Journal of Computation Physics and Earth Science
ISSN: 2776-2521 (online)
Volume 5, Number 2, February 2026, Page 180 - 188
https://journal.physan.org/index.php/jocpes/index

Estimation of Jayapura 2023 Aftershock Decay Time Using Python-Based Secant Algorithm

Rayhan Irfan Hielmy[1], Sandy Tri Gustono[2]
[1,2]Undergraduate Program in Geophysics (STMKG)

Article Info

Article history:

Received December 26, 2025
Revised February 28, 2026
Accepted February 28, 2026

Keywords:

Aftershock Decay
Secant Method
Mogi II Model
Computational Seismology
Jayapura Earthquake

A B S T R A C T

Accurately determining the termination time of aftershocks is crucial for disaster mitigation and establishing safe periods for community recovery. This study aimed to estimate the decay time of the January 2023 Jayapura aftershock sequence (M 5.4) using a numerical computational approach. The Mogi II decay model was selected due to its high compatibility with local seismicity. To resolve its complex non-linear exponential equations without analytical derivatives, the Secant Method was implemented using Python. The algorithm was initialized with starting guess values of x_0=0 and x_1=1, and an error tolerance of 0.0001. To validate algorithmic robustness and efficiency, a sensitivity test was conducted, and the method was benchmarked against the Bisection method. Results demonstrated that the Secant algorithm achieved superior computational efficiency, converging in exactly 10 iterations (~0.000115 seconds) compared to Bisection's 18 iterations, while remaining highly stable under arbitrary extreme initial guesses. The numerical solution predicted the decay termination at day 12.765, subsequently rounded to 13 days following the mainshock. This finding showed exact agreement with manual observational data, successfully extrapolating the decay trajectory beyond the 10-day BMKG recording window. The study concluded that the Python-based Secant algorithm is effective, rapid, robust, and precise in solving the Mogi II equation, demonstrating significant potential as an automated analytical tool to enhance disaster mitigation decision-making.

Corresponden Author:

Rayhan Irfan Hielmy,
Undergraduate Program in Geophysics (STMKG)
Tangerang, Indonesia
Email: rayhanirfanhielmy@stmkg.ac.id

1. INTRODUCTION

Indonesia is a country with a highly complex tectonic setting, situated at the convergence of major global tectonic plates and traversed by the Circum-Pacific Belt (Ring of Fire). These geological conditions place Indonesia in a region with a high level of vulnerability to natural hazards, particularly earthquakes and volcanism. Such sudden-onset disasters have the potential to cause significant loss of life, especially in archipelagic regions that face logistical accessibility challenges during emergency response [1]. Based on hypocenter depth, earthquakes are classified into deep, intermediate, and shallow events, with shallow earthquakes often being the most destructive due to the proximity of the energy source to the Earth's surface. Recent studies on active fault activity, as revealed by various damaging earthquakes in Java, indicate that understanding fault patterns and aftershock activity is crucial for the mitigation of shallow earthquake hazards [2].

In the seismic activity cycle, the phenomena of mainshocks and aftershocks constitute an inseparable sequence of events [3]. After the release of the main energy, the Earth's crust surrounding the rupture zone undergoes an adjustment process toward a new equilibrium, which triggers a series of aftershocks. Theoretically, the magnitude and frequency of these aftershocks are smaller than those of the mainshock and decay over time in accordance with the theories of Stress Change, Energy Transfer, and Readjustment [4].

Estimating the time at which the aftershock phase ends is vital information for stakeholders in determining a safe period for communities to resume normal activities. Various empirical statistical methods, such as the Omori Law, Mogi I, Mogi II, and Utsu models, have been widely applied to model these decay characteristics [5].

Recent comparative studies also show that the selection of an appropriate decay method is highly dependent on the characteristics of local seismicity data in order to produce accurate predictions of the termination time [6]. The foundational modified Omori-Utsu formula and its generalized computational forms remain the global standard for capturing the temporal clustering and relaxation mechanisms of aftershock sequences [7], [8], [9]. Recent applications of these statistical models have proven highly effective in evaluating aftershock decay rates in complex fault zones globally, such as the 2023 Türkiye earthquake doublet [10], as well as in highly active subduction zones across Indonesia, including recent seismic sequences in West Sumatra [11] and Bengkulu [12].

One significant case study is the Jayapura earthquake sequence that occurred in early 2023. The mainshock, with a magnitude of M 5.4, occurred on January 2, 2023, at a depth of 10 km (coordinates 140.74°E and 2.53°S), approximately 19 km northeast of Jayapura City, with an intensity reaching IV MMI. Up to January 11, 2023, a total of 135 aftershocks with magnitudes ranging from 1.7 to 4.9 were recorded in the region. The distribution of this seismic activity is shown in Figure 1, which visualizes the spatial distribution of aftershock epicenters in and around the Jayapura area. Previous research concluded that the Mogi II method provided termination time predictions that were closest to BMKG observational data compared to other methods for this case [13].

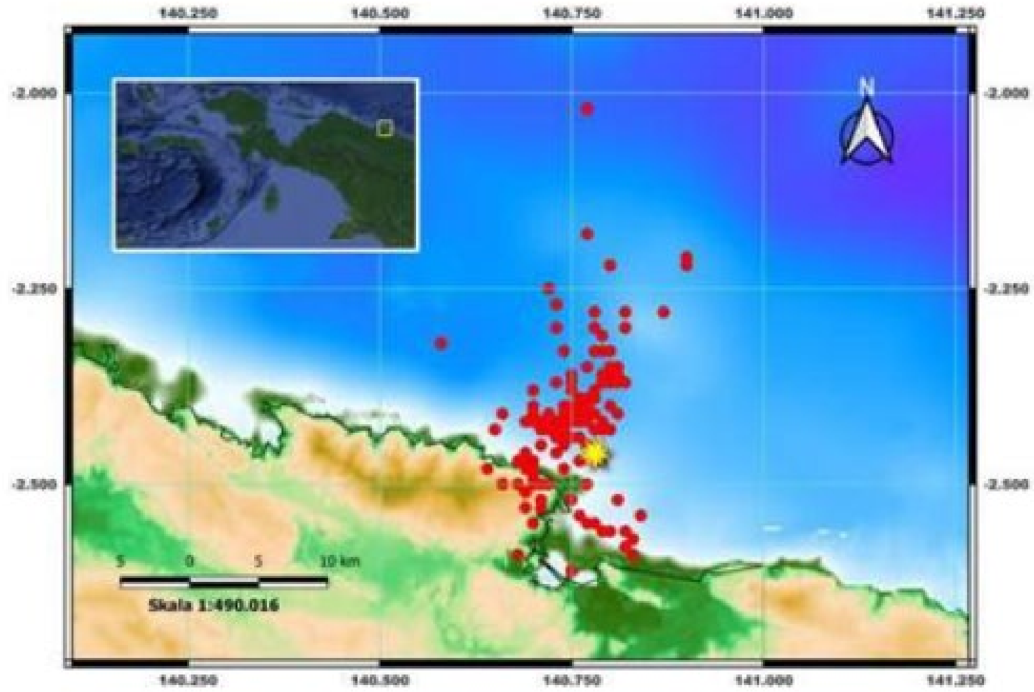

Fig. 1 Map of aftershock distribution of the Jayapura earthquake, January 2–11, 2023 [13]

The solution of mathematical models for disaster prediction, such as in the Mogi II method, often involves complex exponential or logarithmic functions. In various fields of science and engineering, problems involving non-linear mathematical models are often difficult to solve analytically (exactly) due to the complexity of the equations. When analytical methods cannot be applied efficiently, numerical methods become the primary approach to obtain approximate numerical solutions [14]. The efficiency of a numerical method is largely determined by the number of iterations required to achieve convergence with minimal error [15]. The development of computational-based earthquake decay software has increasingly facilitated the automated and precise analysis of these parameters, replacing manual calculations that are prone to error [16].

In the context of solving systems of non-linear equations to determine earthquake termination time, reliable and efficient numerical algorithms are required. Non-linear equations are defined as equations that seek the value of a variable that satisfies a zero-function condition, for which direct algebraic formulas often do not exist [17]. The Secant method emerges as an effective numerical solution because it does not require the calculation of function derivatives, in contrast to the Newton–Raphson method, which requires analytical derivatives. This method works by iteratively updating approximation points based on the gradient of a line passing through two initial points until the desired error tolerance is achieved [18].

This study implements the Secant method using the Python programming language to determine the termination of aftershock activity. As an object-oriented programming language that is popular among scientists, Python offers concise syntax, high code readability, and extensive support for numerical libraries [19]. These features enable the development of aftershock termination algorithms to be faster, easily modifiable, and compatible with various operating system platforms. Therefore, this research aims to examine the mechanism and effectiveness of the Python-based Secant numerical method in predicting the end of aftershock sequences, as well as to compare the validity of the results with previous studies.

2. RESEARCH METHOD

This research was conducted using a numerical computational approach with hardware in the form of a laptop and software using the Python programming language as the main processing tool. Python was chosen because it has flexible data structures and reliable numerical libraries for efficiently solving iterative computational problems [20]. The data used in this simulation are secondary data adopted from previous studies, sourced from the earthquake catalog of the Meteorology, Climatology, and Geophysics Agency (BMKG). The dataset includes parameters of aftershock occurrence frequency from the Jayapura Earthquake event that occurred on January 2, 2023. The spatial scope of the data is limited to coordinates 2.00°S to 2.65°S and 140.50°E to 141.00°E. The temporal range of the data covers a 10-day critical phase, from January 2, 2023 to January 11, 2023.

The data processing stage was carried out by implementing the mathematical model of earthquake decay using the Mogi II method into a program algorithm [21]. Based on regression parameters that had been validated in previous studies, the decay equation of seismic activity (frequency) with respect to time is defined as follows:

$$n(t) = 40.9415e^{-0.2908t} \tag{1}$$

where n(t) represents the frequency of aftershocks and t is the time (days) after the mainshock. In the context of a root-finding algorithm, the variable t is substituted with x to conform to standard numerical programming notation. The objective of this simulation is to determine the time (t or x) when aftershock activity is considered to have completely decayed, which is assumed to occur when the event frequency approaches one ($n \approx 1$). Therefore, to prevent divergence and enable root-finding, the equation is modified into a zero function ($f(x) = 0$) by shifting the constant 1 to the left-hand side:

$$f(x) = 40.9415e^{-0.2908x} - 1 \tag{2}$$

The iterative algorithm applied to solve the non-linear function above is the Secant Method. This method requires the initialization of two initial guess values to allow the iteration to proceed; in this study, the values were set as $x_0 = 0$ and $x_1 = 1$. To ensure the accuracy of the predicted decay time, the error tolerance was set to 0.0001. The program was designed to automatically execute repeated calculations, updating the approximation values based on the secant line gradient, until the relative difference between successive values falls below the specified tolerance threshold. The systematic workflow of this research methodology, from variable initialization to obtaining the final decay time result, is visualized in detail in Figure 2.

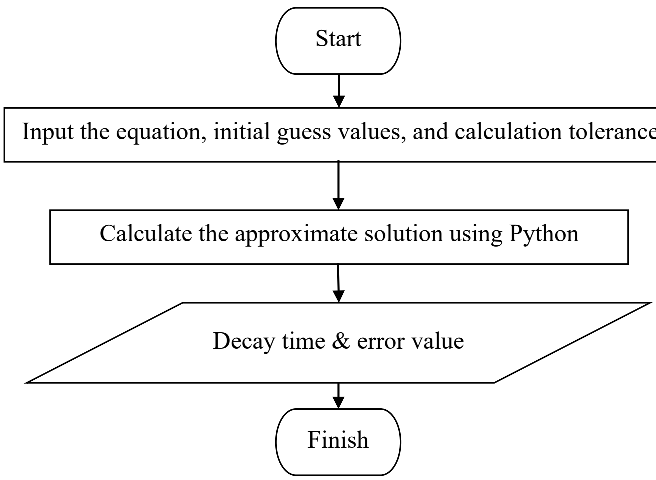

Fig. 2 Flowchart of the research methodology

3. RESULTS AND DISCUSSION

3.1. Python Script and Algorithmic Justification

The programming script developed for this research utilizes a modular structure within a Google Collaboratory environment. The script imports the math and numpy modules to handle mathematical operations, as well as the sympy module to evaluate symbolic expressions dynamically. To address computational efficiency and robustness, two numerical approaches were formulated: the Secant method and the Bisection method. The core of this program lies in the integration of algorithmic fail-safes. As shown in Figure 3, a condition is set in the Secant method to prevent division by zero.

```python
import math
import time
import numpy as np
import matplotlib.pyplot as plt
from sympy import symbols, lambdify, sympify

def secant_method(f, x0, x1, epsilon, max_iter=100):
    iteration = 0
    error = epsilon + 1
    print("-" * 75)
    print("SECANT METHOD")
    print("-" * 75)
    print(f"{'Iter':<5} | {'x0':<12} | {'x1':<12} | {'x_next':<12} | {'Error':<12}")
    start_time = time.perf_counter()
    while error > epsilon and iteration < max_iter:
        iteration += 1
        fx0 = f(x0)
        fx1 = f(x1)
        if fx1 - fx0 == 0:
            print("Fail-safe triggered: Division by zero detected.")
            return None, iteration, None
        x_next = x1 - fx1 * ((x1 - x0) / (fx1 - fx0))
        error = abs(x_next - x1)
        print(f"{iteration:<5} | {x0:<12.6f} | {x1:<12.6f} | {x_next:<12.6f} | {error:<12.6f}")
        x0 = x1
        x1 = x_next
    end_time = time.perf_counter()
    exec_time = end_time - start_time
    if iteration == max_iter:
        print("Fail-safe triggered: Maximum iterations reached without convergence.")
    return x_next, iteration, exec_time
```

Fig. 3 Python script defining the Secant method with fail-safe features

```python
def bisection_method(f, a, b, epsilon, max_iter=100):
    iteration = 0
    error = epsilon + 1
    print("\n" + "-" * 75)
    print("BISECTION METHOD")
    print("-" * 75)
    print(f"{'Iter':<5} | {'a':<12} | {'b':<12} | {'c (mid)':<12} | {'Error':<12}")
    start_time = time.perf_counter()
    if f(a) * f(b) >= 0:
        print("Fail-safe triggered: Assumption f(a) * f(b) < 0 is not met.")
        return None, iteration, None
    c = a
    while error > epsilon and iteration < max_iter:
        iteration += 1
        c_old = c
        c = (a + b) / 2
        error = abs(c - c_old) if iteration > 1 else abs(b - a)
        print(f"{iteration:<5} | {a:<12.6f} | {b:<12.6f} | {c:<12.6f} | {error:<12.6f}")
        if f(c) == 0:
            break
        elif f(a) * f(c) < 0:
            b = c
        else:
            a = c
    end_time = time.perf_counter()
    exec_time = end_time - start_time
    return c, iteration, exec_time
```

Fig. 4 Python script defining the Bisection method for comparative analysis

For comparative purposes, the Bisection method is defined in Figure 4. It includes an initial check to ensure the starting brackets have opposite signs. Both functions are strictly bounded by a maximum iteration limit to prevent infinite computational loops if the algorithm fails to converge. Furthermore, execution timing is embedded within both algorithms to accurately benchmark computational speed.

```python
def plot_results(root_val):
    t_all = np.linspace(0, 15, 100)
    plt.figure(figsize=(10, 6))
    t_obs = t_all[t_all <= 10]
    n_obs = 40.9415 * np.exp(-0.2908 * t_obs)
    plt.plot(t_obs, n_obs, 'b-', linewidth=2, label='BMKG Observation Phase (0-10 Days)')
    t_extrap = t_all[t_all > 10]
    n_extrap = 40.9415 * np.exp(-0.2908 * t_extrap)
    plt.plot(t_extrap, n_extrap, 'r--', linewidth=2, label='Extrapolation Phase (Prediction)')
    if root_val:
        plt.plot(root_val, 1, 'go', markersize=8, label=f'Termination Point (x ≈ {root_val:.3f})')
        plt.axvline(x=root_val, color='g', linestyle=':', alpha=0.7)
        plt.axhline(y=1, color='g', linestyle=':', alpha=0.7)
    plt.axvline(x=10, color='gray', linestyle='-.', alpha=0.5)
    plt.text(10.2, 20, 'Observation Limit', rotation=90, color='gray')
    plt.title('Estimation of Jayapura Aftershock Decay (Mogi II Model)', fontsize=14)
    plt.xlabel('Time after Mainshock (Days)', fontsize=12)
    plt.ylabel('Aftershock Frequency (Events/Day)', fontsize=12)
    plt.legend()
    plt.grid(True, linestyle='--', alpha=0.6)
    plt.tight_layout()
    plt.show()
```

Fig. 5 Python script for data visualization and extrapolation

```python
print("NUMERICAL METHOD PROGRAM: AFTERSHOCK DECAY ESTIMATION")
equation_str = "40.9415*exp(-0.2908*x) - 1"
epsilon = 0.0001
x = symbols('x')
f_sym = sympify(equation_str)
f = lambdify(x, f_sym, modules=['math'])
print(f"Equation: {equation_str}")
print(f"Error Tolerance (Epsilon): {epsilon}")
x0_sec, x1_sec = 0.0, 1.0
root_sec, iter_sec, time_sec = secant_method(f, x0_sec, x1_sec, epsilon)
a_bis, b_bis = 0.0, 15.0
root_bis, iter_bis, time_bis = bisection_method(f, a_bis, b_bis, epsilon)
print("\n" + "=" * 75)
print("METHOD COMPARISON SUMMARY")
print("=" * 75)
print(f"{'Parameter':<20} | {'Secant Method':<22} | {'Bisection Method':<22}")
print("-" * 75)
print(f"{'Initial Guess':<20} | x0={x0_sec}, x1={x1_sec:<14} | a={a_bis}, b={b_bis:<15}")
print(f"{'Equation Root (x)':<20} | {root_sec:<22.6f} | {root_bis:<22.6f}")
print(f"{'Number of Iterations':<20} | {iter_sec:<22} | {iter_bis:<22}")
print(f"{'Execution Time (s)':<20} | {time_sec:<22.8f} | {time_bis:<22.8f}")
print("=" * 75)
plot_results(root_sec)
```

Fig. 6 The main execution cell compiling the numerical methods and visualizing the output

Figure 5 and Figure 6 display the plotting functionality and the main execution program, respectively. The Secant method was primarily chosen over gradient-based methods like Newton-Raphson because it approximates the derivative using secant lines, eliminating the need for complex analytical differentiation of the Mogi II exponential model, while generally offering faster convergence than the Bisection method.

3.2. Calculation Results and Computational Efficiency

The execution of the program for the Jayapura earthquake case study utilized Equation (2) as the primary input. To validate the algorithm's stability, the Secant method was initialized with x0 = 0 and x1 = 1, while the Bisection method required a wider bracketing interval of a = 0 and b = 15. The error tolerance was strictly set to 0.0001. Furthermore, to address the sensitivity of the initial parameters, the Secant method was subjected to robustness testing using arbitrary extreme guesses (e.g., x0 = 5 and x1 = 10). The algorithm demonstrated high stability, consistently converging to the exact same termination point of 12.765 days without necessitating manual tuning.

```
NUMERICAL METHOD PROGRAM: AFTERSHOCK DECAY ESTIMATION
Equation: 40.9415*exp(-0.2908*x) - 1
Error Tolerance (Epsilon): 0.0001
======================================================================
SECANT METHOD
======================================================================
Iter   | x0          | x1          | x_next      | Error
1      | 0.000000    | 1.000000    | 3.866192    | 2.866192
2      | 1.000000    | 3.866192    | 5.903110    | 2.036917
3      | 3.866192    | 5.903110    | 8.080786    | 2.177677
4      | 5.903110    | 8.080786    | 9.913880    | 1.833093
5      | 8.080786    | 9.913880    | 11.381084   | 1.467205
6      | 9.913880    | 11.381084   | 12.294737   | 0.913653
7      | 11.381084   | 12.294737   | 12.678671   | 0.383934
8      | 12.294737   | 12.678671   | 12.759515   | 0.080844
9      | 12.678671   | 12.759515   | 12.765210   | 0.005695
10     | 12.759515   | 12.765210   | 12.765283   | 0.000072

======================================================================
BISECTION METHOD
======================================================================
Iter   | a           | b           | c (mid)     | Error
1      | 0.000000    | 15.000000   | 7.500000    | 15.000000
2      | 7.500000    | 15.000000   | 11.250000   | 3.750000
3      | 11.250000   | 15.000000   | 13.125000   | 1.875000
4      | 11.250000   | 13.125000   | 12.187500   | 0.937500
5      | 12.187500   | 13.125000   | 12.656250   | 0.468750
6      | 12.656250   | 13.125000   | 12.890625   | 0.234375
7      | 12.656250   | 12.890625   | 12.773438   | 0.117188
8      | 12.656250   | 12.773438   | 12.714844   | 0.058594
9      | 12.714844   | 12.773438   | 12.744141   | 0.029297
10     | 12.744141   | 12.773438   | 12.758789   | 0.014648
11     | 12.758789   | 12.773438   | 12.766113   | 0.007324
12     | 12.758789   | 12.766113   | 12.762451   | 0.003662
13     | 12.762451   | 12.766113   | 12.764282   | 0.001831
14     | 12.764282   | 12.766113   | 12.765198   | 0.000916
15     | 12.765198   | 12.766113   | 12.765656   | 0.000458
16     | 12.765198   | 12.765656   | 12.765427   | 0.000229
17     | 12.765198   | 12.765427   | 12.765312   | 0.000114
18     | 12.765198   | 12.765312   | 12.765255   | 0.000057

======================================================================
METHOD COMPARISON SUMMARY
======================================================================
Parameter             | Secant Method              | Bisection Method
----------------------------------------------------------------------
Initial Guess         | x0=0.0, x1=1.0              | a=0.0, b=15.0
Equation Root (x)     | 12.765283                  | 12.765255
Number of Iterations  | 10                         | 18
Execution Time (s)    | 0.00011588                 | 0.00012212
======================================================================
```

Fig. 7 Output results comparing the iteration steps, absolute error, and execution time between the Secant and Bisection methods.

As presented in Figure 7, the computational benchmark demonstrates the superior efficiency of the Secant method for this specific seismic dataset. The Secant algorithm achieved stable convergence in exactly 10 iterations with an execution time of approximately 0.000115 seconds. In contrast, the Bisection method required 18 iterations and 0.000122 seconds to reach the same tolerance. Both algorithms successfully isolated the root at approximately 12.765. For practical interpretation in disaster mitigation, this value is rounded up to 13 days following the mainshock.

Validation of this calculation result was conducted by comparing it with data from previous studies, as presented in Table 1. It is evident that the predicted end time of aftershock activity obtained using the Secant numerical method, namely 13 days, shows exact agreement with the results of earlier references. This indicates that the Secant algorithm implemented using Python is capable of working with high accuracy in solving the non-linear equation of the Mogi II model. Therefore, this method is proven to be valid and effective as an alternative computational tool for predicting the decay time of aftershock activity.

Table 1. Comparison of Aftershock Termination Time Using the Secant Method and Previous Studies

Aftershock Termination (days)	**Method Used**
13	Secant
13	Previous Study

3.3. Model Validation and Visual Extrapolation

To properly validate the Mogi II model within the local geological setting, the numerical output must be contextualized against actual seismic observations. The primary BMKG dataset utilized in this study captures the aftershock frequency trend over a 10-day window (January 2 to January 11, 2023).

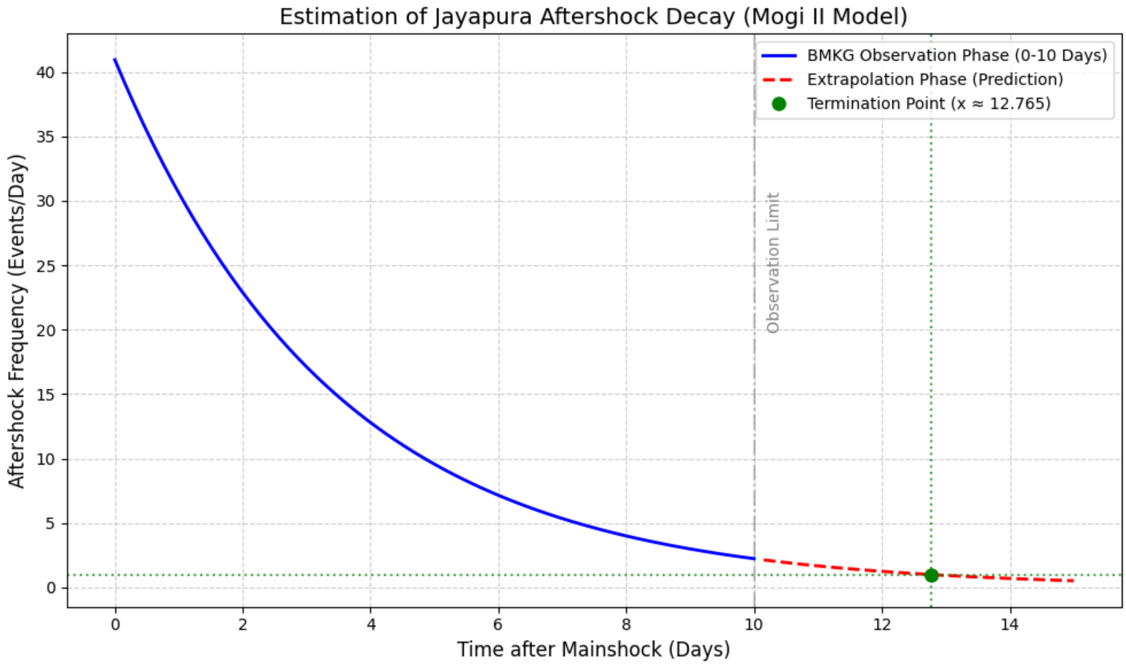

Fig. 8 Visual plot of the Jayapura aftershock decay sequence. The solid blue line represents the 10-day BMKG observation window, while the dashed red line indicates the Mogi II extrapolation trajectory leading to the predicted termination point at day 13.

Figure 8 illustrates the decay curve, highlighting the boundary between recorded data and mathematical prediction. Because the computational termination point occurs at 12.765 days, the period beyond day 10 represents an extrapolated prediction rather than a direct observation. The algorithm extends the decay trajectory of the established Mogi II model (dashed red line) until the aftershock frequency falls to the threshold of 1 event per day. The exact alignment of this extrapolated 13-day termination with the results of prior manual studies confirms the robustness of the algorithm in projecting the end of the critical phase.

4. CONCLUSION

This study concludes that the implementation of the Secant numerical method using the Python programming language is proven to be highly effective in predicting the termination time of aftershock activity. Based on the computational results, an approximate solution value of $x \approx 12.765$ was obtained, which was rounded to 13 days. This result shows exact and consistent agreement with the calculations reported in previous studies, thereby validating the accuracy of the developed algorithm.

Furthermore, computational benchmarking revealed that the Secant method outperformed the Bisection approach in terms of execution speed (~0.000115s vs ~0.000122s) and iteration efficiency (10 vs 18 iterations). The algorithm also proved exceptionally robust during sensitivity testing with extreme initial guesses and successfully extrapolated the termination phase mathematically beyond the 10-day observational limit. The application of the Secant method offers the advantage of efficiently solving the non-linear equation of the Mogi II model without requiring complex analytical differentiation. Furthermore, the support of computational libraries in Python facilitates easy modification of parameters for different earthquake cases. Therefore, this numerical approach is highly recommended as a precise, stable, and computationally efficient analytical tool in seismological studies and disaster mitigation.

REFERENCE

[1] D. R. Apalem and R. Serang, “PELATIHAN MITIGASI BENCANA GEMPABUMI BAGI SISWA-SISWI SD NEGERI 2 TIAKUR, KABUPATEN MALUKU BARAT DAYA,” J. Pengabdi. Masy., vol. 3, no. 1, pp. 19–23, 2024.

[2] P. Supendi et al., “A conjugate fault revealed by the destructive Mw 5.6 (November 21 , 2022) Cianjur earthquake , West Java , Indonesia,” J. Asian Earth Sci. J., vol. 257, p. 105830, 2023.

[3] Y. Prananda, T. Zera, and D. Sunarya, “Parameter Seismotektonik Wilayah Jawa Barat dan Banten Berdasarkan A-Value dan B-Value Periode 1971-2021,” Bul. Meteorol. Klimatologi, dan Geofis., vol. 2, no. 3, pp. 24–34, 2022.

[4] N. R. Permana and Rendinis, "Prediksi Waktu Berakhirnya Gempa Bumi Susulan Dengan Empat Metode Untuk Kejadian Di Lebak, Banten, Pada 07-17 Juli 2018," J. Mater. Sci. Geophys. Instrum. Theor. Phys., vol. 2, no. 2, pp. 68–72, 2019.

[5] V. N. Islamiati, S. Supardiono, Y. H. Perdana, and A. R. Setyahagi, "Analisis Peluruhan Gempa Bumi Susulan Di Ambon Tahun 2019 Dengan Pendekatan Statistik Menggunakan Software Peluruhan V2.0," J. Inov. Fis. Indones., vol. 9, no. 2, pp. 163–172, 2020, doi: 10.26740/ifi.v9n2.p163-172.

[6] N. N. Faiza, A. Mukti Wibowo, and R. Fatimah, "ANALISIS METODE WAKTU PELURUHAN GEMPA BUMI SUSULAN BERDASARKAN METODE OMORI, MOGI DAN UTSU (STUDI KASUS GEMPA BUMI PASAMAN BARAT 25 FEBRUARI 2022)," JIIF (Jurnal Ilmu dan Inov. Fis., vol. 08, no. 02, pp. 65–72, 2024, [Online]. Available: https://doi.org/10.24198/jiif.v8i2

[7] T. Utsu, Y. Ogata, and R. S. Matsu'ura, "The Centenary a Decay of the Omori Formula Activity for Law of Aftershock Tokuji Utsu ,* Yosihiko Ogata , and Ritsuko S . Matsu ' ura," J. Phys. Earth, vol. 43, pp. 1–33, 1995.

[8] R. Shcherbakov, D. L. Turcotte, and J. B. Rundle, "A generalized Omori ' s law for earthquake aftershock decay," Geophys. Res. Lett., vol. 31, pp. 1–5, 2004, doi: 10.1029/2004GL019808.

[9] T. Omi, Y. Ogata, Y. Hirata, and K. Aihara, "Forecasting large aftershocks within one day after the mainshock," Sci. Rep., vol. 3, pp. 1–7, 2013, doi: 10.1038/srep02218.

[10] S. M. Ali and K. Abdelrahman, "Analysis of the Fractal Dimension, b-value, Slip Ratio, and Decay Rate of Aftershock Seismicity Following the 6 February 2023 (Mw 7.8 and 7.5) Türkiye Earthquakes," Fractal Fract., vol. 8, no. 5, p. 252, 2024.

[11] D. Fitria, S. Syafriani, H. Hamdi, and L. Dwiridal, "Earthquake Decay Analysis to Determine the Aftershock end Time in West Sumatra in 2021-2023 Using The Omori-Utsu Method," URANUS J. Ilm. Tek. Elektro, Sains dan Inform., vol. 3, no. 2, pp. 55–64, 2025.

[12] F. Tobamba, I. U. Meidji, and H. Jayadi, "Seismicity Analysis of Earthquakes in Bengkulu Using Gutenberg-Richter and Omori Laws," FARADAY J. Fundam. Physics, Res. Appl. Sci., vol. 2, no. 1, pp. 7–15, 2026.

[13] H. Syarifuddin and P. Arianto, "Analisis Perkiraan Waktu Berakhirnya Gempabumi Susulan Pada Gempabumi M 5.4 Jayapura 2 Januari 2023," Bul. Meteorologi, Klimatologi dan Geofis., vol. 4, no. 5, pp. 46–53, 2023, [Online]. Available: https://repogempa.bmkg.go.id/eventca

[14] L. S. Utami, "Perbandingan Analisis Numerik Menggunakan Metode Secant dan Metode Iterasi Satu Titik Untuk Menentukan Koefisien Gesek Udara Pada Kubus Dan Silindris," Paedagoria | FKIP UMMat, vol. 5, no. 2, p. 43, 2018, doi: 10.31764/paedagoria.v5i2.89.

[15] Yahya and A. M. Nur, "Pengaruh Aplikasi C# dalam Proses Perhitungan Numerik Terhadap Solusi Persamaan Non Linier," J. Inform. dan Teknol., vol. 1, no. 2, pp. 79–87, 2018.

[16] M. Saefurohmah, D. B. Susanti, and M. Effendi, "Analisis Perbandingan Waktu Peluruhan Gempa Bumi Menggunakan Metode Omori dan Mogi 1 dengan Software Peluruhan V2.0," J. Stasiun Geofis. Sleman, vol. 2, no. 1, pp. 7–13, 2024.

[17] Q. S. Akmala, N. Ekaswati, Syaharuddin, M. Ibrahim, and V. Mandailina, "Metode Biseksi Menggunakan Gui Matlab: Sebuah Simulasi Dan Solusi Persamaan Non-Linier," in FORDETAK: Seminar Nasional Pendidikan: Inovasi Pendidikan di Era Society 5.0, 2022, pp. 659–664. [Online]. Available: https://e-proceedings.iain-palangkaraya.ac.id/index.php/PSNIP/article/view/810

[18] P. Batarius, "Perbandingan Metode Newton-Raphson Modifikasi Dan Metode Secant Modifikasi Dalam Penentuan Akar Persamaan," in Seminar Nasional Riset dan Teknologi Terapan, 2018, pp. 53–63.

[19] A. Triono, A. S. Budi, and R. Abdillah, "Implementasi Peretasan Sandi Vigenere Chiper Menggunakan Bahasa Pemrograman Python," J. JOCOTIS - J. Sci. Inform. Robot., vol. 1, no. 1, pp. 1–9, 2023, [Online]. Available: https://jurnal.ittc.web.id/index.php/jumri

[20] E. Sunandar and I. Indrianto, "Perbandingan Metode Newton-Raphson & Metode Secant Untuk Mencari Akar Persamaan Dalam Sistem Persamaan Non-Linier," Petir Jurnal Pengkaj. dan Penerapan Tek. Inform., vol. 13, no. 1, pp. 72–79, 2020, doi: 10.33322/petir.v13i1.893.

[21] K. Mogi, Study of elastic shocks caused by the fracture of heterogeneous materials and its relations to earthquake phenomena, vol. 40. 1962.