Palestine Journal of Mathematics

Vol. 11(Special Issue I)(2022) , 38–45

AN IMPROVED SRGM CONSIDERING UNCERTAIN OPERATING ENVIRONMENT

Bhoopendra Pachauri, Anamika Jain and Subadra Raman

Communicated by Shimpi Singh Jadon

MSC 2010 Classifications: Primary 20M99, 13F10; Secondary 13A15, 13M05.

Keywords and phrases: Uncertainty, NHPP, Operating environment, Software reliability.

Abstract Contemporary software systems are used in various walks of life, in diverse environments that differs from the environment in which it was tested. The software reliability is evaluated based on the past failure data with no regard to software environment in the development process. In this paper, an improved Non-homogeneous Poisson Process based software reliability model is proposed by including vagueness in operating environment. The uncertainty due to operating environment is incorporated in the form of a generalized two-parameter probability density function. The proposed model is validated using secondary data set which performs considerably better than the present models.

1 Introduction

Software reliability means, the probability that a software is execute its proposed functions in a system's environment, without failing, for a given period [9]. In order, for a software to withstand in the heavily competitive market, it is essential to perform software reliability analysis, to ensuring that the quality of the software system lives up to perform its desired functions. With the growing needs of mankind, the complexity of software systems also extends. Hence there arises a need for more rigorous testing processes and reliability models before its release in the market.

Moreover, it is not sufficient to test the software for all possible errors, it is also essential to keep in mind the different environments and situations it might be used in and consider the effects it might have on implementation. From 1970, there are many non-homogeneous Poisson process (NHPP) based software reliability growth models (SRGMs) with different assumptions have been studied for the evaluation of the number of lasting faults in the system, and reliability of the software in the given period. In 1979, Goel and Okumoto [2] firstly introduced a NHPP based SRGM in perfect debugging environment without considering environmental factor. Huang and Kuo [4] considered logistic testing effort function in their model. Yang and Xie [23] studied software release time policy and considered resource allocation in their SRGM. Zhu and Pham [25] divide the faults into two types and analyze them on a two-phase software reliability model. Recently, most of NHPP based SRMs were proposed by considering that the faults identified in the testing were eliminated instantly without any delay and no new faults were created in the correction process. However, software correction is a complicated and ambiguous process, that is affected by various factors i.e., tester skill, fault reduction factor, test resource, operating environment, test tool, testing effort function etc. Hence, new faults may be created in the fault correction process [21], [8], [22], [10]. In general, the fault insertion rate per fault was considered constant [22] or decreasing changes [10] over time in perfect debugging environment. Such a consideration may not be true for all the situation, therefore, fault insertion rate per fault may change over time [19]

Recently, a few researchers have studied the effect of the environmental aspect on software reliability throughout development and operation. Zhang and Pham [24] firstly discussed 32 environmental aspects that have some influence on system reliability. These environmental factors

include each stage of the software development procedure, teamwork, human behavior, and the interactions amongst them etc. Later, Zhu et al. [26] did a comparative study to understand the influence level of environmental factors in software reliability evaluation and discussed important environmental aspects.

A little work has been done including uncertainty of operating environment in the software growth models. Teng and Pham [18] discussed a general unintentional environment, in which the fault-finding rate is depends on time and considering in testing as well as operation. Hsu et al. [3] discussed an SRGM incorporating constant, increasing, and decreasing fault reduction factor. Pham [11] considered the uncertainty in the operating environment with a V-tub shaped fault recognition rate. Song et al. [13] represented a model with three-parameter fault recognition rate function and relate it to the fault exposure rate function using uncertainty in the operating environment. However, over the past 4-5 years, the importance of studying the vagueness of the operating environment has significantly increased. This type of study has been incorporated in various SRGMs [14], [15], [16], [27], [17], [20], [5]. Xu and Yao proposed a NHPP based model using partial differential equation, to measure the ambiguities in the perfect and imperfect debugging environment [20]. Jain et al [6] optimized the cost of a fault -tolerant system using fuzzy metrices. Recently, Pradhan et al. [12] proposed a SRGM which incorporates the three-parameter generalized inflection S-shaped function as a fault reduction factor and also considered the time lag between the fault detection and correction process. Dhaka et al [1] used Gompertz testing effort function to model a SRGM with uncertainty in operating environment.

Keeping in mind the above literature, the purpose of this article is to introduce a SRGM, which has a fault detection rate that is simple enough to compute but also produces a good fit to the datasets on which it is being experimented. It is also essential to incorporate the uncertainty due to operating environments as parameters of a generalized probability function, which may represent any of the previously discussed factors representing changes in environment. The proposed study may be support to the software development companies to make prediction about faults present in the software system and remove them timely so that the possibility of software failure can be minimized. The remain manuscript is designed as follow: In section 2, proposed model is discussed, and result analysis is given in section 3. Finally, concluding remark with future scope is given in section 4.

2 Proposed Model

In the NHPP based models, the main task is to determine mean value function. Keeping in mind the assumptions, methodology [8], [11], [14], [15], [16], [27], [17] and above literature review, following are the assumptions for the proposed model.

- The fault removal process follows the NHPP.
- The failure of software is subject to the demonstration of the residual faults in the program.
- There is no dependency among the faults.
- The failure intensity is proportional to the left-over faults in the system.
- η is a generalised probability density function with parameters, α and β.
- The fault recognition rate function is considered as $b(t) = be^{-at}$.

The cumulative number of fault function $m(t)$ of model is built upon the foundation of the differential equation given below,

$$\frac{dm(t)}{dt} = b(t)(a(t) - m(t))$$

Hence, the differential equation representing an NHPP based software reliability growth model considering the vagueness in the operating environments is,

$$\frac{dm(t)}{dt} = \eta b(t)(a(t) - m(t)).$$

The mean value function $m(t)$ after solving above differential equation with initial condition $m(0) = 0$ is,

$$m(t) = \int_\eta N\left(1 - e^{-\eta \int_0^t b(s)ds}\right) d\eta,$$
$$m(t) = N\left(1 - \frac{\beta}{\beta + \int_0^t b(s)ds}\right)^\alpha.$$

Here, the fault detection rate $b(t) = be^{-at}$ which when substituted and simplifies the previous set of equations to find the main equation of this model,

$$m(t) = N\left(\frac{1}{1 + \frac{\beta, a}{b(1-e^{-at})}}\right)^\alpha,$$

where, b stands for the proportion of the initial fault content in the system and a stands for the percentage rate of debugging of the faults. α and β represent the parameters of the probability density function defined on the random variable η.

3 Result Analysis

3.1 Comparsion Criteria and Data Set

Generally, it has been seen that for the comparison of various SRGMs, four different criteria are used: mean squared error (MSE), Variation, predictive power (PP) and predictive-ratio risk (PRR). Each of them signifies a different extent of the spread of the data used from the model predictions.

$$\text{Mean Squared Error (MSE)} = \frac{\sum_{i=1}^m (\hat{m}(t_i) - y_i)^2}{n - N},$$

where n and N are the total observations and total unknown parameters, respectively. y_i is the number of faults at time t_i.

$$\text{Predictive-ratio risk (PRR)} = \sum_{i=1}^m \left(\frac{\hat{m}(t_i) - y_i}{\hat{m}(t_i)}\right)^2,$$

$$\text{Predictive power (PP)} = \sum_{i=1}^m \left(\frac{\hat{m}(t_i) - y_i}{y_i}\right)^2.$$

To compare the predictive performance of the model the PRR and PP are used. The PRR measures the distance from the actual data against the estimated data by assigning a large penalty and PP measures the distance from the estimated to the actual data.

The Variation of a dataset is defined as

$$\text{Variation} = \sqrt{\frac{1}{n-1} \sum_{i=1}^n \left(y_i - \hat{m}(t_i) - Bias\right)^2}$$
$$\text{where, } Bias = \sum_{i=1}^n \frac{1}{n}(\hat{m}(t_i) - y_i).$$

A SRGM with smaller valuer of these criteria is better compared to the others. The model is tested on the data set that is the result of the testing phase of GNOME 2.0, which is a free open-source desktop environment for mainly UNIX operating systems. Its data recorded over the period of 24 weeks was obtained by secondary means for the purpose of this analysis during testing. In this time duration total 85 faults were found [7]. for more detail see Table 3.1.

3.2 Model Validation

For the estimation of the five unknown parameters in the model equation, IBM SPSS Statistics Data Editor was used. Since the process of choosing the starting values for the iterations was tiresome, Desmos Graphing calculator was used to adjust parameter values and look for a rough fit of the data. Firstly, the output from IBM SPSS was obtained for dataset (GNOME 2.0 testing data) after multiple combinations of starting values were input, in a total of 22 iterations.

Table 3.1: Dataset - GNOME 2.0

Testing Time (Weeks)	Detected Fault	Cumulative Detected Faults
1	6	6
2	5	11
3	3	14
4	2	16
5	5	21
6	5	26
7	8	34
8	4	38
9	8	46
10	3	49
11	2	51
12	1	52
13	6	58
14	8	66
15	6	72
16	2	74
17	2	76
18	1	77
19	1	78
20	1	79
21	1	80
22	2	82
24	3	85

From Figure 3.1 we see that the fit of the model could be better around the halfway mark of the testing period (around 9 to 15 weeks). However, when the curve of the obtained data is smoothened, we can say that the model fits almost perfectly.

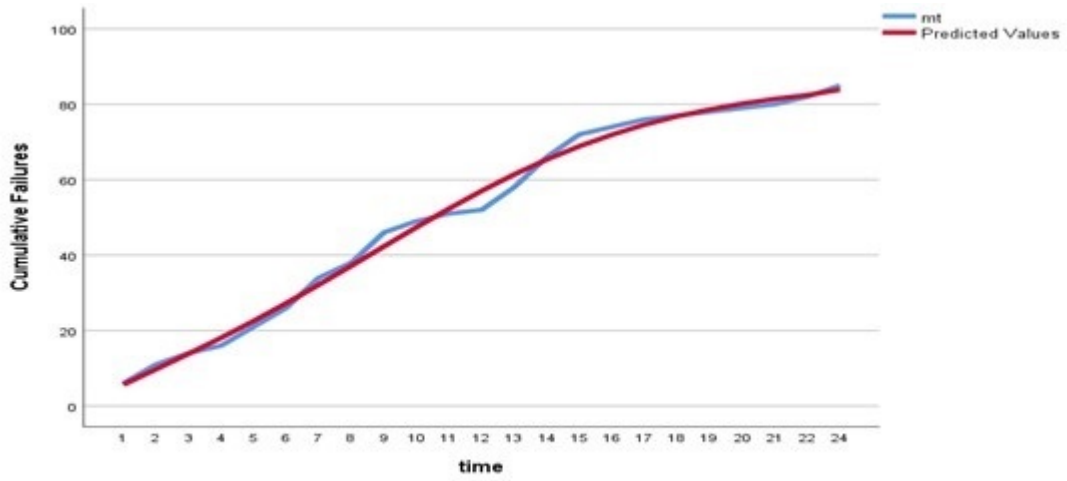

Figure 3.1: Graph representing actual and predicted values of Dataset

For the purpose of this analysis, five NHPP based existing models from literature have been chosen, with which we shall compare the values obtained for the proposed model. In order to achieve unbiased results, we have to test all six of the models using the same datasets. So there

arises a need to estimate the corresponding parameters of the five existing models. This was done by the same procedure as of the proposed model.

Table 3.2: Estimated Results with Comparison on Real Dataset

Model	Estimated Parameters	MSE	PRR	PP	Variation
G-O Model	$\hat{a} = 147.581, \hat{b} = 0.039$	13.68	0.1907	0.2749	3.440
Inflection S-shaped Model	$\hat{a} = 89.766, \hat{b} = 0.181$ $\hat{\beta} = 3.455$	5.451	0.5039	0.2535	2.162
Delayed S-shaped	$\hat{a} = 90.976, \hat{b} = 0.181$	9.318	14.65	1.081	2.794
Yamada Imperfect Debugging Model	$\hat{a} = 147.581, \hat{b} = 0.039$	11.51	0.1798	0.2575	3.157
Pham-Zhang IFD	$\hat{a} = 6639.787, \hat{b} = 0.001$ $\hat{\alpha} = -0.028$	6.465	0.7596	0.3251	2.358
Proposed Model	$\hat{a} = 0.001, \hat{b} = 0.759$ $\hat{\alpha} = -0.8099, \hat{\beta} = -0.759$ $\hat{N} = 119.6$	**4.766**	**0.0693**	**0.0697**	**2.029**

We can clearly verify from Table 3.2 that the fault detection rate b(t) in each of the six models, lies between 0 and 1, for all values of time. Moreover, the initial number of faults in the software can also be verified to be close to the actual data. So, we proceed further to calculate values for comparison of the models. From Table 3.2 an observation can, be made that the proposed model has values in all the four characteristics: MSE, PRR, PP and Variation which are smaller than every other existing model discussed in this paper. It is also useful to consider the goodness-of-fit of the proposed model as compared to the existing models. To visualize this, six graphs representing the six models have been plotted for comparison against the actual mean value function in Figure 3.1 and Figures 3.2-3.6.

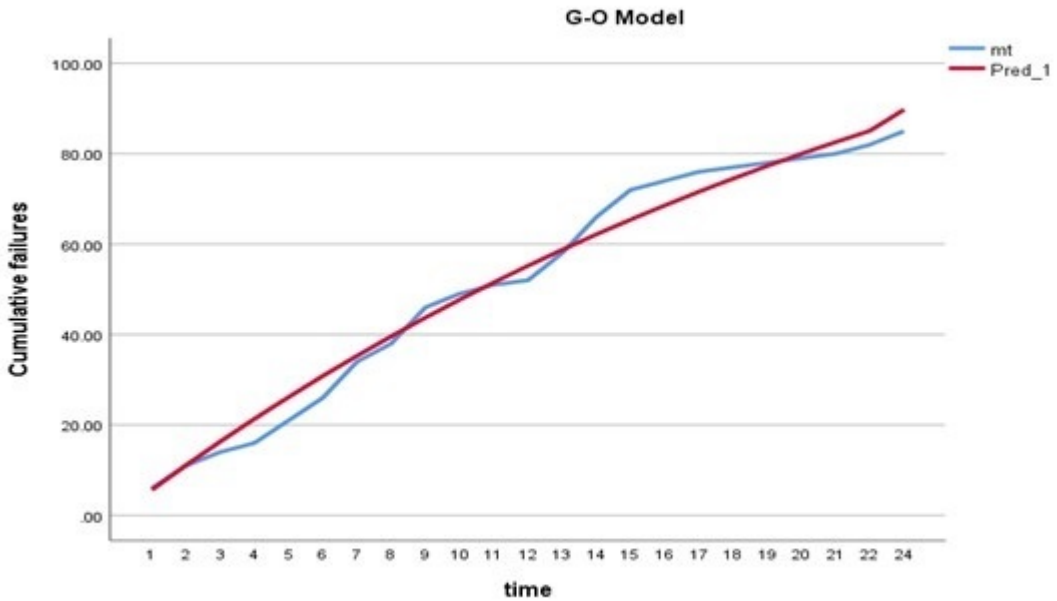

Figure 3.2: Goodness of fit of the G-O model

From Figure 3.1 and Figures 3.2-3.6, it can be clearly seen that the fit of the proposed model is good compared to the existing models. Hence, we can conclude that on the basis of results, the proposed model is significantly better than the existing models.

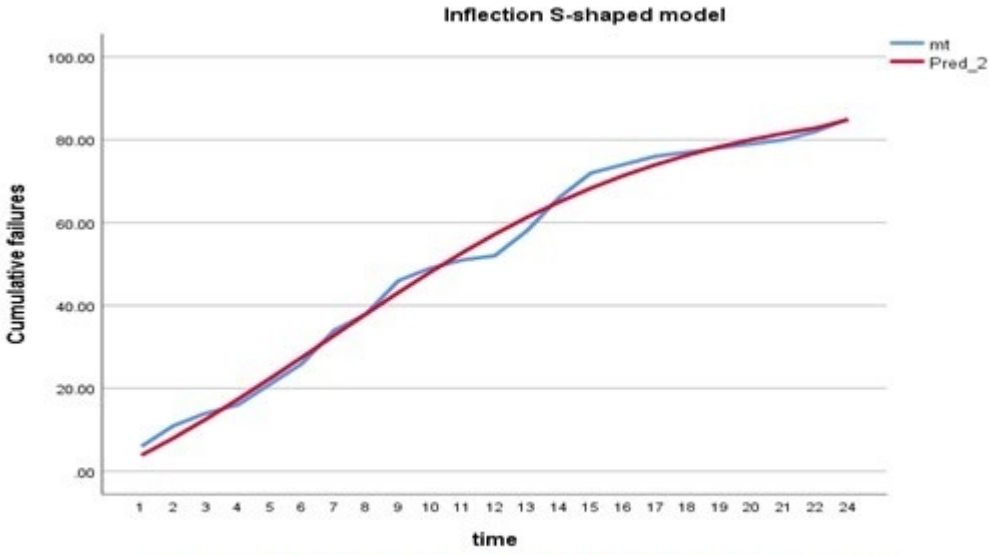

Figure 3.3: Goodness of fit of the Inflection S-Shaped model

Figure 3.4: Goodness of fit of the Delayed Shaped model

4 Conclusion

In this research work, an improved SRGM by considering the uncertainty of the operating environment has been discussed. The cumulative number of faults are calculated for an exponential decay fault detection rate function with uncertainty of the operating environment. The model is validated using real data sets and the results have been found with five comparison criteria. In retrospect, it can be said that these results are comparable to literature where incorporation of an uncertainty factor proved better results of the model to the data. However, it is evident that we have included more criteria for the comparison with existing models, so our results are more valid.

In addition, the accuracy of the results obtained, can be checked on primary data, collected specifically for the purpose of this analysis, instead of using secondary data from the testing

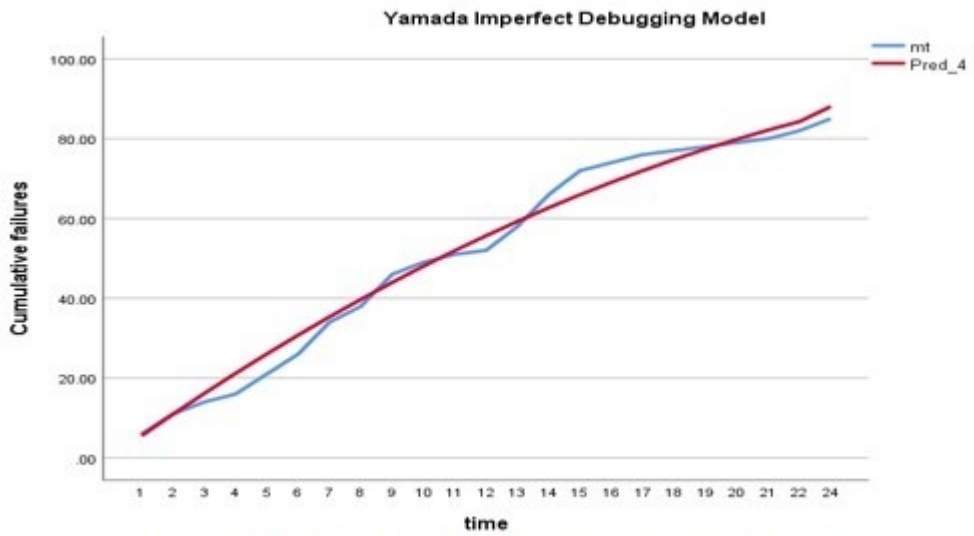

Figure 3.5: Goodness of fit of theYamada Imperfect model

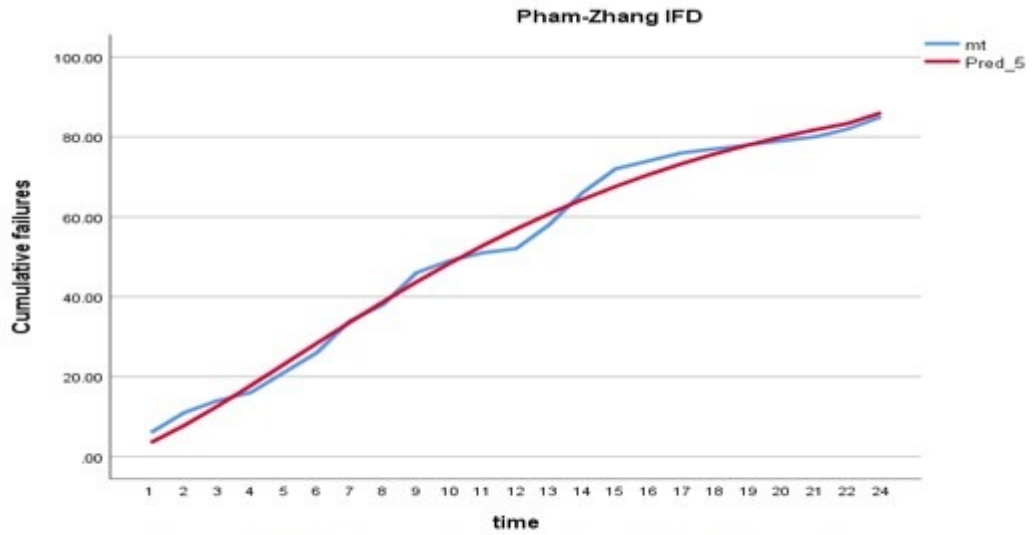

Figure 3.6: Goodness of fit of the Pham-Zhang IFD model

phase. Yet another development of the proposed model could be the analysis of optimal release time. The analysis can also be extended by the addition of a delay factor to the existing model.

References

[1] R. Dhaka, B. Pachauri and A. Jain, *SRGM using testing-effort function with uncertainty in operating environment*, IOP Conf. Ser.: Mater. Sci. Eng. **1099(1)**, 012020 (2021).

[2] A. Goel and K. Okumoto, Time-dependent error-detection rate model for software reliability and other performance measures, *IEEE Trans Reliab*, **28(3)**, 206–211 (1979).

[3] C. J. Hsu, C. Y. Huang and J. R. Chang, Enhancing software reliability modeling and prediction through the introduction of time-variable fault reduction factor, *Appl. Math. Model*, **35(1)**, 506-521 (2011).

[4] C. Y. Huang and S. Y. Kuo, Analysis of incorporating logistic testing-effort function into software reliability modeling, *IEEE Trans Reliab*, **51(3)**, 261–270 (2002).

[5] J. Iqbal, Software reliability growth models: A comparison of linear and exponential fault content functions for study of imperfect debugging situations, *Cogent. Eng.*, **4**, 1286739 (2017).

[6] M. Jain, P. Kumar and R. K. Meena, Fuzzy metrics and cost optimization of a fault-tolerant system with a vacationing and unreliable server. *J Ambient Intell Humaniz Comput J AMB INTEL*, **11**, 5755-5770 (2020). https://doi.org/10.1007/s12652-020-01951-x.

[7] X. Li, Y. F, Li and M. Xie, Reliability analysis and optimal version-updating for open-source software, *Inf Softw Technol*, **53(9)**, 929-936 (2011).

[8] M. Ohba, *Inflection S-shaped software reliability growth models*, In: Osaki S., Hatoyama Y. (eds) Stochastic Models in Reliability Theory, Lect. Notes Econ. Math. Syst., **235**, Springer, Berlin, Heidelberg (1984).

[9] H. Pham, *System Software Reliability*, Springer, London, UK, (2006).

[10] H. Pham and X. Zhang An NHPP software reliability models and its comparison, *Int. J. Reliab. Qual. Saf. Eng.*, **4**, 269-282 (1997).

[11] H. Pham, A new software reliability model with Vtub-shaped fault-detection rate and the uncertainty of operating environments, *Optimization*, **63(10)**, 1481-1490 (2014).

[12] V. Pradhan, A. Kumar and J. Dhar, Modelling software reliability growth through generalized inflection S-shaped fault reduction factor and optimal release time, *Proc Inst Mech Eng O J Risk Reliab P I MECH*, 2021, https://doi.org/10.1177/1748006X211033713.

[13] K. Y. Song, I. H. Chang and H. Pham, A Three-parameter fault-detection software reliability model with the uncertainty of operating environments, *Int. J. Syst. Sci.*, **26**, 121-132 (2017).

[14] K. Y. Song, I. H. Chang and H. Pham, A software reliability model with a Weibull fault detection rate function subject to operating environments, *Appl. Sci.*, **7(10)**, 983 (2017).

[15] K. Y. Song, I. H. Chang and H. Pham, An NHPP software reliability model with S-shaped growth curve subject to random operating environments and optimal release time, *Appl. Sci.*, **7**, 1304 (2017).

[16] K. Y. Song, I. H. Chang and H. Pham, Optimal release time and sensitivity analysis using a new NHPP software reliability model with probability of fault removal subject to operating environments, *Appl. Sci.*, **8**, 714 (2018).

[17] K. Y. Song, I. H. Chang and H. Pham, A testing coverage model based on NHPP software reliability considering the software operating environment and the sensitivity analysis, *Mathematics*, **7**, 450 (2019).

[18] X. Teng and H. Pham, A new methodology for predicting software reliability in the random field environments, *IEEE Trans Reliab*, **55(3)**, 458-468 (2006).

[19] J. Wang, Z. Wu, Y. Shu and Z. Zhang, An imperfect software debugging model considering log-logistic distribution fault content function, *J. Syst. Softw.*, **100**, 167-181 (2015).

[20] J. Xu and S. Yao, Software reliability growth model with partial differential equation for various debugging processes, *Math. Probl. Eng.*, **2016**, 2476584 (2016).

[21] S. Yamada, M. Ohba and S. Osaki, S-shaped reliability growth modeling for software error detection, *IEEE Trans Reliab*, **32**, 475-484 (1983).

[22] S. Yamada, K. T. Okumoto and S. Osaki, Imperfect debugging models with fault introduction rate for software reliability assessment, *Int. J. Syst. Sci.*, **23(12)**, 2253-2264 (1992).

[23] B. Yang and M. Xie, A study of operational and testing reliability in software reliability analysis, *Reliab. Eng. Syst. Saf.*, **70(3)**, 323–329 (2000).

[24] X. Zhang and H. Pham, Analysis of factors affecting software reliability, *J. Syst. Softw.*, **50(1)**, 43-56 (2000)

[25] M. Zhu and H. Pham, A two-phase software reliability modeling involving with software fault dependency and imperfect fault removal, *Comput. Lang. Syst. Struct.*, **53**, 27–42 (2018).

[26] M. Zhu, X. Zhang and H. Pham, A comparison analysis of environmental factors affecting software reliability, *J. Syst. Softw.*, **109**, 150-160 (2015)

[27] M. Zhu and H. Pham, A software reliability model incorporating martingale process with gamma-distributed environmental factors, *Ann Oper Res*, 265 (2018).

Author information

Bhoopendra Pachauri, Anamika Jain and Subadra Raman, Manipal University Jaipur, Jaipur, Rajasthan,, INDIA.
E-mail: bhupdma@gmail.com,anamika.jain@jaipur.manipal.edu, raman.subadra@gmail.com