MATLAB GRAPHICAL USER INTERFACE (GUI) DESIGN FOR PROCESSING AND INTERPRETATION OF WIRELINE LOG DATA EXPLAINATION AND SHOWCASE USING BALKASSAR AREA, UPPER INDUS BASIN OF PAKISTAN

By
M. Omar Bakht

B.S (GEOPHYSICS)

2016-2020

DEPARTMENT OF EARTH SCIENCES
QUAID-I- AZAM UNIVERSITY

CERTIFICATE OF APPROVAL

This dissertation submitted by **MUHAMMAD OMAR BAKHT** S/O **NAVEED AHMAD SHAMI** is accepted in its present form by the Department of Earth Sciences, Quaid-I-Azam University Islamabad as satisfying the requirement for the award of degree of **BS Geophysics.**

RECOMMENDED BY

Dr.Faisal Rahman _____
(Dissertation Supervisor)

Dr. Aamir Ali _____
(Chairperson Department of Earth Scirnces)

External Examiner _____

ACKNOWLOEDMENT

In the name of **Allah**, the most Beneficent, the most Merciful. All praises to **Almighty Allah**, the creator of universe. Secondly, my humblest gratitude to the **Holy Prophet Muhammad (Peace Be Upon Him)**. Without the blessing of Allah, I could not be able to complete my work as well as to be at such a place. This thesis appears in its current form due to the assistance and guidance of several people. It gives me great pleasure to express my gratitude to all those who supported me and have contributed in making this thesis possible.

I express my profound sense of reverence to **Dr. Aamir Ali**, who gave me the opportunity to work under his guiding hand on such an outstanding research. His continuous support, motivation and untiring guidance have made this thesis possible. His vast knowledge, calm nature and positive criticism motivated me to push harder to get the best forms of results. I thank him for bearing my mistakes and being the "teacher" that many fails to understand the true concept of this elevated post bestowed upon them.

Also, I am immensely pleased to place on record my deep gratitude and heartfelt thanks to **Umer Farooq, Shahzaib Sheikh** and **Yawar Amin** who helped me throughout this research period, and my close friends who are more brothers than friends **Wajahat Maqsood and Usama shah**.

Last but not the least, I would like to acknowledge my family for their constant support, unceasing prayers and best wishes. My parents specially to whom I thank the most who helped me most with their constant support throughout the time for the successful completion of my thesis. Those missed to be named are always close to my heart, and if not named in the acknowledgment are fully thanked.

MUHAMMAD OMAR BAKHT
BS GEOPHYSICS
2016-2020

Table of Contents

ABSTRACT

Applications and programs for the use of processing and interpreting well log data are costly as well as rarely easy to use. This fact makes it difficult for students of the field to acquire and utilize the said applications and programs. Not only is the cost of the tools a hurdle but even if overcome the second hurdle of learning these complex tools is a larger hurdle right after.

This study focuses on the showcase of an application developed by myself, utilizing matlab programming that is able to perform basic processes of well log processing and hence can be used as a tool for basic interpretation of well logs. It should however be duly noted that the created program is made using self-taught knowledge of matlab programming and hence is highly constrained by my personal knowledge of programming and coding abilities as well as the allotted time for the thesis.

The well used as an example for the showcase of the GUIs capabilities was BALKASSAR-OXY-01 well and for comparison purposes the IHS Kingdom software was used.

Chapter 1:

Introduction

1.1 Importance of well logs

Many different modern geophysical well logs exist. They are records of sophisticated geophysical measurements along a borehole. These may be measurements of spontaneous phenomena, such as natural radioactivity, which requires a tool consisting simply of a very sensitive radiation detector; or they may be induced, as with the formation velocity log(sonic log), in which a tool emits sound into the formation and measures the time taken for the sound to reach a receiver at a set distance along the tool.

Geophysical well logging is necessary because geological sampling during drilling, 'cuttings sampling', leaves a very imprecise record of the formations encountered. Entire formation samples can be brought to the surface by mechanical coring, but this is both slow and expensive. The results of coring are unequivocal. Logging is precise but equivocal, in that it needs interpretation to bring a log to the level of geological or petrophysical experience. However, logs fill the gap between 'cuttings' and 'cores', and with experience, calibration and computers, they can almost replace cores. As they certainly contain enough information to put outcrop reality into the subsurface

As Logging tools and interpretive methods are developing in accuracy and sophistication, they are playing an expanded role in the geological decision-making process. Today petrophysical log interpretation is one of the most useful and important tools available to a petroleum geologist.

Besides their traditional use in exploration to correlate zones and to assist with structure and isopach mapping, logs help define physical rock characteristics such as lithology, porosity, pore geometry and permeability. Logging data is used to identify productive zones, to determine depth and thickness of zones. To distinguish between oil, gas or water in a reservoir. And to estimate hydrocarbon reserves. Also, geologic maps developed from log interpretation help with determining facies relationships and drilling locations.

A geologist's first exposure to log interpretation can be a frustrating experience due to its lengthy and unfamiliar terminology as well as the required knowledge of many parameters, concepts and measurements needed in order to gain an understanding of the logging process. The same can be said for applications developed for use in well log interpretation i.e. they are complex and need the guidance of an experienced user in order to fully grasp leading to an intimidating and frustrating first exposure.

1.2 Aim for this thesis

This thesis aims to create a user friendly Graphical User Interface (GUI) application through the use of matlab programming. It also aims to explain the coding methods used in the development of this application so that this paper can be used as a form of reference for any students of geophysics aspiring to create their own similar GUI application.

The focus of this study has been to create a GUI application that fulfills the two following criteria above all:

1. User friendly easy to understand interface
2. Ability to perform basic well log processes.

The Balkassar area of the upper Indus basin is used as an example to showcase the GUIs abilities and is compared to results using similar method in commercial software, namely IHS Kingdom Software.

The processes that the program can competently perform are namely:

- Display of individual well logs.
- Display of all well logs side by side.
- Calculation and display of Density porosity, using user input values of Matrix density and Matrix Fluid Density.
- Calculation and display of Sonic Porosity, using user input values of Matrix transit time, Fluid Transit time and Compaction factor.
- Calculation and display of volume of shale, using Larionov (1969) formulas for Tertiary and older rocks.
- Calculation and display of Saturation of water (Sw), using user input values of Matrix Density, Matrix Fluid Density, Cementation Factor, Saturation exponent, Rw and user selection of Larionov equation for Tertiary or Older rocks.
- Calculation and display of Vp and Acoustic Impedance.
- Calculation and display of Vs calculated using Castagna (93), Castagna-mudrock line (85) or Han-Empirical Relation (86) Equations utilizing Vp.
- Calculation and display of Young's Modulus, Poisson's Ratio and Brittleness index.

- Ability to display scatter type Cross-plot of any two selected logs.

A workflow of the general wireline logs interpretation is given below in figure 1.1.

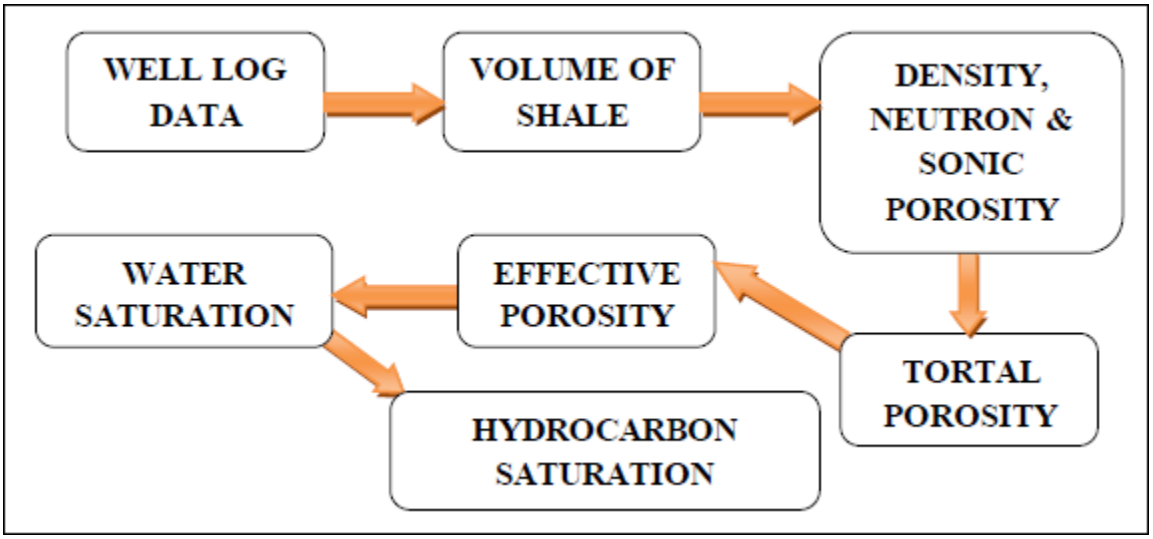


Figure (1.1) workflow of general wireline logs interpretation

1.3 Example Well Data.

The well data includes the following files:

- Balkassar OXY-01.las

These files store all the information about the logs run in the well and well tops. The technical information of the well data along with the information of the formation top name is present in following table.

Technical Well Data			
Operator	OXY	Province	Punjab
Type	Exploratory	Status	Abandoned
Well Bore Name	BALKASSAR-OXY-01	Concession	
Longitude	72° 39' 52.50"	Latitude	32° 56' 38.80"
Spud Date	20-June-1981	Completion Date	26-Sep-1981
Depth Reference Elevation (m)	535.53	Total Depth (m)	3130.60
Depth Reference	KB	Formation Top	Salt Range
Logging Start Depth KB (m)	360.8103	Logging End Depth KB (m)	3132.8103

List of Well Tops

Formation	Formation Age	Top (m)	Thickness (m)
NAGRI	PLIOCENE	0.00	478.82
CHINJI	MIOCENE	478.82	929.29
KAMLIAL	MIOCENE	1408.11	106.68
MURREE	MIOCENE	1514.78	906.74
CHORGALI	LOWER EOCENE		45.72
(BHADRAR)		2421.52	
SAKESAR	EOCENE	2467.24	135.63
PATALA	PALEOCENE	2602.87	21.34
LOCKHART	PALEOCENE	2624.20	35.05
HANDU	PALEOCENE	2569.25	27.43
SARDHAI	EARLY PERMIAN	2686.68	109.72
WARCHA	EARLY PERMIAN	2796.40	141.73
DANDOT	EARLY PERMIAN	2938.13	60.96
TOBRA	EARLY PERMIAN	2999.09	51.81
KHEWRA SANDSTONE	EARLY CAMBRIAN	3050.90	78.33
SALT RANGE FORMATION	PRE-CAMBRIAN	3129.229	0.77

Chapter 2:

Petrophysical Analysis

2.1 Introduction

Well logging is a tool to measure the subsurface properties of earth. The physical and the chemical properties of the rock explained existence and behavior of the rocks, fluids and soils (Rider, 1996). Well logs used by the petrophysicist are caliper, resistivity, gamma ray (GR), sonic (DT), density (RHOB), and neutron (NPHI) logs etc., and all other desired information is obtained from these logs. Significance of each log cannot be ignored as they play vital role in quantifying reservoir parameters such as porosity, permeability, net pay zone, fluid content, and shale volume. Petrophysical interpretation generally has little concern with seismic, while offers detailed information about borehole measurements, ultimately contributing in reservoir characterization (Krygowski et al., 2004).

2.2 Reservoir Petrophysical Properties

Most petrophysicist are employed to compute what are commonly called reservoir petrophysical properties which include the following.

2.2.1 Lithology

When combined with local geology and core study, geoscientists can use log measurements such gamma, neutron, density, photoelectric, resistivity or their combination to determine the lithology down hole.

2.2.2 Porosity (ϕ)

The amount of pore (or fluid occupied) space in the rock. This is typically measured using an instrument that measures the reaction of the rock to bombardment by neutrons or by gamma rays. Sonic wave speed and NMR logs are also used to measure and derive rock porosity.

2.2.3 Water Saturation (Sw)

The fraction of the pore space occupied by water is water saturation. This is typically measured using an instrument that measures the resistivity of the rock. However, in this paper the archie equation is used to calculate Sw which is explained further in chapter 3.

2.2.4 Hydrocarbon saturation (Sh)

The fraction of pore space occupied by the hydrocarbon is known as the hydrocarbon saturation. This is typically measured by subtracting the water saturation from one.

2.3 Classification of Geophysical Well Logs

Well log is a profile showing different properties of formation, which is measured through wells. Every log gives some information about the subsurface. Some logs are correlated with other log to assure our prediction of lithologies. Geophysical well logs can be classified into three categories.

- Lithology logs.
- Resistivity logs.
- Porosity logs.

2.3.1 Lithology logs

Lithology logs are mostly used to identify the boundaries between permeable and impervious layers, extracted information about permeable formations assist in correlation with other wells. Lithology logs are caliper (CALI), spontaneous potential (SP) and gamma ray (GR).

a) Caliper (CALI) log

Caliper log is used to measure the diameter of the borehole. Moreover, it provides detail information about the formation's cavities portraying loose lithology along with presence of dense rocks where caving is absent. In porous layers, formation of mud cake reduces the diameter of borehole and these variations in diameter influence the logs measurements (Bjorlykke et al., 2010).

b) Gamma Ray (GR) Log

Natural radioactivity of a formation is measured by using the Gamma-ray logs, also known as the lithology logs. The radioactive materials have high concentration in shale while shale free sand and carbonates have low gamma-ray reading.

c) Spontaneous Potential (SP) Log

The spontaneous potential log measures the natural or spontaneous potential difference that exists between the borehole and the surface in the absence of any artificially applied current. It is a very simple log that requires only an electrode in the borehole and a reference electrode at the surface. These spontaneous potentials arise from the different access that different formations provide for charge carriers in the borehole and formation fluids, which lead to a spontaneous current flow, and hence to a spontaneous potential difference. The SP log has four main uses:

- The detection of permeable beds.
- Determination of Rw.
- Indication of the shaliness of a formation.

- Correlation

2.3.2Resistivity logs

Resistivity logs provide details about formation thickness, accurate value for the true formation resistivity and used for correlation purposes. Resistivity logs are plotted on the logarithmic scale due to more variation in resistivity (0.2 to 2000 ohm) with depth.

Resistivity well logs are:

- Deep Laterolog(LLD)
- Shallow Laterolog(LLS)

a) Deep Laterolog (LLD)

Deep laterolog also termed as electrode log, mostly incorporate in measuring salt water muds filled boreholes resistivity (Rmf). The surveying current basically controls the effective depth of this log investigation (Krygowski et al., 2004).

b) Shallow Laterolog (LLS)

Shallow laterolog measures resistivity of fluids present in invaded zone (Rt). In water beating zone, the shallow laterolog records a low resistivity because mud filtrate resistivity (Rmf) is approximately equal to mud resistivity (Rm) (Krygowski et al., 2004).

2.3.3 Porosity logs

Porosity logs are used to measure water saturation in a formation, furthermore, it provides reliable information about lithology and porosity along with discrimination of oil and gas bearing zones.

Porosity well logs are:

- Sonic/Acoustic (DT)
- Neutron Porosity (NPHI)
- Density (RHOB)

a) Sonic/Acoustic(DT) log

Sonic logs measure the interval transit time (□t) of the compressional sound wave through the formation. The interval transit time is related to the porosity of the formation. The interval transit time is related to the porosity of the formation. The unit of measure is the microseconds per feet (Krygowski et al., 2004). Porosity of the formation can be calculated by using equation below.

$$\phi_s = \frac{\Delta t_{log} - \Delta t_m}{\Delta t_f - \Delta t_m}$$

where, ϕs represents the calculation that is derived from the sonic log, Δtm is the interval transient time of the matrix, Δtlog is the interval transit time of formation

represents the transient time of the fluid (salt mud=185 and fresh mud=189). The interval transient time of the formation depends upon the matrix material, its shape, and cementation (Wyllie et al., 1956). If fluid (hydrocarbon or water) is present in the formation, transient interval time is increased, and this behavior shows increase in porosity which can be calculated by using sonic log (Krygowski et al., 2004).

b) Neutron Porosity (ϕ_n) log

Neutron porosity log is also termed as porosity log; basically, it is used to measure the hydrogen ion (HI) concentration in the formation (Krygowski et al., 2004). The neutron log gives value of water filled porosity if the shale free formation is filled with water. In gas reservoir, porosity measured by the neutron log is low then formation true porosity as the hydrogen ions concentration are less in gas reservoir than that of oil and water (Krygowski et al.,2004). It is the one limitation of neutron log that is known as the gas effect.

c) Density (RHOB) log

This log is also known as porosity log that is used to measure electron density of the formation, (Krygowski et al., 2004). Formation electron density is related to bulks density of formation. The density logs are used with other logs and separately or different purposes (Tittman and Wahl, 1965).

Density log can be used to find out the correct porosity of the formation (Asquith and Gibson, 2004). The rock type of my research work is shale. By using following equation, density porosity can be calculated as.

$$\phi_d = \frac{\rho_m - \rho_b}{\rho_m - \rho_f}$$

Where, ϕ_d represents porosity derived from the density log, ρ_b represents bulk density of formation derived from the RHOB log, ρ_m represents matrix density, ρ_f represents the density of fluid. The main purpose of present petrophysics is to obtain calculation about porosity, saturation of water and hydrocarbon.

2.3.4Average Porosity (ϕ_t)

The total porosity is calculated by adding all three porosity values. Average porosity can then be calculated in order to get the effect of all the pores. Average porosity is measured by adding neutron porosity and density porosity values. Whereas, the number of interconnected pores give effective porosity. The calculation of average porosity can be done by using following equation

$$\phi_t = \frac{(\phi N + \phi D)}{2},$$

where, ϕt is average porosity, ϕD is density porosity and ϕN is neutron porosity.

2.4 Well Log and Total Organic Carbon (TOC) Content Estimation.

2.4.1 Gamma Ray (GR) Log

The GR log is used to measure the radioactivity of the formation. This log is also known to be the lithology log. This log shows high value where there is shale, because shale consists of more radioactive elements. Therefore, shale free sand has low gamma ray readings (Krygowski et al., 2004)

Organic rich rocks have high values of GR log. The use of the gamma ray log has been used now a days as an empirical relationship which is observed between the radioactive material and the organic matter (Swanson, 1960). There is no other tool present that gives information about the relation between organic matter and radioactive material, so this too may prove to be of significant value (passey et al., 1990).

2.4.2 Density (RHOB) Log

As the solid organic matter are less dense than that of the surrounding rock matrix, the density log estimates organic matter content (Schmoker, 1979). Although density log application is slightly more accurate than that of the gamma-ray log, but its usefulness is decreased in well bore where shale is washed-out (Passey et al., 1990).

2.4.3 Sonic (DT) Log

Average transit time of 180 μs/ft. is suggested for organic matter. The acoustic log response to organic matter is the increasing of the transit time over 140 μs/ft depending upon the distribution of organic matter in the matrix (Aadil et al., 2014). The DT log is used to further generate another track of P-wave velocity (Vp) which is later used to generate the S-wave velocity log, since the given data is usually not provided with additional logs such as DT4P and DT4S.

Calculation of Petrophysical and Petro-Elastic Properties

Shale oil zone quality can be evaluated with the help of mineral constituents, concentration of oil/gas content, level of organic maturity and petrophysical and petro-elastic properties. This information about the shale oil/gas reservoir give insight to evaluate the well location, hydraulic fracturing, and well drilling design. Examples of calculations are the Young's Modulus , Poisson's Ratio and Brittleness Index. These are further discussed in Chapter 3.

Chapter 3:

Methodology

This chapter explains the methodology and code behind the designed application.

3.1 Use of global variables

Global variables are a type of variable used in programming which can, unlike local variables, carry values stored in it over to other functions as well. In the code used the global variables were used to store the values of the different well logs. This allowed the use of the stored well log values without the need to reload the well log data.

3.2 The GUI

Figure 3.1 shows the designed GUI interface for the main screen, the buttons for different processes to be executed are organized into corresponding button groups. The push buttons used to display individual logs are categorized in the 'logs' button group, velocity analysis tools are organized into the 'velocities' button group, and similarly so with rock physics, porosity and volume of shale and water saturation.

If any user input variables are required then they are placed in the corresponding button group's text box e.g. for display of density porosity log the variable values for matrix density and matrix fluid density are required which must be written within the assigned text boxes. However, if matrix density and matrix fluid density values are entered into the text boxes of 'Volume of Shale & Sw' then the log of density porosity will fail to load in the axes

The radio buttons in the 'velocities' and 'volume of shale &Sw' are used to select one of two options. For the velocities they are used to select the units of the DT log and for the Volume of shale & Sw they are used to select the variant of the Larionov formula.

At the top left of the GUI the tool bar is visible which is used to interact with the axes, it can be used to pan the graph, select data cursor, zoom in, zoom out, print, save figure, place point, draw line, draw rectangle. At the top right of the GUI the three large push buttons from left to right are used to load data from wells in ' *.xlsx' format, access the Cross-plot GUI and access the GUI to display all logs side by side.

It should be noted that when zoom in is selected the axes can be right clicked to open a menu where horizontal only zooming and vertical only zooming are available for selection.

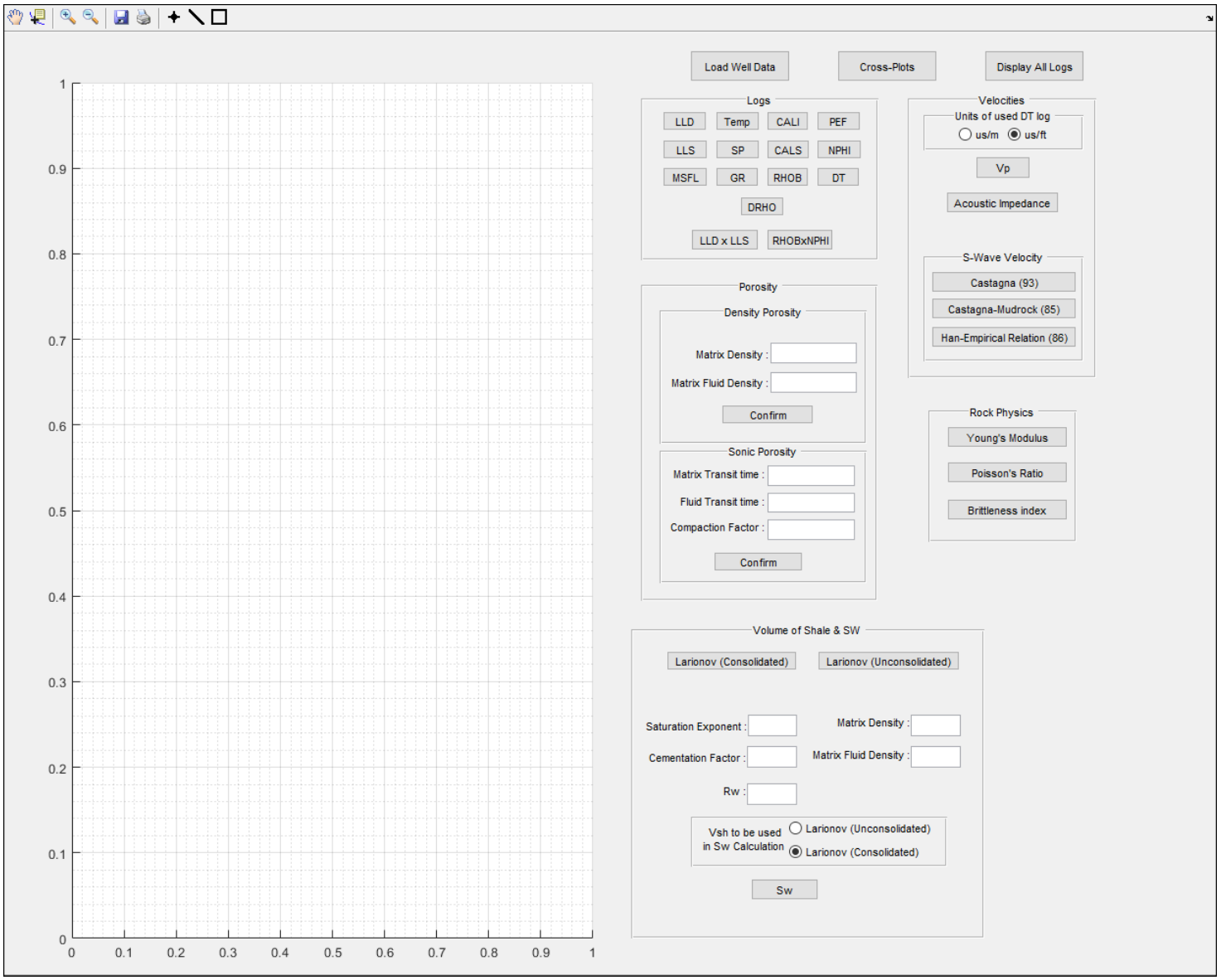


Figure (3.1) WellGUI.m interface

3.3 Data Loading

For any processing or interpretation to be done the data must first be loaded into the software. The data is loaded from an excel file using the following code.

```
419 -    global Depth;
420 -    global LLD;
421 -    global NPHI;
422 -    global DRHO;
423 -    global CALS;
424 -    global RHOB;
425 -    global GR;
426 -    global MSFL;
427 -    global CALI;
428 -    global PEF;
429 -    global SP;
430 -    global LLS;
431 -    global Temp;
432 -    global DT;
433
434
435      %selecting and loading excel file
436 -    [filename,filepath]=uigetfile('*.xlsx');
437 -    [Well,LogName,~]=xlsread([filepath filename]);
438
439      %removing invalid readings from data
440 -    for i=1:numel(Well)
441 -        if Well(i) == -999.25
442 -            Well(i) = NaN;
443 -        end
444 -    end
```

Figure (3.2)-code block for data loading and removal of -999.25 values

In the figure (3.2) 419-432 are used to initialize global variables which will store the values of their corresponding logs.

In line 436 the function “uigetfile” opens a browser UI (User Interface) which is used to navigate to and select the appropriate excel file containing the well log readings. The function however, will not load the file it will merely store the name of selected file in variable ‘filename’ and store the selected files directory in the variable filepath

Line 437 will be the used to load the data of the file selected by line 436, the ‘xlsread’ function will go to the selected file and store the any numeric values in the variable ‘well’ and store any string values in the variable ‘LogName’. If a single variable was used in place of ‘[Well,Logname,~]’ the entire data of the file would have been stored in the single variable instead of being separated into string and numerical data. This separation aids us in further processing of data as well as the separation of the logs by their names string.

Line 440-444 the for loop is used to replace all -999.25 with NaN which indicates that the value is null or invalid. This is done in order to avoid unnecessary errors in further calculations and plots.

In figure (3.3) on the following page the IF statement covering line 447-489. Is used to distinguish and separate the different log data available. It will identify the name of the logs and according to which column of the excel file contains the name of the log the program will assign the log data into its appropriate variable in the program for further use. This is achieved simply by using matrix indexing options. E.g. If Depth is stored in the first column then the program will detect the string in the cell which has its title as ‘DEPTH’ or ‘Depth’ it will not however detect the column title if it is written as ‘depth’. Next, once the program has detected that the depth logs presence and its column location in the excel file then it will use the column number (i) and use it to assign the log values to the variable ‘Depth’ using ‘well (: , i)’. The ‘well’ as described above contains the data of all the well logs and the ‘(: , i)’ is used to signify that only the column ‘i’ is to be singled out. This is repeated will all logs with following names (any log names not listed are ignored. These names are case sensitive):

→Depth or DEPTH	→LLD or ResD
→NPHI or PHIN	→DRHO
→CALS	→RHOB
→GR	→MSFL or ResM
→CALI	→PEF
→SP	→LLS or ResS
→DT	→Temp

```
447 -   for i=1:length(LogName)
448 -       if  count(LogName(i),"Depth") ||count(LogName(i),"DEPTH")~=0
449 -           Depth=Well(:,i);
450 -       end
451 -       if  count(LogName(i),"LLD") || count(LogName(i),"ResD")~=0
452 -           LLD=Well(:,i);
453 -       end
454 -       if  count(LogName(i),"NPHI") || count(LogName(i),"PHIN")~=0
455 -           NPHI=Well(:,i);
456 -       end
457 -       if  count(LogName(i),"DRHO")~=0
458 -           DRHO=Well(:,i);
459 -       end
460 -       if  count(LogName(i),"CALS")~=0
461 -           CALS=Well(:,i);
462 -       end
463 -       if  count(LogName(i),"RHOB")~=0
464 -           RHOB=Well(:,i);
465 -       end
466 -       if  count(LogName(i),"GR")~=0
467 -           GR=Well(:,i);
468 -       end
469 -       if  count(LogName(i),"MSFL") || count(LogName(i),"ResM")~=0
470 -           MSFL=Well(:,i);
471 -       end
472 -       if  count(LogName(i),"CALI")~=0
473 -           CALI=Well(:,i);
474 -       end
475 -       if  count(LogName(i),"PEF")~=0
476 -           PEF=Well(:,i);
477 -       end
478 -       if  count(LogName(i),"SP")~=0
479 -           SP=Well(:,i);
480 -       end
481 -       if  count(LogName(i),"LLS") || count(LogName(i),"ResS")~=0
482 -           LLS=Well(:,i);
483 -       end
484 -       if  count(LogName(i),"DT")~=0
485 -           DT=Well(:,i);
486 -       end
487 -       if  count(LogName(i),"Temp")~=0
488 -           Temp=Well(:,i);
489 -       end
490
```

Figure (3.3) code block for separation of well data into respective logs.

Now the data has been loaded into matlab and is ready for further calculation and processing.

```
108      % --- Executes on button press in LLD.
109      function LLD_Callback(hObject, eventdata, handles)
110      % hObject    handle to LLD (see GCBO)
111      % eventdata  reserved - to be defined in a future version of MATLAB
112      % handles    structure with handles and user data (see GUIDATA)
113 -    global Depth;
114 -    global LLD;
115
116 -    plot(LLD,Depth)
117 -    hold on
118 -    xlabel('LLD');ylabel('Depth');grid on;grid minor;
119 -    hold off
120      %adjusting axes properties
121 -    axis tight
122 -    ax = gca;
123 -    ax.XAxisLocation = 'top';
124 -    ax.YDir = 'reverse';
```

Figure (3.4). display of LLD log

3.4 Singular log Display

The block of code in figure 3.4 from line 116-124 shows how the graph for the LLD log is created. All other log graphs are created using the same method changing only the variables used. First the logs of depth and LLD are plotted using the plot function after which the x and y labels are added and the minor and major grids are activated. Once activated the axis is adjusted using the “axis tight” which moves the axis borders to the minimum and maximum values from the logs. Examples of other log creation codes are shown below in figures 3.5 and

```
162      % --- Executes on button press in NPHI.
163      function NPHI_Callback(hObject, eventdata, handles)
164      % hObject    handle to NPHI (see GCBO)
165      % eventdata  reserved - to be defined in a future version of MATLAB
166      % handles    structure with handles and user data (see GUIDATA)
167 -    global Depth;
168 -    global NPHI;
169
170 -    plot(NPHI,Depth)
171 -    hold on
172 -    xlabel('NPHI');ylabel('Depth');grid on;grid minor;
173 -    hold off
174      %adjusting axes properties
175 -    axis tight
176 -    ax = gca;
177 -    ax.XAxisLocation = 'top';
178 -    ax.YDir = 'reverse';
```

Figure (3.5) code block for Display of NPHI log

```
288      % --- Executes on button press in PEF.
289      function PEF_Callback(hObject, eventdata, handles)
290      % hObject    handle to PEF (see GCBO)
291      % eventdata  reserved - to be defined in a future version of MATLAB
292      % handles    structure with handles and user data (see GUIDATA)
293 -    global Depth;
294 -    global PEF;
295
296 -    plot(PEF,Depth)
297 -    hold on
298 -    xlabel('PEF');ylabel('Depth');grid on;grid minor;
299 -    hold off
300      %adjusting axes properties
301 -    axis tight
302 -    ax = gca;
303 -    ax.XAxisLocation = 'top';
304 -    ax.YDir = 'reverse';
```

Figure (3.6) Code block for display of PEF log

The simultaneous plots for RHOBxNPHI and LLDxLLS are done slightly different due to constraints imposed by my personal lack of knowledge in matlab programming. These are shown in figure (3.7) and figure(3.8).

```
636      % --- Executes on button press in RHOBxNPHI.
637      function RHOBxNPHI_Callback(hObject, eventdata, handles)
638      % hObject    handle to RHOBxNPHI (see GCBO)
639      % eventdata  reserved - to be defined in a future version of MATLAB
640      % handles    structure with handles and user data (see GUIDATA)
641 -    global Depth;
642 -    global RHOB;
643 -    global NPHI;
644
645 -    nphi = -NPHI;
646 -    plotyy(Depth,RHOB,Depth,nphi)
647 -    xlabel('Depth');grid on;grid minor;
648 -    view(90,90)
```

Figure (3.7) Code block for display of RHOBxNPHI simultaneous plot

In figures 3.7-3.8 instead of the plot function the plotyy function is used. This specific plot function allows for the simultaneous plotting of two y-axis variables alongside a single x-axis variable. As it only allows for two y-axis variable and not two x-axis variables To overcome this hurdle the view ([90,90]) function is used. This rotates the view angle of the plot to be rotated so visually the y-axes appears as the x-axis and vice versa. This rotates the view angle of the plot

```
656     % --- Executes on button press in LLDxLLS.
657     function LLDxLLS_Callback(hObject, eventdata, handles)
658     % hObject    handle to LLDxLLS (see GCBO)
659     % eventdata  reserved - to be defined in a future version of MATLAB
660     % handles    structure with handles and user data (see GUIDATA)
661 -   global Depth;
662 -   global LLD;
663 -   global LLS;
664
665 -   plotyy(Depth,LLD,Depth,LLS)
666 -   [AX,H1,H2]=plotyy(Depth,LLD,Depth,LLS,@semilogy,@semilogy)
667 -   linkaxes(AX,'x');
668 -   set(AX(2),'XTickLabel',[]);
669 -   xlabel('Depth');grid on;grid minor;
670 -   view([90,90])
```

Figure (3.8) Code block for simultaneous display of LLDxLLS log

This rotates the view angle of the plot to be rotated so visually the y-axes appears as the x-axis and vice versa. Also an addition to be noted is the "@semilogy, @semilogy" at the end of line 666 and [AX,H1, H2] at the start. The start simply means that the variable AX will hold the axes properties for both y-axes, while H1 and H2 will hold the values of the said axes. Following this, in line 667 and 668the linkaxes () function is used along with "set (AX (2),'XTickLabel',[]);" and will 'link' both x-axes or merge them so there are no inconsistencies of the x-axes with either of the y-axes.

In addition to this change, for the RHOBxNPHI simultaneous plot in figure 3.7 the values of NPHI could not be reversed also due to my lack of knowledge in matlab programming, hence all values of NPHI log are made negative values so that visually the plot appears as it should.

Figure 3.9 shows the code used to generate the Density porosity log. In the "get (handles.Pm,'string')" function the get() function is used to retrieve values from the GUI that can be edited. This specific get function retrieves the contents of the text box which will contain the matrix density entered. It does so by first locating the textbox with the tag 'Pm' signified by 'handles.Pm' then continues to retrieve the 'string' section of the selected text box via 'string'.

As the value retrieved is a of the string type it must be changed into a number so that it can be properly calculated further. This is done using the conversion function 'str2num()'. This function as its name suggests converts the string inside the function parenthesis into a numerical values.

The formula used to calculate the density porosity is PhiD=(RhoM-RhoB)/(RhoM-RhoF). The RhoM is the matrix density, RhoF is the matrix fluid density RHOB is the density log and PhiD is the density porosity. The plotting for the density porosity x depth is done in the same method as that for other logs.In Figure 3.10 the sonic porosity is calculated using PhiS=(DT-DTma) / (DTfld

– Dtma) * (1/cp). Where PhiS is sonic porosity, DTma is matrix transit time, DTfld is fluid transit time and cp is compaction factor. Other aspects of this code block are the same as in figure 3.9.

```
386      % --- Executes on button press in PorRHOB.
387      function PorRHOB_Callback(hObject, eventdata, handles)
388      % hObject    handle to PorRHOB (see GCBO)
389      % eventdata  reserved - to be defined in a future version of MATLAB
390      % handles    structure with handles and user data (see GUIDATA)
391      global Depth;
392      global RHOB;
393
394
395
396
397      Pm = get(handles.Pm,'string');
398      Pm = str2num(Pm);
399      Pf = get(handles.Pmf,'string');
400      Pf = str2num(Pf);
401
402      porRHOB=(Pm-RHOB./(Pm-Pf));
403
404      plot(porRHOB,Depth)
405      hold on
406      xlabel('Density Porosity');ylabel('Depth');grid on;grid minor;
407      hold off
408      %adjusting axes properties
409      axis tight
410      ax = gca;
411      ax.XAxisLocation = 'top';
412      ax.YDir = 'reverse';
```

Figure (3.9) code block for calculation and display of PHID

```
586      % --- Executes on button press in PorDT.
587      function PorDT_Callback(hObject, eventdata, handles)
588      % hObject    handle to PorDT (see GCBO)
589      % eventdata  reserved - to be defined in a future version of MATLAB
590      % handles    structure with handles and user data (see GUIDATA)
591      global DT;
592      global Depth;
593
594      Tm = get(handles.Tm,'string');
595      Tm = str2num(Tm);
596      Tmf = get(handles.Tmf,'string');
597      Tmf = str2num(Tmf);
598      Cp = get(handles.Cp,'string');
599      Cp = str2num(Cp);
600
601      porSONIC = ((DT - Tm)./(Tmf - Tm).*(1./Cp));
602
603      plot(porSONIC,Depth)
604      hold on
605      xlabel('Sonic Porosity');ylabel('Depth');grid on;grid minor;
606      hold off
607      %adjusting axes properties
608      axis tight
609      ax = gca;
610      ax.XAxisLocation = 'top';
611      ax.YDir = 'reverse';
```

Figure (3.10) code block For and display Of sonic porosity

3.5 Volume of Shale

Volume of shale can be calculated using multiple methods, namely: Linear scaling, Clavier shale correction (1971), Steiber shale correction (1970), and Larionov shale correction formulas for Consolidated and Unconsolidated (1969).

A comparison of the methods is given in figure 3.11 observing this figure we can see that linear method overestimates the volume of shale while the two Larionov formulas give a more accurate general trend of volume of shale for their corresponding rock ages as compared to Steiber (1070) and clavier (1971). Hence, in the program the Larionov equations are used. Which are as follows:

(a) Larionov (Tertiary rocks): $Vsh = 0.33[2^{(2*Igr)} - 1]$

(b) Larionov (Older rocks): $Vsh = 0.083[2^{(3.7*Igr)} - 1]$

The implementation of this formula in matlab code is done as shown in figure 3.12 and figure 3.13. First Igr is calculated in Line 349 and line 371 after which the formula for the corresponding equation is applied using the Igr value, the calculated vales of the consolidated

and unconsolidated rocks are then stored in the variables of Vsh_old and Vsh_ter.

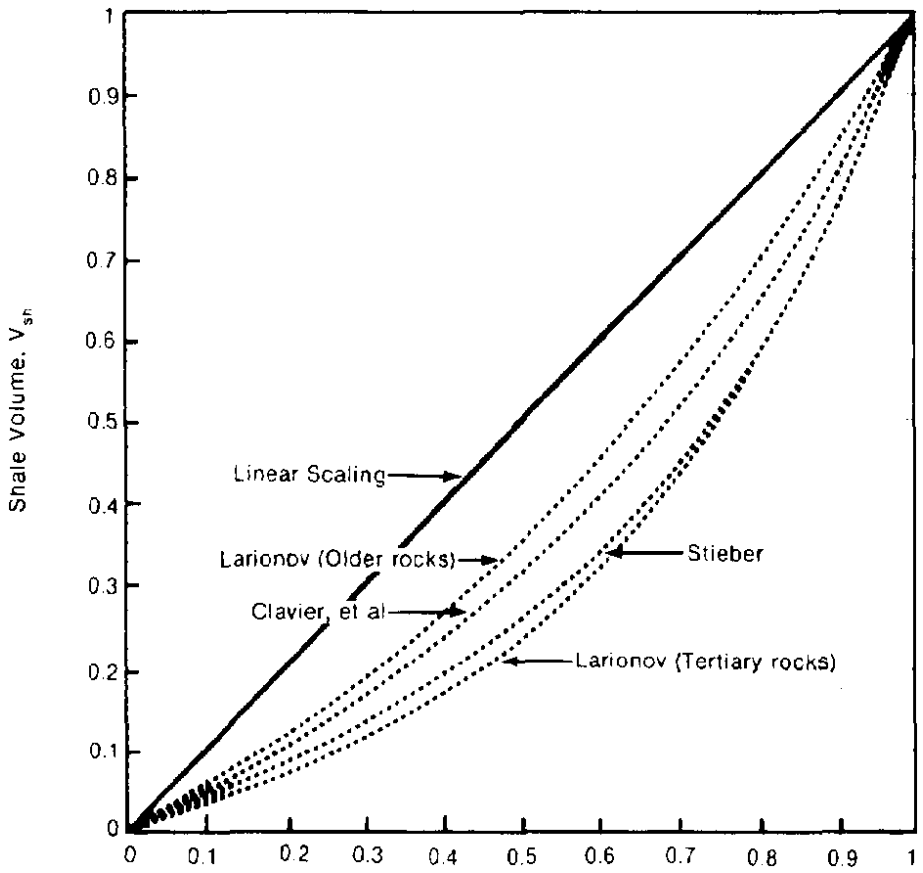


Figure (3.11)

```
341      % --- Executes on button press in Vsh_old.
342      function Vsh_old_Callback(hObject, eventdata, handles)
343      % hObject    handle to Vsh_old (see GCBO)
344      % eventdata  reserved - to be defined in a future version of MATLAB
345      % handles    structure with handles and user data (see GUIDATA)
346 -    global Depth;
347 -    global GR;
348
349 -    Igr=(GR-min(GR))./(max(GR)-min(GR));
350 -    Vsh_old=0.33*(2.^(2.*Igr)-1);
351
352 -    plot(Vsh_old,Depth)
353 -    hold on
354 -    xlabel('Volume of shale(Consolidated)');ylabel('Depth');grid on;grid minor;
355 -    hold off
356      %adjusting axes properties
357 -    axis tight
358 -    ax = gca;
359 -    ax.XAxisLocation = 'top';
360 -    ax.YDir = 'reverse';
```

Figure (3.12) Code block for calculation and display of Vsh Larionov eqution for older rocks

```
363     % --- Executes on button press in Vsh_ter.
364     function Vsh_ter_Callback(hObject, eventdata, handles)
365     % hObject    handle to Vsh_ter (see GCBO)
366     % eventdata  reserved - to be defined in a future version of MATLAB
367     % hglobal Depth;
368 -   global Depth;
369 -   global GR;
370
371 -   Igr=(GR-min(GR))./(max(GR)-min(GR));
372 -   Vsh_ter=0.083.*(2.^(3.7.*Igr)-1);
373
374
375 -   plot(Vsh_ter,Depth)
376 -   hold on
377 -   xlabel('Volume of shale(Unconsolidated');ylabel('Depth');grid on;grid minor;
378 -   hold off
379     %adjusting axes properties
380 -   axis tight
381 -   ax = gca;
382 -   ax.XAxisLocation = 'top';
383 -   ax.YDir = 'reverse';
```

Figure (3.13) Code block for calculation and display of Vsh of tertiary rocks

The implementation of this formula in matlab code is done as shown in figure 3.12 and figure 3.13. First Igr is calculated in Line 349 and line 371 after which the formula for the corresponding equation is applied using the Igr value, the calculated vales of the consolidated and unconsolidated rocks are then stored in the variables of Vsh_old and Vsh_ter. Then the lines 352-360 and 375-383 show how the values of the calculated volume of shale will be plotted onto the axes.

```
1109      %calculating Density Porosity
1110 -    RhoMa = get(handles.RhoMa,'string');
1111 -    RhoMa = str2num(RhoMa);
1112 -    RhoF = get(handles.RhoF,'string');
1113 -    RhoF = str2num(RhoF);
1114
1115 -    PhiD=(RhoMa-RHOB./(RhoMa-RhoF));
1116
1117      %Calculating Vsh using selected method
1118 -    Vsh_formula = get(handles.Vsh_formula,'SelectedObject');
1119 -    Vsh_formula = get(Vsh_formula,'string');
1120
1121 -    Igr=(GR-min(GR))./(max(GR)-min(GR));
1122
1123 -    if Vsh_formula=='Larionov (Consolidated)   '
1124 -        Vsh=0.083.*(2.^(3.7.*Igr)-1);
1125 -    end
1126 -    if Vsh_formula=='Larionov (Unconsolidated)'
1127 -        Vsh=0.33*(2.^(2.*Igr)-1);
1128 -    end
1129
1130      %Calculating average and effective porosity
1131 -    PhiA = (PhiD + NPHI) ./ 2;
1132 -    PhiE = PhiA .* (1-Vsh);
1133
1134      %calculating formaiton factor
1135 -    m = get(handles.CemFac,'string');
1136 -    m = str2num(m);
1137
1138 -    F = 1./(PhiE .* m);
1139
1140      %calculating Sw
1141 -    Rw = get(handles.Rw,'string');
1142 -    Rw = str2num(Rw);
1143 -    n = get(handles.SaturationExp,'string');
1144 -    n = str2num(n);
1145
1146 -    Sw = ( (F .* Rw)./(LLD) ).^(1./n);
1147
1148
1149 -    plot(Sw,Depth)
1150 -    hold on
1151 -    xlabel('Sw');ylabel('Depth');grid on;grid minor;
1152 -    hold off
1153      %adjusting axes properties
1154 -    axis tight
1155 -    ax = gca;
1156 -    ax.XAxisLocation = 'top';
1157 -    ax.YDir = 'reverse';
```

Figure (3.14) Code block for calculation and display of Sw

3.6 Saturation of water

Water saturation is the percentage of pore volume in rock that is occupied by water of formation. If it is not confirmed that pores in the formation are filled by hydrocarbons. It is assume that these are filled with water. To determine the water and hydrocarbon saturation is one of the basic goals of well logging. To calculate saturation of water in the formation a mathematical equation known as the Archie equation is used which is given below.

$$S_w = \sqrt[n]{\frac{F \times R_W}{R_T}},$$

Where, F is formation factor, $F = \frac{a}{\emptyset m}$, Rw represents resistivity of water, Rt represents the true formation resistivity which is applied in use of the lateral log deep (LLD), n represents the saturation exponent with its value varying from 1.8 to 2.5, a is constant and its value is assumed as 1, Ø represents effective porosity, m represents the cementation factor.

Figure 3.14 shows the code used in the button for calculating Sw in the volume of shale & Sw button group. Lines 1110-1113, 1135-1136 and 1141-1144 are used to retrieve the string input then change it into numerical values from the text boxes containing values of matrix density, matrix fluid density, cementation factor, Rw and saturation exponent respectively and must be all input by the user into their respective text boxes. Once the user selects which formula of Vsh to apply using the radio buttons the code will identify the selected radio button using lines 1123-1128. The program first calculates the density porosity which is used in order to calculate effective porosity. Then calculates Vsh using selected formula followed by the calculation of effective porosity and formation factor which is finally used in calculation of saturation of water. Lastly, the calculated Sw values are plotted as shown in lines 1149-1157.

```
673 -    m_ft = get(handles.units,'SelectedObject');
674 -    m_ft = get(m_ft,'string');
675 -    if m_ft=='us/ft'
676 -        Vp=1000000./(DT.*3.28);
677 -    end
678 -    if m_ft=='us/m '
679 -        Vp=1000000./(DT);
680 -    end
```

Figure (3.15) Code for identification of selected unit and calculation of Vp log

3.7 Velocity analysis

After generation of the three petrophysical logs and the volume of shale calculated we then use the DT log to generate the compressional wave velocity within the zone of interest. Velocity is the reciprocal of the sonic transit time. Even on logs with a metric depth scale, the

transit time is mostly still given in μs/ft. The necessary conversions must be applied to extract the metric velocity thus the following equation is used if the DT measurement is in μs/ft.

$$Vp = \frac{1 \times 1{,}000{,}000}{DT \times 3.28}$$

Where, DT is the sonic log. This calculation is done as shown in figure 3.15, where the units of the DT are first selected by identifying which radio button has been selected and the appropriate formula is used i.e. the units are converted into metric units if they are in μs/ft or are unchanged if already in μs/m. this is then plotted in similar way as other plots shows prior.

Acoustic impedance is a layer property of a rock and is equal to the product of compressional velocity and density (Onajite, 2014). The density log and the compressional wave velocity log generated from the DT log are used to compute the acoustic impedance log by following equation

$$AI = Vp \times \rho_b$$

Where, ρ_b is the density of the formation taken from the RHOB log and Vp is the compressional wave velocity calculated from the DT log. In the code this is done as shown in figure 3.16 where lines 961-968 are the exact same and serve the same purpose of calculation Vp as in figure 3.15 and line 970 is the acoustic impedance formula. Once calculated it is plotted using lines 972-980.

After the calculation of the compressional wave velocity log in the zone of interest the shear wave velocity log calculation is the next step for which proper empirical relations are to be considered to be used, considering the lithology of the study zone. There are three key empirical relations, namely:

- Castagna et al., (1993). Least square linear fit to the data

  $$\mathrm{Vs} = 0.804\mathrm{Vp} - 0.856 \ (\mathrm{km/s})$$

- Famous “mudrock” line derived from in-situ data by Castagna et al., (1985)

  $$\mathrm{Vs} = 0.862\mathrm{Vp} - 1.172 \ (\mathrm{km/s})$$

- Empirical relation of Han (1986)

  $$\mathrm{Vs} = 0.794\mathrm{Vp} - 0.787 \ (\mathrm{km/s})$$

Of these three relations, those by Han (1986) and Castagna et al., (1985) are essentially the same and give the best overall fit to the sandstones. The mudrock line predicts systematically lower Vs because it is best suited to the most shaley samples, as seen in Figure 3.17. Castagna et al., (1993) suggests that if the lithology is well known, one can fine tune these relations to slightly lower Vs/Vp for high shale content and higher Vs/Vp in cleaner sands. When the lithology is not well constrained, the Han and Castagna et al., (1985) lines give a reasonable average (Mavko et al., 2009). Figures 3.18-3.20 show the implementation of the three formula into their corresponding push button

```
961 -    m_ft = get(handles.units,'SelectedObject');
962 -    m_ft = get(m_ft,'string');
963 -    if m_ft=='us/ft'
964 -        Vp=1000000./(DT.*3.28);
965 -    end
966 -    if m_ft=='us/m '
967 -        Vp=1000000./(DT);
968 -    end
969
970 -    AI = Vp .* RHOB;
971
972 -    plot(AI,Depth)
973 -    hold on
974 -    xlabel('Acoustic Impedance');ylabel('Depth');grid on;grid minor;
975 -    hold off
976      %adjusting axes properties
977 -    axis tight
978 -    ax = gca;
979 -    ax.XAxisLocation = 'top';
980 -    ax.YDir = 'reverse';
```

Figure (3.16) Code block for calculation and display of Acoustic Impedance log

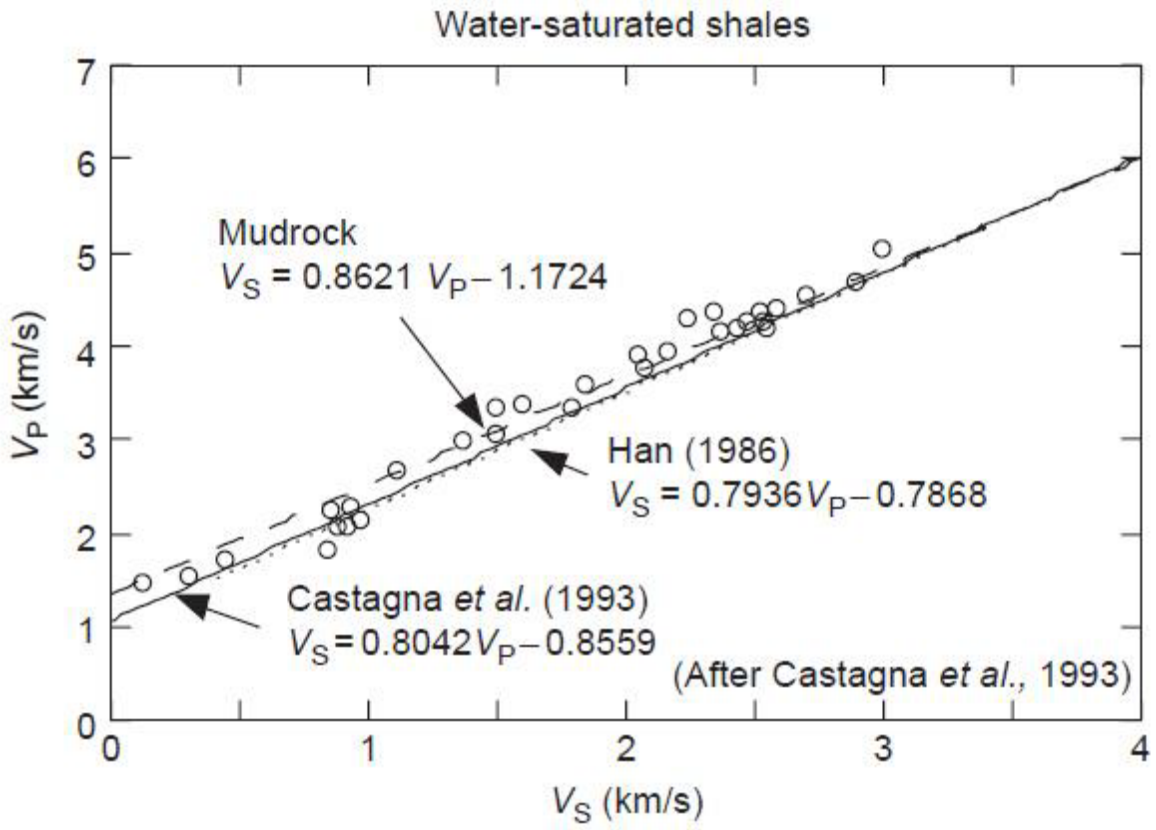


Figure (3.17)

```
m_ft = get(handles.units,'SelectedObject');
m_ft = get(m_ft,'string');
if m_ft=='us/ft'
    Vp=1000000./(DT.*3.28);
end
if m_ft=='us/m '
    Vp=1000000./(DT);
end

Vs = (0.804.*Vp)-0.856;

plot(Vs,Depth)
hold on
xlabel('Vs');ylabel('Depth');grid on;grid minor;
hold off
%adjusting axes properties
axis tight
ax = gca;
ax.XAxisLocation = 'top';
ax.YDir = 'reverse';
```

Figure (3.18) Castagna et al. (1993)

```
732 -    m_ft = get(handles.units,'SelectedObject');
733 -    m_ft = get(m_ft,'string');
734 -    if m_ft=='us/ft'
735 -        Vp=1000000./(DT.*3.28);
736 -    end
737 -    if m_ft=='us/m '
738 -        Vp=1000000./(DT);
739 -    end
740
741
742 -    Vs=(0.862*Vp)-1.172;
743
744 -    plot(Vs,Depth)
745 -    hold on
746 -    xlabel('Vs');ylabel('Depth');grid on;grid minor
747 -    hold off
748      %adjusting axes properties
749 -    axis tight
750 -    ax = gca;
751 -    ax.XAxisLocation = 'top';
752 -    ax.YDir = 'reverse';
```

Figure (3.19) Mudrock line (1985)

```
763 -    m_ft = get(handles.units,'SelectedObject');
764 -    m_ft = get(m_ft,'string');
765 -    if m_ft=='us/ft'
766 -        Vp=1000000./(DT.*3.28);
767 -    end
768 -    if m_ft=='us/m '
769 -        Vp=1000000./(DT);
770 -    end
771
772
773 -    Vs=(0.794*Vp)-0.787;
774
775 -    plot(Vs,Depth)
776 -    hold on
777 -    xlabel('Vs');ylabel('Depth');grid on;grid minor;
778 -    hold off
779      %adjusting axes properties
780 -    axis tight
781 -    ax = gca;
782 -    ax.XAxisLocation = 'top';
783 -    ax.YDir = 'reverse';
```

Figure (3.20) Han(1986)

3.8 Young's Modulus

This Modulus is obtained to measure the stiffness of the material. The relation between the density, compressional wave velocity, young's modulus, and shear wave velocity is given in the following equation.

$$E = \frac{\rho Vs^2(3Vp^2 - 4Vs^2)}{Vp^2 - Vs^2}$$

Where, ρ is the density that is obtained from the density (RHOB) log, Vs and Vp are the shear wave and compressional wave velocity respectively which are obtained from the sonic log (DT). The code for the applications push button of young's modulus is given in figure 3.21 below

```
786     function YoungModulus_Callback(hObject, eventdata, handles)
787     % hObject    handle to YoungModulus (see GCBO)
788     % eventdata  reserved - to be defined in a future version of MATLAB
789     % handles    structure with handles and user data (see GUIDATA)
790 -   global Depth;
791 -   global Vs;
792 -   global RHOB;
793 -   global DT;
794
795
796 -   m_ft = get(handles.units,'SelectedObject');
797 -   m_ft = get(m_ft,'string');
798 -   if m_ft=='us/ft'
799 -       Vp=1000000./(DT.*3.28);
800 -   end
801 -   if m_ft=='us/m '
802 -       Vp=1000000./(DT);
803 -   end
804
805
806     %computed by calculating numerator and denomenator seperately
807 -   num = (RHOB .* Vs.^2) .* (3.*Vp.^2 - 4.*Vs.^2);
808 -   den = Vp.^2 .* Vs.^2;
809
810 -   youngs = num./den;
811
812 -   plot(youngs,Depth)
813 -   hold on
814 -   xlabel('Youngs modulus');ylabel('Depth');grid on;grid minor;
815 -   hold off
816     %adjusting axes properties
817 -   axis tight
818 -   ax = gca;
819 -   ax.XAxisLocation = 'top';
820 -   ax.YDir = 'reverse';
```

Figure (3.21) Code block for calculation and display of young's modulus

3.9 Poisson's Ratio

The Poisson's ratio is used to indicate the maturity of the shale oil/gas zone. The low value of poisson's ratio will indicate the mature oil/gas shale zone. The relation between the poisson's ratio, compressional wave velocity, and shear wave velocity is given in following equation (Mavko et al., 2009).

$$\sigma = \frac{\left(\frac{Vp}{Vs}\right)^2 - 2}{2\left(\frac{Vp}{Vs}\right)^2 - 2},$$

Where, Vp/VS is the ratio of compressional wave and shear wave velocities. Figure 3.22 shows the implementation of the formula into the matlab code

```
833 -    m_ft = get(handles.units,'SelectedObject');
834 -    m_ft = get(m_ft,'string');
835 -    if m_ft=='us/ft'
836 -        Vp=1000000./(DT.*3.28);
837 -    end
838 -    if m_ft=='us/m '
839 -        Vp=1000000./(DT);
840 -    end
841
842
843 -    poisson=( (Vp./Vs) .^2 -2 ) ./ ( 2.*(Vp./Vs).^2 -2);
844
845 -    plot(poisson,Depth)
846 -    hold on
847 -    xlabel('Poisson ratio');ylabel('Depth');grid on;grid minor;
848 -    hold off
849      %adjusting axes properties
850 -    axis tight
851 -    ax = gca;
852 -    ax.XAxisLocation = 'top';
853 -    ax.YDir = 'reverse';
```

Figure (3.22) Code block for calculation and display of Poisson ratio

3.10 Brittleness

This is used to measure the energy that can be stored in the rock before the rock is going to fracture. This property is dependent on the lithology, texture, effective stress and some other important parameters. To estimate this parameter, the poisson's ratio and the young's modulus are the main key parameter. The brittleness index (B) can be estimated by using the relationship of Rickman et al., (2008) from the well log data.

$$B \approx \frac{(B_E + B_\sigma)}{2}$$

Where, $$B_E \approx \frac{E - E_{MIN}}{E_{MAX} - E_{MIN}}, \text{ and } B_\sigma \approx \frac{\sigma - \sigma_{MIN}}{\sigma_{MAX} - \sigma_{MIN}}$$

Implementation of the Brittleness index formula in the applications corresponding push button is shown in the block of code below (figure 3.23).

```
867 -    m_ft = get(handles.units,'SelectedObject');
868 -    m_ft = get(m_ft,'string');
869 -    if m_ft=='us/ft'
870 -        Vp=1000000./(DT.*3.28);
871 -    end
872 -    if m_ft=='us/m '
873 -        Vp=1000000./(DT);
874 -    end
875
876      %calculating youngs modulus
877 -    num = (RHOB .* Vs.^2) .* (3.*Vp.^2 - 4.*Vs.^2);
878 -    den = Vp.^2 .* Vs.^2;
879
880 -    youngs = num./den;
881
882      %calculating poisson ratio
883 -    poisson=( (Vp./Vs).^2-2 ) ./ ( 2*(Vp./Vs).^2-2);
884
885      %calculating brittleness index
886 -    By = (youngs - min(youngs)) ./ (max(youngs)-min(youngs));
887 -    Bp = (poisson - min(poisson)) ./ (max(poisson)-min(poisson));
888
889 -    BI= (By + Bp) ./ 2;
890
891 -    plot(BI,Depth)
892 -    hold on
893 -    xlabel('Brittleness Index');ylabel('Depth');grid on;grid minor;
894 -    hold off
895      %adjusting axes properties
896 -    axis tight
897 -    ax = gca;
898 -    ax.XAxisLocation = 'top';
899 -    ax.YDir = 'reverse';
```

Figure (3.23) Code block for calculation and display of Brittleness Index

This concludes the code used in the main GUI named “WellGUI.m” now the two secondary GUIs shall be discussed starting with the simple method of accessing them from the main “WellGUI.m”. This is done simply by calling the name of the target gui in the callback for the corresponding push button as shown below in figure 3.24. it shows that only a single word need be written to open the corresponding GUI which is the name of the “.m” file containing the code for the GUI.

```
937      % --- Executes on button press in wellprofile.
938      function wellprofile_Callback(hObject, eventdata, handles)
939      % hObject    handle to wellprofile (see GCBO)
940      % eventdata  reserved - to be defined in a future version of MATLAB
941      % handles    structure with handles and user data (see GUIDATA)
942 -    Wellprofile
943
944      % --- Executes on button press in crossplot.
945      function crossplot_Callback(hObject, eventdata, handles)
946      % hObject    handle to crossplot (see GCBO)
947      % eventdata  reserved - to be defined in a future version of MATLAB
948      % handles    structure with handles and user data (see GUIDATA)
949 -    CrossPlots
```

Figure (3.24) Code for opening the Wellprofile.m and CrossPlots.m

Figure (3.25) GUI of wellprofile.m

3.11 Well Profile

The title and name of the ".m" file can prove to be very misleading as due to constraints on my programming knowledge they are not true well log profiles but only side by side displays of all logs which can be viewed in the main WellGUI.m as shown below in figure 3.25.

```
135 -    if  exist('LLD','var') == 0
136 -        LLD=NaN;
137 -    end
138 -    if  exist('NPHI','var') == 0
139 -        NPHI=NaN;
140 -    end
141 -    if  exist('DRHO','var') == 0
142 -        DRHO=NaN;
143 -    end
144 -    if  exist('CALS','var') == 0
145 -        CALS=NaN;
146 -    end
147 -    if  exist('RHOB','var') == 0
148 -        RHOB=NaN;
149 -    end
150 -    if  exist('GR','var') == 0
151 -        GR=NaN;
152 -    end
153 -    if  exist('MSFL','var') == 0
154 -        MSFL=NaN;
155 -    end
156 -    if  exist('CALI','var') == 0
157 -        CALI=NaN;
158 -    end
159 -    if  exist('PEF','var') == 0
160 -        PEF=NaN;
161 -    end
162 -    if  exist('SP','var') == 0
163 -        SP=NaN;
164 -    end
165 -    if  exist('LLS','var') == 0
166 -        LLS=NaN;
167 -    end
168 -    if  exist('DT','var') == 0
169 -        DT=NaN;
170 -    end
```

Much like in the main WellGUI.m the plots can be interacted with using the tools available in the tool bar at the top left of the window. At the center top is the "Load" button which uses the same coding as shown in figure 3.2 and 3.3. However, instead of simply loading in the data this will in addition plot all graphs onto their corresponding axes. This is done by simply adding a single line of code "axes (handles.axes1)" which is a function that changes the currently selected axes to the axes mentioned after "handles". The axis is repeatedly changed for each plot and the code used for plotting purposes is the same as shown in figure 3.4.

A very important and notable addition of code in the wellprofile.m is the inclusion of a series of existence checks shown in figure 3.26 that verify if a log exists in the given data or not. If the log does not exist then its values are replaced by "NaN" or a null value. If these checks are not put in place then the absence of any log listed will result in the termination of the running program and failure in plotting any logs ahead of the absent one.

Figure (3.26) if statement chain used to check for variable existence

3.12 Cross-plots

Figure 3.27 shows the GUI used for the cross-plots. At the top left the “Load Data” push button contains the same code as shown in Figure 3.2 and Figure 3.3. at the center top are two list boxed from which a log can be selected. Once the first and second logs are selected and the push button “Update Plot” is pressed the program will plot the selected logs as a scatter plot in the large axes to the left and plot the singular selected logs against depth on the two smaller axes to the right.

Figure 3.28-3.30 show the callback code for the “Update Plot” push button. In figure 3.28 and figure 3.29 a switch statement is used to identify which log has been selected in each list box. First the selected ‘value’ or index of the selected log is identified which is then used to assign values to the ‘x’ variable and the ‘x-tag’ in figure 3.28 and ‘y’ variable and the ‘y-tag’ in figure 3.29. The ‘x’ and ‘y’ variable contain the values of the selected log while the ‘xtag’ and ‘ytag’ variables contain the name of the selected logs.

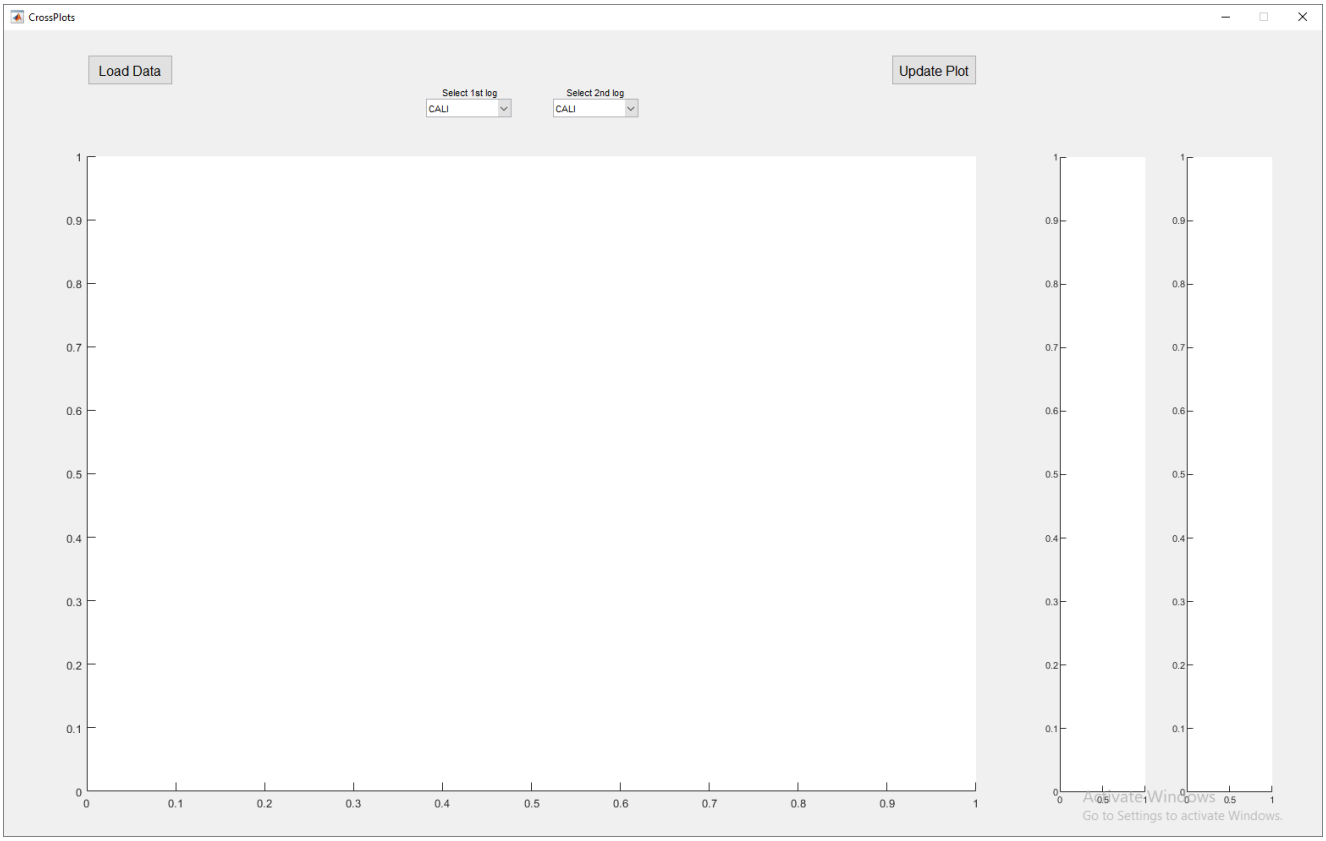


Figure (3.27) the GUI for the CrossPlots.m

Next in figure 3.29 from line 321-328 the large axes to the right is selected and the selected logs are plotted in it as a scatter plot. Lines 330-350 the selected logs are plotted against depth on the two smaller axes on the right of the GUI.

```
232 -    popup_sel_index = get(handles.FirstLog, 'Value');
233 -    switch popup_sel_index
234 -        case 1
235 -            x = CALI;
236 -            xtag = 'CALI';
237 -        case 2
238 -            x = CALS;
239 -            xtag = 'CALS';
240 -        case 3
241 -            x = Temp;
242 -            xtag = 'Temp';
243 -        case 4
244 -            x = GR;
245 -            xtag = 'GR';
246 -        case 5
247 -            x = DT;
248 -            xtag = 'DT';
249 -        case 6
250 -            x = LLD;
251 -            xtag = 'LLD';
252 -        case 7
253 -            x = LLS;
254 -            xtag = 'LLS';
255 -        case 8
256 -            x = MSFL;
257 -            xtag = 'MSFL';
258 -        case 9
259 -            x = NPHI;
260 -            xtag = 'NPHI';
261 -        case 10
262 -            x = RHOB;
263 -            xtag = 'RHOB';
264 -        case 11
265 -            x = PEF;
266 -            xtag = 'PEF';
267 -        case 12
268 -            x = SP;
269 -            xtag = 'SP';
270 -        case 13
271 -            x = DRHO;
272 -            xtag = 'DRHO';
273
```

Figure 3.28

```
275 -    popup_sel_index = get(handles.SecondLog, 'Value');
276 -    switch popup_sel_index
277 -        case 1
278 -            y = CALI;
279 -            ytag = 'CALI';
280 -        case 2
281 -            y = CALS;
282 -            ytag = 'CALS';
283 -        case 3
284 -            y = Temp;
285 -            ytag = 'Temperature';
286 -        case 4
287 -            y = GR;
288 -            ytag = 'GR';
289 -        case 5
290 -            y = DT;
291 -            ytag = 'DT';
292 -        case 6
293 -            y = LLD;
294 -            ytag = 'LLD';
295 -        case 7
296 -            y = LLS;
297 -            ytag = 'LLS';
298 -        case 8
299 -            y = MSFL;
300 -            ytag = 'MSFL';
301 -        case 9
302 -            y = NPHI;
303 -            ytag = 'NPHI';
304 -        case 10
305 -            y = RHOB;
306 -            ytag = 'RHOB';
307 -        case 11
308 -            y = PEF;
309 -            ytag = 'PEF';
310 -        case 12
311 -            y = SP;
312 -            ytag = 'SP';
313 -        case 13
314 -            y = DRHO;
315 -            ytag = 'DRHO';
```

Figure 3.29

```
321 -       axes(handles.axes1)
322 -       scatter(x,y);
323 -       hold on
324 -       xlabel(xtag);
325 -       ylabel(ytag);
326 -       grid on;grid minor;
327 -       axis tight
328 -       hold off
329
330 -       axes(handles.axes2)
331 -       plot(x,Depth)
332 -       hold on
333 -       xlabel(xtag);
334 -       grid on;grid minor;
335 -       axis tight
336 -       ax = gca;
337 -       ax.XAxisLocation = 'top';
338 -       ax.YDir = 'reverse';
339 -       hold off
340
341 -       axes(handles.axes3)
342 -       plot(y,Depth)
343 -       hold on
344 -       xlabel(ytag);
345 -       grid on;grid minor;
346 -       axis tight
347 -       ax = gca;
348 -       ax.XAxisLocation = 'top';
349 -       ax.YDir = 'reverse';
350 -       hold off
```

Figure 3.30

Chapter 4:

Showcase/Example

This chapter focuses on comparison of displayed results of similar well log processing techniques on the BALKASSAR-OXY-01 well using the professionally made program, IHS Kingdom and my own GUI. This is done by comparing the displayed results of both programs side by side.

Two extremely key criteria for the comparision to be noted are as follows:

1. The used well log is NOT conditioned and is used as was given.
2. The comparison is limited only to the initially displayed results and not the interpretation of the results.

The reason that these two are set as criteria is because if the displayed results using unconditioned well log data are the same then there is no reason to suggest that condition data would show differently. Secondly, if displayed results are the same despite the unconditioned data then it only further shows the similarity of the self-made GUI.

Furthermore, the purpose of not performing interpretation of the well log results is simply because that is not the focus of this thesis. This thesis focuses on the creation of an accurate self-made GUI for well log interpretation and if the displayed results are similar to a professionally made program then surely the interpretation of these similar results will also be similar.

4.1 Display of all logs

Figure 4.1 and figure 4.2 show the result of displaying all logs side by side while using the HIS Kingdom software and the self-made GUI. As can be seen immediately at first glance there are more logs displayed in the IHS kingdom display than in the self-made GUI. This is because the GUI is not programmed to read those said logs.

4.2 Comparison of singular logs

Figures 4.3 to figure 4.20 show the result of singular well log displays side by side with the singular well log displays from the WellGUI.m program. Namely, figures 4.3, 4.5, 4.7, 4.9, 4.11, 4.13, 4.15, 4.17 and 4.19 show results from the IHS Kingdom software while figures 4.4, 4.6, 4.8, 4.10, 4.12, 4.14, 4.16, 4.18 and 4.20. The first visible difference in the logs are that of scale. This is especially prevalent in the LLD, LLS and MSFL logs as they are on a log scale which is heavily affected by the scaling of the axes.

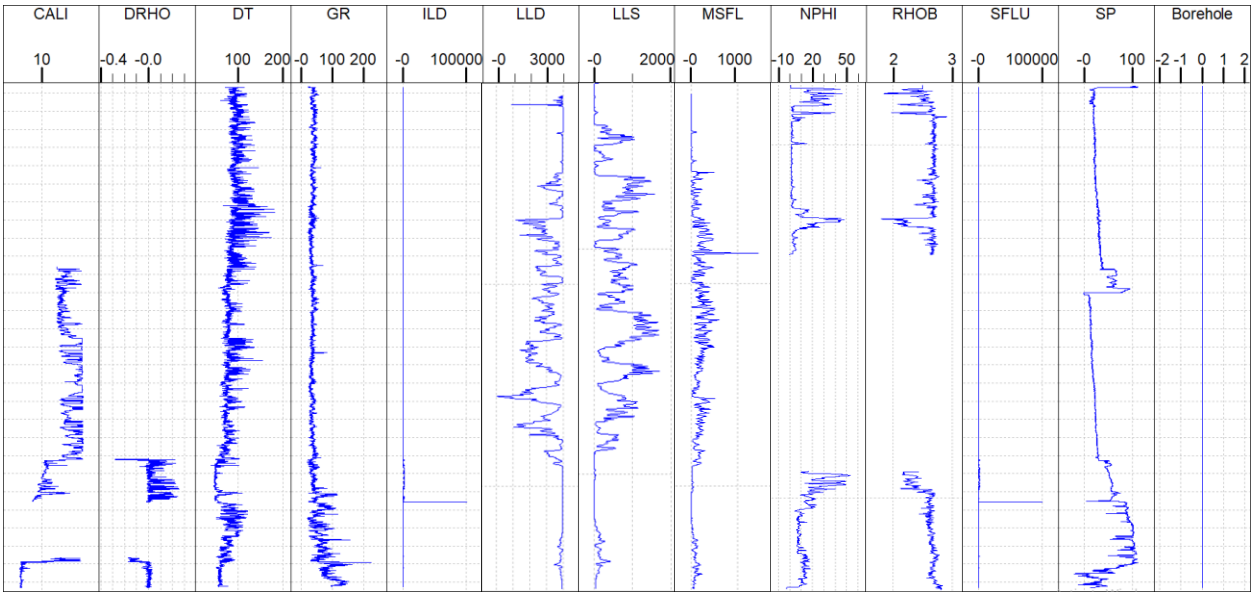


Figure (4.1) display of all logs using IHS Kingdom

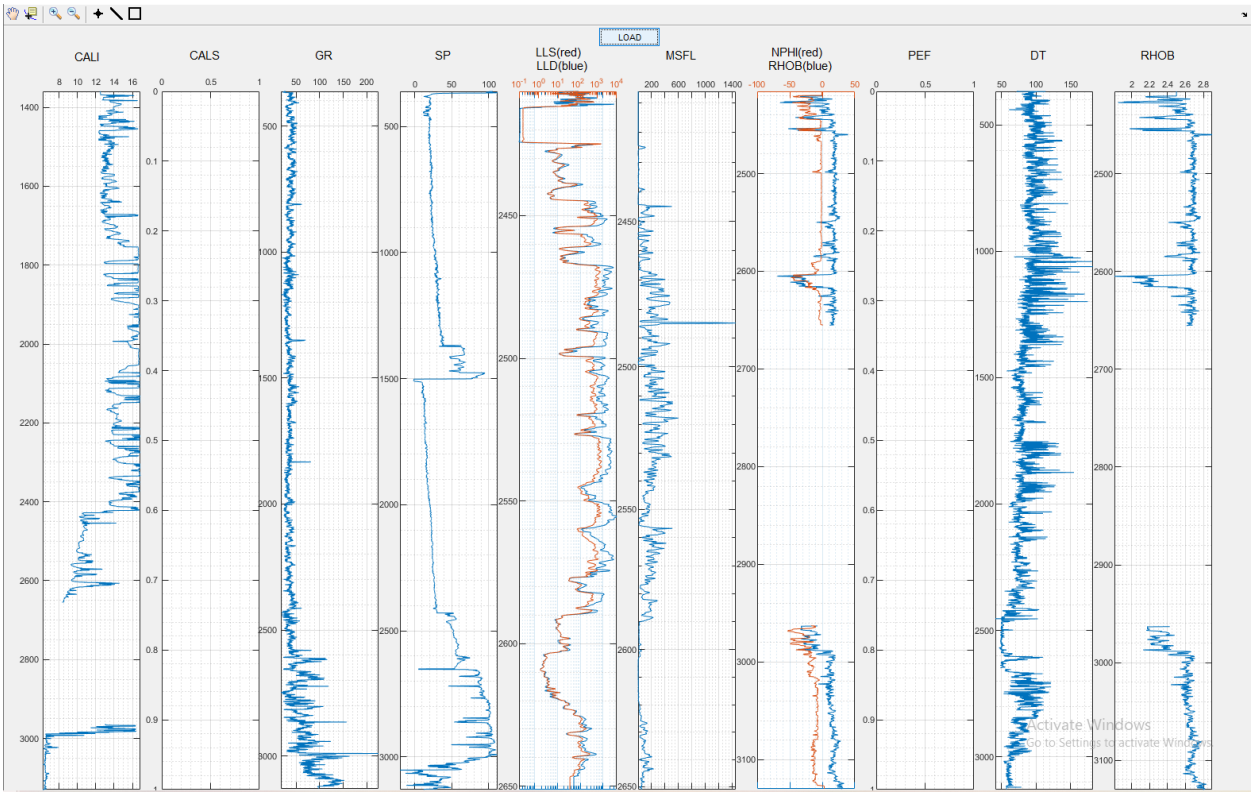


Figure (4.2) display of all logs using self-made GUI

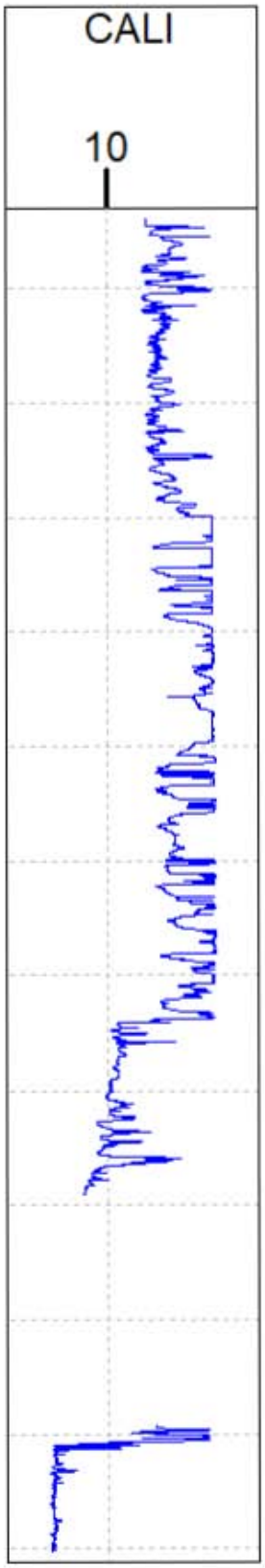


Figure (4.3) CALI from Kingdom

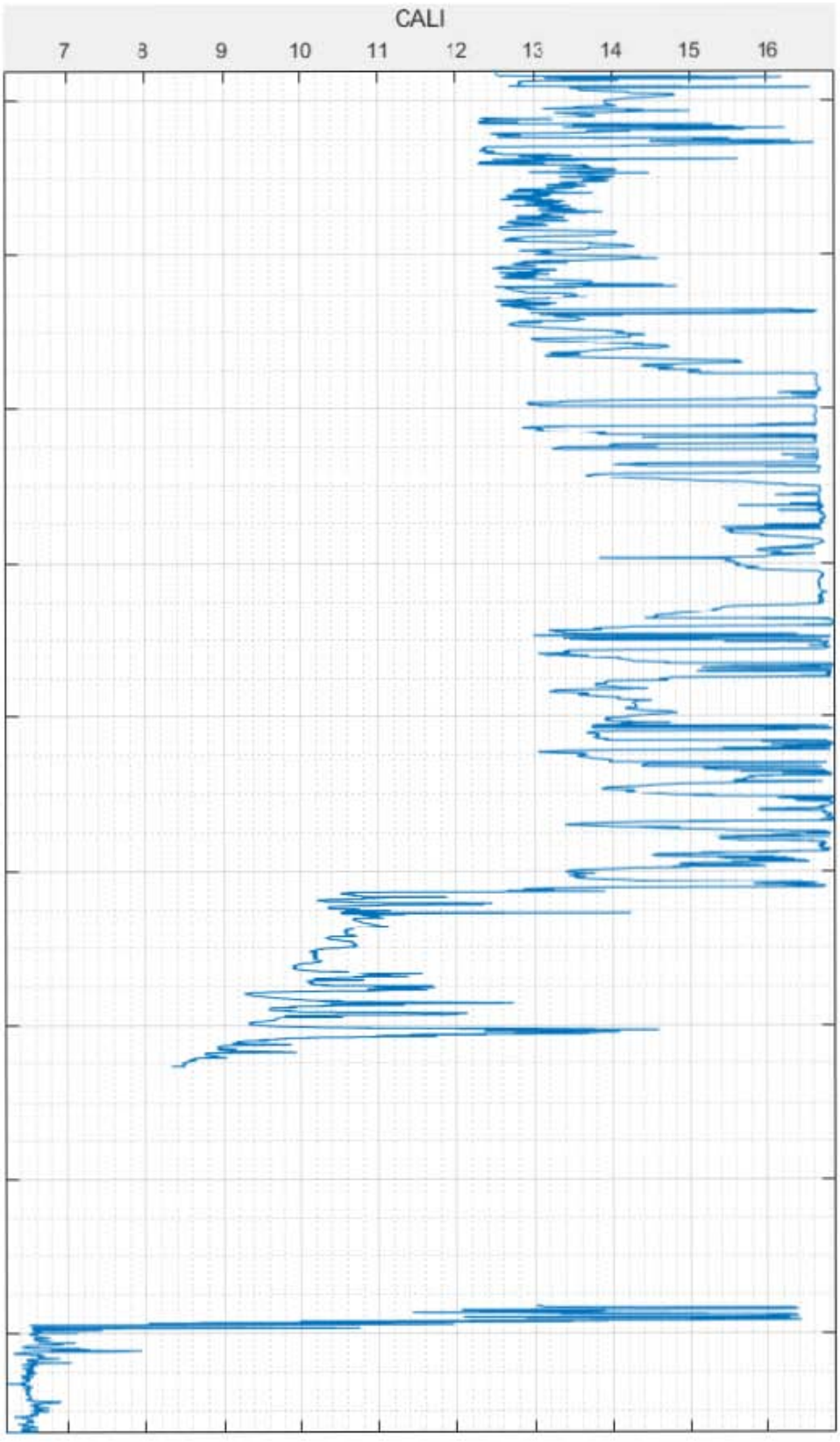


Figure (4.4) CALI from WellGUI.m

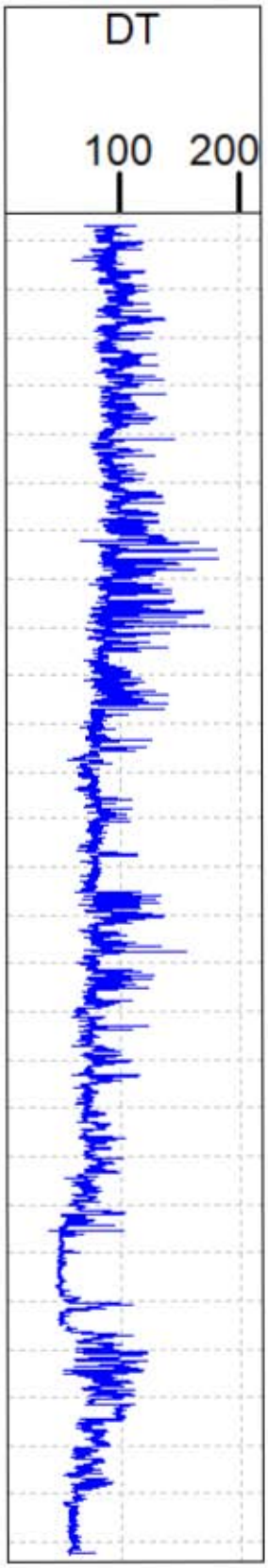
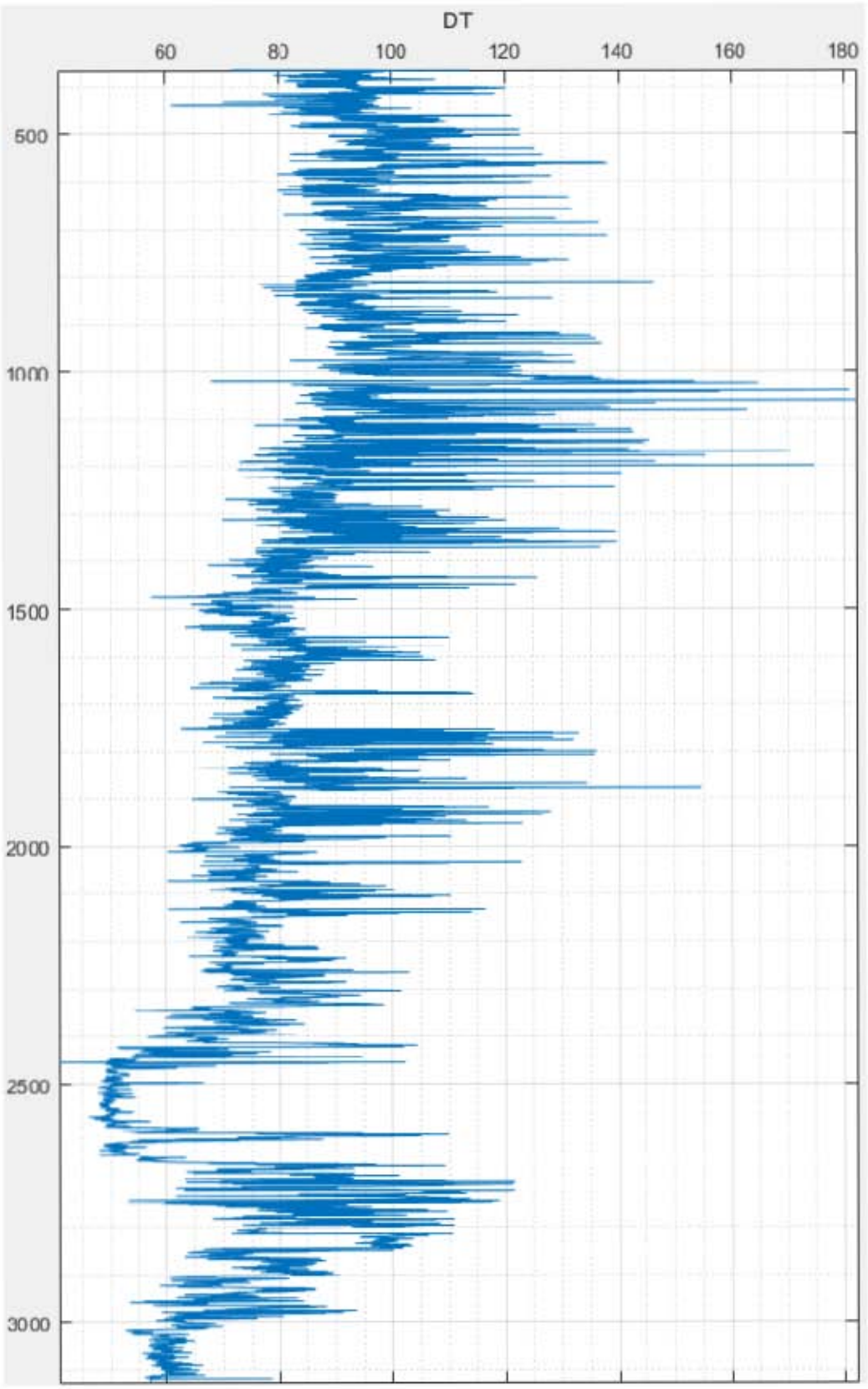


Figure (4.5) DT from Kingdom



Figure(4.6) DT from WellGUI.m

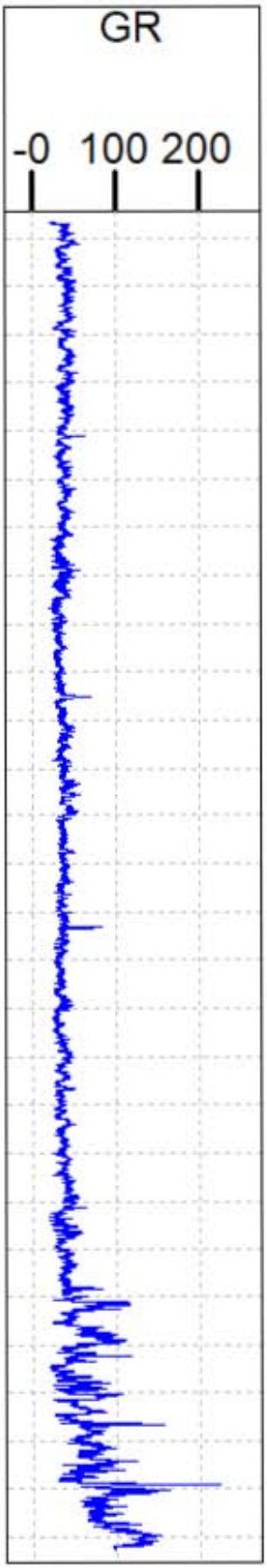


Figure (4.7) GR from Kingdom

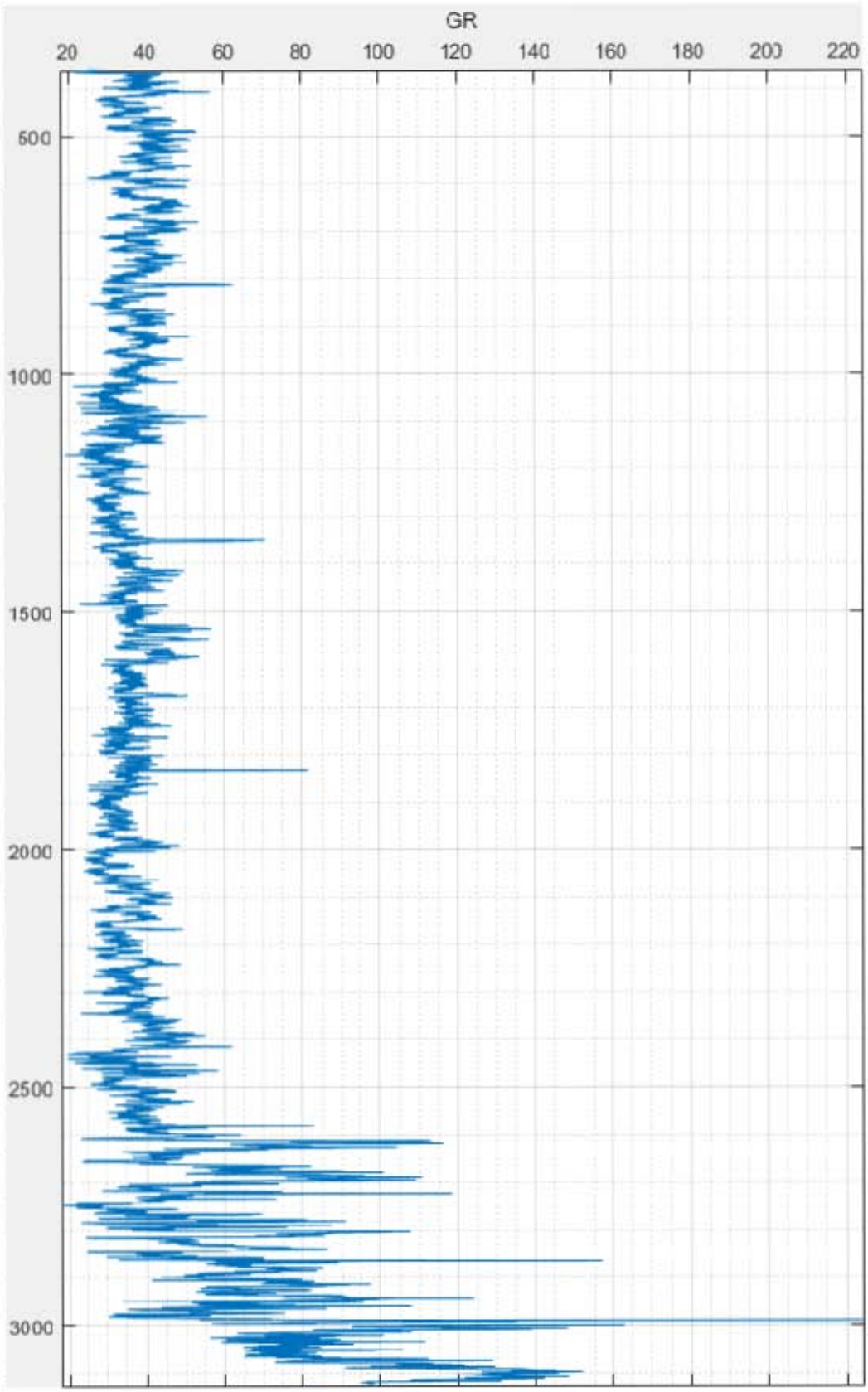


Figure (4.8) GR from Kingdom

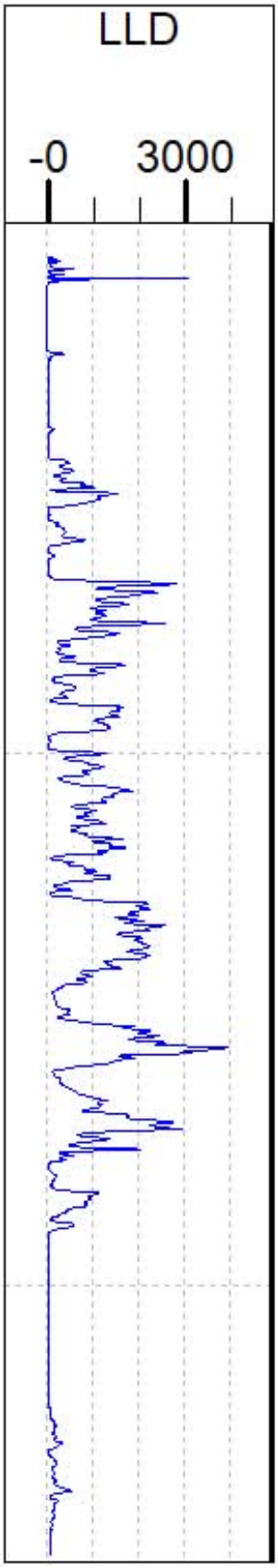


Figure (4.9) LLD from Kingdom

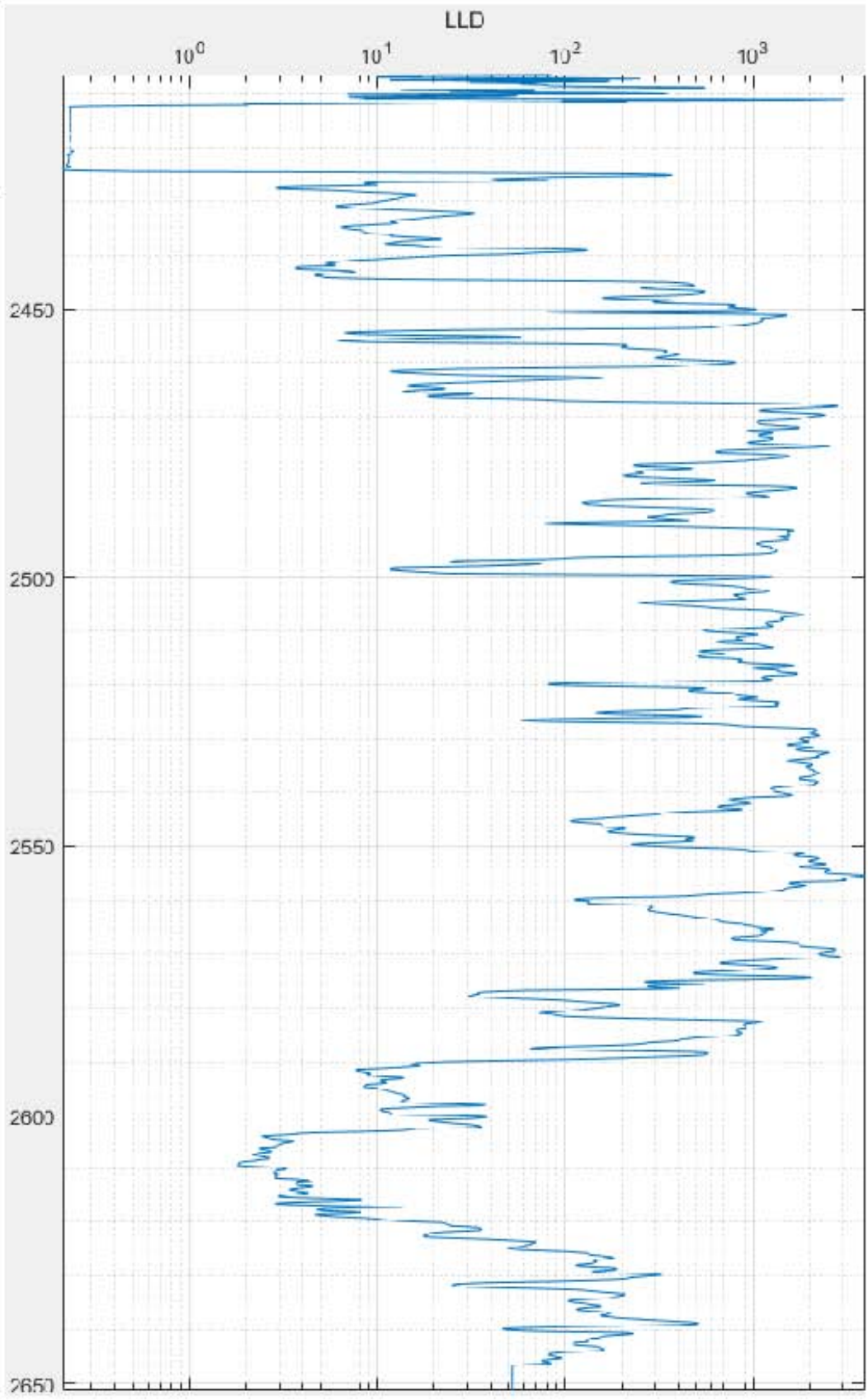


Figure (4.10) LLD from WellGUI.m

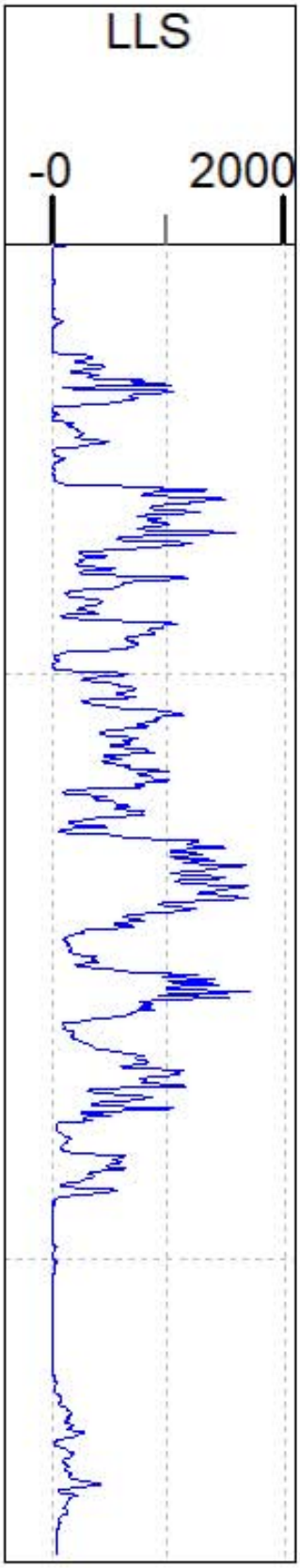


Figure (4.11) LLS from Kingdom

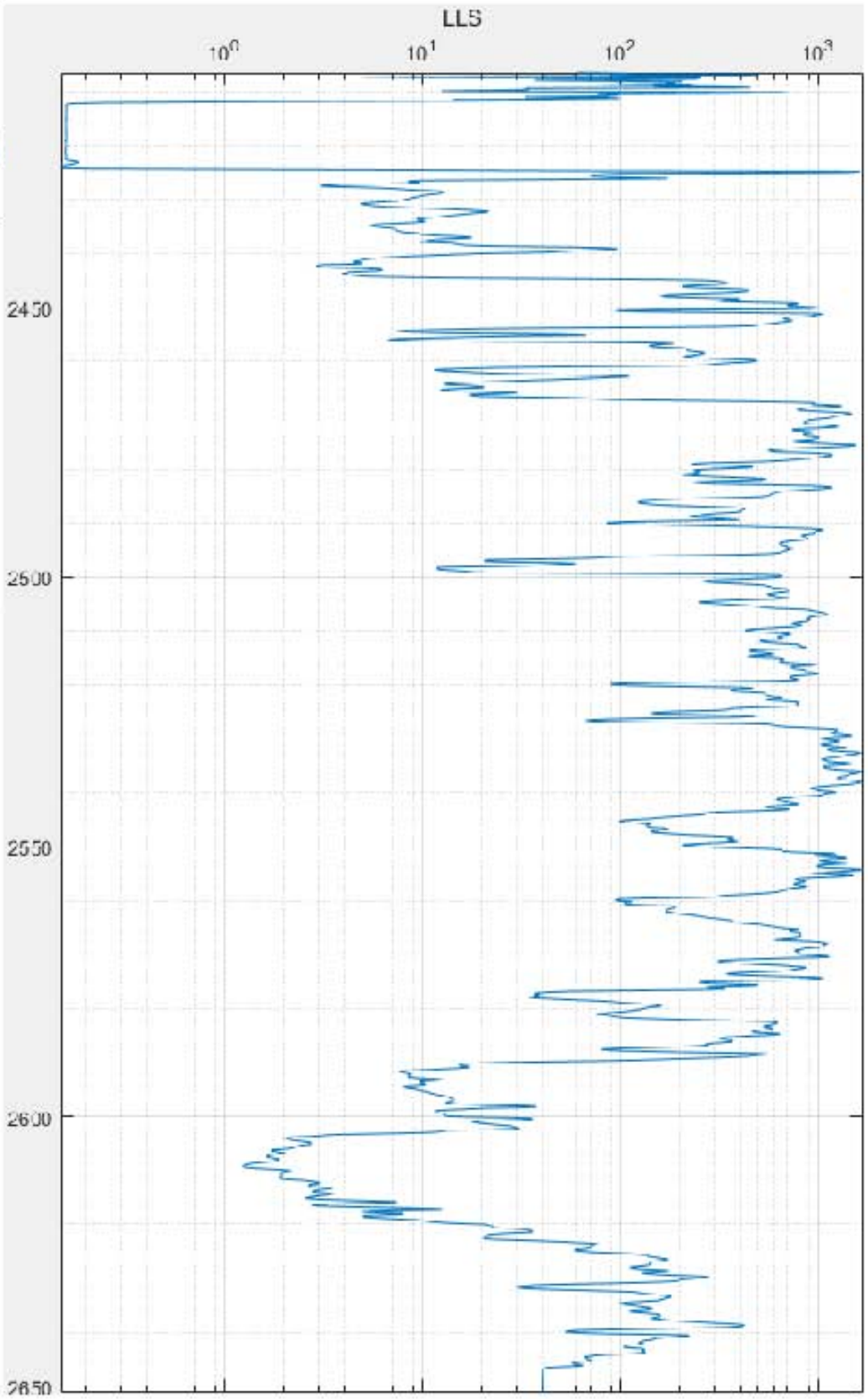


Figure (4.12) LLS from WellGUI.m

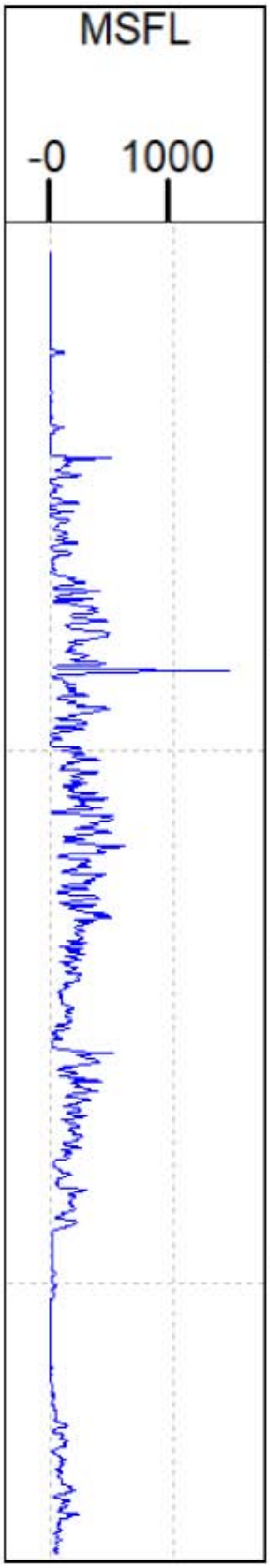


Figure (4.13) MSFL from Kingdom

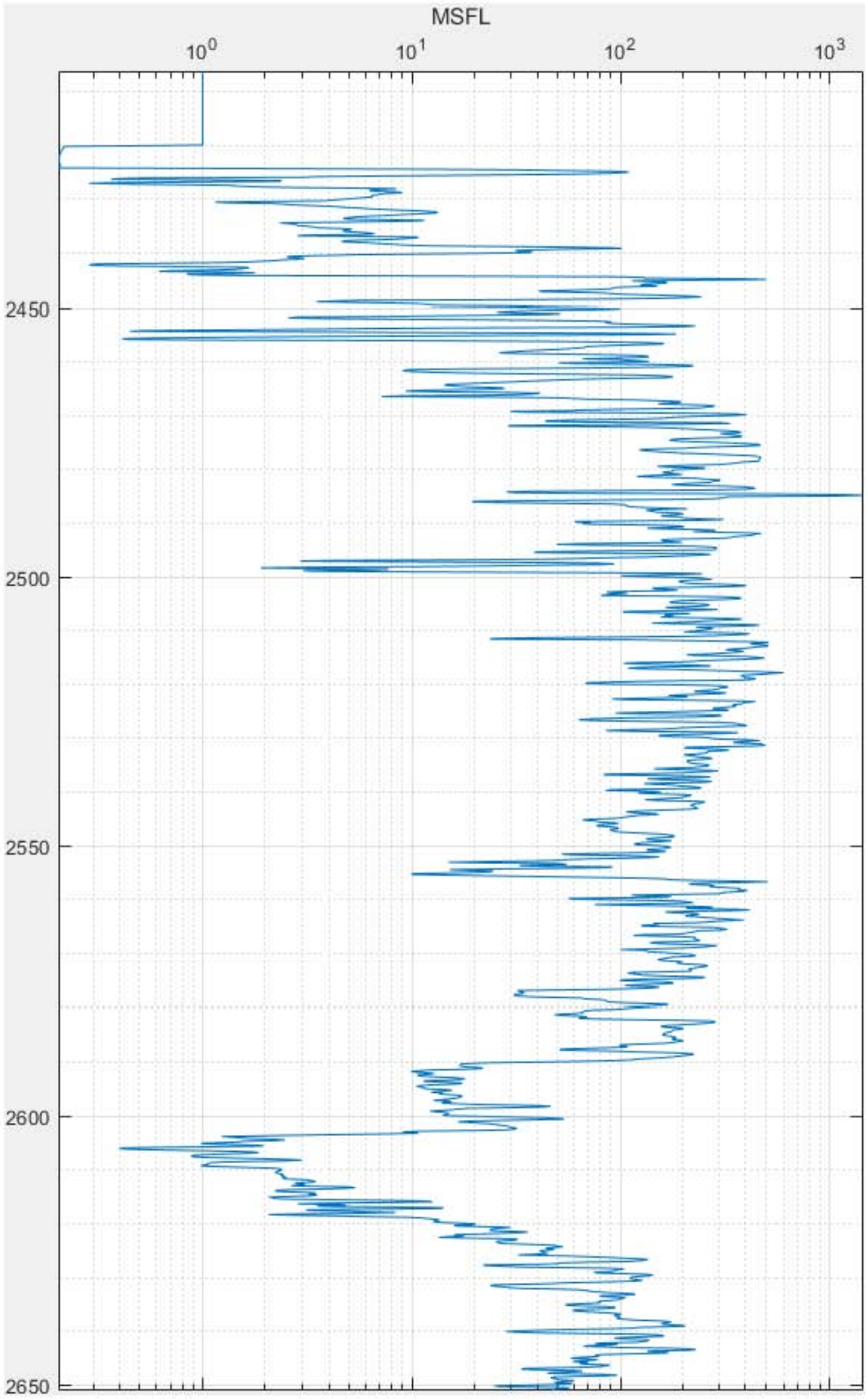


Figure (4.14) MSFL from WellGUI.m

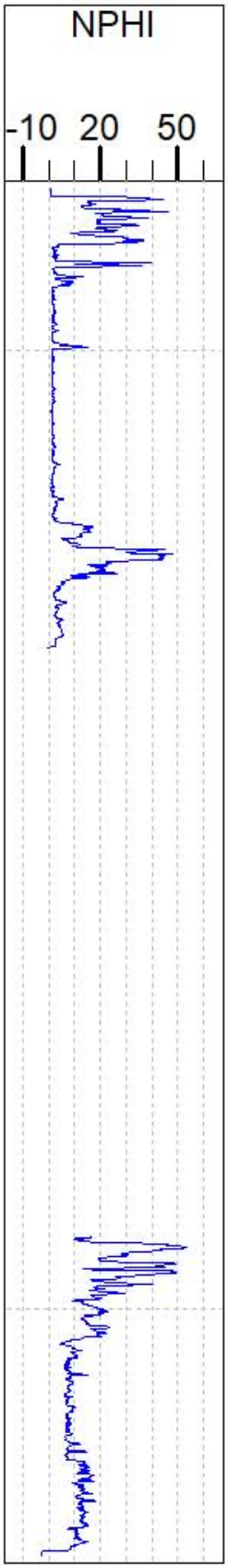


Figure (4.15)NPHI from Kingdom

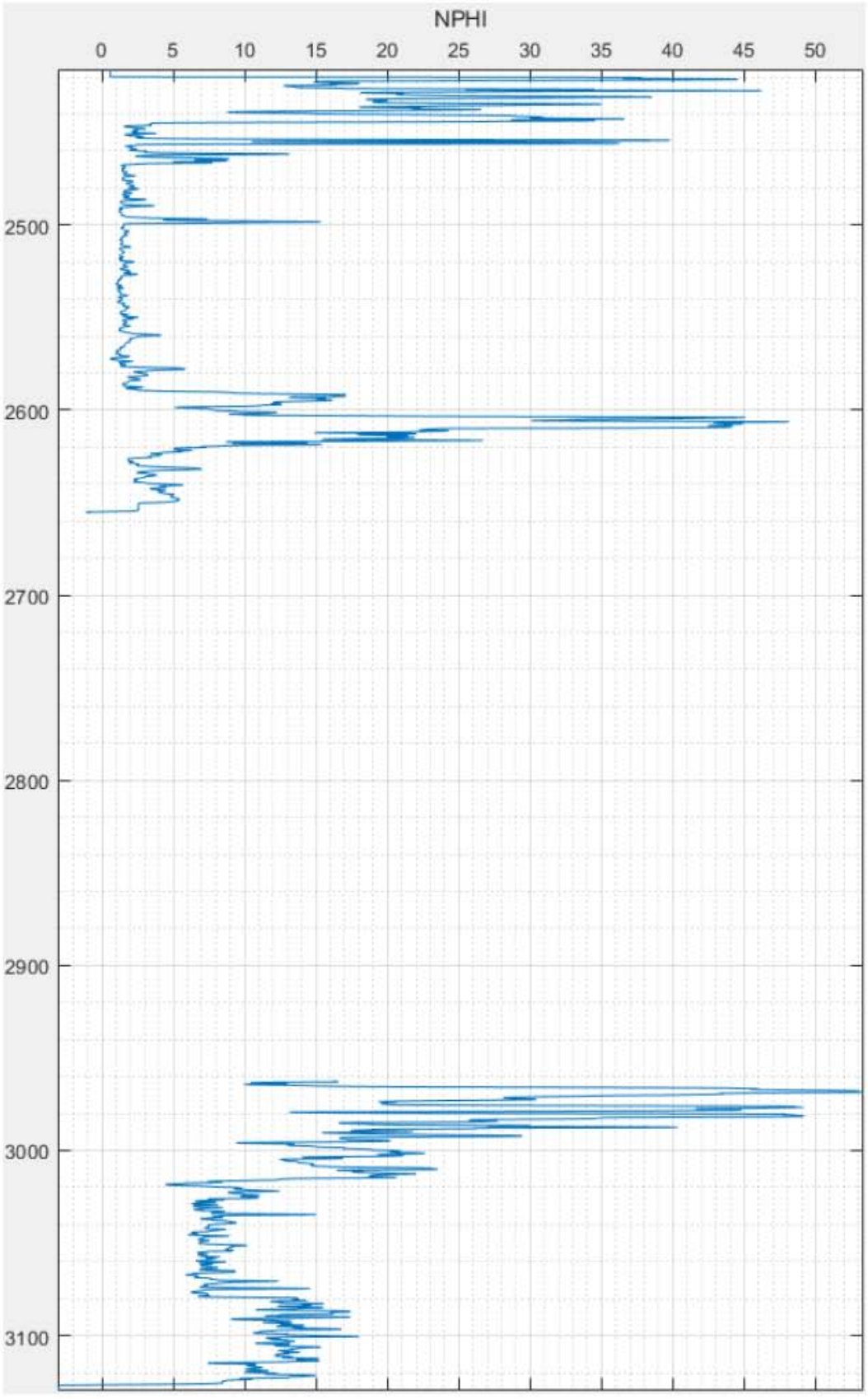


Figure (4.16) NPHI from WellGUI.m

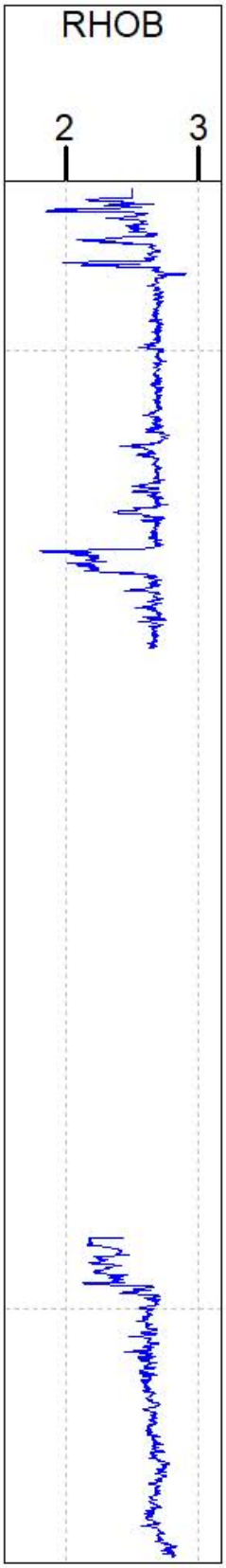


Figure (4.17) RHOB from Kingdom

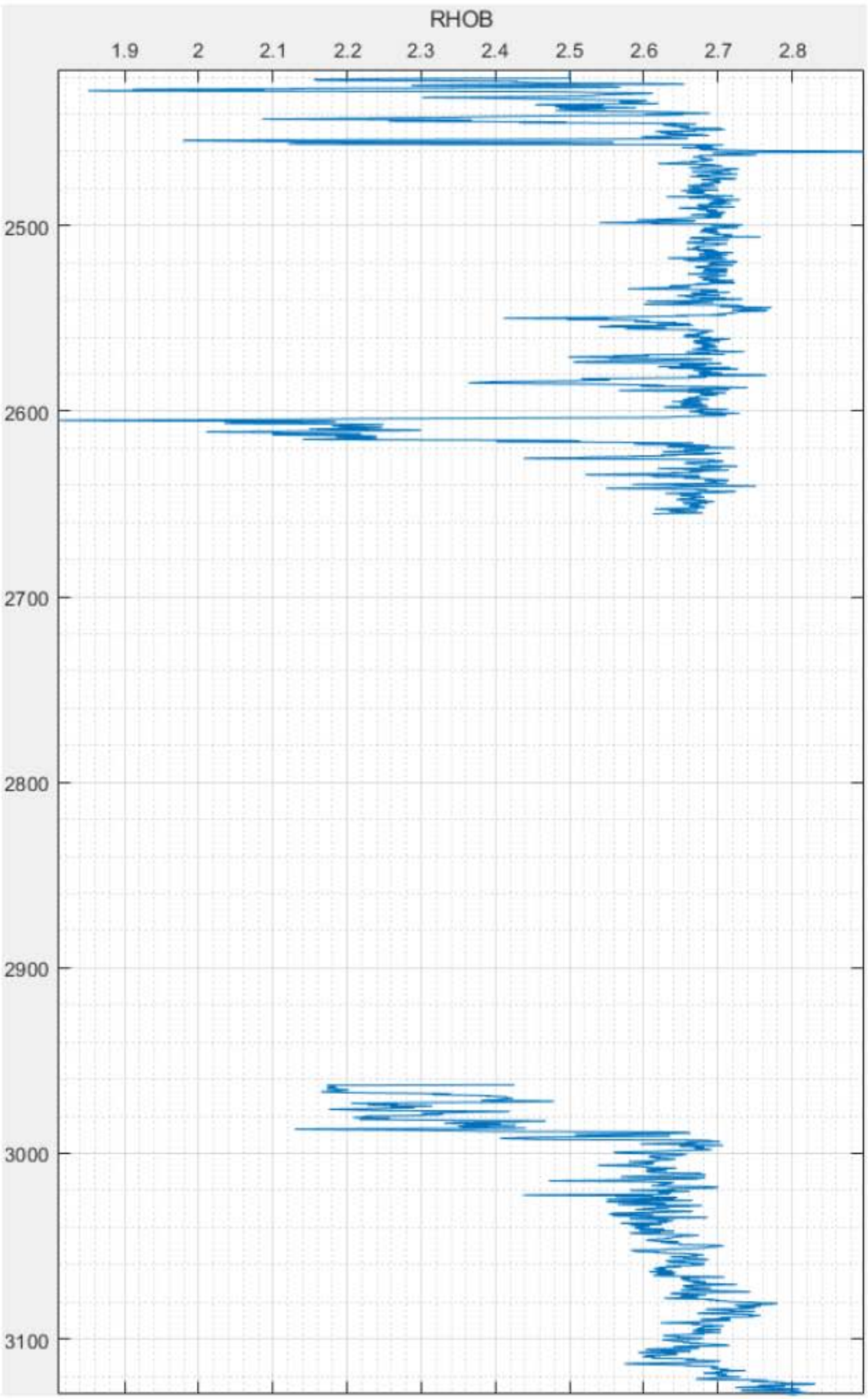


Figure (4.18) RHOB from WellGUI.m

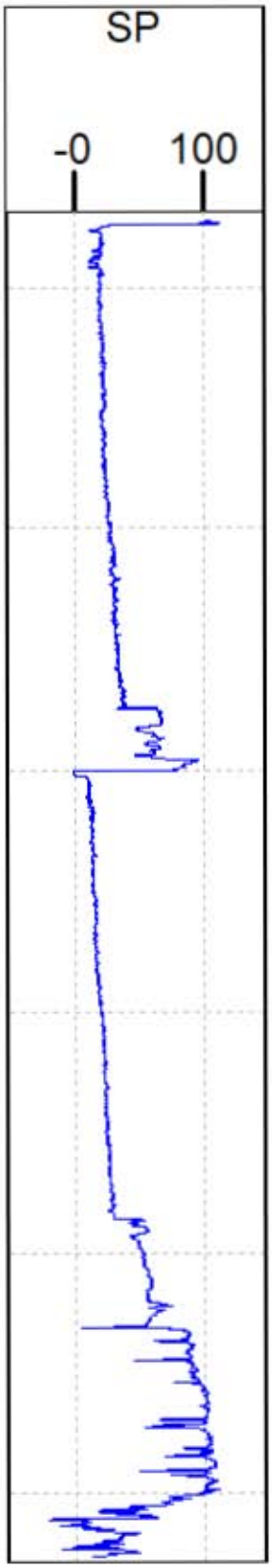


Figure (4.19)

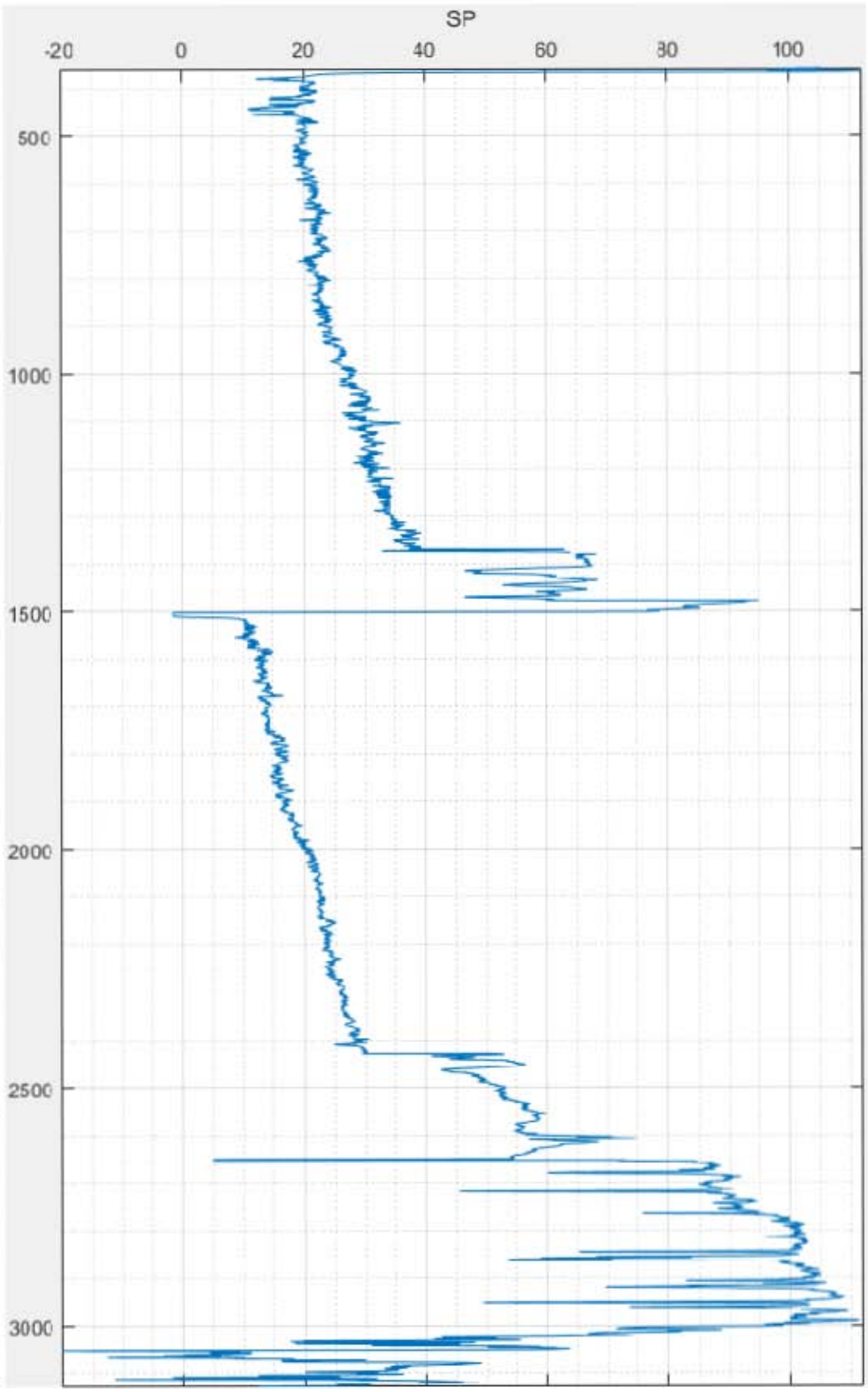


Figure (4.20)

4.3 Comparison of Calculated logs.

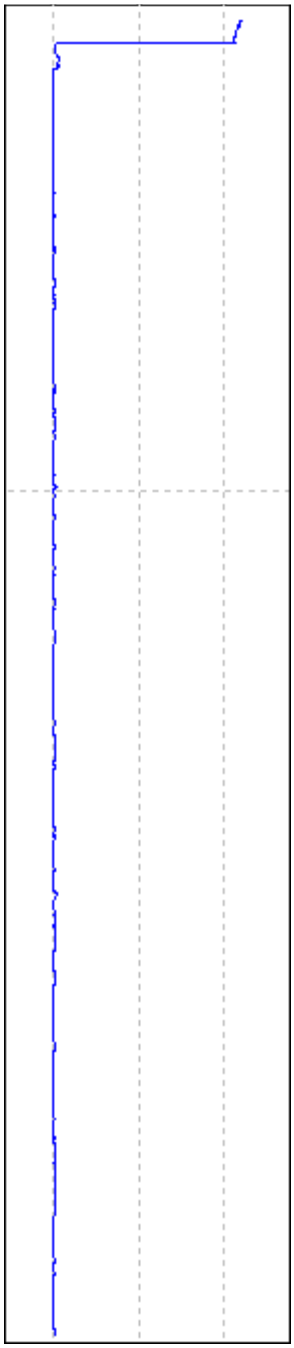


Figure (4.21) Sw from Kingdom

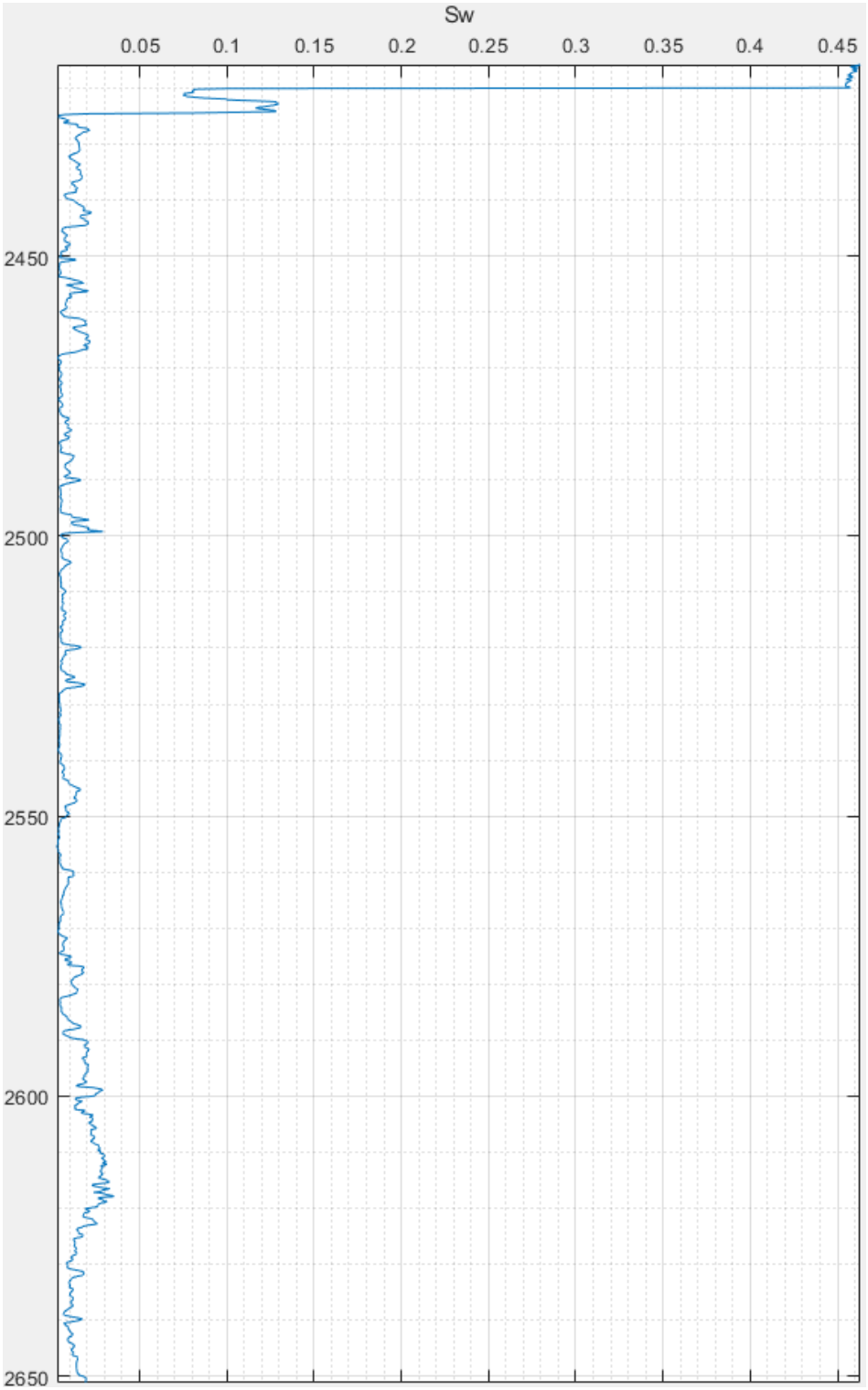


Figure (4.22) Sw from self-made software

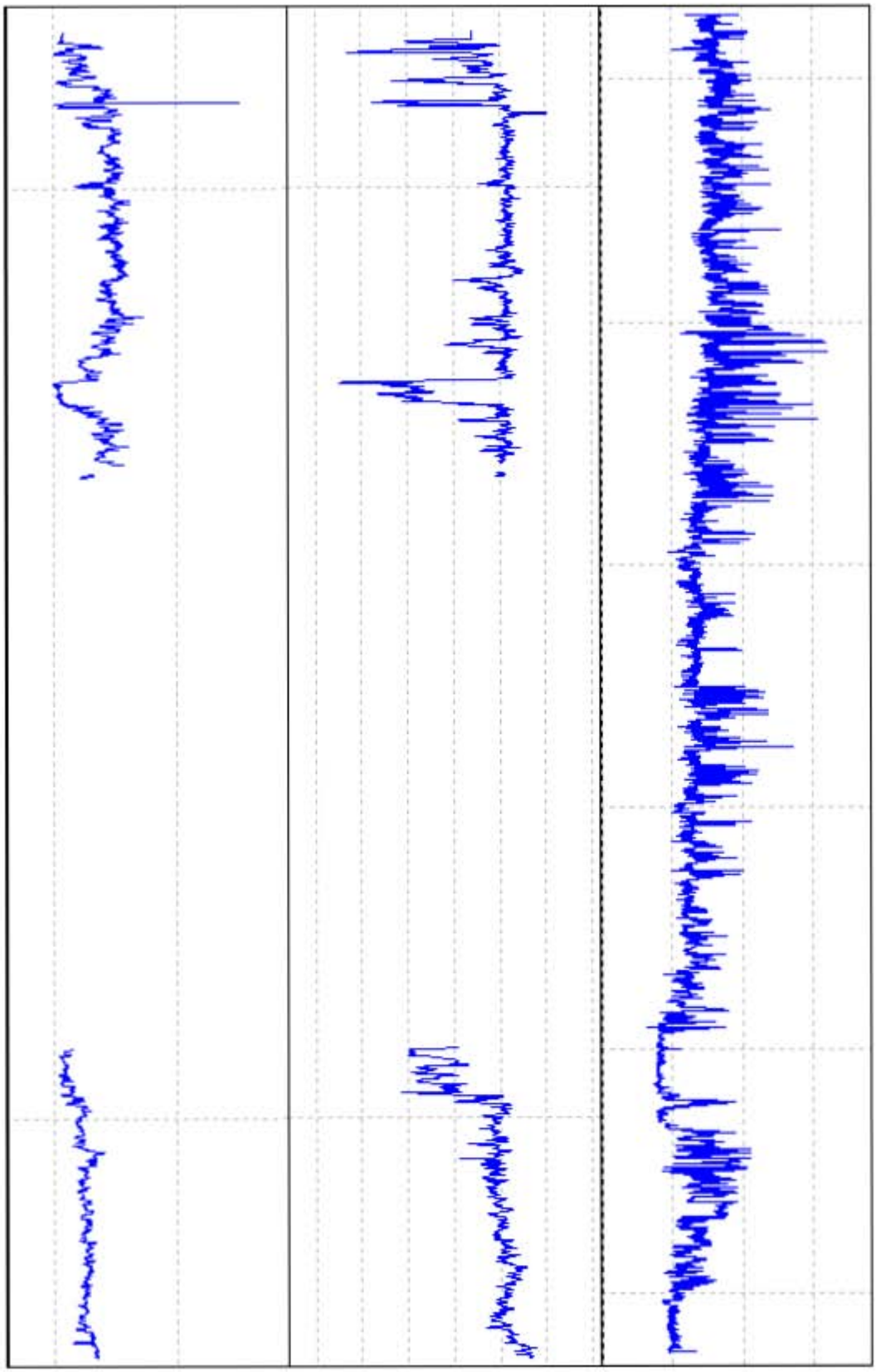


Figure (4.23) left to right- Brittleness Index, Young's Modulus and Poisson's Ratio from IHS Kingdom

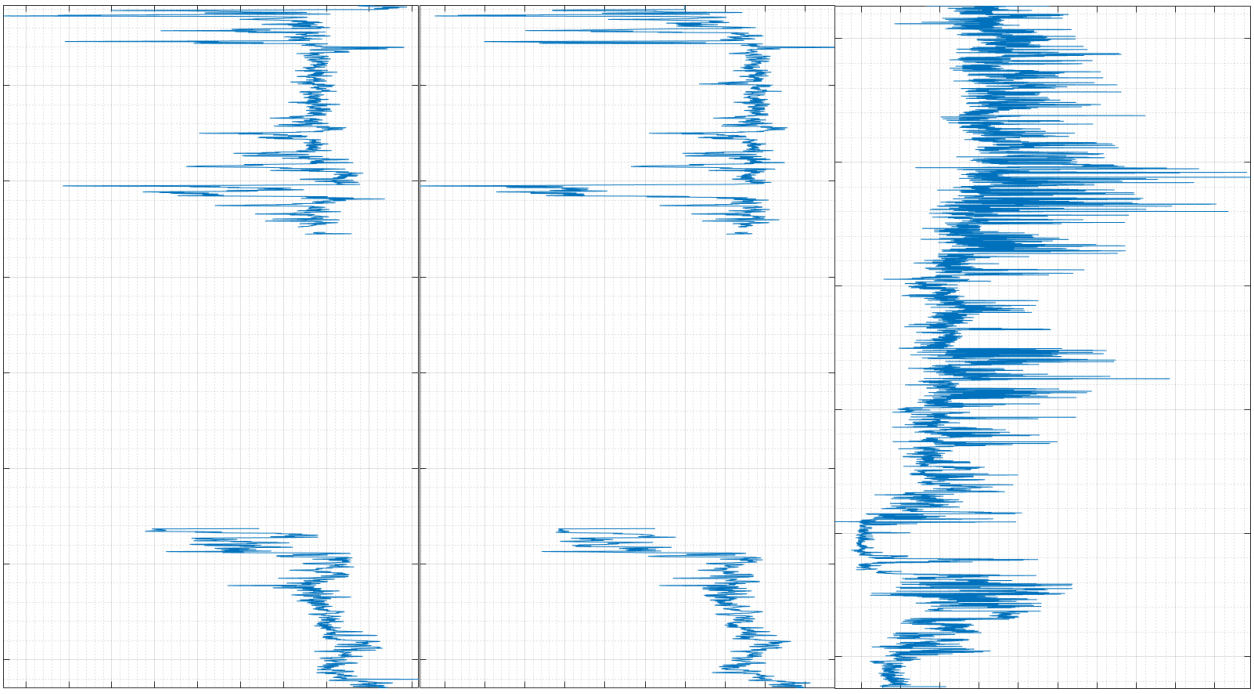
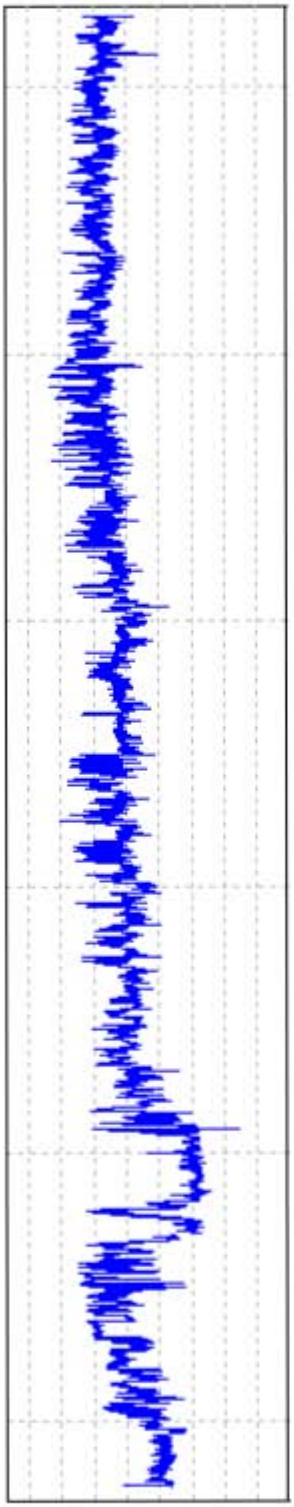


Figure (4.24) left to right- Brittleness Index, Young's Modulus and Poisson's Ratio from IHS Kingdom

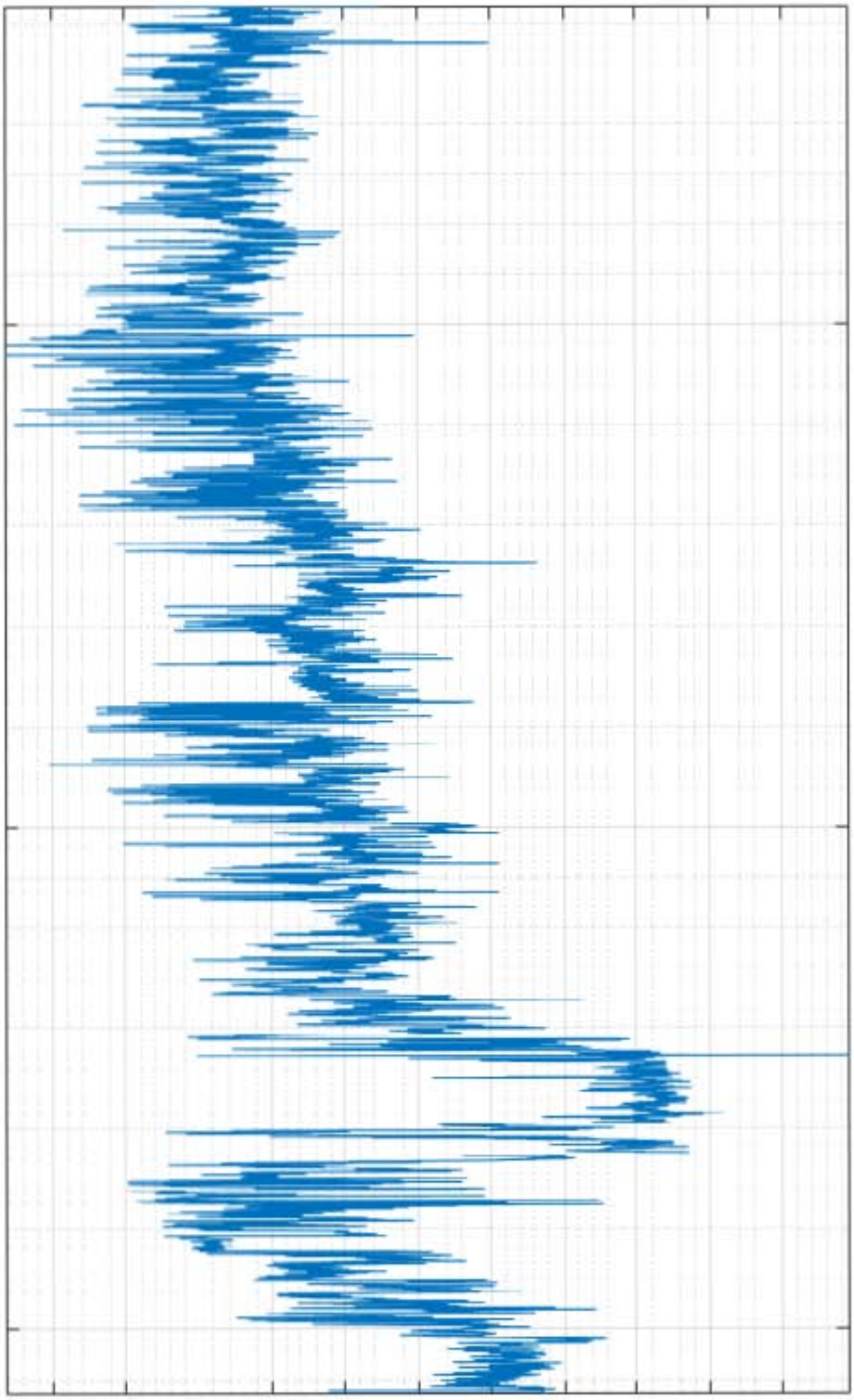
Figure 4.21 and Figure 4.22 show the Sw logs calculated using the archie equation in the IHS Kingdom software and the WellGUI.m respectively. The difference in the graphs shape is mainly due to the difference in scaling, the WellGUI.m shows a magnified x-axis which is the cause of the difference in shape.

Figure 4.23 and 4.24 show the brittleness index, young's modulus and Poisson's Ratio from the kingdom software and the WellGUI.m respectively.

Similarly figure 4.25 and 4.26 show the Vp log display from the kingdom and WellGUI.m respectively. The Vs calculations are not shown as since they are done from the Vp they are have the same general trend. The same is the reason for the AI log. If the trend of the Vp log is the same then the AI log will show the same trend aswell.



Figure(4.25) Vp as calculated from kingdom



Figure(4.26) Vp as calculated from WellGUI.m

Chapter 5:

Limitations and Conclusion

As shown in chapter 4 the program can run simplistic processes for well log data and the results are comparable to the IHS kingdom software results of the same processes. It however has several limitations and areas that can be highly improved upon if given sufficient time and knowledge about matlab programming.

5.1 Limitations

There are several limitations in the program mainly due to the constraints of time, knowledge of matlab coding and the criteria set at the start for the program to run a very user friendly and simplistic GUI for basic processing.

Firstly, and possibly most notably the log displays are always of the full log. The axes are set according to the 'axes tight' command which forces the x and y axes to be set according to their corresponding maximum and minimum values. Hence, the depth shown is always of the full log values. This means that user defined range of depth is not accepted and must alternatively be zoomed in to the desired depth range. For this the room in function has the ability to only zoom the vertical depth axes.

Secondly, and also very notably the program can only run the following logs and no other logs. Additionally, the log titles in the excel sheet can also only be names as shown and are case sensitive.

→Depth or DEPTH
→NPHI or PHIN
→CALS
→GR
→CALI
→SP
→Temp
→LLD or ResD
→DRHO
→RHOB
→MSFL or ResM
→PEF
→LLS or ResS
→DT

Thirdly, the program cannot display formation tops. This is due to my personal lack of coding knowledge as I did not know how to add a superficial line that would denote the formation top.

The cross-plot program is also limited to utilizing the entire log and cannot display only the selected depth. As such it can only show the general trend of the entire selected logs. Additionally, the cross-plot program cannot make polygons which would highlight the area in its

corresponding log. This is completely due to my lack of knowledge is coding. Once again this limitation is only due to personal lack of coding knowledge.

Another limitation for the code is that users can only use the limited number of provided formulae. E.g. if a user wishes to use the Steiber or clavier formulae for Volume of shale, they will be unable to do so. Similarly if user will be unable to apply a different formula for calculation of saturation of water. This can be fixed if simply given more time.

This GUI is also limited to only the use calculated Vs and cannot read or calculate the logs for shear wave transit time.

5.2 Possible Improvements

While this GUI has very significant room for improvement, it must be kept in mind that above all the purpose of creating this GUI is for a user friendly experience for simple petrophysical analysis and interpretation. Hence, to avoid complexity the number of possible improvements are reduced. Some of these improvements are discussed below.

A very important improvement that can be made in all three GUIs is the inclusion of displaying values from only a selected range of depth. This is doubly important for the cross-plots GUI as it will significantly improve interpretation capabilities. The cross-plots GUI can also be improved by adding a color bar which would show a changing gradient of color in the scatter plot according to a selected third log.

Another possible improvement can be made in the loading of data. Currently the program is only able to load log data if it is in the '.xlsx' Microsoft excel format. So an improvement can be made such that the log data can be directly loaded from a '.las' file.

As stated above, the program has many more areas for improvement and the above mentioned are only a handful that can be made.

5.3 Conclusion

In a nutshell, the thesis shows that the program developed with constrained knowledge of matlab programming and limited time is able to competently show accurate, if constrained, results of well log petrophysical analysis which was proven via comparison with the IHS Kingdom software's results of the same logs utilizing similar formulas. It should be noted that my focus in the programs development was a user friendly program that can perform basic petrophysical analysis for beginners in the field of geophysics and is by no means a replacement for professional software. It does however show that given enough programming knowledge it is very possible for geophysicists to compile blocks of codes for simple processes themselves which will provide competent and accurate results.

This program was made to be used with any well however it is also possible to obtain more accurate results if the program is hard coded to provide results for a specific wells specific area of interest as well.

References

✦ Castagna, J.P., Batzle, M.L. and Eastwood, R.L. (1985). Relationships between compressional wave and shear-wave velocities in clastic silicate rocks. Geophysics., 50, 571-581.

✦ Castagna, J.P., Batzle, M.L., Kan, T.K., 1993. Rock physics-the link between rock properties and AVO response. In: Castagna, J.P., Backus, M.M. (Eds.), Offset e Dependent Reflectivity e Theory and Practice of AVO Analysis. Society of Exploration Geophysicists, Tulsa, pp. 135e171.

✦ Han, De-hua 1986 Effects of porosity and clay content on wave velocities in sandstones. 51(11):2093.

✦ Passey, Q., Creaney, S., Kulla, J., et al., 1990. A Practical Model for Organic Richness from Porosity and Resistivity Logs. *AAPG Bulletin*, 74(12): 1777-1794

✦ George Asquith, Charles Gibson (1982) Basic Well Log Analysis for Geologists. AAPG.

✦ Telford, W.M., Geldart, L.P., Sheriff, R.E., and Keys, D.A., 1999, Applied Geophysics, Cambridge University Press, London

✦ Asquith, G. B., Krygowski, D., & Gibson, C. R. (2004). Basic well log analysis (Vol.16). Tulsa, OK: American association of petroleum geologists

✦ Passey, Q., Creaney, S., Kulla, J., et al., 1990. A Practical Model for Organic Richness from Porosity and Resistivity Logs. AAPG Bulletin, 74(12): 1777-1794

✦ Aadil, N., Tayyab, M., & Naji, A. (2014). Source Rock Evaluation with Interpretation of Wireline Logs: A Case Study of Lower Indus Basin, Pakistan. Nucleus, 51(1), 139-145.

✦ Han, De-hua 1986 Effects of porosity and clay content on wave velocities in sandstones. 51(11):2093.