Romanian Journal of Petroleum & Gas Technology
Vol. VI (LXXVII) • No. 1/2025

BASIC WELL LOG INTERPRETATION IN PYTHON PROGRAMMING LANGUAGE

Mihai Iorgandopol [1]

Patricia Tita [1]

[1] OMV Petrom, Bucharest, Romania
email (corresponding author): m.iorgandopol@gmail.com; patricia.tita@proton.me

DOI: 10.51865/JPGT.2025.01.20

ABSTRACT

Python has rapidly become the programming language of petroleum geoscientists during the last decade. It is used widely in development, research and scientific activities and is included in many commercial software as an option for additional scripting purposes.

Geophysical well log data interpretation is one of the most useful and important tools available to a petroleum geologist [12]. The results of log analysis and interpretation play an important role in the decision making in oil and gas activities throughout the exploration, development and production stages.

The standard log interpretation workflow for petrophysical analysis of the wells includes: data loading, QC and editing (1); borehole environmental corrections (2); calculation of shale volume (3); porosity calculation (4); water saturation calculation (5); identification of lithology type (6); estimation of net and pay thickness (7).

In this tutorial it is shown how to use Python programming code to carry out the standard well log interpretation steps. Third-party libraries have been used in the tutorial: Matplotlib was used in the graphical parts of the scripts; Pandas library was used as a data analysis and manipulation tool. In addition, the following specifically designed Python libraries were used for log data reading and calculations: Lasio, a Python package to read and write Log Ascii Standard (LAS) files and petrophysics, which is a library containing a petrophysical collection of formulas written as Python functions.

Keywords: Python, basic well log interpretation, geophysics, petrophysics, third-party libraries

INTRODUCTION

Well Log Interpretation can be performed using many commercial software, but unfortunately no free software with a GUI (graphical user interface) is currently available in the geoscientists community. However, the necessity of interpreting the well log data for small projects, research or scientific purposes can be achieved using programming scripts or spreadsheet calculations software. In the past, petrophysicists have used programming languages like Fortan, C, Matlab (widely used in engineering, science, and mathematics), Octave (a free and open-source alternative to Matlab) or Microsoft Excel spreadsheets in order to perform simple or advanced log calculations.

Gaining popularity in many domains, Python is currently the best option for writing programming codes for well log analysis due to the simplicity of the language commands and the availability of several third-party Python libraries which can be used for visualization and data analysis. These additional libraries are discussed in the first part of the article.

The second part of this paper presents a standard well log interpretation workflow, and using a case example, tries to demonstrate that each step in the workflow can be efficiently solved with programming tools. While the full code is not included in the paper, a web link to its location is available in the references section. The results and observations are formulated in the *Conclusions* section.

PYTHON AND THIRD-PARTY LIBRARIES

In this chapter the tools used in the tutorial are discussed.

Python is a programming language which was designed by Guido van Rossum, and first became available in 1991. It is defined in the Help Section of the Python website as: "an interpreted, interactive, object-oriented programming language. It supports multiple programming paradigms beyond object-oriented programming, such as procedural and functional programming". [1]

NumPy, developed in 2005, is a third-party Python library designed to enhance support for "large, multi-dimensional arrays and matrices, along with an extensive collection of high-level mathematical functions for operating on these arrays". [2]

Pandas, built on top of NumPy, is a third-party Python library recognized as "a fast, powerful, flexible and easy to use open-source data analysis and manipulation tool". The name derives from "panel data," and its initial release occurred in 2008. [3]

Matplotlib ("Matlab-like plotting library") is a plotting library for Python that emulates MATLAB, offering support for both 2D and 3D plots since its introduction in 2003. [4]

Lasio is a specialized Python package for reading and writing CWLS Log ASCII Standard files, primarily "used for borehole data, including geophysical, geological, and petrophysical logs."[5]

Petrophysics is another dedicated Python package that incorporates functions for petrophysical calculations, such as shale volume, lithology, porosity, permeability, water saturation, rock physics, etc. [6]

BASIC WELL LOG INTERPRETATION WORKFLOW

A simplified workflow schema in order to interpret well logs data includes the following key steps [7], [18], [19]:

(1) Data Import, Editing & Visualization

(2) Borehole Environmental Corrections

(3) Shale Volume Calculation

(4) Porosity & Lithology Identification

(5) Water Saturation

(6) Net & Pay Thickness Computation. Interpretation of Results.

(1) Data Import, Editing and Visualization

The most widely recognized ASCII format for well log data is the LAS (Log Ascii Standard by CWLS). This format can be efficiently read and imported into Python using standard text processing commands of the programming language or, more rapidly, with the *lasio* package [5], which facilitates the reading and loading of data into a Pandas DataFrame using simple commands, see below:

```
import lasio
las = lasio.read ('test_well.las')
data = las.df()
data.head()
```

DEPTH	SP	GR	CALI	BITSIZE	LL8	ILM	ILD	RHOB	NPHI	DT
101.0	NaN	NaN	NaN	12.25	NaN	NaN	NaN	NaN	NaN	NaN
101.5	NaN	NaN	NaN	12.25	NaN	NaN	NaN	NaN	NaN	NaN

Log editing (e.g., shifting, splicing, multiple log depth referencing) is a difficult process due to the lack of dedicated GUI (graphical user interface) tools, and extra coding is necessary to accomplish this task.

Visualization of data can be achieved with the Matplotlib package, which makes it possible to create log tracks, plot curves, and display track header details e.g. curve name, curve units, values scale. Furthermore, various options for customization, including the modification of colors and text settings, are available. Log plotting templates (e.g. for resistivity, triple combo and interpretation) as well as the top and bottom of zones of interest can be stored in predefined functions, allowing for easy access whenever needed (Figure 1).

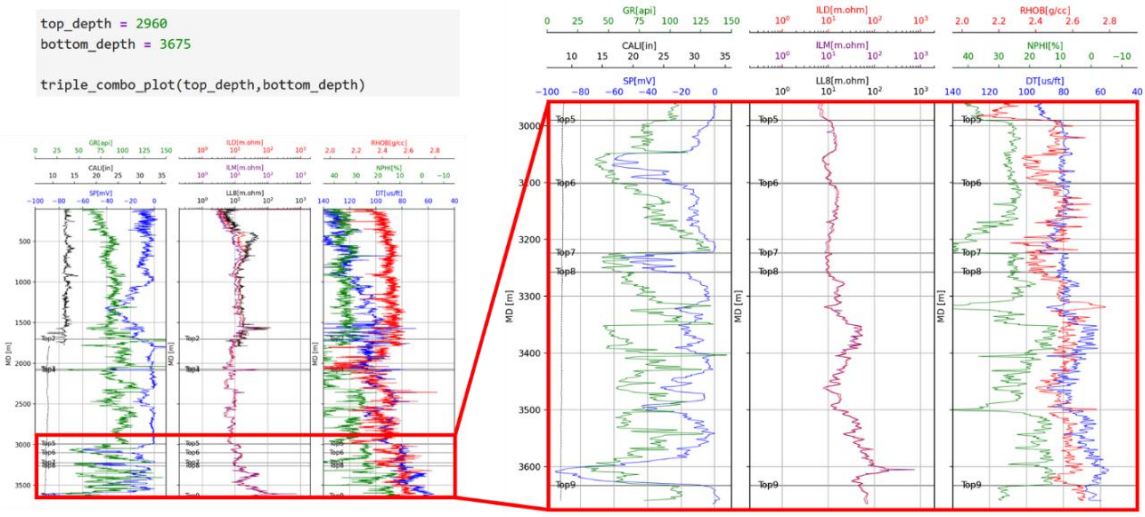

Figure 1. *Visualization of triple combo log composite with Matplotlib*

(2) Borehole Environmental Corrections

The logging companies provide table & chart books for borehole environmental corrections. These are mostly related to borehole size diameter, caverns, drilling mud types and are specific to each logging tool contractor. Commercial softwares have modules implemented to apply these corrections, and the logging acquisition companies nowadays deliver the corrected logs so that they can be directly used and interpreted.

(3) Shale Volume Calculation

All methods of calculating the shale volume, using gamma ray log with or without Clavier [14], Stieber [25], Larionov [17] corrections, spontaneous potential, resistivity, and combination of neutron-density can be defined with Python functions.

In order to choose the best parameters for the equations or to decide the method for calculation of final VSH (Volume of Shale) when multiple logs are available as input, histograms and overlay plotting of VSH from different methods are recommended to be used. (Figure 2).

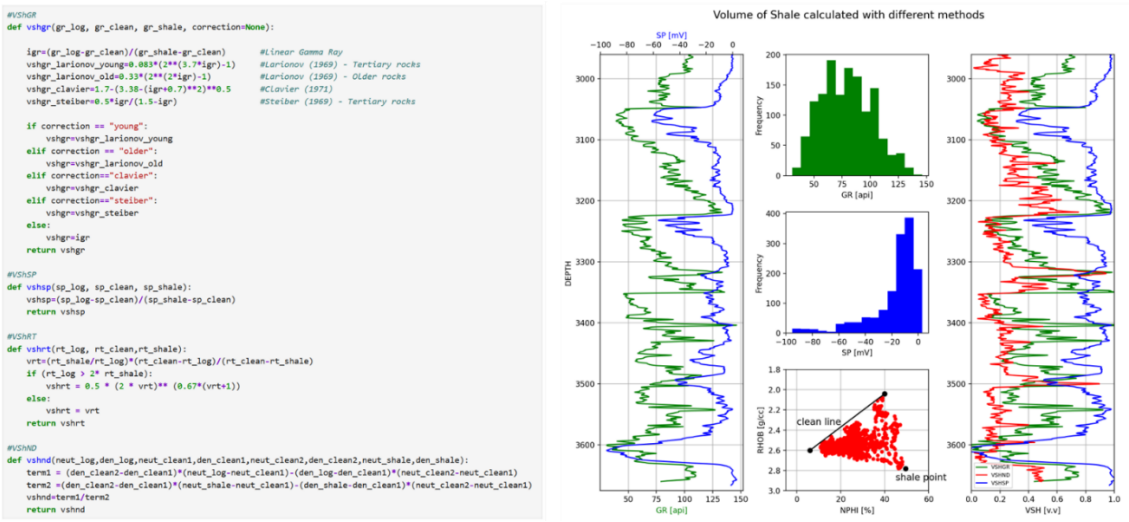

***Figure 2**. Shale volume calculation by different methods*

(4) Porosity & Lithology Identification

Porosity can be calculated using single & dual methods derived from neutron, sonic (Willie [26], Raymer, [22]) and density logs. The porosity equations can be defined using functions and they can also be used to implement the shale correction [11], [12]. How these functions are defined depends on the interpreter's scripting choice. Below are several examples of function definitions for porosity equations, as illustrated in Figure 3.

***Figure 3**. Porosity methods, Python functions and Matplotlib plot results*

Lithology & Porosity can be determined using:

a. Analytical Solution

For lithologies deposited in a siliclastic depositional environment, where porosity and clay volume are known, the volume of quartz can be calculated using a simple equation: $V_q = 1 - \phi - V_{cl.}$

When dealing with lithologies deposited in a more complex depositional environment, Donovan (1994) proposed solutions to calculate mineral volumes. [15],[16]. A simple approach is to determine mineral volumes V by solving a system of equations: $C \times V = L \Rightarrow V = C^{-1} \times L$, where C is mineral matrix values and L is the log data. The Numpy library offers solutions for inverse and multiplication matrixes, as shown below.

$$C \times V = L \qquad V = C^{-1} \times L$$

$$\begin{bmatrix} n_q & n_{ca} & n_d & n_{fl} \\ \Delta t_q & \Delta t_{ca} & \Delta t_d & \Delta t_{fl} \\ \rho_q & \rho_{ca} & \rho_d & \rho_{fl} \\ 1 & 1 & 1 & 1 \end{bmatrix} \times \begin{bmatrix} V_q \\ V_{ca} \\ V_d \\ \varphi \end{bmatrix} = \begin{bmatrix} n_{log} \\ \Delta t_{log} \\ \rho_{log} \\ 1 \end{bmatrix} \Rightarrow \begin{bmatrix} V_q \\ V_{ca} \\ V_d \\ \varphi \end{bmatrix} = \begin{bmatrix} n_q & n_{ca} & n_d & n_{fl} \\ \Delta t_q & \Delta t_{ca} & \Delta t_d & \Delta t_{fl} \\ \rho_q & \rho_{ca} & \rho_d & \rho_{fl} \\ 1 & 1 & 1 & 1 \end{bmatrix}^{-1} \times \begin{bmatrix} n_{log} \\ \Delta t_{log} \\ \rho_{log} \\ 1 \end{bmatrix}$$

```
import numpy as np
C = np.array ([[-0.028, 0, 0.01, 1],[56, 49, 44, 189],[2.64, 2.71, 2.85, 1.05],[1, 1, 1, 1]])      #quartz, calcite, dolomite and fluid mineral properties
L = np.array (['NPHI_log', 'DT_log', 'DEN_log', 'Porosity'])                                        #replace matrix values with log data
InvC = np.linalg.inv(C)                                                                             #inverse of mineral properties matrix
V = InvC @ L1                                                                                       #calculated mineral volumes matrix
```

b. Graphical crossploting allows solutions for both porosity and lithology to be estimated at the same time [24]. The main useful crossplots are formed by:

- Dual combination of Neutron, Sonic & Density Logs [23],[24] (Figure 4)
- Triple combination of Neutron Sonic & Density Logs: *MN* Lithology Plot and Matrix Identification Plots (ρ_{maa} *vs* Δ_{maa} & U_{maa} *vs* ρ_{maa}) [13],[24]

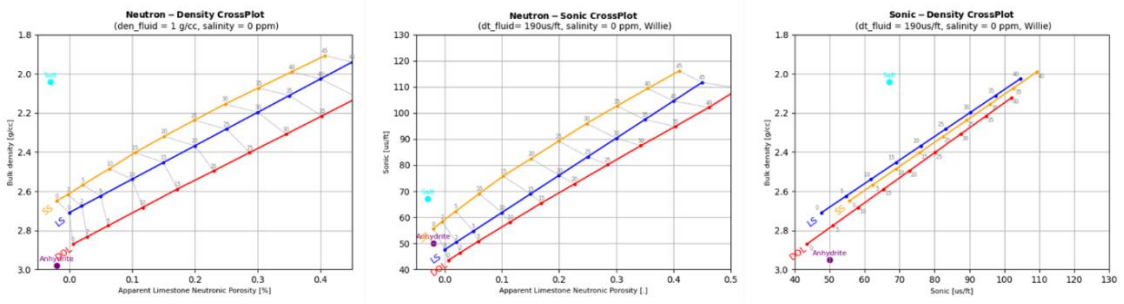

Figure 4. *Determination of porosity/lithology using graphical solution (dual logs combinations)*

(5) Water Saturation

In this tutorial, water saturation calculated using Archie [9] equation can be defined in a Python function using water resistivity, true resistivity and porosity as inputs and defining the tortuosity factor (a), cementation exponent (m) and saturation exponent (n) using function arguments. In cases where Archie's parameters are not known, a Picket [20],[21] plot can be created from the Matplotlib library and used in order to graphically determine a, m, n *coefficients.* (Figure 5). For reservoirs where Archie [9],[10] equations have some limitations (e.g. non-clean formations) it is recommended to import and use an appropriate water saturation equation that is already defined in the *petrophysics* package.

(6) Net & Pay Thickness. Interpretation of Results

The results of petrophysical interpretation can be plotted using a pre-defined Matplotlib template (Figure 6) which includes the following tracks: (1) GR, SP, CALI, (2) resistivities, (3) sonic, density & neutron logs, (4) water saturation using Archie Swa, (5) porosity and bulk volume of water, (6) matrix, shale volume, and porosity in a color filled track for fast visualization. The net and pay thickness are displayed in tracks (7) and (8) and they can be evaluated by defining cut-off value limits for shale volume, effective porosity and water saturation. (Figure 5)

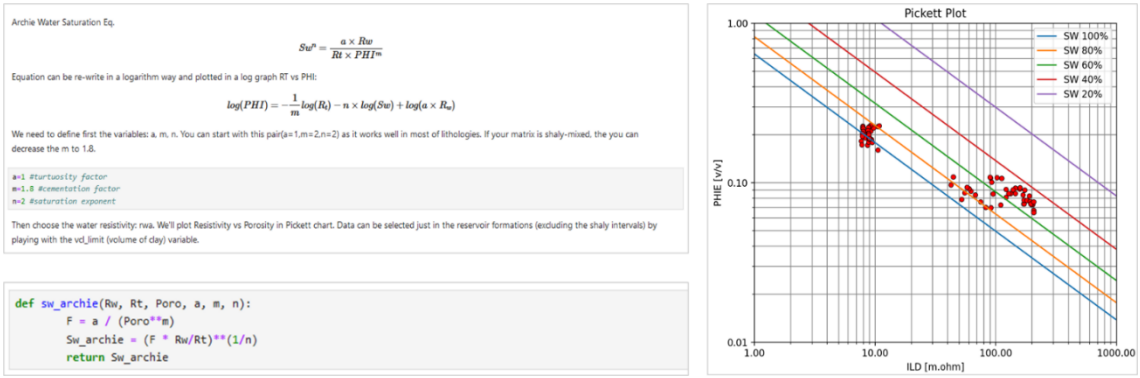

Figure 5. *Archie equation function and Picket plot example*

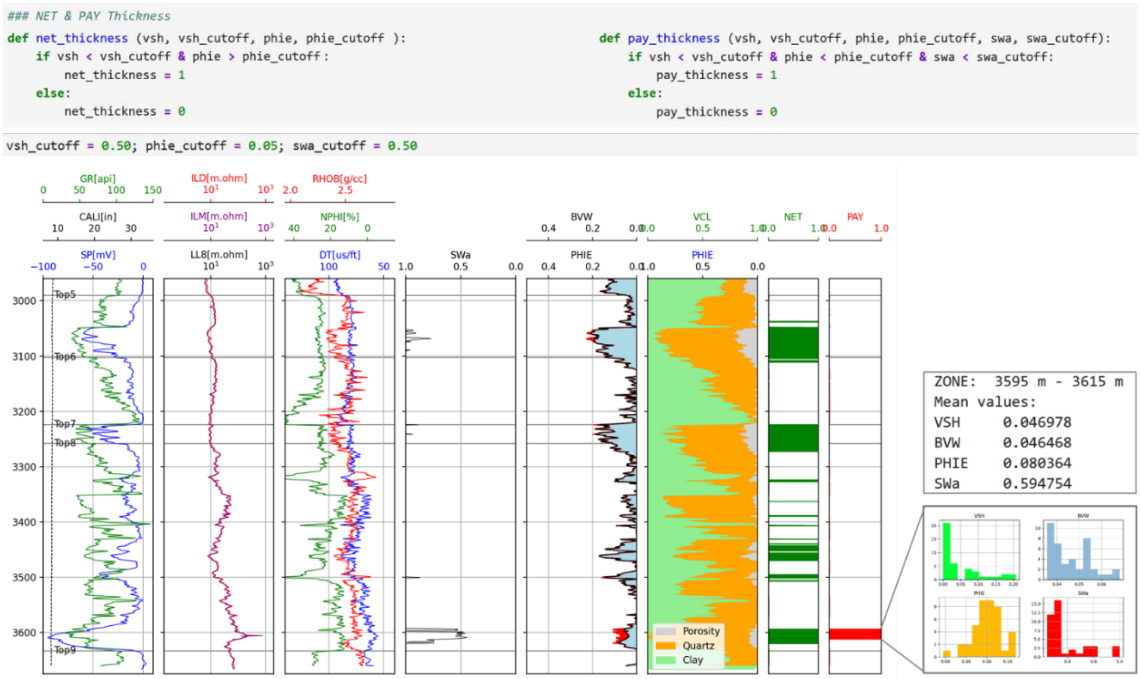

Figure 6*. Interpretation Plot results*

CONCLUSIONS

This tutorial proves that programming languages can be a powerful tool for petrophysicists in order to interpret geophysical well logs data and create detailed petrophysical reports.

The case study presented in this paper shows that well logs interpretation workflow steps such as calculation of clay volume, porosity, mineral volumes, water and hydrocarbon saturations, net & pay thickness, as well as the presentation of results through integrated plots, charts or statistical analyses can be easily accomplished using Python by log analysts with minimal programming skills.

However, certain limitations in the interpretation workflow are associated with log editing processes and borehole environmental corrections. The pre-conditioning of log data is difficult in the absence of a graphical user interface because it requires extra dedicated coding, which is time-consuming in Python.

The complete programming code is not included in the article, but it is posted on a software code repository website and the link [8] to this resource could be found in the *References* section. The tutorial can serve as a foundation for more advanced interpretation workflows in the future.

In conclusion, the Python programming language, along with various third-party and specialized libraries, offers petrophysicists the flexibility and freedom to manipulate and interpret log data using their preferred analytical methods.

REFERENCES

[1] https://www.python.org/ - Official Python Website

[2] https://numpy.org/ - Official Numpy Website

[3] https://pandas.pydata.org/ - Official Pandas Website

[4] https://matplotlib.org/ - Official Matplotlib Website

[5] https://lasio.readthedocs.io/

[6] https://pypi.org/project/petrophysics/

[7] https://www.spec2000.net/ - Crain's Petrophysical Handbook

[8] https://github.com/petroGG/Basic-Well-Log-Interpretation

[9] Archie, G.E., 1942, The electrical resistivity log as an aid in determining some reservoir characteristics: Transactions of the AIME, 146, 01, 54–62. https://doi.org/10.2118/942054-G.

[10] Archie, G.E., 1952, Classification of reservoir rocks and petrophysical considerations: AAPG Bull., v. 36, no. 2, p. 278-298

[11] Asquith, G., 1982, Basic Well Log Analysis for Geologists, AAPG Memo, AAPG, Tulsa, Oklahoma, 216 pp.

[12] Asquith, G., Krygowski, D., 2004, Basic Well Log Analysis, AAPG Methods in Exploration Series, No. 16, Second Edition, American Association of Petroleum Geologists (AAPG), https://doi.org/10.1306/Mth16823.

[13] Clavier, C., Rust, D.H., 1976, MID plot: a new lithology technique: The Log Analyst, v. 17, p. 16-24.

[14] Clavier, C, Hoyle, W, Meunier, D, 1971, Quantitative interpretation of thermal neutron decay time logs: part I. Fundamentals and techniques. J Petrol Technol 23(6): 743–755. https ://doi.org/10.2118/26.

[15] Doveton, J.H., 1986, Log analysis of subsurface geology: Concepts and computer methods.

[16] Doveton, J.H., 1994, Geological Log Analysis Using Computer Methods: Computer Applications in Geology, No. 2: Tulsa, Oklahoma, The American Association of Petroleum Geologists, 169 p.

[17] Larionov, V., 1969, Borehole Radiometry: Nedra. The SPWLA Annual Logging Symposium Transactions, Vol. 10, 26.

[18] Mălureanu, I., 2007, Geofizică de sondă, Editura Universităţii Petrol-Gaze, Ploieşti.

[19] Negut, A., 1972, Geofizică de sondă, Vol. 2 - Interpretarea rezultatelor investigaţiilor geofizice de sondă: IPGG Bucureşti, 250 p.

[20] Pickett, G.R., 1968, A review of current techniques for determination of water saturation from logs: Jour. Pet. Tech., p. 1425-33.

[21] Pickett, G.R., 1973, Pattern recognition as a means of formation evaluation: The Log Analyst, vol. 14, no. 4, p. 3–11.

[22] Raymer, L., Hunt, E., Gardner, J.S., 1980, An Improved Sonic Transit Time-to-Porosity Transform. Proceedings of Society of Petrophysicists and Well-Log Analysts, Houston, 1-13.

[23] Schlumberger, 1996, Log Interpretation Charts.

[24] Schlumberger, 1991, Log Interpretation Principles/Applications. 3rd Printing Schlumberger Educational Services, Houston

[25] Thomas, E.C., Stieber, S.J., 1975, The distribution of shale in sandstones and its effect upon porosity. In: SPWLA 16th annual logging symposium. New Orleans, Louisiana: Society of Petrophysicists and Well-Log Analysts.

[26] Wyllie, M.R.J., Gregory, A.R., Gardner, L.W., 1956, Elastic wave velocities in heterogeneous and porous media: Geophysics, V. 21, p. 41-70.

Received: May 2025; Revised: June 2025; Accepted: June 2025; Published: June 2025