Normal pressure gradient

Figure 3.3 shows the some of the pressure gradients of Figure 3.2 the way they have been imported to an Excel file. In addition, a normal pressure gradient has been calculated, assumed to be a hydrostatic water column. It is based on the water density assumed to be 1,044 kg/l for the North Sea (Bourgoyne et.al., 1986), and have been corrected for the RKB height. When compared to the reported pore pressure gradient of the upper well section, the calculated normal curve is observed to have lower values. For the calculations performed by the Matlab program, a corrected curve will be used.

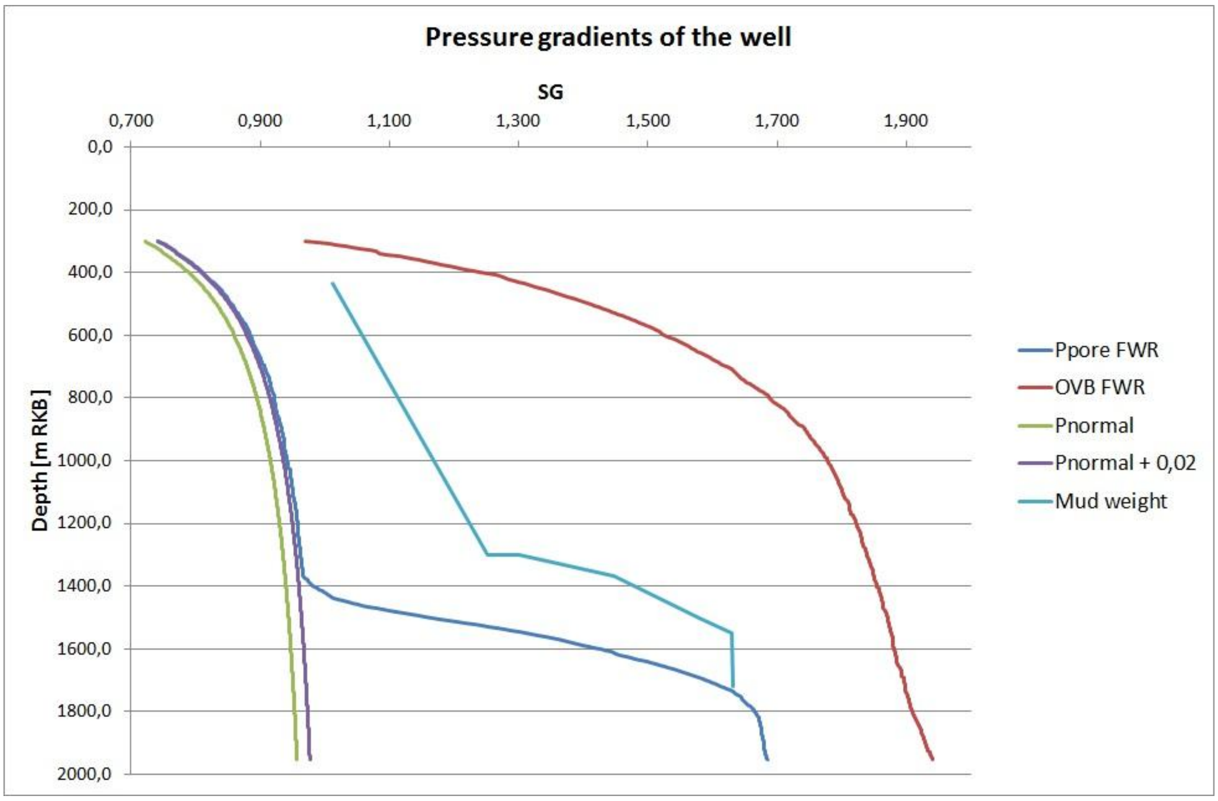


Figure 3.3: Pressure gradients imported to excel from Figure 3.1, in addition to a calculated normal pressure gradient adjusted to fit reported pore pressure of the upper section of the well.

4 Development of on-line tool for pore pressure detection

4.1 The models to be tested

The Matlab program that has been created will utilize available drilling data from the selected well to create pore pressure gradients. The pore pressure gradients will be calculated using three different methods, to provide improved evaluation. The methods selected for testing is:

- Zamora's method
- Eaton's method
- The Bourgoyne-Young drilling model

The two first are both based on use of the d_c-exponent method, presented in section 2.2.1. The choice of methods is based on which drilling parameters that have been made available. Even if this experiment is conducted upon pre-recorded drilling data, it will try to replicate a process that can be performed in real-time, making it viable as a method of pressure detection, not prediction.

4.1.1 d_c-exponent methods

The d_c-exponent is a method to normalize the penetration rate of drilling. As shown in equations 2.2 and 2.3 it utilizes the input parameters RPM, WOB, mud weight and bit diameter in addition to the penetration rate, all of which is made available from the data package.

$$d = \frac{\log \frac{ROP}{60\ RPM}}{\log \frac{12\ WOB}{10^3\ d_{bit}}} \tag{2.2}$$

Modified with respect to mud weight:

$$d_c = d\ \frac{\rho_{normal}}{\rho_{mud}} \tag{2.3}$$

The program will compute a trend line from the assumed normally pressured zones in the upper section of the well, and then present a plot of this trend line compared to the values computed for the full length of the well. As this method only detects at which depth a change of pressure occurs, the methods of Zamora and Eaton will be applied in order to quantify values of the pore pressure gradient.

Zamora's method, presented in section 2.2.2:

$$\rho_{pore} = \rho_{normal} \left(\frac{d_{c,normal}}{d_c}\right) \tag{2.4}$$

Eaton's method, presented in section 2.2.3, computed from d_c-exponent:

$$\rho_{pore} = \rho_{pore} - \left((\rho_{ovb} - \rho_{normal}) \left(\frac{d_c}{d_{c,normal}}\right)^{1,2} \right) \tag{2.8}$$

These methods require the normal pressure gradient and the overburden gradient in order to be computed. The normal pressure gradient to be used is the gradient calculated in chapter 3.2.1, adjusted to fit the pressure presented in Figure 3.2. The overburden gradient used in calculations will be based on the values provided in final well report, which have been imported to an Excel file.

4.1.2 Bourgoyne-Young drilling model

The full Bourgoyne-Young equation is presented in section 2.2.4.

With the data made available in the data package and final well report, it will be possible to compute only the factors $f_2 - f_6$. The factors f_7 and f_8, modeling bit wear effects and bit hydraulics respectively, will be neglected and given value 1. Also the threshold bit weight of f_5 will be assumed to have a value of zero.

As data from surrounding wells and formations is not available, the program will calculate the drillability factor, f_1, based on the data available for this well. Utilizing a built in Matlab function, a linear approximation will be made, both for the entire length of the well, and for each of the lithological zones presented in section 3.1.

The program will then compute the pore pressure gradient based on the Bourgoyne-Young drilling model, and produce a plot comparing it to the one provided by the final well report.

4.2 Importing field data

The data package provided by IPT contains recorded drilling data, which is stored in the Hierarchical Data Format (HDF5). This format designed to contain large amounts of numerical data. Each file holds the recorded data for a given time interval of the operation, with the different drilling data stored as one-dimensional arrays which can be read separately by a built in read-function in Matlab.

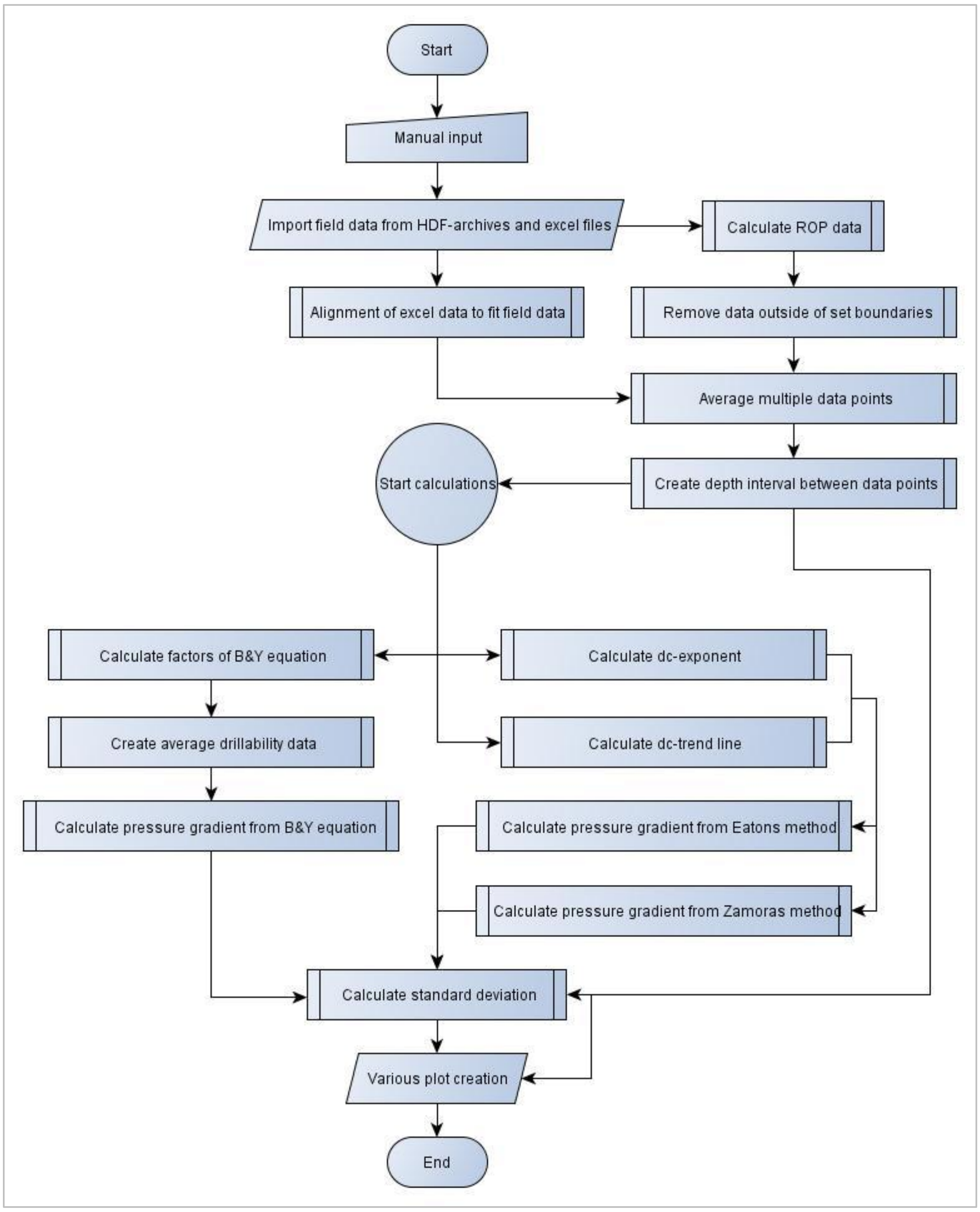
The total data amount in the data package is stored in 100 files, each containing a number of data points in the order of 10^4. The data have been stored with time as the indexing variable, where every data point in each different array corresponds to the same time. The time difference between each recording is 5 seconds.

The data have been recorded over the total time it takes to create the well. As such, in addition to containing actual drilling data, it holds records from periods where the drilling is at a halt, for instance during tripping or when casing is installed and cemented. To reduce the amount of files and data to be read and processed, the files are manually examined to determine if they contain actual records of drilling before importing data to the Matlab workspace. In the case of the data provided by IPT this evaluation reduced the number of files necessary for processing to 33.

In addition to the files of the data package, the pressure gradients presented in Figure 3.2 have been imported to an Excel file, which in turn will be read by a built in read-function in Matlab. As the Excel data does not contain the same time index as the HDF field data, a separate function will align the two with respect to vertical depth, creating arrays of corresponding length. These gradients will be used both in some of the calculations, and for comparison with the pore pressure estimates.

4.3 Flowchart

Figure 4.1 shows a graphical presentation of the data flow and calculations which is implemented in the Matlab program. The full source code is presented in appendix B. Some of the processing steps will be further explained in the following sections.



Start
Manual input
Import field data from HDF-archives and excel files
Calculate ROP data
Alignment of excel data to fit field data
Remove data outside of set boundaries
Average multiple data points
Start calculations
Create depth interval between data points
Calculate factors of B&Y equation
Calculate dc-exponent
Create average drillability data
Calculate dc-trend line
Calculate pressure gradient from B&Y equation
Calculate pressure gradient from Eatons method
Calculate pressure gradient from Zamoras method
Calculate standard deviation
Various plot creation
End

Figure 4.1: Flowchart for the main program, created to estimate pore pressure in three different ways.

4.3.1 Calculation of ROP data

The majority of this sub chapter is copied from a previous project (Stunes, 2011).

The data package includes calculated ROP-values for the well. Due to uncertainties regarding how these values have been calculated, the program will instead calculate its own values for ROP. This is done to be more certain that the ROP-data used in calculation of the d_c-exponent corresponds with the other drilling parameters that are used.

The new ROP values are calculated using a derivative of the block position recorded in the data. The block position data is first averaged for every three data points. Secondly the difference between them is divided with the time interval separating them. This would yield a more accurate measure of the ROP in each point, compared with the recorded data, which seem to be averaged over a larger time interval.

The manner of which these data have been calculated yields negative values for every instance where the block is being pulled up, for instance when the drillstring has a new pipe inserted. The method of computing may also yield some of very high values for the ROP, as it only represent the ROP for one very small time interval. This has to be considered erroneous data, and will be removed with further data processing.

4.3.2 Removal of unwanted data points

The majority of this sub chapter is copied from a previous project (Stunes, 2011).

The data files that are imported to Matlab have been controlled to ensure that they contain actual drilling data. They do however still contain many data points recorded at times where the drilling process have been at a halt, for instance each time a new pipe is installed in the drill string. These data is not wanted for the calculation of the d_c-exponent. Also there may be data points containing unrealistic values which will cause unwanted results, decreasing the quality of the final calculation.

The program will search the data for values of RPM, WOB and ROP which is not within predefined boundaries, and remove these values along with their corresponding entries in the other data arrays.

4.3.3 Averaging the data and creating depth interval between data points

The majority of this sub chapter is copied from a previous project (Stunes, 2011).

In order to improve the readability of the result plot the program will create new data arrays with a predefined vertical depth interval between each data entry. This is accomplished by a loop reading the vertical depth value of a data point, and then checking the following entries until a value with the required depth difference is found.

As a consequence of this method, a substantial amount of data entries will be removed before the final calculation is made. The decision on which data is kept is based entirely on the depth parameter, making the data selection from this process random and uncertain with regards to the quality of data being kept. In order to minimize this risk of error, the program will read multiple data entries and create average values before the depth intervals are made.

4.3.4 Calculating the standard deviation

In order to better be able to evaluate the accuracy of each of the methods used to calculate the formation pore pressure, a value of the standard deviation will be estimated. The function will read the difference between the estimated values created from field data, and the actual pore pressure provided in the final well report. Standard deviation is then calculated, based on the equation (Walpole et.al., 2006):

$$\sigma = \sqrt{\frac{\sum_{i=1}^{n}(X_i - \mu_i)^2}{n}} \tag{4.1}$$

Here X_i represents the pressure value computed from each respective method tested, and the expected value, μ_i, is given the value provided in the final well report.

5 Results from data analysis

This chapter will present the results from data analysis and calculations performed by the Matlab program. It will illustrate how the various functions of the program work, and form a base for evaluating the quality and reliability of the different models of pressure detection that have been applied.

5.1 Printout of input parameters

Most of the graphs described in this section are presented in Appendix A. Appendix A.1 shows the various input parameters used in the pressure calculations of the program. As continuous field data were not available until the depth of 612 m RKB, data points from the upper section of the well is not included. The data from the horizontal reservoir section of the well were removed, as this will not be evaluated, and not be of use when plotted versus vertical depth. Hence maximum depth of the data plots will be 1900 m RKB.

5.1.1 Printout of RPM, WOB and ROP

The figures A-1, A-3 and A-5 present the input parameters RPM, WOB and ROP respectively, as they are used in the upcoming calculations. In order to ensure that only data recorded during actual periods of drilling is included, the following boundaries have been applied, disallowing any data ranging beyond these values:

- 100 – 250 RPM
- 3 – 30 tons WOB
- 0 – 50 m/h ROP

In addition to this, every 54 data points have been averaged, and a 5 m depth interval has been created between each of the plotted values. The effect of these data processing steps can be seen when comparing to figures A-2, A-4 and A-6, where only negative values have been removed, and no data averaging occurs. The 5 m depth interval has been kept to improve readability.

The RPM values in figure A-1 are observed mainly to range between approximately a 115 – 225 RPM. The exception is the depth interval from 1300 – 1700 m, where it does not exceed 200 RPM.

The WOB values in figure A-3 are observed mainly to range between approximately 5 – 25 tons. However, there is a significant variation within the data throughout the length of the well. The section around 1200 m and the interval from approximately 1400 – 1500 m separates from the rest of the well, scarcely containing values below the value of 18 tons.

The ROP values in figure A-5 are observed mainly to range between approximately a 3 – 30 m/h. When comparing with figure A-6 the effect of averaging and data boundaries are especially significant for the ROP values, as opposed to WOB and RPM.

In general all the three parameters have quite large variations over fairly small depth intervals, even after the described data processing.

5.1.2 Mud weight and ECD

In Figure 5.1 the mud density from the field data package is compared with the mud weight provided in the final well report. The latter does not contain as many variations as the recorded field data, and it is only plotted until a depth of 1736 m RKB. This is where drilling of the Shetland formation is initiated, using MPD techniques. The two mud weight gradients are observed to correlate, both with respect to the density magnitude and depth.

Figure A-7 shows the development of the mud weight and the effective circulating density, both originating from the field data package. This parameter is included in all drilling models that are to be tested. The difference between mud density and ECD is observed to increase slightly alongside increasing depth.

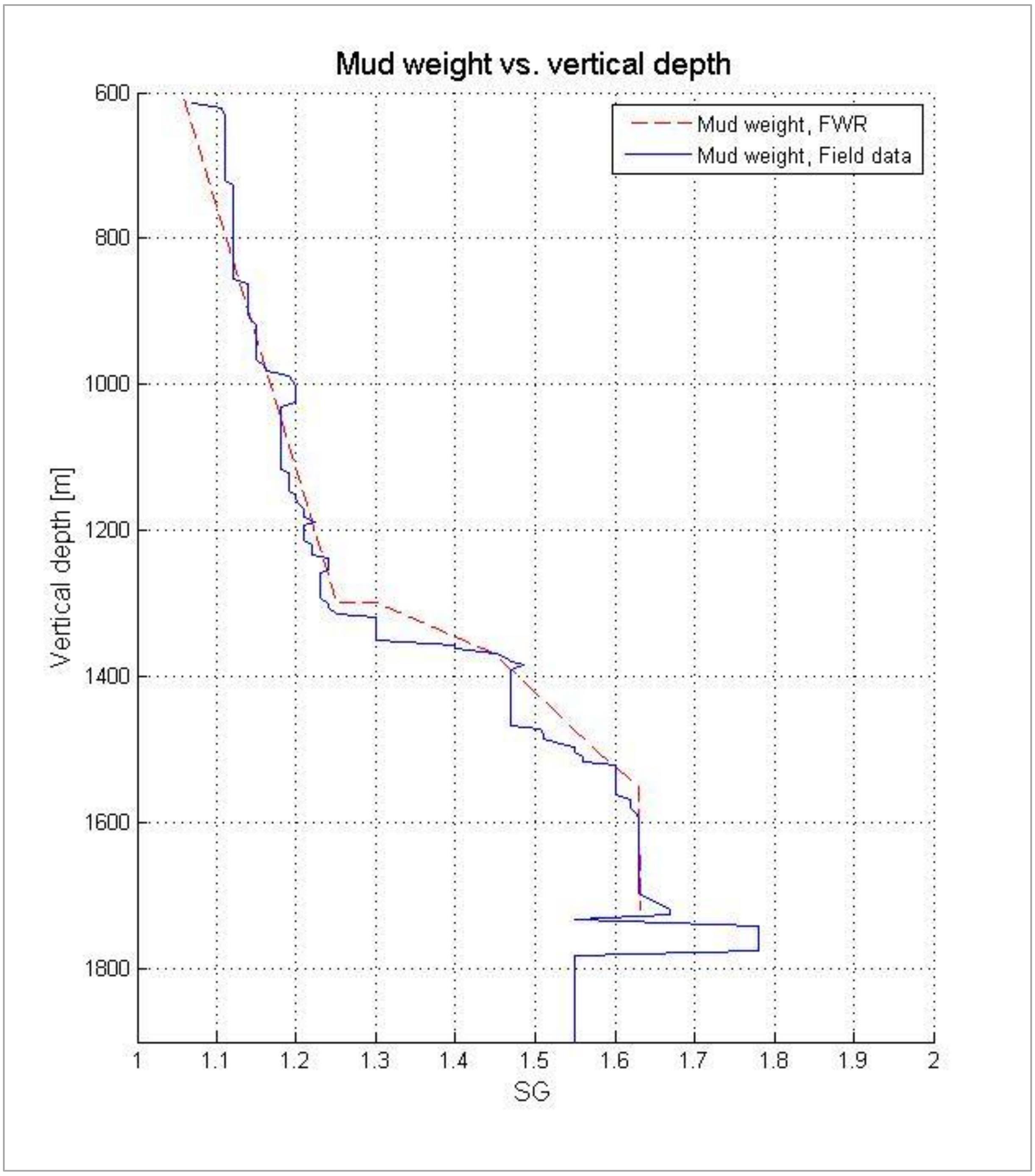


Figure 5.1: Comparison of mud weight reported in the final well report and the continuously recorded mud weight from the field data package.

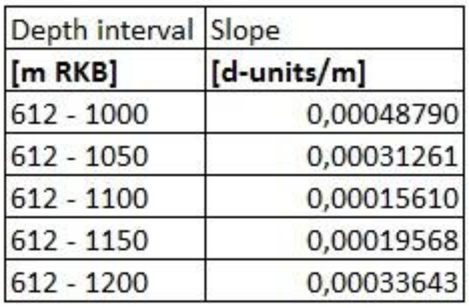
5.2 d_c-exponent trend line

Figure 5.2 shows a plot of the calculated d_c-exponent values. The trend line is based on the assumed normal pressured zones in the upper well sections. In this case the basis for trend line creation is 612 – 1150 m RKB.

The values can be observed to have significant variations throughout the length of the plotted data. In terms of deviation from the trend line, a major positive displacement is observed from approximately 1175 – 1300 m RKB. Below 1325 m RKB, a fairly consistent negative displacement is observed, indicating that an abnormal pressured zone is being entered. When comparing this to the reported pore pressure in Figure 3.2, this overpressured zone should not be encountered until approximately 1400 m RKB.

Figures A-8, A-9 and A-10 present plots where a different depth interval has been utilized, which result in variations of the trend line slope. The different slopes, with respect to depth interval for its estimation, are presented in Table 5-1. The differences of the trend line slopes are seen to have significant effect on the depth where deviations from the trend are observed.

Table 5-1: The slopes of the different d_c-exponent trend lines, varying with respect to the depth intervals used as basis for its creation.



Depth interval	Slope
[m RKB]	**[d-units/m]**
612 - 1000	0,00048790
612 - 1050	0,00031261
612 - 1100	0,00015610
612 - 1150	0,00019568
612 - 1200	0,00033643

Figure A-11 shows a plot where the field data has not been processed with respect to data boundaries or averaging of data.

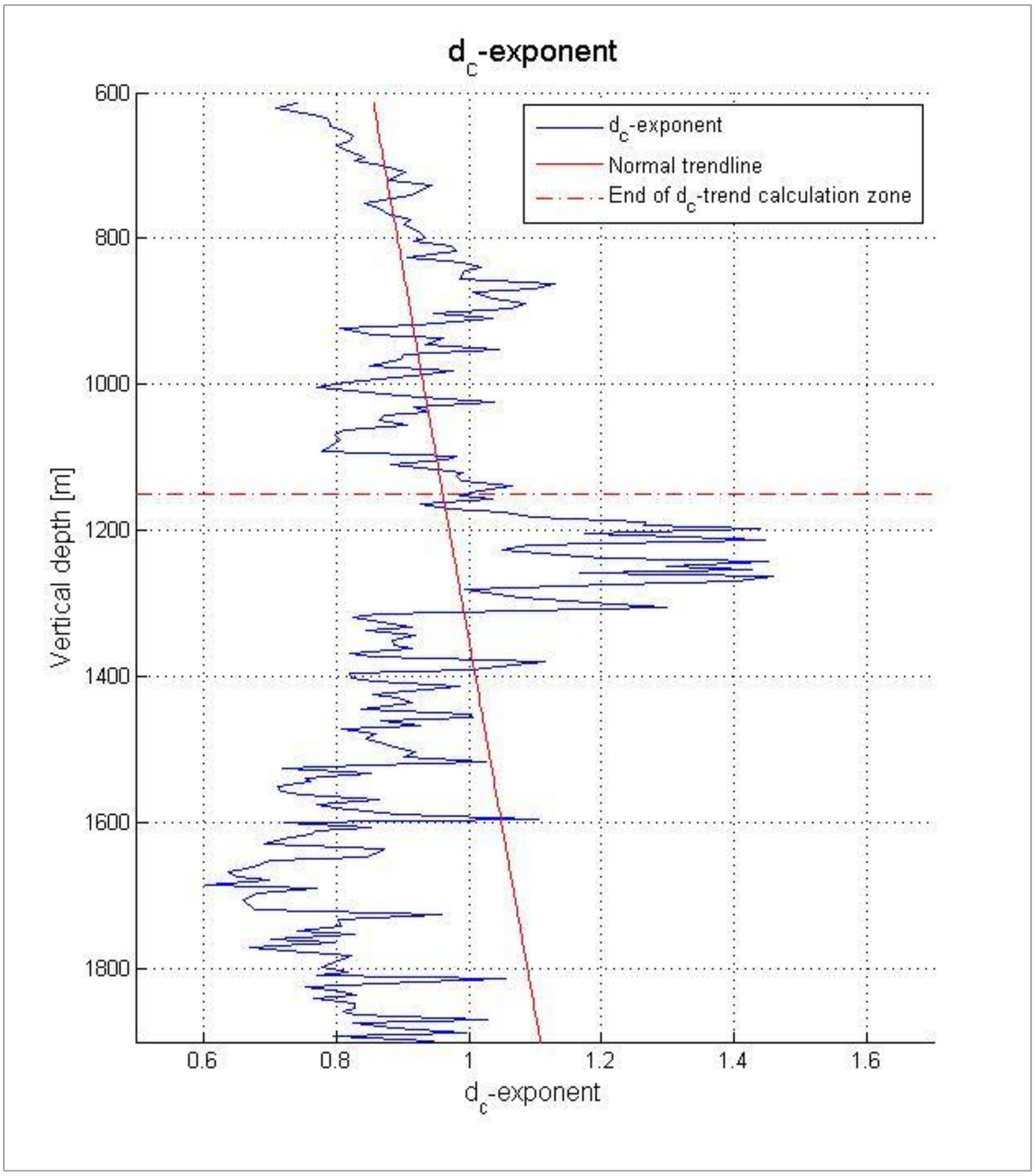


Figure 5.2: d_c-exponent plot from Matlab program. Trendline is established on basis of assumed normal conditions in the 612 – 1150 m RKB interval.

5.3 Zamora pressure estimation

Figure 5.3 shows a plot of calculated pore pressure gradient, using Zamora's empirical relation. The calculated values are compared with the pore pressure gradient provided in the final well report.

The values can be observed to have significant variations throughout the length of the plotted data. When comparing the estimated values to the assumed real ones, an overall correlation is observed. However, in the interval from 1175 – 1300 m RKB, a significant decrease in the pore pressure gradient is shown. This interval lies within a section of the well that has been identified, in the final well report, as being placed within a sandy member of the Hordaland formation group. Below approximately 1700 m RKB, the estimated values are shown to have a large negative displacement in comparison with the final well report values.

The Figures A-12, A-13 and A-14 present plots where a different depth interval has been utilized for the d_c-exponent. The different trend lines are observed to have an impact on how well the estimated pore pressure gradient values correlate with the values from the final well report.

Figure A-15 shows a plot where the field data has not been processed with respect to data boundaries or averaging of data.

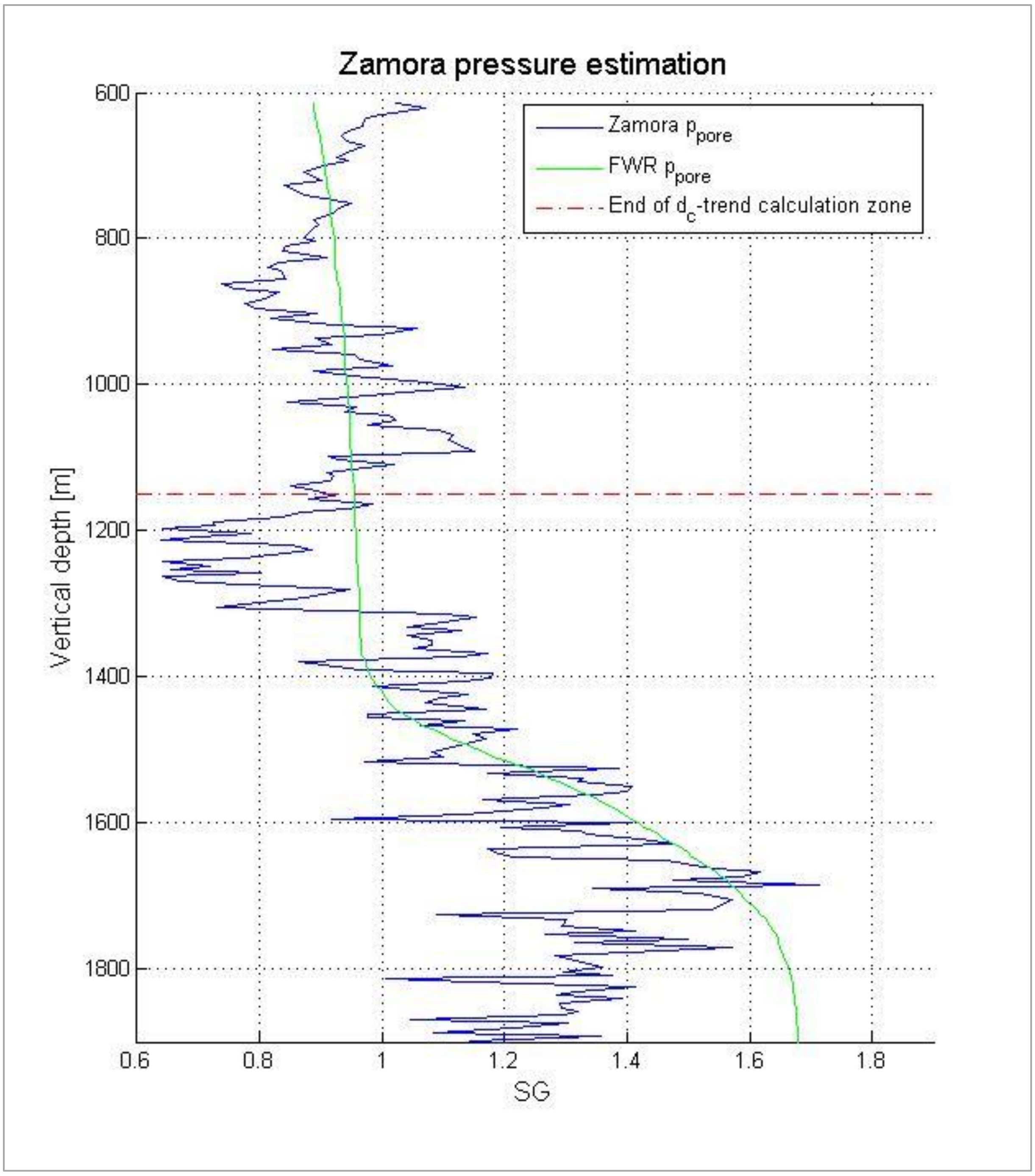


Figure 5.3: Zamora pressure gradient, with d_c–exponent trendline interval of 612 - 1150 m RKB, compared with pore pressure reported by Statoil.

5.4 Eaton pressure estimation

Figure 5.4 shows a plot of calculated pore pressure gradient, using Eaton's method. The calculated values are compared with the pore pressure gradient provided in the final well report.

Compared with the results from use of Zamora's method, the overall data variations are observed to be smaller. However, in the interval from 1175 – 1300 m RKB, a significant decrease in the pore pressure gradient is shown. This is observed to be a larger deviation from the final well report values than with Zamora's method. Also, the estimated values are shown to have a large negative displacement in comparison with the final well report values, starting at a depth of approximately 1580 m RKB.

The Figures A-16, A-17 and A-18 present plots where a different depth interval has been utilized for the d_c-exponent. The different trend lines is observed to have an impact on how well the estimated pore pressure gradient values correlate with the values from the final well report.

Figure A-19 shows a plot where the field data has not been processed with respect to data boundaries or averaging of data.

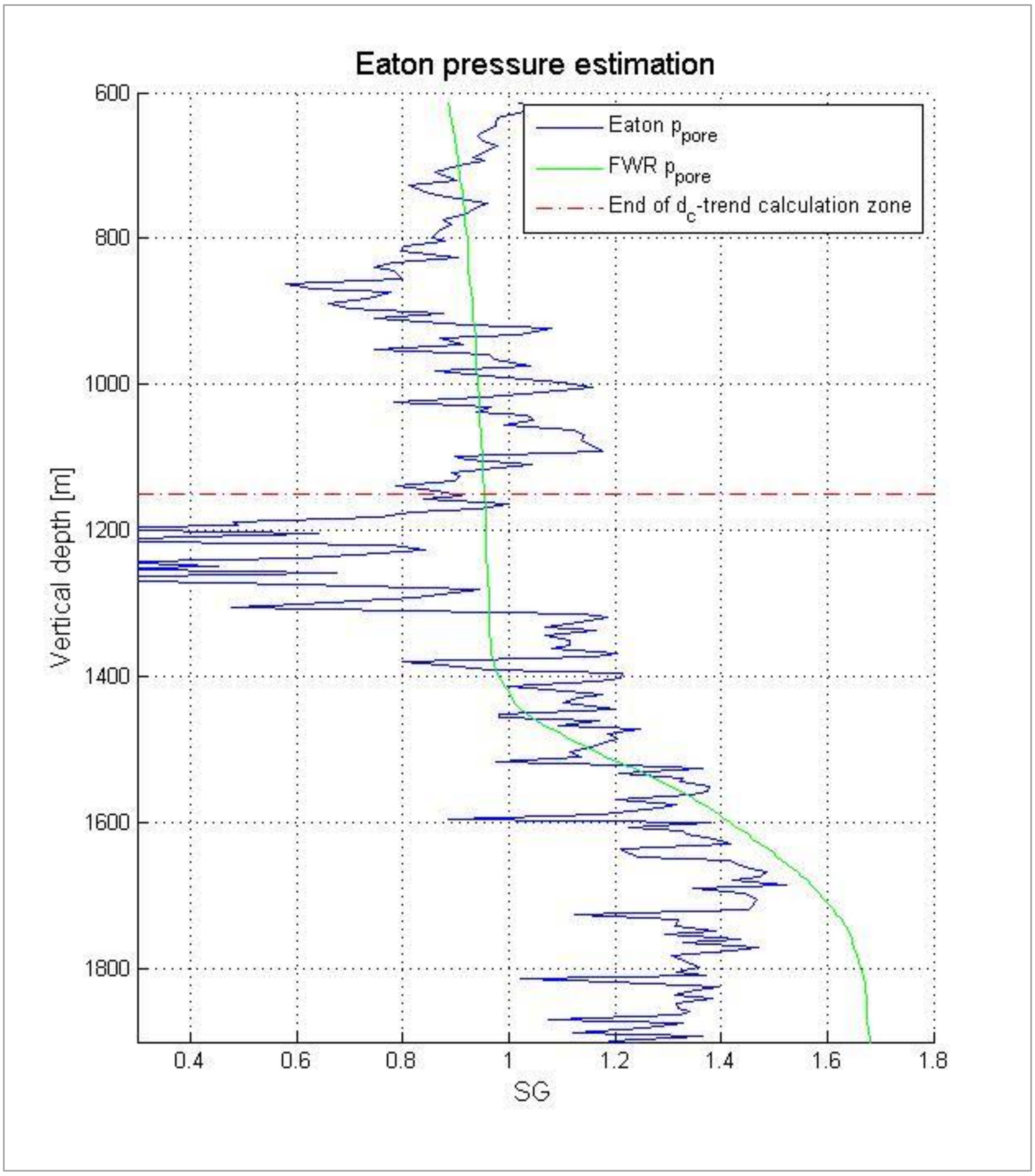


Figure 5.4: Eaton pressure gradient, with d_c–exponent trendline interval of 612 - 1150 m RKB, compared with pore pressure reported by Statoil.

5.5 Bourgoyne-Young pressure estimation

Figure 5.5 shows a plot of a calculated pore pressure gradient, using the Bourgoyne-Young drilling model. The calculated values are compared with the pore pressure gradient provided in the final well report.

In general, the calculated values of this drilling model are observed to have a good correlation with the values of the final well report. In this case, the overall data variation is reduced below a depth of 1400 m RKB. As with the previous result plots, based on the other estimation methods, a large negative displacement is observed in the depth interval from 1175 – 1300 m RKB. At 1736 m RKB, a line is indicating the top of the Shetland formation, where an increase of the pore pressure is reported in the final well report. This increase may also be seen in the calculated results. However, with the amount of deviations in the data, this cannot be determined to be conclusive.

The drillability factor estimate utilized in Figure 5.5 is presented in figure A-20 of appendix A. Here, a linear regression is performed such that it creates a different drillability trend for each of the lithological zones presented in Table 3-2, found on page 17.

Figure A-22 presents a pore pressure gradient where the drillability factor trend, shown in Figure A-21, is calculated as one continuous linear estimate covering the whole length of the well. The results are observed to deviate more from the reported values than what is seen in Figure 5.5.

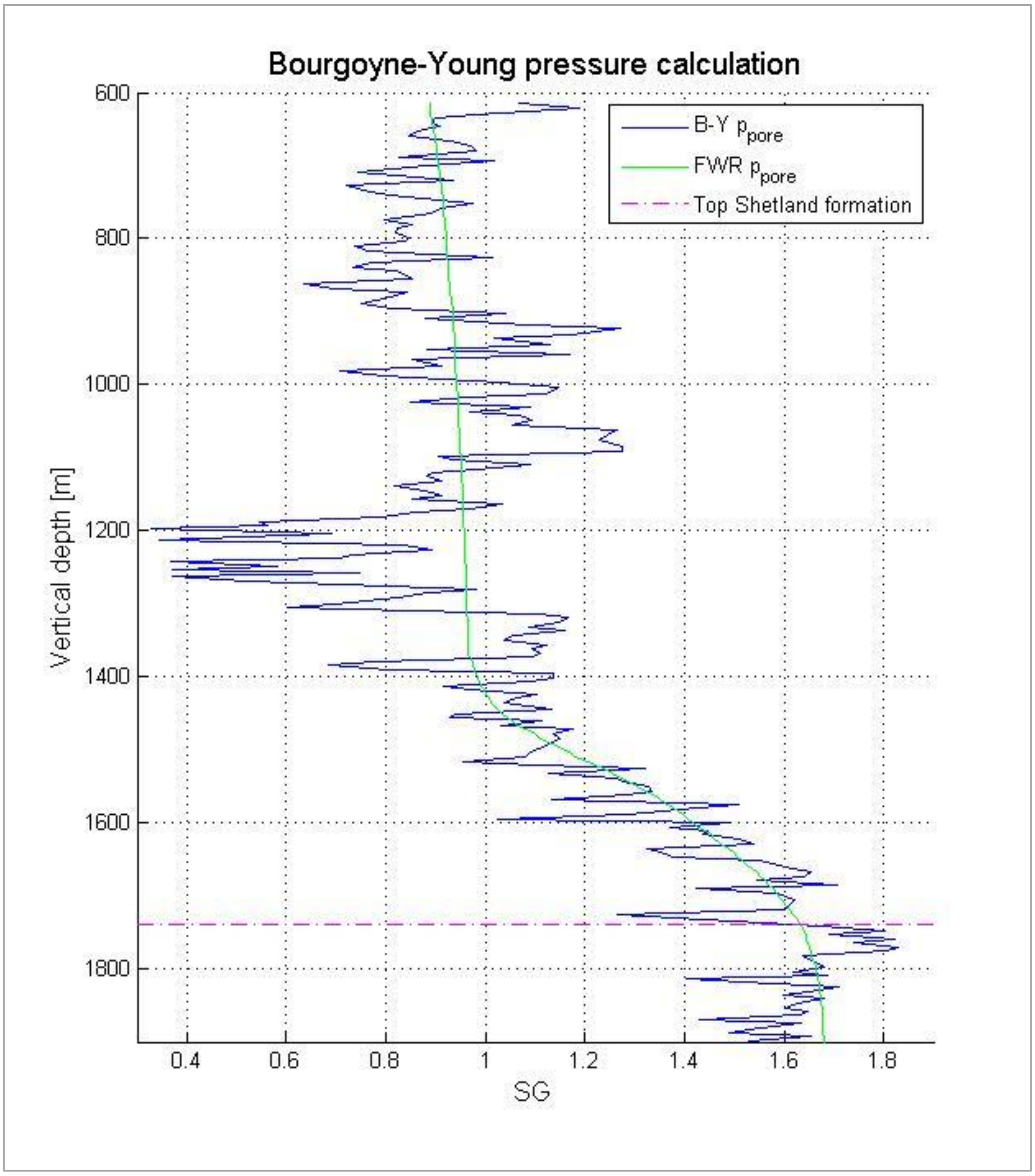
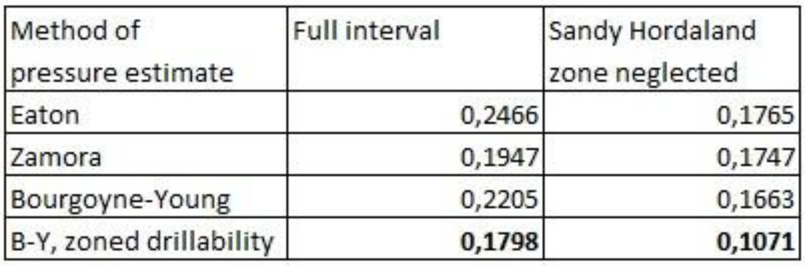


Figure 5.5: Bourgoyne-Young pressure gradient compared with pore pressure reported by Statoil.

5.6 Standard deviations

Standard deviations, of the difference between the estimated pressure gradients and the reported pore pressure of the final well report, have been calculated. Results are presented in Table 5-2. As major deviations are observed in the sandy Hordaland Group member, a separate calculation has been made, neglecting this interval.

Table 5-2: Standard deviation calculated from the difference between the calculated pore pressure gradients and the pore pressure reported by Statoil.



Method of pressure estimate	Full interval	Sandy Hordaland zone neglected
Eaton	0,2466	0,1765
Zamora	0,1947	0,1747
Bourgoyne-Young	0,2205	0,1663
B-Y, zoned drillability	**0,1798**	**0,1071**

The lowest value of standard deviation is found using the Bourgoyne-Young drilling model, with the drillability trend estimated separately for each lithology section. This is in compliance with the observations made in the plots previously presented in this chapter.

Figures A-23, A-24 and A-25 of Appendix A.6 present the estimated pressure gradients compared with the reported pore pressure and the calculated standard deviation, indicating how much of the data is within this range.

6 Discussion

This report presents the results from utilizing different models of pressure estimation, as presented in sections 2.2, to analyze real time drilling data recorded from a North Sea well. The intention was to calculate the pore pressure gradient, and compare the results from the different models with values provided in a final well report provided alongside the drilling data. This comparison is observed in sections 5.2 – 5.4, with standard deviation presented in section 5.5. A certain correlation can be observed, but the resulting plots also contain deviations from the trend line, which is not in compliance with the values of the final well report. This indicates that the calculations have uncertainties to be assessed.

6.1 Weaknesses and limitations

The following weaknesses and limitations to the model, data and implementation of the Matlab program can be identified:

6.1.1 General

These general remarks have implications on all the three estimation methods:

- All data values occurring during pipe trip operations, cement squeeze jobs etc. were removed by the use of data boundaries. If these boundaries are not properly designated, it will cause erroneous results in the final calculation of the pressure gradients.

- Data boundaries for selection of drilling data, as well as the creation of average values, may result in a reduction of data quality and accuracy. However, the boundaries selected are assumed to be wide enough not to exclude critical data. To be able to display plots with a desired level of readability, only data points having a depth interval of 5 meters between them have been plotted. The average values that have been created serves to minimize the error of this random selection. The evaluation of the plots presented in this report, without the use of boundaries and average values, are confirming the necessity of these processing steps.

- ROP values of the initial recorded data were calculated by an external party; the drilling company. The data manipulation process is unknown to the present author. As a consequence, the ROP values actually used for d_c-exponent calculation have been the travelling block velocity, which is calculated as a derivative of the block position. We are now confident with the ROP data values.

- Calculations made by the Matlab program are based upon data from two separate sources; the drilling data package and the final well report. The consistency of these two, especially with respect to the depth parameter, is considered to be very important. The two data sources have been shown to correspond well with each other in Figure 5.1, which enables their simultaneous use in calculations.

6.1.2 d_c-exponent methods

- When utilizing the d_c-exponent method for analyzing the relationship between ROP and formation pore pressure, the drillability of the formation is assumed to have a constant value. No lithology changes are addressed when implementing the model. The d_c-methods assume that the trend line is based on data recorded when drilling in normal pressured shale formations. The real-time data provided does not contain gamma ray values, which are needed for identifying such shale sections.

- Both methods utilizing the d_c-exponent are shown to rely on anestimated trend line. Variations of the calculated pore pressure gradients, with respect to the selected depth interval used for trend line estimation, is observed. The best result was produced when the trend line interval ended directly above a zone where the d_c-exponent had a significant positive deviation from the trend.

- Choice of bit type and bit wear has an impact on the ROP values of drilling. This may not be recognized on a local scale, but rather globally; from the start to the end of the section. As the d_c-exponent based methods only takes bit size into account, it is possible that such effects are a source of error in the pore pressure estimates.

- Both methods are observed to have significant deviations from the reported pressure in zones drilled in near balance. The d_c-exponent utilizes the effect that bottom hole differential pressure has on the ROP to estimate pore pressure. Using MPD, this differential pressure is kept at a minimum, suggesting that these methods are not viable during MPD.

6.1.3 Bourgoyne-Young drilling model

- The drilling constants used in the calculation of the different factors of the model have been assumed to be equal to values obtained from shale formations in the Gulf of Mexico. These may be applicable also in the shale formations in the North Sea. However, the well does also contain zones of sedimentary sandstone, in which it must be assumed that these constants will cause errors.

- The drillabilty factor have been estimated in two ways, one evaluated in basis of for the whole well section, the other separating the well in intervals based on reported lithology's. Both estimates utilized the pore pressure gradient obtained from the final well report. This had to be done due to lack of drillability data from other wells in the same formations.

- The factors estimating bit tooth wear and bit hydraulics have been neglected in the calculations. However, with the magnitude of the variations over even small depth intervals, which is seen in the results, it is not likely that these will have a major impact. Also, as the drillability factor is estimated from the same well data as the pressure gradient, any effects of bit wear or bit hydraulics may be incorporated within this factor.

6.2 Future improvements

The following issues should be addressed in future development of the tool for pore pressure detection:

- The mathematical functions for removal of unwanted data should be improved, in order to reduce uncertainties regarding data quality. This involves better evaluation of the input parameters, and selection of the data suitable for calculations and trend line estimation. The addition of gamma ray values, providing reliable identification of shale sections, would be particularly beneficial. A preliminary separation of the well into smaller sections for more detailed data analysis is an ad-hoc possibility.

- Type of drilling bit and bit tooth wear are important factors for the ROP. As a first step, these effects should be checked in terms of magnitude/importance for the result. If significant, the effects should be implemented in some way.

- The implementation of the model should be tested for the complete overburden section, and for more than one well.

- The Bourgoyne-Young drilling model should be tested with drilling constants calculated from data of other wells in the same region.

- The quality of the pore pressure gradient provided in the final well report should be evaluated. Which methods that have been utilized for its creation is unknown.

- Matlab is considered to have the necessary functions to be used as a work platform for analyzing real time data. More work on the implementation of the model is required for it to yield the desired accuracy in its result data graphical plots. This includes better functions for selecting data to be used for calculations. Also a separate input file, for manual input of filenames, data boundaries and other values should be implemented, making the use of the program more intuitive.

7 Conclusion

The results of the data analysis performed in this thesis enable us to draw the following conclusions:

- Three methods of pore pressure estimation have been implemented in a Matlab program, to be utilized for calculations based on recorded drilling data. The chosen methods were the Zamora and Eaton methods, based on calculation of the d_c-exponent, and the Bourgoyne-Young drilling model. The implementation has been successful in yielding an estimated pore pressure gradient for all the tested methods.

- All the different mathematical methods of pore pressure estimation that have been evaluated, is observed to yield results that in part is in compliance with those provided in the final well report of the operating company.

- Based on the data available, and the method that data analysis and calculations has been implemented in Matlab, the Bourgoyne-Young drilling model yields more accurate results of pore pressure detection than the d_c-exponent based methods of Zamora and Eaton.

- The raw data from the field data package is not of sufficient quality to provide a basis for calculating the pore pressure gradient. The data requires certain processing and selection to be executed, before being able to produce pore pressure estimates of desired quality.

- To be able to improve the accuracy and reliability of the calculations in the future, the inclusion of gamma ray values in the field data is considered to be most relevant.

Nomenclature

a_1-a_8	-	Constants of the Bourgoyne-Young drilling model
BPOS	-	Block position
d_{bit}	-	Bit diameter
D	-	Depth
ECD	-	Effective circulating density
f_1-f_8	-	Functions of the Bourgoyne-Young drilling model
FWR	-	Final well report
HDF5	-	Hierarchical Data Format
IPT	-	Department of Petroleum Technology and Applied Geophysics
K	-	Drillability
MSL	-	Measured sea level
OVB	-	Overburden
PDC	-	Polycrystalline Diamond Compact
ρ	-	Density
R	-	Resistivity
RKB	-	Rotary kelly bushing
ROP	-	Rate of penetration
RPM	-	Rotations per minute
$\sigma_{vertical}$	-	Vertical stress
TVD	-	True vertical depth
WOB	-	Weight on Bit

References

[1] Bourgoyne A.T., Millheim K.K., Chenevert M.E. and Young F.S.: "Applied Drilling Engineering", Society of Petroleum Engineers, Richardson (1986).

[2] Bourgoyne A.T., Young F.S.: "A Multiple Regression Approach to Optimal Drilling and Abnormal Pressure Detection", paper SPE 4238, presented at SPE AIME Sixth Confernce on Drilling and Rock Mechanics, Austin (January, 1974)

[3] Eaton B.A.: "The equation for Geopressure Prediction from well logs", paper SPE 5544, presented at SPE AIME Annual fall meeting, Dallas (September 1975).

[4] Eaton B.A.: "The Effect of overburden Stress on Geopressure Prediction from Well Logs", paper SPE 3719, presented at SPE Abnormal Subsurface Pore Pressure Symposium, Baton Rouge (May 1972).

[5] Holbrook P.W., Maggiori D.A., Hensley R.: "Real-Time Pore Pressure and Fracture-Pressure Determination in All Sedimentary Lithologies", paper SPE 26791, presented at Offshore European Conference, Aberdeen (September 1995).

[6] Jorden, J.R. and Shirley, O.J.: "Application of Drilling Performance Data to Overpressure Detection" paper SPE 1407, presented at SPE Symposium on Offshore Technology and Operations, New Orleans (May 1966)

[7] Norlex (Norwegian Interactive Offshore Stratigraphic Lexicon), University of Oslo (2012): http://nhm2.uio.no/norges/litho/nordland.php
http://nhm2.uio.no/norges/litho/hordaland.php
http://nhm2.uio.no/norges/litho/rogaland.php
http://www.nhm2.uio.no/norges/litho/shetland.php

[8] Norwegian Petroleum Directorate (NPD), (2012):
http://factpages.npd.no/factpages/Default.aspx?culture=en&nav1=wellbore&nav2=PageView|Exploration|All&nav3=524

[9] Rehm, B. and McClendon, R.: "Measurment of Formation Pressure from Drilling Data", paper SPE 3601, presented at SPE Annual Fall Meeting, New Orleans (October 1971).

[10] Statoil ^{v}/ Christophersen L., Gjerde J., Valdem S.: “Final Well Report”, Statoil (2007)

[11] Stunes, S., “Development of on-line tool for detection of pore pressure”, student project, NTNU, Trondheim (December 2011).

[12] Vidrine, D.J. and Benit, E.J.: “Field verification of the Effect of Differential Pressure on Drilling Rate”, paper SPE 1859, presented at SPE Annual Fall Meeting, Houston (October 1967).

[13] Walpole R.E., Myers R.H., Myers S.L. and Ye K.: “Probability and Statistics for scientists and engineers”, Prentice Hall (2007).

[14] Yassir N.A., Bell J.S.: “Abnormally High Fluid Pressures and Associated Porosities and Stress Regimes in Sedimentary basins” paper SPE 28139, presented at SPE/IRSM Rock Mechanics in Petroleum Engineering Conference, Delft (August 1994)

[15] Yoshida C., Ikeda S., Eaton B.A.: “An Investigative Study of Recent Technologies Used for Prediction, Detection, and Evaluation of Abnormal Formation Pressure and Fracture Pressure in North and South America”, paper IADC/SPE 36381, presented at IADC/SPE Asia Pacific Drilling Technology Conference, Kuala Lumpur (September 1996)

Appendices

Appendix A:	**Additional result plots**
A.1:	**Field data**
A.2:	**d_c-exponent**
A.3:	**Zamora’s method**
A.4:	**Eaton’s method**
A.5:	**Bourgoyne-Young drilling model**
A.6:	**Standard deviation**
Appendix B:	**Matlab code**
B.1:	**Main script**
B.2:	**Import functions**
B.3:	**Data processing functions**
B.4:	**Calculation functions**
B.5:	**Output functions**

A Additional result plots

This appendix will present various plots in order to further illustrate the calculations made in the Matlab program, and visualize how changes in the parameters influence the result.

A.1 Field data

A.1.1 RPM

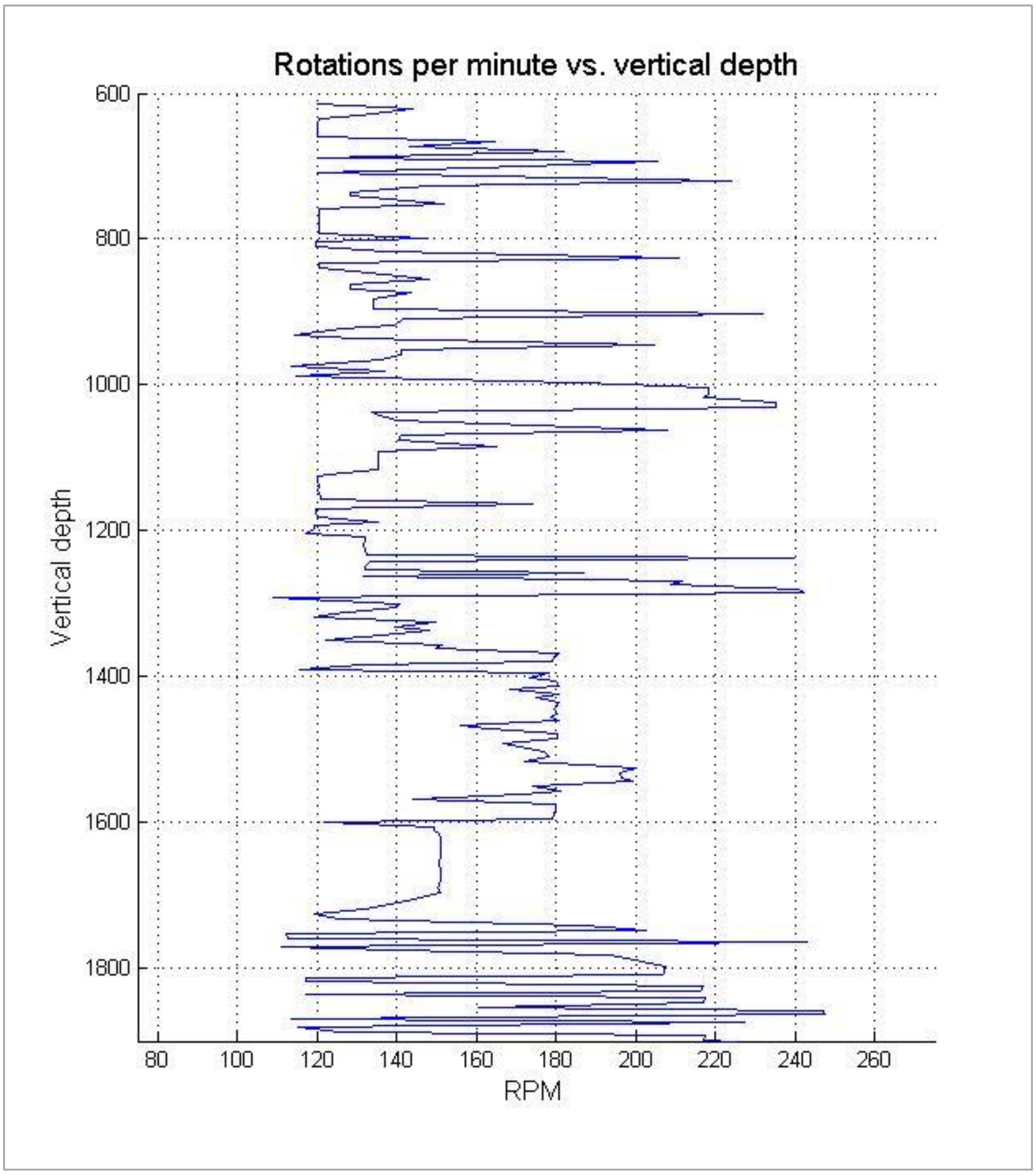


Figure A-1: RPM values as they appear when used in the calculations.

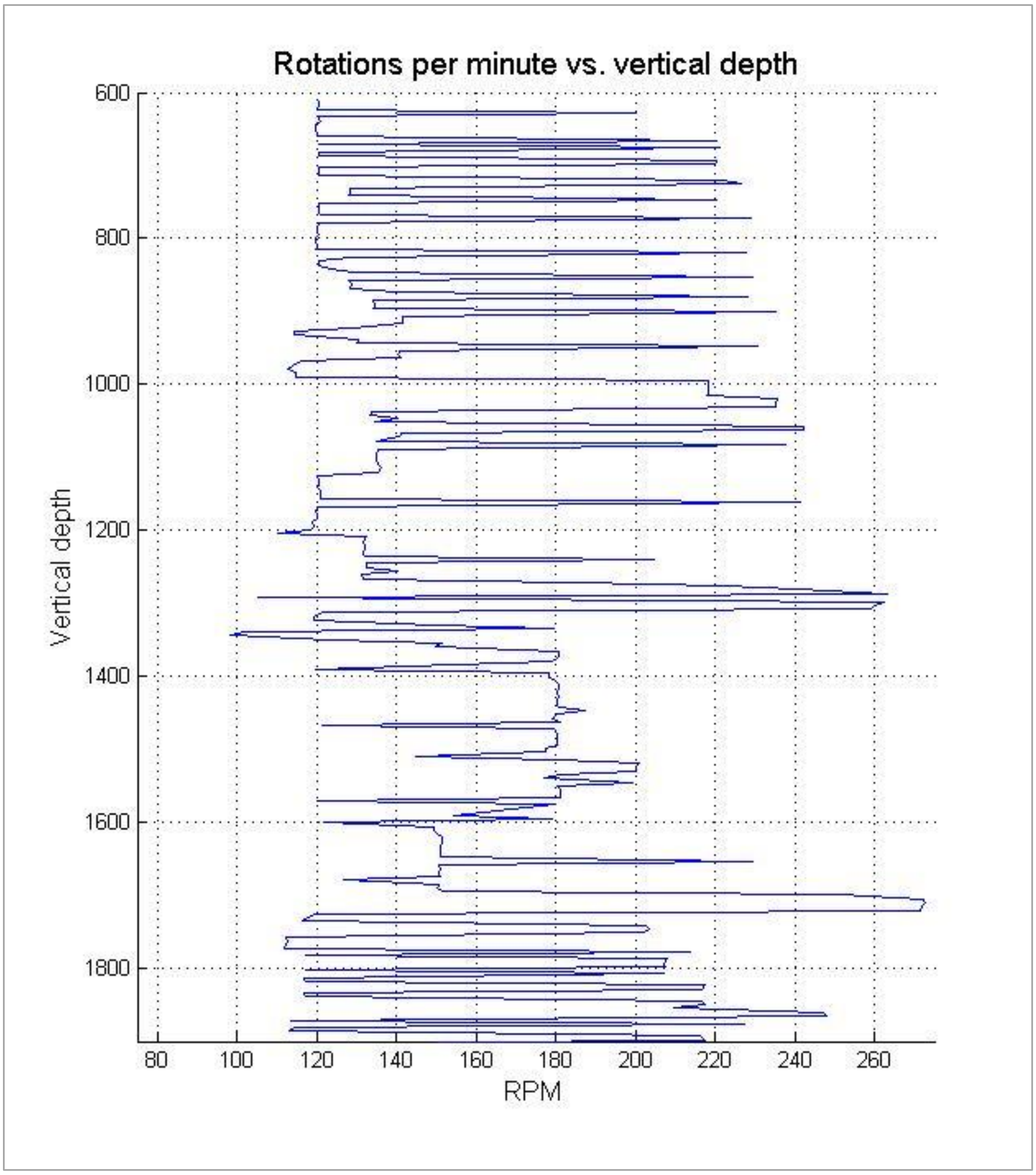


Figure A-2: RPM before data boundaries have been applied and average values created. The 5 m interval between plotted data points is still included.

A.1.2 WOB

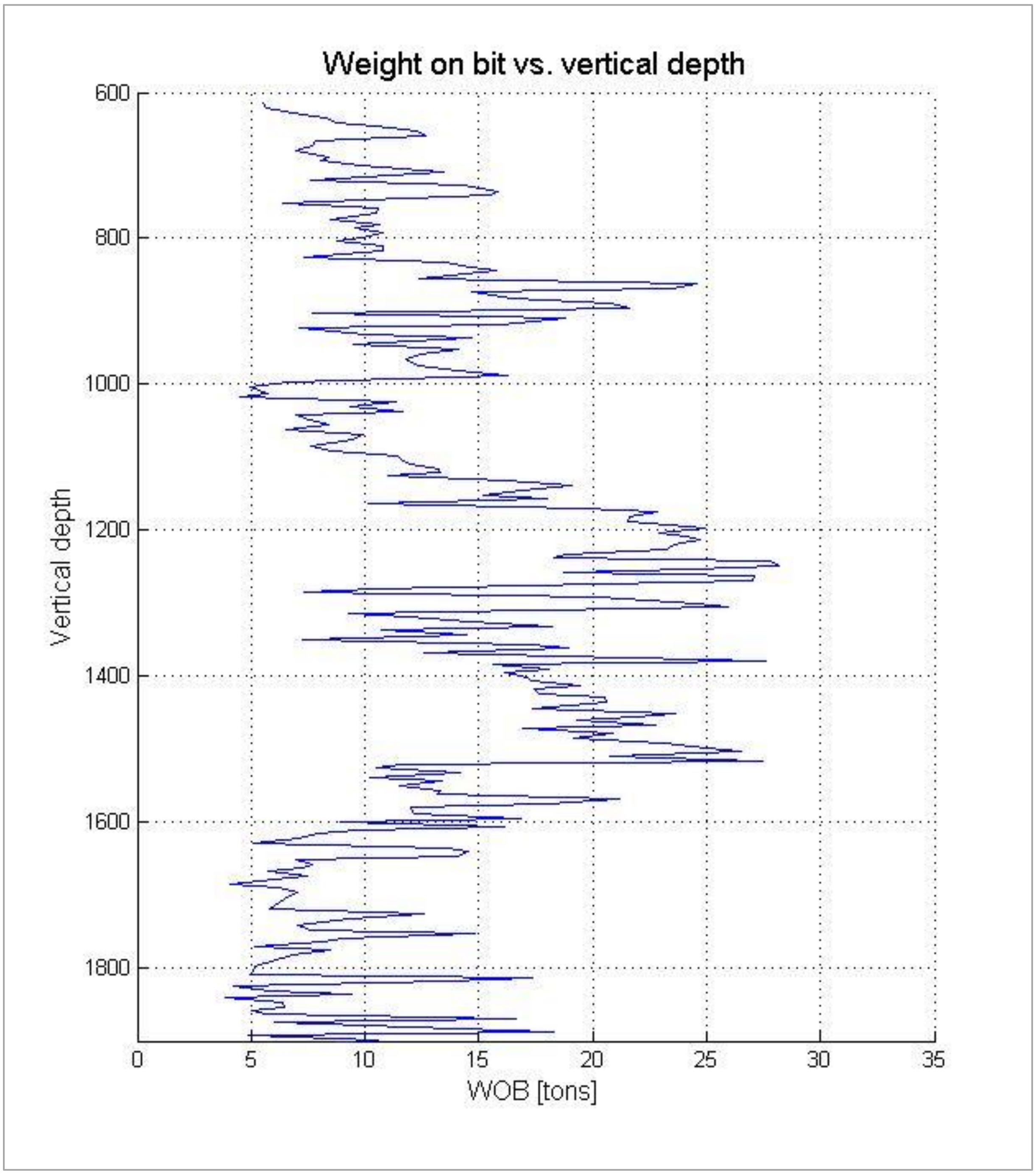


Figure A-3: WOB values as they appear when used in the calculations.

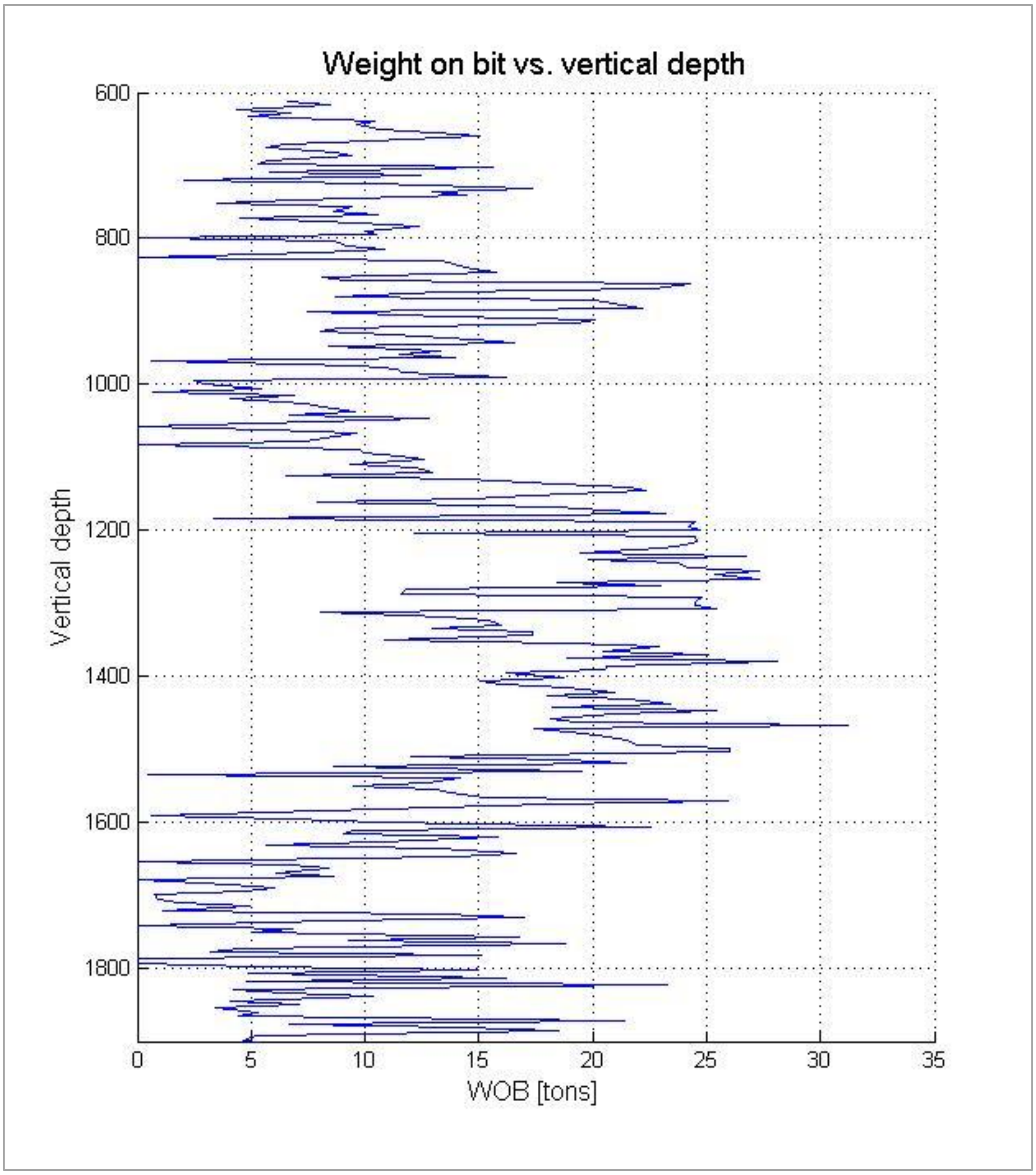


Figure A-4: WOB before data boundaries have been applied and average values created. The 5 m interval between plotted data points is still included.

A.1.3 ROP

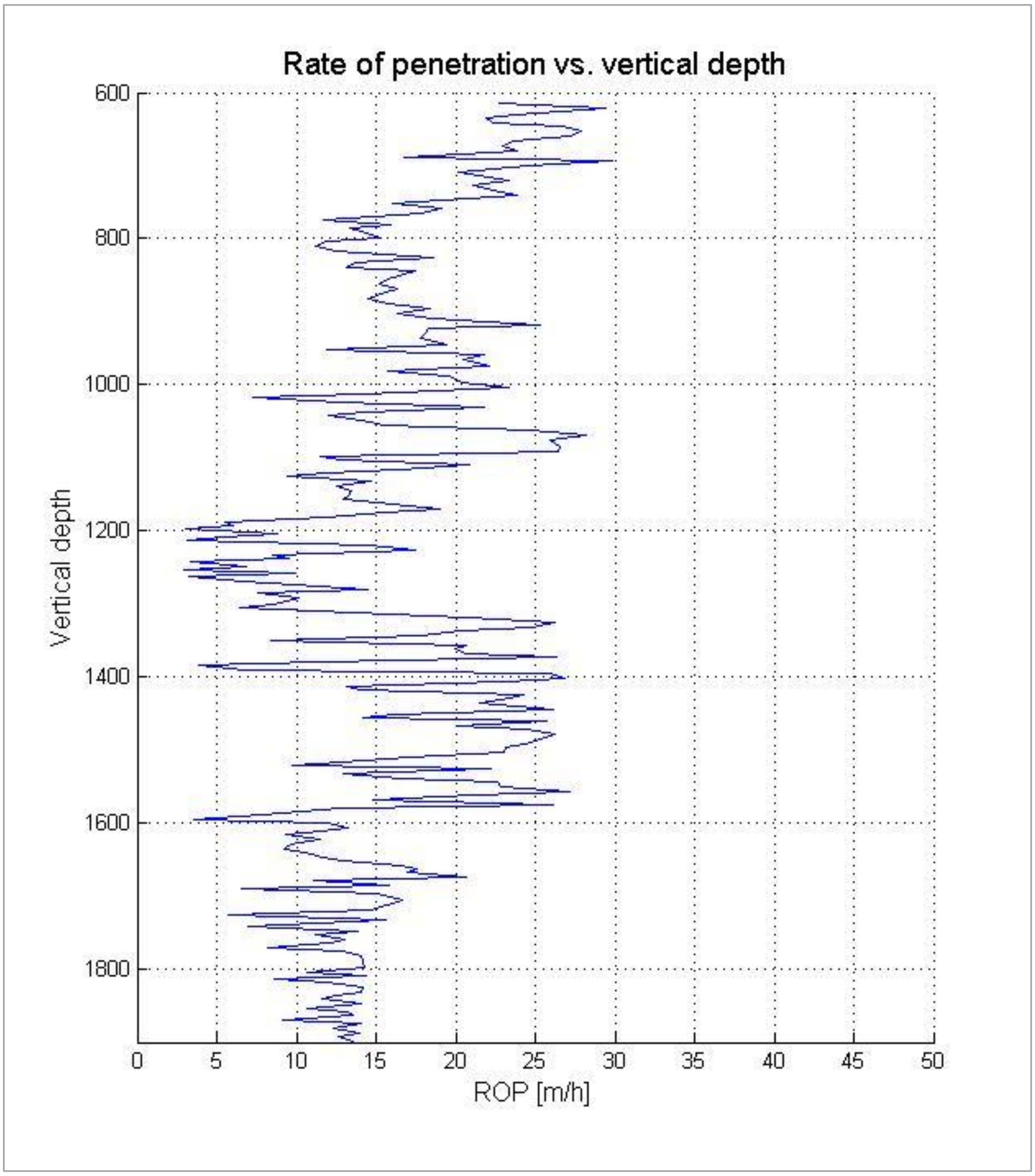


Figure A-5: ROP values as they appear when used in the calculations.

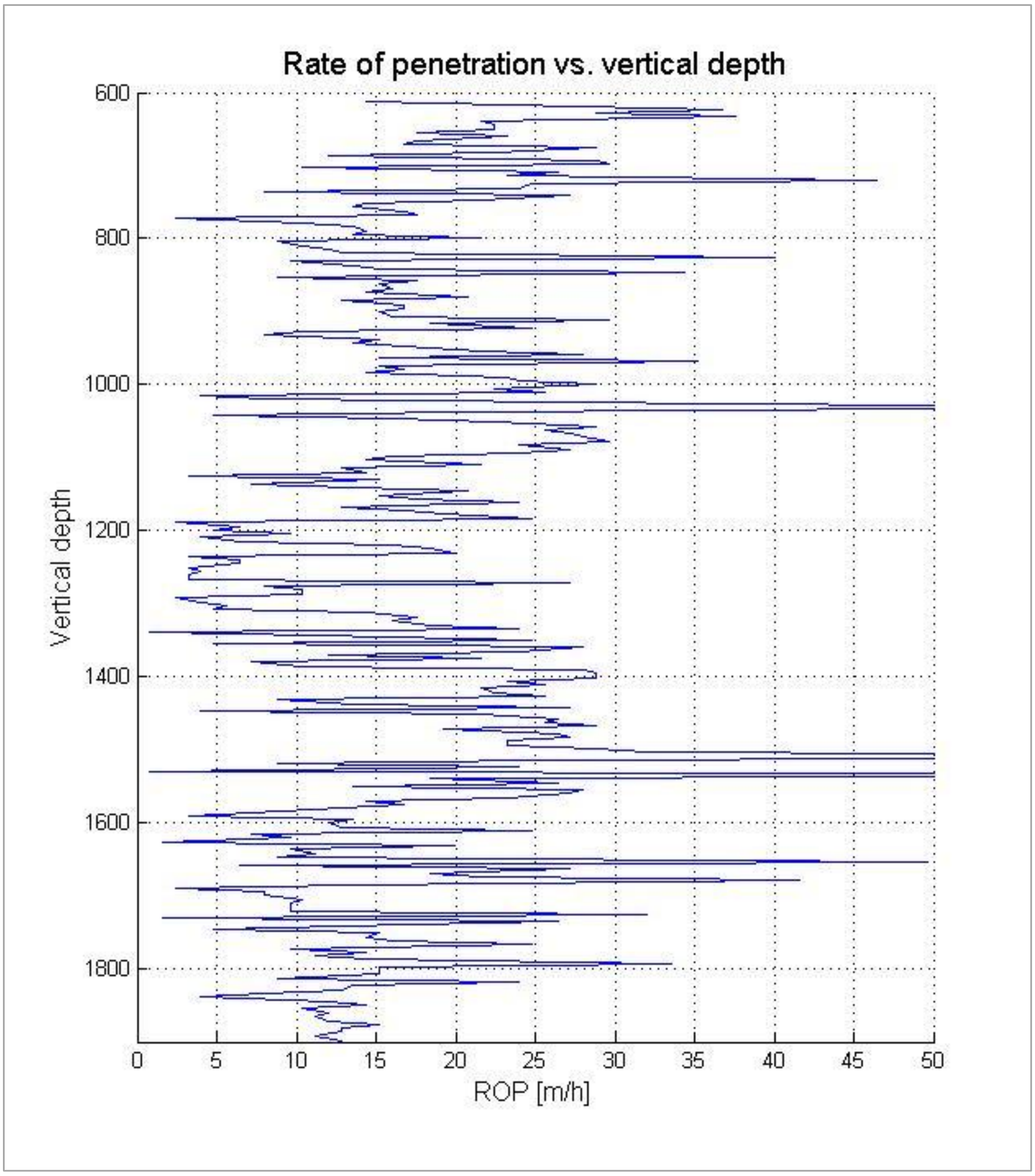


Figure A-6: ROP before data boundaries have been applied and average values created. The 5 m interval between plotted data points is still included.

A.1.4 Mud weight and ECD

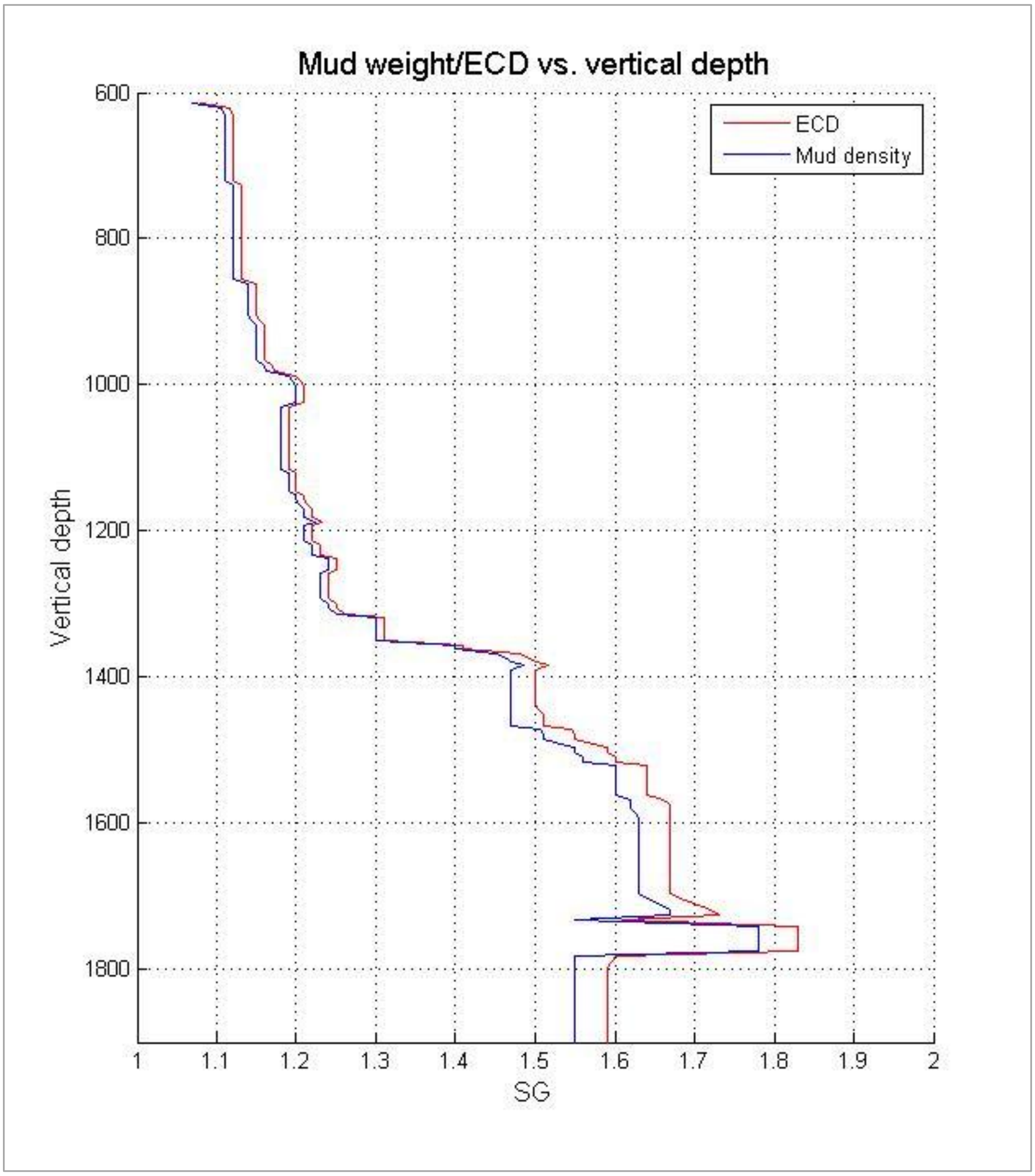


Figure A-7: A comparison of the ECD gradient with the mud weight gradient whilst not circulating.

A.2 d_c-exponent

The plots presented in this section will further illustrate how the slope of the trend line varies with different depth intervals selected for its calculation. A plot showing the d_c-exponent calculated using data without boundaries or average values is also included.

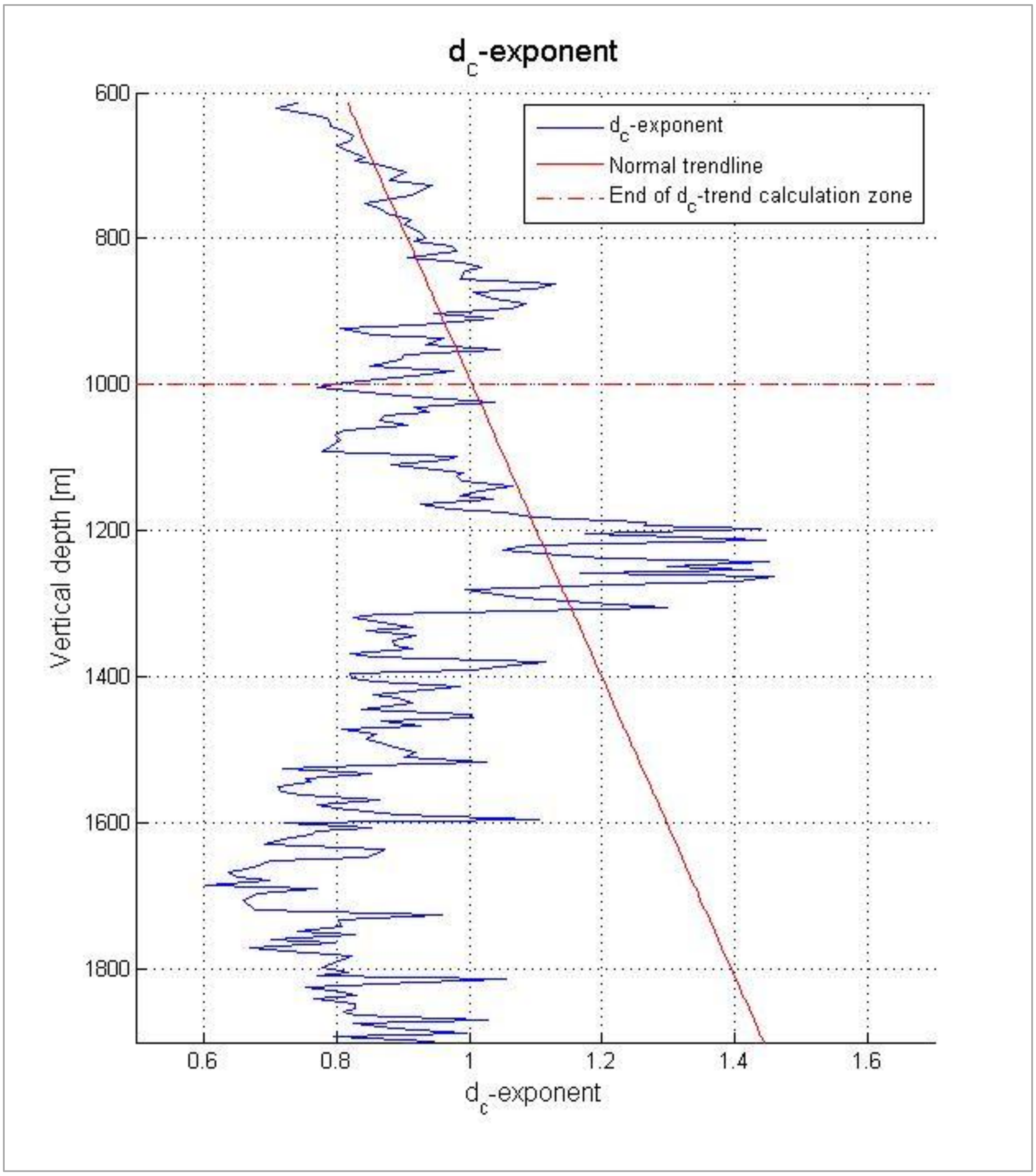


Figure A-8: d_c-exponent plot from Matlab program. Trendline is established on basis of assumed normal conditions in the 612 – 1000 m RKB interval.

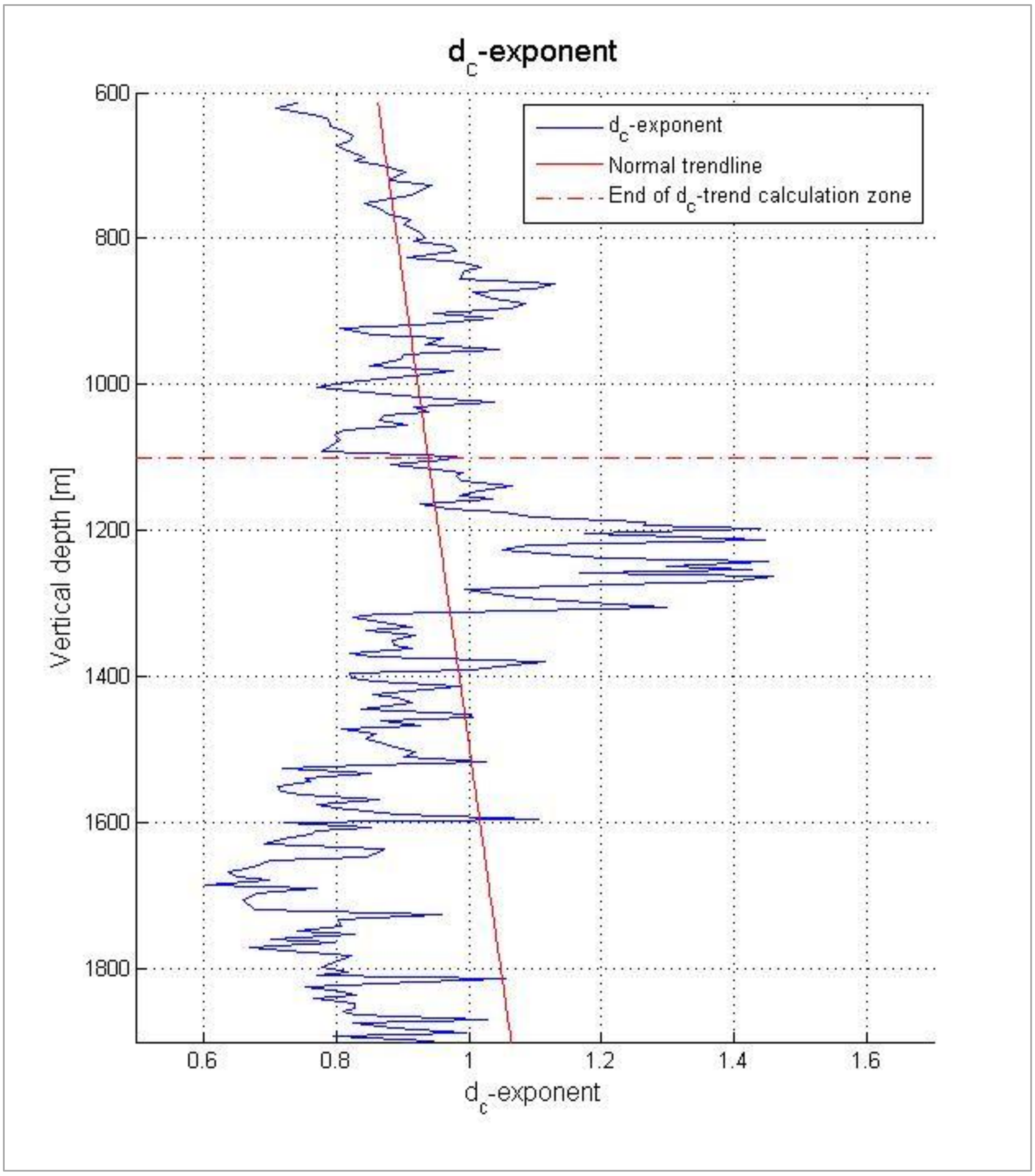


Figure A-9: d_c-exponent plot from Matlab program. Trendline is established on basis of assumed normal conditions in the 612 – 1100 m RKB interval.

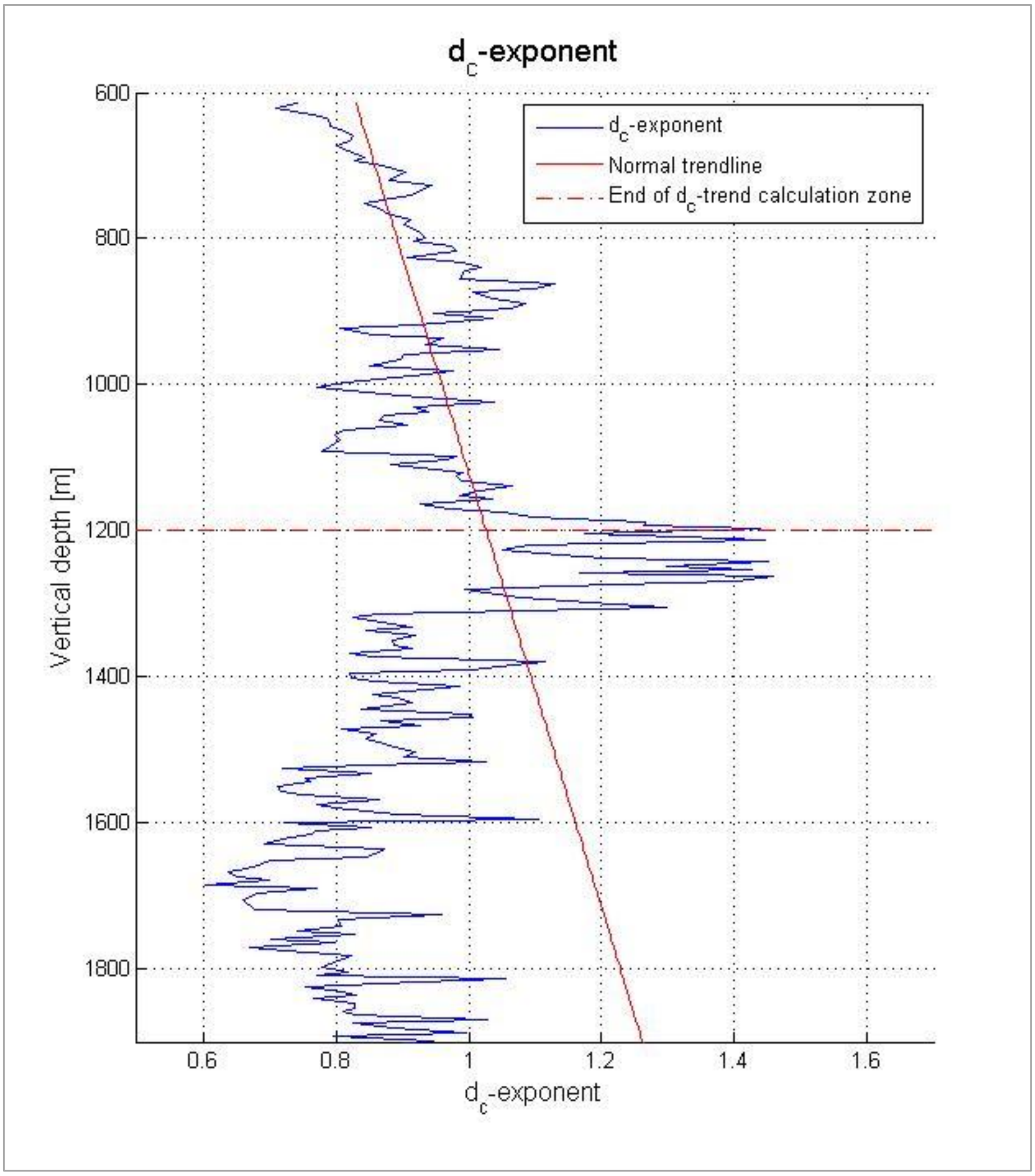


Figure A-10: d_c-exponent plot from Matlab program. Trendline is established on basis of assumed normal conditions in the 612 – 1200 m RKB interval.

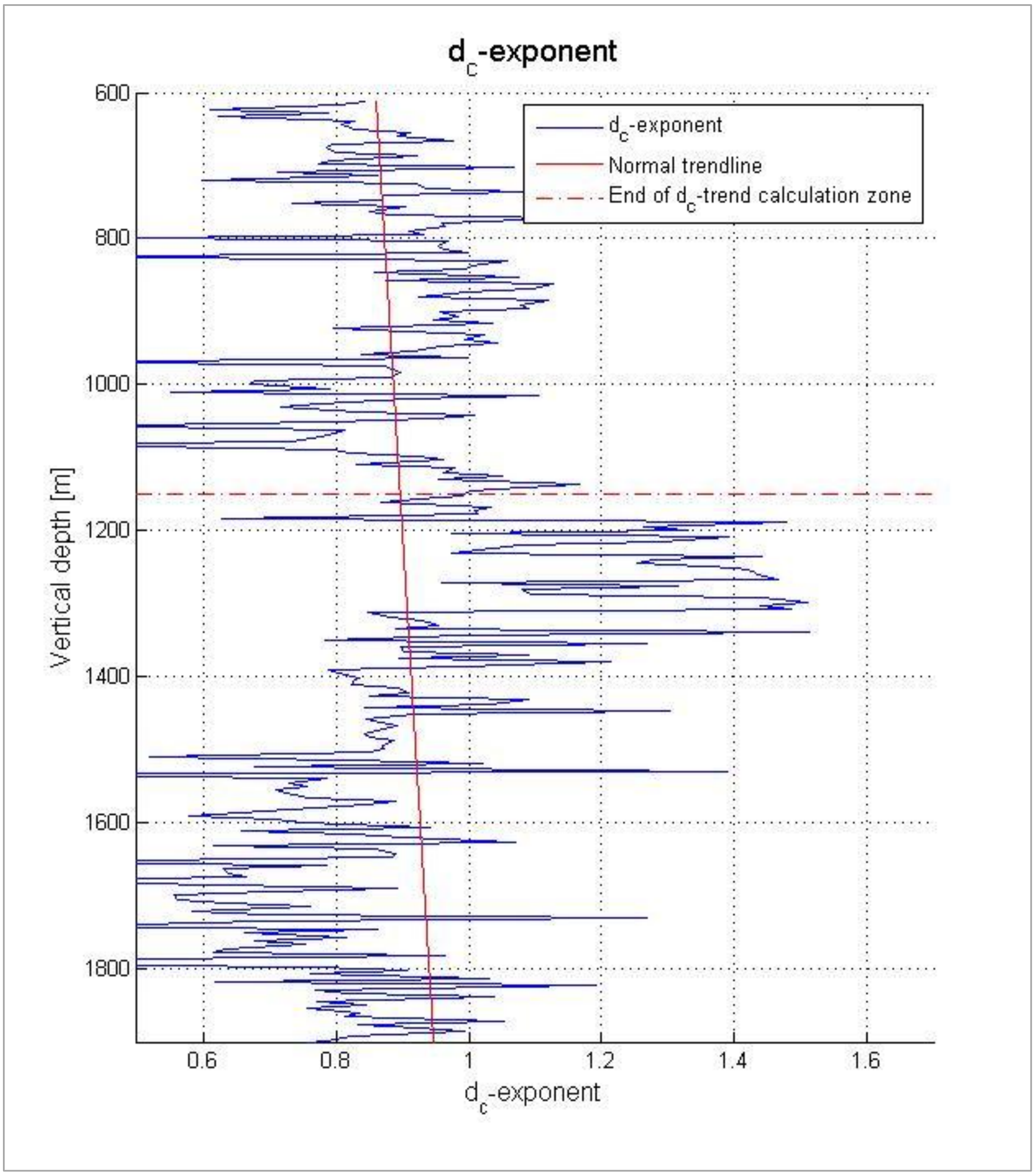


Figure A-11: d_c-exponent plot from Matlab program. Trendline is established on basis of assumed normal conditions in the 612 – 1150 m RKB interval. The input data parameters have not had boundaries applied or average values created. The 5 m interval between plotted data points is still included.

A.3 Zamora's method

The plots presented in this section will further illustrate how the different depth intervals selected for trend line calculation influence the result of Zamora's method. A plot showing the pressure gradient calculated using data without boundaries or average values is also included.

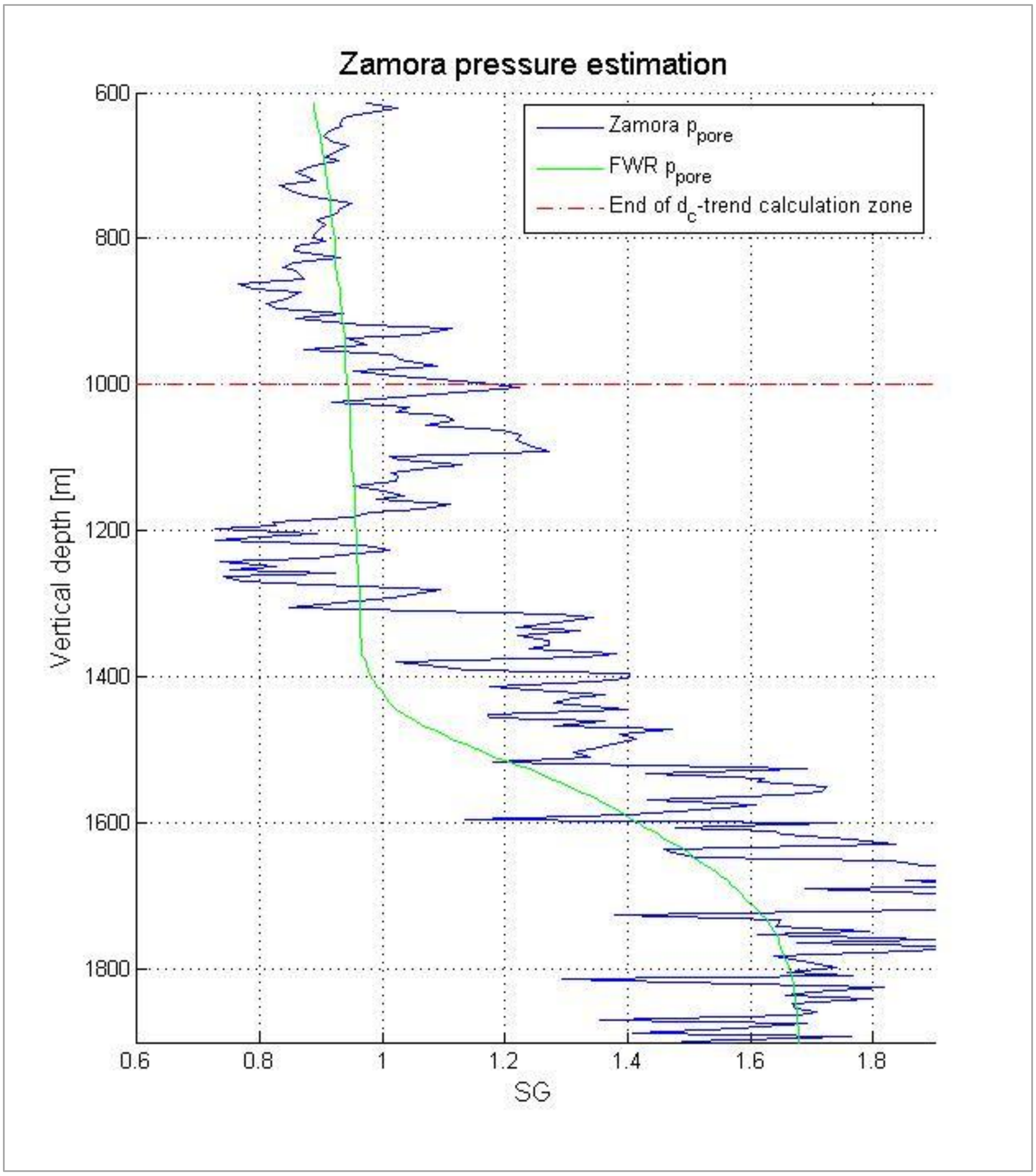


Figure A-12: Zamora pressure gradient, with d_C-exponent trendline interval of 612 – 1000 m RKB, compared with pore pressure reported by Statoil.

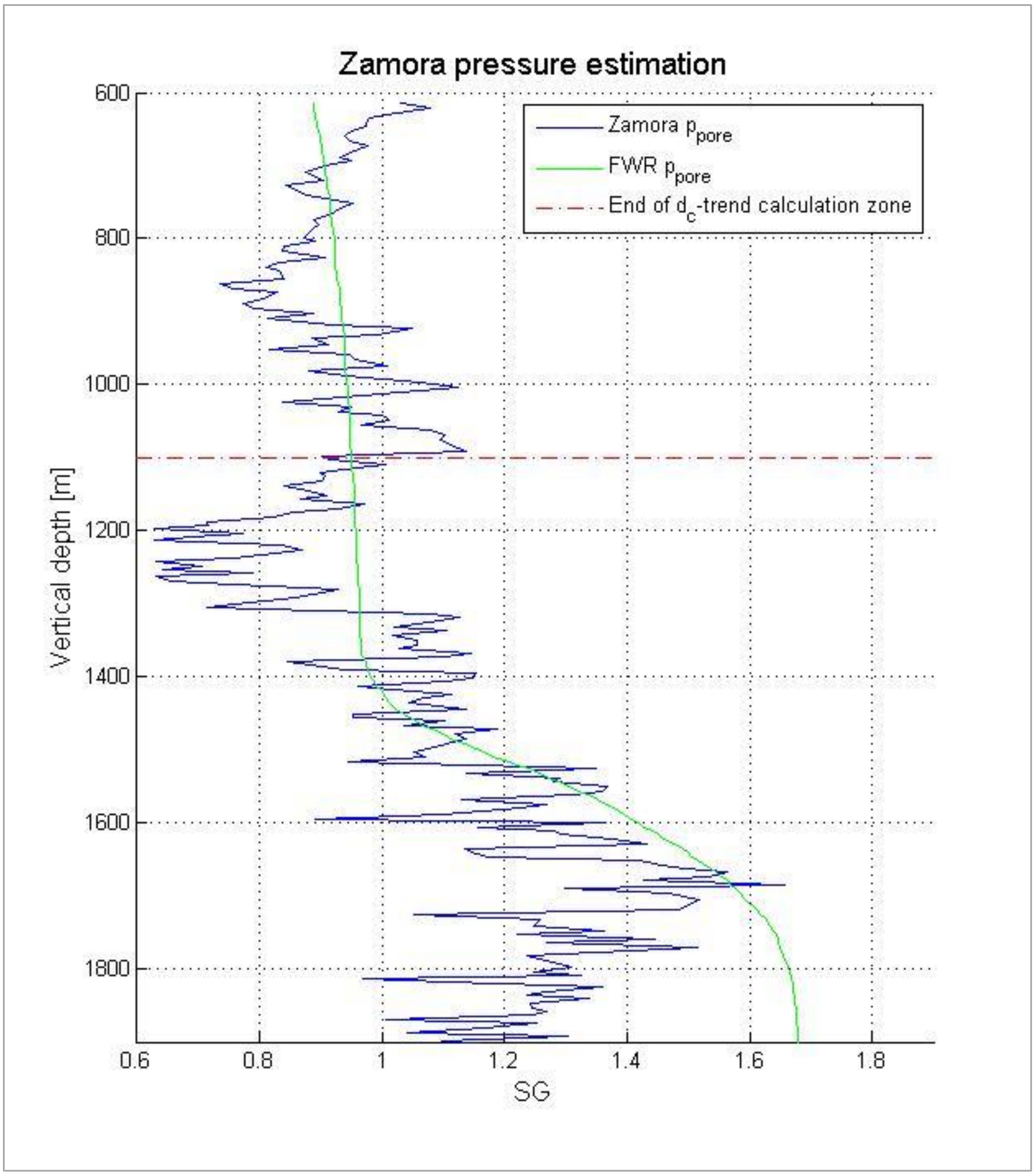


Figure A-13: Zamora pressure gradient, with d_c-exponent trendline interval of 612 – 1100 m RKB, compared with pore pressure reported by Statoil.

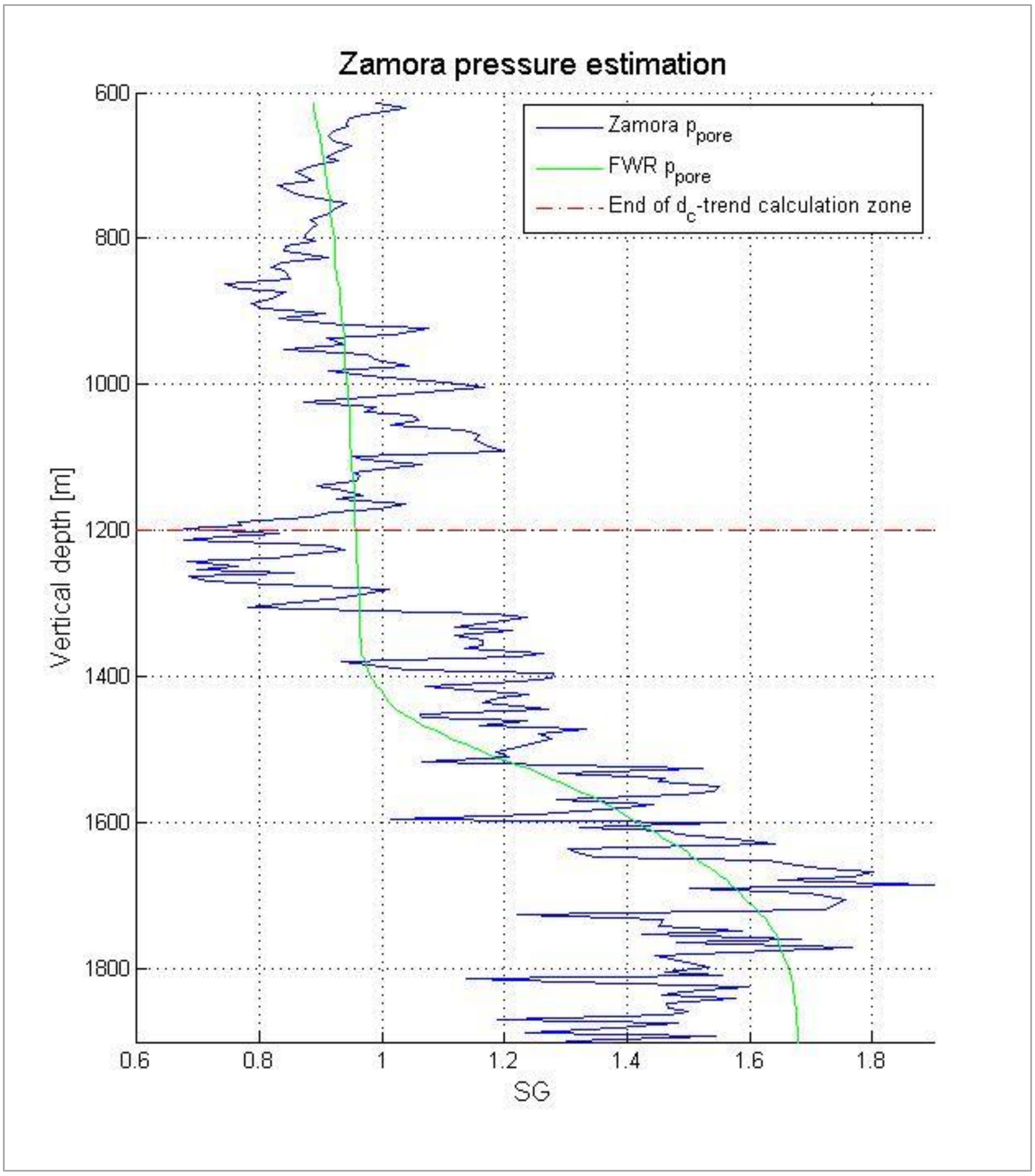


Figure A-14: Zamora pressure gradient, with d_c-exponent trendline interval of 612 – 1200 m RKB, compared with pore pressure reported by Statoil.

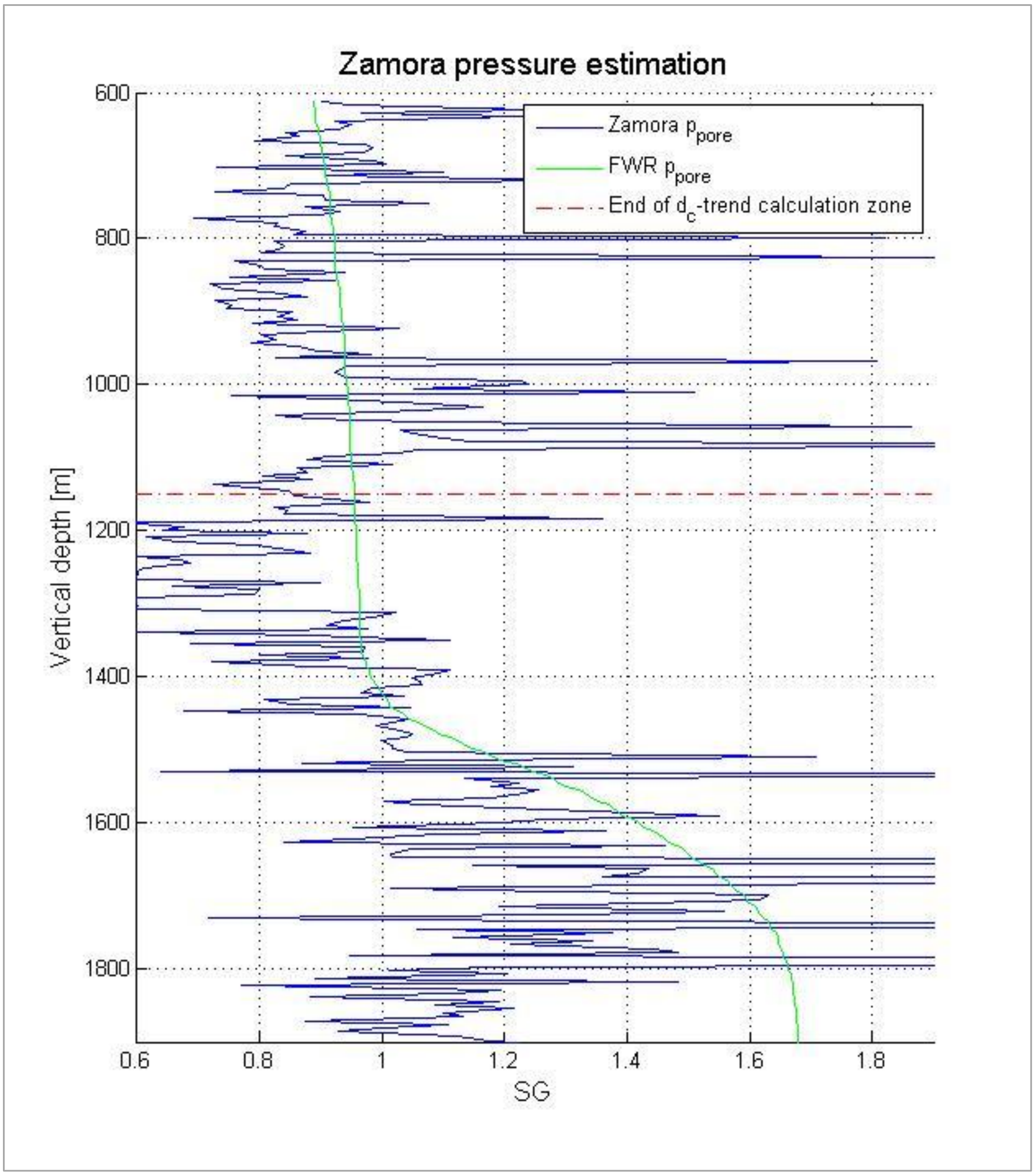


Figure A-15: Zamora pressure gradient, with d_c-exponent trendline interval of 612 – 1150 m RKB, compared with pore pressure reported by Statoil. The input data parameters have not had boundaries applied or average values created. The 5 m interval between plotted data points is still included.

A.4 Eaton's method

The plots presented in this section will further illustrate how the different depth intervals selected for trend line calculation influence the result of Eaton's method. A plot showing the pressure gradient calculated using data without boundaries or average values is also included.

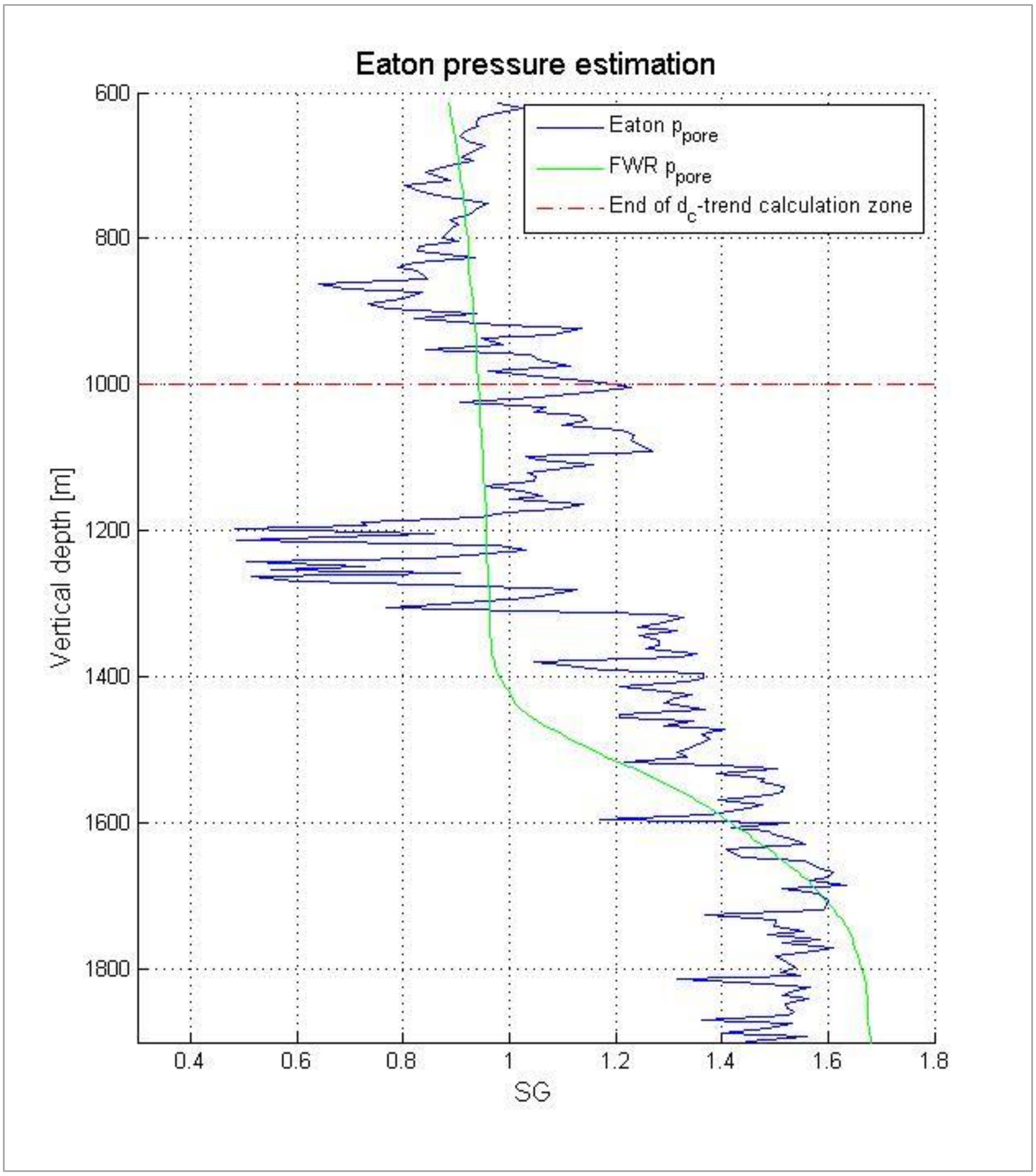


Figure A-16: Eaton pressure gradient, with d_c-exponent trendline interval of 612 – 1000 m RKB, compared with pore pressure reported by Statoil.

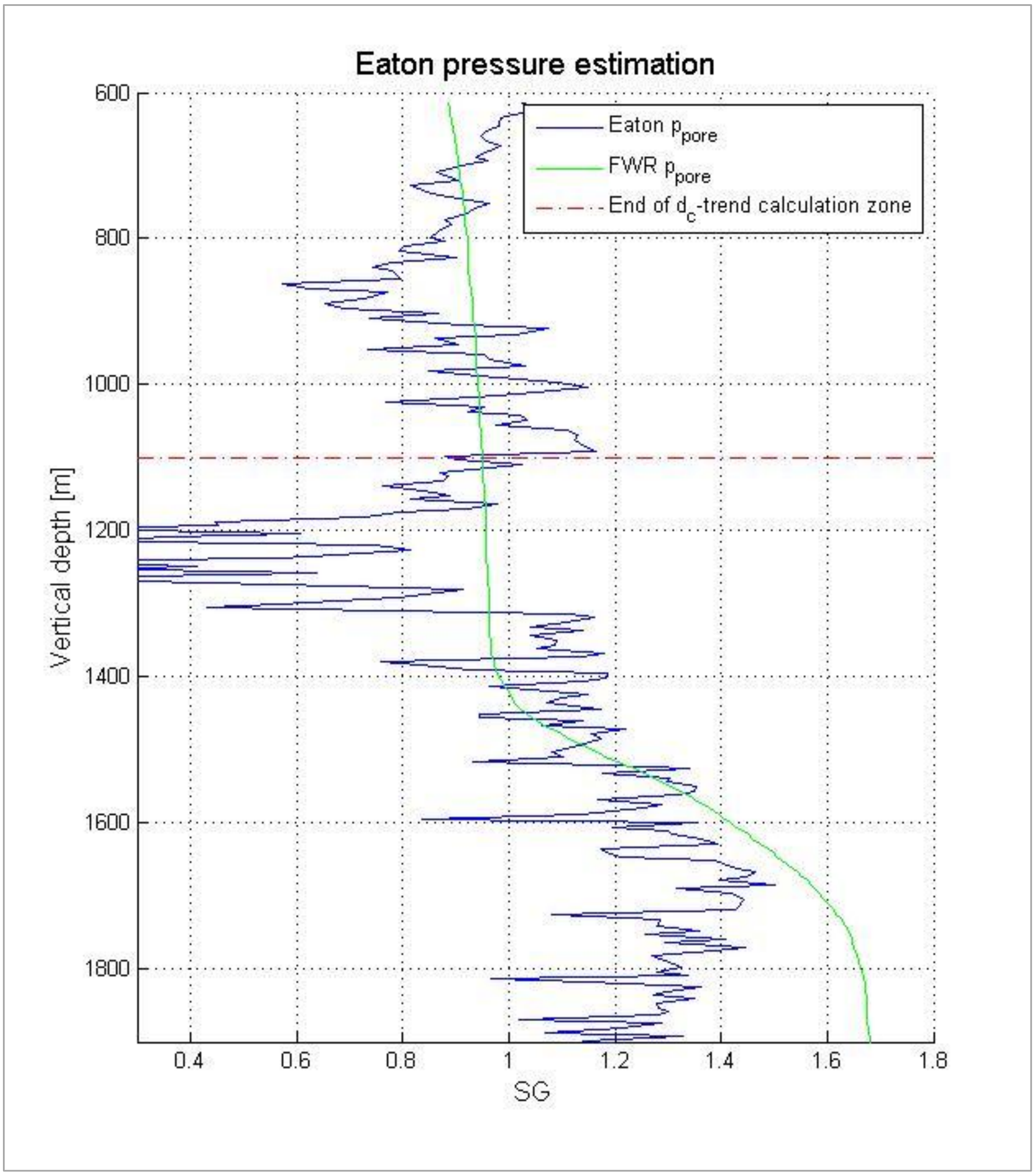


Figure A-17: Eaton pressure gradient, with d_c-exponent trendline interval of 612 – 1100 m RKB, compared with pore pressure reported by Statoil.

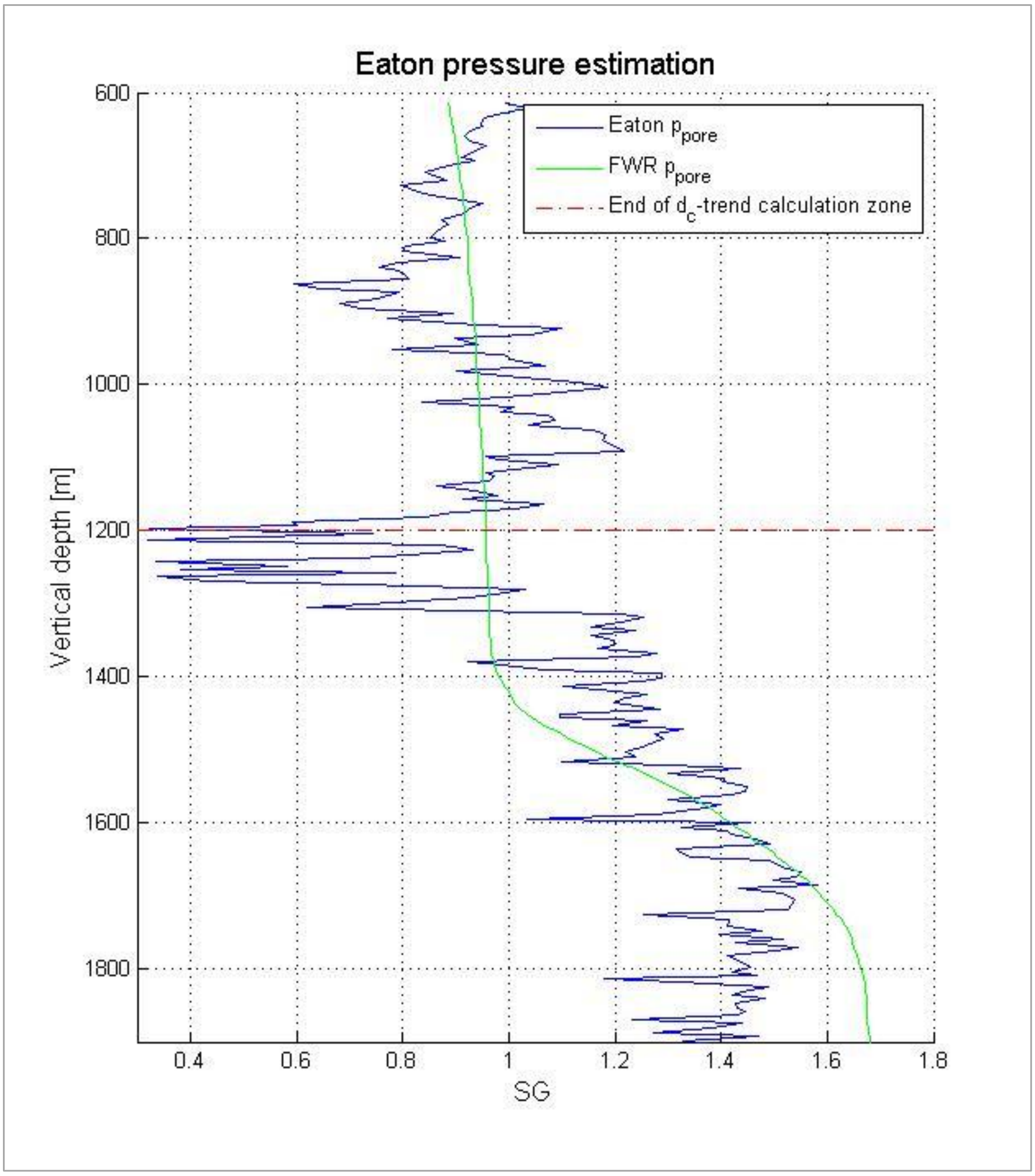


Figure A-18: Eaton pressure gradient, with d_c-exponent trendline interval of 612 – 1200 m RKB, compared with pore pressure reported by Statoil.

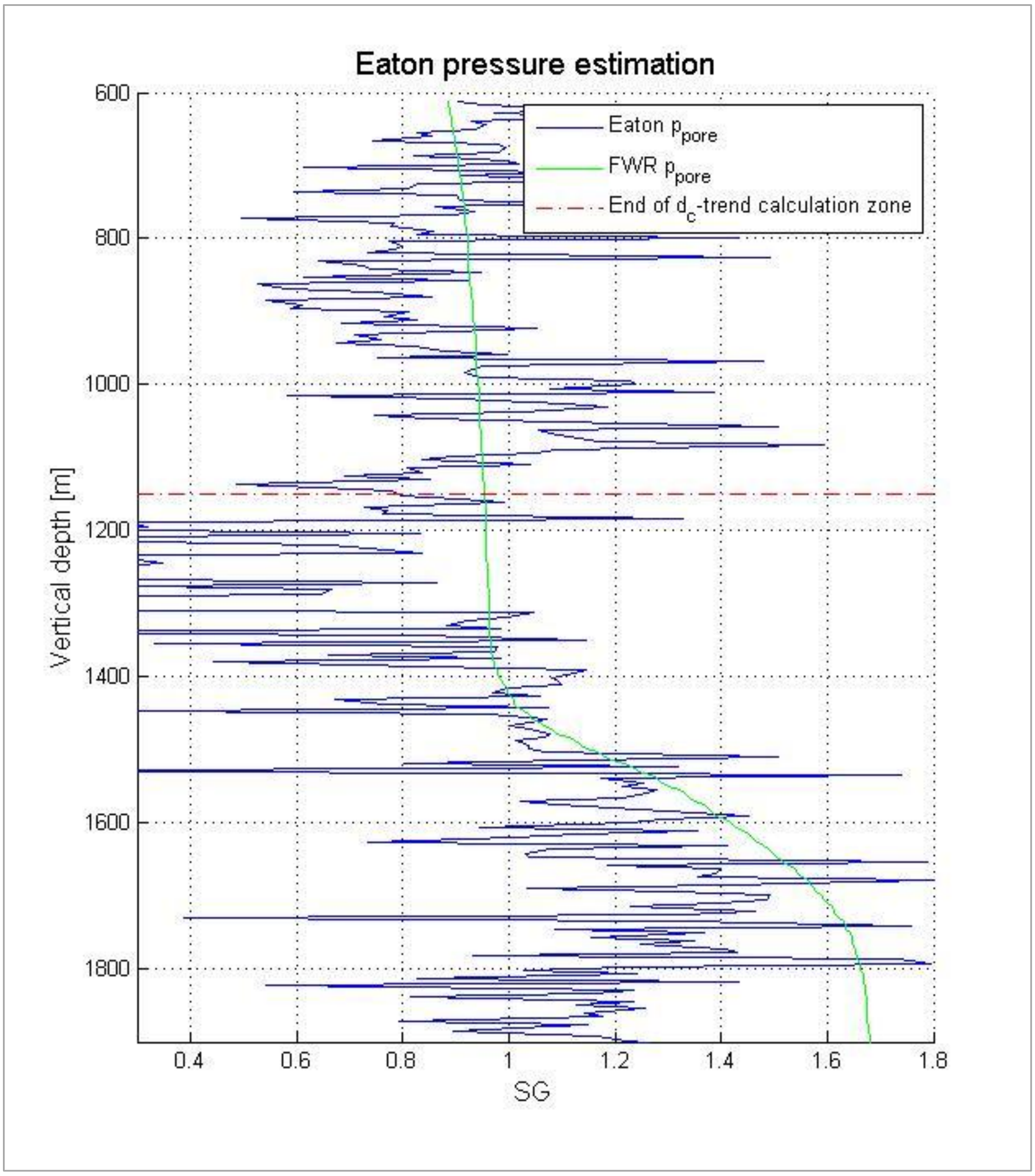


Figure A-19: Eaton pressure gradient, with d_c-exponent trendline interval of 612 – 1150 m RKB, compared with pore pressure reported by Statoil. The input data parameters have not had boundaries applied or average values created. The 5 m interval between plotted data points is still included.

A.5 Bourgoyne-Young drilling model

Plots presented in this section will illustrate the result of different methods that have been used to estimate the drillability factor, and how this influences the calculation of the pore pressure gradient.

A.5.1 Drillability

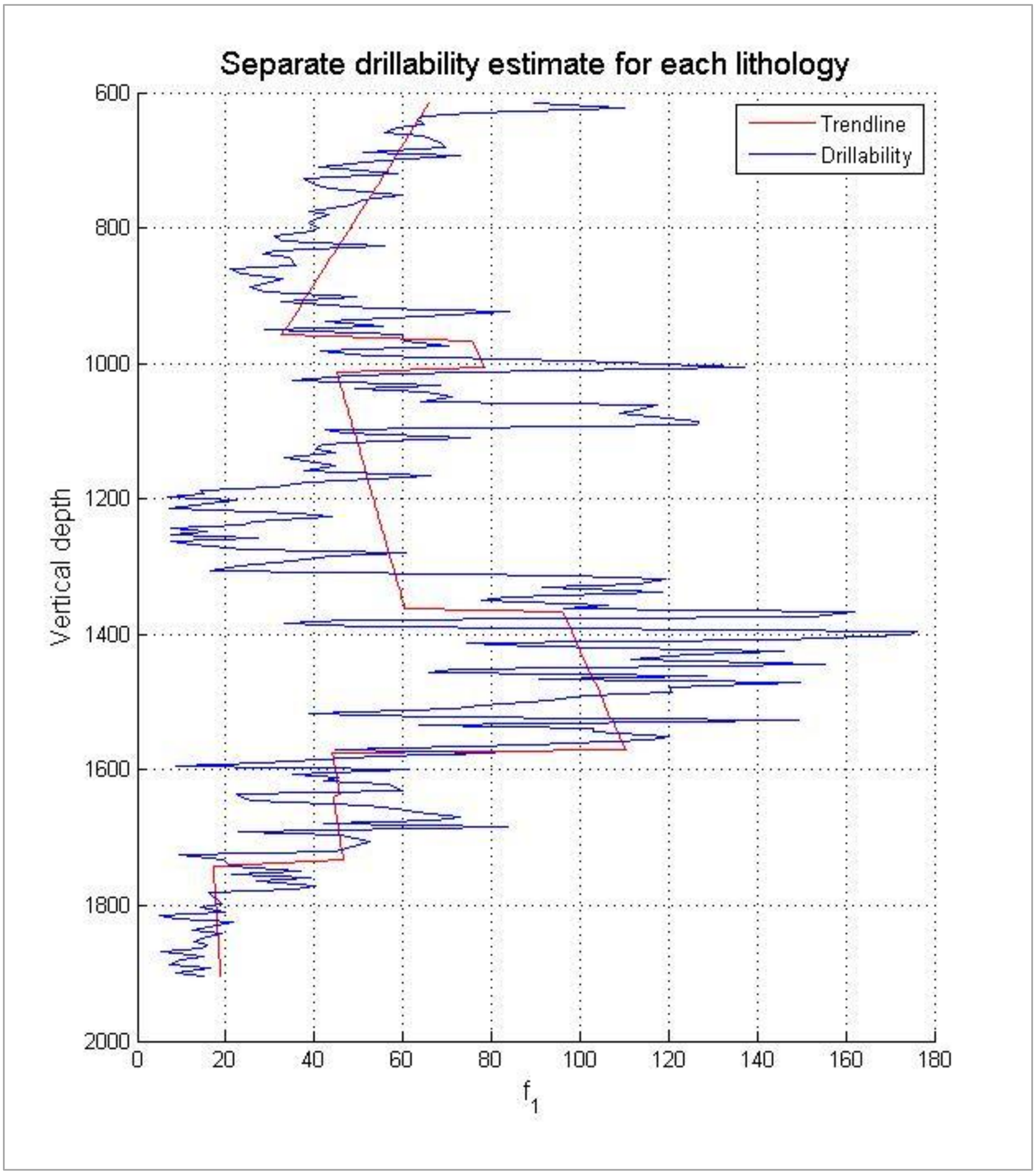


Figure A-20: Drillability factor, with separate linear estimates for each lithological zone encountered in the well.

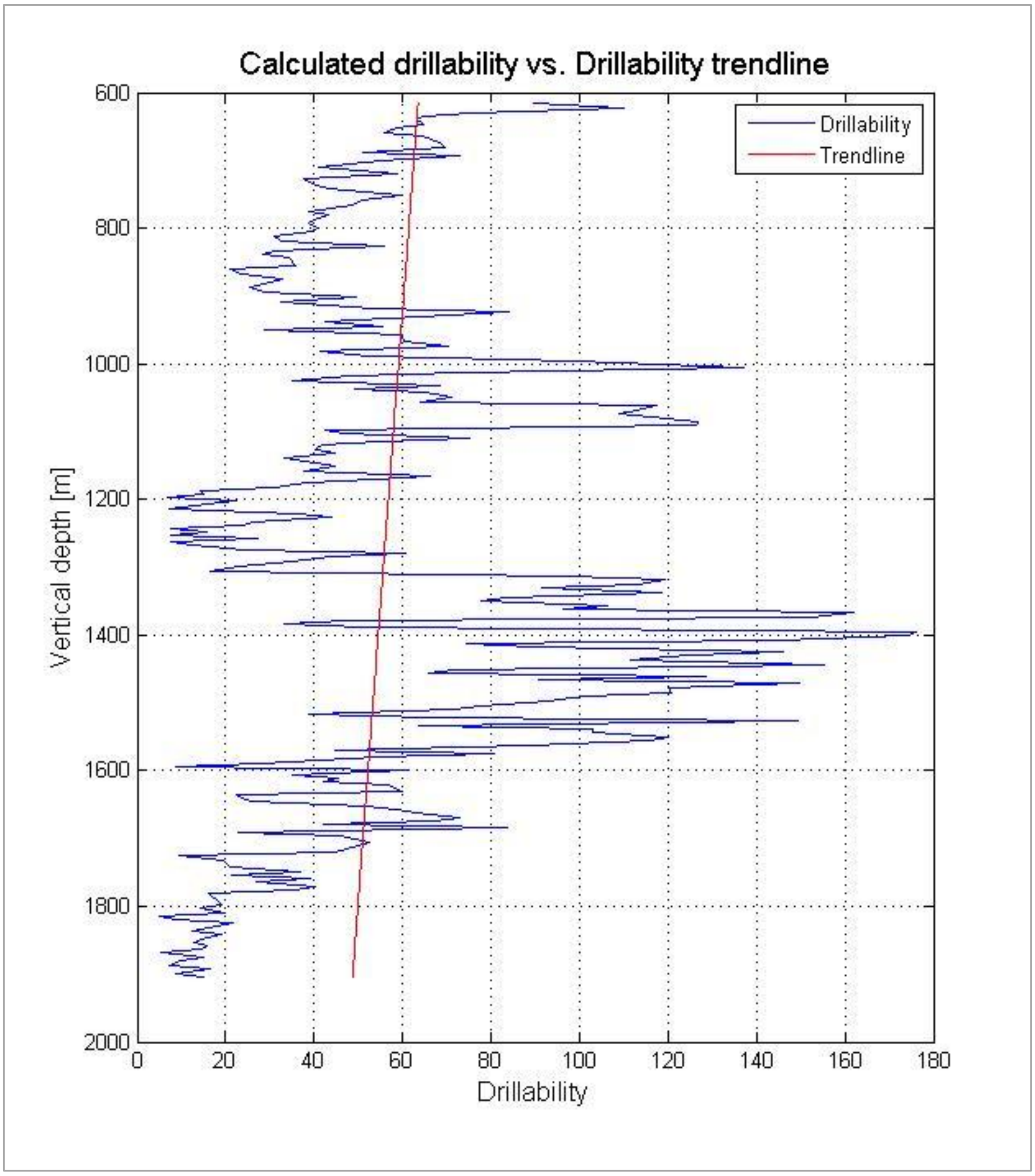


Figure A-21: Drillability factor, with one linear estimate for the full length of the well.

A.5.2 Pressure gradient

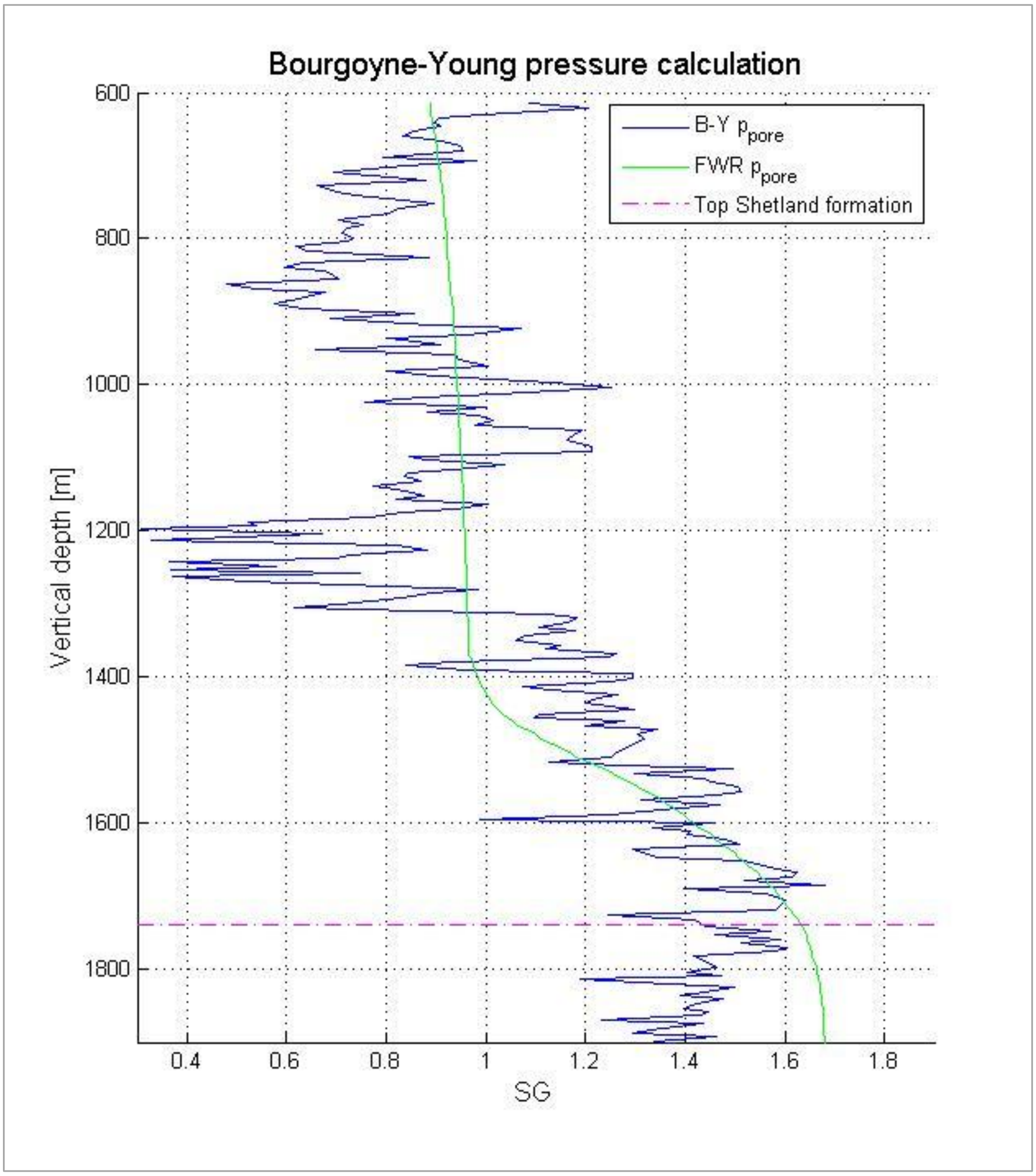


Figure A-22: Bourgoyne-Young pressure gradient compared with pore pressure reported by Statoil. The drillability factor used in this calculation is shown in Figure A-21.

A.6 Standard deviation

This section contains plots illustrating calculated standard deviation for each of the applied methods of pore pressure detection. Due to the deviation observed repeatedly in the sandy Hordaland group member, this depth interval has been neglected when calculating standard deviation.

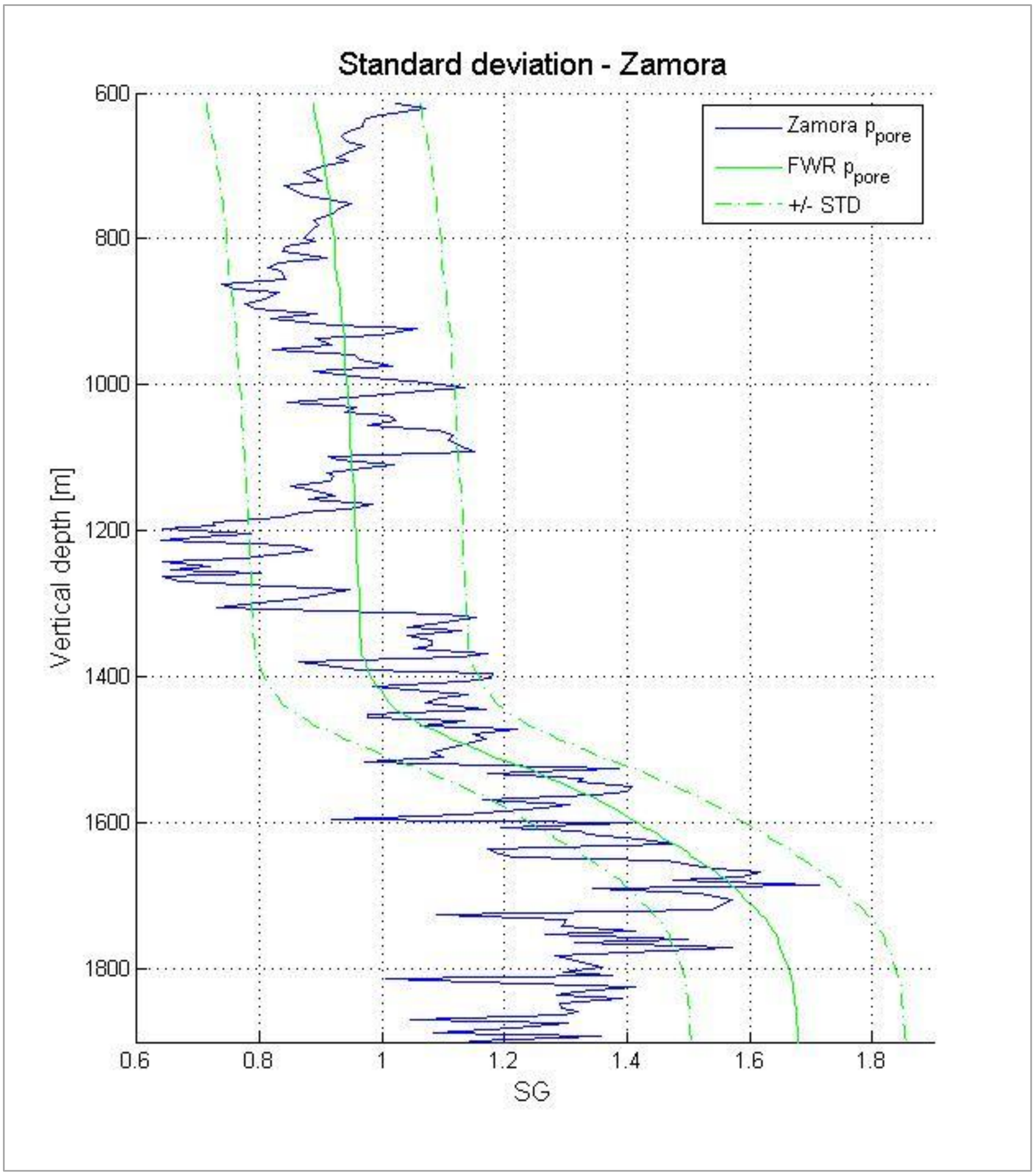


Figure A-23: Zamora pressure gradient compared to pressure values provided in final well report, with a standard deviaton of 0,1747 SG.

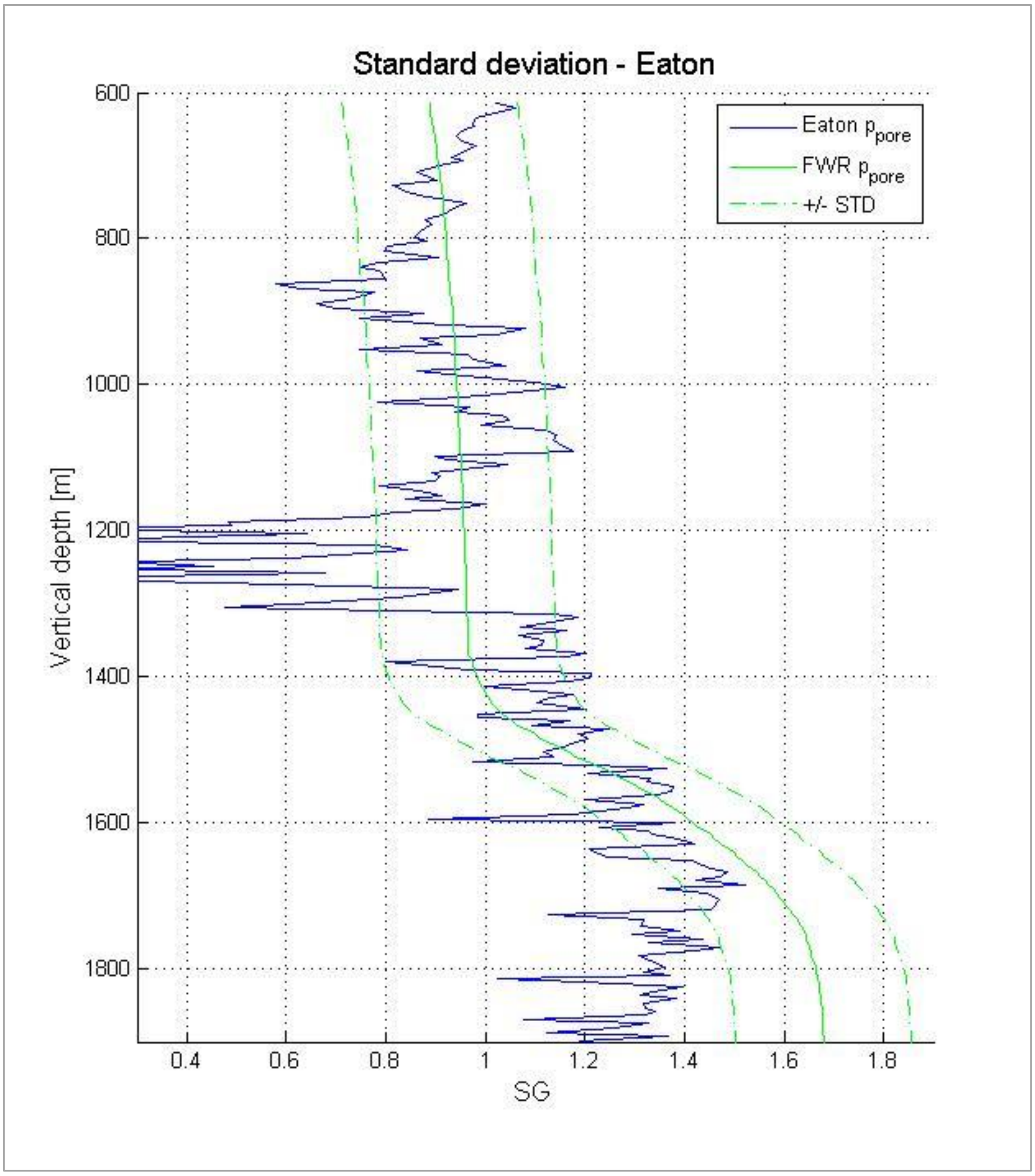


Figure A-24: Eaton pressure gradient compared to pressure values provided in final well report, with a standard deviaton of 0,1765 SG.

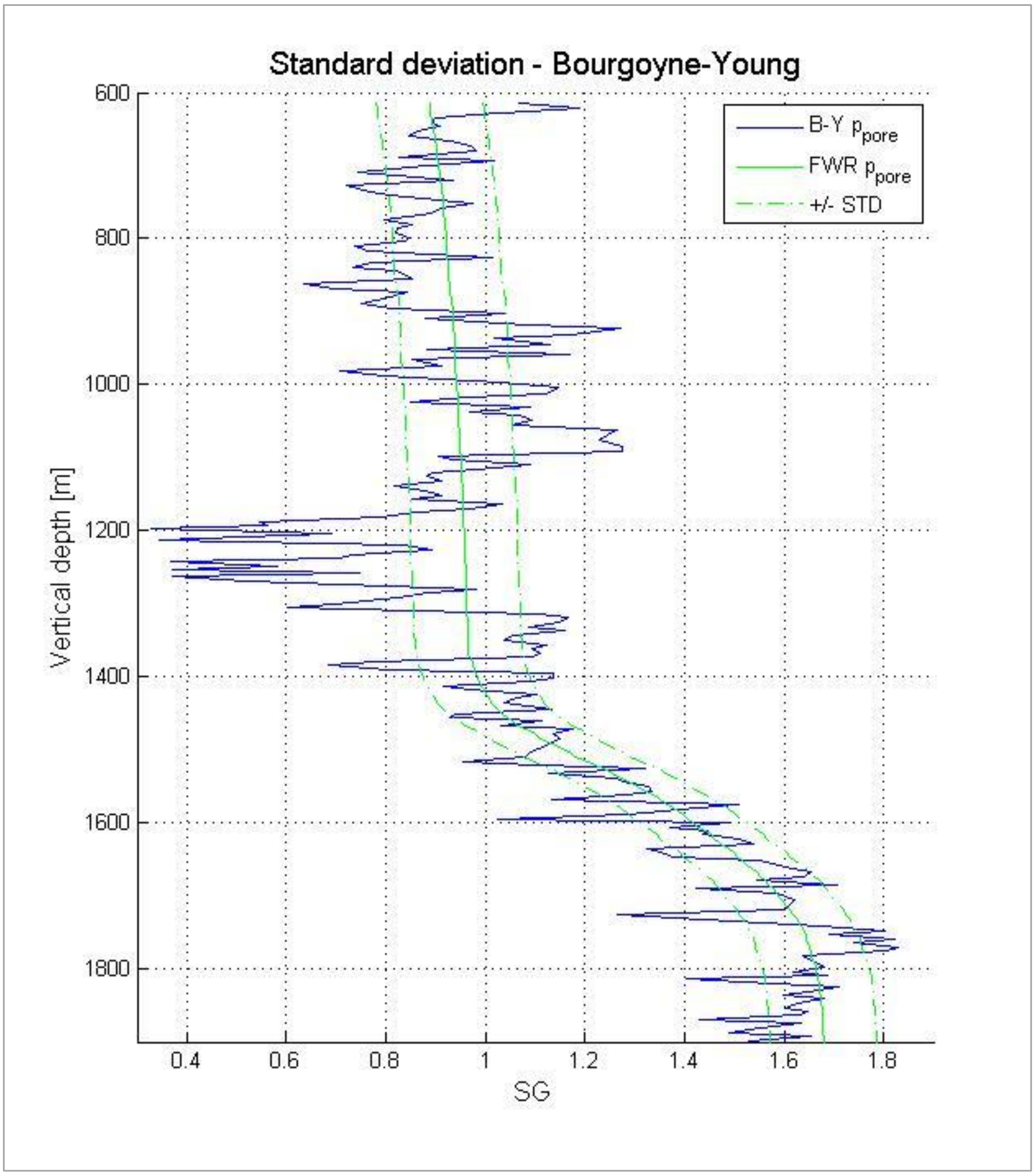


Fig A-25: Bourgoyne-Young pressure gradient compared to pressure values provided in final well report, with a standard deviaton of 0,1071 SG.

B Matlab Code

B.1 Main script

```matlab
% On-line pore pressure detection tool %

% Created by Sindre Stunes (2012)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

clc
clear all
close all

%Assign locations for data directory and matlab work directory:

data_dir = 'C:\Users\Sindre\Documents\MATLAB\RTDD_mme\Data';

work_dir = 'C:\Users\Sindre\Documents\MATLAB\RTDD_mme\Prosjekt';

%DATA IMPORT FROM FILES

%Drilling data
%Input of file names, must be of file format HDF5

%24" section
file{1} = 'LT112805.ASC.h5';
file{2} = 'LT113005.ASC.h5';
file{3} = 'LT120105.ASC.h5';
file{4} = 'LT120205.ASC.h5';
file{5} = 'LT120305.ASC.h5';
file{6} = 'LT120405.ASC.h5';
file{7} = 'LT120505.ASC.h5';
file{8} = 'LT120605.ASC.h5';
file{9} = 'LT120705.ASC.h5';

%17,5" section
file{10} = 'LT122205.ASC.h5';
file{11} = 'LT122305.ASC.h5';
file{12} = 'LT122405.ASC.h5';
file{13} = 'LT122505.ASC.h5';
file{14} = 'LT123105.ASC.h5';
file{15} = 'LT010106.ASC.h5';
file{16} = 'LT010206.ASC.h5';
file{17} = 'LT010406.ASC.h5';
file{18} = 'LT010506.ASC.h5';

%12,25" section
file{19} = 'LT012606.ASC.h5';
file{20} = 'LT012706.ASC.h5';
file{21} = 'LT020706.ASC.h5';
file{22} = 'LT020806.ASC.h5';
file{23} = 'LT020906.ASC.h5';

%8,5" section
file{24} = 'LT022506.ASC.h5';
file{25} = 'LT022606.ASC.h5';
file{26} = 'LT022706.ASC.h5';
```

```
file{27} = 'LT022806.ASC.h5';
file{28} = 'LT030106.ASC.h5';
file{29} = 'LT030206.ASC.h5';
file{30} = 'LT030306.ASC.h5';
file{31} = 'LT030406.ASC.h5';
file{32} = 'LT030506.ASC.h5';
file{33} = 'LT030606.ASC.h5';

%Total number of files to be imported:
file_len = length(file);


%Pressure data from Final Well report

%Name of Excel-file:
file_excel = 'trykk_FWR_digital.xlsx';


% VARIOUS OTHER INPUT, to be read by functions:

% Boundaries for data removal, used by funk_dataclean:
low_wob = 5;
high_wob = 30;

low_rpm = 100;
high_rpm = 250;

low_rop = 0;
high_rop = 50;

low_dver = 600;
high_dver = 2000;

% Desired length interval beetween data points, used by funk_depth:
d_int = 4.99;

% Max depth for dc-trend line creation:
d_trend = 1100 ;

% Depth interval for drillability trend line creation:
d_trend_Kmin = 600; %600 ;
d_trend_Kmax = 1900; %1150 ;

% Drilling coefficients:

%a1 = 0.5;
a2 = 0.000090;
a3 = 0.000100;
a4 = 0.000035;
a5 = 0.9;
a6 = 0.5;
a7 = 0.3;
a8 = 0.4;
```

```
% IMPORT AND DATA PROCESSING:

% Calling function for import of data from files
[len_tot rpm_tot wob_tot bpos_tot dmea_tot dver_tot bdia_tot mdi_tot
ecd_tot]  = funk_import2(file, file_len, data_dir, work_dir);

%Funktion for importing Excel data:
[dver_fwr ppore_fwr ovb_fwr norm_fwr normfit_fwr] = funk_import_excelFWR
(file_excel, data_dir, work_dir);

% Calling function to create correct ROP-data
[len_corr rpm_corr wob_corr rop_corr dmea_corr dver_corr bdia_corr mdi_corr
ecd_corr] = funk_data_ropcorr(len_tot, rpm_tot, wob_tot, bpos_tot, ...
            dmea_tot, dver_tot, bdia_tot, mdi_tot, ecd_tot);

% Calling function for removal of "non-drilling" data
[len_cl rpm_cl wob_cl rop_cl dmea_cl dver_cl bdia_cl mdi_cl ecd_cl] =
funk_data_clean(low_wob, high_wob, low_rpm, high_rpm, low_rop, ...
high_rop, low_dver, high_dver, len_corr, rpm_corr, wob_corr, ...
rop_corr, dmea_corr, dver_corr, bdia_corr, mdi_corr, ecd_corr);

% Function to align FWR data with field drilling data
[pporefwr_cl ovbfwr_cl normfwr_cl normfit_cl] = funk_data_fwralign
(dver_fwr, ppore_fwr, ovb_fwr, norm_fwr, normfit_fwr, dver_cl, len_cl);

% Calling function to combine 3 data-points
[len_evn rpm_evn wob_evn rop_evn dmea_evn dver_evn bdia_evn mdi_evn ecd_evn
pporefwr_evn ovbfwr_evn normfwr_evn normfit_evn] = funk_even54(len_cl, ...
rpm_cl, wob_cl, rop_cl, ...dmea_cl, dver_cl, bdia_cl, ...mdi_cl, ecd_cl,
pporefwr_cl, ...ovbfwr_cl, normfwr_cl, normfit_cl) ;

% Calling function to create per meter intervall between data points
[len_dpt rpm_dpt wob_dpt rop_dpt dmea_dpt dver_dpt bdia_dpt mdi_dpt ecd_dpt
pporefwr_dpt ovbfwr_dpt normfwr_dpt normfit_dpt] = funk_data_depth(d_int,
len_evn, ...rpm_evn, wob_evn, rop_evn, ...dmea_evn, dver_evn, bdia_evn, ...
mdi_evn, ecd_evn, pporefwr_evn, ...ovbfwr_evn, normfwr_evn, normfit_evn);

%CALCULATIONS:

% Creating the dc trend line
[len_trend coeff] = funk_calc_dctrend(d_trend, len_dpt, rop_dpt, rpm_dpt,
wob_dpt, bdia_dpt, mdi_dpt, dver_dpt, normfit_dpt);

% Calling function to calculate the dc-exponent
[dcc dc] = funk_calc_dcc(len_dpt, rop_dpt, rpm_dpt, wob_dpt, bdia_dpt,
mdi_dpt, normfit_dpt);

% Calling function to calculate pressure gradient, based on Zamora method
[p_zamora] = funk_calc_zamora (dcc, dver_dpt, normfit_dpt, len_dpt, coeff);

% Function to determine pore pressure from eaton’s method
[p_eaton] = funk_calc_eaton (dver_dpt, ovbfwr_dpt, normfit_dpt, dcc, coeff,
len_dpt);

% Calculating factors of B&Y equation
[f1 f2 f3 f4 f5 f6] = funk_calc_BYfactor (pporefwr_dpt, dver_dpt, rop_dpt,
wob_dpt, bdia_dpt, rpm_dpt, ecd_dpt, a2, a3, a4, a5, a6, len_dpt);
```

```
% Creating the drillability trend line
[coeff_K drill_K dver_trend_K] = funk_calc_avK (dver_dpt, rop_dpt, f1,
len_dpt, d_trend_Kmin, d_trend_Kmax);

% Creating linear trends of rop, f1, k5, k6 for each formation zone:
[rop_lin f1_lin f5_lin f6_lin] = test_lin_zones (dver_dpt, rop_dpt, f1, f5,
f6, len_dpt );

% Calculating pressure gradient from B&Y eq.
[p_by k] = funk_calc_BYf3 (coeff_K, dver_dpt, rop_dpt, f2, f5, f6, ecd_dpt,
a3, a4, len_dpt);

% Calculating pressure gradient from B&Y eq. WITH LINNEAR ESTIMATES OF
% EITHER ROP, f1, f5, and/or f6 for each formation zone
%[p_by] = test_zonelin_BYf3 (dver_dpt, rop_dpt, f1_lin, f2, f5, f6,
ecd_dpt, a3, a4, len_dpt);

%Calculating standard deviation of the various pressure gradient estimates
%vs true pressure
[std_eaton std_zamora std_by] = funk_calc_std (dver_dpt, pporefwr_dpt,
p_eaton, p_zamora, p_by, len_dpt);




%PLOT OUTPUT:

% Calling function for dc-plot creation
funk_plot_dc(dcc, dver_dpt, coeff);

% Calling function for zamora pressure-plot creation
funk_plot_zamora (p_zamora, dver_dpt, pporefwr_dpt);

% Calling function for Eaton pressure-plot creation
funk_plot_eaton (dver_dpt, pporefwr_dpt, p_eaton);

% Calling function for B&Y pressure-plot creation
funk_plot_BY (p_by, dver_dpt, ecd_dpt, mdi_dpt , pporefwr_dpt);

% Function for Drillability trend-plot creation
funk_plot_K (dver_dpt, dver_trend_K, drill_K, coeff_K);

%Plotting all estimated pore pressures with their respective standard
%deviation
funk_plot_std (dver_dpt, std_eaton, std_zamora, std_by, pporefwr_dpt,
p_eaton, p_zamora, p_by, len_dpt);

% Plotting B&Y factors
funk_plot_f (dver_dpt, rop_dpt, f1_lin, f2, f5, f6);

% Function for creation various feild data graphs
funk_plot_field (dver_dpt, rop_dpt, rpm_dpt, wob_dpt, bdia_dpt, mdi_dpt,
ecd_dpt, pporefwr_dpt, ovbfwr_dpt, normfwr_dpt );

%Plot comparing mdi to erroneus ecd data, to better to be able to estimate
%ecd
funk_plot_ecdvsmdi (dver_dpt, ecd_dpt, mdi_dpt)
```

B.2 Import functions

B.2.1 Function for importing drilling data

```
function [len_tot rpm_tot wob_tot bpos_tot dmea_tot dver_tot bdia_tot
mdi_tot ecd_tot]  = funk_import2(file, file_len, data_dir, work_dir)


%Assign current Matlab working directory to file location
cd(data_dir);


% Calculate total number of data:

arr_len = 0;

for i = ( 1 : file_len)

    rpm_count = h5read(file{i},'/RPMB');

    arr_len = arr_len + length(rpm_count);

    clear('rpm_count');

end

% Allocates the total length variables, for better prossessing

rpm_tot  = zeros(1,arr_len);
wob_tot  = zeros(1,arr_len);
bpos_tot  = zeros(1,arr_len);
dmea_tot = zeros(1,arr_len);
dver_tot = zeros(1,arr_len);
bdia_tot = zeros(1,arr_len);
mdi_tot  = zeros(1,arr_len);
ecd_tot = zeros(1,arr_len);


% Add data from file1 to the total data string:

rpm1 = h5read(file{1},'/RPMB');
wob1 = h5read(file{1},'/WOB');
bpos1 = h5read(file{1},'/BPOS');
dmea1 = h5read(file{1},'/DMEA');
dver1 = h5read(file{1},'/DVER');
bdia1 = h5read(file{1},'/BDIA');
mdi1 = h5read(file{1},'/MDI');
ecd1 = h5read(file{1},'/ECDB');

len = length(rpm1);
len_tot = length(rpm1);

for i = ( 1 : len );

        rpm_tot(i)   = rpm1(i);
        wob_tot(i)   = wob1(i);
        bpos_tot(i)  = bpos1(i);
        dmea_tot(i) = dmea1(i);
        dver_tot(i) = dver1(i);
        bdia_tot(i) = bdia1(i);
        mdi_tot(i)  = mdi1(i);
        ecd_tot(i)  = ecd1(i);

end;


% Add data from the rest of the files to the total data string:



for i = ( 2 : file_len)

    rpm_tmp = h5read(file{i},'/RPMB');
    wob_tmp = h5read(file{i},'/WOB');
    bpos_tmp = h5read(file{i},'/BPOS');
    dmea_tmp = h5read(file{i},'/DMEA');
    dver_tmp = h5read(file{i},'/DVER');
    bdia_tmp = h5read(file{i},'/BDIA');
    mdi_tmp = h5read(file{i},'/MDI');
    ecd_tmp = h5read(file{i},'/ECDB');

    len = length(rpm_tmp);

    for j = ( 1 : len )

        rpm_tot(j+len_tot)  = rpm_tmp(j);
        wob_tot(j+len_tot)  = wob_tmp(j);
        bpos_tot(j+len_tot)  = bpos_tmp(j);
        dmea_tot(j+len_tot) = dmea_tmp(j);
        dver_tot(j+len_tot) = dver_tmp(j);
        bdia_tot(j+len_tot) = bdia_tmp(j);
        mdi_tot(j+len_tot)  = mdi_tmp(j);
        ecd_tot(j+len_tot)  = ecd_tmp(j);

    end

    len_tot = len_tot + length(rpm_tmp);

    clear('rpm_tmp');
    clear('wob_tmp');
    clear('bpos_tmp');
    clear('dmea_tmp');
    clear('dver_tmp');
    clear('bdia_tmp');
    clear('mdi_tmp');
    clear('ecd_tmp');

end

%Assign current Matlab working directory back to program location, to be
%able to call functions
cd(work_dir);

return
```

B.2.2 Import of excel data

```
function [dver_fwr ppore_fwr ovb_fwr norm_fwr normfit_fwr] =
funk_import_excelFWR (file_excel, data_dir, work_dir)

%Assign current Matlab working directory to file location
cd(data_dir);


[dver_temp] = xlsread(file_excel,'O10:O421');

[ppore_temp] = xlsread(file_excel,'P10:P421');

[ovb_temp] = xlsread(file_excel,'Q10:Q421');

[norm_temp] = xlsread(file_excel,'T10:T421');

[normfit_temp] = xlsread(file_excel,'U10:U421');


arr_len = length(dver_temp);

dver_fwr = zeros(1,arr_len);
ppore_fwr = zeros(1,arr_len);
ovb_fwr = zeros(1,arr_len);
norm_fwr = zeros(1,arr_len);
normfit_fwr = zeros(1,arr_len);

for i = 1 : arr_len;

    dver_fwr(i) = dver_temp(i);
    ppore_fwr(i) = ppore_temp(i);
    ovb_fwr(i) = ovb_temp(i);
    norm_fwr(i) = norm_temp(i);
    normfit_fwr(i) = normfit_temp(i);

end


%Assign current Matlab working directory back to program location, to be
%able to call functions
cd(work_dir);


return
```

B.3 Data processing functions

B.3.1 Function to align data

```
function [pporefwr_cl ovbfwr_cl normfwr_cl normfit_cl] = funk_data_fwralign
(dver_fwr, ppore_fwr, ovb_fwr, norm_fwr, normfit_fwr, dver_cl, len_cl)

%Function aligning the excel data to other field data, with respect to
%vertical depth and array length

pporefwr_cl  = zeros(1,len_cl);
ovbfwr_cl  = zeros(1,len_cl);
normfwr_cl  = zeros(1,len_cl);
normfit_cl  = zeros(1,len_cl);

len_fwr = length(dver_fwr);


for i = (1 : len_cl);
    for j = (1 : len_fwr);

        if dver_cl(i) > dver_fwr(j)

            pporefwr_cl(i) = ppore_fwr(j);
            ovbfwr_cl(i) = ovb_fwr(j);
            normfwr_cl(i) = norm_fwr(j);
            normfit_cl(i) = normfit_fwr(j);

        end
    end
end



return
```

B.3.2 Function for ROP calculation

```
function [len_corr rpm_corr wob_corr rop_corr dmea_corr dver_corr bdia_corr
mdi_corr ecd_corr] = funk_data_ropcorr(len_tot, rpm_tot, wob_tot, bpos_tot,
                     dmea_tot, dver_tot, bdia_tot, mdi_tot, ecd_tot)

% Function for calculation of the new ROP data, from BPOS

k = 1 ;

for i = 1 : 3 : (len_tot-2)

   rpm_evn(k) = ( ( rpm_tot(i) + rpm_tot(i+1) + rpm_tot(i+2) ) / 3 );

   wob_evn(k) = ( ( wob_tot(i) + wob_tot(i+1) + wob_tot(i+2) ) / 3 );

   bpos_evn(k) = ( ( bpos_tot(i) + bpos_tot(i+1) + bpos_tot(i+2) ) / 3 );

   dmea_evn(k) = ( ( dmea_tot(i) + dmea_tot(i+1) + dmea_tot(i+2) ) / 3 );

   dver_evn(k) = ( ( dver_tot(i) + dver_tot(i+1) + dver_tot(i+2) ) / 3 );

   bdia_evn(k) = ( ( bdia_tot(i) + bdia_tot(i+1) + bdia_tot(i+2) ) / 3 );

   mdi_evn(k) = ( ( mdi_tot(i) + mdi_tot(i+1) + mdi_tot(i+2) ) / 3 );

   ecd_evn(k) = ( ( ecd_tot(i) + ecd_tot(i+1) + ecd_tot(i+2) ) / 3 );

   k = k + 1 ;

end

   len_evn = length(rpm_evn);



k = 1;

for i = 1 : (len_evn - 1)

    rop_corr(k) = ( (bpos_evn(i) - bpos_evn(i+1)) / (15 / 3600) );
    rpm_corr(k) = rpm_evn(i);
    wob_corr(k) = wob_evn(i);
    dmea_corr(k) = dmea_evn(i);
    dver_corr(k) = dver_evn(i);
    bdia_corr(k) = bdia_evn(i);
    mdi_corr(k) = mdi_evn(i);
    ecd_corr(k) = ecd_evn(i);

    k = k + 1;

end

len_corr = length(rpm_corr);

return
```

B.3.3 Function for selection of data

```
function [len_cl rpm_cl wob_cl rop_cl dmea_cl dver_cl bdia_cl mdi_cl
ecd_cl] = funk_data_clean(low_wob, high_wob, low_rpm, high_rpm, low_rop,
            high_rop, low_dver, high_dver, len_tot, rpm_tot, wob_tot, ...
            rop_tot, dmea_tot, dver_tot, bdia_tot, mdi_tot, ecd_tot)


% Removing data outside of WOB boundaries

j = 0;

for i = ( 1 : (len_tot) )

    if ( wob_tot(i) > low_wob && wob_tot(i) < high_wob )
        j = j + 1;
    end
end

len_tmp = j;
j = 0;

rpm_tmp  = zeros(1,len_tmp);
wob_tmp  = zeros(1,len_tmp);
rop_tmp  = zeros(1,len_tmp);
dmea_tmp = zeros(1,len_tmp);
dver_tmp = zeros(1,len_tmp);
bdia_tmp = zeros(1,len_tmp);
mdi_tmp  = zeros(1,len_tmp);
ecd_tmp  = zeros(1,len_tmp);


for i = ( 1 : (len_tot) )

    if ( wob_tot(i) > low_wob && wob_tot(i) < high_wob )

        j = j + 1;

        rpm_tmp(j)  = rpm_tot(i);
        wob_tmp(j)  = wob_tot(i);
        rop_tmp(j)  = rop_tot(i);
        dmea_tmp(j) = dmea_tot(i);
        dver_tmp(j) = dver_tot(i);
        bdia_tmp(j) = bdia_tot(i);
        mdi_tmp(j)  = mdi_tot(i);
        ecd_tmp(j)  = ecd_tot(i);

    end
end
```

```
% Removing data outside of RPM boundaries
j = 0;

for i = ( 1 : (len_tmp) )

    if ( rpm_tmp(i) > low_rpm && rpm_tmp(i) < high_rpm )
        j = j + 1;
    end
end

len_tmp2 = j;
j = 0;

rpm_tmp2  = zeros(1,len_tmp2);
wob_tmp2  = zeros(1,len_tmp2);
rop_tmp2  = zeros(1,len_tmp2);
dmea_tmp2 = zeros(1,len_tmp2);
dver_tmp2 = zeros(1,len_tmp2);
bdia_tmp2 = zeros(1,len_tmp2);
mdi_tmp2  = zeros(1,len_tmp2);
ecd_tmp2  = zeros(1,len_tmp2);


for i = ( 1 : (len_tmp) )

    if ( rpm_tmp(i) > low_rpm && rpm_tmp(i) < high_rpm )

        j = j + 1;

        rpm_tmp2(j)  = rpm_tmp(i);
        wob_tmp2(j)  = wob_tmp(i);
        rop_tmp2(j)  = rop_tmp(i);
        dmea_tmp2(j) = dmea_tmp(i);
        dver_tmp2(j) = dver_tmp(i);
        bdia_tmp2(j) = bdia_tmp(i);
        mdi_tmp2(j)  = mdi_tmp(i);
        ecd_tmp2(j)  = ecd_tmp(i);

    end
end
```

```
% Removing data outside of ROP boundaries
j = 0;

for i = ( 1 : (len_tmp2) )

    if ( rop_tmp2(i) > low_rop && rop_tmp2(i) < high_rop )
        j = j + 1;
    end
end

len_tmp3 = j;
j = 0;

rpm_tmp3  = zeros(1,len_tmp3);
wob_tmp3  = zeros(1,len_tmp3);
rop_tmp3  = zeros(1,len_tmp3);
dmea_tmp3 = zeros(1,len_tmp3);
dver_tmp3 = zeros(1,len_tmp3);
bdia_tmp3 = zeros(1,len_tmp3);
mdi_tmp3  = zeros(1,len_tmp3);
ecd_tmp3  = zeros(1,len_tmp3);


for i = ( 1 : (len_tmp2) )

    if ( rop_tmp2(i) > low_rop && rop_tmp2(i) < high_rop )

        j = j + 1;

        rpm_tmp3(j)  = rpm_tmp2(i);
        wob_tmp3(j)  = wob_tmp2(i);
        rop_tmp3(j)  = rop_tmp2(i);
        dmea_tmp3(j) = dmea_tmp2(i);
        dver_tmp3(j) = dver_tmp2(i);
        bdia_tmp3(j) = bdia_tmp2(i);
        mdi_tmp3(j)  = mdi_tmp2(i);
        ecd_tmp3(j)  = ecd_tmp2(i);

    end
end
```

```
% Removing data outside of DVER boundaries
j = 0;

for i = ( 1 : (len_tmp3) )

    if ( dver_tmp3(i) > low_dver && dver_tmp3(i) < high_dver )
        j = j + 1;
    end
end

len_cl = j;
j = 0;

rpm_cl  = zeros(1,len_cl);
wob_cl  = zeros(1,len_cl);
rop_cl  = zeros(1,len_cl);
dmea_cl = zeros(1,len_cl);
dver_cl = zeros(1,len_cl);
bdia_cl = zeros(1,len_cl);
mdi_cl  = zeros(1,len_cl);
ecd_cl  = zeros(1,len_cl);

for i = ( 1 : (len_tmp3) )

    if ( dver_tmp3(i) > low_dver && dver_tmp3(i) < high_dver )

        j = j + 1;

        rpm_cl(j)  = rpm_tmp3(i);
        wob_cl(j)  = wob_tmp3(i);
        rop_cl(j)  = rop_tmp3(i);
        dmea_cl(j) = dmea_tmp3(i);
        dver_cl(j) = dver_tmp3(i);
        bdia_cl(j) = bdia_tmp3(i);
        mdi_cl(j)  = mdi_tmp3(i);
        ecd_cl(j)  = ecd_tmp3(i);

    end
end




return
```

B.3.4 Function to average multiple data points

```
function [len_evn rpm_evn wob_evn rop_evn dmea_evn dver_evn bdia_evn
mdi_evn ecd_evn pporefwr_evn ovbfwr_evn normfwr_evn] = funk_even3(len_cl,
rpm_cl, wob_cl, rop_cl, dmea_cl, dver_cl, bdia_cl, ...
mdi_cl, ecd_cl, pporefwr_cl, ovbfwr_cl, normfwr_cl)


 k = 1 ;

for i = 1 : 3 : (len_cl - 2)

   rpm_evn(k) = ( ( rpm_cl(i) + rpm_cl(i+1) + rpm_cl(i+2) ) / 3 );

   wob_evn(k) = ( ( wob_cl(i) + wob_cl(i+1) + wob_cl(i+2) ) / 3 );

   rop_evn(k) = ( ( rop_cl(i) + rop_cl(i+1) + rop_cl(i+2) ) / 3 );

   dmea_evn(k) = ( ( dmea_cl(i) + dmea_cl(i+1) + dmea_cl(i+2) ) / 3 );

   dver_evn(k) = ( ( dver_cl(i) + dver_cl(i+1) + dver_cl(i+2) ) / 3 );

   bdia_evn(k) = ( ( bdia_cl(i) + bdia_cl(i+1) + bdia_cl(i+2) ) / 3 );

   mdi_evn(k) = ( ( mdi_cl(i) + mdi_cl(i+1) + mdi_cl(i+2) ) / 3 );

   ecd_evn(k) = ( ( ecd_cl(i) + ecd_cl(i+1) + ecd_cl(i+2) ) / 3 );

   pporefwr_evn(k) = ( ( pporefwr_cl(i) + pporefwr_cl(i+1) +
pporefwr_cl(i+2) ) / 3 );

   ovbfwr_evn(k) = ( ( ovbfwr_cl(i) + ovbfwr_cl(i+1) + ovbfwr_cl(i+2) ) / 3
);

   normfwr_evn(k) = ( ( normfwr_cl(i) + normfwr_cl(i+1) + normfwr_cl(i+2) )
/ 3 );

   k = k + 1 ;

end

   len_evn = length(rpm_evn);


return
```

B.3.5 Function to create depth interval between data points

```
function [len_dpt rpm_dpt wob_dpt rop_dpt dmea_dpt dver_dpt bdia_dpt
mdi_dpt ecd_dpt pporefwr_dpt ovbfwr_dpt normfwr_dpt normfit_dpt] =
funk_data_depth(d_int, len_cl, ...
dmea_cl, dver_cl, bdia_cl, ...
mdi_cl, ecd_cl, pporefwr_cl, ...
ovbfwr_cl, normfwr_cl, normfit_cl)


% Input desired length interval beetween data points:

%d_int = 0.01;


% Creating array with a minimum depth between each data point

k = 1;
low = 0;

for i = 1 : len_cl

    if ( dver_cl(i) >= (low + d_int)  )

        for j = i : len_cl

            diff = dver_cl(j) - dver_cl(i);

            if ( diff > d_int )

                rpm_dpt(k)  = rpm_cl(i);
                rpm_dpt(k+1)= rpm_cl(j);

                wob_dpt(k)  = wob_cl(i);
                wob_dpt(k+1)= wob_cl(j);

                rop_dpt(k)  = rop_cl(i);
                rop_dpt(k+1)= rop_cl(j);

                dmea_dpt(k)  = dmea_cl(i);
                dmea_dpt(k+1)= dmea_cl(j);

                dver_dpt(k)  = dver_cl(i);
                dver_dpt(k+1)= dver_cl(j);

                bdia_dpt(k)  = bdia_cl(i);
                bdia_dpt(k+1)= bdia_cl(j);

                mdi_dpt(k)  = mdi_cl(i);
                mdi_dpt(k+1)= mdi_cl(j);


                ecd_dpt(k)  = ecd_cl(i);
                ecd_dpt(k+1)= ecd_cl(j);

                pporefwr_dpt(k)  = pporefwr_cl(i);
                pporefwr_dpt(k+1)= pporefwr_cl(j);
                ovbfwr_dpt(k)  = ovbfwr_cl(i);
                ovbfwr_dpt(k+1)= ovbfwr_cl(j);

                normfwr_dpt(k)  = normfwr_cl(i);
                normfwr_dpt(k+1)= normfwr_cl(j);

                normfit_dpt(k)  = normfit_cl(i);
                normfit_dpt(k+1)= normfit_cl(j);

                low = dver_cl(j);
                k = k + 2;

                break

            end
        end
    end
end

len_dpt = length(rpm_dpt);


return
```

B.4 Calculation functions

B.4.1 Function to create d_c-trend line coefficients

```
function [len_trend coeff] = funk_calc_dctrend(d_trend, len_dpt, rop_dpt,
rpm_dpt, wob_dpt, bdia_dpt, mdi_dpt, dver_dpt, normfwr_dpt)


% Input max depth for trend line cration:

%d_trend = 1000 ;


% Determines for wich max depth data will be used to create trend line

k = 0;

for i = ( 1 : len_dpt )
    if ( dver_dpt(i) < d_trend )

        k = k + 1;

    end
end

len_trend = k;

dc  = zeros(1, len_trend);
dcc  = zeros(1, len_trend);
dver_trend  = zeros(1, len_trend);

% Calculates dc-exponent for the selected intervall

for i = ( 1 : len_trend )

            dc(i) = (log( (rop_dpt(i)/0.3048) / (60 * rpm_dpt(i)) )) /
(log( (12 * (wob_dpt(i)/0.45359)) / (1000 * bdia_dpt(i)) ));

            dcc(i) = dc(i) * (normfwr_dpt(i) / mdi_dpt(i) );

            dver_trend(i) = dver_dpt(i);

end

%Utilize built in matlab function to make a linnear regression curve for
%the selected data

coeff = polyfit(dver_trend(1:len_trend), dcc(1:len_trend), 1);



return
```

B.4.2 Function to calculate d_c-exponent

```
function [dcc dc] = funk_calc_dcc(len_cl, rop_cl, rpm_cl, wob_cl, bdia_cl,
mdi_cl, normfwr_cl)

%Calculate the dc-exponent

dc   = zeros(1,len_cl);
dcc   = zeros(1,len_cl);

for i = (1 : len_cl);

    dc(i) = (log( (rop_cl(i)/0.3048) / (60 * rpm_cl(i)) )) / (log(
(12*(wob_cl(i)/0.45359)) / (1000 * bdia_cl(i)) ));

    dcc(i) = dc(i) * (normfwr_cl(i) / mdi_cl(i) );

end;



return
```

B.4.3 Function to calculate pore pressure gradient from Eaton's method

```
function [p_eaton] = funk_calc_eaton (dver_dpt, ovbfwr_dpt, normfwr_dpt,
dcc, coeff, len_dpt)



p_eaton = zeros(1,len_dpt);

trend = ( dver_dpt * coeff(1) ) + coeff(2) ;

for i = 1 : len_dpt

    p_eaton(i) = ovbfwr_dpt(i) - ( (ovbfwr_dpt(i) - normfwr_dpt(i)) * (
(dcc(i)/trend(i))^1.6 ) ) ;

end

return
```

B.4.4 Function to calculate pore pressure gradient from Zamora's method

```
function [p_zamora] = funk_calc_zamora (dcc, dver_cl, normfwr_cl, len_cl,
coeff)

p_zamora  = zeros(1,len_cl);

trend = ( dver_cl * coeff(1) ) + coeff(2) ;

for i = (1 : len_cl);

   p_zamora(i) = normfwr_cl(i) * ( trend(i) / dcc(i) );

end;



return
```

B.4.5 Function to calculate factors of Bourgoyne-Young equation

```
function [f1 f2 f3 f4 f5 f6] = funk_calc_BYfactor (pporefwr_dpt, dver_dpt,
rop_dpt, wob_dpt, bdia_dpt, rpm_dpt, ecd_dpt, a2, a3, a4, a5, a6, len_dpt)

%Factors of B&Y equation, calculated separately for code readability and
%possible separate aveluation.

% f1 calculated separately
% f7 f8 and threshold bit weight neglected

rop = rop_dpt / 0.3048;
dver = dver_dpt / 0.3048;
wob = wob_dpt / 0.45359;
ppore = pporefwr_dpt * 8.345404 ;
ecd = ecd_dpt * 8.345404 ;

f1  = zeros(1, len_dpt);
f2  = zeros(1, len_dpt);
f3  = zeros(1, len_dpt);
f4  = zeros(1, len_dpt);
f5  = zeros(1, len_dpt);
f6  = zeros(1, len_dpt);


for i = 1 : len_dpt

    f2(i) = exp( 2.303 * a2 * (10000 - (dver(i)))) ;

    f3(i) = exp( 2.303 * a3 * (dver(i).^0.69) * (ppore(i) - 9) );

    f4(i) = exp( 2.303 * a4 * dver(i) * (ppore(i) - ecd(i)) );

    f5(i) =  ( ( (wob(i)) / bdia_dpt(i) ) / 4 ) ^ a5 ;

    f6(i) = ( rpm_dpt(i) / 60 ) ^ a6 ;

end



for i = ( 1 : len_dpt )

    f1(i) = ( (rop(i)) / ( f2(i) * f3(i)* f4(i) * f5(i) * f6(i) ) );

end


return
```

B.4.6 Function to calculate linear drillability trend for a desired interval

```
function [coeff_K drill_K dver_trend_K] = funk_calc_avK (dver_dpt, rop_dpt,
f1, len_dpt, d_trend_Kmin, d_trend_Kmax)

% Determines number of data points to be included in trend
k = 0;

for i = ( 1 : len_dpt )
    if ( dver_dpt(i) > d_trend_Kmin  &&  dver_dpt(i) < d_trend_Kmax )

        k = k + 1;

    end
end

len_trend = k;


rop = rop_dpt / 0.3048;

drill_K  = zeros(1, len_trend);
dver_trend_K  = zeros(1, len_trend);



% Calculates assumed normal drillability development for the selected
% intervall, from f1

% f3 f4 f7 f8 neglected

k = 0;

for i = ( 1 : len_dpt )

    if ( dver_dpt(i) > d_trend_Kmin  &&  dver_dpt(i) < d_trend_Kmax )

            k = k + 1;

            drill_K(k) = f1(i);

            dver_trend_K(k) = dver_dpt(i);
    end
end



%Utilize built in matlab function to make a linnear regression curve for
%the selected data

coeff_K = polyfit(dver_trend_K(1:len_trend), drill_K(1:len_trend), 1);



return
```

B.4.7 Function to calculate linear drillability trend for each separate lithology

```
function [rop_lin f1_lin f5_lin f6_lin] = test_lin_zones (dver_dpt,
rop_dpt, f1, f5, f6, len_dpt )


% Creates separat arrays of data (dver, rop, f1, f5, f6) for each formation
% encountered in the well

for i = 1 : len_dpt

    if dver_dpt(i) < 966

        dver_nord(i) = dver_dpt(i);
        rop_nord(i) = rop_dpt(i);
        f1_nord(i) = f1(i);
        f5_nord(i) = f5(i);
        f6_nord(i) = f6(i);

        len_nord = i;

    elseif dver_dpt(i) > 966 &&  dver_dpt(i) < 1011

        dver_utsira(i) = dver_dpt(i);
        rop_utsira(i) = rop_dpt(i);
        f1_utsira(i) = f1(i);
        f5_utsira(i) = f5(i);
        f6_utsira(i) = f6(i);

        len_nord = i;

    elseif dver_dpt(i) > 1011 &&  dver_dpt(i) < 1366

        dver_shord(i) = dver_dpt(i);
        rop_shord(i) = rop_dpt(i);
        f1_shord(i) = f1(i);
        f5_shord(i) = f5(i);
        f6_shord(i) = f6(i);

        len_nord = i;

    elseif dver_dpt(i) > 1366 &&  dver_dpt(i) < 1570

        dver_hord(i) = dver_dpt(i);
        rop_hord(i) = rop_dpt(i);
        f1_hord(i) = f1(i);
        f5_hord(i) = f5(i);
        f6_hord(i) = f6(i);

        len_nord = i;

    elseif dver_dpt(i) > 1570 &&  dver_dpt(i) < 1639

        dver_bald(i) = dver_dpt(i);
        rop_bald(i) = rop_dpt(i);
        f1_bald(i) = f1(i);
        f5_bald(i) = f5(i);
        f6_bald(i) = f6(i);
        len_nord = i;

    elseif dver_dpt(i) > 1639 &&  dver_dpt(i) < 1739

        dver_list(i) = dver_dpt(i);
        rop_list(i) = rop_dpt(i);
        f1_list(i) = f1(i);
        f5_list(i) = f5(i);
        f6_list(i) = f6(i);

        len_nord = i;

    elseif dver_dpt(i) > 1739 &&  dver_dpt(i) < 2000

        dver_shet(i) = dver_dpt(i);
        rop_shet(i) = rop_dpt(i);
        f1_shet(i) = f1(i);
        f5_shet(i) = f5(i);
        f6_shet(i) = f6(i);

        len_nord = i;

    end

end

% Create different linear approximations for each of the different factors:

c_rop_nord = polyfit(dver_nord, rop_nord, 1);
c_f1_nord = polyfit(dver_nord, f1_nord, 1);
c_f5_nord = polyfit(dver_nord, f5_nord, 1);
c_f6_nord = polyfit(dver_nord, f6_nord, 1);

c_rop_utsira = polyfit(dver_utsira, rop_utsira, 1);
c_f1_utsira = polyfit(dver_utsira, f1_utsira, 1);
c_f5_utsira = polyfit(dver_utsira, f5_utsira, 1);
c_f6_utsira = polyfit(dver_utsira, f6_utsira, 1);

c_rop_shord = polyfit(dver_shord, rop_shord, 1);
c_f1_shord = polyfit(dver_shord, f1_shord, 1);
c_f5_shord = polyfit(dver_shord, f5_shord, 1);
c_f6_shord = polyfit(dver_shord, f6_shord, 1);

c_rop_hord = polyfit(dver_hord, rop_hord, 1);
c_f1_hord = polyfit(dver_hord, f1_hord, 1);
c_f5_hord = polyfit(dver_hord, f5_hord, 1);
c_f6_hord = polyfit(dver_hord, f6_hord, 1);

c_rop_bald = polyfit(dver_bald, rop_bald, 1);
c_f1_bald = polyfit(dver_bald, f1_bald, 1);
c_f5_bald = polyfit(dver_bald, f5_bald, 1);
c_f6_bald = polyfit(dver_bald, f6_bald, 1);

c_rop_list = polyfit(dver_list, rop_list, 1);
c_f1_list = polyfit(dver_list, f1_list, 1);
c_f5_list = polyfit(dver_list, f5_list, 1);
c_f6_list = polyfit(dver_list, f6_list, 1);
```

```
c_rop_shet = polyfit(dver_shet, rop_shet, 1);
c_f1_shet = polyfit(dver_shet, f1_shet, 1);
c_f5_shet = polyfit(dver_shet, f5_shet, 1);
c_f6_shet = polyfit(dver_shet, f6_shet, 1);



%Combining linearized data into arrays covering the total length of the
%well
for i = 1 : len_dpt

    if dver_dpt(i) < 966

        rop_lin(i) = (dver_dpt(i) * c_rop_nord(1) ) + c_rop_nord(2) ;
        f1_lin(i) = (dver_dpt(i) * c_f1_nord(1) ) + c_f1_nord(2) ;
        f5_lin(i) = (dver_dpt(i) * c_f5_nord(1) ) + c_f5_nord(2) ;
        f6_lin(i) = (dver_dpt(i) * c_f6_nord(1) ) + c_f6_nord(2) ;



    elseif dver_dpt(i) > 966 &&  dver_dpt(i) < 1011

        rop_lin(i) = (dver_dpt(i) * c_rop_utsira(1) ) + c_rop_utsira(2) ;
        f1_lin(i) = (dver_dpt(i) * c_f1_utsira(1) ) + c_f1_utsira(2) ;
        f5_lin(i) = (dver_dpt(i) * c_f5_utsira(1) ) + c_f5_utsira(2) ;
        f6_lin(i) = (dver_dpt(i) * c_f6_utsira(1) ) + c_f6_utsira(2) ;



    elseif dver_dpt(i) > 1011 &&  dver_dpt(i) < 1366

        rop_lin(i) = (dver_dpt(i) * c_rop_shord(1) ) + c_rop_shord(2) ;
        f1_lin(i) = (dver_dpt(i) * c_f1_shord(1) ) + c_f1_shord(2) ;
        f5_lin(i) = (dver_dpt(i) * c_f5_shord(1) ) + c_f5_shord(2) ;
        f6_lin(i) = (dver_dpt(i) * c_f6_shord(1) ) + c_f6_shord(2) ;



    elseif dver_dpt(i) > 1366 &&  dver_dpt(i) < 1570

        rop_lin(i) = (dver_dpt(i) * c_rop_hord(1) ) + c_rop_hord(2) ;
        f1_lin(i) = (dver_dpt(i) * c_f1_hord(1) ) + c_f1_hord(2) ;
        f5_lin(i) = (dver_dpt(i) * c_f5_hord(1) ) + c_f5_hord(2) ;
        f6_lin(i) = (dver_dpt(i) * c_f6_hord(1) ) + c_f6_hord(2) ;



    elseif dver_dpt(i) > 1570 &&  dver_dpt(i) < 1639

        rop_lin(i) = (dver_dpt(i) * c_rop_bald(1) ) + c_rop_bald(2) ;
        f1_lin(i) = (dver_dpt(i) * c_f1_bald(1) ) + c_f1_bald(2) ;
        f5_lin(i) = (dver_dpt(i) * c_f5_bald(1) ) + c_f5_bald(2) ;
        f6_lin(i) = (dver_dpt(i) * c_f6_bald(1) ) + c_f6_bald(2) ;
    elseif dver_dpt(i) > 1639 &&  dver_dpt(i) < 1739

        rop_lin(i) = (dver_dpt(i) * c_rop_list(1) ) + c_rop_list(2) ;
        f1_lin(i) = (dver_dpt(i) * c_f1_list(1) ) + c_f1_list(2) ;
        f5_lin(i) = (dver_dpt(i) * c_f5_list(1) ) + c_f5_list(2) ;
        f6_lin(i) = (dver_dpt(i) * c_f6_list(1) ) + c_f6_list(2) ;



    elseif dver_dpt(i) > 1739 &&  dver_dpt(i) < 2000

        rop_lin(i) = (dver_dpt(i) * c_rop_shet(1) ) + c_rop_shet(2) ;
        f1_lin(i) = (dver_dpt(i) * c_f1_shet(1) ) + c_f1_shet(2) ;
        f5_lin(i) = (dver_dpt(i) * c_f5_shet(1) ) + c_f5_shet(2) ;
        f6_lin(i) = (dver_dpt(i) * c_f6_shet(1) ) + c_f6_shet(2) ;



    end

end


return
```

B.4.8 Function to calculate pressure gradients from Bourgoyne-Young equation

```
function [p_by k] = funk_calc_BYf3 (coeff_K, dver_dpt, rop_dpt, f2, f5, f6,
mdi_dpt, a3, a4, len_dpt)

% Calculating pressure gradient from B&Y equation:

    %Neglected :
            %f7 bit wear = 1
            %f8 jet force = 1


p_by_imp  = zeros(1,len_dpt);

by_fac1  = zeros(1,len_dpt);
by_fac2  = zeros(1,len_dpt);
by_fac3  = zeros(1,len_dpt);

k = ( dver_dpt * coeff_K(1) ) + coeff_K(2) ;

rop = rop_dpt / 0.3048;
dver = dver_dpt / 0.3048;
mdi = mdi_dpt * 8.345404;



%Pressure gradient:
for i = ( 1 : len_dpt )

    by_fac1(i) = ( log( rop(i) / (k(i) * f2(i) * f5(i) * f6(i)) ) / 2.303
);

    by_fac2(i) = ( (9 * (dver(i)^0.69) * a3) + (mdi(i) * dver(i) * a4) );

    by_fac3(i) = ( ((dver(i)^0.69) * a3) + (dver(i) * a4) );

    p_by_imp(i) = ( (by_fac1(i) + by_fac2(i)) / by_fac3(i) );

end


p_by = (p_by_imp / 8.345404);

return
```

B.4.9 Function to calculate standard deviation

```
function [std_eaton std_zamora std_by] = funk_calc_std (dver_dpt,
pporefwr_dpt, p_eaton, p_zamora, p_by, len_dpt)


diff_eaton = zeros(1, len_dpt);
diff_zamora = zeros(1, len_dpt);
diff_by = zeros(1, len_dpt);

k = 0;

for i = 1 : len_dpt

    if dver_dpt(i) > 600 &&  dver_dpt(i) < 966

        k = k + 1;

        diff_eaton(k) = ((p_eaton(i) - pporefwr_dpt(i)).^2);

        diff_zamora(k) = ((p_zamora(i) - pporefwr_dpt(i)).^2);

        diff_by(k) = ((p_by(i) - pporefwr_dpt(i)).^2);

    end

    if dver_dpt(i) > 1366 &&  dver_dpt(i) < 1739

        k = k + 1;

        diff_eaton(k) = ((p_eaton(i) - pporefwr_dpt(i)).^2);

        diff_zamora(k) = ((p_zamora(i) - pporefwr_dpt(i)).^2);

        diff_by(k) = ((p_by(i) - pporefwr_dpt(i)).^2);

    end



end

len_std = length(diff_by);


std_eaton = sqrt( sum(diff_eaton) / len_std );

std_zamora = sqrt( sum(diff_zamora) / len_std );

std_by = sqrt( sum(diff_by) / len_std );


return
```

B.5 Output functions

B.5.1 Various plot functions

```
function funk_plot_dc (dcc, dver_cl, coeff)

trend = ( dver_cl * coeff(1) ) + coeff(2) ;

figure(1)
hold on
plot(dcc, dver_cl),
hold on
plot(trend, dver_cl,'r')
    xlabel('d_c-exponent', 'fontsize', 12),
    ylabel('Vertical depth [m]', 'fontsize', 12),
    grid on
    set(gca,'YDir','reverse'),
    legend('d_c-exponent','Normal trendline'),
    title('d_c-exponent', 'fontsize', 12),
    axis([0.3 1.7 600 2000])
return



function funk_plot_zamora (p_zamora, dver_dpt, pporefwr_dpt)

figure(2)
plot(pporefwr_dpt, dver_dpt, 'r')
hold on
plot(p_zamora, dver_dpt)
    xlabel('SG', 'fontsize', 12),
    ylabel('Vertical depth [m]', 'fontsize', 12),
    grid on
    set(gca,'YDir','reverse'),
    legend('FWR p_p_o_r_e','Zamora p_p_o_r_e'),
    title('Zamora pressure estimation', 'fontsize', 12),
    axis([0.6 1.8 600 2000])


return



function funk_plot_eaton (dver_dpt, pporefwr_dpt, p_eaton)

figure(3)
plot(pporefwr_dpt, dver_dpt, 'r')
hold on
plot(p_eaton, dver_dpt)
    xlabel('SG', 'fontsize', 12),
    ylabel('Vertical depth [m]', 'fontsize', 12),
    grid on
    set(gca,'YDir','reverse'),
    legend('FWR p_p_o_r_e','Eaton p_p_o_r_e'),
    title('Eaton pressure estimation', 'fontsize', 12),
    axis([0.6 1.8 600 2000])


return
```

B.5.2 Function to plot standard deviation

```
function funk_plot_std (dver_dpt, std_eaton, std_zamora, std_by,
pporefwr_dpt, p_eaton, p_zamora, p_by, len_dpt)

for i = 1 : len_dpt

    eaton_mstd(i) = pporefwr_dpt(i) - std_eaton ;
    eaton_pstd(i) = pporefwr_dpt(i) + std_eaton ;

    zamora_mstd(i) = pporefwr_dpt(i) - std_zamora ;
    zamora_pstd(i) = pporefwr_dpt(i) + std_zamora ;

    by_mstd(i) = pporefwr_dpt(i) - std_by ;
    by_pstd(i) = pporefwr_dpt(i) + std_by ;



end


figure(51)
plot(zamora_pstd, dver_dpt, 'g')
hold on
plot(pporefwr_dpt, dver_dpt, 'r')
hold on
plot(p_zamora, dver_dpt)
hold on
plot(zamora_mstd, dver_dpt, 'g')
    xlabel('SG', 'fontsize', 12),
    ylabel('Vertical depth [m]', 'fontsize', 12),
    grid on
    set(gca,'YDir','reverse'),
    legend('+/- STD','FWR p_p_o_r_e','Zamora p_p_o_r_e'),
    title('Standard deviation - Zamora', 'fontsize', 12),
    axis([0.6 1.8 600 2000])

figure(52)
plot(eaton_pstd, dver_dpt, 'g')
hold on
plot(pporefwr_dpt, dver_dpt, 'r')
hold on
plot(p_eaton, dver_dpt)
hold on
plot(eaton_mstd, dver_dpt, 'g')
    xlabel('SG', 'fontsize', 12),
    ylabel('Vertical depth [m]', 'fontsize', 12),
    grid on
    set(gca,'YDir','reverse'),
    legend('+/- STD','FWR p_p_o_r_e','Eaton p_p_o_r_e'),
    title('Standard deviation - Eaton', 'fontsize', 12),
    axis([0.6 1.8 600 2000])
```

```
figure(53)
plot(by_pstd, dver_dpt, 'g')
hold on
plot(pporefwr_dpt, dver_dpt, 'r')
hold on
plot(p_by, dver_dpt)
hold on
plot(by_mstd, dver_dpt, 'g')
    xlabel('SG', 'fontsize', 12),
    ylabel('Vertical depth [m]', 'fontsize', 12),
    grid on
    set(gca,'YDir','reverse'),
    legend('+/- STD','FWR p_p_o_r_e','B-Y p_p_o_r_e'),
    title('Standard deviation - Bourgoyne-Young', 'fontsize', 12),
    axis([0.4 2 600 2000])



end
```