IMPLEMENTATION OF NOISE FILTERING ALGORITHM FOR AUTOMATED PRESSURE TRANSIENT ANALYSIS

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

DOĞUHAN COŞAR

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
PETROLEUM AND NATURAL GAS ENGINEERING

SEPTEMBER 2022

Approval of the thesis:

IMPLEMENTATION OF NOISE FILTERING ALGORITHM FOR AUTOMATED PRESSURE TRANSIENT ANALYSIS

submitted by **DOĞUHAN COŞAR** in partial fulfillment of the requirements for the degree of **Master of Science in Petroleum and Natural Gas Engineering, Middle East Technical University** by,

Prof. Dr. Halil Kalıpçılar
Dean, Graduate School of **Natural and Applied Sciences** _____

Asst. Prof. Dr. İsmail Durgut
Head of the Department, **Petroleum and Natural Gas Eng.** _____

Assoc. Prof. Dr. Çağlar Sınayuç
Supervisor, **Petroleum and Natural Gas Eng., METU** _____

Examining Committee Members:

Asst. Prof. Dr. İsmail Durgut
Petroleum and Natural Gas Eng., METU _____

Assoc. Prof. Dr. Çağlar Sınayuç
Petroleum and Natural Gas Eng., METU _____

Asst. Prof. Dr. Doruk Alp
Petroleum and Natural Gas Eng., METU NCC _____

Date: 02.09.2022

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name Last name : Doğuhan Coşar

Signature :

ABSTRACT

IMPLEMENTATION OF NOISE FILTERING ALGORITHM FOR AUTOMATED PRESSURE TRANSIENT ANALYSIS

Coşar, Doğuhan
Master of Science, Petroleum and Natural Gas Engineering
Supervisor: Assoc. Prof. Dr. Çağlar Sınayuç

September 2022, 102 pages

In order to estimate reservoir parameters such as permeability and skin factor, well tests are performed. Pressures are recorded periodically during these tests. These parameters are crucial in predicting the amount of oil or gas that can be produced. The analysis of pressure changes provide information about the reservoir characteristics and its ability to produce hydrocarbons at a certain amount of time. Three different cases of drawdown tests are analyzed. Derivative curves are plotted using data points that are separated by certain intervals, ranging from 0.10 to 0.50 of a log cycle, because using adjacent data points results in very noisy derivative values while differentiating. Forty one different differentiation intervals are used in total. A new method is developed to find the log cycle interval that eliminates the highest level of noise without overly flattening the derivative curve. Second and third derivatives are also calculated for each and every log cycle interval and checked in order to find the most suitable one.

Early, middle and late time regions, hence different flow regimes, of both cases are identified using this log cycle interval. Pressure and time data from middle time region are selected and used for the estimation process since best estimates of permeability can be made from this region.

Skin factor and permeability values for both cases are estimated with two different methods. The first one is an iterative method and takes around 1-5 minutes to run while the second one is a graphical method and takes just seconds. Both methods can estimate permeability and skin factor with high accuracy and low error. Negligible amount of difference occurs between the two methods, the first method being slightly more accurate.

The results obtained are compared with the results from KAPPA's Saphir Module, an industry standard PTA module and it is seen that the Python code can make more accurate estimations of permeability and skin factor. Also, it can handle noises up to ±2.5%.

Keywords: Well Test Analysis, Derivative Plot, Reservoir Parameter Estimation, Skin Factor, Permeability

ÖZ

OTOMATİKLEŞTİRİLMİŞ GEÇİCİ BASINÇ ANALİZİNDE GÜRÜLTÜ FİLTRELEME ALGORİTMASININ KULLANIMI

Coşar, Doğuhan
Yüksek Lisans, Petrol ve Doğal gaz Mühendisliği
Tez Yöneticisi: Doç. Dr. Çağlar Sınayuç

Eylül 2022, 102 sayfa

Rezervuar parametrelerinin belirlenebilmesi için kuyu testleri yapılır ve bu testler sırasındaki basınç değerleri periyodik olarak kaydedilir. Bu parametreler belirli bir zamanda üretilebilecek olan petrol veya doğal gaz miktarının belirlenebilmesi için son derece önemlidir. Testler sırasında kaydedilen basınç değerleri arasındaki farklara bakılarak rezervuarın karakteristik özellikleri ve hidrokarbon üretim kapasitesi hakkında önemli bilgiler elde edilir. Toplamda üç farklı basınç düşüm testi analiz edilmiştir. Türev grafikleri oluşturulurken çok fazla gürültüye neden olan ardışık noktaları kullanmak yerine 0.10 ile 0.50 arasında değişen logaritmik periyot aralıkları kullanılmıştır. Toplamda kırk bir farklı aralık kullanılmıştır. Python kullanılarak yeni bir metod geliştirilmiş, gürültüyü en çok elimine eden ve türev grafiğini aşırı düzleştirmeden en pürüzsüz olacak şekilde oluşturan logaritmik periyot aralık değeri bulunmuş ve parametre tahminlerinde kullanılmıştır. En uygun logaritmik periyot aralığını bulabilmek için kırk bir farklı aralık için de ikinci ve üçüncü türev değerleri hesaplanmış ve kontrol edilmiştir. Erken, orta ve geç zaman bölgeleri ve dolayısıyla farklı akış rejimleri bu aralık kullanılarak belirlenmiştir. Estimasyon aşamasında orta zaman bölgelerindeki basınç verileri kullanılmaktadır

çünkü en isabetli saptamalar bu bölgeden yapılmaktadır. Her iki rezervuarın da skin faktörü ve geçirgenlik değerleri iki farklı metod kullanılarak hesaplanmıştır.
İlk metod tekrarlı metoddur ve estimasyon süreci yaklaşık bir ile beş dakika arasında değişen süreleri bulmaktadır. İkinci metod ise grafiksel metoddur ve saniyeler içinde estimasyon yapabilmektedir. İki metod da geçirgenlik ve skin faktör değerlerini yüksek isabet ve düşük hata ile tahmin etmektedir. İlk metod biraz daha isabetli tahminler yapsa da iki metod arasındaki fark ihmal edilebilecek seviyelerdedir.
Elde edilen sonuçlar, endüstri standardı PTA programı olan KAPPA'nın Saphir Modülü ile elde edilen sonuçlarla kıyaslanmış ve Python kodunun daha isabetli geçirgenlik ve skin faktörü tahminleri yaptığı gözlemlenmiştir. Ayrıca Python kodu ±%2.5 gürültüye kadar dayanım göstermektedir.

Anahtar Kelimeler: Kuyu Test Analizi, Türev Grafiği, Rezervuar Parametre Estimasyonu, Skin Faktörü, Geçirgenlik

to my family

ACKNOWLEDGMENTS

I would like to express my deepest gratitudes to my supervisor Assoc. Prof. Dr. Çağlar Sınayuç and my father Prof. Dr. Ahmet Coşar, for their guidance and support throughout the research.

I would also like to thank my family and friends, they were always there for me and made me the person that I am today.

TABLE OF CONTENTS

ABSTRACT v

ÖZ vii

ACKNOWLEDGMENTS x

TABLE OF CONTENTS xi

LIST OF TABLES xiv

LIST OF FIGURES xvi

LIST OF SYMBOLS xix

CHAPTERS

1 INTRODUCTION 1

1.1 Traditional Graphical Analysis 1

1.2 Computer Aided Analysis 3

2 LITERATURE REVIEW 5

2.1 Well Tests 5

2.1.1 Drawdown Test 6

2.1.2 Buildup Test 7

2.2 Well Test Analysis 8

2.2.1 Previous Studies on Well Test Analysis 8

2.2.1.1 Well Test Analysis Between 1930s-1950s 8

2.2.1.2 Well Test Analysis Between 1950s-1960s 9

2.2.1.3 Well Test Analysis Between 1960s-1980s 9

2.2.1.4 Well Test Analysis Between 1980s-2000s 10

2.2.1.4.1 Diagnostic Plots 10

2.2.1.5 Well Test Analysis From 2000s to Present 12

2.3 Pressure Data Measurements 13
2.4 Reservoir Parameters 14
2.4.1 Permeability 14
2.4.2 Porosity 16
2.4.3 Skin Factor 18
3 STATEMENT OF THE PROBLEM 19
4 RESERVOIR PARAMETER ESTIMATION 21
4.1 Derivative Plots 21
4.1.1 Noise Elimination of Derivative Plots 24
4.1.1.1 Selecting the Smoothest Derivative Plot for Drawdown Case I 30
4.1.1.2 Selecting the Smoothest Derivative Plot for Drawdown Case II 39
4.2 Determination of the Middle Time Region 44
4.2.1 Determination of the Middle Time Region for Drawdown Case I 45
4.2.2 Determination of the Middle Time Region for Drawdown Case II .. 50
4.3 Permeability and Skin Factor Estimation with Iterative Approach 52
4.3.1 Drawdown Case I 53
4.3.2 Drawdown Case II 56
4.4 Permeability and Skin Factor Estimation with Graphical Analysis 59
4.4.1 Drawdown Case I 60
4.4.2 Drawdown Case II 63
4.5 Permeability and Skin Factor Estimation Using KAPPA Saphir Module 66
4.5.1 Drawdown Case I 66

4.5.2 Drawdown Case II 71
4.6 Drawdown Case III 75
4.6.1 Gaussian Noise Addition 80
5 RESULTS AND DISCUSSION 83
6 CONCLUSIONS 87
REFERENCES 89
APPENDICES
A. Python Code 93

LIST OF TABLES

TABLES

Table 1.1 Characteristics of Different Flow Periods[3] 2
Table 1.2 Characteristics of Different Flow Periods on Derivative Plots[3] 3
Table 2.1 History of Well Test Analysis[2] 12
Table 2.2. Porosity Values 17
Table 4.1 Drawdown Test Data for Case I 22
Table 4.2 Drawdown Test Data for Case II 39
Table 4.3 Middle Time Region Estimation For Drawdown Case I 48
Table 4.4 Middle Time Region Estimation For Drawdown Case II 50
Table 4.5 Known Parameters of Drawdown Case I 53
Table 4.6 Comparison of Pwf(calculated with k=75.58 md and s=+5.85) and Pwf(measured) 54
Table 4.7 Known Parameters of Drawdown Case II 56
Table 4.8 Comparison of Pwf(calculated with k=9.16 md and s=+4.00) and Pwf(measured) 57
Table 4.9 Comparison of Pwf(calculated with k=76.58 md and s=+6.01) and Pwf(measured) 60
Table 4.10 Comparison of $P_{wf(\text{calculated with } k=9.14 \text{ md and } s=+3.97)}$ and $P_{wf(\text{measured})}$ 63
Table 4.11 Comparison of Pwf(calculated with k=76.5316 md and s=+6.01963) and Pwf(measured) 70
Table 4.12 Comparison of Pwf(calculated with k=9.65699 md and s=+4.59090) and Pwf(measured) 74
Table 4.13 Drawdown Test Data for Case III 75
Table 4.14 Known Parameters of Drawdown Case III 76
Table 4.15 Comparison of Actual and Estimated Values 79
Table 4.16 Permeability and Skin Factor Estimations with Randomly Added Noises 81

Table 5.1 Absolute Error Percentages of Different Methods 84
Table 5.2 Results for ±0.5% Error 84
Table 5.3 Results for ±1% Error 84
Table 5.4 Results for ±1.5% Error 84
Table 5.5 Results for ±2% Error 85
Table 5.6 Results for ±2.5% Error 85
Table 5.7 Results for ±3% Error 85

LIST OF FIGURES

FIGURES

Figure 2.1 P vs t and q vs t Plot of an Ideal Drawdown Test 6
Figure 2.2 P vs t and q vs t Plot of an Ideal Buildup Test 7
Figure 2.3 A diagnostic plot [1] 10
Figure 2.4 Shapes of Log-Log Derivative Curves at Different Flow Regimes[2] .. 11
Figure 2.5 Illustration of PDG[25] 13
Figure 2.6 Linear Flow Model[26] 15
Figure 2.7 Radial Flow Model[26] 15
Figure 2.8 Schematic of wellbore skin effect[27] 18
Figure 4.1 Derivative Plot Formed by Using Eq 4.1 23
Figure 4.2 Derivative Plot Formed by Using Eq 4.2 25
Figure 4.3 Derivative Plot Formed by Using Eq 4.3 (0.10 Log Cycle Interval) 27
Figure 4.4 Derivative Plot Formed by Using Eq 4.3 (0.50 log cycle interval) 28
Figure 4.5 Derivative Plot Formed by Using Eq 4.3 (Average 0.10-0.50) 29
Figure 4.6 1st and 2nd Derivative Plots(0.26 Log Cycle Interval) 31
Figure 4.7 1st and 3rd Derivative Plots(0.26 Log Cycle Interval) 32
Figure 4.8 1st and 2nd Derivative Plots(0.31 Log Cycle Interval) 33
Figure 4.9 1st and 3rd Derivative Plots(0.31 Log Cycle Interval) 34
Figure 4.10 Comparison of First Derivatives of 0.26 and 0.31 Log Cycle Intervals 35
Figure 4.11 Comparison of Second Derivatives of 0.26 and 0.31 Log Cycle Intervals 36
Figure 4.12 Comparison of Third Derivatives of 0.26 and 0.31 Log Cycle Intervals 37
Figure 4.13 Log Cycle Interval vs Sign Change Ratio Plot for Case I 38
Figure 4.14 Comparison of the Derivative Plots of 0.32 and 0.40 Log Cycle Intervals 40

Figure 4.15 Comparison of the 2nd Derivatives for 0.32 and 0.40 Log Cycle Intervals....41
Figure 4.16 Comparison of the 3rd Derivatives for 0.32 and 0.40 Log Cycle Intervals....42
Figure 4.17 Log Cycle Interval vs Sign Change Ratio Plot for Case II....43
Figure 4.18 Eyeball Estimation of Middle Time Region for Case I....45
Figure 4.19 1st, 2nd and 3rd Derivative Plots (0.31 Log Cycle Interval)....46
Figure 4.20 1st, 2nd and 3rd Derivative Plots (0.26 Log Cycle Interval)....47
Figure 4.21 1st and 2nd Derivative Plots of Drawdown Case I with Early, Middle and Late Time Regions....49
Figure 4.22 1st and 2nd Derivative Plots of Drawdown Case II with Early, Middle and Late Time Regions....51
Figure 4.23 Semilog Plot of P_{wf} vs t, with Best Match Semilog Straight Line(k=75.58 md and s=+5.85)....55
Figure 4.24 Comparison of ΔPwf(measured) and ΔPwf(calculated with k=75.58 md and s=+5.85)....55
Figure 4.25 Semilog Plot of Pwf vs t, with Best Match Semilog Straight Line(k=9.16 md, s=+4.00)....58
Figure 4.26 Comparison of ΔPwf(measured) and ΔPwf(calculated with k=9.16 md and s=+4.00)....58
Figure 4.27 Comparison of ΔPwf(measured) and ΔPwf(calculated with k=76.58 md and s=+6.01)....61
Figure 4.28 Semilog Plot of Pwf vs t, with Best Match Semilog Straight Line(k=76.58 md and s=+6.01)....62
Figure 4.29 Comparison of ΔPwf(measured) and ΔPwf(calculated with k=9.14 md and s=+3.97)....64
Figure 4.30 Semilog Plot of Pwf vs t, with Best Match Semilog Straight Line (k=9.14 md and s=+3.97)....65
Figure 4.31 History Plot of P vs t and Q vs t for Drawdown Case I....66

Figure 4.32 Derivative Plot Drawn by Saphir for Drawdown Case I (0.1 Log Cycle) 67
Figure 4.33 k and S Estimation Made by Saphir for Drawdown Case I(0.1 Log Cycle) 67
Figure 4.34 Derivative Plot Drawn by Saphir for Drawdown Case I (0.31 Log Cycle) 68
Figure 4.35 k and S Estimation Made by Saphir for Drawdown Case I(0.31 Log Cycle) 69
Figure 4.36 History Plot of P vs t and Q vs t for Drawdown Case II 71
Figure 4.37 Derivative Plot Drawn by Saphir for Drawdown Case II (0.1 Log Cycle) 71
Figure 4.38 k and S Estimation Made by Saphir for Drawdown Case II(0.1 Log Cycle) 72
Figure 4.39 Derivative Plot Drawn by Saphir for Drawdown Case I (0.31 Log Cycle) 72
Figure 4.40 k and S Estimation Made by Saphir for Drawdown Case I(0.31 Log Cycle) 73
Figure 4.41 Diagnostic Plot of Drawdown Case III 76
Figure 4.42 Log Cycle Interval vs Sign Change Ratio Plot for Case III 77
Figure 4.43 First and Second Derivative Curves (0.26 Log Cycle Interval) 78
Figure 4.44 Comparison of Derivative Plots 79
Figure 4.45 Pwf vs t Plots with Randomly Added Noises 80

LIST OF SYMBOLS

SYMBOLS

μ	Viscosity, cp
A	Cross sectional area, ft^2
B_o	Oil formation volume factor, bbl/stb
C_t	Total compressibility, $1/psi$
h	Thickness, ft
k	Permeability, md
L	Length, ft
P	Pressure, psi
Q	Flow rate, $^{bbl}/_{day}$
r_w	Wellbore radius, ft
v	Apparent fluid flowing velocity, $^{bbl}/_{day \times ft^2}$
V_B	Bulk volume
V_G	Grain volume
V_P	Pore volume
ϕ	Porosity

CHAPTER 1

INTRODUCTION

In order to make the most meaningful decisions on any reservoir and maximize profitability, the reservoir, and every parameter, and characteristics it has should be identified before decision making processes. There are different methods to identify these parameters and characteristics, and a most important one is well test analysis.

1.1 Traditional Graphical Analysis

Well test analysis was first used for groundwater hydrology before it became an essential analysis method in the petroleum industry. Well tests had been interpreted by traditional graphical analysis, way before computer aided analysis. The flow periods are identified by checking derivative plots and semi-log plots. Reservoir parameters are estimated from the log-log and semi-log plots. As the final step, type curve matching of the log-log derivative type curves are made and reservoir parameters are re-estimated if necessary.

The flow periods that take place during a well test, which are also mentioned on the next page, can be interpreted from the corresponding characteristic plots. Some of these plots are used for estimating parameters.

Table 1.1 Characteristics of Different Flow Periods[3]

Flow Period	**Plot**	**Characteristic**
Infinite Acting Radial Flow Drawdown	P vs logΔt	Semilog Straight Line
Infinite Acting Radial Flow Buildup	P vs $\log(t_P+\Delta t)/\Delta t$	Straight Line
Wellbore Storage	logΔP vs logΔt	Unit Slope Straight Line (P vs t)
Finite Conductivity Fracture	logΔP vs logΔt	Straight Line with Slope ¼
Infinite Conductivity Fracture	logΔP vs logΔt	Straight Line with Slope ½
Double Porosity	P vs logΔt	S-Shaped Transition Between Semilog Straight Lines
Closed Boundary	P vs Δt	Pseudosteady State Pressure Linear with Time
Impermeable Boundary	P vs logΔt	Doubling of Slope on Semi-log Straight Line
Constant Pressure Boundary	any	Constant Pressure Flat Line on All P/t Plots

In addition to these characteristic plots, derivative plots were introduced by Bourdet et al. in 1983, which became one of the greatest milestones of the modern well test analysis. Derivative plots contain $\Delta t \log {}^{\partial P}/_{\partial t}$ vs $\log \Delta t$, diagnostic plots on the other hand contain both $\log \Delta P$ vs $\log \Delta t$ and $\Delta t \log {}^{\partial P}/_{\partial t}$ vs $\log \Delta t$. They are arguably the most useful tools for diagnosis and will be the main focus of this study.

1.2 Computer Aided Analysis

Computer aided analysis is much faster compared to traditional graphical techniques. It also eliminates human error that can occur while plotting or reading the graph. Computer aided analysis can handle situations that would be impossible to handle with traditional methods, such as complex geometries, varying rate etc.

Table 1.2 Characteristics of Different Flow Periods on Derivative Plots[3]

Flow Period	**Characteristic**
Wellbore Storage	Unit Slope Line plus a Hump
Infinite Acting Radial Flow Buildup	Flat Region
Infinite Acting Radial Flow Drawdown	Flat Region
Finite Conductivity Fracture	Straight Line with Slope ¼
Infinite Conductivity Fracture	Straight Line with Slope ½
Double Porosity	Reverse Hump with a Minimum
Closed Boundary	Steep Rising Straight Line
Impermeable Boundary	Second Flat Region
Constant Pressure Boundary	Continuously Decreasing Line

CHAPTER 2

LITERATURE REVIEW

2.1 Well Tests

It is a necessity for a reservoir engineer to have sufficient information about well/reservoir to be able to analyze the reservoir performance and predict the future performance under various circumstances.[4] Well tests provide information about in situ reservoir conditions. These conditions determine the production capacity of a reservoir. Pressures and their changes over time make it possible to determine unknown reservoir parameters such as permeability and skin factor. Hence well tests are the most valuable for reservoir engineering without a doubt.

Primary objectives of the oil well tests are:

- Evaluating well conditions
- Reservoir characterization
- Obtaining unknown reservoir parameters for reservoir description
- Estimating damage caused during drilling/completion and deciding if stimulation is necessary or not.
- Determining pay zones

2.1.1 Drawdown Test

A well that was initially shut-in and at static condition is opened to flow. Ideally, the flow rate should be kept constant in order to conduct traditional analysis but in reality this is hard to achieve. This type of test is mostly used for testing the reservoir's limits.

The pressure is constant until the beginning of production. As production begins, pressure starts to drop. Initially produced fluids comes from the wellbore itself, not the reservoir and this period is called "wellbore storage". Analyzing the pressure drops make it possible to estimate some unknown reservoir parameters, such as permeability and skin factor. One of the advantages of this type of testing is that fluid production continues while conducting the test.

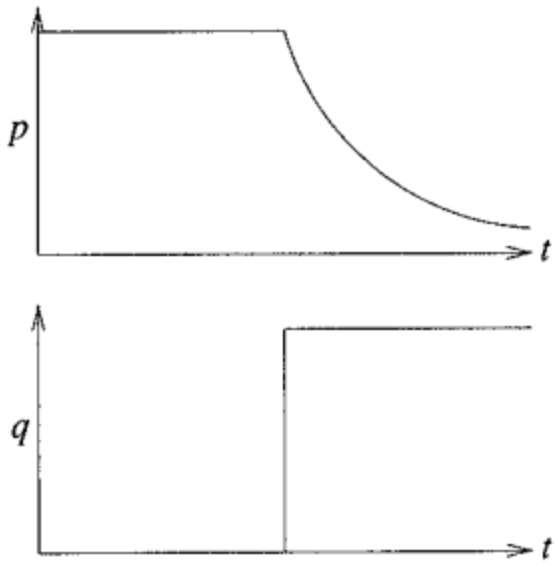


Figure 2.1 P vs t and q vs t Plot of an Ideal Drawdown Test

2.1.2 Buildup Test

The well that was initially flowing, ideally at a constant rate, is shut-in. Pressure begins to increase right after this shut-in. A short period of flow called "afterflow" occurs from reservoir to the wellbore right after the well is shut-in.

It is much easier to sustain a constant flow rate compared to a drawdown test because desired flow rate is zero. Biggest disadvantage of this test is that the well is not producing any fluids and this means a loss of profit.

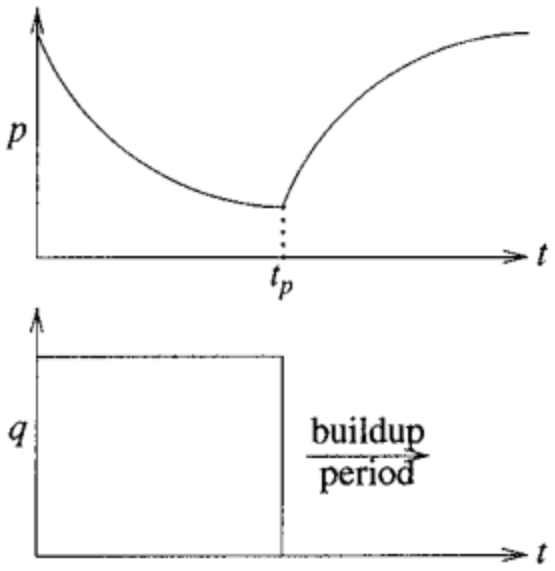


Figure 2.2 P vs t and q vs t Plot of an Ideal Buildup Test

2.2 Well Test Analysis

The main aim of the well test analysis is to interpret the pressure response due to change in flow rate. Well test analysis plays a crucial role in reservoir management. In order to be able to make the most meaningful decisions on any reservoir and maximize profitability, the reservoir and every parameter and characteristics it has should be identified before decision making processes.

There are different methods to identify these parameters and characteristics and one of the most important methods is well test analysis.

2.2.1 Previous Studies on Well Test Analysis

Gringarten summarized the most important developments in well test analysis in detail with his paper.[2]

2.2.1.1 Well Test Analysis Between 1930s-1950s

Well test analysis techniques started with the groundwater hydrology. Semilog straight line analysis and type curve matching was introduced by Theis in 1935[5]. In 1946, Cooper and Jacob[6] applied semilog straight line analysis and suggested a generalized graphical method that evaluates reservoir parameters.

2.2.1.2 Well Test Analysis Between 1950s-1960s

The interpretation techniques that were used during the 1950s and 1960s are mostly developed by oil companies and illustrated by the works of Miller et al.[7], Horner[8], Warren and Root[9], Odeh and Jones[10], Matthews et al.[11]. These interpretation methods were composed of semilog straight lines that apply to middle time data and simple boundary effects that can be inferred from late time data. Due to the technological constraints at those times, analysis were performed by hand, using pencil and graph papers. The data was obtained from production operations and well tests. Interpretable parameters at these years were limited to average reservoir pressure, permeability, drainage area and skin factor.

2.2.1.3 Well Test Analysis Between 1960s-1980s

The development of well test analysis in the late 1960s and early 1970s was mostly led by universities. The scope of analysis shifted to early time behavior because of the ambiguity of some obtained results from straight line analysis[2]. Ramey[12] introduced type curves in 1970 and the understanding of skin factor greatly improved during these years. He found out that positive skin factor may result from well damage or partial penetration and negative skin factor can be achieved by well stimulation operations such as acidizing and fracturing. Type curve matching greatly enhanced semilog straightline analysis by making it possible to select the most applicable straight line. Agarwal et al.[13] focused on the wellbore storage while Gringarten et al.[14] and Mavor and Cinco-Ley[15] focused on the fractures.

Gringarten et al.[16] and Bourdet and Gringarten[17] greatly enhanced the type curve analysis by introducing the concept of independent variables and integrated well test analysis methodology. Thanks to these developments, analysis results were more consistent and reliable. They also ended the era of manual analysis and started the era of computers in well test analysis which caused it to become more useful reservoir description tool.

2.2.1.4 Well Test Analysis Between 1980s-2000s

Identification of more complex reservoir behaviors such as double porosity became possible thanks to the new interpretation models based on Stehfest's Laplace inversion algorithm[18]. With these developments, well test analysis became more descriptive during exploration and reservoir simulation. In addition to the type curve developments in 1970s and 1980s, derivative plots were introduced in 1983 by Bourdet et al.[19], which became one of the greatest milestones of the modern well test analysis. It made well test analysis a true reservoir characterization method.

2.2.1.4.1 Diagnostic Plots

A diagnostic plot contains both $\log \Delta P$ vs $\log \Delta t$ and $\Delta t \log \frac{\partial P}{\partial t}$ vs $\log \Delta t$. According to Horne[3], it was by far the most useful diagnosis tool at those times.

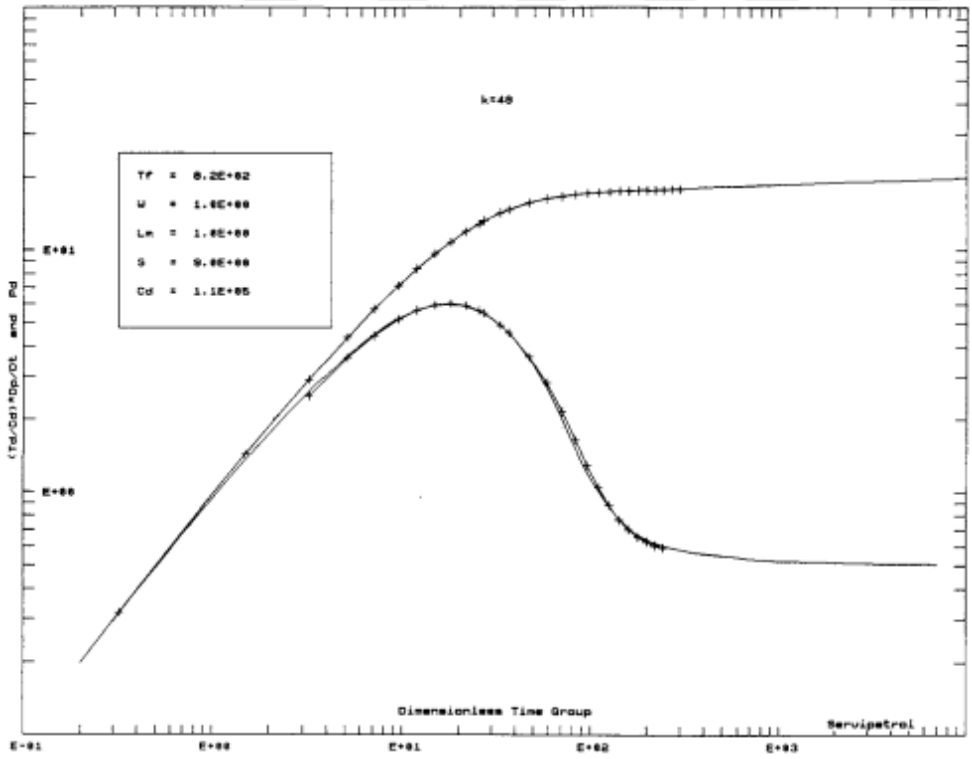


Figure 2.3 A diagnostic plot [1]

Thanks to the derivative plots:

- Heterogeneous reservoir behaviors such as double permeability can be recognized.[20, 21]
- Partial penetration, limited entry and other effects near wellbore can be identified.[22]
- Horizontal wells can be analyzed.[23]
- Wide range of boundary effects can be recognized.[24]

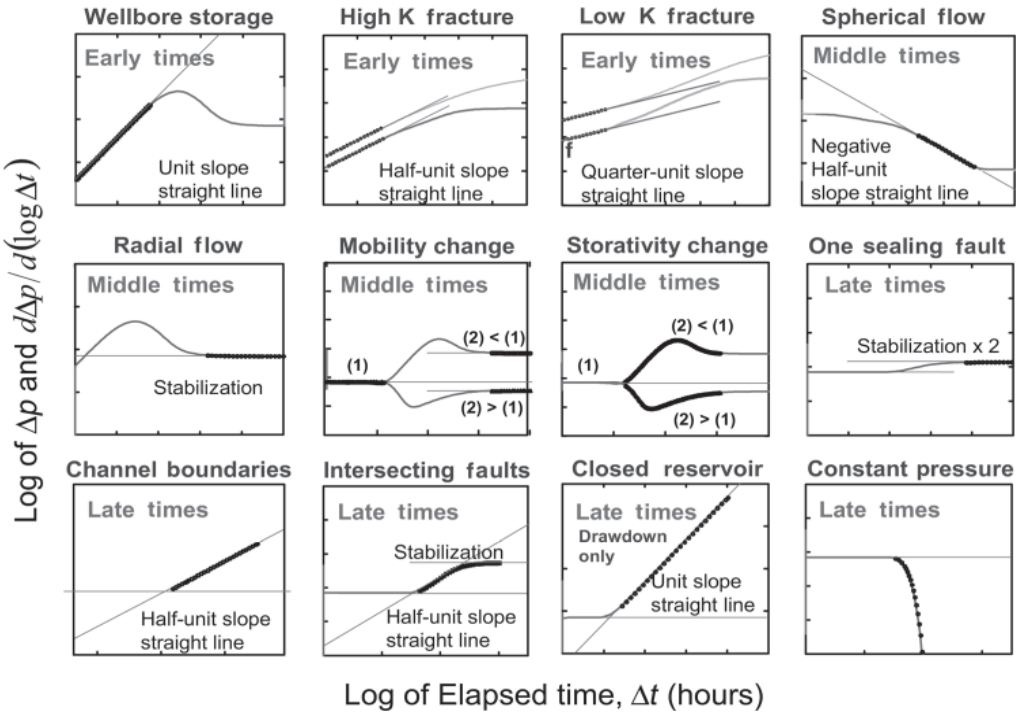


Figure 2.4 Shapes of Log-Log Derivative Curves at Different Flow Regimes[2]

2.2.1.5 Well Test Analysis From 2000s to Present

Schroeter et al. introduced an effective deconvolution algorithm in 2001.[25] It greatly enhanced the capability of well test analysis by converting variable rate pressure data into constant rate single drawdown with duration of all variable rate test durations combined. Thanks to this, more data become interpretable, hence identifying the interpretation model is an easier task.

The table below summarizes the history of well test analysis.

Table 2.1 History of Well Test Analysis[2]

Date	Interpretation Method	Tools	Emphasis
50s	Straight lines	Laplace transform	Homogeneous reservoir behavior
Late 60s Early 70s	Pressure type-curve analysis	Green's functions	Near-wellbore effects
Late 70s	Type curves with independent variables	Integrated methodology Stehfest algorithm	Dual-porosity behavior
Early 80s	Derivatives	Computerized analysis	Heterogeneous reservoir behavior and boundaries
90s		Computer-aided analysis downhole rate measurements integration with interpretation models from other data	Multilayered reservoir
Early 00s		Deconvolution	Enhanced radius of investigation boundaries

2.3 Pressure Data Measurements

Pressure measurements were made by Bourdon-tupe mechanical gauges until early 1970s. Their resolution and accuracy were limited. Electronic gauges were invented in early 1970s and improved greatly throughout 1980s. They enhanced the quality of pressure data recorded and made it possible to monitor bottomhole pressure at the surface. Permanent downhole pressure gauges were commonly used after late 1980s, as the name implies, they continuosly monitor and record pressure, flow rate and temperature changes during production phase. They are installed during the well completion phase.

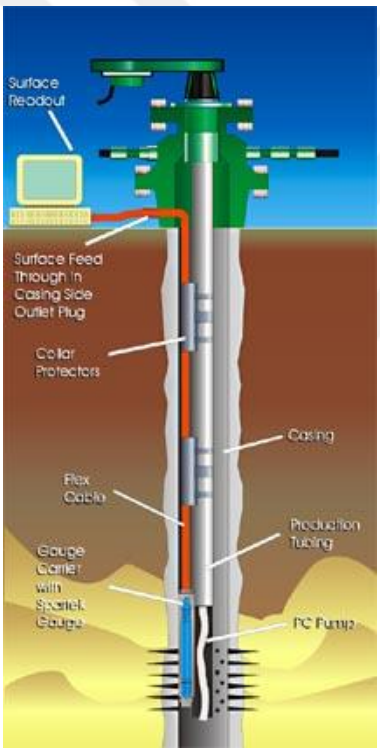


Figure 2.5 Illustration of PDG[25]

2.4 Reservoir Parameters

2.4.1 Permeability

Permebility is a measure of the capacity of a porous medium to transmit fluids. It directly affects the flow rate and directional movement of the reservoir fluids, hence it is one of the most important rock properties.

Henry Darcy defined it mathematically for the first time in 1856. It is called Darcy's Law and can be seen from the equations below. Eq 2.1 is in linear form while Eq 2.2 is in radial form.

1 Darcy is a very high permeability and the term millidarcy(md) is used. Most of the reservoir rocks have permeabilities less than 1 Darcy.

$$v = \frac{Q}{A} = -\frac{k}{\mu}\frac{dP}{dL} \qquad \text{Eq 2.1}$$

$$Q = \frac{k\ A}{\mu}\frac{dP}{dr} \qquad \text{Eq 2.2}$$

where

Q = flow rate of fluid, $^{bbl}/_{day}$
A = cross sectional area, ft^2
v = apparent fluid flowing velocity, $^{bbl}/_{day \times ft^2}$
μ = viscosity of the fluid, cp
k = permeability of the rock, $Darcy$
P = pressure, psi
L = length, ft
r = radius, ft

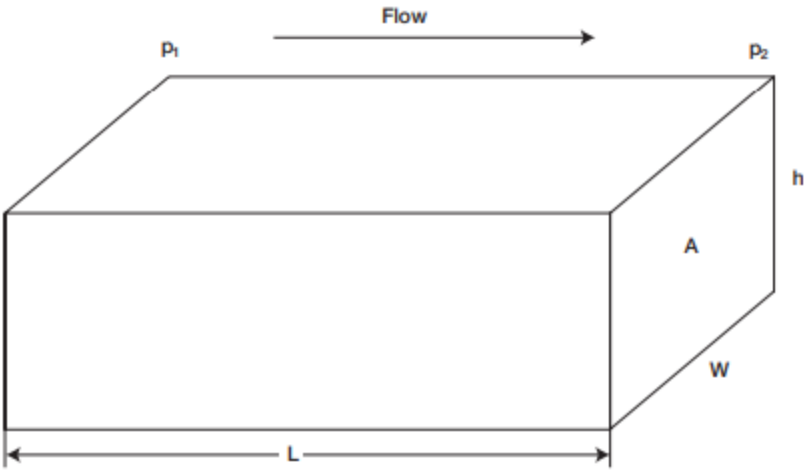


Figure 2.6 Linear Flow Model[26]

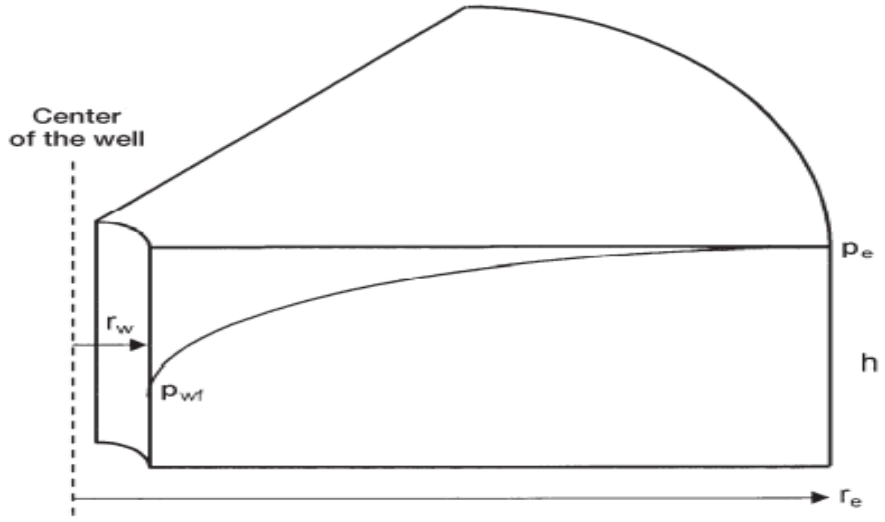


Figure 2.7 Radial Flow Model[26]

2.4.2 Porosity

Porosity is the ratio of the total void volume to the bulk volume of the rock. Reservoir rocks composed of rock grains(matrix) and pore spaces. Most of these pore spaces are filled with reservoir fluids such as oil, water and gas. Some pore spaces are connected with each other and some are isolated. In general, there are three main types of pores:

i) Interconnected Pores: The network of empty spaces that are interconnected with each other.

ii) Dead End Pores: The empty spaces that are connected with each other but they are trapped inside the rock with dead end.

iii) Isolated Pores: The empty spaces that are completely isolated from other empty spaces mostly due to cementation.

Total Porosity is the ratio of the total pore spaces to the bulk volume and can be calculated as:

$$\phi = \frac{V_P}{V_B} \qquad \text{Eq 2.3}$$

$$V_P = V_B - V_G \qquad \text{Eq 2.4}$$

where

V_P: Pore volume

V_B: Bulk volume

V_G: Grain volume

ϕ: Total porosity

Effective Porosity is the ratio of the volume of interconnected and dead end pores to the bulk volume and can be calculated as:

$$\phi = \frac{Volume\ of\ interconnected\ pores + Volume\ of\ dead\ end\ pores}{Bulk\ Volume} \quad \text{Eq 2.5}$$

Ineffective Porosity is the ratio of the volume of isolated pores to the bulk volume and can be calculated as:

$$\phi = \frac{Volume\ of\ isolated\ pores}{Bulk\ Volume} \quad \text{Eq 2.6}$$

The porosity is one of the most important reservoir rock properties and it directly affects the amount of reservoir fluids present in the reservoir and hence a reservoir with high porosity is much desired. Porosity is affected mostly by the grain size distribution, grain shape, grain sorting, clay content, compaction and cementation. All of these factors determine the porosity of the rocks. Porosity values are accepted as:

Table 2.2. Porosity Values

Negligible	$\phi < 5\%$
Low	$5\% < \phi < 10\%$
Good	$10\% < \phi < 15\%$
Very Good	$\phi < 20\%$

2.4.3 Skin Factor

Skin factor is actually a measure of the damage given during drilling/completion operations or enhancement to near wellbore payzone. This causes a pressure drop between the wellbore and formation[27].

Positive skin factor values indicate wellbore damage, hence reduced permeability while negative values indicate enhanced permeability.

Van Everdingen[28] and Hurst[29] defined the skin factor based on this pressure drop as can be seen from Eq 2.7.

$$S = \frac{2\pi kh}{Q\mu}\Delta P_s \qquad \text{Eq 2.7}$$

Hawkins[30] defined it as a zone with finite radius r_s having a permeability k_s as can be seen from Eq 2.8.

$$S = \left(\frac{k}{k_s} - 1\right)\ln\frac{r_s}{r_w} \qquad \text{Eq 2.8}$$

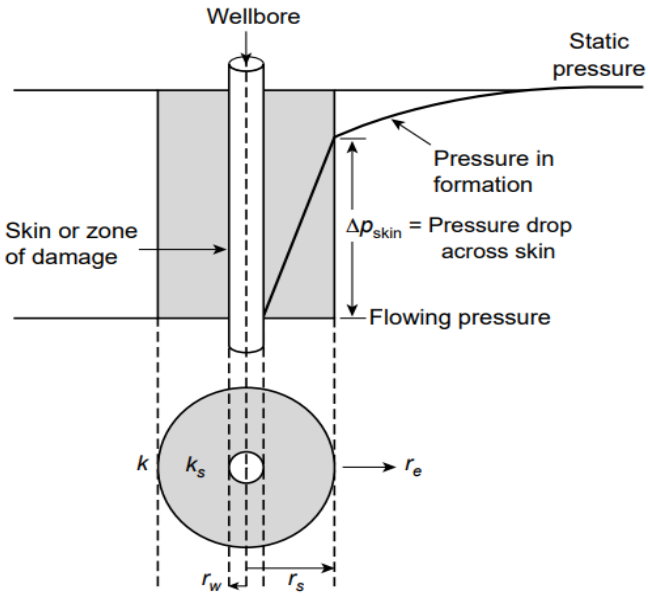


Figure 2.8 Schematic of wellbore skin effect[27]

CHAPTER 3

STATEMENT OF THE PROBLEM

Determination of the reservoir parameters such as permeability, porosity, skin factor etc. is crucial in petroleum engineering. In order to conduct a feasibility analysis, it is a must to determine these parameters as accurate as possible since millions of dollars are being spent throughout discovery and development of reservoirs. Well testing gives precious data for production evaluation and optimization. The accuracy of these data is crucial for the decision making processes and their feasibility. In order to make the most logical decisions, the reservoir parameters and characteristics should be known. Despite all the efforts given for the reliability of the well test data, some uncertainties are still present due to the multiphase flow measurement, complex fluid flow dynamics etc. They all cause noise in the data and they should be eliminated in order to be able to make the most logical decisions. Derivative plots can provide us with important information about reservoirs but calculating pressure derivatives is a sensitive process since the noise in the data can easily be amplified while doing so.

I will try to estimate the crucial reservoir parameters as accurate as possible in order to minimize the risk of spending unnecessary money. When a well is drilled for the first time and a reservoir is discovered, the first thing to consider is "Should I invest more and develop this reservoir?". The correct answer for this question can only be given after the determination of reservoir parameters, which is the ultimate goal of this Master's thesis. Various methods are used to estimate these parameters and they are used with more recent technological applications in order to further increase the accuracy and minimize errors of predictions.

CHAPTER 4

RESERVOIR PARAMETER ESTIMATION

4.1 Derivative Plots

Bourdet et al. introduced the derivative plots in addition to the characteristic plots in 1983[31]. As Horne stated in his book[3], derivative plot was the most useful diagnosis tool at the time.

Calculating pressure derivatives is a sensitive process since the noise in the data can easily be amplified while doing so. There are several different differentiation methods that can be applied.

The first one is numerically differentiating the adjacent points by using Eq 4.1. It is almost never used in well test analysis because it results in a very noisy derivative.

$$t\left(\frac{\partial P}{\partial t}\right)_i = t_i\left[\frac{(t_i - t_{i-1})\Delta p_{i+1}}{(t_{i+1} - t_i)(t_{i+1} - t_{i-1})} + \frac{(t_{i+1} + t_{i-1} - 2t_i)\Delta p_i}{(t_{i+1} - t_i)(t_i - t_{i-1})} - \frac{(t_{i+1} - t_i)\Delta p_{i-1}}{(t_i - t_{i-1})(t_{i+1} - t_{i-1})}\right] \quad \text{Eq 4.1}$$

72 hour long drawdown test data is taken from Horne book.[3].

Table 4.1 Drawdown Test Data for Case I

t, hours	P_{wf}, psi	t, hours	P_{wf}, psi	t, hours	P_{wf}, psi	t, hours	P_{wf}, psi
0	6009.00	0.14972	5009.74	1.55366	3318.80	16.1223	3018.23
0.0167	5867.82	0.1733	4886.13	1.79829	3289.38	18.6608	3002.85
0.01933	5845.93	0.20058	4769.13	2.08144	3263.02	21.60	2988.93
0.02237	5819.44	0.23217	4635.16	2.40918	3231.28	24.25	2939.30
0.0259	5792.50	0.26872	4501.08	2.78852	3216.27	26.37	2921.50
0.02997	5765.01	0.31103	4365.35	3.22758	3200.34	28.67	2902.30
0.03469	5720.90	0.36001	4219.70	3.73579	3175.40	31.17	2881.40
0.04016	5688.36	0.41669	4089.84	4.32401	3162.30	33.90	2858.70
0.04648	5642.92	0.4823	3960.16	5.00485	3139.87	36.86	2834.00
0.0538	5597.43	0.55824	3835.59	5.79289	3133.46	40.07	2807.20
0.06227	5521.66	0.64614	3727.20	6.70502	3114.87	43.57	2778.00
0.07207	5459.70	0.74788	3630.08	7.76076	3092.78	47.38	2746.20
0.08342	5389.75	0.86564	3538.77	8.98274	3081.99	51.51	2711.70
0.09655	5306.48	1.00194	3465.23	10.3971	3062.07	56.01	2674.20
0.11176	5211.11	1.1597	3411.56	12.0342	3047.29	60.90	2633.40
0.12935	5117.79	1.3423	3361.60	13.9291	3037.98	72.00	2540.70

A derivative plot is formed in Excel by using a straightforward numerical differentiation of the adjacent points, Eq 4.1, and it is quite clear that high amount of noise is present in the derivative values, as can be seen from the figure below:

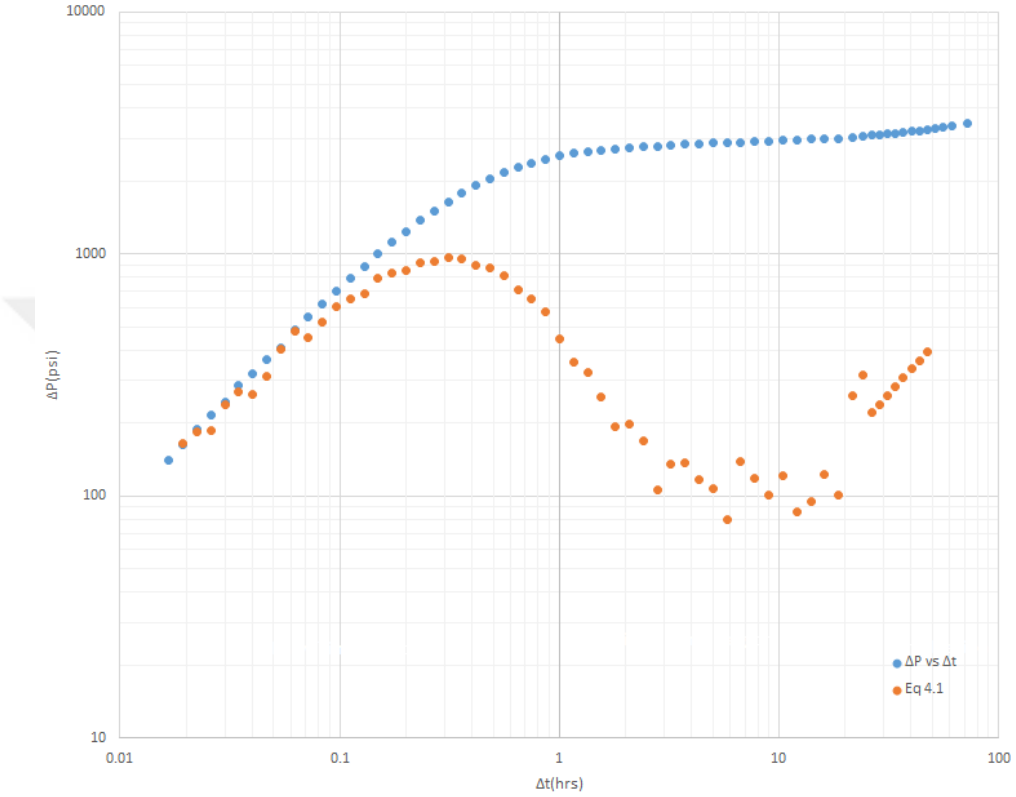


Figure 4.1 Derivative Plot Formed by Using Eq 4.1

4.1.1 Noise Elimination of Derivative Plots

It is suggested in the literature that numerical differentiation with respect to natural logarithm of time by using Eq 4.2 would reduce the noise in the derivative calculation compared to the previous method.

$$t\left(\frac{\partial P}{\partial t}\right)_i = \left(\frac{\partial P}{\partial \ln t}\right)_i = A + B - C$$

where Eq 4.2

$$A = \frac{\ln\left(\frac{t_i}{t_{i-1}}\right)\Delta p_{i+1}}{\ln\left(\frac{t_{i+1}}{t_i}\right)\ln\left(\frac{t_{i+1}}{t_{i-1}}\right)}$$

$$B = \frac{\ln\left(\frac{t_{i+1}t_{i-1}}{t_i^2}\right)\Delta p_i}{\ln\left(\frac{t_{i+1}}{t_i}\right)\ln\left(\frac{t_i}{t_{i-1}}\right)}$$

$$C = \frac{\ln(t_{i+1}/t_i)\Delta p_{i-1}}{\ln(t_i/t_{i-1})\ln(t_{i+1}/t_{i-1})}$$

Another derivative plot is drawn in Excel, this time using Eq 4.2

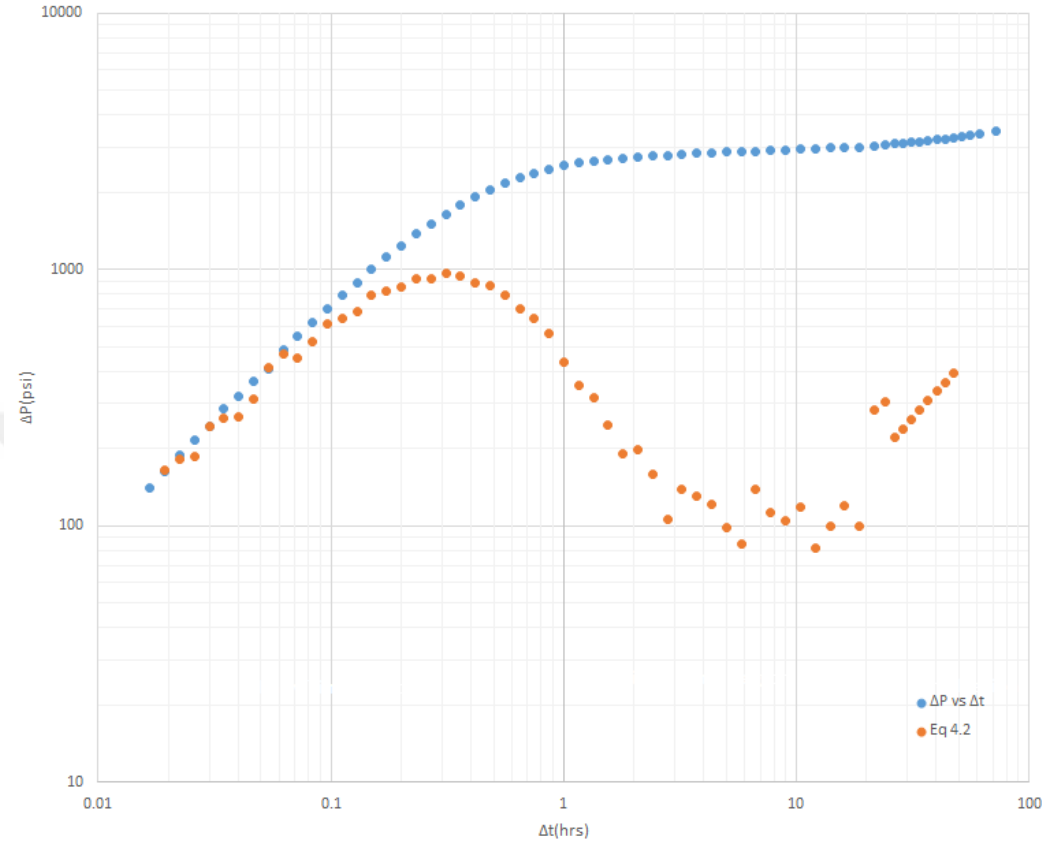


Figure 4.2 Derivative Plot Formed by Using Eq 4.2

It can be easily seen that there is not much of a difference between the Figure 4.1 and Figure 4.2, meaning that there is still quite high noise in the derivative plot and we need do something else to get rid of the noise.

It is said that while differentiating with respect to natural logarithm of time, using data points that are separated by at least 0.2 of a log cycle instead of using adjacent points would further reduce the noise. One of the shortcomings of this method includes running out of data during the first and last differentiation intervals. So the third differentiation method is

$$t\left(\frac{\partial p}{\partial t}\right)_i = \left(\frac{\partial p}{\partial \ln t}\right)_i = A + B - C \qquad \text{Eq 4.3}$$

$$A = \frac{\ln(t_i/t_{i-k})\Delta p_{i+j}}{\ln(t_{i+j}/t_i)\ln(t_{i+j}/t_{i-k})}$$

$$B = \frac{\ln(t_{i+j}t_{i-k}/{t_i}^2)\Delta p_i}{\ln(t_{i+j}/t_i)\ln(t_i/t_{i-k})}$$

$$C = \frac{\ln(t_{i+j}/t_i)\Delta p_{i-k}}{\ln(t_i/t_{i-k})\ln(t_{i+j}/t_{i-k})}$$

where

$$\ln t_{i+j} - \ln t_i \geq 0.2$$

$$\ln t_i - \ln t_{i-k} \geq 0.2$$

Horne[3] stated that this differentiation interval can be replaced by values in between 0.1-0.5 depending on the case.

When differentiating the late time data, length between the points becomes larger than the last data point and the previous differentiation point. This is called the end effect and smoothing is not possible on the right side. The shape of the derivative curve can be distorted by this effect. Bourdet[32] suggests a solution in which we add a pseudo point to the right and fix it, such that the difference ΔX between the pseudo point and the point before it, is greater than this length or at least equal to it.

A Python code is written that reads the pressure and time data and uses data points that are separated by a proportion of a log cycle while calculating the derivative values. New derivative values are calculated with 0.10 to 0.50 of a log cycle with increments of 0.01. In total, 41 different derivative values with different separation intervals are calculated. The results for 0.10 log cycle interval, 0.50 log cycle interval, and the average of all 41 different derivative values can be seen from the figures below.

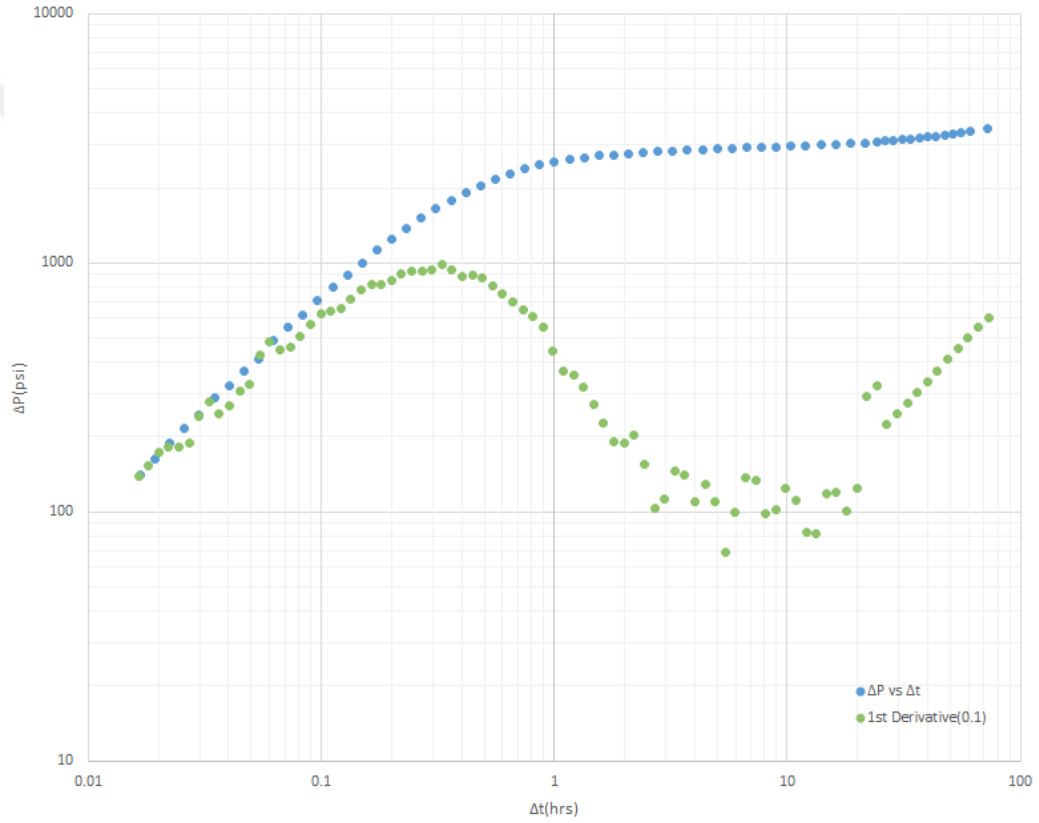


Figure 4.3 Derivative Plot Formed by Using Eq 4.3 (0.10 Log Cycle Interval)

It is quite obvious that the derivative plot is still too noisy, so using 0.10 log cycle intervals didn't make that much of a difference.

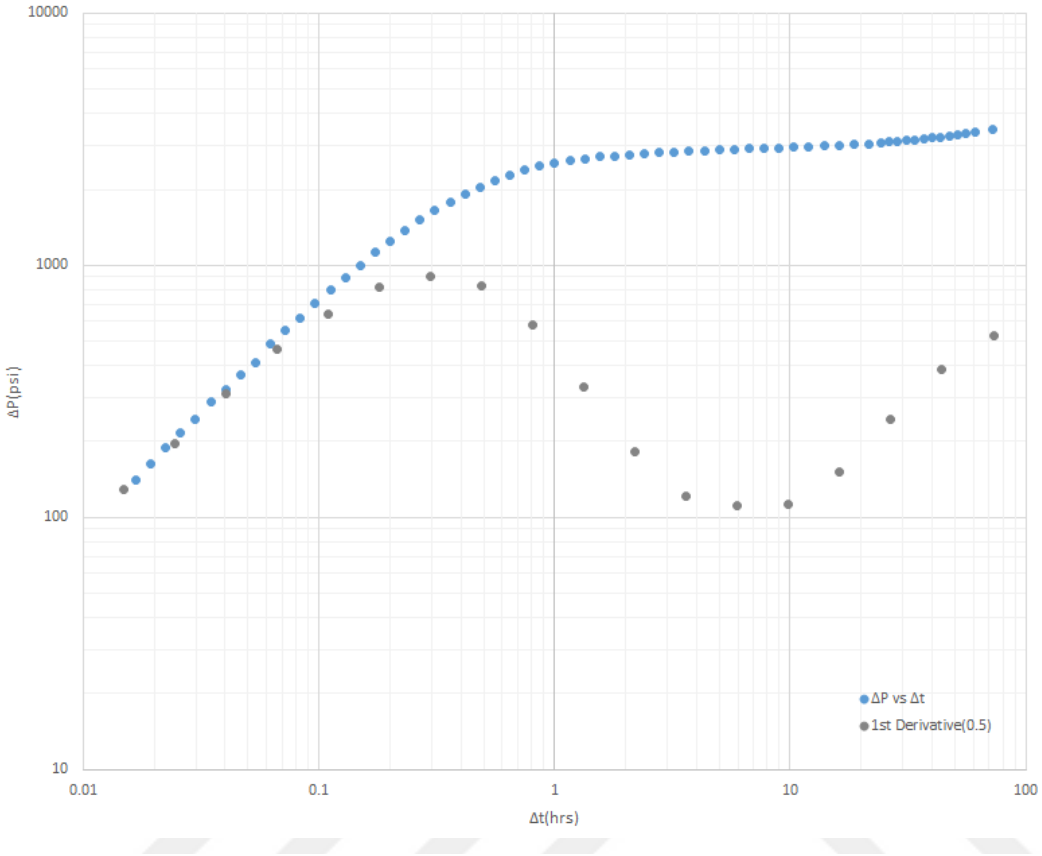


Figure 4.4 Derivative Plot Formed by Using Eq 4.3 (0.50 log cycle interval)

When 0.5 log cycle intervals are used, much smoother derivative plot is obtained but this time overly smoothing can be an issue. Also notice that there are very few points compared to the 0.1 log cycle interval case. As the log cycle interval increases, number of calculated derivative values decreases, hence 0.50 log cycle has the lowest amount of data points among the all 41.

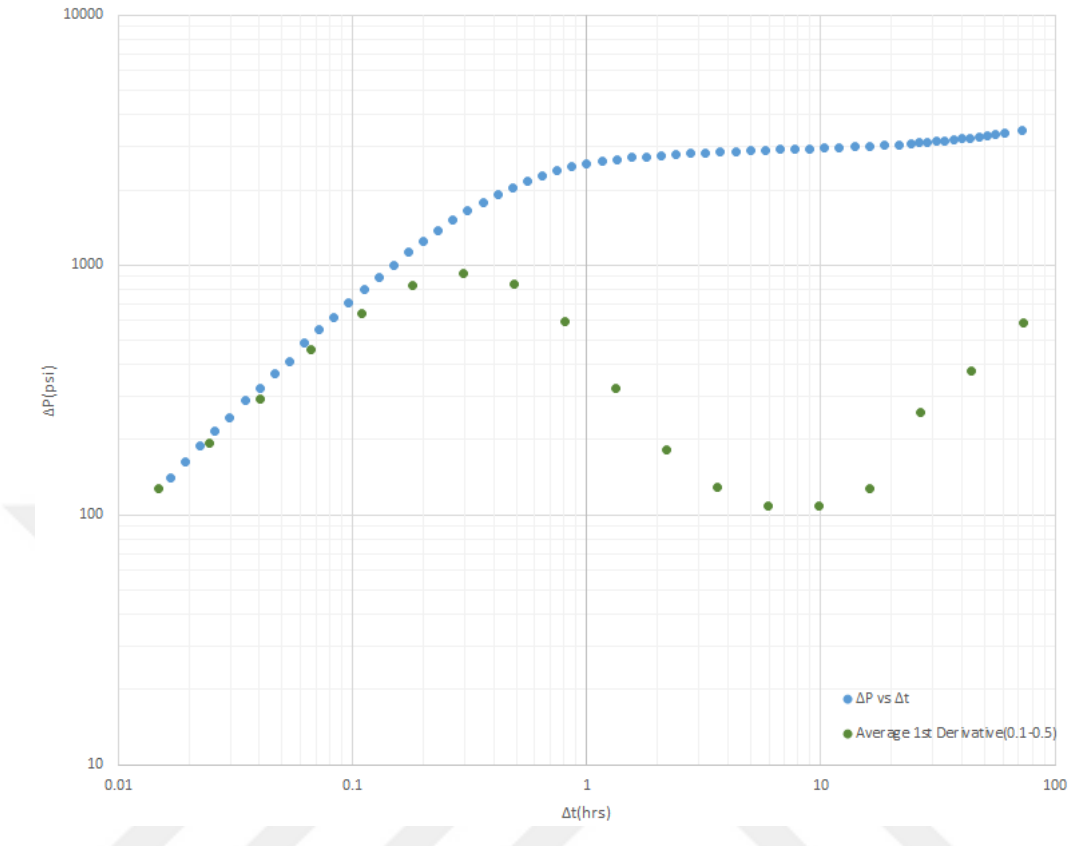


Figure 4.5 Derivative Plot Formed by Using Eq 4.3 (Average 0.10-0.50)

The average first derivative plot looks a lot like the one in Figure 4.4, but with slight differences. It contains the same amount of points as the 0.50 log cycle case since it had the lowest number of data points.

4.1.1.1 Selecting the Smoothest Derivative Plot for Drawdown Case I

The new aim is to find the best log cycle interval among the 41 different intervals(between 0.10-0.50), the one that eliminates the most amount of noise without overly smoothing the derivative plot. In order to be able to do that Python is used again.

Second and third derivative values are calculated for every one of the 41 different derivative values. Number of sign changes in the second and third derivatives are calculated separately and divided to the total number of data points.

For example when using 0.26 log cycle intervals,

Number of sign changes is 2 for the second derivative values and 8 for the third derivative values, meaning that there are 8 different inflection points. There are 53 data points in total and the ratio for the second derivative is:

$$\frac{Number\ of\ sign\ changes}{Number\ of\ data\ points} = \frac{2}{53} = 0.03774$$

And the ratio for the third derivative is:

$$\frac{8}{53} = 0.15094$$

It has the lowest ratio among the all 41 second derivative values meaning that it is the smoothest of them. The first and second derivative plots combined can be seen from the figure below.

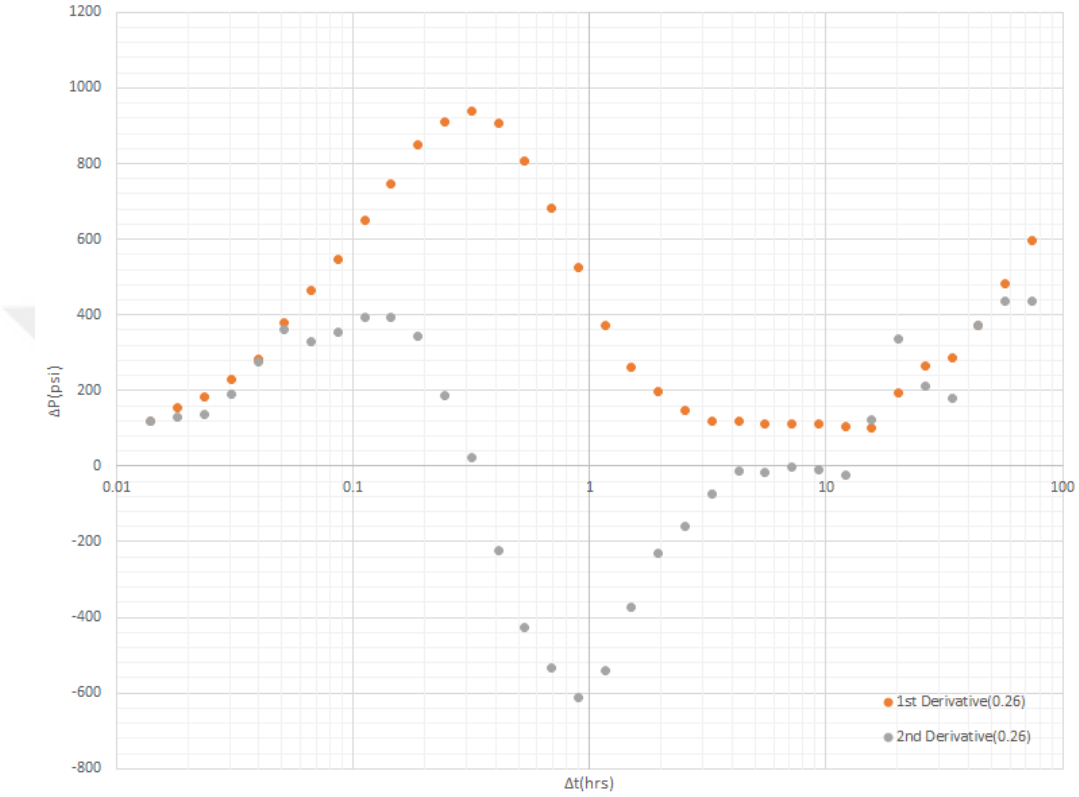


Figure 4.6 1st and 2nd Derivative Plots(0.26 Log Cycle Interval)

Please notice that this is a semi-log plot, because there are negative values in the second derivative and log-log plots can only contain positive values. The first and second derivatives both look smooth enough. Some noise can be seen in the second derivative values at the early and late times.

On the other hand, the ratio for the third derivative is quite high and can be seen from the figure below.

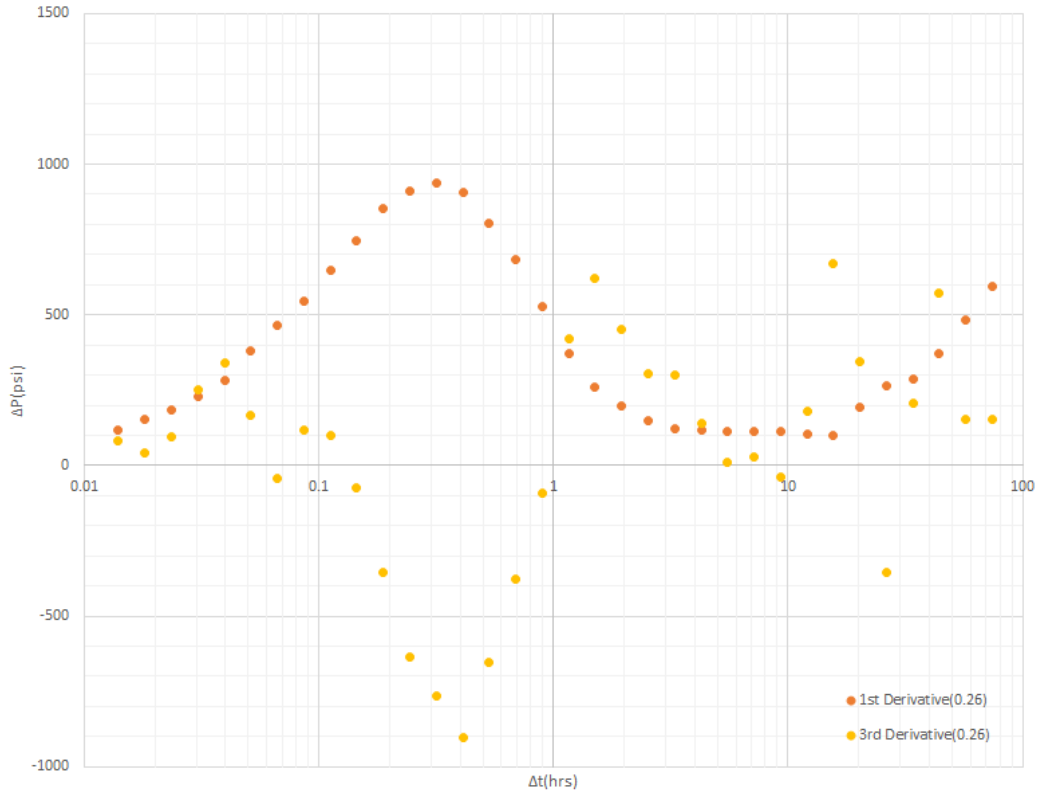


Figure 4.7 1st and 3rd Derivative Plots(0.26 Log Cycle Interval)

It is quite clear from the Figure 4.7 that third derivative values contain high amount of noise, as expected from the high ratio.

For the 0.31 log cycle interval case, numbers of sign changes are 2 for both of the second and third derivatives. The total number of data points is 45 so the ratio for the second and third derivatives is 0.04444, it is the interval that has the lowest ratio combined, which might mean that it is the best interval to use for derivative plot. The resulting first, second and third derivatives can be seen from the figures below.

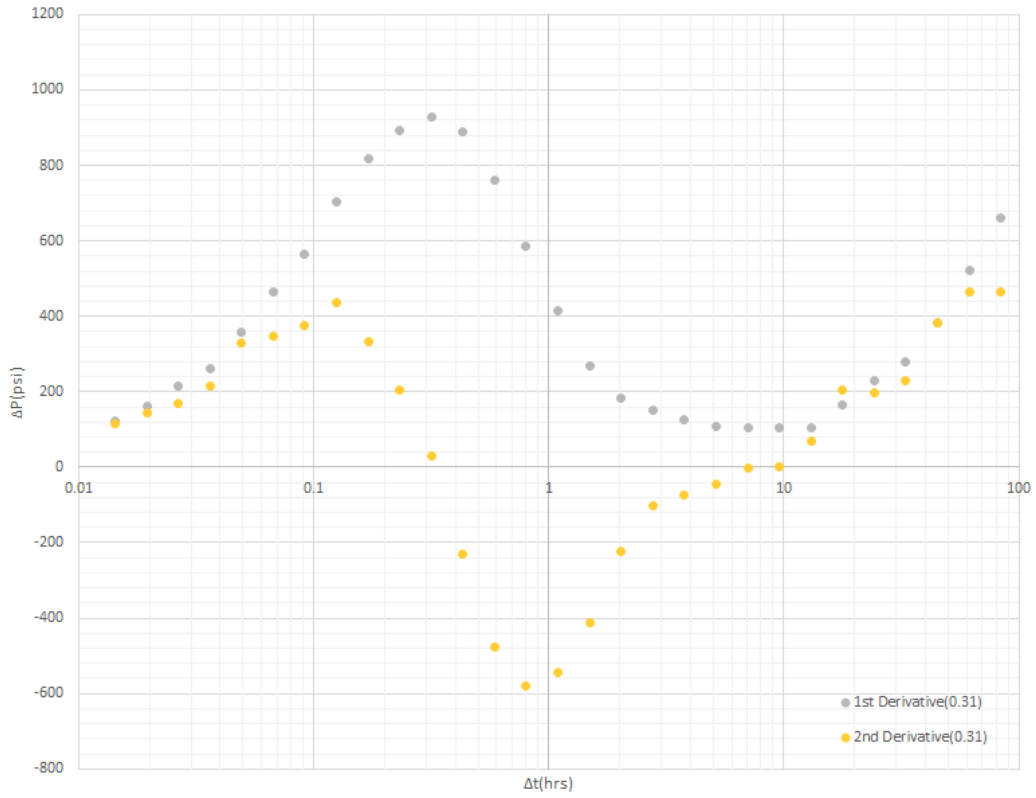


Figure 4.8 1st and 2nd Derivative Plots(0.31 Log Cycle Interval)

First and second derivatives look quite smooth, maybe smoother than the previous case.

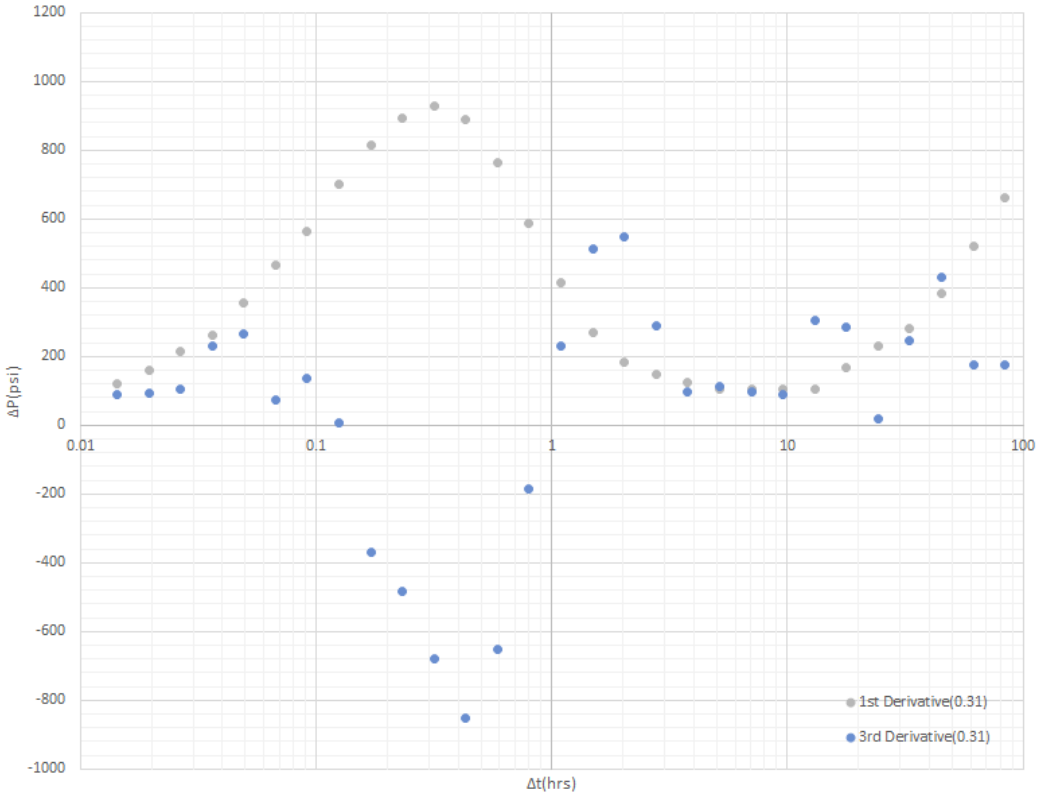


Figure 4.9 1st and 3rd Derivative Plots(0.31 Log Cycle Interval)

There are still some noise present in the third derivative but it is much smoother compared to the previous case.

Now that the possible best intervals are selected, the first, second and third derivatives will be compared.

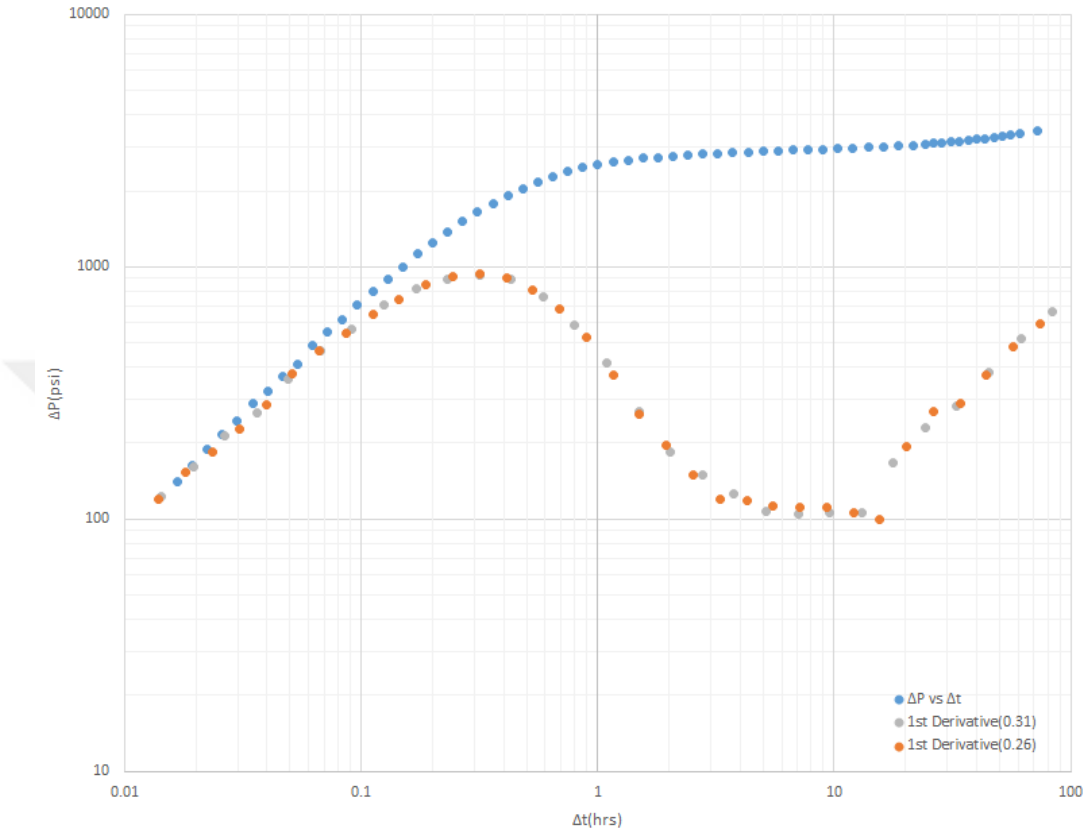


Figure 4.10 Comparison of First Derivatives of 0.26 and 0.31 Log Cycle Intervals

Although that there isn't much difference between the two, the one with 0.31 log cycle interval is slightly more smooth and seems to be the interval that gives the derivative plot with least amount of noise, without overly smoothing the curve.

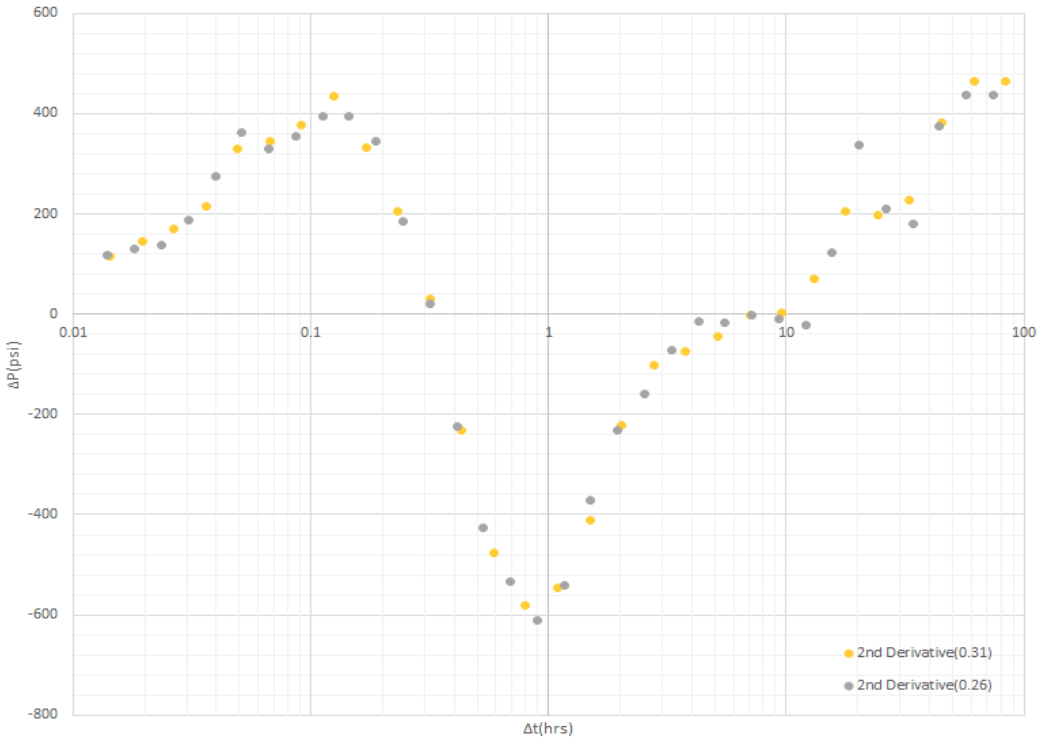


Figure 4.11 Comparison of Second Derivatives of 0.26 and 0.31 Log Cycle Intervals

Both intervals have two sign changes in their second derivatives. 0.31 interval results in smoother derivative curve at late times. 0.26 interval has a lower ratio due to having more number of data points.

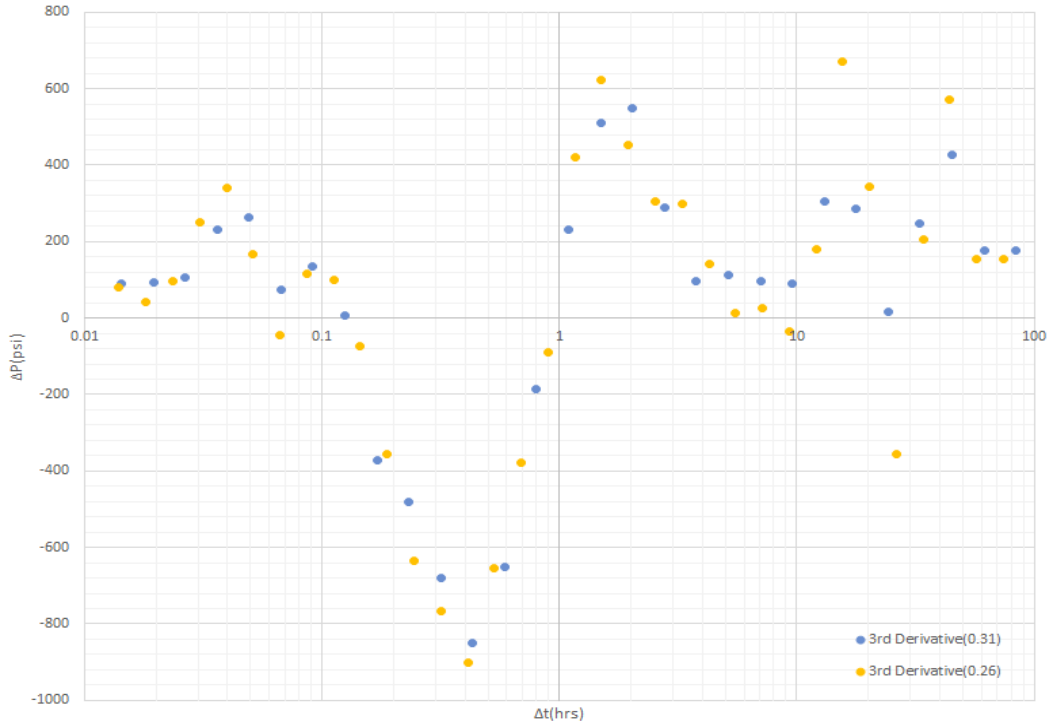


Figure 4.12 Comparison of Third Derivatives of 0.26 and 0.31 Log Cycle Intervals

0.31 log cycle interval has a lower ratio and results in a smoother third derivative curve as can be seen from Figure 4.12

Figure below shows the log cycle intervals and their sign change ratios for Case I.

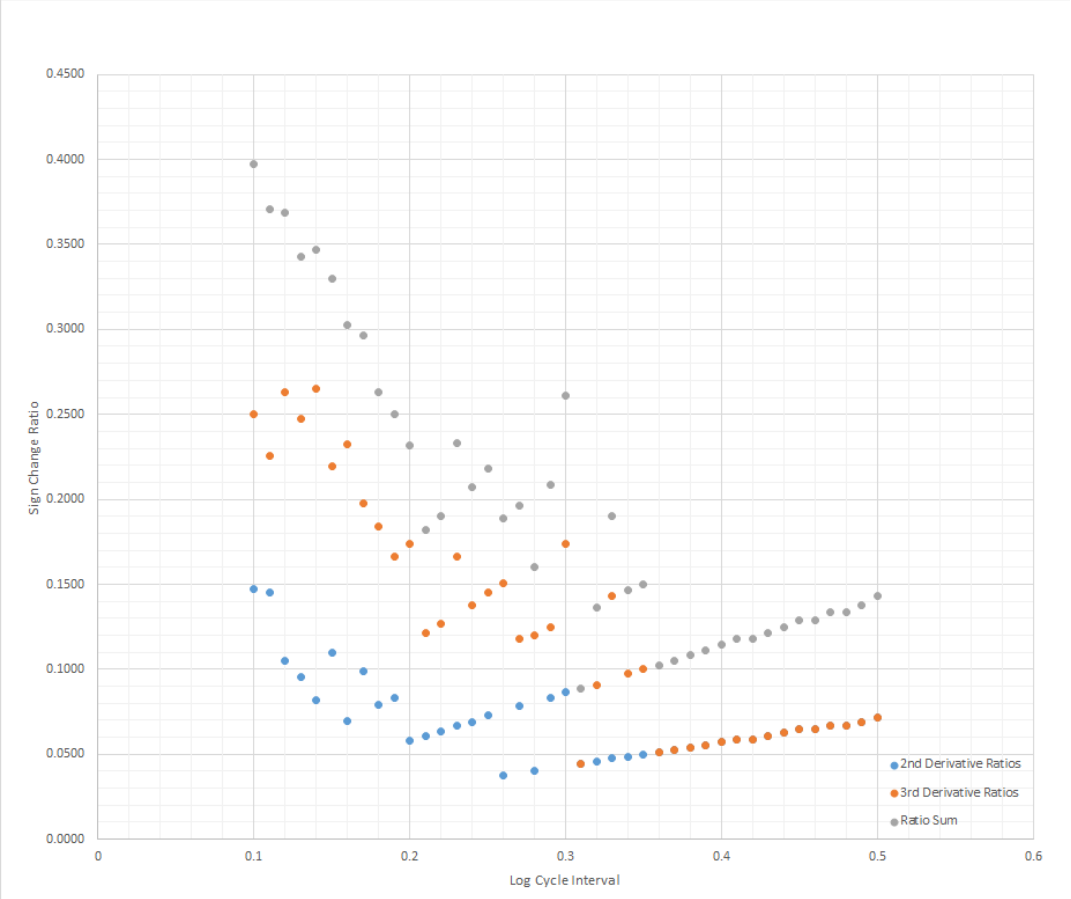


Figure 4.13 Log Cycle Interval vs Sign Change Ratio Plot for Case I

As can be seen from the Figure 4.13, lowest sign change ratio sum occurs at 0.31 interval, meaning that it plots the smoothest derivative curve.

4.1.1.2 Selecting the Smoothest Derivative Plot for Drawdown Case II

A drawdown test with a flowrate of 500 stb/day that lasts 327.6 hours is to be analyzed and flow periods are going to be estimated.

Table 4.2 Drawdown Test Data for Case II

t, hours	P_{wf}, psi	t, hours	P_{wf}, psi	t, hours	P_{wf}, psi	t, hours	P_{wf}, psi
0	3000	0.164	2693	3.28	1712	38.2	1533
0.0109	2976	0.218	2611	3.82	1696	43.7	1525
0.0164	2964	0.273	2536	4.37	1684	49.1	1517
0.0218	2953	0.328	2469	4.91	1674	54.6	1511
0.0273	2942	0.437	2352	5.46	1665	65.5	1500
0.0328	2930	0.491	2302	6.55	1651	87.4	1482
0.0382	2919	0.546	2256	8.74	1630	109.2	1468
0.0437	2908	1.09	1952	10.9	1617	163.8	1440
0.0491	2897	1.64	1828	16.4	1588	218.4	1416
0.0546	2886	2.18	1768	27.3	1554	273	1393
0.109	2785	2.73	1734	32.8	1543	327.6	1370

Same Python code and methodologies are used and pressure and time data is fed into the code. 0.32 and 0.40 log cycle intervals turn out to be the possible best intervals that can be used.

0.40 interval has the minimum sign change ratio for second derivatives with 0.05128 while 0.32 interval has a ratio of 0.08333. For the third derivatives the ratio for 0.32 interval is 0.08333 and 0.10256 for 0.40 interval.

The comparisons of first, second and third derivatives of both intervals can be seen from the figures below.

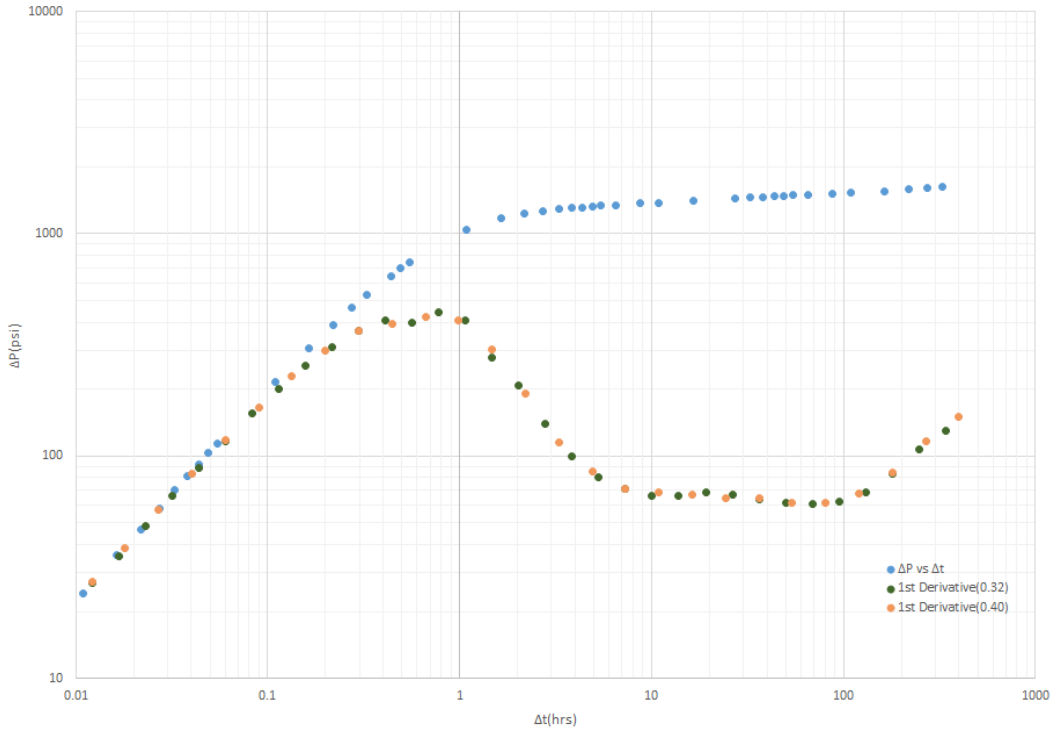


Figure 4.14 Comparison of the Derivative Plots of 0.32 and 0.40 Log Cycle Intervals

Their first derivatives look quite similar so the second and third derivatives will also be checked.

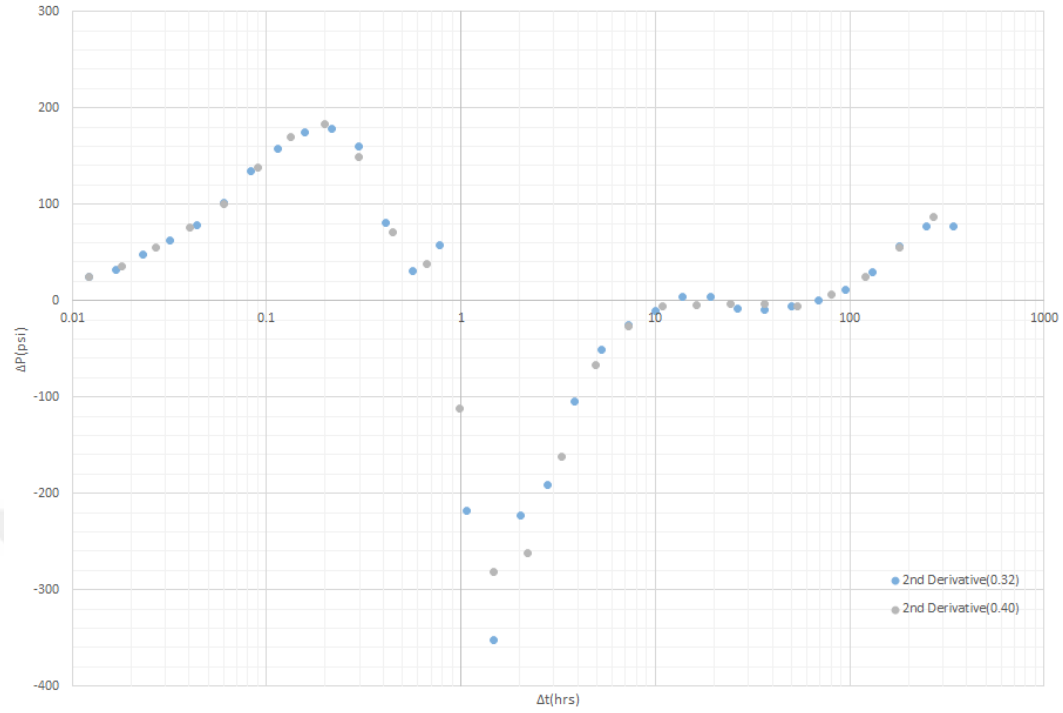


Figure 4.15 Comparison of the 2nd Derivatives for 0.32 and 0.40 Log Cycle Intervals

Although they look similar, 0.40 interval is slightly smoother. This was expected as it has less sign change ratio for second derivatives compared to the 0.32 interval.

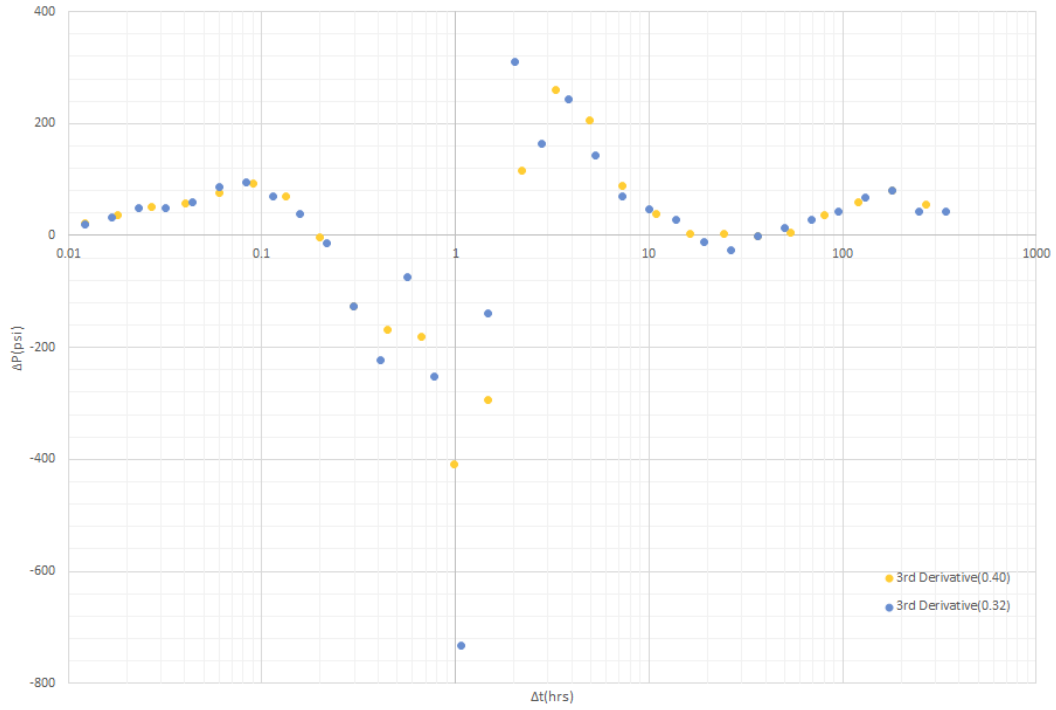


Figure 4.16 Comparison of the 3rd Derivatives for 0.32 and 0.40 Log Cycle Intervals

When you look at Figure 4.16, the smoother one seems to be the one calculated with 0.40 log cycle interval. This wasn't expected because 0.32 has a lower sign change ratio for third derivatives. There are 4 sign changes and hence 4 inflection points for both cases. The reason for 0.32 interval to have smaller sign change ratio is that it has more data points than 0.40 interval case. So the 0.40 interval is selected as the the interval that gives the derivative plot with least amount of noise, without overly smoothing the curve.

Figure below shows the log cycle intervals and their sign change ratios for Case II.

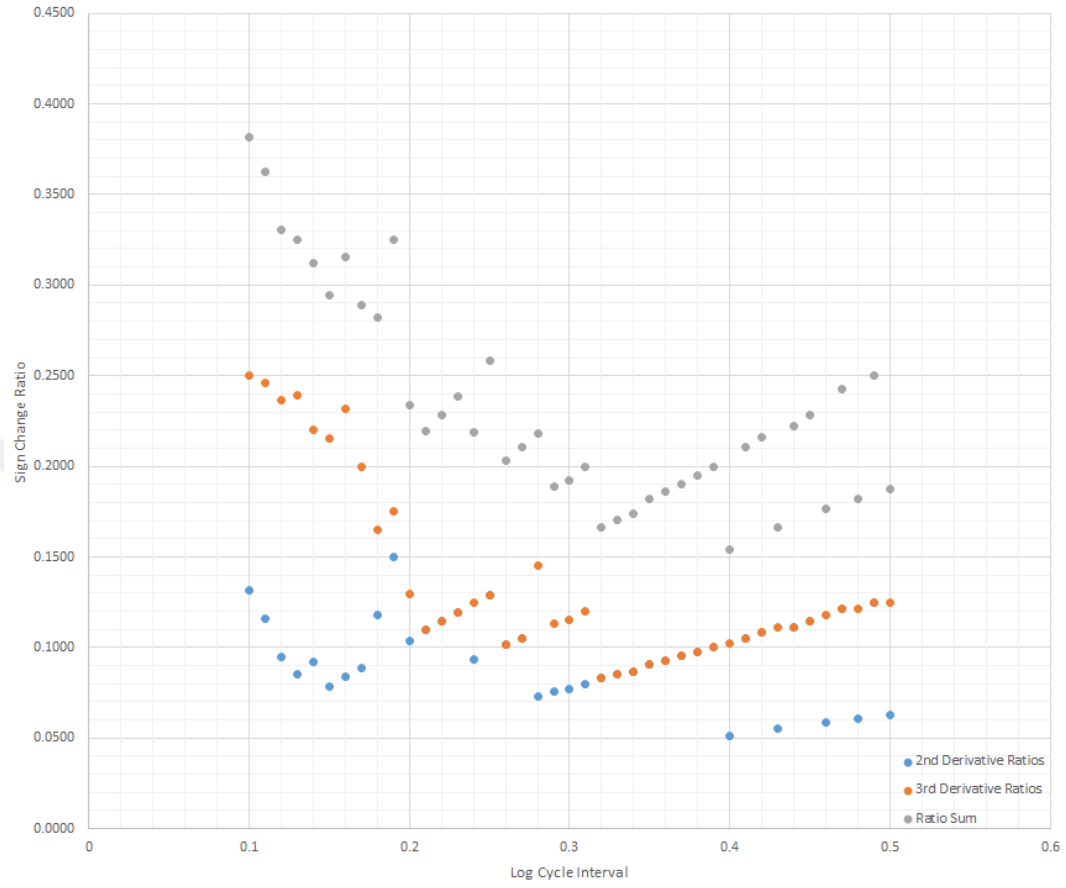


Figure 4.17 Log Cycle Interval vs Sign Change Ratio Plot for Case II

Please note that the best intervals for both drawdown cases are the ones that have the smallest sum of sign change ratios of second and third derivatives.

0.31 interval for the first drawdown case had the smallest sum of second and third derivative sign change ratios with $0.04444 + 0.04444 = 0.08888$

0.40 interval for the second drawdown case had the smallest sum of second and third derivative sign change ratios with $0.05128 + 0.10256 = 0.15384$.

4.2 Determination of the Middle Time Region

Now that we are able to identify and select the best log cycle interval that gives us the derivative plot with least amount of noise without overly smoothing the curve, we need to determine middle time regions because the most accurate estimates of permeability can be inferred from this middle time region[33]. Our ultimate goal is to estimate the permeability(k) and skin factor(S) as accurate as possible.

Infinite acting radial flow is present for both of the drawdown cases at the middle time regions, meaning that the derivative curve should be horizontal at middle time regions. In other words, first derivative values should stay fairly constant and second derivative values should be close to 0.

Python is used again to identify the middle time regions and in order to be able to do that all three of the first, second and third derivative values are checked.

4.2.1 Determination of the Middle Time Region for Drawdown Case I

0.31 log cycle interval was selected as the best interval and will be used in the determination of the middle time region. First things first, diagnostic plot is formed in Excel, from the output data of the Python code.

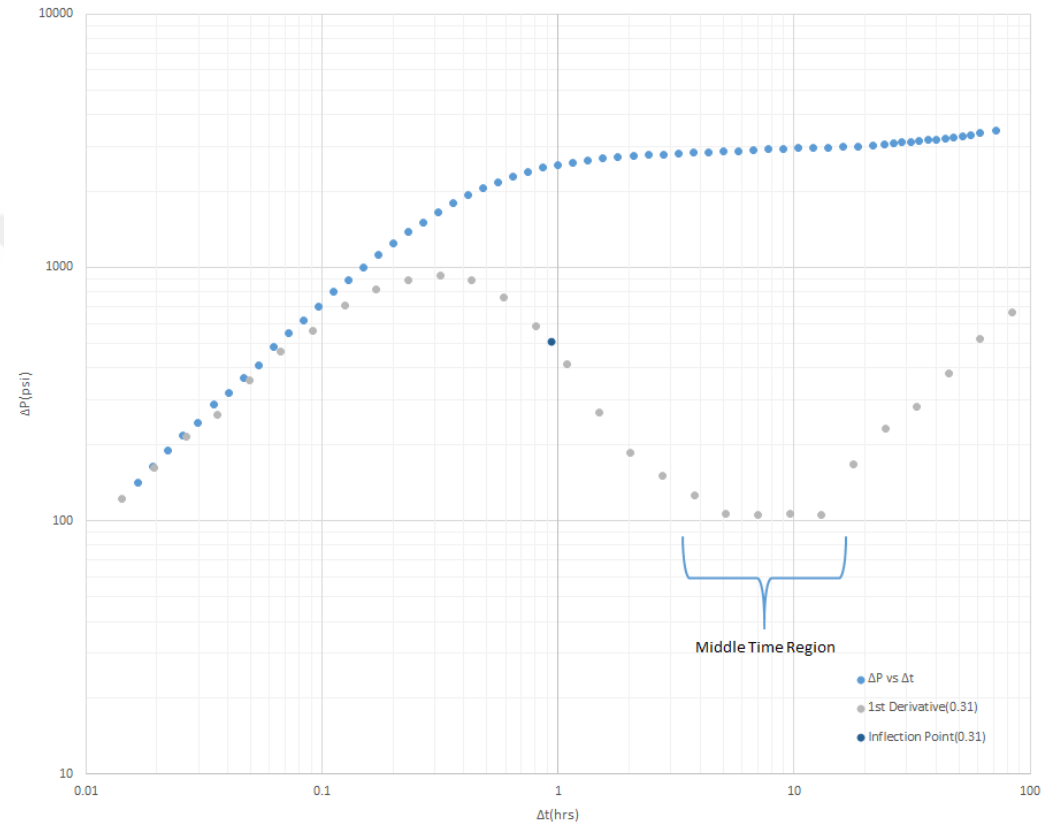


Figure 4.18 Eyeball Estimation of Middle Time Region for Case I

The middle time region is eyeball estimated by just looking at the diagnostic plot, probably with a low accuracy but this is more or less the region that we are trying to identify. We know that the middle time region comes after a specific inflection point(t = 0.933 hours, first derivative = 509.92), the dark blue dot that can be seen from the Figure 4.18.

Inflection points are the points that the third derivative values are 0. If we are able to identify that specific inflection point using Python, then we can get one step closer to correctly identifying the middle time region.

When using different log cycle intervals, the location of this specific inflection point and number of total inflection points changes. A method to identify the correct one is needed. First, second and third derivatives are plotted on the same plots for log cycle intervals of 0.31 and 0.26 in order to be able to find that method.

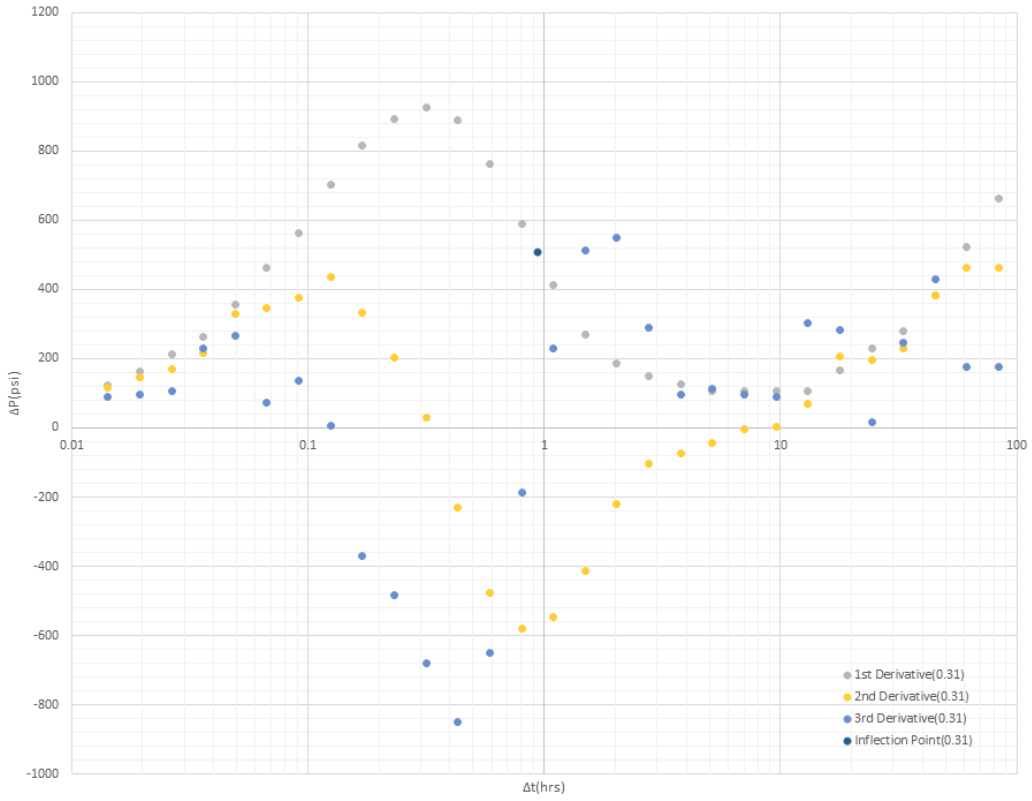


Figure 4.19 1st, 2nd and 3rd Derivative Plots (0.31 Log Cycle Interval)

That specific inflection point that we are after can be seen from the Figure 4.19. It is the dark blue point and lies at t = 0.933 hours, first derivative = 509.92. It is the second inflection point.

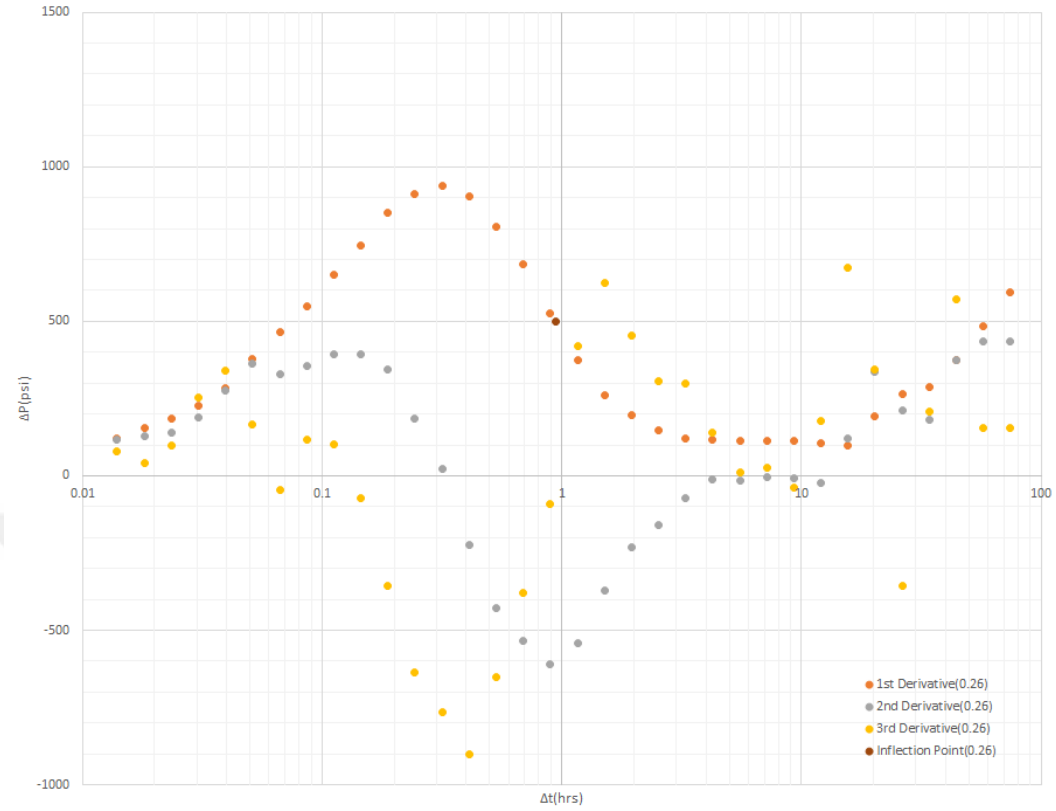


Figure 4.20 1st, 2nd and 3rd Derivative Plots (0.26 Log Cycle Interval)

That specific inflection point that we are after can be seen from the Figure 4.20. It is the brown point and lies at t = 0.943 hours, first derivative = 511.38. It is the fourth inflection point this time.

It caught my attention that both inflection points are the first inflection points that come after the minimum second derivative values. This was tested for the other 39 different differentiation intervals and they stand correct for each and every one of them.

So necessary additions to the Python code was made and it was able to identify these specific inflection points, the ones that are the first inflection points after the minimum value of second derivatives.

Now we have to find the region that comes after these specific inflection points previously found, where the first derivative values are fairly constant, also meaning that the second derivative values are close to 0.

This has been done easily by adding a constraint to the same Python code that selects points that come after the inflection point, where the second derivative values are in between -100 and +100.

The middle time region estimation result can be seen below.

Table 4.3 Middle Time Region Estimation For Drawdown Case I

Time(hours)	1st Derivative	2nd Derivative
3.780	126.21	-74.25
5.153	106.89	-44.05
7.026	104.97	-2.61
9.580	106.30	2.24
13.061	105.75	69.48

The infinite acting radial flow starts approximately at t = 3.78 hours and ends at t = 13.061 hours.

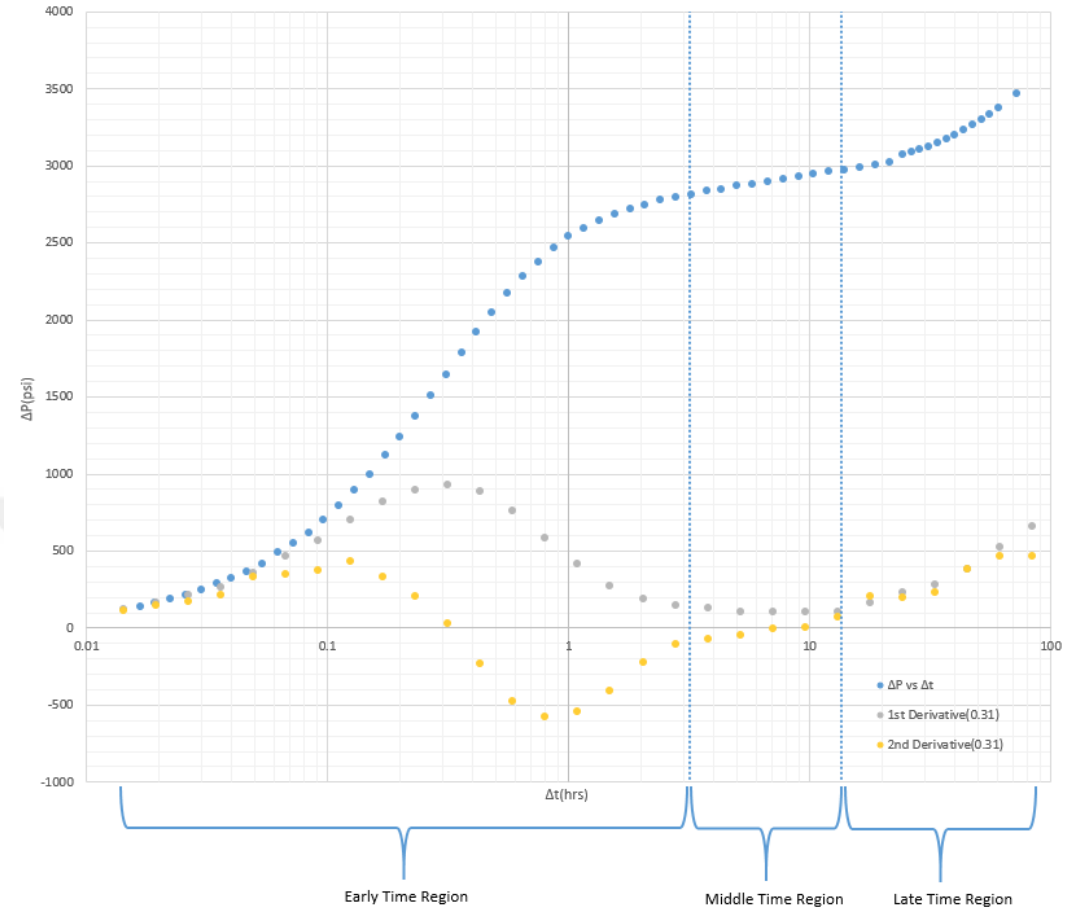


Figure 4.21 1st and 2nd Derivative Plots of Drawdown Case I with Early, Middle and Late Time Regions

So far the flow regions of early time, middle time and late time have been identified, as can be seen from the figure above. The semi-log plot shows clearly that the time region estimates are accurate for the Drawdown Case I.

At the early time region, wellbore and near wellbore effects such as wellbore storage, formation damage/stimulation etc. dominate the flow.

At the middle time region, the flow is expected to be infinite acting. If the reservoir is homogenous(it is in both cases), the pressure derivatives will be horizontal.

4.2.2 Determination of the Middle Time Region for Drawdown Case II

By feeding the pressure and time data of Drawdown Case II into the same Python code, time region estimations are as follows.

Table 4.4 Middle Time Region Estimation For Drawdown Case II

Time(hours)	1st Derivative	2nd Derivative
4.902	85.18	-66.35
7.313	71.50	-27.05
10.910	68.78	-6.46
16.275	66.91	-4.97
24.280	64.99	-3.68
36.222	64.76	-3.20
54.036	61.36	-5.97
80.613	61.61	5.71
120.260	68.05	23.58
179.407	83.84	55.46
267.645	116.67	86.40

The infinite acting radial flow starts approximately at t = 4.902 hours and ends at t = 267.645 hours. It lasts much more longer than the previous case, which could mean that the permeability is much less.

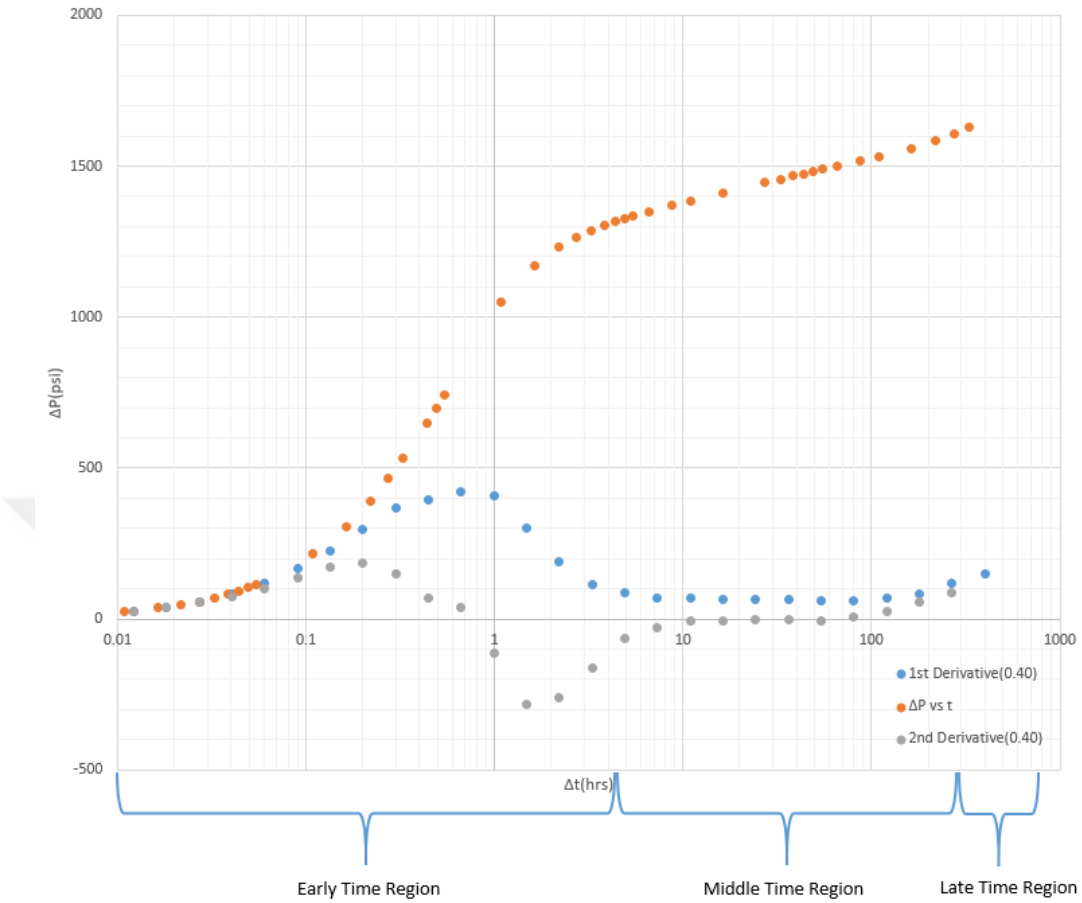


Figure 4.22 1st and 2nd Derivative Plots of Drawdown Case II with Early, Middle and Late Time Regions

The semilog plot above shows clearly that the time region estimates are also accurate for the drawdown case II. The Python code written can select the best derivation interval and correctly identify the infinite acting radial flow region.

4.3 Permeability and Skin Factor Estimation with Iterative Approach

In order to estimate permeability and skin factor, a MATLAB code is written. It reads the pressure and time data from Excel and uses data that lies in the middle time region(infinite acting radial flow) to estimate permeability and skin factor. Infinite acting radial flow equation is used in calculation process.

$$p_{wf} = p_i - \frac{q\mu}{4\pi kh}\left(\ln\frac{4kt}{\gamma\phi\mu c r_w^2} + 2S\right) \qquad \text{(Eq. 4.4)}$$

or in oilfield units;

$$p_{wf} = p_i - \frac{162.6q\mu B_0}{kh}\left(\log t + \log\frac{k}{\phi\mu c r_w^2} - 3.23 + 0.87S\right) \qquad \text{(Eq. 4.5)}$$

The working principle of the code is that it calculates P_{wf} values with k values ranging from 0 to 1000 md and s values ranging from -15 to +15, with an incremention of 0.01 for each parameter, hence it is an iterative method. These ranges and incrementation values can be adjusted, but it should be noted that it will impact the running time of the code.

Since only the data that lie in the middle time region are needed for the estimation of both parameters, pressure readings that lie in between early time region and late time region are used.

The best match is obtained by selecting k and s values that give the minimum value of $\sum_{i=1}^{n}(P_{wf(measured)} - P_{wf(calculated)})^2$. Thanks to this method, human error factor is eliminated completely.

4.3.1 Drawdown Case I

Table 4.5 Known Parameters of Drawdown Case I

Parameters	
ϕ	0.21
h, ft	23
B_o, bbl/stb	1.21
μ, cp	0.92
r_w, ft	0.401
C_t, 1/psi	8.72×10^{-6}

The infinite acting radial flow period has been identified successfully for the case I and the values from Table 4.3 and Table 4.5 are used for permeability and skin factor estimation.

The k and s values that give the best match to the recorded pressure values in the middle time region are calculated, and in this case they are 75.58 md for permeability and +5.85 for skin factor.

Comparison of $P_{wf(calculated\ with\ k=75.58\ md\ and\ s=+5.85)}$ and $P_{wf(measured)}$ values for each data point can be seen below.

Table 4.6 Comparison of Pwf(calculated with k=75.58 md and s=+5.85) and Pwf(measured)

t(hours)	$P_{wf(measured)}$(psi)	$P_{wf(calculated)}$(psi)	Absolute Error(%)
3.23	3200.34	3195.44	0.153
3.74	3175.40	3178.91	0.111
4.32	3162.30	3162.38	0.003
5.00	3139.87	3145.85	0.190
5.79	3133.46	3129.32	0.132
6.71	3114.87	3112.79	0.067
7.76	3092.78	3096.26	0.112
8.98	3081.99	3079.73	0.073
10.40	3062.07	3063.19	0.037
12.03	3047.29	3046.66	0.021
Total Error			0.899

It is quite clear that pressure readings and calculated values are so close to each other. Highest amount of difference occurs at t=5.00 hours with 5.98 psi, an error equal to 0.19%. The results are plotted in MATLAB and Excel, as can be seen from Figure 4.19 and Figure 4.20.

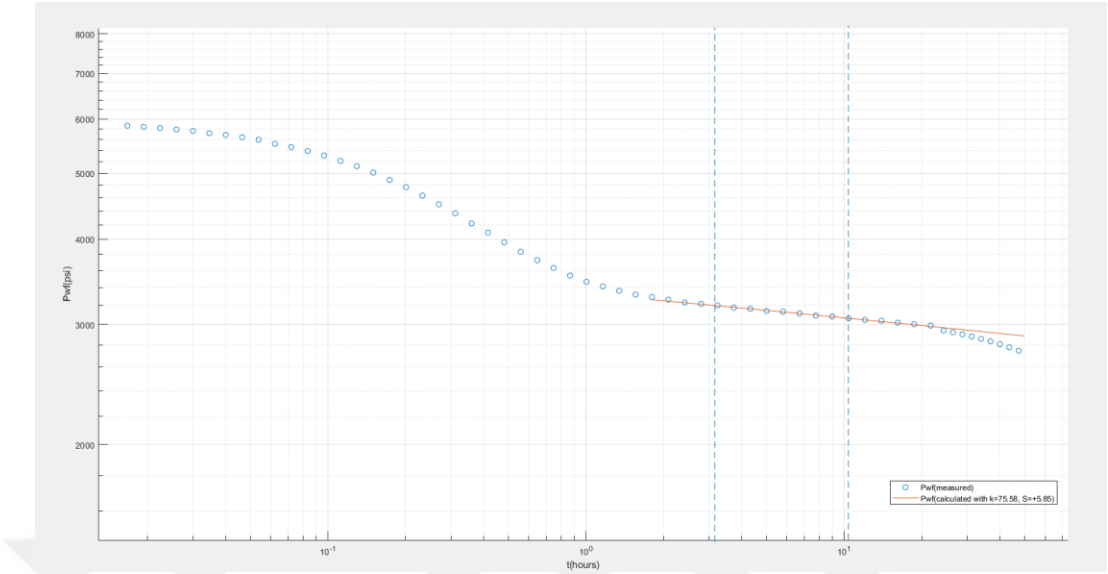


Figure 4.23 Semilog Plot of P_{wf} vs t, with Best Match Semilog Straight Line(k=75.58 md and s=+5.85)

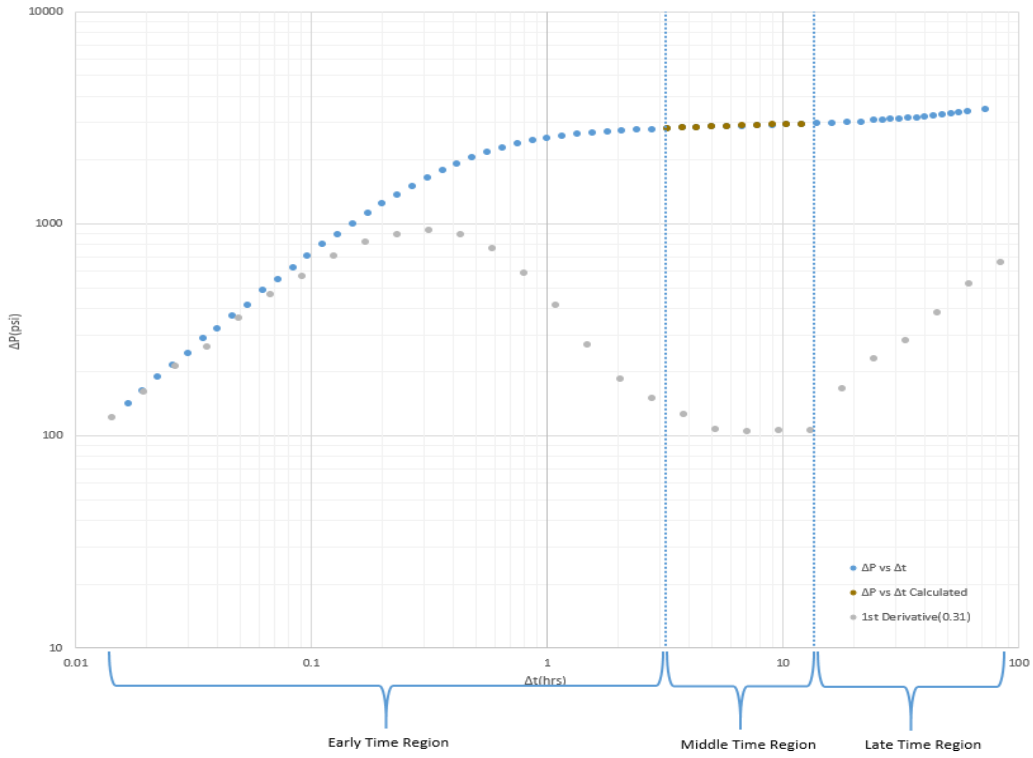


Figure 4.24 Comparison of ΔPwf(measured) and ΔPwf(calculated with k=75.58 md and s=+5.85)

4.3.2 Drawdown Case II

Table 4.7 Known Parameters of Drawdown Case II

Parameters	
ϕ	0.2
h, ft	56
B_o, bbl/stb	1.2
μ, cp	0.8
r_w, ft	0.3
C_t, 1/psi	$1x10^{-5}$

The infinite acting radial flow period has been identified successfully for the case II and the values from Table 4.4 and Table 4.7 are used for permeability and skin factor estimation.

The k and s values that give the best match to the recorded pressure values in the middle time region are calculated, and in this case they are 9.16 md for permeability and +4.00 for skin factor.

Table 4.8 Comparison of Pwf(calculated with k=9.16 md and s=+4.00) and Pwf(measured)

t(hours)	$P_{wf(measured)}$(psi)	$P_{wf(calculated)}$(psi)	Absolute Error(%)
4.91	1674.00	1669.96	0.241
5.46	1665.00	1662.94	0.124
6.55	1651.00	1650.92	0.005
8.74	1630.00	1631.86	0.114
10.90	1617.00	1617.26	0.016
16.40	1588.00	1590.27	0.143
27.30	1554.00	1556.59	0.167
32.80	1543.00	1544.47	0.095
38.20	1533.00	1534.39	0.091
43.70	1525.00	1525.51	0.033
49.10	1517.00	1517.81	0.053
54.60	1511.00	1510.79	0.014
65.50	1500.00	1498.76	0.082
87.40	1482.00	1479.70	0.155
109.20	1468.00	1464.99	0.205
163.80	1440.00	1438.20	0.125
218.40	1416.00	1419.19	0.225
Total Error			1.889

It is quite clear that pressure readings and calculated values are so close to each other. Highest amount of difference occurs at t=4.91 hours with 4.04 psi, an error less than 0.25%. The results are plotted in MATLAB and Excel, as can be seen from Figure 4.21 and Figure 4.22.

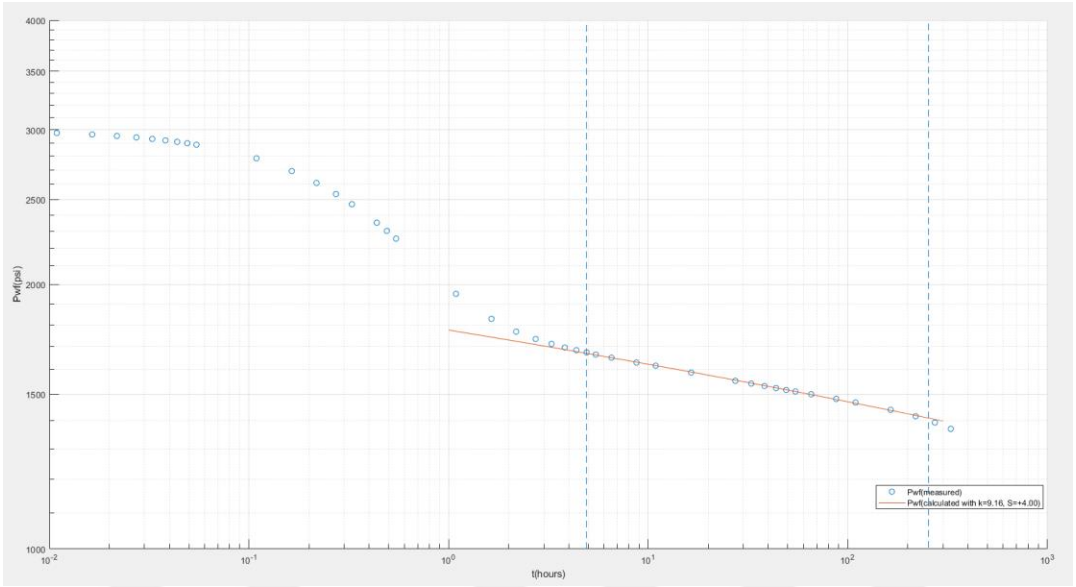


Figure 4.25 Semilog Plot of Pwf vs t, with Best Match Semilog Straight Line(k=9.16 md, s=+4.00)

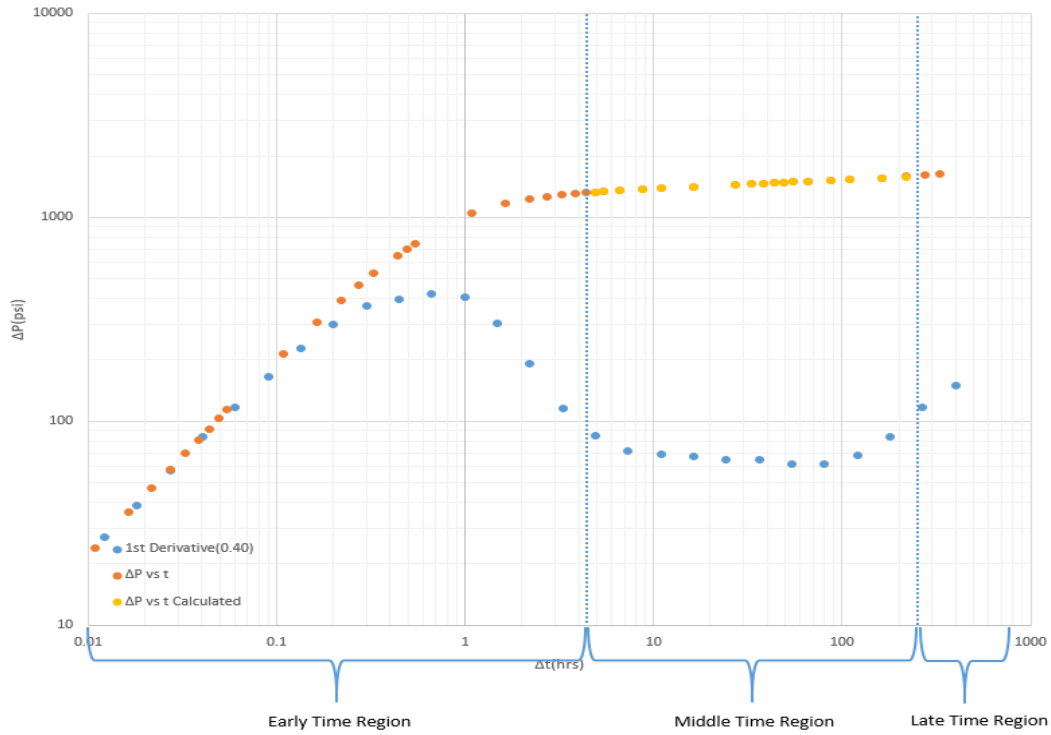


Figure 4.26 Comparison of ΔPwf(measured) and ΔPwf(calculated with k=9.16 md and s=+4.00)

4.4 Permeability and Skin Factor Estimation with Graphical Analysis

Graphical analysis method is much simpler and faster compared to the iterative method that has been used previously. Python is used again to estimate permeability and skin factor with graphical analysis method.

Semilog plot of P vs t is drawn using Python. A best line passing through the middle time region points is drawn. By calculating the slope of this line, which is called m, the permeability can be estimated from the equation below.

$$k = -162.6\frac{Q \times B \times \mu}{m \times h} \qquad \text{Eq 4.1}$$

In order to estimate the skin factor, S, following equation is used.

$$s = 1.151\left[\frac{P_i - P_{1hr}}{|m|} - \log\frac{k}{\phi\mu c_t r_w^2} + 3.2274\right] \qquad \text{Eq 4.2}$$

For the P_{1hr} value, the point on the semi-log straight line should be used rather than the measured P value at 1 hour.

4.4.1 Drawdown Case I

The values from Table 4.3 and Table 4.5 are used for the estimation process.

The equation of the best line is:

$$y = -256.932\,x + 3324.82$$

The permeability is calculated as 76.58 md using Eq 4.1 while the skin factor is calculated as +6.01 using Eq 4.2.

Comparison of $P_{wf(calculated\ with\ k=76.58\ md\ and\ s=+6.01)}$ and $P_{wf(measured)}$ values for each data point can be seen below.

Table 4.9 Comparison of Pwf(calculated with k=76.58 md and s=+6.01) and Pwf(measured)

t(hours)	$P_{wf(measured)}$(psi)	$P_{wf(calculated)}$(psi)	Absolute Error(%)
3.23	3200.34	3194.42	0.185
3.74	3175.40	3178.10	0.085
4.32	3162.30	3161.79	0.016
5.00	3139.87	3145.47	0.178
5.79	3133.46	3129.15	0.137
6.71	3114.87	3112.84	0.065
7.76	3092.78	3096.52	0.121
8.98	3081.99	3080.20	0.058
10.40	3062.07	3063.89	0.059
12.03	3047.29	3047.57	0.009
Total Error			0.915

Despite having a little bit more error than the iterative method, it is quite clear that pressure readings and calculated values are still close to each other. Highest amount of difference occurs at t=5.79 hours with 4.31 psi, an error less than 0.18%.

The results are plotted in Excel and Python, as can be seen from Figure 4.23 and Figure 4.24.

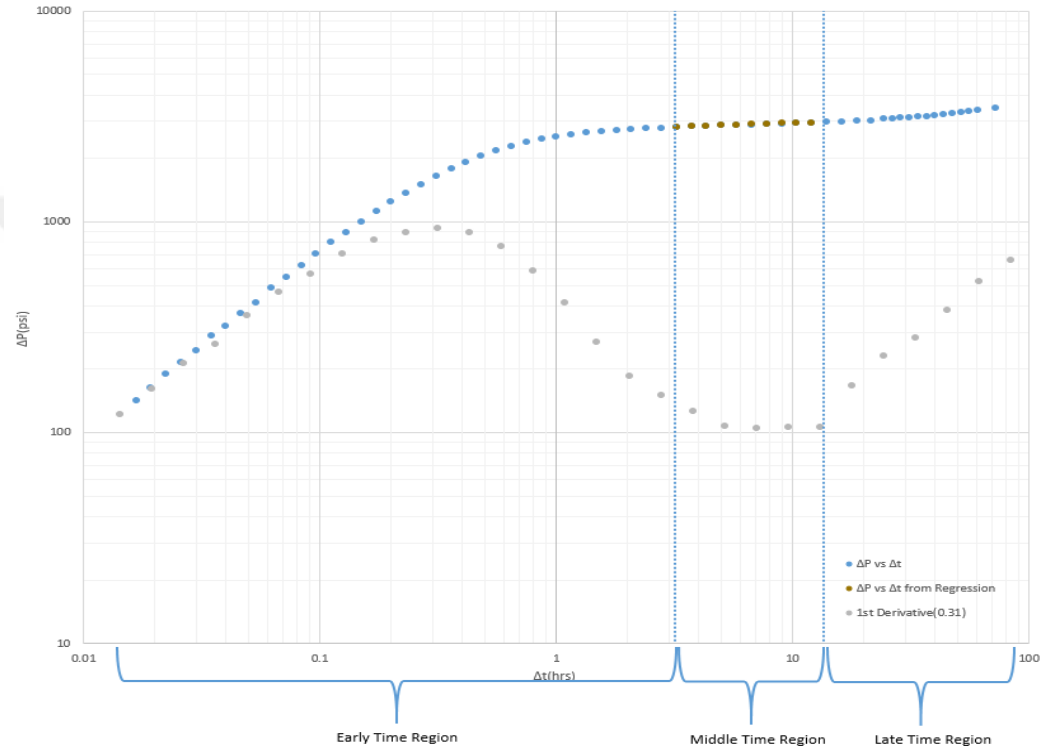


Figure 4.27 Comparison of ΔPwf(measured) and ΔPwf(calculated with k=76.58 md and s=+6.01)

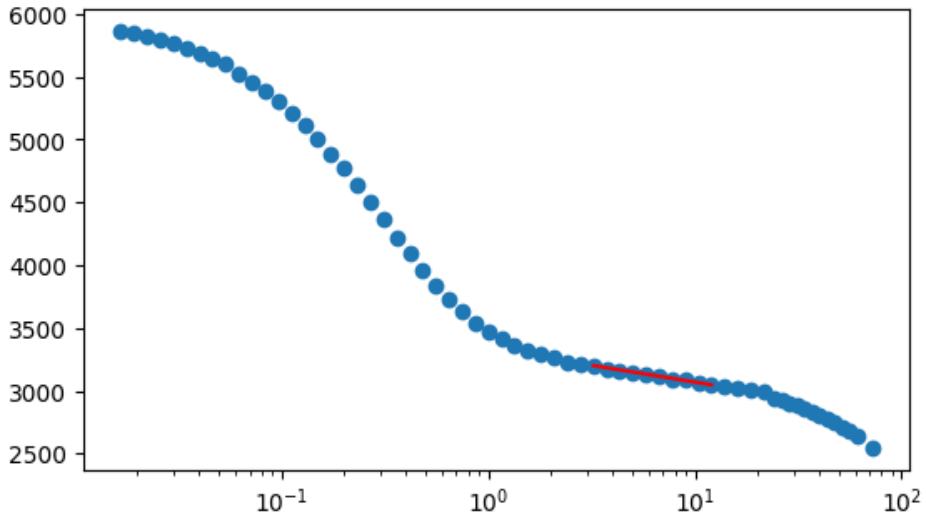


Figure 4.28 Semilog Plot of Pwf vs t, with Best Match Semilog Straight Line(k=76.58 md and s=+6.01)

4.4.2 Drawdown Case II

The values from Table 4.4 and Table 4.7 are used for the estimation process.

The equation of the best line is:

$$y = -152.529x + 1775.70$$

The permeability is calculated as 9.14 md using Eq 4.1 while the skin factor is calculated as +3.97 using Eq 4.2.

Table 4.10 Comparison of $P_{wf(calculated\ with\ k=9.14\ md\ and\ s=+3.97)}$ and $P_{wf(measured)}$

t(hours)	$P_{wf(measured)}$(psi)	$P_{wf(calculated)}$(psi)	Absolute Error(%)
4.91	1674.00	1670.43	0.213
5.46	1665.00	1663.39	0.096
6.55	1651.00	1651.34	0.020
8.74	1630.00	1632.23	0.137
10.90	1617.00	1617.60	0.037
16.40	1588.00	1590.54	0.160
27.30	1554.00	1556.78	0.179
32.80	1543.00	1544.62	0.105
38.20	1533.00	1534.53	0.100
43.70	1525.00	1525.62	0.040
49.10	1517.00	1517.90	0.059
54.60	1511.00	1510.87	0.009
65.50	1500.00	1498.81	0.079
87.40	1482.00	1479.70	0.155
109.20	1468.00	1464.95	0.208
163.80	1440.00	1438.09	0.133
218.40	1416.00	1419.03	0.214
Total Error			1.946

Despite having a little bit more error than the iterative method, it is quite clear that pressure readings and calculated values are still close to each other. Highest amount of difference occurs at t=4.91 hours with 3.57 psi, an error less than 0.22%.

The results are plotted in Excel and Python, as can be seen from Figure 4.25 and Figure 4.29.

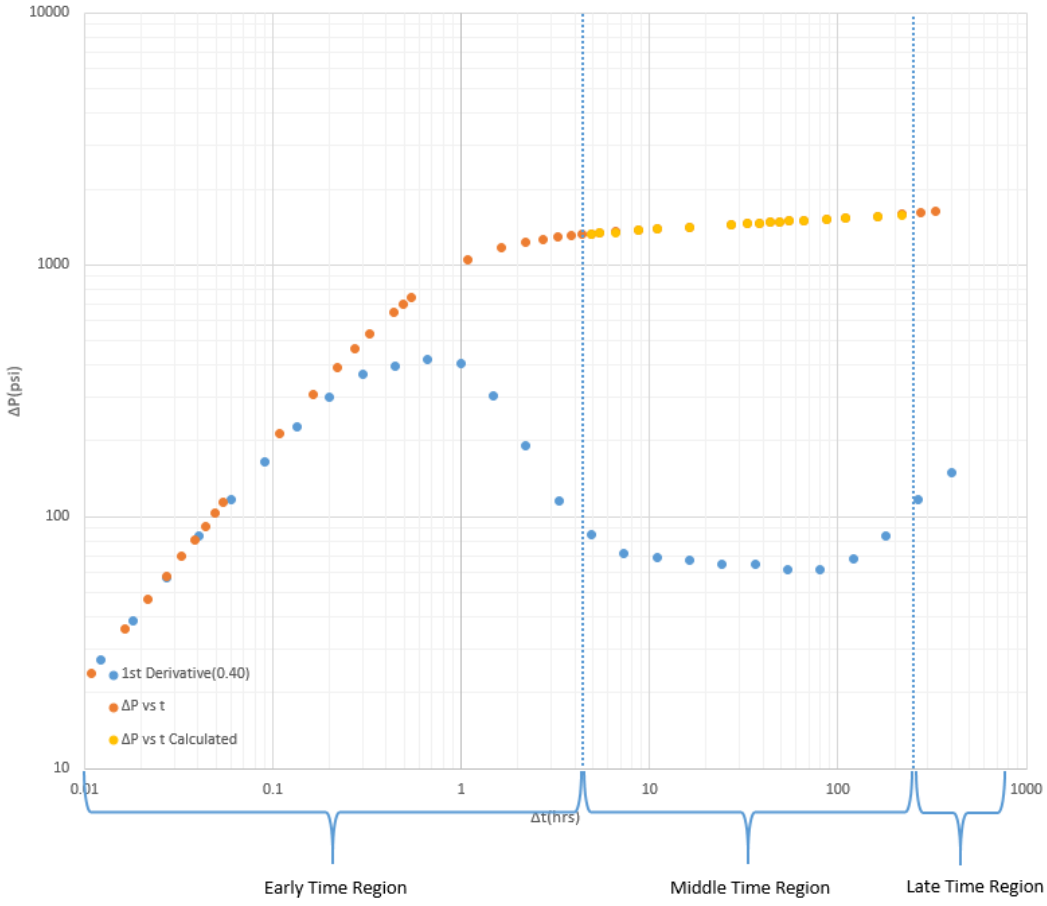


Figure 4.29 Comparison of ΔPwf(measured) and ΔPwf(calculated with k=9.14 md and s=+3.97)

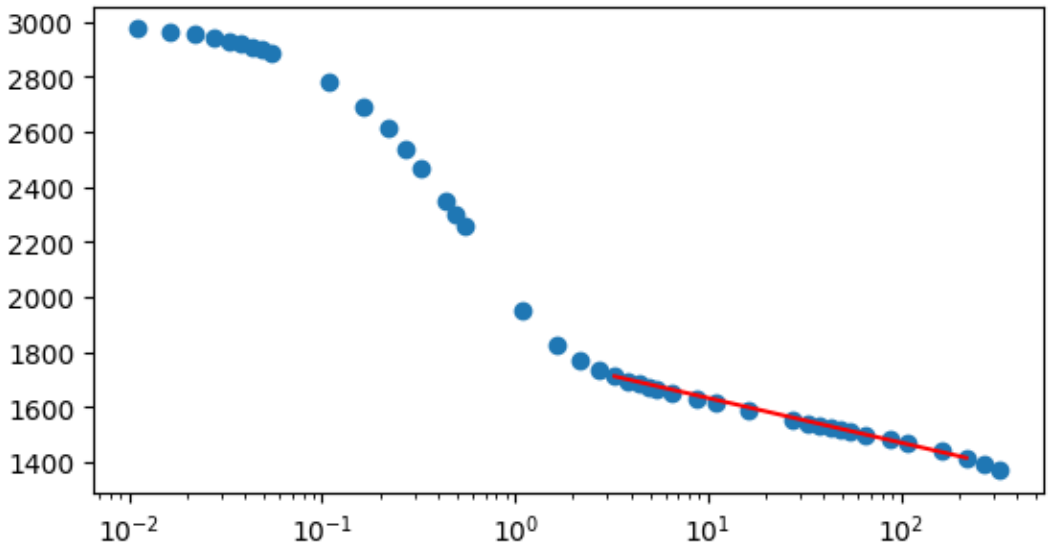


Figure 4.30 Semilog Plot of Pwf vs t, with Best Match Semilog Straight Line (k=9.14 md and s=+3.97)

4.5 Permeability and Skin Factor Estimation Using KAPPA Saphir Module

KAPPA's Saphir Module is an industry standard pressure transient analysis module, used by nearly all major international oil companies, national oil companies, independents and service companies.

It should be noted that Saphir doesn't find and use the best log cycle interval while calculating pressure derivatives. The default derivation interval is 0.1 of a log cycle and it is up to the user to find the appropriate interval with eyeball estimate.

Both cases are analyzed using this module to see the results and compare them with the previous results.

4.5.1 Drawdown Case I

Known reservoir parameters, flow rate, pressure and time data for drawdown case I was entered into KAPPA's Saphir Module as can be seen from the figure below.

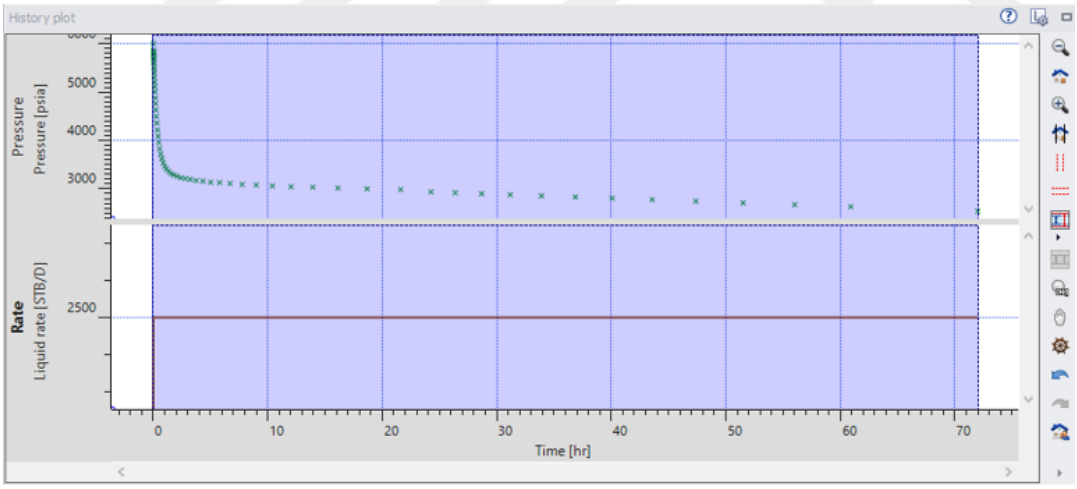


Figure 4.31 History Plot of P vs t and Q vs t for Drawdown Case I

Following derivative plot is drawn automatically by Saphir, with a default log cycle interval of 0.1.

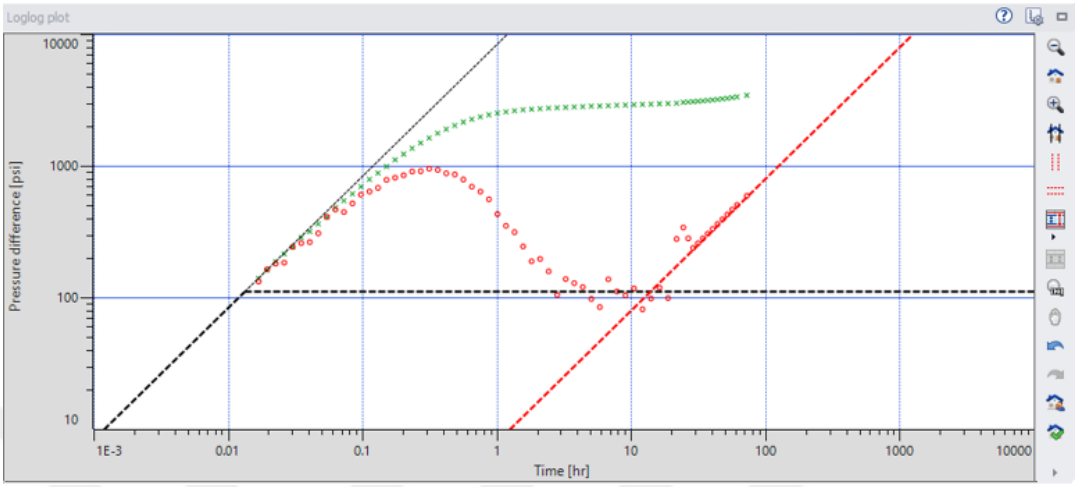


Figure 4.32 Derivative Plot Drawn by Saphir for Drawdown Case I (0.1 Log Cycle)

It is quite clear that the derivative plot is very noisy.

Following estimates of k and S had been made by Saphir.

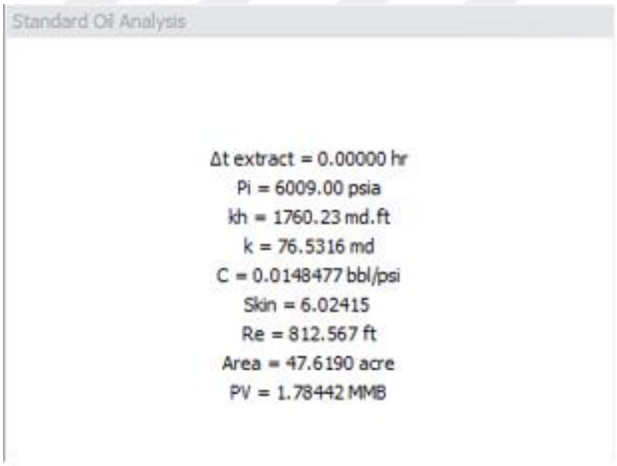


Figure 4.33 k and S Estimation Made by Saphir for Drawdown Case I(0.1 Log Cycle)

It estimated permeability as 76.5316 md and skin factor as +6.02415.

When the differentiation interval of 0.31 log cycle is used, following results are obtained.

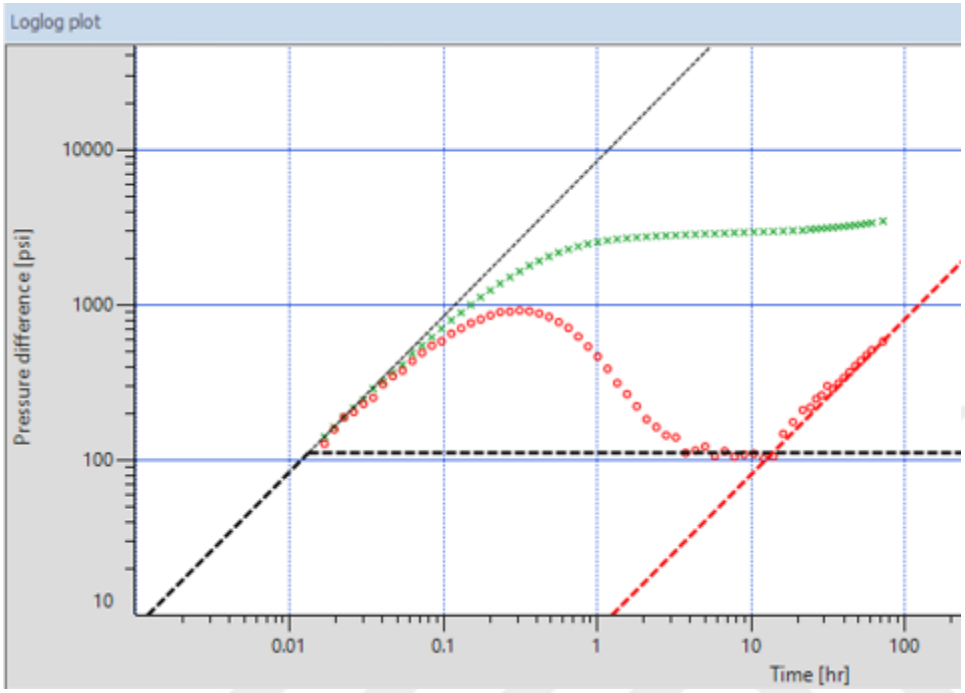


Figure 4.34 Derivative Plot Drawn by Saphir for Drawdown Case I (0.31 Log Cycle)

The derivative plot drawn with 0.31 log cycle interval is much more smooth compared to the Figure 4.29.

Following estimates of k and S had been made.

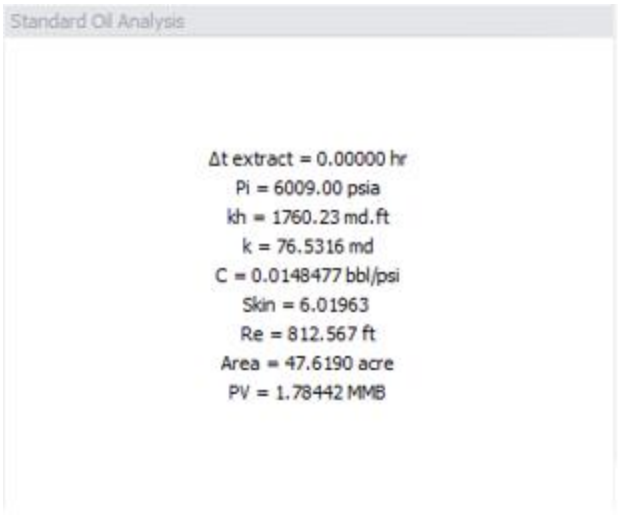


Figure 4.35 k and S Estimation Made by Saphir for Drawdown Case I(0.31 Log Cycle)

Even though the derivative plot is enhanced greatly by using 0.31 log cycle interval instead of 0.1, the estimated permeability stayed the same while the skin factor only changed from +6.02415 to +6.01963.

Comparison of $P_{wf(\text{calculated with } k=76.5316 \text{ md and } s=+6.01963)}$ and $P_{wf(\text{measured})}$ values for each data point can be seen from Table 4.11 on the next page.

Table 4.11 Comparison of Pwf(calculated with k=76.5316 md and s=+6.01963) and Pwf(measured)

t(hours)	$P_{wf(measured)}$(psi)	$P_{wf(calculated)}$(psi)	Absolute Error(%)
3.23	3200.34	3191.15	0.287
3.74	3175.40	3174.83	0.018
4.32	3162.30	3158.50	0.120
5.00	3139.87	3142.18	0.073
5.79	3133.46	3125.85	0.243
6.71	3114.87	3109.52	0.172
7.76	3092.78	3093.20	0.014
8.98	3081.99	3076.87	0.166
10.40	3062.07	3060.55	0.050
12.03	3047.29	3044.22	0.101
Total Error			1.243

Total error for the drawdown case I was 0.899% for the iterative method and 0.915% for the graphical analysis method. Saphir has a total error of 1.243%, which is larger than both methods, meaning that it is less accurate.

4.5.2 Drawdown Case II

Known reservoir parameters, flow rate, pressure and time data for drawdown case II was entered into KAPPA's Saphir Module as can be seen from the figure below.

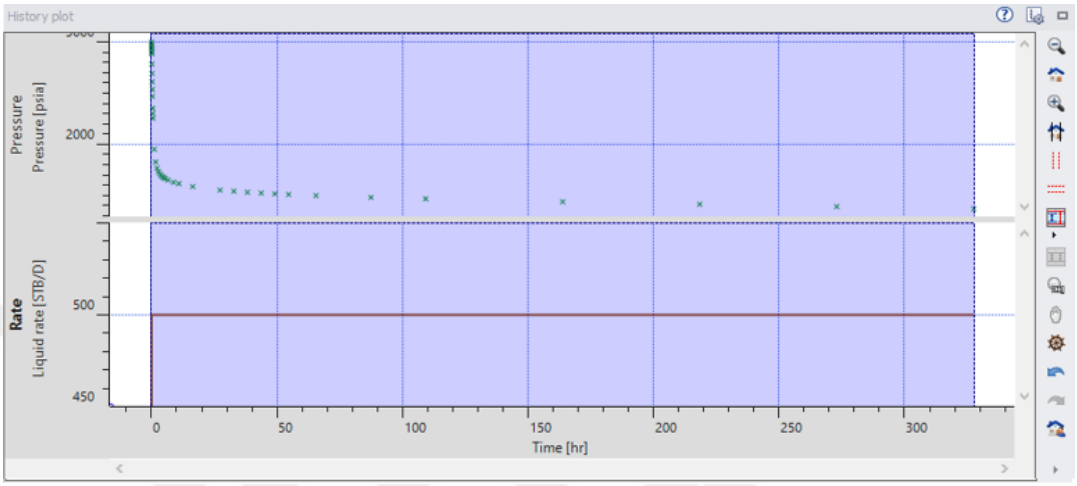


Figure 4.36 History Plot of P vs t and Q vs t for Drawdown Case II

Following derivative plot is drawn automatically by Saphir, with a default log cycle interval of 0.1.

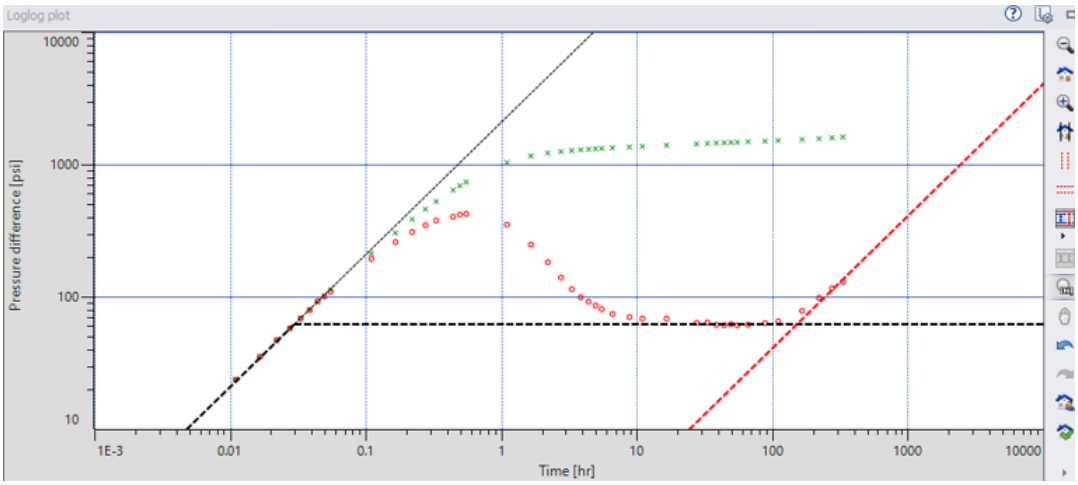


Figure 4.37 Derivative Plot Drawn by Saphir for Drawdown Case II (0.1 Log Cycle)

Following estimates of k and S had been made by Saphir.



Figure 4.38 k and S Estimation Made by Saphir for Drawdown Case II(0.1 Log Cycle)

It estimated permeability as 9.65699 md and skin factor as +4.58184.

When the differentiation interval of 0.40 log cycle is used, following results are obtained.

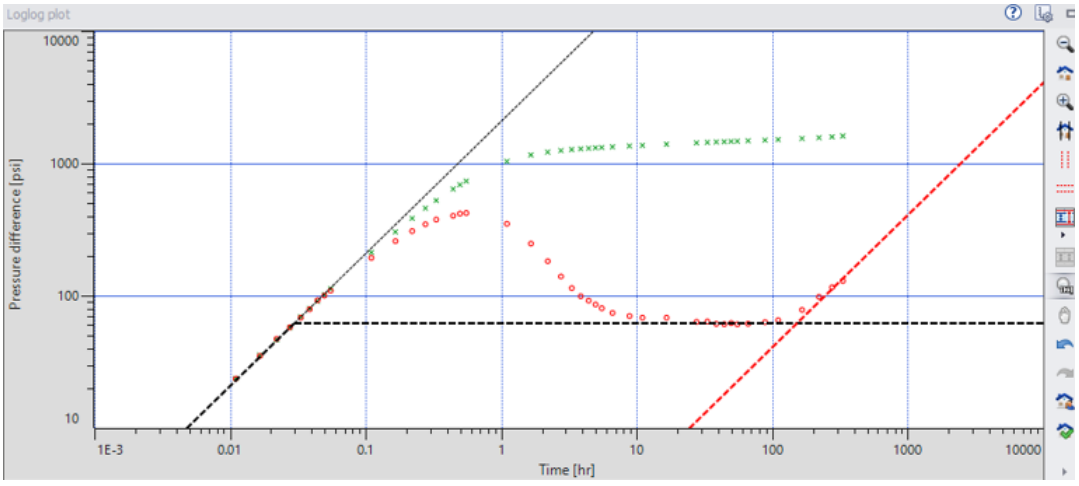


Figure 4.39 Derivative Plot Drawn by Saphir for Drawdown Case I (0.31 Log Cycle)

The derivative plot drawn with 0.40 log cycle interval is smoother compared to the one drawn with 0.10 log cycle interval.

Following estimates of k and S had been made.



Figure 4.40 k and S Estimation Made by Saphir for Drawdown Case I(0.31 Log Cycle)

Even though the derivative plot is enhanced again by using 0.40 log cycle interval instead of 0.1, the estimated permeability stayed the same while the skin factor only changed from +4.58184 to +4.59090.

Comparison of $P_{wf(\text{calculated with } k=9.65699 \text{ md and } s=+4.59090)}$ and $P_{wf(\text{measured})}$ values for each data point can be seen from Table 4.12 on the next page.

Table 4.12 Comparison of Pwf(calculated with k=9.65699 md and s=+4.59090) and Pwf(measured)

t(hours)	$P_{wf(measured)}$(psi)	$P_{wf(calculated)}$(psi)	Absolute Error(%)
4.91	1674.00	1661.02	0.775
5.46	1665.00	1654.37	0.639
6.55	1651.00	1642.96	0.487
8.74	1630.00	1624.88	0.314
10.90	1617.00	1611.04	0.369
16.40	1588.00	1585.43	0.162
27.30	1554.00	1553.49	0.033
32.80	1543.00	1541.99	0.066
38.20	1533.00	1532.44	0.037
43.70	1525.00	1524.00	0.065
49.10	1517.00	1516.70	0.020
54.60	1511.00	1510.05	0.063
65.50	1500.00	1498.64	0.091
87.40	1482.00	1480.56	0.097
109.20	1468.00	1466.60	0.095
163.80	1440.00	1441.19	0.082
218.40	1416.00	1423.16	0.505
Total Error			3.900

Total error for the Drawdown Case II was 1.889% for the iterative method and 1.946% for the graphical analysis method. Saphir has a total error of 3.90%, more than double the error of both methods, meaning that it is much less accurate.

4.6 Drawdown Case III

A drawdown test data without any noise have been simulated.

Table 4.13 Drawdown Test Data for Case III

t, hours	P_{wf}, psi	t, hours	P_{wf}, psi	t, hours	P_{wf}, psi	t, hours	P_{wf}, psi
0.000	5000.00	0.110	3905.00	1.000	2608.00	8.000	2316.56
0.010	4900.00	0.120	3810.00	1.200	2580.67	9.000	2300.17
0.015	4850.00	0.150	3580.00	1.400	2559.21	10.000	2285.50
0.020	4800.00	0.180	3360.00	1.600	2540.62	12.000	2255.00
0.025	4750.00	0.220	3170.00	1.800	2524.22	14.000	2224.50
0.030	4700.00	0.250	3050.00	2.000	2509.56	16.000	2194.00
0.040	4600.00	0.300	2940.00	2.500	2478.49	18.000	2163.50
0.050	4500.00	0.350	2850.00	3.000	2453.11	20.000	2133.00
0.060	4400.00	0.400	2800.00	3.500	2431.65	22.000	2102.50
0.070	4300.00	0.500	2740.00	4.000	2413.06	24.000	2072.00
0.080	4200.00	0.600	2692.00	5.000	2381.99	26.000	2041.50
0.090	4100.00	0.700	2665.00	6.000	2356.61		
0.100	4000.00	0.800	2643.00	7.000	2335.15		

Table 4.14 Known Parameters of Drawdown Case III

Parameters	
Q, stb/day	1500
ϕ	0.2
h, ft	21
B_o, bbl/stb	1.2
μ, cp	0.92
r_w, ft	0.401
C_t, 1/psi	$9x10^{-6}$
k, md	4.00
S	+2.90

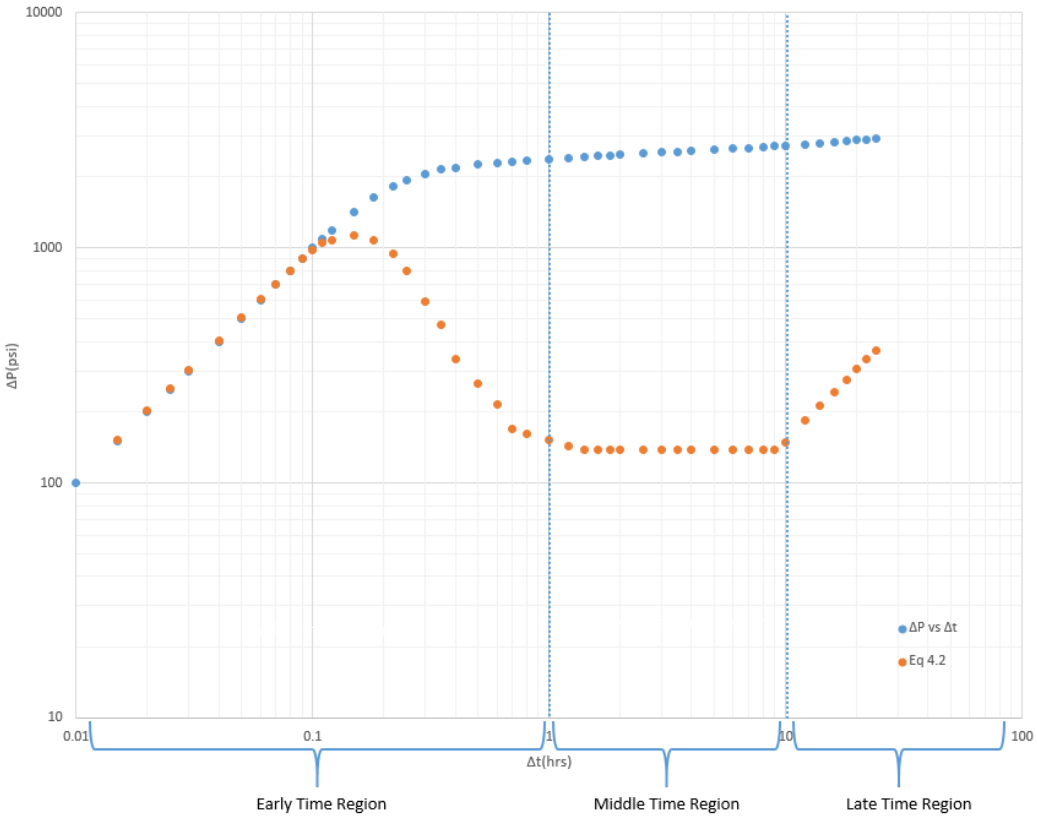


Figure 4.41 Diagnostic Plot of Drawdown Case III

It is known that the infinite acting radial flow starts at 1 hours and ends at 10 hours. Unlike the previous cases, using Eq 4.2 results in a perfectly smooth derivative curve as can be seen from Figure 4.41.

Pressure and time data is fed into same Python code and following results are obtained.

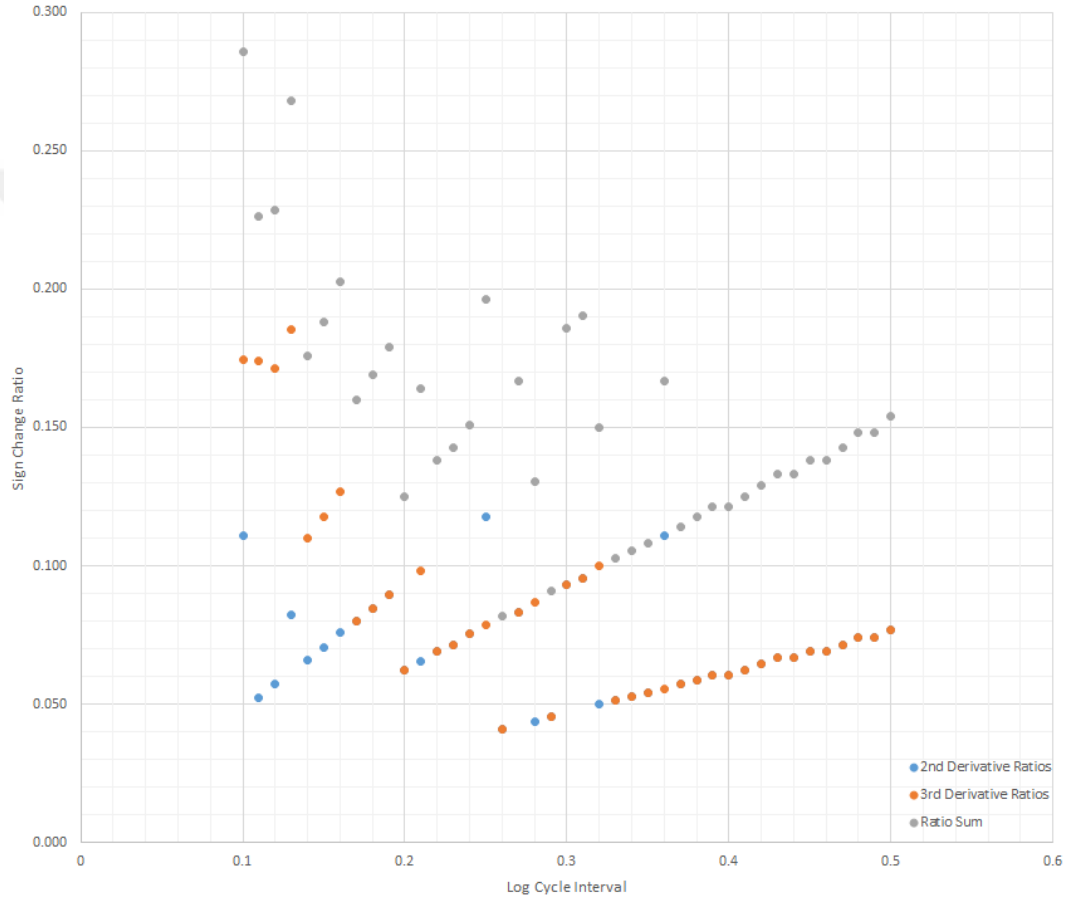


Figure 4.42 Log Cycle Interval vs Sign Change Ratio Plot for Case III

0.26 interval turns out to be the best interval, having smallest sum of second and third derivative sign change ratios with $0.04082 + 0.04082 = 0.08164$

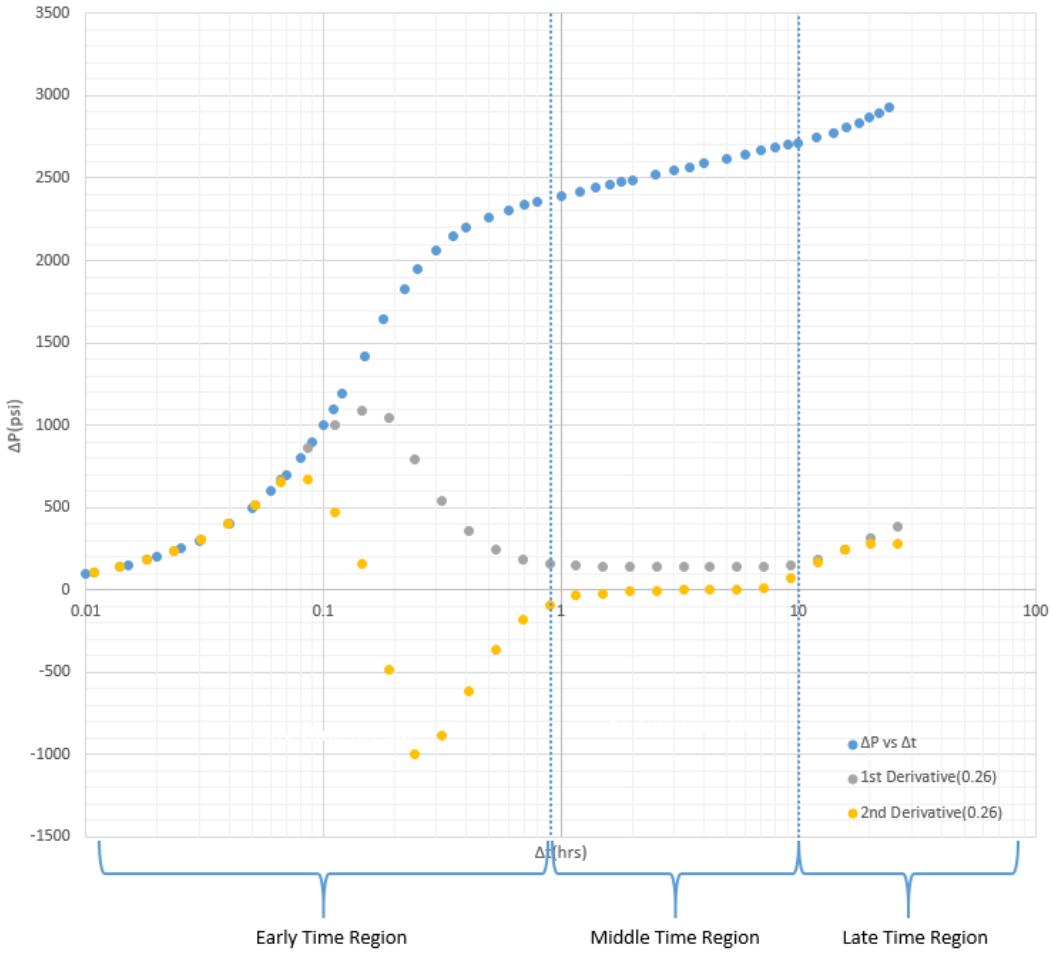


Figure 4.43 First and Second Derivative Curves (0.26 Log Cycle Interval)

Infinite acting radial flow is estimated to start at 0.90 hours and end at 10 hours.

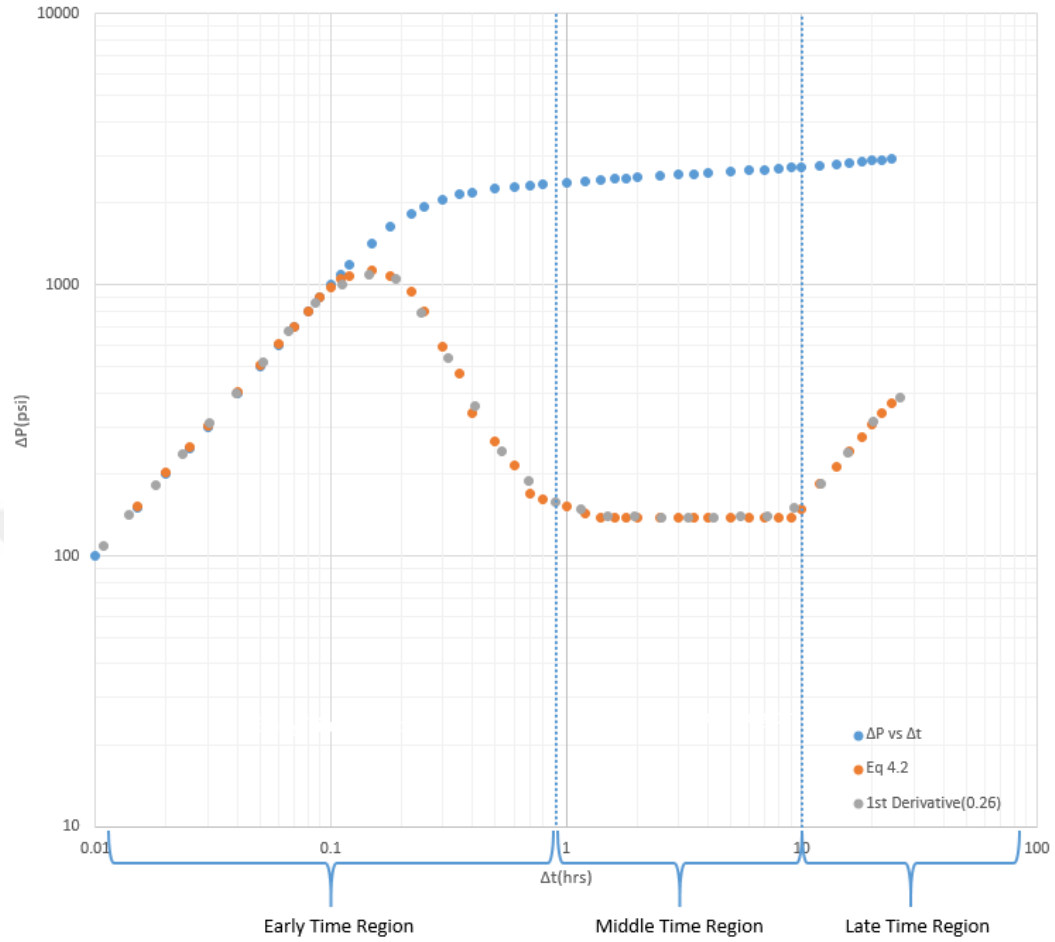


Figure 4.44 Comparison of Derivative Plots

Permeability is estimated as 39.915 md while skin factor is estimated as 2.88.

Table 4.15 Comparison of Actual and Estimated Values

Parameters	Actual Value	Estimated Value	Error(%)
k(md)	40.000	39.915	0.213
S	+2.90	+2.88	0.681

4.6.1 Gaussian Noise Addition

To test the methods tolerance to noise, random artificial Gaussian noise of ±0.5%, ±1%, ±1.5%, ±2%, ±2.5%, ±3% is added to the Drawdown Case III pressure data. The Python code had been run 20 times for each case. It is seen that the Python code can handle noises up to ±2.5%. Confidence interval of 95% is used for the calculations.

Figure 4.45 and Table below shows examples of Pwf vs t for ±0.5%, ±1%, ±1.5%, ±2%, ±2.5%, ±3%.

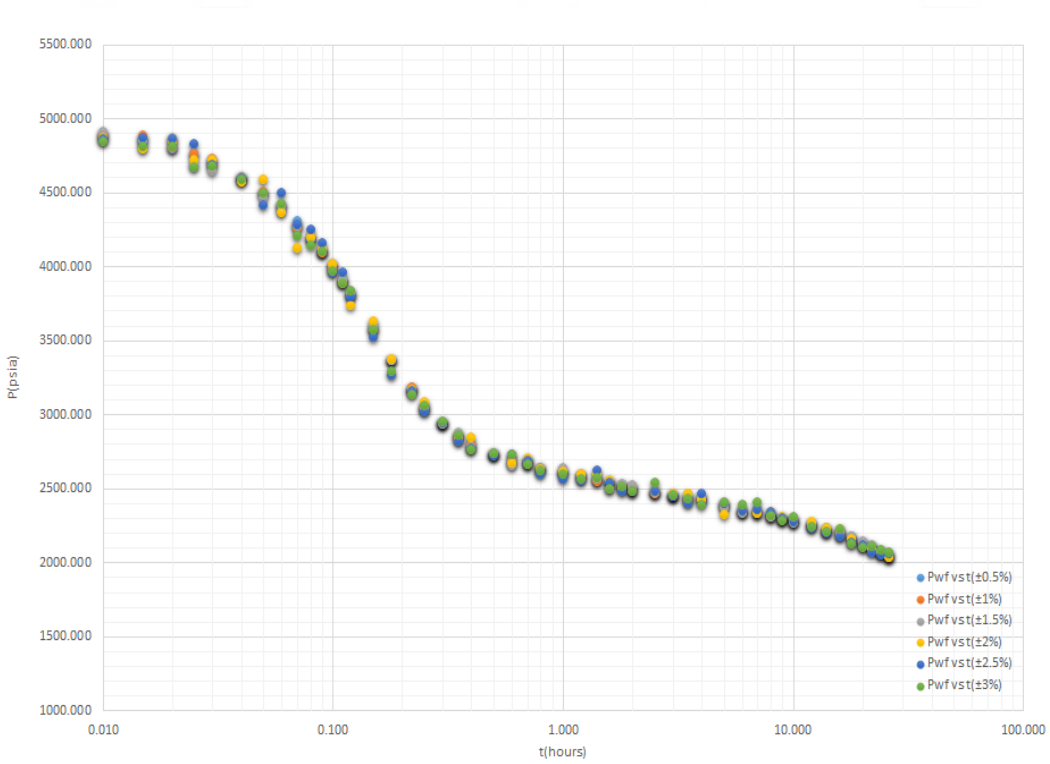


Figure 4.45 Pwf vs t Plots with Randomly Added Noises

Table 4.16 Permeability and Skin Factor Estimations with Randomly Added Noises

Parameter	**±0.5%**	**±1%**	**±1.5%**	**±2%**	**±2.5%**	**±3%**
k(md)	39.365	39.893	39.825	41.424	40.746	37.050
S	2.758	2.973	2.833	3.133	3.086	2.170

Permeability and skin factor estimates and their confidence interval results can be seen from the tables at Results and Discussion Part.

CHAPTER 5

RESULTS AND DISCUSSION

Despite all the efforts given for the reliability of the well test data, some uncertainties are still present and they cause noise in the data and they should be eliminated in order to be able to make the most logical decisions. These noises are amplified while calculating pressure derivatives and make analysis and interpretation processes more complex.

I have shown that these noises can be dealt with by finding the best log cycle interval and using that interval while calculating the pressure derivatives. As stated earlier, pressure derivatives are arguably the most useful diagnosis tool when it comes to well test analysis. Accuracies of time region identifications and as a result permeability and skin factor estimations are enhanced.

Two different methods are used for the estimation process and both had negligible error. Iterative method was slightly more accurate compared to the graphical analysis method but it required more time to run. Graphical analysis method did calculations in matter of seconds while iterative one took around one to five minutes. The difference between the two methods can be checked from the tables on the next pages.

It is shown that the Python code written can estimate permeability and skin factor more accurately than KAPPA's Saphir Module, an industry standard Pressure Transient Analysis tool, as can be seen from Table 5.1.

Table 5.2-5.7 show the permeability and skin factor estimates and their 95% confidence intervals.

Table 5.1 Absolute Error Percentages of Different Methods

Case	Iterative Method	Graphical Analysis Method	KAPPA Saphir Module
Drawdown I	0.899%	0.915%	1.243%
Drawdown II	1.889%	1.946%	3.900%

Table 5.2 Results for ±0.5% Error

Parameter	Estimate	Confidence Interval(Absolute)	Confidence Interval(%)
Permeability(md)	40.010	±0.943	±2.358
Skin Factor	2.894	±0.203	±7.008

Table 5.3 Results for ±1% Error

Parameter	Estimate	Confidence Interval(Absolute)	Confidence Interval(%)
Permeability(md)	39.334	±2.756	±7.007
Skin Factor	2.772	±0.634	±22.856

Table 5.4 Results for ±1.5% Error

Parameter	Estimate	Confidence Interval(Absolute)	Confidence Interval(%)
Permeability(md)	39.727	±2.993	±7.536
Skin Factor	2.859	±0.722	±25.262

Table 5.5 Results for ±2% Error

Parameter	Estimate	Confidence Interval(Absolute)	Confidence Interval(%)
Permeability(md)	40.454	±4.671	±11.547
Skin Factor	3.017	±0.941	±31.175

Table 5.6 Results for ±2.5% Error

Parameter	Estimate	Confidence Interval(Absolute)	Confidence Interval(%)
Permeability(md)	39.713	±6.044	±15.219
Skin Factor	2.833	±1.239	±43.714

Table 5.7 Results for ±3% Error

Parameter	Estimate	Confidence Interval(Absolute)	Confidence Interval(%)
Permeability(md)	37.210	±19.242	±51.711
Skin Factor	2.377	±4.001	±168.370

CHAPTER 6

CONCLUSIONS

Well testing gives precious data for production evaluation and optimization. It is an important part of reservoir management. The accuracy of these data is crucial for understanding the reservoir itself and making most logical decisions accordingly.

Flow period identification is the first step of well test analysis and arguably the most important step. Derivative plots and hence diagnostic plots are crucial for flow period identification, reliability of these plots directly affect the accuracy of permeability and skin factor estimates.

Despite the improvements in pressure and flow rate measurement technology, some noise is still present in most well test data and should be dealt with before making reservoir characterization and parameter estimation processes.

It is shown that the Python code can handle noises up to ±2.5% and make accurate estimations of permeability and skin factor.

REFERENCES

[1] H. Ramey, "Advances in Practical Well-Test Analysis (includes associated paper 26134)," *Journal of Petroleum Technology,* vol. 44, no. 06, pp. 650-659, 1992.

[2] A. C. Gringarten, "From straight lines to deconvolution: The evolution of the state of the art in well test analysis," *SPE Reservoir Evaluation & Engineering,* vol. 11, no. 01, pp. 41-62, 2008.

[3] R. N. Horne, "Modern Well Test Analysis: A Computer-Aided Approach, Palo Alto, California: Petroway," ed: Inc, 1995.

[4] A. Chaudhry, *Oil well testing handbook*. Elsevier, 2004.

[5] C. V. Theis, "The relation between the lowering of the piezometric surface and the rate and duration of discharge of a well using ground-water storage," *Eos, Transactions American Geophysical Union,* vol. 16, no. 2, pp. 519-524, 1935.

[6] H. Cooper Jr and C. E. Jacob, "A generalized graphical method for evaluating formation constants and summarizing well-field history," *Eos, Transactions American Geophysical Union,* vol. 27, no. 4, pp. 526-534, 1946.

[7] C. Miller, A. Dyes, and C. Hutchinson, "The estimation of permeability and reservoir pressure from bottom hole pressure build-up characteristics," *Journal of Petroleum Technology,* vol. 2, no. 04, pp. 91-104, 1950.

[8] D. Horner, "Pressure build-up in wells," in *3rd world petroleum congress*, 1951: OnePetro.

[9] J. Warren and P. J. Root, "The behavior of naturally fractured reservoirs," *Society of Petroleum Engineers Journal,* vol. 3, no. 03, pp. 245-255, 1963.

[10] A. Odeh and L. Jones, "Pressure drawdown analysis, variable-rate case," *Journal of Petroleum Technology,* vol. 17, no. 08, pp. 960-964, 1965.

[11] C. Matthews, F. Brons, and P. Hazebroek, "A method for determination of average pressure in a bounded reservoir," *Transactions of the AIME,* vol. 201, no. 01, pp. 182-191, 1954.

[12] H. Ramey, "Short-time well test data interpretation in the presence of skin effect and wellbore storage," *Journal of Petroleum Technology,* vol. 22, no. 01, pp. 97-104, 1970.

[13] R. G. Agarwal, R. Al-Hussainy, and H. Ramey, "An investigation of wellbore storage and skin effect in unsteady liquid flow: I. Analytical treatment," *Society of Petroleum Engineers Journal,* vol. 10, no. 03, pp. 279-290, 1970.

[14] A. Gringarten, H. Ramey, and R. Raghavan, "Applied pressure analysis for fractured wells," *Journal of Petroleum Technology,* vol. 27, no. 07, pp. 887-892, 1975.

[15] M. J. Mavor and H. Cinco-Ley, "Transient pressure behavior of naturally fractured reservoirs," in *SPE California regional meeting*, 1979: OnePetro.

[16] A. C. Gringarten, D. P. Bourdet, P. A. Landel, and V. J. Kniazeff, "A comparison between different skin and wellbore storage type-curves for early-time transient analysis," in *Spe annual technical conference and exhibition*, 1979: OnePetro.
[17] D. Bourdet and A. C. Gringarten, "Determination of fissure volume and block size in fractured reservoirs by type-curve analysis," in *SPE annual technical conference and exhibition*, 1980: OnePetro.
[18] H. Stehfest, "Algorithm 368: Numerical inversion of Laplace transforms [D5]," *Communications of the ACM,* vol. 13, no. 1, pp. 47-49, 1970.
[19] D. Bourdet, T. Whittle, A. Douglas, and Y. Pirard, "A new set of type curves simplifies well test analysis," *World oil,* vol. 196, no. 6, pp. 95-106, 1983.
[20] D. Bourdet, "Pressure behavior of layered reservoirs with crossflow," in *SPE California regional meeting*, 1985: OnePetro.
[21] J. Joseph, A. Bocock, F. Nai-Fu, and L. Gui, "A Study of Pressure Transient Behavior in Bounded Two-Layered Reservoirs: Shengli Field, China," in *SPE Annual Technical Conference and Exhibition*, 1986: OnePetro.
[22] F. J. Kuchuk and P. A. Kirwan, "New skin and wellbore storage type curves for partially penetrated wells," *SPE Formation Evaluation,* vol. 2, no. 04, pp. 546-554, 1987.
[23] F. Daviau, G. Mouronval, G. Bourdarot, and P. Curutchet, "Pressure analysis for horizontal wells," *SPE Formation Evaluation,* vol. 3, no. 04, pp. 716-724, 1988.
[24] D. Clark and V. Golf-Racht, "Pressure-derivative approach to transient test analysis: a high-permeability north sea reservoir example (includes associated papers 15270 and 15320)," *Journal of petroleum technology,* vol. 37, no. 11, pp. 2023-2039, 1985.
[25] S. Thomas von Schroeter, S. Florian Hollaender, and C. G. Alain, "Deconvolution of Well Test Data as a Nonlinear Total Least Squares Problem," 2001.
[26] T. Ahmed, *Reservoir engineering handbook*. Gulf professional publishing, 2018.
[27] F. J. Kuchuk, M. Onur, and F. Hollaender, *Pressure transient formation and well testing: convolution, deconvolution and nonlinear estimation*. Elsevier, 2010.
[28] A. Van Everdingen, "The skin effect and its influence on the productive capacity of a well," *Journal of petroleum technology,* vol. 5, no. 06, pp. 171-176, 1953.
[29] W. Hurst, "Establishment of the skin effect and its impediment to fluid flow into a well bore," *Pet. Eng,* vol. 25, no. 11, pp. B6-B16, 1953.
[30] M. F. Hawkins, "A note on the skin effect," *Journal of Petroleum Technology,* vol. 8, no. 12, pp. 65-66, 1956.

[31] D. Bourdet, J. Ayoub, V. Kniazeff, Y. Pirard, and T. Whittle, "Interpreting well tests in fractured reservoirs," *World Oil;(United States)*, vol. 197, no. 5, 1983.

[32] D. Bourdet, J. Ayoub, and Y. Plrard, "Use of pressure derivative in well-test interpretation," *SPE Formation Evaluation*, vol. 4, no. 02, pp. 293-302, 1989.

[33] PetroWiki. (2016). *Diagnostic plots*. Available: https://petrowiki.spe.org/Diagnostic_plots

APPENDICES

A. Python Code

```
import math
import sys
import numpy as np
import json

def LN(x):
  return math.log(x)

def FDRV(i,T,DP): # calculate Derivative at point "i" for T and DP
values
  t1=LN(T[i]/T[i-1])*DP[i+1]/LN(T[i+1]/T[i])/LN(T[i+1]/T[i-1])
  print("t1=",t1)
  t2=LN(T[i+1]*T[i-1]/T[i]**2)*DP[i]/LN(T[i+1]/T[i])/LN(T[i]/T[i-
1])
  print("t2=",t2)
  t3=LN(T[i+1]/T[i])*DP[i-1]/LN(T[i]/T[i-1])/LN(T[i+1]/T[i-1])
  print("t3=",t3)
  return t1+t2-t3

def Ftik(T,i,k):
  return math.log(T[i])-math.log(T[i-k])

def Ftij(T,i,j):
  return math.log(T[i+j])-math.log(T[i])

def ReadWellFile(fname): # reads T and P columns from TEXT file
  dt1= open(fname).read().split("\n")
  dta1=[]
  for x in dt1:
    dta1.append( x.split("\t"))
  wd2float(dta1)
  return dta1

def GetWellData(weda):
  TT= [0]*(len(weda)-1)
  PP= [0]*(len(weda)-1)
  for i in range(1,len(weda)):
    TT[i-1]= weda[i][0]
    PP[i-1]= weda[i][1]
  return (TT,PP)

def Fdrv2(TL,PL,DT,DP,DRV):
  for i in range(1,len(T)):
    DT[i]= T[i]-T[0]
    DP[i]= P[0]-P[i]
  for i in range(2,len(T)-2):
    DRV[i]= LN(TL[i]/TL[i-
1])*DP[i+1]/LN(TL[i+1]/TL[i])/LN(TL[i+1]/TL[i-1])
```

```
    DRV[i] += LN(TL[i+1]*TL[i-
1]/TL[i]**2)*DP[i]/LN(TL[i+1]/TL[i])/LN(TL[i]/TL[i-1])
    DRV[i] -= LN(TL[i+1]/TL[i])*DP[i-1]/LN(TL[i]/TL[i-
1])/LN(TL[i+1]/TL[i-1])

def findInterval(T, val):
  for i in range(len(T)-1):
    if T[i]<=val and T[i+1]>val:
      return i
  return len(T)-1

def calcDRV(T,DP,i,j,k):
    DRV  = LN(T[i]/T[i-k])*DP[i+j]/LN(T[i+j]/T[i])/LN(T[i+j]/T[i-
k])
    DRV += LN(T[i+j]*T[i-
k]/T[i]**2)*DP[i]/LN(T[i+j]/T[i])/LN(T[i]/T[i-k])
    DRV -= LN(T[i+j]/T[i])*DP[i-k]/LN(T[i]/T[i-k])/LN(T[i+j]/T[i-
k])
    return DRV

def Fdrv2LC(TL,PL,DT,DP,DRV):
  for i in range(1,len(TL)):
    DT[i]= TL[i]-TL[0]
    DP[i]= PL[0]-PL[i]
  for i in range(1,len(TL)-1):
    DRV[i]= calcDRV(TL,DP,i,1,1)
  DRV[-1]= DRV[-2]+(DRV[-2]-DRV[-3])

def Fdrv2JK(T,P,DT,DP,DRV,j,k):
  for i in range(1,len(T)):
    DT[i]= T[i]-T[0]
    DP[i]= P[0]-P[i]
  for i in range(k+1,len(T)-1-j):
    DRV[i]= LN(T[i]/T[i-k])*DP[i+j]/LN(T[i+j]/T[i])/LN(T[i+j]/T[i-
k])
    DRV[i] += LN(T[i+j]*T[i-
k]/T[i]**2)*DP[i]/LN(T[i+j]/T[i])/LN(T[i]/T[i-k])
    DRV[i] -= LN(T[i+j]/T[i])*DP[i-k]/LN(T[i]/T[i-
k])/LN(T[i+j]/T[i-k])

def FLC(T,i,j):
  return math.log(T[i]) - math.log(T[j])

def interpolate(x0,y0,x1,y1,X):
  return y0+(X-x0)*(y1-y0)/(x1-x0)

def findZero(x0,y0,x1,y1): # find the X-axis value intersected by
LINE:(x0,y0)-(x1,y1)
  My= (y1-y0)/(x1-x0)
  return x0-y0/My

def LCTP(T,P,LC):
  tx= max(T[0],0.00001)
  Tval=[tx]
  Pval=[P[0]]
```

```
  while tx<T[-1]:
    tx= math.exp(math.log(tx)+LC)
    Tval.append(tx)
    fint= findInterval(T,tx)
    if fint<(len(T)-1):
      Pvalint= interpolate(T[fint],P[fint],T[fint+1],P[fint+1],tx)
    else:
      Pvalint= interpolate(T[fint-1],P[fint-1],T[fint],P[fint],tx)
    Pval.append(Pvalint)
  return Tval,Pval

def FindLCj(T, LC): # i+j
  maxtmp=-1
  for i in range(1, len(T)):
    for j in range(1, len(T)-i):
      if FLC(T,i+j,i)>=LC:
        maxtmp=max(maxtmp, j)
        break
  return maxtmp

def FindLCk(T, LC): # i-k
  maxtmp=-1
  for i in range(1,len(T)):
    for k in range(1, i):
      if FLC(T,i,i-k)>=LC:
        maxtmp=max(maxtmp,k)
        break
  return maxtmp

def main1():
  Fdrv2JK(T,P,DT,DP,DRV,5,5)
  for i in range(len(DRV)):
    print(T[i], DRV[i])

def isfloat(x):
  for a in x:
    if a.isdigit(): continue
    if a==".": continue
    return False
  return True

def wd2float(dta):
  for i in range(len(dta)):
    for j in range(len(dta[i])):
      if isfloat( dta[i][j] ):#
print("(",i,",",j,",",dta[i][j],")",end="")
        dta[i][j]= float(dta[i][j])

def ffExp(t,a,b,c):
  return a*np.exp(b*t)+c

def getExp(T,P, deg=1):
  import scipy.optimize as opt
  x= np.asarray(T)
  y= np.asarray(P)
```

```
  pfit,perr=opt.curve_fit(lambda t,a,b,c:
a*np.exp(b*t)+c,x,y,maxfev=5000)
  return ffExp(x, *pfit)

def getLinear(T,P,deg): # fit a polynomial with DEGREE=deg to given
T,P values
  print("getlinear degree:",deg)
  import numpy as np
  import copy
  x= np.asarray(T)
  y= np.asarray(P)
  print(T[:10])
  from sklearn.preprocessing import PolynomialFeatures
  poly = PolynomialFeatures(degree=deg, include_bias=False)
  poly_features = poly.fit_transform(x.reshape(-1, 1))
  poly.transform(x.reshape(-1,1))
  #
  from sklearn.linear_model import LinearRegression
  poly_reg_model = LinearRegression()
  poly_reg_model.fit(poly_features,y)
  y_predicted = poly_reg_model.predict(poly_features)
  return y_predicted

def drawPlot(X,Y, regtype="linear", deg=2,plog=""):
  print("regtype:",regtype," deg:",deg," plog:", plog)
  import numpy as np
  npX= np.asarray(X)
  npY= np.asarray(Y)
  import matplotlib.pyplot as plt
  plt.figure(figsize=(10,6))
  if plog=="loglog":
    plt.xscale("log")
    plt.yscale("log")
  if plog=="semilog":
    plt.xscale("log")
  plt.scatter(X,Y)
  if regtype=="linear":
    ypredicted= getLinear(X,Y,deg)
    plt.plot(X,ypredicted, c="red")
  if regtype=="linear2":
     import copy
     mid=len(X)//2
     X1=copy.copy(X[:mid+1])
     Y1=copy.copy(Y[:mid+1])
     X2=copy.copy(X[mid-1:])
     Y2=copy.copy(Y[mid-1:])
     ypred1= getLinear(X1,Y1,deg)
     plt.plot(X1,ypred1,c="red")
     ypred2= getLinear(X2,Y2,deg)
     plt.plot(X2,ypred2, c="blue")
  if regtype=="exp":
     ypredicted= getExp(X,Y)
     plt.plot(X,ypredicted,"b", label="fit")
  if regtype=="exp2":
     import copy
```

```
      mid=len(X)//2
      X1=copy.copy(X[:mid])
      Y1=copy.copy(Y[:mid])
      X2=copy.copy(X[mid+1:])
      Y2=copy.copy(Y[mid+1:])
      ypred1= getExp(X1,Y1)
      plt.plot(X1,ypred1,c="red")
      ypred2= getExp(X2,Y2)
      plt.plot(X2,ypred2, c="blue")
  plt.show( )

def drawFigure(Xpoints, Ypoints, T, P):
  tt=0
  tt0=-1
  for i in range(len(T)):
    if tt0==-1 and T[i]>=3:
      tt0=i
    if T[i]>=Xpoints[0]:
      tt=i
      break
  LLEN= len(Xpoints)
  lineM, lineB= findMB(T[tt:tt+LLEN], P[tt:tt+LLEN])
  newX= [T[tt0], T[tt+LLEN-1]]
  newY= [lineM*newX[0] + lineB,lineM*newX[1] + lineB]
  #
  import matplotlib.pyplot as plt
  plt.figure(figsize=(10,6))
  #
  plt.xscale("log")
  plt.scatter(T[1:], P[1:])
  plt.plot([T[tt0],Xpoints[-1]],[P[tt0],Ypoints[-1]], c="red")
  #plt.plot(newX, newY,c="red")
  plt.show()

def findInflection(D):
   I=[]
   for i in range(len(D)-1):
     if (D[i]>0 and D[i+1]<0) or (D[i]<0 and D[i+1]>0):
        I.append(i)
   return I

def DPTcalc(TLC,DERV,i):
  if i==0 or i==len(TLC)-1: return 0
  Ti=TLC[i]
  Tim1= TLC[i-1]
  Tip1= TLC[i+1]
  DPi= DERV[i]
  DPim1= DERV[i-1]
  DPip1= DERV[i+1]
  dpti=Ti*( (Ti-Tim1)*DPip1/(Tip1-Ti)/(Tip1-Tim1) + (Tip1+Tim1-
2*Ti)*DPi/(Tip1-Ti)/(Ti-Tim1) - (Tip1-Ti)*DPim1/(Ti-Tim1)/(Tip1-
Tim1))
  return dpti

def CalcLCs(TLCv,ALL):
```

```
  AvgDRVs=[0]*len(TLCv)
  Flag=True
  for j in range(len(TLCv)):
    x= TLCv[j]
    for y in ALL.values():
      xx= list(map(lambda aa: aa[0], y))
      yy= list(map(lambda aa: aa[1], y))
      i= findInterval(xx, x)
      #print(i,x, " xx:",xx)
      #print("yy:",yy)
      if i<len(xx)-1:
        AvgDRVs[j] += interpolate(xx[i], yy[i], xx[i+1], yy[i+1],
x)
      else:
        AvgDRVs[j] += interpolate(xx[-2], yy[-2], xx[-1], yy[-1],
x)
  return list(map(lambda x: x/len(ALL.keys()), AvgDRVs))

def find100(Point, D): # find +100/-100 range in 2nd Derivative
   i= Point
   while i<len(D) and (D[i]>100 or D[i]<-100):
      i=i+1
   basladi=i
   while i<len(D) and (D[i]<100 and D[i]>-100):
      i=i+1
   bitti=i
   if bitti>= len(D):
      bitti= bitti-1
   return (basladi,bitti)

def LCval2digits(LCval): # truncate to 2 decimal points
  return int(LCval*100)/100

def findMB(pX,pY): # find SLOPE and B points of straight line fit
to (pX,pY)
  n= len(pX)
  sumX= sum(pX)
  sumY= sum(pY)
  sumX2= sum(map(lambda a: a*a, pX))
  sumXY= sum(map(lambda a: a[0]*a[1], zip(pX,pY)))
  calcM= (n*sumXY-sumX*sumY)/(n*sumX2 - sumX*sumX)
  calcB= (sumY-calcM*sumX)/n
  return calcM,calcB

def findK(jx,m):
  return -162.6*jx["Q"]*jx["B"]*jx["mu"]/(m*jx["h"])

def SkinFactor(Pi,P1hr,M,K,jx):   #phi,mu,ct,rw):
  return 1.151*((Pi-P1hr)/abs(M)-
math.log10(K/(jx["phi"]*jx["mu"]*jx["ct"]*jx["rw"]*jx["rw"]))+3.227
4)

if __name__ == "__main__":  #   <datafile.txt>  LCval LCend LCinc
  args= sys.argv            #
  print(args[0], "datafile:",args[1]," LCbegin:",args[2], "
```

```
LCend:",args[3])
  dta1= ReadWellFile(args[1]+".txt")
  jx= json.loads(open(args[1]+".json").read())
  print("jx:",jx)
  T,P= GetWellData(dta1)
  if T[0]==0: T[0]= 0.0001
  LCval=float(args[2])
  LCend=float(args[3])
  LCinc=float(args[4])
  #
  minDRV2LC=-1
  minDRV2Val=0
  minDRV3LC=-1
  minDRV3Val=0
  ALL0={}
  ALL1={}
  ALL2={}
  ALL3={}
  DF100={}
  MinRatio={}
  while LCval<LCend+LCinc:
     TLC,PLC = LCTP(T,P,LCval)
     ALL0[LCval2digits(LCval)]= list(zip(TLC,PLC))
     print("TLC(", LCval,"):", len(TLC))
     DT=[0]*len(TLC)
     DP=[0]*len(TLC)
     DRV=[0]*len(TLC)
     Fdrv2LC(TLC,PLC,DT,DP,DRV)
     #
     ALL1[LCval2digits(LCval)]=list(zip(TLC,DRV))
     #
     for i in range(len(TLC)):
       print("{:.6f}".format(TLC[i]),",",end="")
     print()
     print("First Derivative(", LCval,"):", len(DRV))
     DRV[0]=DRV[1]
     for i in range(len(DRV)):
       print("{:.2f}".format(DRV[i]),",",end="")
     print()
     DPT1=[0]*len(TLC)
     DPT2=[0]*len(TLC)
     #
     for i in range(1,len(TLC)):
       DPT1[i]= DPTcalc(TLC,DRV,i)
     DPT1[0]=DPT1[1]
     DPT1[len(TLC)-1]=DPT1[len(TLC)-2]
     PMlist= list(map(lambda x: [-1,1][x>0], DPT1))
     Chgcnt= sum(map(lambda i: PMlist[i]*PMlist[i+1]==-1,
range(len(PMlist)-1)))
     print("Second Derivative(", LCval,"):", len(TLC)," Changes:",
Chgcnt, " Ratio:",Chgcnt/len(TLC))
     MinRatio[LCval2digits(LCval)]= Chgcnt/len(TLC)
     ALL2[LCval2digits(LCval)]=list(zip(TLC,DPT1))
     if minDRV2LC==-1 or minDRV2Val>Chgcnt/len(TLC):
        minDRV2Val= Chgcnt/len(TLC)
```

```
            minDRV2LC= LCval
        for i in range(len(TLC)):
          print("{:.2f}".format(DPT1[i]),end=",")
        print()
        #
        for i in range(1,len(TLC)):
          DPT2[i]= DPTcalc(TLC,DPT1,i)
        DPT2[0]=DPT2[1]
        DPT2[len(TLC)-1]=DPT2[len(TLC)-2]
        PMlist= list(map(lambda x: [-1,1][x>0], DPT2))
        Chgcnt= sum(map(lambda i: PMlist[i]*PMlist[i+1]==-1,
range(len(PMlist)-1)))
        print("Third Derivative(", LCval,"):", len(TLC)," Changes:",
Chgcnt, " Ratio:",Chgcnt/len(TLC))
        # add to list of Ratio's, to calculate the minimum after all
ratios are calculated
        MinRatio[LCval2digits(LCval)] += Chgcnt/len(TLC)
        ALL3[LCval2digits(LCval)]=list(zip(TLC,DPT2))
        if minDRV3LC==-1 or minDRV3Val>Chgcnt/len(TLC):
            minDRV3Val= Chgcnt/len(TLC)
            minDRV3LC= LCval
        for i in range(len(TLC)):
          print("{:.2f}".format(DPT2[i]),end=",")
        print()
        # inflection points
        MinPos2ndDerivative= DPT1.index(min(DPT1))
        IP= findInflection(DPT2)
        print("Inflection points:", IP, "Minimum 2nd Derivative
position:", MinPos2ndDerivative, " Minimum
value:","{:.2f}".format(min(DPT1)))
        if MinPos2ndDerivative> max(IP):
          LCval= LCval + LCinc
          continue
        FirstIPafterMin= min(list(filter(lambda x:
x>=MinPos2ndDerivative, IP)))
        print("FirstIPafterMin2ndDerivative:", FirstIPafterMin)
        #
        IP1=FirstIPafterMin
        print("(TLC,Derivative)=
(",TLC[IP1],DPT2[IP1],")","(",TLC[IP1+1],DPT2[IP1+1],")")
        print("ZeroPoint LC:",
findZero(TLC[IP1],DPT2[IP1],TLC[IP1+1],DPT2[IP1+1]))
        F100= find100(IP1, DPT1)
        print("F100 IP:",F100)
        print("Second Derivative(+100/-
100):","{:.6f}".format(TLC[F100[0]]),"to
{:.6f}".format(TLC[F100[1]-1]), " DRV:", ''.join(map(lambda x:
"{:.2f} ".format(x), DPT1[F100[0]:F100[1]])))
        DF100[LCval2digits(LCval)]= (F100[0],F100[1]-1)
        #
        LCval = LCval + LCinc
    #
    TLCv= list(map(lambda x: x[0], ALL1[LCend]))
    AvgDRV1= CalcLCs(TLCv,ALL1)
    print("AvgDRV1:", len(AvgDRV1))
```

```
  for x in AvgDRV1:
    print(x,end=",")
  print()
  #
  AvgDRV2= CalcLCs(TLCv,ALL2)
  print("AvgDRV2:", len(AvgDRV2))
  for x in AvgDRV2:
    print(x,end=",")
  print()
  #
  AvgDRV3= CalcLCs(TLCv,ALL3)
  print("AvgDRV3:", len(AvgDRV3))
  for x in AvgDRV3:
    print(x,end=",")
  print()
  #
  print("TLC values:", len(TLCv))
  for x in TLCv:
    print(x,end=",")
  print()
  print("Minimum Change Ratio for Second Derivative: ", minDRV2Val,
" LC:", minDRV2LC)
  print("Minimum Change Ratio for Third Derivative: ", minDRV3Val,
" LC:", minDRV3LC)
  #
  MinRatioLC= float(''.join(map(lambda x:
['',str(x)][MinRatio[x]==min(MinRatio.values())],
MinRatio.keys())))
  print("Minimum Change Ratio Second+Third:",
min(MinRatio.values())," LC:", MinRatioLC,'
TLC:',DF100[MinRatioLC])
  TLCbegin=DF100[MinRatioLC][0]
  TLCvalbeg= ALL0[MinRatioLC][TLCbegin][0]
  TLCend=DF100[MinRatioLC][1]
  TLCvalend= ALL0[MinRatioLC][TLCend][0]
  XY2points= ALL0[MinRatioLC][TLCbegin:TLCend+1]
  X2points= list(map(lambda x: x[0], XY2points))
  Y2points= list(map(lambda x: x[1], XY2points))
  #
  XYpoints = list(filter(lambda x: x[0]>TLCvalbeg and
x[0]<TLCvalend, list(zip(T,P))))
  Xpoints = list(map(lambda x: x[0], XYpoints))
  Ypoints = list(map(lambda x: x[1], XYpoints))
  print("X2points:", X2points)
  print("Xpoints:", Xpoints)
  print("Y2points:", Y2points)
  print("Ypoints:", Ypoints)
  LXpoints= list(map(lambda a: math.log10(a), Xpoints))
  foundM, foundB= findMB(LXpoints, Ypoints)
  # k= findK(Q,h,B,u,m)
  foundK= findK(jx,foundM)
  print("M:", foundM, " k:", foundK)
  pfit= np.polyfit(list(map(lambda a: math.log10(a), Xpoints)),
Ypoints,1)
  print("Pfit:", pfit)
```

```
P1hr= np.polyval(pfit, math.log10(1))
for xx in Xpoints:
  print("{:.2f}".format(np.polyval(pfit, xx)),end=" ")
print()
print("P1hrs({:.2f})={:.2f}".format(P[0],P1hr))
print("Skin factor:", SkinFactor(P[0],P1hr,foundM,foundK,jx))
#
```