J. Indones. Math. Soc.
Vol. 32, No. 02 (2026), pp. 1–13.

Stochastic Influence on Levenberg-Marquardt Method for Nonlinear Least Squares Problems

Gerend Christopher[1]*, Janson Naiborhu[1]

[1]Department of Mathematics, Institut Teknologi Bandung, Indonesia

Abstract. In this digital era, a lot of data can be collected to be extracted and used for decision making through mathematical models in solving cases in certain domains. This paper will focus on solving nonlinear least squares problems in building an optimal model, namely one that has good accuracy and is efficient. In practice, the model is built using numerical methods. The main objective of this study is to investigate the effect of stochasticity in numerical methods that utilize gradients, namely Levenberg-Marquardt, on accuracy and computational efficiency. In addition, several numerical results from several variants of Stochastic Levenberg-Marquardt sampling and data taken in clusters with K-Means will be compared. The results of this paper are Stochastic Levenberg-Marquardt with 100, 200, 300, 400, and 500 data sample sizes outperformed classical Levenberg-Marquardt since they have very low relative errors to Levenberg-Marquardt and are faster in computational time than Levenberg-Marquardt. Therefore, when solving nonlinear least squares problems with large data sets, the Levenberg-Marquardt method requires only a small number of samples, which suggests that using the Stochastic Levenberg-Marquardt approach is advantageous.

Key words and Phrases: nonlinear least squares, stochastic Levenberg-Marquardt, cluster, accuracy, K-means, computational efficiency.

1. INTRODUCTION

Advancements in mathematics, science, and technology has driven humans to digital and information era. This modern era leads to the exponential growth of creating, collecting, and utilizing data as stated in Kapadiya et al. [1] and Horvitz and Mitchell [2]. As data collection and processing have advanced, the need for effective methods to extract meaningful insights from this data has become increasingly apparent. The development of mathematical models that capable of handling

*Corresponding author: 20924009@mahasiswa.itb.ac.id
2020 Mathematics Subject Classification: 60H30, 62M20.
Received: 03-03-2025, accepted: 12-12-2025.

complex dataset is a necessity as well. Hokanson [3] stated that among the various approaches to model data, nonlinear least squares problems have emerged as a critical area of focus, particularly for their applications in data fitting and parameter estimation across diverse fields such as engineering, economics, and machine learning.

Nonlinear Least Squares problems involve finding the best fit curve to a set of data points by minimizing the sum of the squares of the residuals, which usually called the objective function of this problem. This task for finding optimal model that has high accuracy or lower error and has computational efficiency led to the development of numerous of numerical methods [4, 5]. In Nocedal [6] and Yuan [5], Gradient-based numerical methods is one of those methods for solving nonlinear least squares problems.

With the growth of data created, recent advances have explored the integration of stochastic elements into these numerical methods to potentially enhance their performance and its applications [7, 8, 9, 10]. One specific method, Stochastic Gradient Descent that introduce randomness into the optimization process can influence both accuracy and computational efficiency as in Robbins [7] and Kiefer [8]. The impact of such stochastic methods on gradient-based numerical methods remains a topic of ongoing research.

The Levenberg-Marquardt algorithm has been extensively studied for solving nonlinear least squares problems. Early applications include the analysis of EPR spectra (1996) [11] and the development of damping and undamping strategies (1997) [12]. More recent efforts have focused on improving the accuracy and computational efficiency of the algorithm. For example, Yan et al. proposed AdaLM (Adaptive Levenberg-Marquardt), which adapts the step size to enhance performance [13]. Building on this line of work, Hong et al. (2020) introduced a stochastic version of the Levenberg-Marquardt algorithm by using randomly selected subsets of data. [14]. In contrast, our approach introduces a new variation of stochastic sampling: at each iteration, a different random subset of the data is selected. Furthermore, this study incorporates K-Means clustering, using cluster centroids as representative data points, which differentiates our method from previous work.

The main purpose of this research is to investigate the influence of stochastic in numerical methods that utilize gradients, especially for Levenberg-Marquardt method, on accuracy and computational efficiency. In this study, numerical result with data taken in clusters will be compared to Stochastic Levenberg-Marquardt.

2. PROBLEM STATEMENT AND METHODS

2.1. **Least Squares Problem.**

When we have data $(x_i, y_i),\ i = 1, 2, \ldots, m$, the relationship of x_i and y_i can be written in equation (1)

$$y_i = M(\mathbf{p}, x_i) + \epsilon_i \ , \tag{1}$$

where $\mathbf{p} = [p_1, p_2, \ldots, p_n]^{\mathrm{T}}$ is a vector consist of model parameters, $M(\mathbf{p}, x_i)$ is a fitting model, and e_i is measurement error on the data ordinate.

For any choice of $\mathbf{p}$, residuals can be computed by

$$f_i(\mathbf{p}) = y_i - M(\mathbf{p}, x_i). \tag{2}$$

The least squares problem is to minimize the sum of the squared residuals by obtaining $\mathbf{p}^*$ as the minimizer. Madsen [15] define least squares problem in Definition 2.1

Definition 2.1. *[Least Squares Problem] Find a local minimizer* $\mathbf{p}^*$ *for*

$$F(\mathbf{p}) = \frac{1}{2}\sum_{i=1}^{m}(f_i(\mathbf{p}))^2 \ , \tag{3}$$

where $f_i : \mathbb{R}^n \mapsto \mathbb{R},\ i = 1, 2, \ldots, m$ *are given functions, and* $m \geq n$.

As an objective function, $F(\mathbf{p})$ can also be called as *loss function* or *cost function*.

2.2. **Gradient Descent.**

Gradient Descent is an iterative optimization numerical method that uses first order derivative approach with the aim of minimizing the objective function. In Ruder [16], Gradient Descent is a way to minimize objective function $F(\mathbf{p}_k; \boldsymbol{\zeta})$ parameterized by its model parameters $\mathbf{p}_k \in \mathbb{R}^n$ by updating the value of parameters with opposite direction of gradient of objective function. In Tian et al. [17], updating the value of parameters $\mathbf{p}$ using Gradient Descent can be written as in equation (4)

$$\mathbf{p}_{k+1} = \mathbf{p} - \boldsymbol{\tau}_k \nabla F(\mathbf{p}_k; \boldsymbol{\zeta}), \tag{4}$$

where k denoted as iteration of the simulation. Symbol $\boldsymbol{\tau}_k$ called as *learning rate* which usually constant greater than zero that regulate updating steps using gradient. Symbol $\boldsymbol{\zeta} : (x_i, y_i),\ i = 1, 2, \ldots, m$ is all samples used when calculating $\nabla F(\mathbf{p}_k; \boldsymbol{\zeta})$ for every iteration.

Garrigos and Gower [18] showed one way about convergence of Gradient Descent as written in Theorem 2.3

Definition 2.2. *Let* $F : \mathbb{R}^n \to \mathbb{R}$ *and* $L > 0$. *F is said $L-$smooth if it is differentiable and if* $\nabla F : \mathbb{R}^n \to \mathbb{R}^n$ *is $L-$Lipschitz:*

$$\textit{for all } x, y \in \mathbb{R}^n,\ \|\nabla F(x) - \nabla F(y)\| \leq L \|x - y\| .$$

Theorem 2.3. *Suppose we want to minimize a differentiable function F and minimizer (argmin) $F \neq \emptyset$. Assume F convex and $L-$smooth, $L > 0$. Let $(p^k)_{k\in\mathbb{N}}$ be a sequence of iterates generated by Gradient Descent algorithm, with a learning rate satisfying $0 < \tau_k < \frac{1}{L}$. Then, for all $p^* \in$ argmin F, for all $k \in \mathbb{N}$, we have*

$$F(p^k) - \inf F \leq \frac{\left\|p^0 - p^*\right\|^2}{2\tau_k k}.$$

As the number of data samples increases, the associated computational cost also rises accordingly. Therefore, recent advances introduce randomness or stochastic elements into Gradient Descent. It is done by sampling the data uniformly. In Tian et al. [17] stated that Stochastic Gradient Descent computes gradient only uses one random sample in each iteration. Mini-Batch Stochastic Gradient Descent randomly sampled small-batch data $\mathcal{B}_j$ to compute gradient. These two variants save a lot of memory in computation which might lead to speed increased. In this study, Stochastic Gradient Descent term will be used interchangeably for those two variants. With $\mathcal{B}_j,\ j = 1, 2, \ldots, m$ batch data, updating parameters value can be written by

$$\mathbf{p}_{k+1} = \mathbf{p} - \boldsymbol{\tau}_k \nabla F(\mathbf{p}_k; \mathcal{B}_j). \tag{5}$$

To show one way about the convergence of Stochastic Gradient Descent, Garrigos and Gower [18] depicts the objective function as a sum of functions $F(\mathbf{p}) \stackrel{\text{def}}{=} \frac{1}{d}\Sigma_{i=1}^{d} F_i(\mathbf{p})$, where $F_i : \mathbb{R}^n \to \mathbb{R}$ is bounded from below and *argmin* $F \neq \emptyset$. They said sum of functions problem as Sum of Convex when $F_i : \mathbb{R}^n \to \mathbb{R}$ is assumed to be convex. They said sum of functions problem as Sum of $L_{max}-$smooth when $F_i : \mathbb{R}^n \to \mathbb{R}$ is assumed to be L_i-smooth and denoted $L_{max} \stackrel{\text{def}}{=} \max L_i$. Garrigos and Gower [18] showed one way about convergence of Stochastic Gradient Descent as written in Theorem 2.4.

Theorem 2.4. *Let the problem hold Sum of $L_{\max}-$smooth and Sum of Convex assumptions. Consider $(p^k)_{k\in\mathbb{N}}$ be a sequence of iterates generated by Stochastic Gradient Descent algorithm, with a learning rate satisfying $0 < \tau_k < \frac{1}{2\mathcal{L}_j}$. It follows that*

$$\mathbb{E}[F(p^k) - \inf F] \leq \frac{\left\|p^0 - p^*\right\|^2}{2\sum_{t=0}^{k-1} \tau_t(1 - 2\tau_t\mathcal{L}_j)} + \frac{\sum_{t=0}^{k-1} \tau_t^2}{\sum_{t=0}^{k-1} \tau_t(1 - 2\tau_T\mathcal{L}_j)}\sigma_j^*,$$

where $\bar{p}^k \stackrel{\text{def}}{=} \sum_{t=0}^{k-1} u_{k,t}p^k$ with $u_{k,t} \stackrel{\text{def}}{=} \frac{\tau_t(1-2\tau_t\mathcal{L}_j)}{\sum_{i=0}^{k-1} \tau_i(1-2\tau_i\mathcal{L}_j)}$.

2.3. **Levenberg-Marquardt.**

Levenberg-Marquardt (LM) is an iterative optimization numerical method that uses second order derivative approach with the aim of minimizing the objective function. While Gradient Method solely uses first order derivative, this

method leverage the second order derivative. As this method may be considered as an extension of Gauss-Newton's method, the Levenberg-Marquardt also based on solving linearized least squares problems. In Madsen [15], the Levenberg-Marquardt method is based on a linear approximation to the components of $\mathbf{f}$ in the neighborhood of $\mathbf{p}$. For small $||\mathbf{h}||$, the Taylor expansion is

$$\mathbf{f}(\mathbf{p}+\mathbf{h}) \approx \mathbf{l}(\mathbf{h}) \equiv \mathbf{f}(\mathbf{p}) + \mathbf{J}(\mathbf{p})\mathbf{h}. \tag{6}$$

Thus, F denoted as

$$\begin{aligned} F(\mathbf{p}+\mathbf{h}) \approx L(\mathbf{h}) &\equiv \frac{1}{2}\mathbf{l}(\mathbf{h})^{\mathrm{T}}\mathbf{l}(\mathbf{h}) \\ &= \frac{1}{2}\mathbf{f}^{\mathrm{T}}\mathbf{f} + \mathbf{h}^{\mathrm{T}}\mathbf{J}^{\mathrm{T}}\mathbf{f} + \frac{1}{2}\mathbf{h}^{\mathrm{T}}\mathbf{J}^{\mathrm{T}}\mathbf{J}\mathbf{h} \\ &= F(\mathbf{p}) + \mathbf{h}^{\mathrm{T}}\mathbf{J}^{\mathrm{T}}\mathbf{f} + \frac{1}{2}\mathbf{h}^{\mathrm{T}}\mathbf{J}^{\mathrm{T}}\mathbf{J}\mathbf{h}, \end{aligned} \tag{7}$$

where $\mathbf{J}$ is Jacobian. Linear approximation for Gradient and Hessian respectively are

$$F'(\mathbf{p}) = L'(\mathbf{0}) = \mathbf{J}^{\mathrm{T}}\mathbf{f} \qquad F''(\mathbf{p}) = L''(\mathbf{0}) = \mathbf{J}^{\mathrm{T}}\mathbf{J}. \tag{8}$$

Updating the value of $\mathbf{p}$ using $\mathbf{h}_{lm}$ with Levenberg-Marquardt method is done by solving equation (9)

$$(\mathbf{J}^{\mathrm{T}}\mathbf{J} + \mu\mathbf{I})\mathbf{h}_{lm} = -\mathbf{g} \quad \text{with } \mathbf{g} = \mathbf{J}^{\mathrm{T}}\mathbf{f} \text{ and } \mu \geq 0. \tag{9}$$

Constant μ is the damping parameter which influences both the direction and the size of the step [15]. It is stated that the initial value of μ is related to the size of the elements in $\mathbf{A}_0 = \mathbf{J}(\mathbf{p}_0)^{\mathrm{T}}\mathbf{J}(\mathbf{p}_0)$ by letting

$$\mu_0 = \theta \cdot \max_i\{a_{ii}^{(0)}\} \tag{10}$$

where θ is chosen by user. Updating value $\mathbf{p}$ is controlled by *gain ratio* q in equation (11). If q is large value, $L(\mathbf{h}_{lm})$ is a good approximation on $F(\mathbf{p}+\mathbf{h}_{lm})$. If q is small, $L(\mathbf{h}_{lm})$ is a poor approximation.

$$q = \frac{F(\mathbf{p}) - F(\mathbf{p}+\mathbf{h}_{lm})}{L(\mathbf{0}) - L(\mathbf{h}_{lm})}. \tag{11}$$

The pseudocode which details the step for updating parameters value using Levenberg-Marquardt are shown in Algorithm 1.

Algorithm 1 Levenberg-Marquardt

Input: $\theta > 0$, initial $\mathbf{p} \in \mathbb{R}^n$, data $\boldsymbol{\zeta} : (x_i, y_i), i = 1, 2, \ldots, m$
Output: $\mathbf{p}$
Initialize: $\nu = 2$, $\mathbf{A} = \mathbf{J}(\mathbf{p}, \boldsymbol{\zeta})^T \mathbf{J}(\mathbf{p}, \boldsymbol{\zeta})$, $\mathbf{g} = \mathbf{J}(\mathbf{p}, \boldsymbol{\zeta})^T \mathbf{f}(\mathbf{p}, \boldsymbol{\zeta})$, $\mu = \theta \cdot \max_i\{a_{ii}^0\}$

for $k = 1, 2, \ldots,$ maximum iteration **do**
  Solve $(\mathbf{A} + \mu \mathbf{I})\mathbf{h}_{lm} = -\mathbf{g}$
  $\mathbf{p}_{new} = \mathbf{p} + \mathbf{h}_{lm}$
  Calculate $q = (\mathbf{F}(\mathbf{p}, \boldsymbol{\zeta}) - \mathbf{F}(\mathbf{p}_{new}, \boldsymbol{\zeta}))/(\mathbf{L}(\mathbf{0}, \boldsymbol{\zeta}) - \mathbf{L}(\mathbf{h}_{lm}, \boldsymbol{\zeta}))$
  if $q > 0$ **then** ▷ Step accepted
    $\mathbf{p} = \mathbf{p}_{new}$
    Calculate $\mathbf{A} = \mathbf{J}(\mathbf{p}, \boldsymbol{\zeta})^T \mathbf{J}(\mathbf{p}, \boldsymbol{\zeta})$, $\mathbf{g} = \mathbf{J}(\mathbf{p}, \boldsymbol{\zeta})^T \mathbf{f}(\mathbf{p}, \boldsymbol{\zeta})$
    $\nu = 2$
  else
    $\mu = \mu * \nu$; $\nu = 2 * \nu$
  end if
end for

2.4. **Stochastic Levenberg-Marquardt.**

While Stochastic Gradient Descent randomly sampled data for calculating gradient (first order derivative), in Stochastic Levenberg-Marquardt (SLM), randomly sampled data used for calculating gradient as well as Hessian matrix, specifically for calculating Jacobian matrix (first order and second order derivative) [19]. In Levenberg-Marquardt, Jacobian matrix, gradient, and Hessian matrix are calculated with all data samples, while Stochastic Levenberg-Marquardt calculate those with randomly sampled data. The pseudocode that details the step for updating parameters value using Stochastic Levenberg-Marquardt are shown in Algorithm 2.

2.5. **K-Means.**

In this study, data will be sampled with centroid-based clustering algorithm, specifically K-Means. Centers from clustering algorithm will represent the data to get the optimized model. K-Means is preferred in this study since the number of centroids (K) could be specified. K-Means algorithm initially proposed by MacQueen [20] in 1967. This method classifies given object to k different clusters [21].

K-Means algorithm used in this study is K-Means++. Let $\mathcal{X}$ be data points, let $\mathcal{C} = \{c_1, c_2, \ldots, c_k\}$ are centers, and let $D(x)$ denote the shortest distance from a data point to the closest center which have already chosen, as stated in Arthur and Vassilvitskii [22] the steps for K-Means++ are

(1) Take one center c_1, chosen uniformly at random from $\mathcal{X}$.
(2) Take a new center c_i, choosing $x \in \mathcal{X}$ with probability $\frac{D(x)^2}{\sum_{x \in \mathcal{X}} D(x)^2}$.
(3) Repeat step 2 until k centers have taken altogether.
(4) For each $i \in \{1, 2, \ldots, k\}$, set the cluster C_i to be the set of points in $\mathcal{X}$ that are closer to c_i than they are to c_j for all $j \neq i$.

Algorithm 2 Stochastic Levenberg-Marquardt

Input: $\theta > 0$, initial $\mathbf{p} \in \mathbb{R}^n$, data $\boldsymbol{\zeta} : (x_i, y_i), i = 1, 2, \ldots, m$
Output: $\mathbf{p}$
Initialize: $\nu = 2$, select one or few samples data randomly $\mathcal{B}_j,\ j \in \{1, 2, \ldots, m\}$, $\mathbf{A} = \mathbf{J}(\mathbf{p}, \mathcal{B}_j)^T \mathbf{J}(\mathbf{p}, \mathcal{B}_j)$, $\mathbf{g} = \mathbf{J}(\mathbf{p}, \mathcal{B}_j)^T \mathbf{f}(\mathbf{p}, \mathcal{B}_j)$, $\mu = \theta \cdot \max_i\{a_{ii}^0\}$

for $k = 1, 2, \ldots,$ maximum iteration **do**
  Solve $(\mathbf{A} + \mu \mathbf{I})\mathbf{h}_{lm} = -\mathbf{g}$
  $\mathbf{p}_{new} = \mathbf{p} + \mathbf{h}_{lm}$
  Calculate $q = (\mathbf{F}(\mathbf{p}, \mathcal{B}_j) - \mathbf{F}(\mathbf{p}_{new}, \mathcal{B}_j))/(\mathbf{L}(\mathbf{0}, \mathcal{B}_j) - \mathbf{L}(\mathbf{h}_{lm}, \mathcal{B}_j))$
  if $q > 0$ **then** ▷ Step accepted
    $\mathbf{p} = \mathbf{p}_{new}$
    Calculate $\mathbf{A} = \mathbf{J}(\mathbf{p}, \mathcal{B}_j)^T \mathbf{J}(\mathbf{p}, \mathcal{B}_j)$, $\mathbf{g} = \mathbf{J}(\mathbf{p}, \mathcal{B}_j)^T \mathbf{f}(\mathbf{p}, \mathcal{B}_j)$
    $\nu = 2$
  else
    $\mu = \mu * \nu$; $\nu = 2 * \nu$
  end if
end for

(5) For each $i \in \{1, 2, \ldots, k\}$, set c_i to be the center of mass of all points in C_i.
(6) Repeat step 4 and step 5 until there are no any changes in $\mathcal{C}$.

3. SIMULATION SETUP

3.1. **Data and Preprocessing.**

Apple's Stock Price Data (AAPL) from December 12, 1980 to May 16, 2024 was used for building the model. The dataset was taken from Yahoo! Finance with 10,948 rows of data. In this study, 5^{th} degree polynomial model $l(x) = a + bx + cx^2 + dx^3 + ex^4 + fx^5$, $x \in [0, 10947]$ is built.

In the dataset, there was a difference in x and y scale. With scale different, it caused the sensitivity for model's parameters. Wan [23] stated that to obtain better convergence using Gradient Descent, the normalization method could help. Thus, in this study, the data were normalized with *Min-Max Normalization* [24]. This method is chosen since the distribution of x and y are not normal or close to normal. Min-Max Normalization is shown in equation (12)

$$x_{\text{new}} = \frac{x - \min(x)}{\max(x) - \min(x)}. \tag{12}$$

Loss function used in this study is Mean Squared Error (MSE) which show in equation (13)

$$MSE = \frac{1}{m}\sum_{i=1}^{m} f_i(\mathbf{p})^2. \tag{13}$$

Apart from Levenberg-Marquardt simulation, Stochastic Levenberg-Marquardt with 1, 100, 200, 300, 400, and 500 randomly sampled data were simulated. For Cluster-Levenberg-Marquardt, cluster with 100, 200, 300, 400, and 500 centers were simulated. These numbers were selected after experimenting with random sample far below 100, which had similar result with 1 random sample, and above 500, which had similar result with 500 random samples.

3.2. **Parameters Setup.**

Initial setups for Levenberg-Marquardt, Stochastic Levenberg-Marquardt and Cluster-Levenberg-Marquardt are

(1) Maximum iteration = 50
In the next section, it will be shown that it does not require higher iteration since the fluctuation of loss function was similar in higher maximum iteration.

(2) $\theta = 1 \times 10^{-7}$
As a rule of thumb, small value of θ is chosen, but the algorithm is not very sensitive by the choice of θ [15].

(3) Initial $\mathbf{p} = (a, b, c, d, e, f) =$
$(-0.75276, 2.70429, 1.39196, 0.59195, -2.06389, -2.06403)$.
The initial guess is uniform randomly generated. The initial guess could be chosen by the user or randomly generated.

Computer specifications for running Python simulation program was *ASUS Desktop 12th Gen Intel(R) Core(TM) i7-12700, 2100 Mhz 12 Core(s) 20 Logical Processor(s).*

4. SIMULATION RESULTS AND DISCUSSION

The loss function comparison of Levenberg-Marquardt, Stochastic Levenberg-Marquardt, and Cluster-Levenberg-Marquardt is displayed in Figure 1. Levenberg-Marquardt has the lowest loss function since it computes the Jacobian using all sample data, as illustrated in Figure 1. It is observed that after completing 50 iterations, the loss function of the stochastic Levenberg-Marquardt with 1 sample model does not approach Levenberg-Marquardt, as shown in Figure 1a, where the loss function is significantly higher than Levenberg-Marquardt. The minimum MSE of SLM 1 sample from 50 iterations is significantly higher, as displayed in Table 1, than the minimum MSE of LM. It is suspected that additional iterations are required for loss function of SLM 1 sample to get closer to LM.

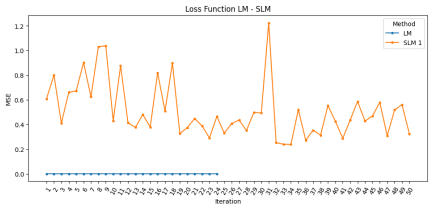
TABLE 1. Loss Function Comparison and Relative Error

Method	Minimum MSE	Relative Error (%)
LM	0.001574695911	0
SLM 1	0.239841356826	15130.96333412
SLM 100	0.001575429783	0.046604042483
SLM 200	0.001578011113	0.210529695116
SLM 300	0.001575371382	0.042895308592
SLM 400	0.001576264816	0.099632261320
SLM 500	0.001575898006	0.076338215123
Cl 100	0.001610635152	2.282297240011
Cl 200	0.001606480400	2.018452474183
Cl 300	0.001617227325	2.700928708518
Cl 400	0.001613599200	2.470527099832
Cl 500	0.001612992396	2.431992407551

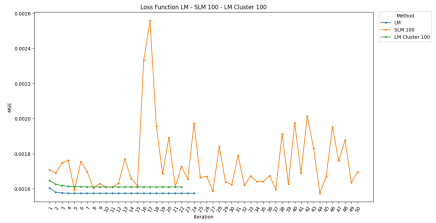
Next, Figure 1b, Figure 1c, Figure 1d, Figure 1e, and Figure 1f compares loss function for Levenberg-Marquardt, Stochastic Levenberg-Marquardt with 100 samples, and Cluster-Levenberg-Marquardt with 100 centers, and so forth. With Cluster-Levenberg-Marquardt are slightly higher than Levenberg-Marquardt. However, it never approach Levenberg-Marquardt loss function. In Table 1, the relative error of the minimum MSE of Cluster Levenberg-Marquardt to the minimum MSE of Levenberg-Marquardt are below 2.71%. On the other hand, because of the stochastic elements in iterations, in Stochastic Levenberg-Marquardt, there are some iterations has MSE are high, even higher than Cluster-Levenberg-Marquardt. Nevertheless, in certain instances, the MSE of Stochastic Levenberg-Marquardt approaches the MSE of Levenberg-Marquardt. In Table 1, the relative error of the minimum MSE of Stochastic Levenberg-Marquardt to the minimum MSE of Levenberg-Marquardt are below 0.22%. Thus, the accuracy of Stochastic Levenberg-Marquardt is superior than the Cluster Levenberg-Marquardt.

Figure 2 shows the 5^{th} degree polynomial curve with Levenberg-Marquardt, Stochastic Levenberg-Marquardt, and Cluster Levenberg-Marquardt. It is difficult to see any differences in the curve, except SLM 1 sample, as the relative error are below 2.71%.

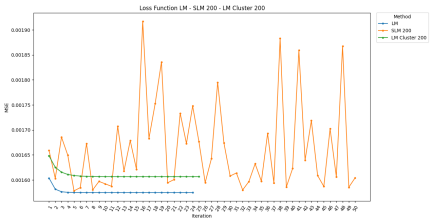
Figure 3 shows the computation time comparison among the variants of methods and Table 2 shows total iteration of optimization simulation. For Cluster Levenberg-Marquardt, merely cluster with 100 centers have a lower cumulative computation time compared to Levenberg-Marquardt. In contrast, clusters with 200, 300, 400, and 500 centers have a higher cumulative computation time since K-Means clustering requires a large amount of computation time. On the other hand, despite the fact that the total number of iterations required for the simulation reaches the maximum number, Stochastic Levenberg-Marquardt has lower computation time compared to Levenberg-Marquardt.



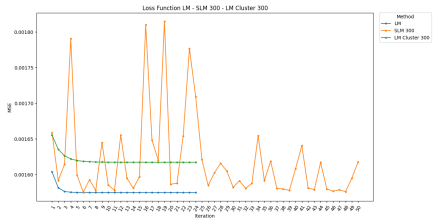
(A) Loss Comparison LM - SLM 1 sample



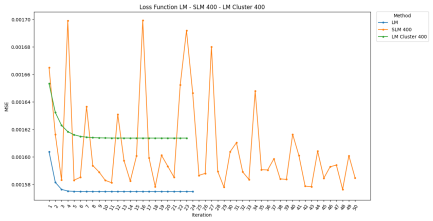
(B) Loss Comparison LM - SLM 100 samples - Cluster LM with 100 centers



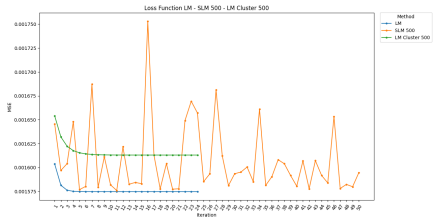
(C) Loss Comparison LM - SLM 200 samples - Cluster LM with 200 centers



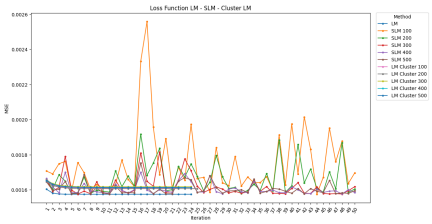
(D) Loss Comparison LM - SLM 300 samples - Cluster LM with 300 centers



(E) Loss Comparison LM - SLM 400 samples - Cluster LM with 400 centers



(F) Loss Comparison LM - SLM 500 samples - Cluster LM with 500 centers



(G) Loss Comparison LM - SLM - Cluster LM

FIGURE 1. Loss Comparison of Levenberg-Marquardt, Stochastic Levenberg-Marquardt, and Cluster-Levenberg-Marquardt

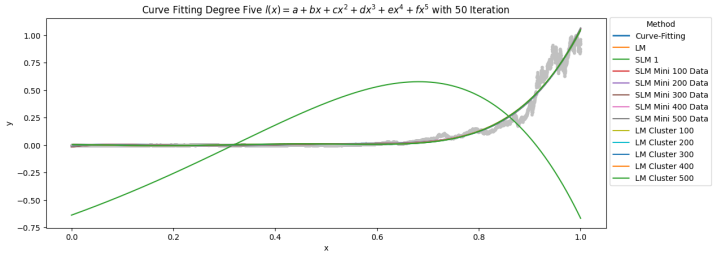


FIGURE 2. Curve of 5^{th} Degree Polynomial Model Comparison

TABLE 2. Iteration Comparison

Method	Total Iteration
LM	24
SLM 1	50
SLM 100	50
SLM 200	50
SLM 300	50
SLM 400	50
SLM 500	50
Cl 100	22
Cl 200	25
Cl 300	24
Cl 400	23
Cl 500	24

In general, the computation time of Stochastic Levenberg-Marquardt and Cluster-Levenberg-Marquardt (excluding K-Means computation) are lower than Levenberg-Marquardt as shown in Figure 3. Reducing computational time comes at the cost of accuracy, as these methods typically use smaller sample sizes. Smaller sample sizes contributes in faster computational cost for calculating Jacobian matrix, gradient, and Hessian matrix. Although computational time is reduced, the relative error compared to the Levenberg-Marquardt method remains low as the relative error are below 0.22% for the Stochastic Levenberg-Marquardt approach and below 2.71% for Cluster-Levenberg-Marquardt. With a small relative error compared to the Levenberg-Marquardt method and reduced computation time, this research demonstrates that Stochastic Levenberg-Marquardt outperforms to Levenberg-Marquardt and Cluster-Levenberg-Marquardt in computational efficiency.

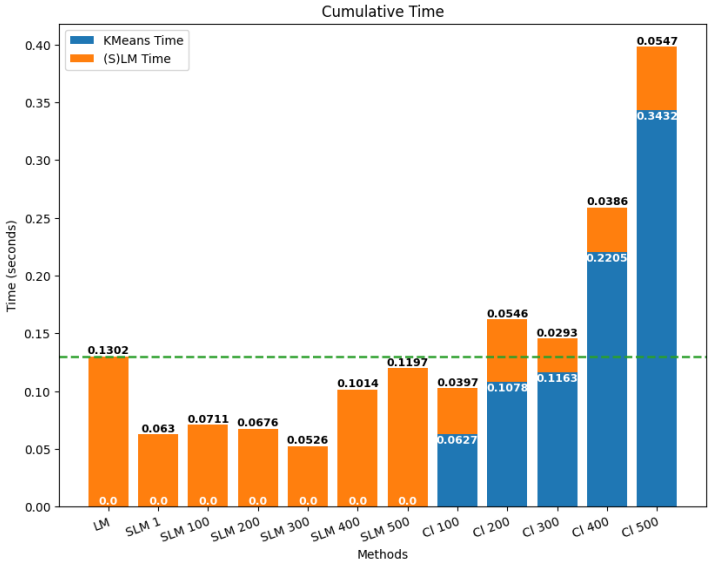


FIGURE 3. Computation Time Comparison

5. CONCLUSION

This research aims to investigate the influence of stochastic in the Levenberg-Marquardt numerical method and sampling using cluster centers on accuracy and computational efficiency. Stochastic Levenberg-Marquardt with 100, 200, 300, 400, and 500 have relative error to Levenberg-Marquardt below 0.22% and have lower computational time in 50 maximum iteration. Cluster Levenberg-Marquardt with 100, 200, 300, 400, and 500 centers have relative error to Levenberg-Marquardt below 2.71% and have higher computational time, except 100 centers. To summarize, Stochastic Levenberg-Marquardt with 100, 200, 300, 400, and 500 outperformed Levenberg-Marquardt and Cluster-Levenberg-Marquardt with very low relative error and reduced computational time, which suggests that using Stochastic Levenberg-Marquardt approach is advantageous. Finally, further research may include a better stopping criterion for Stochastic Levenberg-Marquardt to stop before maximum iteration, applying the methods in bigger datasets, and exploring other clustering methods for better results in Levenberg-Marquardt.

Acknowledgement. This work is supported by Department of Mathematics, Faculty of Mathematics and Natural Sciences, Institut Teknologi Bandung.

REFERENCES

[1] D. Kapadiya, C. Shekhawat, and P. Sharma, "A study on largescale applications of big data in modern era," in *Proceedings of the 5th International Conference on Information Management & Machine Intelligence*, ACM, 2023. https://doi.org/10.1145/3647444.3647880.

[2] E. Horvitz and T. Mitchell, "From data to knowledge to action: A global enabler for the 21st century," 2020. https://doi.org/10.48550/ARXIV.2008.00045.

[3] J. M. Hokanson, "Projected nonlinear least squares for exponential fitting," *SIAM Journal on Scientific Computing*, vol. 39, no. 6, pp. A3107–A3128, 2017. https://doi.org/10.1137/16M1084067.

[4] G. Grisetti, T. Guadagnino, I. Aloise, M. Colosi, B. Della Corte, and D. Schlegel, "Least squares optimization: From theory to practice," *Robotics*, vol. 9, no. 3, p. 51, 2020. https://doi.org/10.3390/robotics9030051.

[5] Y.-X. Yuan, "Recent advances in numerical methods for nonlinear equations and nonlinear least squares," *Numerical Algebra, Control & Optimization*, vol. 1, no. 1, pp. 15–34, 2011. https://doi.org/10.3934/naco.2011.1.15.

[6] J. Nocedal and S. J. Wright, *Numerical Optimization*. Springer, 2006. https://doi.org/10.1007/978-0-387-40065-5.

[7] H. Robbins and S. Monro, "A stochastic approximation method," *The Annals of Mathematical Statistics*, vol. 22, no. 3, pp. 400–407, 1951. https://doi.org/10.1214/aoms/1177729586.

[8] J. Kiefer and J. Wolfowitz, "Stochastic estimation of the maximum of a regression function," *The Annals of Mathematical Statistics*, vol. 23, no. 3, pp. 462–466, 1952. https://doi.org/10.1214/aoms/1177729392.

[9] E. H. Bergou, Y. Diouane, V. Kungurtsev, and C. W. Royer, "A stochastic levenberg–marquardt method using random models with complexity results," *SIAM/ASA Journal on Uncertainty Quantification*, vol. 10, no. 1, pp. 507–536, 2022. https://doi.org/10.1137/20M1366253.

[10] R. S. V. Kumar, M. D. Alsulami, I. E. Sarris, G. Sowmya, and F. Gamaoun, "Stochastic levenberg–marquardt neural network implementation for analyzing the convective heat transfer in a wavy fin," *Mathematics*, vol. 11, no. 10, p. 2401, 2023. https://doi.org/10.3390/math11102401.

[11] D. E. Budil, S. Lee, S. Saxena, and J. H. Freed, "Nonlinear-least-squares analysis of slow-motion epr spectra in one and two dimensions using a modified levenberg–marquardt algorithm," *Journal of Magnetic Resonance Series A*, vol. 120, no. 2, pp. 155–189, 1996. https://doi.org/10.1006/jmra.1996.0113.

[12] M. Lampton, "Damping–undamping strategies for the levenberg–marquardt nonlinear least-squares method," *Computers in Physics*, vol. 11, no. 1, pp. 110–115, 1997. https://doi.org/10.1063/1.168600.

[13] Z. Yan, S. Zhong, L. Lin, and Z. Cui, "Adaptive levenberg–marquardt algorithm: A new optimization strategy for levenberg–marquardt neural networks," *Mathematics*, vol. 9, no. 17, p. 2176, 2021. https://doi.org/10.3390/math9172176.

[14] Y. Hong *et al.*, "Stochastic levenberg–marquardt for solving optimization problems on hardware accelerators," in *IEEE Conference Proceedings*, IEEE, 2020. https://repository.kaust.edu.sa/handle/10754/666072.

[15] K. Madsen, H. B. Nielsen, and O. Tingleff, *Methods for Non-Linear Least Squares Problems*. 2 ed., 2004.

[16] S. Ruder, "An overview of gradient descent optimization algorithms," 2016. https://doi.org/10.48550/ARXIV.1609.04747.

[17] Y. Tian, Y. Zhang, and H. Zhang, "Recent advances in stochastic gradient descent in deep learning," *Mathematics*, vol. 11, no. 3, p. 682, 2023. https://doi.org/10.3390/math11030682.

[18] G. Garrigos and R. M. Gower, "Handbook of convergence theorems for (stochastic) gradient methods," 2023. https://doi.org/10.48550/ARXIV.2301.11235.

[19] W.-Y. Shao and J.-Y. Fan, "Global convergence of a stochastic levenberg–marquardt algorithm based on trust region," *Journal of the Operations Research Society of China*, 2024. https://doi.org/10.1007/s40305-023-00529-6.

[20] J. B. MacQueen, "Some methods for classification and analysis of multivariate observations," in *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability*, vol. 1, pp. 281–297, 1967. http://digitalassets.lib.berkeley.edu/math/ucb/text/math_s5_v1_article-17.pdf.

[21] S. Na, L. Xumin, and G. Yong, "Research on k-means clustering algorithm: An improved k-means clustering algorithm," in *2010 Third International Symposium on Intelligent Information Technology and Security Informatics*, pp. 63–67, IEEE, 2010. https://doi.org/10.1109/IITSI.2010.74.

[22] D. Arthur and S. Vassilvitskii, "k-means++: The advantages of careful seeding," in *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, pp. 1027–1035, 2007. https://doi.org/10.5555/1283383.1283494.

[23] X. Wan, "Influence of feature scaling on convergence of gradient iterative algorithm," *Journal of Physics: Conference Series*, vol. 1213, no. 3, p. 032021, 2019. https://doi.org/10.1088/1742-6596/1213/3/032021.

[24] S. C. Nayak, B. B. Misra, and H. S. Behera, "Impact of data normalization on stock index forecasting," *International Journal of Computer Information Systems and Industrial Management Applications*, vol. 6, pp. 357–369, 2014.