Approximating earthquake source of Italy using Steepest Descent method

F. A. Aziz$^1$ and P. H. Gunawan$^2$

School of Computing, Telkom University, Jalan Telekomunikasi No. 1 Terusan Buah Batu, Bandung 40257, Indonesia.

E-mail: $^1$alaziz.fauzi@gmail.com, $^2$phgunawan@telkomuniversity.ac.id

Abstract. This paper describes the comparison of Newton’s method and Steepest Descent method for determining the coordinates of earthquake source. Here, the Steepest Descent method is used because it is a Newton-based method. The earthquake case used in this research is Italian earthquake August 24, 2016, which has a seismic phase of Pg. The calculation was supported by Azimuth Coordinate equations to find the coordinates and Haversine formula to find the distance between five earthquake stations to the earthquake source. The final result of calculations was path’s graph from the iteration of Steepest Descent method. Moreover, the results will be compared with the results of Newton’s method that has been successfully approaching the point of earthquake source in the same case study of previous research. The result shows the number of final iterations of two methods using tolerance number 0.01, minimum velocity number 3093 m/s and three cases of initial guess in the form of city coordinate of Rome, Milan, and Palermo. Newton’s method generates 12 iterations in every case, Steepest Descent method generate 7, 6, 5 iteration respectively. However, the final numerical errors for Rome, Milan and Palermo initial guess are 0.1598 by Newton’s method, while Steepest Descent method are 0.1566, 0.1567 and 0.1567.

1. Introduction

Earthquakes is a natural catastrophic which can causes major damage to the infrastructure, social or economic aspects. Earthquakes can be happened due to several things, such as crust dislocations, volcanic eruptions, the collapse of underground mines (karsts) or even by a deliberate explosion [1]. However, finding the source of earthquakes can be a really important task. The data can we use to avoid collateral damage or for the sake of further research. According to that reason, this paper will be focused on approximating the epicenter of earthquake in mathematically, using the data from the earthquake that happened in central Italy at 24 August 2016. The data itself was gathered from five observation stations (channels) that give variables of arrival time, distance, and azimuth. All of that variables can be used to produce nonlinear system of equation.

There was a work that try to discuss the same topic with the same data [2]. In that previous work, the nonlinear system of equations has been approximated by a method called Newton’s method. This method being used because of its ability to find numerical solutions of nonlinear systems. Nevertheless, there is another method who can do the same trick, such as Steepest Descent method, Fixed Points iteration, Quasi-Newton method, Broyden method and all [3]. In this paper we will use Steepest Descent method because it is Newton-based technique and
usually not need a good initial approximation to get converge [3]. So, in the end, we can compare
the effectiveness of the two methods from the solution path they can generate.

Note that, before we use Steepest Descent method and in order to support its process, we
will present several numerical simulations. All of this procedure was the same as what was used
on previous work. Next section will give the detail of the whole simulation.

2. Data preparation
We get the data of the earthquake from the United States Geological Survey database [4]. The
important information about the earthquake specifically can be seen below

\begin{itemize}
  \item Location : (UTC 42.723°N 13.188°E) 10 km southeast of Norcia
  \item Local time : 2016-08-24 01:36:32.00
  \item Waves phase : Pg
\end{itemize}

We need to know that earthquake can categorize by its seismic wave phase, the differences in
wave phase influenced by the factors causing the earthquake, location, and activity. This seismic
wave phase is expressed by the combination of letters and symbols [5]. The earthquake in Italy
has Pg wave phase that has been detected by five channels. However, the channels can give
us more information other than wave phase. Table 1 show that the variable of distance ($L_{AZ}$),
Azimuth ($A_Z$), and arrival time of Pg waves from real earthquake epicenter to each channels
are also known.

| Channel  | Distance ($L_{AZ}$) | Azimuth ($A_Z$) | Arrival Time          |
|----------|---------------------|-----------------|-----------------------|
| IV NRCA BHZ - | 0.123043           | 333.973         | 2016-08-24 01:36:35.31|
| MN AQU BHE -  | 0.402232           | 156.394         | 2016-08-24 01:36:39.62|
| MN AQU BHZ -  | 0.402232           | 156.394         | 2016-08-24 01:36:45.92|
| V MURB BHZ - | 0.726647           | 318.194         | 2016-08-24 01:36:47.01|
| IV LATE BHZ - | 1.02637            | 264.36          | 2016-08-24 01:36:52.73|

From this part, the location of each channel must be converted into a coordinate point of
latitude ($y$) and longitude ($x$). To calculate that, Azimuth equation [6] being used. The formulas
are listed below.

\begin{align}
  x &= x_p + L_{AZ} \sin A_Z, \\
  y &= y_p + L_{AZ} \cos A_Z,
\end{align}

where ($x_p, y_p$) is a coordinate of earthquake epicenter in Norcia, $L_{AZ}$ is the distance of the
channel to the Azimuth and $A_Z$ is the value of Azimuth itself.

After equation (1) and (2) produce observation station coordinate ($x, y$) point, the distance
($d$) of each observation station coordinate to the epicenter coordinate ($x_p, y_p$) can be calculate
by using Haversine formula [7][8]. The following equation can be given as
where $R$ is the default earth sphere radius that have a value of 6,371 km, $\Delta x = x - x_p$ and $\Delta y = y - y_p$. $(x_p, y_p)$ is a coordinate of earthquake epicenter in Norcia.

Table 2. The result of implementation of Azimuth and Haversine by using data in the table 1 and known earthquake data.

| Channel      | Time (seconds) | Distance to epicenter (km) | Latitude ($y$) | Longitude ($x$) |
|--------------|----------------|----------------------------|----------------|-----------------|
| IV NRCA BHZ - | 3.31           | 13.06003                   | 42.83356       | 13.13400        |
| MN AQU BHE -  | 7.62           | 43.05577                   | 42.35442       | 13.34907        |
| MN AQU BHZ -  | 13.92          | 43.05577                   | 42.35442       | 13.34907        |
| IV MURB BHZ - | 15.01          | 71.96845                   | 43.26464       | 12.70360        |
| IV LATE BHZ - | 20.73          | 84.25389                   | 42.62213       | 12.16659        |

In table 2, Time column’s values are gathered from a subtraction of Arrival Time from table 1 and the Local time when the earthquake happened. The result of equation (3) saved in Distance to epicenter column, and then the Latitude and Longitude column obtained using equation (1) and (2).

3. Steepest Descent method and its algorithm

Talking about the Newton’s method that has been used in previous work [2], it is true that Newton’s method is an extremely powerful technique. The advantage of this method to solving system of nonlinear equations is their speed of convergence [3]. The Newton’s method formula can be seen below

$$x_n = x_{n-1} - \frac{f(x_{n-1})}{f'(x_{n-1})},$$

when $x_{n-1}$ is the value of previous $x$, variable $n$ is integer number from 1 to $\infty$, $f(x_{n-1})$ is a continuous function with previous $x$ and $f'(x_{n-1})$ is derivative functions. Newton’s method will try to find the roots of function than make $f(x) = 0$. For this paper case, the Newton’s method will modified as

$$(x_n, y_n) = (x_{n-1}, y_{n-1}) - \frac{J(x_{n-1}, y_{n-1}) f(x_{n-1}, y_{n-1})}{J(x_{n-1}, y_{n-1}) f'(x_{n-1}, y_{n-1})}$$

$J(x, y)$ in this equation (5) is what we can call as Jacobian Matrix [9], the matrix detail shown in equation (9). However, there is not such a perfect method. Newton’s method also has a weakness. To make sure the convergence, Newton’s method need an accurate initial guess value. This was the reason why we pick Steepest Descent to substitute Newton’s method, because it habit to converge even for a poor initial approximation [3][10]. The equation of Steepest Descent method can be defined by
\[ x_n = x_{n-1} - \hat{\alpha} \nabla g(x_{n-1}), \]  

(6)

and we can modified the basic equation to

\[ (x_n, y_n) = (x_{n-1}, y_{n-1}) - \hat{\alpha} \nabla g(x_{n-1}, y_{n-1}). \]  

(7)

\( \nabla g(x_{n-1}, y_{n-1}) \) is gradient (8), \( \hat{\alpha} \) is the variable that will be explained later. For more explanation, see figure 1 that visualize the behavior of Steepest Descent formula (7). The function of \( f(x, y) = x^2 + y^2 + 1 \) is used as an example. From that figure, we can see the red dot as the update for initial guess that in every iteration get closer to the bottom of the graph. It means, this method will determine the value of \((x, y)\) that can make \( f(x, y) \) at its lowest value.

**Figure 1.** Graphic show Steepest Descent iteration approximate the minimum point of function.

But before we can process the data using this equation, there are several components that must be explained and executed. First, the gradient \( \nabla g(x, y) \) for multiple function and \((x_n, y_n)\) variable that can be defined as

\[ \nabla g(x_n, y_n) = 2J(x_n, y_n)^t f(x_n, y_n), \]  

(8)

with

\[ J(x_n, y_n) = \begin{bmatrix} \frac{\partial f_1}{\partial x_n} & \frac{\partial f_1}{\partial y_n} \\ \frac{\partial f_2}{\partial x_n} & \frac{\partial f_2}{\partial y_n} \\ \vdots & \vdots \\ \frac{\partial f_i}{\partial x_n} & \frac{\partial f_i}{\partial y_n} \end{bmatrix}, \quad f(x_n, y_n) = \begin{bmatrix} f_1(x_n, y_n) \\ f_2(x_n, y_n) \\ \vdots \\ f_i(x_n, y_n) \end{bmatrix}. \]  

(9)

where \( J(x, y) \) is the matrix of derivative of every function or called as Jacobian matrix, \( f(x, y) \) is a matrix that contain the value of all function that exist from 1 to \( i \).
To find each value of matrix $J(x, y)$ and $f(x, y)$, we must know the form of the function that related to all of the earthquake data. To obtain that, we can use the relation of time ($t$), distance ($s$) and velocity ($v$) \[11\] which can be seen in the equation bellow,

$$t = \frac{s}{v},$$  

(10)

where distance ($s$) use distance vector formula as follow

$$s = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}.$$  

Therefore, the relationship can be updated as

$$t_i = \frac{\sqrt{(x_n - x_i)^2 + (y_n - y_i)^2}}{v},$$

and then we have the function as bellow

$$f_i(x_n, y_n) = \frac{\sqrt{(x_n - x_i)^2 + (y_n - y_i)^2}}{v} - t_i.$$  

(11)

where $i$ is an iteration as many as the number of the channels. $(x_n, y_n)$ value is initialized first and an update value after, while $(x_i, y_i)$ is always a coordinate of each channel.

After that, we can obtain the derivative of the function as the equation show below

$$\frac{\partial f_i}{\partial x} = \frac{(x_n - x_i)}{v\sqrt{(x_n - x_i)^2 + (y_n - y_i)^2}}, \quad \frac{\partial f_i}{\partial y} = \frac{(y_n - y_i)}{v\sqrt{(x_n - x_i)^2 + (y_n - y_i)^2}}.$$  

(12)

where $i$ is an iteration as many as the number of the channels. $(x_n, y_n)$ value is initialized first and an update value after, while $(x_i, y_i)$ is always a coordinate of each channel.

Finally, the scheme to fulfill equation (8) is complete. But there is one important property which aim to minimize gradient in use of multivariable function [3]. But before we going on that, note that the result of gradient from equation (8) will be a vector of two variable. The form will be like this

$$\nabla g(x_n, y_n) = \begin{bmatrix} a \\ b \end{bmatrix},$$  

(13)

so

$$||\nabla g(x_n, y_n)||_2 = \sqrt{a^2 + b^2}.$$  

(14)

Equation (14) is called Euclidean norm [12]. After that, the value of $\nabla g(x, y)$ can be updated with the value of the vector that has been obtained from equation (8) divided by the result of (14) which is in scalar. Equation (15) illustrate this calculation.

$$\nabla g(x, y) = \frac{\nabla g(x, y)}{||\nabla g(x, y)||_2}.$$  

(15)

Eventually, from here all the attribute that we need to calculate the gradient is gathered, so the gradient part can be considered to be done. Next objective is to set the value of $\hat{\alpha}$ in the main Steepest Descent formula form (6). Normally it is obtained by choosing $\alpha$ that can make $g(x_n, y_n)$ significantly less then $g(x_{n-1}, y_{n-1})$. Here you can see the equation of $g(x_n, y_n)$

$$g(x_n, y_n) = \sum_{i=1}^{k} [f_i(x_n, y_n)]^2.$$  

(16)

And to determine an appropriate choice for the value of $\alpha$, we consider a function
\[ h(\alpha) = g((x_n, y_n) - \alpha \nabla g(x_n, y_n)). \] (17)

The value of \( \alpha \) that minimize \( h \) is the value needed for equation (6).

Nevertheless, to reduce the cost of finding critical point of \( h \) that possibly can be processed by using root-finding formula, we choose another option. We using three numbers \( \alpha_1 < \alpha_2 < \alpha_3 \) that possibly can make the value of \( h(\alpha) \) close to minimum. In the next level of this part, choose \( \alpha_1 = 1, \alpha_3 \) is found with \( h(\alpha_3) < h(\alpha_1) \) and \( \alpha_2 = \alpha_3/2 \). After that, the value of \( g_1, g_2, \) and \( g_3 \) is known by implementing the equation (17) with each \( \alpha \). So \( g_j \) equal to \( h(\alpha_j) \). And then, by using Newton’s forward divided-difference interpolating polynomial [3] (18) we can generate \( \alpha_0 \) with the equation below

\[ h_1 = \frac{g_2 - g_1}{\alpha_2 - \alpha_1}, \quad h_2 = \frac{g_3 - g_2}{\alpha_3 - \alpha_2}, \quad h_3 = \frac{h_2 - h_1}{\alpha_3 - \alpha_1}. \] (18)

From this part, Newton’s forward divided-difference interpolating polynomial used to find \( \alpha_0 \) with it formula that shown below

\[ p(\alpha) = b_1 + b_2(\alpha - \alpha_1) + b_3(\alpha - \alpha_1)(\alpha - \alpha_2), \] (19)

when \( b_1 = g_1, b_2 = h_1, b_3 = h_3, \alpha_1 = 0, \alpha_2 = \alpha_3/2 \). Since \( \alpha = \alpha_0 \) and that is the only variable that unknown, if we initialize \( p(\alpha_0) = 0 \) we can use the relation in equation (19) to find \( \alpha_0 \). By minimize the equation, we can gather equation of \( \alpha_0 \) that shown in (20) who basically will generate the same result as equation (19) with \( p(\alpha_0) = 0 \).

\[ \alpha_0 = 0.5 \left( \alpha_2 - \frac{h_1}{h_2} \right). \] (20)

Apply \( \alpha_0 \) to make \( g_0 \) with (17). Next, choose \( \hat{\alpha} \) from existing \( \alpha \) that can make smallest \( g \). Finally, the Steepest Descent equation (7) can be executed and ready for next iteration.

Before we show the full algorithm to compute all of Steepest Descent method scheme, the algorithm of Newton method is given in algorithm 1 as a comparison material.

**Algorithm 1** Newton’s method algorithm.

1: Set \( n = 1; \) \( N \): maximum number of iteration; \((x_n, y_n)^t\): initial approximation; \( TOL \): tolerance.
2: **while** \( n \leq N \) **do** Step 3-9.
3: \hspace{1cm} Calculate \((x_n, y_n)\) by using (5).
4: \hspace{1cm} **if** \((|((x_n, y_n) - (x_{n-1}, y_{n-1})| < TOL)\) **then**
5: \hspace{2cm} OUTPUT \((x_n, y_n); \) (the procedure was successful.) **STOP.**
6: \hspace{1cm} **end if**
7: \hspace{1cm} Set \((x_{n+1}, y_{n+1}) = (x_n, y_n);\) \( n = n + 1.\)
8: **end while**
9: OUTPUT (’Maximum iterations exceeded.’)

Note that algorithm 1 was basically an algorithm that successfully determining exact earthquake epicenter in previous work [2], the full explanations can also be obtained from there. Right after this, the exact algorithm that we use to execute Steepest Descent method directly shown at algorithm 2. The algorithm may show you step by step of the real plot of Steepest Descent method that used in this paper.
Algorithm 2 Steepest Descent method algorithm.

1: Set \( n = 1 \); \( N \): maximum number of iterations; \((x_n, y_n)^t\): initial approximation; \( TOL \): tolerance.

2: OUTPUT approximate solution \((x_n, y_n)\) or message of failure.

3: while \((n \leq N)\) do step 4-24.
4: \( g_1 = g(x_n, y_n); z = \nabla g(x_n, y_n); z0 = ||z||2.\)
5: if \((z0 = 0)\) then
6: \( \text{OUTPUT} (x_n, y_n, g_1); \text{STOP}.\)
7: end if
8: \( z = z/z0; \alpha_1 = 0; \alpha_3 = 1; g_3 = \text{equation (17)} \) with \( \alpha_3.\)
9: while \((g_3 \geq g_1)\) do step 11-15.
10: \( \text{OUTPUT} (x_n, y_n, g_1); \text{STOP}.\)
11: if \((\alpha_3 < TOL/2)\) then
12: \( \text{STOP}.\)
13: end if
14: end while
15: \( \alpha_2 = \alpha_3/2; g_2 \text{ use (17) with } \alpha_2.\)
16: \( \text{Find } \alpha \text{ from } \{\alpha_0, \alpha_3\} \text{ so that } g = \text{equation (17)} = \min\{g_0, g_3\}.\)
17: \( (x_n, y_n) = (x_n, y_n) - \alpha z.\)
18: if \((|g(x_n, y_n) - g_1| < TOL)\) then
19: \( \text{OUTPUT} (x_n, y_n, g); \text{(the procedure was successful.) STOP.} \)
20: end if
21: \( n = n + 1.\)
22: end while
23: \( \text{OUTPUT} ('\text{Maximum iterations exceeded.}') \)

After the algorithm of Steepest Descent shown, we will describe the syntax in some line of algorithm 2 that might be a little tricky and need explanations. First, in line number 1, we give a value of 100 on \( N \) and \( TOL \) will be 0.01. Line 2 will print the output of initial approximation that has been given in line 1. In line 4, \( g_1 \) obtained by (17), but since \( \alpha_1 = 0 \) constantly, variable \((x_n, y_n)\) will always reduced by 0, so \( g_1 \) can be seen directly use (16). Meanwhile, \( z \) is (8), and \( z0 \) use equation (14). Line 5-7 are a mechanism to detect if \( z0 = 0 \), if that condition happen, the iteration will stop because the value of \((x_n, y_n)\) that we seek, may have been obtained. In line 8 we can see variable \( z \) that has been calculated before in line 4, being updated with the same method as (15), \( \alpha_1 \) initialized by 0, \( \alpha_3 \) initialized by 1, \( g_3 \) use equation (17) with \( \alpha = \alpha_3.\)

Line 9-14 is mechanism to keep the value of \( g_3 \) lesser than \( g_1 \), if that condition has not been obtained, \( g_3 \) will be divided by 2, but if \( g_3 \) value lesser than \( TOL/2 \) the iteration stop because the minimum value may have been obtained. In line 16, the most complex syntax could be declaring \( h_1, h_2, h_3 \) and \( \alpha_0 \) using equation (18) and (20), \( g_0 \) generate using equation (17) with \( \alpha = \alpha_0.\)

In line 17, the algorithm use searching mechanism to search \( \alpha \) that can make the smallest \( g \), \( g \) calculated by using equation (17). And then, the value of next \((x, y)\) has been achieved by doing (7) in line 18. So finally line 19-21 work for comparing \( \Delta g \) to \( TOL \), if the value is lesser than the tolerance, iteration stop and we might have what we seek.
4. Numerical simulation

In this section, three cases will be elaborated to do the numerical simulation. Each case depends on the initial guess of certain city coordinate of Italy.

- **Case 1:** \((x_0 = 12.4964, \ y_0 = 41.9028)\), coordinate capital city of Rome.
- **Case 2:** \((x_0 = 9.1900, \ y_0 = 45.4642)\), coordinate city of Milan.
- **Case 3:** \((x_0 = 13.3613, \ y_0 = 38.1157)\), coordinate city of Palermo.

However, in any case, the velocity of wave is always the same \(v = 3093 \text{ m/s}\). The value is the minimum velocity from the calculation of data in table 2 by using correlation in equation (10).

As we already said in the introduction section, we will compare the result of the numerical simulation of previous work that using Newton’s method with this result that using Steepest Descent method.

Figure 2 shown the trajectory that has been generated using both method, Newton and Steepest Descent with initial approximation using coordinate of the capital city of Rome. As we can see, the solution trajectory that Newton’s method generate moved away to Corse Island in the west before it turn back and get convergence to Norcia, Italy. Meanwhile, the solution path that Steepest Descent method created looks like it heading south to Tyrrhenian Sea, turn north and move a little further before it’s convergence to Norcia. However, for more analytical purpose, the final data of iteration is given.

![Figure 2](image)

**Figure 2.** The solution path of numerical result using initial guess from capital city of Rome. Trajectory of Newton’s method solution (left) [2]. Trajectory of Steepest Descent method solution (right).

**Table 3.** The final result of each method with an initial guess using the capital city of Rome.

| Method               | Iteration | Final Coordinate | Error  |
|----------------------|-----------|------------------|--------|
| Newton               | 12        | \(x = 13.0414\), \(y = 42.7097\) | 0.1598 |
| Steepest Descent     | 7         | \(x = 13.0439\), \(y = 42.7105\) | 0.1566 |

Table 3 shows the numerical result of final iteration of both Newton’s method and Steepest Descent method. If we compare the sum of the iterations, the Newton’s method have more loop. It can be mean that, for this scenario, Steepest Descent method can be more quickly get convergence. And if we see the final coordinate and error, Steepest Decent convergence is also slightly more accurate then Newton.
Moreover, if the coordinate of Milan used as an initial approximation as we can see in figure 3, Newton’s method trajectory move to Corse Island, France. Have a turn a little wider to the east before converge to Norcia. The Steepest Descent solution path more like straightly move to earthquake source just like a line.

Table 4. Final result of each method with initial guess of city of Milan.

| Method            | Iteration | Final Coordinate | Error  |
|-------------------|-----------|------------------|--------|
| Newton            | 12        | $x = 13.0414, y = 42.7097$ | 0.1598 |
| Steepest Descent  | 6         | $x = 13.0438, y = 42.7105$ | 0.1567 |

Table 4 almost can tell the same thing as table 3 tell us. The sum of iteration, final coordinate and error of Newton’s method is not change, but Steepest Descent have lesser loop then before.

Meanwhile, using initial guess of Palermo City as we can see in figure 4, the Newton’s version of solution path seen to move to Corsa Island again before it get converge. However, the Steepest Descent method solution path is move straight north, crossing the sea and converge. If we see table 5, there is not much to tell, because the result of Newton is the same as before. Steepest Descent itself not have much change besides the sum of iterations that get a lesser loop then another result with another initial approximation.

Figure 3. The solution path of numerical result using initial guess from city of Milan. Trajectory of Newton’s method solution (left) [2]. Trajectory of Steepest Descent method solution (right).

Figure 4. The solution path of numerical result using initial guess from city of Palermo. Trajectory of Newton’s method solution (left) [2]. Trajectory of Steepest Descent method solution (right).
Table 5. Final result of each method with initial guess of city of Palermo.

| Method             | Iteration | Final Coordinate | Error   |
|--------------------|-----------|------------------|---------|
| Newton             | 12        | $x = 13.0414, y = 42.7097$ | 0.1598  |
| Steepest Descent   | 5         | $x = 13.0438, y = 42.7105$ | 0.1567  |

If we see the table 3-5, the result of Newton’s method are always the same. The sum of iterations, final coordinate and the error have no change. It could be mean that in this research the Newton’s method result not really depend on correct initial guess. Meanwhile, the result of Steepest Descent are changing often along with the change of the initial guess value, especially on the number of their iterations. The result from the use of Palermo City as the initial approximation has the fastest move to convergence. And if we compare the result of the two methods, the Steepest Descent was cost less in iteration and had more precise convergence then Newton’s method.

5. Conclusion
Few numerical simulations using Steepest Descent method to observing the solution trajectory that approximate the epicenter of an earthquake in Italy has been done. And we can tell that it was well investigated. Three scenario depend on the modification of initial approximation coordinate are elaborated.

- **Case 1:** Here the comparison between the result from the two methods with initial approximation of the capital city or Rome. Newton’s method stopped at 12-th iteration and produce numerical error to exact epicenter 0.1598. Meanwhile Steepest Descent method stopped at 7-th and error 0.1566. In this case we can assume that Steepest Descent method become more efficient then Newton’s method.

- **Case 2:** When the initial approximation use the coordinate city of Milan, the Newton’s method converge at 12-th and come with numerical error 0.1598. The Steepest Descent in other hand, stopped at 6-th iteration and minimize the error until 0.1567. In this case of simulation, Steepest Descent still gave lesser value and more efficient work.

- **Case 3:** However in this final case that we use initial guess of Palermo City, Newton’s method converge at 12-th with numerical error 0.1598 when Steepest Descent can converge at 5-th and have numerical error as 0.1567. In this case the Steepest Descent reach it highest efficiency, still work better then Newton’s method.

The conclusion of this paper would be the Steepest Descent perform more effectively in its way to convergence and approximate exact earthquake epicenter source than Newton’s method. But however, if we compare the complexity of the Newton’s method [2][11] from it’s step in algorithm 1, and Steepest Descent method through explanations at algorithm 2, the Newton’s method had more simple equations and less computation to execute.

References
[1] Elnashai A S and Di Sarno L 2008 Fundamentals of earthquake engineering (Wiley New York)
[2] Gunawan P and Prakoso N 2017 Control, Electronics, Renewable Energy and Communications (ICCREC), 2017 International Conference on (IEEE) pp 202-206
[3] Burden R L and Faires J D 2001 Prindle Weber and Schmidt, Boston
[4] USGS U 2016 M 6.2 - 10km se of norcia, italy. technical report URL https://earthquake.usgs.gov/earthquakes/eventpage/us10006g7d#executive
[5] Storchak D A, Schweitzer J and Bormann P 2003 Seismological Research Letters 74 761–772
[6] Rutstrum C and Rustrum C 2000 The wilderness route finder: the classic guide to finding your way in the wild (University of Minnesota Press)
[7] Feng T and Timmermans H J 2013 Transportation Research Part C: Emerging Technologies 37 118–130
[8] Robusto C C 1957 The American Mathematical Monthly 64 38–40
[9] Mathai A M 1997 Jacobians of matrix transformations and functions of matrix arguments (World Scientific Publishing Company)
[10] Battiti R 1992 Neural computation 4 141–166
[11] Suparno S, Fisika-FMIPA D and Indonesia U 2014 Jakarta: Universitas Indonesia
[12] Barni M, Buti F, Bartolini F and Cappellini V 2000 IEEE Transactions on Image Processing 9 1704–1709