An ANFIS-Based High Precision Error Iterative Analysis Method (HPEIAM) to Improve Existing Software Reliability Growth Models

Gul Jabeen[1,2], Sabit Rahim[1,*], Gul Sahar[1], Luo Ping[2]
[1]Karakoram International University Gilgit, Pakistan
[2]Tsinghua University, Beijing, China
***Corresponding Author**: Sabit Rahim, sabit.rahim@kiu.edu.pk

Citation |Jabeen G., Rahim S, Sahar G, Ping L, "An ANFIS-Based High Precision Error Iterative Analysis Method (HPEIAM) to Improve Existing Software Reliability Growth Models", IJIST, Vol 6, Issue 4. pp 1878-1896, Nov 2024
DOI | https://doi.org/10.33411/IJIST/1091
Received |Oct 14, 2024; **Revised** | Nov 18, 2024; **Accepted** | Nov 19, 2024; **Published** | Nov 20, 2024.

Software Reliability Growth Models (SRGMs) are statistical interpolations of software failures by mathematical modeling. Up till now, more than 200 SRGMs have been proposed to estimate failure occurrence. Research continues to develop more accurate, efficient, and robust models. To overcome the shortcomings of SRGMs and adapt to the current software development process characterized by increasing complexity, a high-precision error iterative analysis method (HPEIAM) is proposed in this paper. HPEIAM combines the parametric SRGMs (PSRGMs) predicted results with their residual errors, which are considered as another source of information that can be modeled with an adaptive neuro-fuzzy inference system (ANFIS). The predicted errors are used to correct the PSRGMs forecasted results repeatedly with the help of ANFIS, which is considered a powerful model to deal with non-linear data. The proposed technique combines the advantages of the neural network with a fuzzy inference system and PSRGMs, which helps to overcome the disadvantages of these models. The performance of the proposed technique is compared with six PSRGMs using three sets of real software failure datasets based on five criteria. Experimental results demonstrate that the HPEIAM can significantly improve the model fitting and predictive performance of every parametric SRGM.

Keywords: Software Reliability, Software Failures, Residual Errors, Artificial-Neuro-Fuzzy-Inference System, Parametric Software Reliability Growth Models, Prediction Accuracy

Introduction:

Software is becoming an increasingly important part of critical and non-critical applications. Ensuring software quality is becoming a key issue. The reliability of software is generally considered the most important attribute [1][2] of software quality measurement. Estimating and predicting software reliability is now increasingly required for highly reliable systems.

Researchers and scholars have developed PSRGMs where the researchers [3][4] mentioned the most reliable and practical method for software reliability estimation using SRGMS models during the test phase. The cost optimization and time of the software development are not separate from the reliability of the software. When software failures are detected in a later stage, the cost and time increase. Before the testing phase, the accurate and concise estimation of software reliability might minimize software development costs and improve software quality effectively. PSRGM requires the application of basic assumptions such as independence of failure time, immediate correction of detected failures, and failure correction without introducing new errors. In addition, any software development process is environmentally dependent, and some assumptions seem unrealistic in certain situations. Many PSRGMs analyze the application of software reliability during the testing phase[5][6]. However, due to unrealistic assumptions, there is no single model that could be widely used in all situations. The [6][7] in the literature proposed non-parametric approaches to address these problems. These models address the problem of unrealistic assumptions and applicability of the issues but fail to predict accurately.

Recently, some authors [8][9] tried to use nonparametric hybrid techniques like ANFIS, which is a combination of Artificial Neural Networks (ANN) and fuzzy logic. It provides an effective technique for modeling non-linear (chaotic) data sequences. It is also called a non-parametric technique since it does not use any predefined assumption for modeling events. ANFIS uses the power of the two patterns such as ANN and FIS in a single framework and overcomes the disadvantages of both models. Studies showed that neuro-fuzzy systems are more effective in handling a large amount of noisy data or random data sequences [10].

In this research, a well-established prediction technique called HPEIAM based on ANFIS has been presented, which addresses the limitations of PSRGMs by using their residual errors as another source of information to a non-parametric approach like ANFIS. Because of a hybrid approach, this technique overcomes the problems associated with PSRGMs that come from restrictive assumptions. In addition, this approach uses the residual errors from the estimated results of PSRGMs as a new source of information to optimize the prediction accuracy of PSRGMs. The residual errors also form a non-linear (chaotic) data sequence.

The proposed method's performance has been measured in conjunction with the traditional SRGMs by using well-known datasets. The result of the experiment has been derived from the three datasets which illustrate that the proposed technique can better fit the failure data and provide a more accurate prediction of the remaining failures in software testing. Our method improves the accuracy of every selected PSRGM model and provides more accurate predictions. It overcomes the disadvantages of PSRGMs and uses the advantages of all ANFIS models. The purpose of this study propose a new optimized technique to improve the performance of SRGMs. From our analysis of the experiment, we show that the residual errors iterative modification can improve the prediction accuracy up to an expected level, no matter how much historical data is used. Our results demonstrate that our technique can work well with every set of data and improves the performance accuracy of every software reliability growth model by using residual error modification.

Objective of the study:

The objective of this research is to propose a novel High Precision Error Iterative Analysis Method (HPEIAM) to enhance the accuracy and performance of parametric software

reliability growth models (PSRGMs). The proposed method aims to address the limitations of existing PSRGMs by incorporating their residual errors as a new source of information to the ANFIS to iteratively correct the predicted results towards making more accurate results for software failures.

Novelty statement:

The research presents a novel residual error iterative method that uses the residual errors from the PSRGMs as an additional source of information to Non-PSRGMs like Artificial Neuro-Fuzzy Inference Systems. However, the proposed method combines the residual error modifications iteratively by using PSRGMs and up to the expected level of the ANFIS modeling to acquire high prediction accuracy. An ideal PSRGM would not deal with residual errors directly because there may be many complex and random fluctuated (positive and negative) signals in them, which are not directly incorporated in PSRGMs. Therefore, ANFIS is used to predict the residual errors. It is considered a more powerful tool to deal with non-linear data sequences [8]. However, by mathematical modeling of known data sequences and repeated computation of error values, the prediction accuracy is improved and an expected level of precision can be achieved.

Related Work:

Software reliability is commonly considered the most significant factor of software quality attributes over the past few decades. The demand for estimation and prediction of software reliability is increasing in projects to get reliable software [3]. PSRGMs are based on underlying distribution assumptions. Statistical methods such as regression, and non-homogeneous Poisson process models are used in these models. These models are grouped into the time between failure models and failure count models. During the last few years, many PSRMs have been proposed. We are presenting some of them here.

Although the wide usage of parametric approaches imposed several restrictions, they cannot retain their accuracy across different projects. These models require underlying assumptions to be used. However, in many cases, these assumptions look questionable and unrealistic. There is no single model which can be universally used in all situations [11]. Therefore, various models are applied, and the efficient model is selected according to the given situation. This process takes more time and requires more experts to select the best model. The ability to predict future reliability values using the PSRGMs is also considered the main goal using the selected data. It is usually in the form of time between failures or the number of failures. Although PSRGMs fit past failure data well, however, they do not give more accurate future predictions [12]. Much research has been carried out to address the PSRGMs issues. The non-PSRGMs are getting attention from researchers, in these models machine learning techniques used such as neural networks [4][5][13][14] support vector machines [15] Genetic Algorithms (GAs) [6][16][17]. The software reliability problems are solved by using GAs. An analytical model has been proposed using a stochastic differential equation to predict the reliability of the project[18].

These models address the issues of unrealistic assumptions and applicability. However, they fail to remedy the issue of predictability. The fuzzy logic is ranked as significant in non-parametric approaches [19][20]. The accuracy of SRGMs is an important factor, no matter which type of model or method is used and how much data is used by the model to achieve an expected accuracy level is a significant factor for any model. Therefore, it is essential to develop a reliability model for software, that can use the data effectively and give accurate prediction outcomes. In this paper, we have proposed a well-established new combined

parametric and non-parametric HPEIAM techniques, which can help software engineers to construct the high-precision prediction model more easily. The proposed technique is termed as HPEIAM. Mathematical modeling and repeated computation of residual error values can enhance the accuracy of prediction up to a projected level.

Proposed Work:

The primary purpose of SRGMs models is to achieve precise prediction accuracy. We claim that our proposed method (HPEIAM) can enhance the accuracy of prediction of PSRGMs effectively. In the following section, our proposed method is described in detail.

High Precision Error Iterative Analysis Method (HPEIAM):

To simplify the explanation of our method, we have introduced some notations first. Assume that a set of known data $x_1(t), x_2(t), x_3(t), \ldots, x_n(t)$ is used to develop a mathematical model such as:

$$y = f(x_1(t), x_2(t), \ldots, x_n(t), a_1, a_2, \ldots, a_n, t_1, t_2, \ldots, t_s) \quad (1)$$

Where $a_1, a_2, \ldots, a_n$ are parameters and $t1, t_2, \ldots, ts$ are specific variables. Specifically, we predict the future trends using these mathematical models: $x_{n+1}(t), x_{n+2}(t), x_{n+3}(t), \ldots, x_{n+k}(t)$.

The $a_1, a_2, a_3, \ldots, a_n$ are unknown parameters which can be determined by using any input sequences $x_1(t), x_2(t), \ldots, x_n(t)$ with multiple methods including the least square method or the maximum likelihood method. For convenience, this paper discusses only one argument which is denoted as $x(t)$. We can write the model (1) as follows:

$$y = f(x_1(t), x_2(t), \ldots, x_n(t), a_1, a_2, \ldots, a_r, t). \quad (2)$$

After applying the input data sequences $x_1(t), x_2(t), \ldots, x_n(t)$ in any SRGM, we determine the first approximation solution and its predicted values as $\overline{x1(t)}^{(1)}, \overline{x2(t)}^{(1)}, \ldots, \overline{xn(t)}^{(1)}$, The exact values corresponding to the predicted values are assumed to be $x_{n+1}(t)^{(1)}, x_{n+2}(t)^{(1)}, \ldots, x_{n+l}(t)^{(1)}$

Suppose the error values can be specified as $\varepsilon_n{}^{(1)}(t) = x_n(t) - \overline{xn(t)}^{(1)} (n = 1,2, \ldots)$, which is referred to as the first error. However, the acquired errors $\varepsilon_n{}^{(1)}(t)$ can be positive or negative based on the predicted values.

For more clarity, we ignore t, such as $x_1(t)^{(1)}, \overline{x2(t)}^{(1)}, \varepsilon_1{}^{(1)}(t)$ and (1) are abbreviated as $x_1{}^{(1)}$, $\overline{x2(t)}^{(1)}, \varepsilon_1{}^{(1)}$and y = f(t) respectively. In the proposed technique, the error value $\varepsilon_n{}^{(1)}$ has been used to get more accurate results than previously predicted solution $x_{n+I}{}^{(1)} (i = 1,2, \ldots, l)$ through multiple iterations. Therefore, we analyze the error data$\varepsilon_1{}^{(1)}, \varepsilon_2{}^{(1)}, \ldots, \varepsilon_n{}^{(1)}$ same as $x_i{}^{(1)} (i = 1, 2, \ldots, n)$ by using a non-parametric model such as in this case we have used an ANFIS to model the errors. So the model can be established as:

$$y = f(\varepsilon_1{}^{(1)}, \varepsilon_2{}^{(1)}, \ldots, \varepsilon_n{}^{(1)}). \quad (3)$$

The $\varepsilon_i{}^{(1)} (i = 1, 2, \ldots, l)$forms a complex sequence (random positive and negative values) that is impossible to incorporate with PSRGMs. Therefore, we used Adaptive Neuro-Fuzzy Interface System (ANFIS). ANFIS is considered an appropriate model to deal with non-linear (chaotic) data sequences. We modeled the residual errors using ANFIS and predicted the expected errors for the SRGMs. By modeling Equation (3) in ANFIS, we obtained residual error approximate and predicted solution $\overline{\varepsilon 1}^{(1)}, \overline{\varepsilon 2}^{(1)}, \ldots, \overline{\varepsilon n}^{(1)}$and $\overline{\varepsilon n + 1}^{(1)}, \overline{\varepsilon n + 2}^{(1)}, \ldots, \overline{\varepsilon n + 1}^{(1)}$ respectively. The detailed residual error approximation modeling with ANFIS is explained in Section III-C.

Hence it is obvious to get the second approximate solution $\overline{x1}^{(1)}, \overline{x2}^{(1)} + \overline{\varepsilon n}^{(1)}(i = 1,2,\dots,n)$, by adding the 1st approximate solution (modeled with SRGMs) and the first error approximate solution (modeled with ANFIS). We observe that $\overline{xi}^{(2)}$ is the more closed approximate solution than $\overline{xi}^{(1)}(i = 1,2,\dots,n)$to the exact solution$x_i(i = 1,2,\dots,n)$.

Likewise,$\varepsilon_i{}^{(2)} = x_i - \overline{xi}^{(2)}(i = 1,2,\dots,n)$, it is known as the 2nd error. Using the same method as defined above, we can get an approximate solution to the 2nd error $\overline{\varepsilon i}^{(2)}(i = 1,2,\dots,n)$ $\varepsilon_i{}^{(2)}$ and the 3rd approximation solution $\overline{xi}^{(3)} = \overline{xi}^{(2)} + \overline{\varepsilon i}^{(2)}(i = 1,2,\dots,n)$.

Hence, by continuing the above multiple error iterative process, we obtain the predicted values closer to the exact values $x_i(i = 1,2,\dots,l)$. We get the k^{th} approximate solution $x_i{}^{(k)}(i = 1,2,\dots,l,k = 1,2,\dots,m)$, predicted solution $\overline{xn+1}^{(k)}(i = 1,2,\dots,l,k = 1,2,\dots,m)$, error sequence $\varepsilon_i{}^{(k)}(i = 1,2,\dots,l,k = 1,2,\dots,m)$ and its error approximation solution sequences$\overline{\varepsilon i}^{(k)}(i = 1,2,\dots,l,k = 1,2,\dots,m)$.

At the same time, we estimate $\overline{xi}^{(k)} = \overline{xi}^{(k-1)} + \overline{\varepsilon i}^{(k-1)}$ and $\varepsilon_i{}^{(k)}i = x_i - \overline{\varepsilon i}^{(k)}(i = 1,2,\dots,n,k = 1,2,\dots,m)$ repeatedly.

Basic Theorem:

The prediction accuracy of any SRGM depends on the residual error modification. If the residual errors are predicted accurately, they can be used to enhance the prediction accuracy to an expected level. In this section, Theorem 1 describes that mathematical modeling and multiple iterations analysis of residual errors obtained from the mathematical models can greatly improve the prediction accuracy of the model or help to achieve the expected optimized results.

Theorem 1:

Assume a known data sequence $x_i(i = 1,2,\dots,l)$ which can be determined by function $y = f(x_1, x_2, \dots, xn, a1, a2, \dots, ar, t)$ By using our proposed technique (HPEIAM), *mth* predictive value $\overline{xn+i}^{(m)}$ can be obtained which is more close to the exact solution $xn + i(i = 1,2,\dots,l)$ than $\overline{xn+1}^{(1)}$.

Proof. For any value n, the following common inequalities exist:

$$0 \le |xn - \overline{xn}^{(m)}| = |\varepsilon n^{(m)}| \le |\overline{\varepsilon n}^{(m-1)}| \le, \dots, \le |\varepsilon n^{(1)}| = |xn - \overline{xn}^{(1)}xn| \quad (4)$$

Therefore, we consider $M = max\left\{|xi - \overline{xi}^{(1)}|i = 1,2,\dots,n\right\}$ and analyze the error sequence$\varepsilon i^{(k)}(i = 1,2,\dots,n,k = 1,2,\dots,m)$. To obtain the value of $\varepsilon n^{(k)}$ and $\overline{xn}^{(m)}$, following equations hold:

$$\varepsilon n^{(1)} = xn - \overline{xn}^{(1)},$$
$$\varepsilon n^{(2)} = xn - \overline{xn}^{(2)} = xn - (\overline{xn}^{(1)} + \overline{\varepsilon n}^{(1)}) = \varepsilon n^{(1)} - \overline{\varepsilon n}^{(1)}$$
$$\varepsilon n^{(3)} = xn - \overline{xn}^{(3)} = xn - (\overline{xn}^{(2)} + \overline{\varepsilon n}^{(2)}) = \varepsilon n^{(2)} - \overline{\varepsilon n}^{(2)}$$

By continuing the same process, we get: $\varepsilon n^{(m)} = \varepsilon n^{(m-1)} - \overline{\varepsilon n}^{(m-1)}$

Now we show that equation (4) is true when $m \to \infty$, and $\varepsilon n^{(m)} \to 0$. Therefore from$\varepsilon n^{(m)} = xn - \overline{xn}^{(m)}$, we found that the approximate solution $\overline{xn+1}^{(m)}(i = 1,2,\dots,l)$ is more closer to exact solution $xn + i(i = 1,2,\dots,l)$ than $\overline{xn+i}^{(1)}(i = 1,2,\dots,l)$ Since

$$0 \leq |\varepsilon n^{(1)}| = |xn - \overline{xn}^{(1)}| \leq M,$$

Then we obtain

$$0 \leq |\varepsilon n^{(2)}| = |xn - \overline{xn}^{(2)}| = |\varepsilon n^{(1)} - \overline{\varepsilon n}^{(1)}| \leq |\varepsilon n^{(1)}| \leq M,$$

So the following inequality can be obtained:

$$0 \leq |\varepsilon n^{(m)}| = |\overline{\varepsilon n}^{(m-1)}| \leq, \ldots . |\varepsilon n^{(1)}| \leq M,$$

As defined in equation 4, when $m \to \infty$ and $\varepsilon n^{(m)} \to 0$

Residual Error Estimation using ANFIS:

The above-mentioned HPEIAM technique improves the accuracy of any prediction model and provides an optimal solution. The residual errors are responsible for the prediction accuracy of the given model. The errors approximation $\varepsilon i^{(k)}$ are responsible for the prediction accuracy of every predicted value. We have used a more powerful tool to model the error approximations such as ANFIS. The ANFIS was introduced by Jang[21]. It is a powerful (ANN) and Fuzzy Inference System (FIS). Combining the ANN and FIS can overcome the disadvantages and provide the advantages of both techniques. ANFIS uses the ability of ANN's to classify the data and identify the patterns. Consequently, ANFIS has several advantages, including its adaptive capability, nonlinear ability, and rapid learning capacity. The are many types of fuzzy interference systems. The segeno-type FIS is used in this study because it is more computationally efficient than other types. To explain the ANFIS structure, we assumed that there are two inputs: e_1 and e_2. The two fuzzy if-then rules for a first-order Surgeno fuzzy model are written as follows:

- $Rule: if\ e1\ is\ A1\ and\ e2\ is\ B1\ then\ f1 = p1e1 + q1e1 + r1,$
- $Rule: if\ e1\ is\ A2\ and\ e2\ is\ B2\ then\ f2 = p2e1 + q2e1 + r2$

Similarly A_i and B_i are fuzzy sets, f_i is the output, and pi, qi and ri are the design parameters which can be obtained through the training process. The ANFIS architecture is in Figure 1.

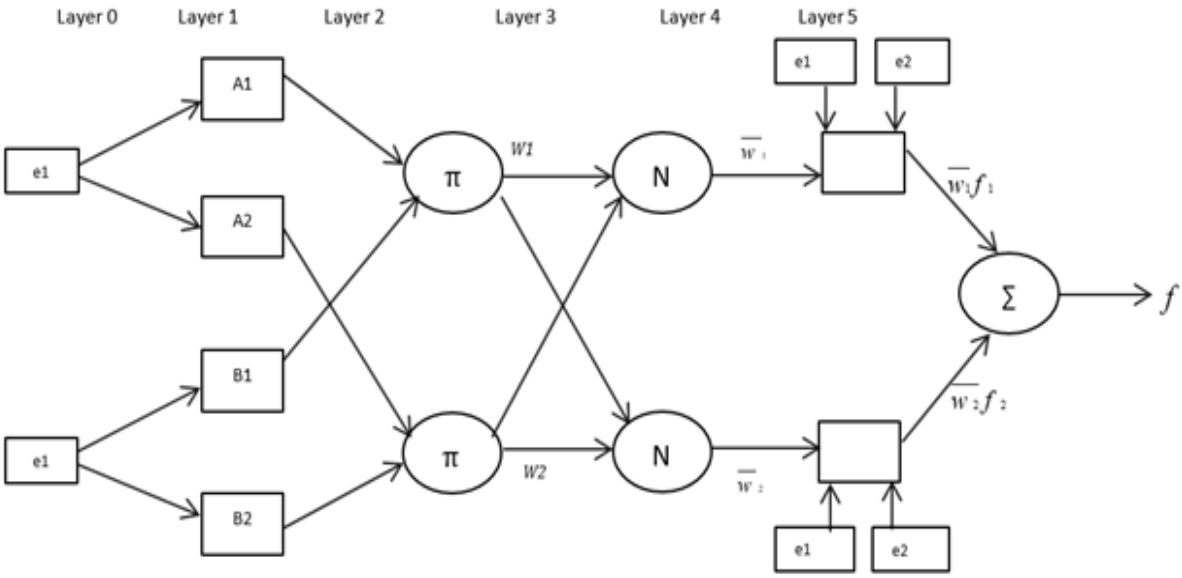


Figure 1. Graphical Representation of Adaptive Neuro-Fuzzy Inference System (ANFIS) [21]

- **Layer 0:** Enable inputs to the system
- **Layer 1:** This layer fuzzify the inputs by using specific fuzzy sets of number and membership functions
- **Layer 2:** In this layer, the specific node for every rule has been added and each node computes the firing strength of a rule by multiplication.

- **Layer 3:** In this layer, the previously calculated firing strengths are normalized.
- **Layer 4:** It fires the rules using the normalized strength of each rule.
- **Layer 5:** In this layer, the result is calculated by adding all the outputs coming from layer 4.

An ANFIS model uses two learning algorithms, backpropagation, and hybrid methods, which are used to minimize the error between the observed and estimated data [22].

Algorithm Description:

To provide a clear description of the proposed technique, we specify the subsequent algorithm.

Algorithm 1 High Precision error iterative analysis algorithm

Input: $x1, x2, \ldots, xn, \delta \geq 0$

Create a model: $y = f(x1, x2, \ldots, xn, a1, \ldots . ar, t)$

Get the approximation solution: $\overline{xi}^{(1)}, (i = 1, 2, \ldots, n)$

Get error values: $\varepsilon i^{(1)} = xi - \overline{xi}^{(1)} (i = 1, 2, \ldots, n)$

0: $for\ i = 1\ to\ m\ do$

01: $y = f(\varepsilon 1^{(i)}, \varepsilon 2^{(i)}, \ldots, \varepsilon n^{(i)})$

02: $if\ \varepsilon^{(i)} = \left\{ \sum_{j-1}^{n} | xi - \overline{xj}^{(1)} | \right\} \leq \delta\ then$

03: show the predicted solution

04: $\overline{xn+1}^{(m)} = \overline{xn+1}^{(m-1)} + \varepsilon n + i^{(m-1)}$

05: Exit

06: else if For some other condition then

07: Repeat for loop

08: end if

09: end for =0

The flow diagram illustrates the methodology used in the proposed approach.

Data Input:

The proposed approach begins with providing failure data based on time for the Parametric Software Reliability Growth Model (PSRGM). The input data becomes the basis of producing predicted values concerning reliability in software.

Application of PSRGM:

The PSRGM takes the input data and generates predicted values. The software reliability growth models predicted values and actual values which gives residual errors $\varepsilon i^{(k)} (i = 1,2, \ldots . , n, k = 1,2, \ldots , m)$ are another source of information that we can model. These residual errors form nonlinear data sequences.

Expected Accuracy:

It is used to obtain the model's accuracy up to an expected level by using specific criteria parameters such as MSE. The repeated computation of residual errors by SRGMs improves and corrects the prediction accuracy up to the expected level.

Application of ANFIS:

If the expected accuracy is not met, the residual error value will pass on to the ANFIS to get the residual error approximation. This approximated error will then be passed on to the PSRGM to refine the predictions and come as close to the expected accuracy level as possible. The predicted failure value ANFIS is used for assessing PSRGMs predicted values residual errors. We have modeled residual error series using the ANFIS model. ANFIS is considered a powerful tool to model the nonlinear data series[22] and deals with both negative and positive sign values. The ANFIS function is used to ANFIS from the Matlab Fuzzy Logic Toolbox. To

model the residual error data in ANFIS, the two input variables are selected. The first input (e_i) denotes the error number. The second input (e_2) contains residual error series. The ANFIS model is used for training the FIS model to assess training data presented to it by altering the parameters of the membership function according to the particular criterion. After training the ANFIS model, the outputs are predicted for the trained data. In the next step, prediction is also calculated using the trained ANFIS model.

The HPEIAM is demonstrated through a flow diagram in Figure 2.

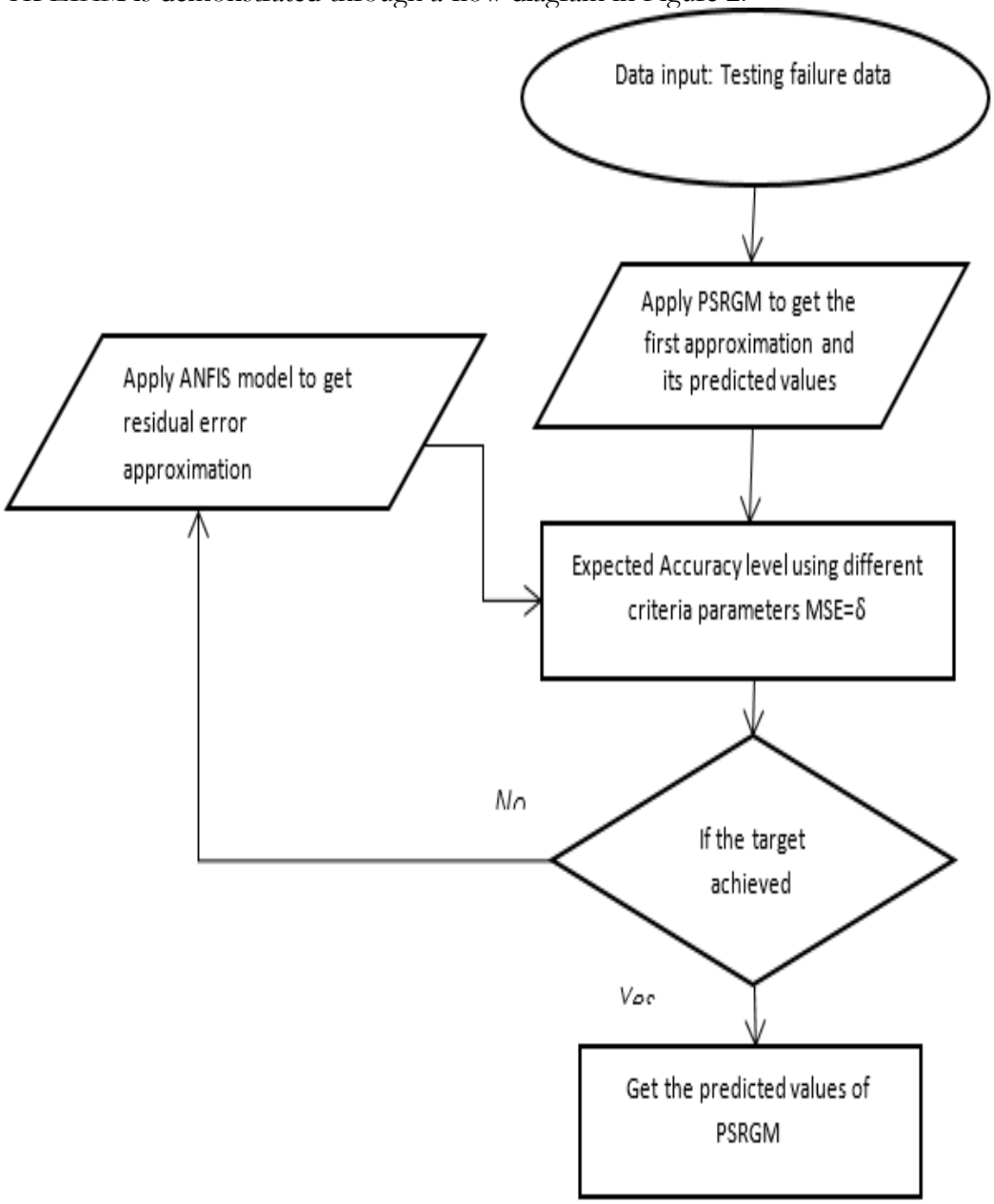


Figure 2. Flow diagram of HPEIAM using ANFIS technique

Get Final Predicted values:

By continuing the aforementioned multiple error iterative process, we obtained the predicted values closer to the exact values The results of the predicted value, as calculated through PSRGM, and modified residual errors by ANFIS will be added to give a better forecasting.

Numerical Illustration: To denote the efficacy of the proposed method, we used six classical PSRGMS as shown in Table 1. The HPEIAM is applied for each model to achieve a more precise and optimal solution. In this study, all model parameters are estimated by the maximum likelihood estimation, as compared with the least square method, which can provide unbiased results.

Data Description:

To validate the proposed model results, we chose three different datasets collected for the real-time command and control systems in actual software development projects. The first dataset (DS-1) is 38 fault-detection given in [23] observed during 14 weeks. The second dataset (DS-2) is collected from a web-based and integrated accounting ERP system [24]. There are 146 errors detected within 60 months. The third dataset is also given in [23]. It was reported by Musa, Iannino, and Okumoto based on failure data from the Real-time command and Control system (RTC and CS), which observed failures through system testing for 25 hours of CPU time.

Performance Evaluation Criteria:

To better illustrate the model's performance of our proposed method and other PSRGMs, we evaluated based on two main criteria: goodness-of-fit and predictive power.

Goodness-of-fit criterion:

We use four goodness-of-fit criteria commonly used to evaluate the performance of SRMs. The criteria are described as follows

Table 1. Parametric Software Reliability Growth Models

Models Name	Mean value function	Description
Geol-Okumoto NHPP model [11]	$m(t) = a(1 - \overline{e}^{bt})$	It is called an exponential software reliability model.
Delayed S-shaped model[8]	$m(t) = a(1(1 + bt)e^{(bt)}$	A change of NHPP model to make it shaped
Schneidewind model[11]	$m(t) = \frac{a}{b}(1 - \overline{e}^{bt})$	This model incorporates the idea that the current fault rate might be of higher importance than the distant past.
Weibull model [9]	$m(t) = a(1 - (1 + \frac{t}{b})^{1-\alpha}$	Depending on the value of the shape parameter, it can take the characteristics of other types of distributions.
Pham and Zhang Model [10]	$m(t) = \frac{1}{(1 + \beta\, exp^{-bt})}((c + a)(1 - exp - bt) - \frac{ab}{b - a}(\overline{exp^{\alpha}\, exp -}$	It assumes the fault introduction rate as an exponential function of the testing time and the fault detection rate as non-decreasing.
Chang et al. 's model[25]	$m(t) = a(1 - (1 + \frac{t}{\beta})^{1-\alpha}$	It assumes fault content as constant and integrate the testing coverage function into the software reliability model.

- The mean square error (MSE) measures the deviation between the actual and predicted values, defined as[26]:

$$MSE = \frac{1}{n-N}\sum_{i=1}^{n}(actuali - predicti)^2$$

Where *n* and N denote the number of predicted values and the parameters respectively.

- The R square (R^2) indicates the correlation index of the regression curve which is denoted as [27]:

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(actuali - predicti)^2}{\sum_{i=1}^{n}(actuali - predicti)}$$

- The predictive power (PP) calculates the difference of the prediction from the real data with the predicted data [10]:

$$PP = \sum_{i=1}^{n}(\frac{predicti - atucali}{actauli})^2$$

The predictive ratio risk (PPR) calculates the difference between the prediction from the actual data against the estimated values and is described in [10]

$$PRR = \sum_{i=1}^{n}(\frac{predicti - atucali}{actauli})^2$$

Predictive power comparison criterion:

Two methods are used to compare the predictive power of SRGMs. The criteria are described as follows.

The sum of square (SSE) criteria is used to assess the predictive performance of SRGMs. It is described in [28]:

$$SSE = \sum_{i=1}^{n}(atuali - predicti)^2$$

Where n denotes the number of predicted values.

Data and Performance Analysis:

The application of the proposed technique is evaluated on several existing classical SRGMs based on three datasets collected from software products.

Dataset 1:

The well-known dataset (DS-1) [29][30] is used which is described above. The detailed data is recorded in Table 2

Table 2. Failure Dataset (DS-1)

Failure Time	Failure No	Failure Time	Failure No
1	12	10	2
2	11	11	2
3	20	12	7
4	21	13	3
5	20	14	2
6	13	15	4
7	12	16	3
8	2	17	3
9	1		

The DS-1 recorded 38 failures within 14 weeks. We divide the dataset into two parts. The first 90% of data points are used for the estimation of the parameters and to find the model's goodness-of-fitting ability and the remaining data points are to compare the predictive power of the improved models. Our proposed technique is applied iteratively on the same

model based on the error data points of the previously predicted results. The residual errors from the SRGMs predicted data provide the source to ANFIS that can be modeled iteratively.

The first part of the dataset is used for parameter estimation and goodness-of-fitting comparison. The calculated criteria values for goodness-of-fitting such as MSE, R_2, PPR, PP, and SSE (fit) are shown in Table 3. From Table 3, it can be seen that compared to all models using all five criteria, the HPEIAM technique when applied to each model provides better results, i.e. the significantly best goodness-of-fit power. We can see that the goodness-of-fitting criteria values such as MSE (fit), PPR, PP, and SSE (fit) values of all improved models (GO, Yamada, Weibull, Schneide wand, Pham, and Chang's) are less than the basic (GO, Yamada, Weibull, Schneide wand, Pham and Chang's) models. The R2 criteria value is higher for every improved model. Considering all model's fitting criteria, we find that the applied HPEIAM technique on SRGMs shows better fitting results for every model. For the predictive power comparison, the estimated results of 10% of DS-2 by using SSE are also listed in Table 3. It shows that the predictive results are also improved and show better results when HPEIAM technique is applied. The HPEIAM technique can obtain good fitting and predictive results that have universality.

Figure 3 illustrates the predicted results of six software reliability models such as GO, Yamada, Schneid wand, Weibull, Pham's, and Chang's models, and their improved results by using the proposed technique. Each figure shows a fitting comparison of all models for DS-1 from point 1 (failure time) to point 14 (failure time). It can be seen clearly that the proposed technique has improved the prediction accuracy of all models. The predictive power comparison of the last two models (Pham and Chang's models) shows higher predictive accuracy than the first four models (G-O, Yamada, Schneide wand, and Weibull). We found that our technique works best with the error data which can demonstrate the predicted values well. From the overall evaluation of fitting and predictive results, it can be seen that the proposed technique works well for DS-1.

Table 3. Predictive Results Comparison of all Selected Models Using HPEIAM Technique for DS-1

Model	MSE	R^2	PPR	PP	SSE(fit)	SSE(predict)
G-O Model	63.011	0.971020	0.401	0.984	805.685	13.458
Improved-G-O Model	0.686	0.999647	0.004	0.004	7.924	0.991
Yamada Model	26.198	0.987217	0.246	0.139	305.237	35.337
Improved-Yamada Model	0.422	0.999805	2.98E-04	2.88E-04	4.01E-12	5.491
Schneide's Model	62.883	0.972552	0.399	0.948	815.825	1.659
Improved-Scheide's Model	0.180	0.999907	2.80E-04	2.80E-04	2.154	0.182
Weibull's Model	1375.959	0.932477	3.949	1.602	15866.040	2021.430
Improved-Weibull's Model	0.097	0.999949	1.79E-04	1.80E-04	1.218	0.050
Pham's Model	34.116	0.983113	0.135	0.202	414.198	29.312
Improved-Pham's Model	0.090	0.999954	1.39E-04	1.39E-04	1.080	0.089
Chang's Model	0.265	0.999947	0.001	0.001	3.274	0.166
Improved-Chang"s Model	0.003	0.999999	1.88E-06	1.87E-06	0.00E+00	0.036

Dataset 2:

In this section, we evaluate models using another dataset (DS-2)[29][30]. This dataset is also widely used in many papers. Compared to other datasets which are used in this paper,

DS-2 has a larger time series with 60 observations. The detailed data is shown in Table 4. There are 146 failures were detected in 60 months. The dataset is divided into two parts; the first 90% of the data is used to estimate parameters and goodness-of-fit power for comparison. The other 10% of data is used for predictive power comparison. Table 5 gives a summary of the criteria values of MSE, R2 PPR, PP, and SSE (fit) for DS-2 based on all basic models and the improved results of each model using HPEAIM technique.

From Table 5, we can see that the improved models' criteria values such as MSE, PPR, PP, and SSE (fit) show lower values than the other basic models. The R^2 criterion values of all improved models also show larger values (near one) than other basic models. Comparing all the criterion values for fitting, we found that the proposed technique improves the performance of every model.

For the predictive power comparison, the first part of the dataset is used to estimate parameters and then the remaining part of the dataset to compare the predictive power. The SSE (predict) value for every improved model based on HPIEAM technique shows smaller values than the basic SRGMs. This may indicate the predictive power of every model is improved using the proposed technique.

Figure 4 demonstrates the predicted results of fundamental models such as GO, Yamada, Schneid wand, Weibull, Pham's, and Chang's models. It also shows the improved results by using the proposed technique. Each figure demonstrates the predicted values and real values from point 1 (failure time) to point 60 (failure time). Figure 4 conclude that our model shows better fitting results for every PSRGMs, and in some cases, the prediction accuracy is higher than existing models but little improved. It is due to the iterations we selected to make the accuracy level. In this paper, we have iterated the model only once. The prediction accuracy is further improved by iterating the same process more times.

Dataset 3:

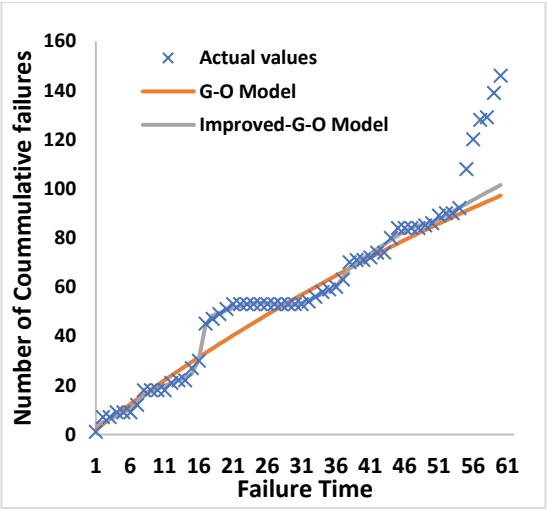
In this section, we evaluate models using another dataset (DS-3)[29][30]. The detailed data is shown in Table 6. The dataset is also divided into two parts; the first 90% of the data is used to estimate parameters and to estimate goodness-of-fit power for comparison. The other 10% of data is used for predictive power comparison. Table 7 gives a summary of the criteria values of MSE, R^2 and PPR, PP, and SSE (fit) for DS-3 based on all basic models such as GO, Yamada, Schneide wand, Weibull, Pham, and Chang's Software reliability models. The Improved results of each model using HPEAIM technique are also shown in the same table. From Table 7, we can see that the improved models' criteria values such as MSE, PPR, PP, and SSE (fit) also show lower values than the other basic models for DS-3. The R2 criterion values of all improved models also show larger values (near to one) than other basic models. Comparing all the criteria values for fitting, we also discovered that the proposed technique improves the performance of every model.

For the predictive power comparison, the first 90% of the dataset is used for estimating parameters and training residual errors, and then we use the remaining 10% of the dataset to compare the predictive power and testing of residual errors. The SSE (predict) value of every improved model based on HPIEAM technique almost shows smaller values than the basic SRGMs. This also indicates that the proposed technique improves the predictive power of every model.

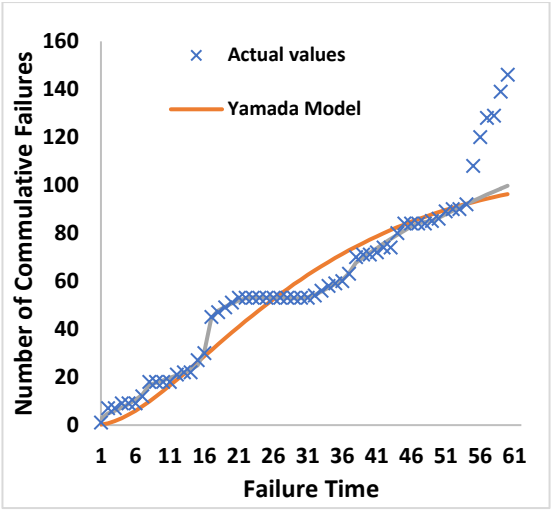
Figure 5 illustrates the predicted results of six software reliability models such as GO, Yamada, Schneidwand, Weibull, Pham's and Chang's models and improved results using the proposed technique. Each figure demonstrates the fitting comparison of all models for DS-3 from point 1 (failure time) to point 25 (failure time). From the overall evaluation of fitting and predictive results, it can be seen that proposed technique yields almost best fitting results using DS-3.

Table 4. Failure Dataset (DS-2)

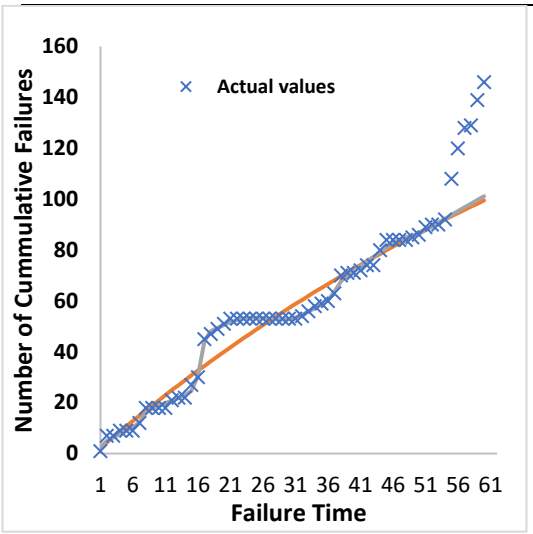
Failure Time	Failure No	Failure Time	Failure No	Failure Time	Failure No
1	1	21	2	41	1
2	6	22	0	42	2
3	0	23	0	43	0
4	2	24	0	44	6
5	0	25	0	45	4
6	0	26	0	46	0
7	3	27	0	47	0
8	6	28	0	48	0
9	0	29	0	49	1
10	0	30	0	50	1
11	0	31	0	51	3
12	3	32	1	52	1
13	1	33	2	53	0
14	0	34	2	54	2
15	5	35	1	55	16
16	3	36	1	56	12
17	15	37	3	57	8
18	2	38	7	58	1
19	2	39	1	59	10
20	2	40	0	60	7



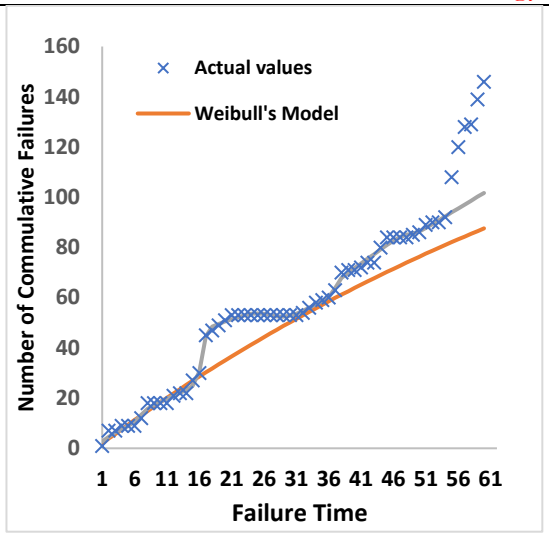
(A) Predicted values of g-o model and improved-g-o model using HPEIAM



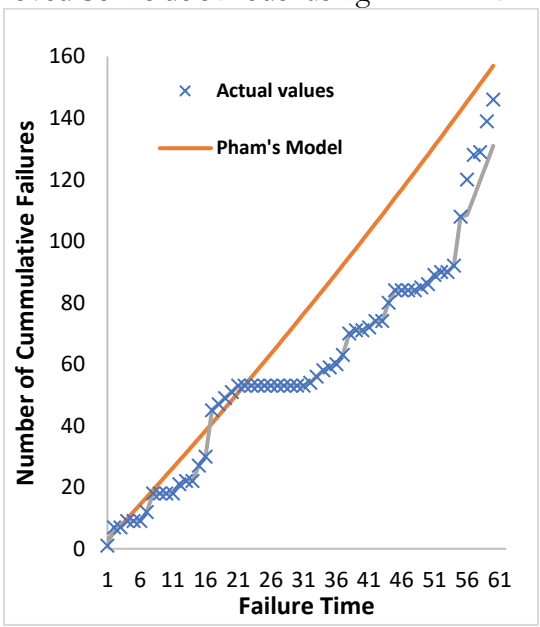
(B) Predicted values of YAMADA's model and improved YAMADA's model using HPEIAM



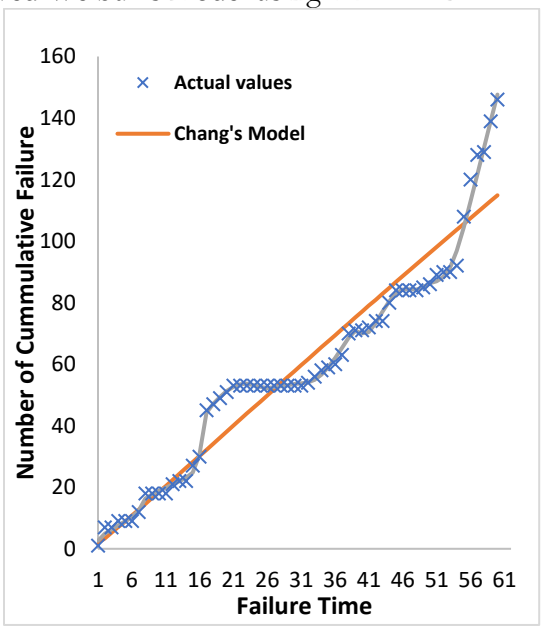
(C) Predicted values of Schneide's model and improved Schneide's model using HPEIAM



(D) Predicted values of Weibull's model and improved Weibull's model using HPEIAM



(E) Predicted values of PHAM's model and improved Pham's model using HPEIAM



(F) Predicted values of Chang's model and improved Chang's model using HPEIAM

FIGURE. 4: Comparison of Predicted Results of PSRGMS And Improved Results of Every PSRGM By applying Proposed HPEIAM Method using DS-2

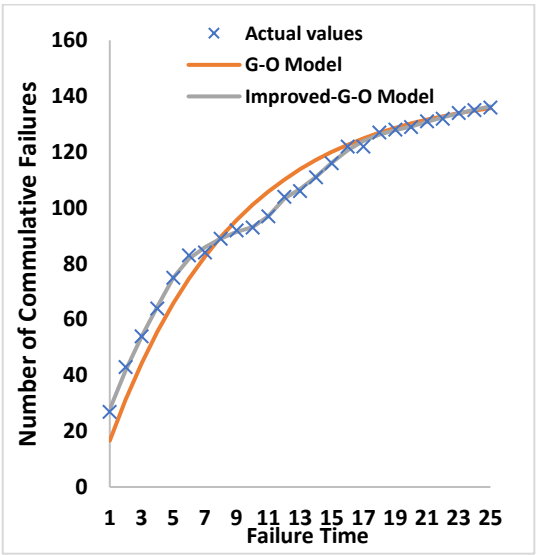
Table 3. Comparison of Predictive Results of all Selected Models Using HPEIAM Technique for DS-1

Model	MSE	R^2(fit)	PPR	PP(fit)	SSE(fit)	SSE(predict)
G-O Model	156.551	0.913709	2.630	2.716	9079.931	7351.426
Improved-G-O Model	108.009	0.937415	1.443	7.543	6264.524	5985.550
Yamada Model	171.619	0.870499	97.365	4.506	9953.879	4347.690
Improved-Yamada Model	115.538	0.932069	1.433	3.495	6701.222	3769.988
Weibull's Model	273.337	0.912739	4.715	3.010	15853.547	6169.454
Improved-Weibull's Model	107.510	0.937516	1.465	3.446	6235.565	3475.411

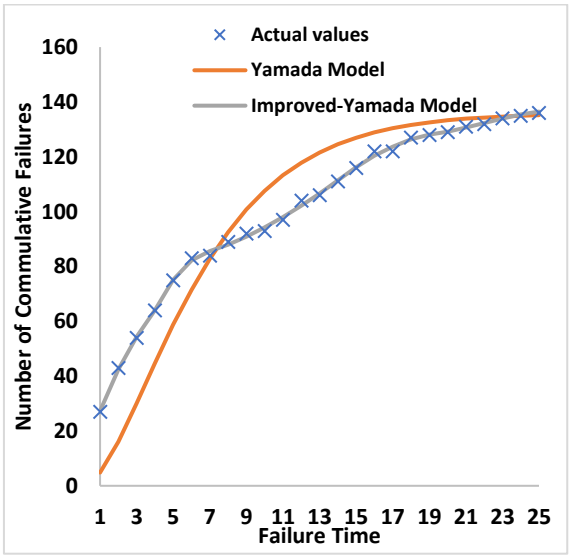
Schneide Model	138.738	0.912115	2.327	2.964	8046.790	3823.724
Improved-Scheide Model	109.655	0.936166	1.282	2.913	2.913	3557.542
Pham's Model	562.830	0.934307	4.161	10.950	32644.157	87.422
Improved-Pham's Model	14.209	0.992911	0.057	0.046	824.127	7.86E-20
Chang's Model	61.280	0.927225	3.006	2.426	3554.213	1641.775
Improved-Chang's Model	3.426	0.997262	0.672	2.757	198.710	2.637

Table 6. Failure Dataset (DS-3)

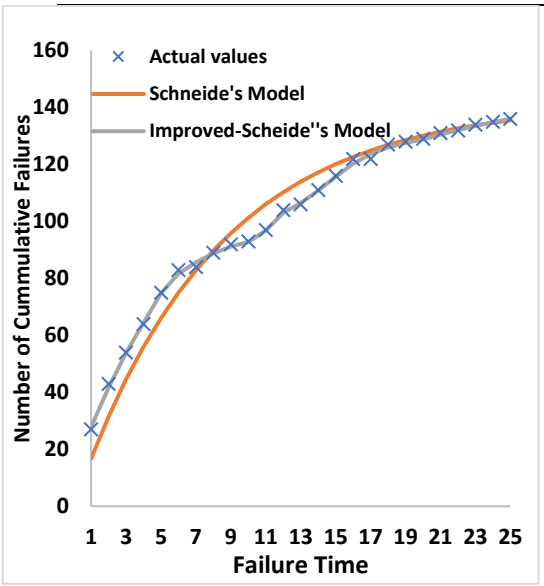
Failure No TBF	TBF	Failure No	TBF	Failure No	TBF
1	27	11	4	21	2
2	16	12	7	22	1
3	11	13	2	23	2
4	10	14	5	24	1
5	11	15	5	25	1
6	8	16	6		
7	1	17	0		
8	5	18	5		
9	3	19	1		
10	1	20	1		



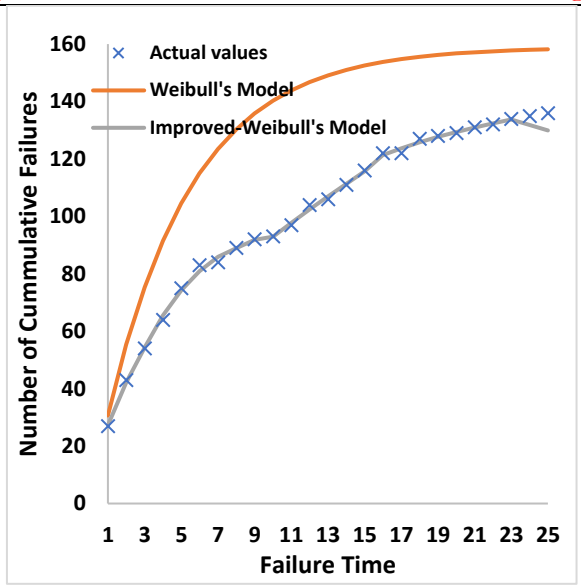
(A) Predicted values of g-o model and improved-g-o model using HPEIAM



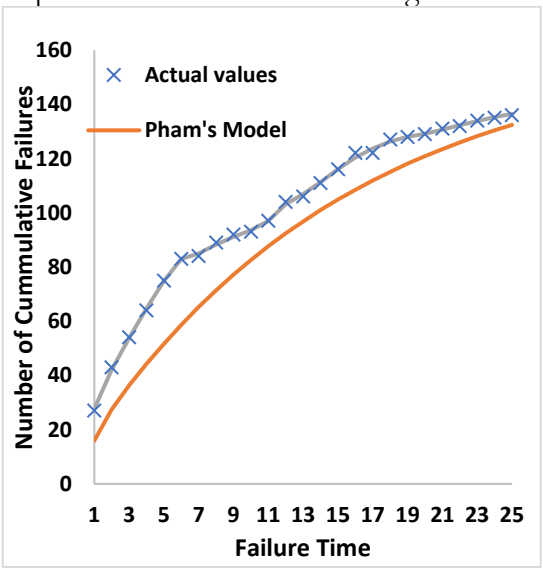
(B) Predicted values of Yamada's model and improved Yamada's model using HPEIAM



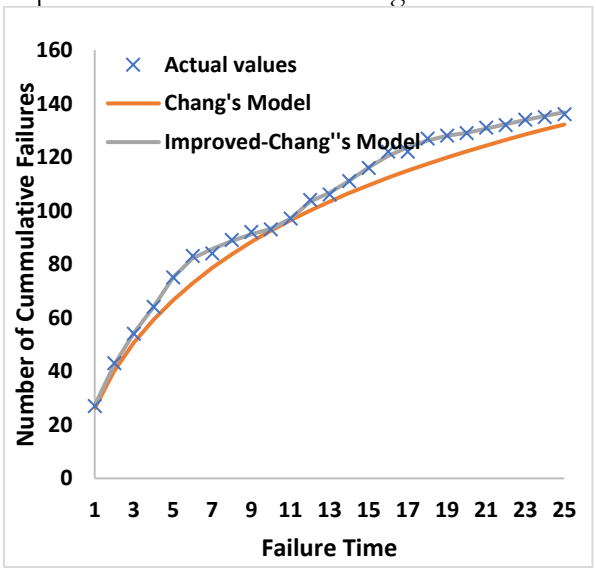
(C) Predicted values of Schneide's model and improved Schneide's model using HPEIAM



(D) Predicted values of Weibull's model and improved Weibull's model using HPEIAM



(E) Predicted values of Pham's model and improved Pham's model using HPEIAM



(F) Predicted values of Chang's model and improved Chang's model using HPEIAM

Figure 5. Comparison of Predicted Results of PSRGMS And Improved Results of Every PSRGM By Applying Proposed HPEIAM Method using DS-3

Discussion and Limitations:

The failure number based on HPEIAM, is likely computed through the iterative process of comparing predicted failures with actual observed failures during the software testing phase. The objective is to optimize the prediction of remaining software failures in general by enhancing the accuracy of SRGM predictions using residual errors from previous models as new input to a non-parametric model like ANFIS.

This study examines the applicability of the proposed HPEIAM based on ANFIS on the existing six PRRGMs: GO, Yamada, Schneide wand, Weibull, Pham, and Chang's model. This study focuses on the improvement of the prediction accuracy of PSRGMs. The prediction capabilities of six PSRGMs have been examined along with MSE, R^2 PPR, PP, and SSE for the different types of datasets. The fitting results indicate that the PSRGMs performance can be improved up to an expected level using the proposed HPEAIM technique. Table 3, 5 and 7

show that the proposed technique can improve the fitting and predictive power of existing PSRGMs. The fitting criterion values (MSE, R2 and PPR, PP and SSE (fit)) based on our proposed technique are better than those of existing models. Moreover, the predictive SSE values of the proposed technique are less than those of existing models.

To reiterate, performance improvement is achieved by combining the most powerful prediction models: PSRGMs, ANN, and FIS. The Improve model is capable of obtaining better predictive results. Moreover, it is universal and does not rely on the nature of a historical dataset and/or a particular environment. The problems that arise from the PSRGMs assumptions are also overcome using the proposed technique. In this study, instead of adding SRGM to a large number of existing models, we tried to propose a new optimized technique to improve the performance of existing PSRGMs. However, owing to a basic limitation of space, only 90% for fitting and 10% predictive power for every dataset is calculated. Another limitation arises due to experimentation with a limited quantity and type of datasets. Further research is needed to evaluate the applicability of the proposed method to more historical datasets and in a different environment.

Conclusion:

This paper proposes a new high-precision error iterative accuracy method and analyzes the method in detail by applying it to classical PSRGMs. The proposed technique combines residual errors with the classical PSRGMs. Furthermore, a comparison of the applied technique results on different PSRGMs has also been provided in terms of five criteria values: MSE, R^2, PP, PPR, and SSE on three datasets. Experimental results show that HPEIAM method can effectively improve and optimize the performance of every existing PSRGM by providing better goodness-of-fit and predictive power. The results of every iteration can be improved by the previously predicted solution. Our proposed HPEIAM techniques have combined the advantages of the most powerful software reliability models: PSRGM, ANN, and FIS, and overcome the disadvantages. It is considered a very flexible and universal technique for improving the performance of different software reliability growth models. The results show that this method can yield more accurate predicted values than the classical PSRGMs and also solves the problem resulting from having an unrealistic assumption.

The proposed HPEIAM has practical applicability in several areas related to software reliability and assurance during the testing phase. It helps developers to better allocate resources, optimize testing schedules, and improve overall software quality. Predicting the remaining number of failures more accurately, it enables better management of the testing phase, minimizing unnecessary testing.

Moreover, we used only PSRGMs with our proposed technique to enhance performance. Thus, further research is needed to give a broader validation. In the future, we will also use the same technique to improve the performance of non-parametric models.

Acknowledgment: We would like to thank our supervisor for his help

Author Contribution: Authors have made an equal contribution to this study.

Conflict of Interest: There is no conflict of interest

References:

[1] M. A. H. and N. Ahmad, "Key issues in software reliability growth models," *Recent Adv. Comput. Sci. Commun. (Formerly Recent Patents Comput. Sci.*, vol. 15, no. 5, pp. 741–747, 2022, doi: 10.2174/2666255813999201012182821.

[2] M. A. F. Shahrzad Oveisi, Ali Moeini, Sayeh Mirzaei, "Software reliability prediction: A survey," *Qual. Reliab. Eng. Int.*, vol. 39, no. 1, pp. 412–453, doi: https://doi.org/10.1002/qre.3220.

[3] E. O. Marko Palviainen, Antti Evesti, "The reliability estimation, prediction and measuring of component-based software," *J. Syst. Software,* vol. 84, no. 6, pp. 1054–1070,

2011, doi: https://doi.org/10.1016/j.jss.2011.01.048.

[4] P. Z. and Y. -t. Chang, "Software fault prediction based on grey neural network," *8th Int. Conf. Nat. Comput. Chongqing, China*, pp. 466–469, 2012, doi: 10.1109/ICNC.2012.6234505.

[5] D. . Bal, P.R., Nachiketa Jena, Mohapatra, "Software Reliability Prediction Based on Ensemble Models," *Proceeding Int. Conf. Intell. Commun. Control Devices*, vol. 479, pp. 895–902, 2017, [Online]. Available: https://link.springer.com/chapter/10.1007/978-981-10-1708-7_105

[6] C.-Y. H. and T.-Y. C. C. -J. Hsu, "A Modified Genetic Algorithm for Parameter Estimation of Software Reliability Growth Models," *19th Int. Symp. Softw. Reliab. Eng.*, pp. 281–282, 2008, doi: 10.1109/ISSRE.2008.35.

[7] and W. A. G. Jabeen, P. Luo, "An improved software reliability prediction model by using high precision error iterative analysis method," *Softw. Testing, Verif. Reliab.*, vol. 29, no. 6–7, p. 1710, 2019, doi: https://doi.org/10.1002/stvr.1710.

[8] M. O. and S. O. S. Yamada, "S-Shaped Reliability Growth Modeling for Software Error Detection," *IEEE Trans. Reliab.*, vol. 32, no. 5, pp. 475–484, 1983, doi: 10.1109/TR.1983.5221735.

[9] Y. Z. and J. Davis, "Open source software reliability model: an empirical approach," *ACM SIGSOFT Softw. Eng. Notes*, vol. 30, no. 4, pp. 1–6, 2005, doi: https://doi.org/10.1145/1082983.1083273.

[10] H. Pham, "Software reliability and cost models: Perspectives, comparison, and practice," *Eur. J. Oper. Res.*, vol. 149, no. 3, pp. 475–489, 2003, [Online]. Available: https://ideas.repec.org/a/eee/ejores/v149y2003i3p475-489.html

[11] A. L. G. and K. Okumoto, "Time-Dependent Error-Detection Rate Model for Software Reliability and Other Performance Measures," *IEEE Trans. Reliab.*, vol. 28, no. 3, pp. 206–211, 1979, doi: 10.1109/TR.1979.5220566.

[12] and C. W. M. Xie, G. Y. Hong, "Software reliability prediction incorporating information from a similar project," *J. Syst. Softw.*, vol. 49, no. 1, pp. 43–48, 1999, doi: https://doi.org/10.1016/S0164-1212(99)00065-5.

[13] S. . Hu, Q.P., Xie, M., Ng, "Software Reliability Predictions using Artificial Neural Networks," *Comput. Intell. Reliab. Eng.*, vol. 40, pp. 197–222, 2007, doi: https://doi.org/10.1007/978-3-540-37372-8_8.

[14] and S. S. M. K. Bhuyan, D. P. Mohapatra, "Prediction Strategy for Software Reliability Based on Recurrent Neural Network," *Comput. Intell. Data Min.*, vol. 2, pp. 295–303, 2016, doi: 10.1007/978-81-322-2731-1_27.

[15] C. J. and S.-W. Jin, "Software reliability prediction model based on support vector regression with improved estimation of distribution algorithms," *Appl. Soft Comput.*, vol. 15, pp. 113–120, 2014, doi: https://doi.org/10.1016/j.asoc.2013.10.016.

[16] X. Guoxing, "Research of Software Reliability Based on Genetic Algorithm," *Int. Conf. Signal Process. Syst.*, pp. 806–809, 2009, doi: 10.1109/ICSPS.2009.208.

[17] and J. B. T. Kim, K. Lee, "An effective approach to estimating the parameters of software reliability growth models using a real-valued genetic algorithm," *J. Syst. Softw.*, vol. 102, pp. 134–144, 2015, doi: https://doi.org/10.1016/j.jss.2015.01.001.

[18] and S. P. S. Singhal, P. Kapur, V. Kumar, "Stochastic debugging based reliability growth

models for Open Source Software project," *Ann. Oper. Res.*, vol. 340, pp. 531–569, 2024, doi: https://doi.org/10.1007/s10479-023-05240-6.

[19] H. B. Y. and D. K. Yadav, "A fuzzy logic based approach for phase-wise software defects prediction using software metrics," *Inf. Softw. Technol.*, vol. 63, pp. 44–57, 2015, doi: https://doi.org/10.1016/j.infsof.2015.03.001.

[20] R. . Rizvi, S.W.A., Singh, V.K., Khan, "The State of the Art in Software Reliability Prediction: Software Metrics and Fuzzy Logic Perspective," *Inf. Syst. Des. Intell. Appl.*, vol. 433, pp. 629–637, 2016, doi: https://doi.org/10.1007/978-81-322-2755-7_65.

[21] J.-S. R. Jang, "Fuzzy modeling using generalized neural networks and Kalman filter algorithm," *AAAI*, vol. 2, pp. 762–767, 1991, [Online]. Available: https://dl.acm.org/doi/10.5555/1865756.1865795

[22] E. E. and E. A. Sezer, "A comparison of some soft computing methods for software fault prediction," *Expert Syst. Appl.*, vol. 42, no. 4, pp. 1872–1879, 2015, doi: https://doi.org/10.1016/j.eswa.2014.10.025.

[23] J. D. Musa, "A theory of software reliability and its application," *IEEE Trans. Softw. Eng.*, vol. SE-1, no. 3, pp. 312–327, 1975, doi: doi: 10.1109/TSE.1975.6312856.

[24] and J.-R. C. C.-J. Hsu, C.-Y. Huang, "Enhancing software reliability modeling and prediction through the introduction of time-variable fault reduction factor," *Appl. Math. Model.*, vol. 35, no. 1, pp. 506–521, 2011, doi: https://doi.org/10.1016/j.apm.2010.07.017.

[25] and K. Y. S. I. H. Chang, H. Pham, S. W. Lee, "A testing-coverage software reliability model with the uncertainty of operating environments," *Int. J. Syst. Sci. Oper. Logist.*, vol. 1, no. 4, pp. 220–227, 2014, doi: https://doi.org/10.1080/23302674.2014.970244.

[26] S. H. and H. Pham, "Quasi-Renewal Time-Delay Fault-Removal Consideration in Software Reliability Modeling," *IEEE Trans. Syst. Man, Cybern. - Part A Syst. Humans*, vol. 39, no. 1, pp. 200–209, 2009, doi: 10.1109/TSMCA.2008.2007982.

[27] and T.-Z. L. K.-C. Chiu, Y.-S. Huang, "A study of software reliability growth from the perspective of learning effects," *Reliab. Eng. Syst. Saf.*, vol. 93, no. 10, pp. 1410–1421, 2008, doi: https://doi.org/10.1016/j.ress.2007.11.004.

[28] X. T. and H. P. Xuemei Zhang, "Considering fault removal efficiency in software reliability assessment," *IEEE Trans. Syst. Man, Cybern. - Part A Syst. Humans*, vol. 33, no. 1, pp. 114–120, 2003, doi: 10.1109/TSMCA.2003.812597.

[29] P. J. Denning, "Statistical computer performance evaluation: Academic Press," *SIGMETRICS Perform. Eval. Rev.*, vol. 2, no. 1, pp. 16–22, 1972, doi: https://doi.org/10.1145/1041606.1041611.

[30] A. L. Goel, "Software Reliability Models: Assumptions, Limitations, and Applicability," *IEEE Trans. Softw. Eng.*, vol. 11, no. 12, pp. 1411–1423, 1985, doi: 10.1109/TSE.1985.232177.