Available online at www.ijpe-online.com

International Journal of Performability Engineering

vol. 15, no. 5, May 2019, pp. 1265-1272
DOI: 10.23940/ijpe.19.05.p1.12651272

Improved Algorithm for Non-Homogeneous Poisson Process Software Reliability Growth Models Incorporating Testing-Effort

Vidhyashree Nagaraju[a], Thierry Wandji[b], and Lance Fiondella[a,*]

[a]*University of Massachusetts Dartmouth, North Dartmouth, 02747, USA*
[b]*Naval Air Systems Command, Patuxent River, 20670, USA*

Abstract

Critical systems are becoming increasingly software intensive, necessitating reliable software to ensure proper operation. Non-homogeneous Poisson process software reliability growth models are commonly used to characterize fault detection as a function of testing time, which enables quantitative assessment of software reliability. Many early models assumed that the testing-effort was constant throughout software testing. To remove this assumption, researchers have proposed models incorporating testing-effort, yet this significantly increases model complexity to the degree that most previous studies utilized a two-step procedure involving least squares estimation (LSE) and algorithms, including Newton's method to estimate the parameters of a testing-effort model. This approach may limit the quality of the model fit achieved. Moreover, the research trend over the past 30 years has been to propose progressively more complex models, sacrificing practical considerations such as predictive accuracy. This paper proposes a two-step procedure that utilizes the expectation conditional maximization (ECM) algorithm, referred to as the ECM/ECM approach, to obtain the parameter estimates of a software reliability growth model incorporating testing-effort. The results of the proposed approach are compared to past methods as well as a simpler model that does not consider testing-effort to assess whether the additional complexity introduced by testing-effort functions compromises predictive accuracy. Our results indicate that the ECM/ECM approach achieves a better goodness of fit with respect to four measures, including three predictive measures. In some cases, the simpler model omitting testing-effort outperforms methods considering testing-effort. These results suggest that the proposed ECM/ECM approach can achieve better parameter estimates than the previously proposed LSE/MLE approach and that algorithms to improve fit and predictive accuracy may better serve users of software reliability models.

Keywords: software reliability; non-homogeneous Poisson process; software reliability growth models; maximum likelihood estimation; expectation conditional maximization algorithm; Newton-Raphson method; testing-effort; least squares estimation

(Submitted on September 26, 2017; Revised on February 19, 2018; Accepted on March 26, 2018)

1. Introduction

Modern technology increasingly depends on software to perform critical tasks. Therefore, it is important to ensure that software is reliable so that a system functions properly. Non-homogeneous Poisson process (NHPP) [1] software reliability models [2] are used to quantitatively assess software, including the mean time to failure (MTTF), failure intensity, optimal release time, and software reliability [3]. Many software reliability growth models (SRGM) characterize the cumulative failures detected during testing [4], assuming constant testing-effort throughout [5]. However, the effort allocated during testing may be applied non-uniformly by activities, including creating, scheduling and executing tests, reviewing the outcomes, and informing developers of necessary corrections. A testing-effort function characterizes effort in terms of various factors including the process, project, and personnel [6]. Because of the dependence of testing-effort on multiple factors, testing time may not enable accurate measurement of testing-effort. Hence, SRGMs that implicitly relate fault detection to testing-effort according to elapsed time during testing may not fail to adequately characterize fault detection [7-10]. To overcome this possible limitation, testing-effort models [11-14] were proposed, which model fault discovery according to the effort dedicated to testing activities.

* Corresponding author.
E-mail address: lfiondella@umassd.edu

Testing-effort models attempt to quantitatively link testing-effort to fault detection. However, a large number of progressively more complex testing-effort models have been developed in the past thirty years, which may characterize the observed data well but fail to predict future failures accurately. Moreover, these testing-effort models are fit with multi-step procedures, which does not guarantee that the parameter estimates obtained are indeed those that achieve the maximum likelihood estimate of the model. Specifically, previous research first identifies the parameters of a model's testing-effort function with least squares estimation [15]. The parameters are then substituted into the mean value function, and Newton's method is applied to determine the remaining model parameters. A disadvantage of this approach is that the primary application of LSE is to linear models, which assumes that the differences between the observed and predicted values are normally distributed and that the variances of these errors are constant.

This paper proposes an alternative approach to determine the parameters of a software reliability model incorporating testing-effort, which employs the expectation conditional maximization [16] algorithm. The illustrations utilize well-studied testing-effort data sets [11] to objectively assess the proposed approach and the results obtained from one of the earliest testing-effort studies by Yamada et al. [14]. We also compare these results with those produced by the simpler Goel-Okumoto (GO) model for failure count data [17] to determine if an explicit testing-effort function achieves improved predictive accuracy. These three alternative approaches are compared with respect to four goodness of fit measures, including the Akaike Information Criterion [18], predictive sum of squares error (PSSE), predictive ratio risk (PRR), and predictive power (PP) [19]. Our results indicate that the proposed approach based on the ECM algorithm achieves better AIC scores than the original approach based on a combination of LSE and MLE. Moreover, the ECM algorithm and simpler Goel-Okumoto failure counts models perform better on the three predictive measures of goodness of fit. These results suggest research efforts should focus on: (1) improving the goodness of fit of existing testing-effort functions, (2) predictive accuracy of models regardless of their simplicity in order to guide practical decisions such as optimal release, and (3) improved measures of goodness of fit to more accurately guide decision making.

The paper is structured as follows: Section 2 reviews NHPP SRGM with testing-effort. Section 3 describes the model fitting algorithms, including least squares, maximum likelihood estimation, expectation conditional maximization, and an initial parameter estimation strategy. Section 4 describes the goodness-of-fit measures. Section 5 illustrates the proposed approach, while Section 6 concludes and indicates potential future research.

2. Non-Homogeneous Poisson Process Software Reliability Growth Model with Testing-Effort Function

In software reliability, the non-homogeneous Poisson process [1] counts the faults detected by time t. In many cases, it is characterized by the mean value function (MVF).

$$m(t) = a \times F(t) \tag{1}$$

Where a is interpreted as the number of faults to be discovered if testing continued forever and $F(t)$ is a cumulative distribution function (CDF) characterizing fault detection.

The instantaneous failure rate is

$$\lambda(t) = \frac{dm(t)}{dt} \tag{2}$$

Characterizing effort expended during testing by an NHPP [14], the MVF with testing-effort is

$$m\big(w(t)\big) = a \times F\big(w(t)\big) \tag{3}$$

Where $w(t)$ represents the MVF of the testing-effort function, which has been characterized by the exponential [20] and Rayleigh [21] distributions.

Early testing-effort models [14] characterize the base counting process with the Goel-Okumoto model [20].

$$m(t) = a(1 - e^{-bt}) \tag{4}$$

Exponential [20] and Rayleigh [21] testing-effort functions are respectively denoted.

$$w_e(t) = \alpha\left(1 - e^{-\beta t}\right) \tag{5}$$

$$w_r(t) = \alpha\left(1 - e^{-\frac{\beta}{2}t^2}\right) \tag{6}$$

Where parameters a and b represent the total number of failures and rate of fault detection, whereas α and β denote the expected effort and rate of effort expenditure, respectively.

Therefore, the MVF of the GO model possessing the exponential testing-effort function is formed by substituting Equations (4) and (5) into Equation (3), which produces

$$m\left(w_e(t)\right) = a\left(1 - e^{-b\,\alpha\left(1-e^{-\beta t}\right)}\right) \tag{7}$$

Similarly, the MVF of the GO model with Rayleigh testing-effort function is obtained by substituting Equations (4) and (6) into Equation (3), which produces

$$m\left(w_r(t)\right) = a\left(1 - e^{-b\,\alpha\left(1-e^{-\frac{\beta}{2}t^2}\right)}\right) \tag{8}$$

Due to challenges in numerical stability, the earliest studies [14] employed a two-step procedure, which first fit the test effort function to identify α and β with least squares and then applied maximum likelihood estimation with Newton's method to identify a and b by holding the values of the effort parameters constant at the numerical estimates obtained in the first step.

3. Parameter Estimation Method

This section discusses the parameter estimation methods including least squares and maximum likelihood estimation methods. The expectation conditional maximization algorithm is described along with an example to illustrate mathematical steps to derive the update rules for the GO failure count model as well as the initial parameter estimation strategy.

3.1. Least Squares Estimation

Least squares estimation [15] minimizes the difference between the squares of the observed and estimated data. Given failure count data $\langle \mathbf{T}, \mathbf{K} \rangle = \langle (t_1, k_1), (t_2, k_2), \cdots, (t_n, k_n) \rangle$, where t_i represents the time spent testing (testing-effort) at which the i^{th} interval ends and k_i is the count of the faults discovered in the i^{th} interval. The least squares expression is

$$L(\boldsymbol{\theta}) = \sum_{i=1}^{n} \left(m(i) - \hat{m}(t_i)\right)^2 \tag{9}$$

Where $\boldsymbol{\Theta}$ is a vector denoting the model parameters and $m(i)$ is the total number of failures including interval i, whereas $\hat{m}(t_i)$ is the estimated number of failures. The objective of LSE is to estimate the model parameters by minimizing Equation (9) by solving the following system of equations:

$$\frac{\partial}{\partial \boldsymbol{\theta}} L(\boldsymbol{\theta}) = \mathbf{0} \tag{10}$$

Where $\mathbf{0}$ is a vector of zeros of length $p = |\boldsymbol{\Theta}|$ corresponding to the number of model parameters.

The primary application of LSE is to linear models, which assume that the differences between the observed and predicted values are normally distributed and that the variances of these errors are constant. However, these two assumptions are not satisfied in many cases; therefore, the model parameter estimates may not be accurate.

3.2. Maximum Likelihood Estimation

Maximum likelihood estimation methods maximize the likelihood function. The logarithm of the likelihood function is often maximized because the maximum of the log-likelihood matches the maximum of the likelihood.

The log-likelihood function of a failure count data is

$$LL(\mathbf{\Theta};\mathbf{T},\mathbf{K}) = \sum_{i=1}^{n} \log\left(\frac{\Delta m^{k_i} e^{\Delta m}}{k_i!}\right) \tag{11}$$

Where $\Delta m = m(t_i) - m(t_{i-1})$ and $m(t_i)$ is the MVF of an NHPP SRGM at time t_i. Traditionally, the MLE is identified numerically by applying Newton-Raphson [22] to the following set of equations:

$$\frac{\partial}{\partial \mathbf{\Theta}} LL(\mathbf{\Theta}) = \mathbf{0} \tag{12}$$

Where $\mathbf{0}$ is a vector of zeros of length p corresponding to the number of model parameters. The expectation-maximization [23] and expectation conditional maximization [24-25] algorithms have been proposed more recently and exhibit improved stability of the Newton-Raphson method.

3.3. Expectation Conditional Maximization Algorithm

The reduced log-likelihood ECM [16, 24-25] algorithm simplifies maximum likelihood estimation, dividing a single M-step of the EM algorithm [26] into p conditional-maximization (CM)-steps. The EM algorithm solves a p-dimensional M-step, whereas the CM-steps of the ECM algorithm update each parameter separately. This reduces parameter estimation to p one-dimensional problems.

The vector of parameters $\mathbf{\Theta}$ is partitioned into subvectors $\langle \mathbf{\Theta}_1, \mathbf{\Theta}_2, \cdots, \mathbf{\Theta}_p \rangle$. Each CM-step identifies $\mathbf{\Theta}_i^{(j)}$, which denotes the value of the i^{th} parameter in iteration j. This process is performed until a specified convergence criterion, such as a small difference between successive log-likelihood values, is achieved.

This section derives the CM-steps of the GO model [17] for failure count data.

- Step one substitutes the MVF of the GO model of Equation (4) into Equation (9) to obtain the log-likelihood function.

$$LL(\mathbf{\Theta};\mathbf{T},\mathbf{K}) = \sum_{i=1}^{n} k_i \log\left(a(e^{-bt_{i-1}} - e^{-bt_i})\right) - \sum_{i=1}^{n} a(e^{-bt_{i-1}} - e^{-bt_i}) - \sum_{i=1}^{n} \log(k_i!) \tag{13}$$

- Step two computes the MLE of parameter $\hat{a}$ differentiating Equation (13) and solving for a.

$$\hat{a} = \frac{\sum_{i=1}^{n} k_i}{1 - e^{-bt_n}} \tag{14}$$

- Substitute Equation (14) for $\hat{a}$ in Equation (13) to obtain the reduced log-likelihood function (RLL).

$$RLL(\mathbf{\Theta}|a;\mathbf{T},\mathbf{K}) = \sum_{i=1}^{n} k_i \log\left(\frac{(e^{-bt_{i-1}} - e^{-bt_i})\sum_{i=1}^{n} k_i}{1 - e^{-bt_n}}\right) - \sum_{i=1}^{n}\left(\frac{(e^{-bt_{i-1}} - e^{-bt_i})\sum_{i=1}^{n} k_i}{1 - e^{-bt_n}}\right) - \sum_{i=1}^{n} \log(k_i!) \tag{15}$$

- Differentiate Equation (15) with respect to b to obtain

$$b'' := \sum_{i=1}^{n} \frac{k_i\left(t_i e^{-b''t_i} - t_{i-1}e^{-b''t_{i-1}}\right)}{e^{-b''t_{i-1}} - e^{-b''t_i}} - \frac{t_n e^{-b''t_n}\sum_{i=1}^{n} k_i}{1 - e^{-b''t_n}} \tag{16}$$

Parameter b'' can be numerically solved using Newton's method. Similar steps can be followed to obtain update rules for the Rayleigh testing-effort function.

3.4. Initial Parameter Estimation

The expectation maximization (EM) [23] algorithm provides an algorithmic approach to determine initial estimates for the parameters of a model. Given an MVF possessing the form specified in Equation (1), the initial estimates of a and α are the observed number of faults and effort, respectively. Other initial parameter estimates are given by

$$\boldsymbol{\Theta}^{(0)} \coloneqq \sum_{i=1}^{n} \frac{\partial}{\partial \boldsymbol{\Theta}} \log[f(t_i; \boldsymbol{\Theta})] = \mathbf{0} \tag{17}$$

Where $f(\cdot; \boldsymbol{\Theta})$ is the probability density function.

4. Goodness-of-Fit Measures

This section discusses four popular model assessment strategies to evaluate how well a model characterizes a data set. The Akaike information criterion [18] enables relative comparison of how well models characterize a data set. The predictive sum of squares error, predictive ratio risk, and predictive power measures [19] quantify the predictive capability of a model and also enable model comparison and ranking.

4.1. Akaike Information Criterion

The Akaike Information Criterion [18] is an information theoretic goodness-of-fit measure, quantifying the trade-off between model complexity and precision. The AIC of model i is

$$AIC_i = 2p - 2LL\left(\widehat{\boldsymbol{\Theta}}; \mathbf{T}, \mathbf{K}\right) \tag{18}$$

The term $2p$ increases by two points with each additional parameter, while $LL\left(\widehat{\boldsymbol{\Theta}}; \mathbf{T}, \mathbf{K}\right)$ is the model's log-likelihood evaluated with the maximum likelihood estimates $\widehat{\boldsymbol{\Theta}}$ and failure count data. Thus, the AIC discourages models with many parameters that may lower predictive accuracy to fit the data well. If $AIC_{i,j} = AIC_i - AIC_j > 2.0$, model j is preferred.

4.2. Predictive Sum of Squares Error

PSSE compares the predictions of a model with data not used to perform model fitting. The predictive sum of squares error for failure time data is

$$PSSE = \sum_{i=n-\ell+1}^{n} \left(\widehat{m}(t_i) - m(i)\right)^2 \tag{19}$$

Where the model's MLEs are determined from the first $n - \ell$ failure counts. A common choice of ℓ is approximately 10% of the available data, so that $\ell/n \approx 0.1$ and $m(i)$ represents the cumulative number of observed failures through the i^{th} interval.

4.3. Predictive Ratio Risk

The predictive ratio risk for failure count data is

$$PRR = \sum_{i=\ell+1}^{n} \left(\frac{\widehat{m}(t_i) - m(i)}{\widehat{m}(t_i)}\right)^2 \tag{20}$$

Note that the term in the denominator penalizes underestimation of the number of faults more heavily than overestimation, which is appropriate in the context of software reliability because underestimation can lead to failures after software release.

4.4. Predictive Power

The predictive power for failure count data is

$$PP = \sum_{i=\ell+1}^{n} \left(\frac{\hat{m}(t_i) - m(i)}{m(i)} \right)^2 \tag{21}$$

5. Illustrations

This section compares historical results [14], which proposed testing-effort functions characterized by exponential and Rayleigh models. These previous studies reported that these two testing-effort functions best characterized DS1 and DS2 [11], respectively. Therefore, we compare the previous results obtained with the two-step procedure proposed in [14] against a similar two-step procedure that applies the ECM algorithm to identify the MLEs of the effort parameters and then applies the ECM algorithm to estimate the parameters of the base model by holding the effort parameters constant. To assess whether the additional complexity of testing-effort is necessary, we also apply the simpler failure count Goel-Okumoto model [17], which employs the ECM algorithm.

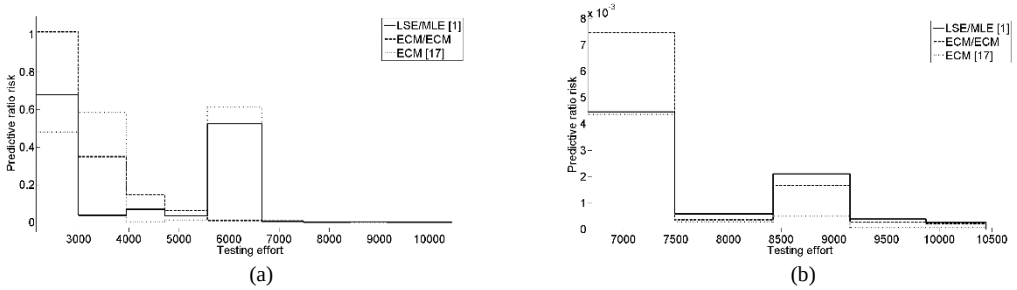
Table 1 lists the method applied, number of parameters, base model parameter estimates, and goodness-of-fit when applied to the DS1 data set [11]. The AIC values were computed by fitting the models to 100% of the data, whereas the other measures were computed by fitting with 90% percent of the data and computing the measures with the remaining 10%.

Table 1. Model results (DS1 data set [11])

Model Name	*p*	*a*	*b*	**AIC**	**PSSE**	**PRR**	**PP**
				100%	90%	90%	90%
LSE/MLE [14]	4	3800.9	1.14×10^{-4}	175.86	11925.5	3.88×10^{-4}	4.04×10^{-4}
ECM/ECM	4	3243.7	3.24×10^{-1}	**172.41**	11922.7	2.69×10^{-4}	2.78×10^{-4}
ECM [17]	**2**	3770.8	1.17×10^{-4}	216.23	**9440.4**	$\mathbf{7.19 \times 10^{-5}}$	$\mathbf{7.31 \times 10^{-5}}$

The first two rows in Table 1 report values obtained for the Goel-Okumoto model with the exponential testing-effort function as identified in [14] and with the proposed ECM/ECM approach, respectively. Table 1 indicates that the ECM/ECM approach achieves a statistically significant better AIC. Moreover, row three indicates that the simpler Goel-Okumoto failure counts model performed better than the models with testing-effort. These observations suggest that our algorithmic approach can achieve better model fit and that greater emphasis should be placed on prediction rather than the trend of proposing more complex testing-effort functions observed in literature over the past 30 years. This focus on prediction will be necessary to ensure the accuracy of optimal release decisions [27] and their corresponding cost and safety implications.

Figure 1(a) shows the PRR values when applied to an increasing prefix of the DS1 data, which is how the model would be used in practice as data is collected. Figure 1(b) shows the zoomed version of Figure 1(a) to compare the PRR values of all the models more clearly.



(a) (b)

Figure 1. (a) PRR (DS1 data set [11]); (b) Zoomed in version

Figure 1(a) indicates that the LSE/MLE method of [14] performs best early on when little data has been observed. However, Figure 1(b) shows that the proposed ECM/ECM method and Goel-Okumoto failure count model produce lower PRR values after approximately 75% of the data has been observed. Thus, these later approaches may be appropriate when testing is approaching completion and an optimal release decision is to be made.

Table 2 lists the method applied, number of parameters, base model parameter estimates, and goodness-of-fit when applied to the DS2 data set [11].

Table 2. Model results (DS2 data set [11])

Model Name	p	a	b	**AIC**	**PSSE**	**PRR**	**PP**
				100%	90%	90%	90%
LSE/MLE [14]	4	1389.5	1.69×10^{-4}	416.73	46120.2	1.86×10^{-3}	1.96×10^{-3}
ECM/ECM	4	1328.6	1.31×10^{0}	**386.27**	**30072.9**	$\mathbf{6.42 \times 10^{-4}}$	$\mathbf{6.61 \times 10^{-4}}$
ECM [17]	**2**	1485.3	1.19×10^{-3}	688.16	94159.1	7.37×10^{-4}	7.61×10^{-4}

The first two rows in Table 2 are the results for the Goel-Okumoto model with the Rayleigh testing-effort function as identified in [14] and the proposed ECM/ECM method, respectively. The ECM/ECM approach achieves a statistically significant better AIC as well as better predictive measures. These results further support the argument that model fitting and prediction are of equal or greater importance to more complex testing-effort functions, which have rarely been validated other than with statistical measures.

Figure 2(a) shows the PRR values when applied to an increasing prefix of the DS2 data, which is how the model would be used in practice as data is collected. Figure 2(b) shows the zoomed version of Figure 2(a) to compare the PRR values of all the models more clearly.

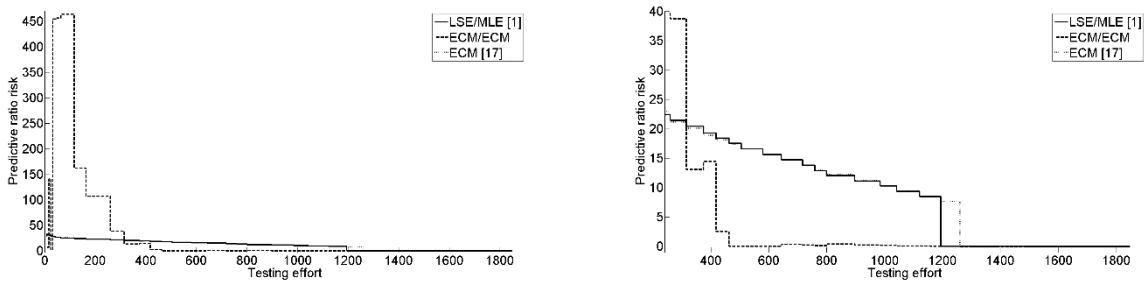


Figure 2. (a) PRR (DS2 data set [11]); (b) Zoomed in version

Figure 2(a) exhibits trends similar to those in Figure 1(a), where the LSE/MLE method attains lower PRR values at the beginning of testing, but the ECM/ECM method consistently produces significantly lower PRR values after approximately 300 units of effort or less than 20% of the data has been observed.

6. Conclusions and Future Work

The present study proposed a two-step procedure to obtain the parameter estimates of software reliability models incorporating testing-effort. The approach first employs the ECM algorithm to identify the MLEs of the testing-effort function parameters and then applies the ECM algorithm to identify the MLEs of the parameters of the base model by holding the effort parameters constant at the values identified in the first step. The results of the proposed approach are compared with some of the earliest testing-effort studies [14], which employed a similar two-step approach, where least squares was employed to identify the effort parameters and maximum likelihood estimation to determine the parameters of the base model. We also compared these results with the simpler failure count Goel-Okumoto model to assess whether the additional complexity introduced by the testing-effort functions compromises predictive capability. Our results indicate that in both cases the ECM/ECM approach achieved a better AIC score than the original LSE/MLE approach and that the simpler Goel-Okumoto model achieves the lowest AIC score. Moreover, the studies also revealed that the ECM/ECM and simpler Goel-Okumoto model attained better goodness of fit with respect to three popular predictive measures. Given the practical importance of prediction to problems such as optimal release, the present study suggests renewed emphasis be placed on predictive accuracy not progressively more complex testing-effort models that may fail to serve the need of practitioners.

Future research will design ECM algorithms to simultaneously identify the MLEs of the effort and base parameters of a SRGM incorporating testing-effort. Improved measures of goodness-of-fit to more effectively mitigate risk will also be developed.

Acknowledgments

This work was partially supported by the Naval Air Warfare Center (NAVAIR) under contract N00421-16-T-0373 and the National Science Foundation under Grant Number #1526128.

References

1. S. Ross, "Introduction to Probability Models," 8th Edition, Academic Press, New York, 2003
2. M. Lyu, "Handbook of Software Reliability Engineering," McGraw-Hill, New York, 1996
3. L. Leemis, "Reliability: Probabilistic Models and Statistical Methods," Prentice-Hall, Englewood Cliffs, NJ, 1995
4. J. Musa, "A Theory of Software Reliability and its Application," *IEEE Transactions on Software Engineering*, Vol. SE-1, No. 3, pp. 312-327, 1975
5. S. Gokhale, P. Marinos, and K. Trivedi, "Important Milestones in Software Reliability Modeling," *Communications in Reliability, Maintainability, and Serviceability*, 1996
6. B. Boehm, "Software Engineering Economics," Prentice-Hall, Englewood Cliffs, NJ, 1981
7. J. Musa and K. Okumoto, "A Logarithmic Poisson Execution Time Model for Software Reliability Measurement," in *Proceedings of International Conference on Software Engineering*, 1984
8. M. Trachtenberg, "A General Theory of Software-Reliability Modeling," *IEEE Transactions on Reliability*, Vol. 39, No. 1, pp. 92-96, 1990
9. W. Everett, "An Extended Execution Time Software Reliability Model," in *Proceedings of International Symposium on Software Reliability Engineering*, pp. 4-13, 1992
10. J. Tian, P. Lu, and J. Palma, "Test-Execution-based Reliability Measurement and Modeling for Large Commercial Software," *IEEE Transactions on Software Engineering*, Vol. 21, No. 5, pp. 405-414, 1995
11. W. Brooks and R. Motley, "Analysis of Discrete Software Reliability Models (RADC-TR-80-84)," 1980
12. M. Ohba, "Software Reliability Analysis Models," *IBM Journal of Research and Development*, Vol. 28, No. 4, pp. 428-443, 1984
13. Y. Tohma, R. Jacoby, Y. Murata, and M. Yamamoto, "Hyper-Geometric Distribution Model to Estimate the Number of Residual Software Faults," in *Proceedings of Computer Software and Application Conference*, 1989
14. S. Yamada, H. Ohtera, and H. Narihisa, "Software Reliability Growth Models with Testing-Effort," *IEEE Transactions on Reliability*, Vol. R-35, No. 1, pp. 19-23, 1986
15. D. Kleinbaum, L. Kupper, A. Nizham, and K. Muller, "Applied Regression Analysis and other Multivariable Methods," 4th Edition, Thomson Learning Inc. Brooks/Cole, Australia, 2008
16. V. Nagaraju, L. Fiondella, P. Zeephongsekul, C. Jayasinghe, and T. Wandji, "Performance Optimized Expectation Conditional Maximization Algorithms for Nonhomogeneous Poisson Process Software Reliability Models," *IEEE Transactions on Reliability*, Vol. 66, No. 3, pp. 722-734, 2017
17. V. Nagaraju, A. K. Murthy, L. Fiondella, P. Zeephongsekul, and T. Wandji, "Expectation Conditional Maximization Algorithms for Failure Count Non-Homogeneous Poisson Process Software Reliability Models," in *Proceedings of ISSAT International Conference on Reliability and Quality in Design*, 2016
18. L. Fiondella and S. Gokhale, "Software Reliability Model with Bathtub-Shaped Fault Detection Rate," in *Proceedings of Annual Reliability and Maintainability Symposium*, Orlando, FL, 2011
19. K. Sharma, R. Garg, C. Nagpal, and R. Garg, "Selection of Optimal Software Reliability Growth Models using a Distance based Approach," *IEEE Transactions on Reliability*, Vol. 59, No. 2, pp. 266-276, 2010
20. A. Goel and K. Okumoto, "Time-Dependent Error-Detection Rate Model for Software Reliability and other Performance Measures," *IEEE Transactions on Reliability*, Vol. 28, No. 3, pp. 206-211, 1979
21. L. Putnam, "A General Empirical Solution to the Macro Software Sizing and Estimating Problem," *IEEE Transactions on Software Engineering*, Vol. SE-4, No. 4, pp. 345-361, 1978
22. R. Burden and J. Faires, "Numerical Analysis," 8th Edition, Brooks/Cole, Belmont, CA, 2004
23. H. Okamura, Y. Watanabe, and T. Dohi, "An Iterative Scheme for Maximum Likelihood Estimation in Software Reliability Modeling," in *Proceedings of International Symposium on Software Reliability Engineering*, 2003
24. P. Zeephongsekul, C. Jayasinghe, L. Fiondella, and V. Nagaraju, "Maximum Likelihood Estimation of Parameters of NHPP Software Reliability Models using EM and ECM Algorithms," *IEEE Transactions on Reliability*, Vol. 65, No. 3, pp. 1571-1583, 2016
25. V. Nagaraju, T. Wandji, and L. Fiondella, "An Implicit Expectation Conditional Maximization Algorithm for Non-Homogeneous Poisson Process Software Reliability Models," in *Proceedings of Conference on Applied Statistics in Defense*, Fairfax, VA, 2016
26. A. Dempster, N. Laird, and D. Rubin, "Maximum Likelihood from Incomplete Data via the EM Algorithm," *Journal of the Royal Statistical Society*: *Series B*, Vol. 39, No. 1, pp. 1-38, 1977
27. M. Zhao and M. Xing, "Robustness of Optimum Software Release Policies," in *Proceedings of IEEE International Symposium on Software Reliability Engineering*, pp. 218-225, 1993