Journal of Modelling in Management

Emerald Article: Modelling and analysis of software reliability with Burr type X testing-effort and release-time determination

N. Ahmad, M.G.M. Khan, S.M.K. Quadri, M. Kumar

Modelling and analysis of software reliability with Burr type X testing-effort and release-time determination

N. Ahmad
School of Computing, Information and Mathematical Sciences, The University of the South Pacific, Suva, Fiji Islands
M.G.M. Khan
Department of Statistics, Iowa State University, Ames, Iowa, USA
S.M.K. Quadri
Department of Computer Science, University of Kashmir, Srinagar, India, and
M. Kumar
Department of Statistics, Mathematics, and Computer Applications, Rajendra Agriculture University, Samastipur, India

Abstract

Purpose – The purpose of this research paper is to discuss a software reliability growth model (SRGM) based on the non-homogeneous Poisson process which incorporates the Burr type X testing-effort function (TEF), and to determine the optimal release-time based on cost-reliability criteria.

Design/methodology/approach – It is shown that the Burr type X TEF can be expressed as a software development/testing-effort consumption curve. Weighted least squares estimation method is proposed to estimate the TEF parameters. The SRGM parameters are estimated by the maximum likelihood estimation method. The standard errors and confidence intervals of SRGM parameters are also obtained. Furthermore, the optimal release-time determination based on cost-reliability criteria has been discussed within the framework.

Findings – The performance of the proposed SRGM is demonstrated by using actual data sets from three software projects. Results are compared with other traditional SRGMs to show that the proposed model has a fairly better prediction capability and that the Burr type X TEF is suitable for incorporating into software reliability modelling. Results also reveal that the SRGM with Burr type X TEF can estimate the number of initial faults better than that of other traditional SRGMs.

Research limitations/implications – The paper presents the estimation method with equal weight. Future research may include extending the present study to unequal weight.

Practical implications – The new SRGM may be useful in detecting more faults that are difficult to find during regular testing, and in assisting software engineers to improve their software development process.

Originality/value – The incorporated TEF is flexible and can be used to describe the actual expenditure patterns more faithfully during software development.

Keywords Program testing, Computer software, Modelling

Paper type Research paper

Journal of Modelling in Management
Vol. 4 No. 1, 2009
pp. 28-54

1746-5664
DOI 10.1108/17465660910943748

1. Introduction

Software is playing an important role in many complex and safety-critical system. As a result, software reliability has become a common concern for both developers and users. In general, software reliability is defined as the probability of failure-free software operation for a specified period of time in a specified environment (Musa *et al.*, 1987; Musa, 1999; Lyu, 1996; Kapur *et al.*, 1999). Therefore, accurately modeling software reliability and predicting its possible trends are essential for determining overall product's reliability. Numerous software reliability growth models (SRGMs) have been developed during the last three decades and they have applied successfully in practice to improve reliability (Musa *et al.*, 1987; Xie, 1991; Lyu, 1996; Pham, 2000). SRGMs provide essential information for decision making in many software development activities, such as in the number of initial faults, failure intensity, reliability within a specific time period, number of remaining faults, mean time between failures (MTBF), mean time to failure (MTTF), cost analysis and release time decision, etc.

The testing phase is an important part of software and the key element in software testing is the testing-effort. The functions that describe how an effort is distributed over the exposure period, and how effective it is, are referred to by us as testing-effort functions (TEF). The TEF can be represented as the number of CPU hours, the number of executed test cases, etc. (Yamada and Osaki, 1985a; Yamada *et al.*, 1986, 1993). Non-homogeneous Poisson process (NHPP) as a stochastic process has been successfully used in the reliability analysis of software system. Several SRGMs based on NHPP which incorporates the TEF have been proposed by Yamada *et al.* (1986, 1987, 1991, 1993), Yamada and Ohtera (1990), Kapur and Garg (1990, 1996), Kapur and Younes (1994), Kapur *et al.* (2004), Huang *et al.* (1997, 1999, 2000, 2007), Kuo *et al.* (2001), Huang and Kuo (2002), Huang (2005), Ahmad *et al.* (2008), Bokhari and Ahmad (2006, 2007) and Quadri *et al.* (2006). Most of these works assume that the time-dependent behavior of TEF expenditure is either Weibull type, logistic, log-logistic or generalized logistic curves. However, in many software testing situations, it is sometimes difficult to describe the TEF expenditure only by these curves, since actual software data show various expenditure patterns.

Burr (1942) introduced 12 different forms of cumulative distribution functions for modeling actual data. Thus, in this paper, we investigate a SRGM based on NHPP with Burr type X distribution, since its curve is a flexible with a wide variety of possible expenditure patterns in actual software projects. Hence, these curves are called a Burr type X TEF, which includes the Rayleigh curve. It is also called generalized Rayleigh curve. Currently, there are few studies for the use of the Burr type X distribution in reliability and survival analysis (Surles and Padgett, 2001, 2005), so this paper is also to promote its use in software reliability applications.

In this paper, we incorporate the Burr type X TEF into SRGM and show that the Burr type X TEF can be expressed as a software development/testing-effort consumption curve. SRGM with Burr type X TEF parameters are estimated by weighted least squares estimation (WLSE) and maximum likelihood estimation (MLE) methods. We also obtain the standard errors and confidence intervals of SRGM parameters. Experiments have been performed based on three actual software data sets (DS) from various projects and the results show that the proposed SRGM with Burr type X TEF is wide and effective models for software reliability analysis and is

more realistic. Comparisons of predictive capability between various SRGMs are presented and the results reveal that the SRGM with Burr type X TEF can estimate the number of initial faults better than that of other traditional SRGMs. In addition, the optimal release-time determination of this model based on cost-reliability criterion is discussed.

2. Software reliability growth modeling

2.1 Burr type X TEF

In software testing, many testing-efforts are consumed, such as the CPU time, the human power and the executed test cases. The consumed testing-effort indicates how the errors are detected effectively in the software and can be modeled by different distributions (Putnam, 1978; Pillai and Nair, 1997; Musa *et al.*, 1987; Musa, 1999; Yamada *et al.*, 1986, 1993; Yamada and Ohtera, 1990; Kapur *et al.*, 1999; Huang and Kuo, 2002; Ahmad *et al.*, 2008; Bokhari and Ahmad, 2006, 2007). Actually, the software reliability is highly related to the amount of testing-effort expenditures spent on detecting and correcting software errors. Therefore, we propose the Burr type X TEF. The Burr type X TEF over time period $(0,t]$ is given by:

$$W(t) = \alpha\left(1 - e^{-\beta t^2}\right)^{\theta}, \quad \alpha > 0, \ \ \beta > 0, \ \ \theta > 0, \tag{1}$$

and the current testing-effort consumed at testing time t is:

$$w(t) = W'(t) = 2\alpha \cdot \beta \cdot \theta \cdot t \cdot e^{-\beta t^2}\left(1 - e^{-\beta t^2}\right)^{\theta - 1}, \tag{2}$$

where α, β and θ are constant parameters, α is the total amount of testing-effort expenditures; β is the scale parameter, and θ is shape parameter.

When $\theta > 1/2$, the Burr type X TEF $w(t)$ reaches its maximum value at the time:

$$t_{\max} = \left[\frac{(2\theta - 1)}{\beta(\theta + 1)}\right]^{1/2}.$$

In particular, for $\theta = 1$, there is a Rayleigh TEF (Yamada *et al.*, 1986), and the cumulative testing-effort consumed in time $(0, t]$ is:

$$W(t) = \alpha\left(1 - e^{-\beta t^2}\right), \quad \alpha > 0, \ \ \beta > 0.$$

2.2 SRGM with Burr type X TEF

Based on the following assumptions, we formulate an SRGM with a Burr type X TEF (Yamada and Osaki, 1985a; Yamada *et al.*, 1986, 1993; Kapur *et al.*, 1999; Kuo *et al.*, 2001; Huang and Kuo, 2002; Huang, 2005; Ahmad *et al.*, 2008):

- The error detection process in software testing follows the NHPP.
- The software system is subject to failures at random times caused by errors remaining in the system.
- Each time a failure occurs, the error that caused it is immediately removed and no new errors are introduced.

- The consumption of testing-effort is modeled by a Burr type X TEF.
- The mean number of errors detected in the time interval $(t, t + \Delta t]$ to the current testing-effort expenditures is proportional to the mean number of remaining errors in the system.
- The proportionality of error detection is a constant over time.

For stochastic modelling of a software error detection phenomenon, we define a counting process, $[N(t), t \geq 0]$, where $N(t)$ represents the cumulative number of software errors detected by testing time t with mean value function $m(t)$. We can then formulate an SRGM based on NHPP as:

$$Pr[N(t) = n] = \frac{[m(t)]^n \cdot e^{-m(t)}}{n!}, \quad n = 0, 1, 2, \ldots \tag{3}$$

During the testing phase, software errors remaining in the system cause software failures and the errors are detected, and corrected by test personnel. Based on the assumptions, if the number of detected errors by the current testing-effort expenditures is proportional to the number of remaining errors, then we obtain the following differential equation (Yamada and Osaki, 1985a; Yamada *et al.*, 1986, 1993; Yamada and Ohtera, 1990; Huang and Kuo, 2002; Huang, 2005; Ahmad *et al.*, 2008; Bokhari and Ahmad, 2006):

$$\frac{\mathrm{d}m(t)}{\mathrm{d}t} \cdot \frac{1}{w(t)} = r[a - m(t)], \quad a > 0, \;\; 0 < r < 1, \tag{4}$$

where $m(t)$ represent the expected mean number of errors detected in time $(0, t]$ which is assumed to be a bounded non-decreasing function of t with $m(0) = 0$, $w(t)$ is the current testing-effort expenditure at time t, α is the expected number of initial error in the system, and r is the error detection rate per unit testing-effort at time t. Solving the above differential equation, we have:

$$m(t) = a\left(1 - e^{-rW(t)}\right). \tag{5}$$

Substituting $W(t)$ from equation (1), we get:

$$m(t) = a\left(1 - e^{-r\alpha(1-e^{-\beta t^2})^{\theta}}\right). \tag{6}$$

This is an NHPP model with mean value function considering the Burr type X testing-effort expenditure. From equation (5), we have the following important relationship between $m(t)$ and $W(t)$:

$$W(t) = \frac{1}{r} \log\left[\frac{a}{a - m(t)}\right]. \tag{7}$$

In addition, the failure intensity at testing time t of the NHPP is given by:

$$\lambda(t) = \frac{\mathrm{d}m(t)}{\mathrm{d}t} = a \cdot r \cdot w(t) e^{-rW(t)}. \tag{8}$$

The expected number of errors to be detected eventually is:

$$m(\infty) = a(1 - e^{-r\alpha}). \tag{9}$$

This implies that even if a software system is tested during an infinitely long duration, all errors remaining in the system cannot be detected (Yamada *et al.*, 1986, 1993). Thus, the mean number of undetected errors if a test is applied for an infinite amount of time is:

$$a - m(\infty) = ae^{-r\alpha}.$$

That is, not all the original errors in a software system can be fully detected with a finite testing effort since the effort expenditure is limited to α.

2.3 Software reliability measures

We can derive the following quantitative measures for software reliability assessments (Goel and Okumoto, 1979; Yamada *et al.*, 1993). If $\bar{N}(t)$ represent the number of errors remaining in the system at testing time t, then the mean of $\bar{N}(t)$ and its variance are given by:

$$\begin{aligned} r(t) &= E[\bar{N}(t)] = E[N(\infty) - N(t)] = m(\infty) - m(t) \\ &= a\left(e^{-rW(t)} - e^{-rW(\infty)}\right) = \mathrm{Var}[\bar{N}(t)]. \end{aligned} \tag{10}$$

The software reliability representing the probability that no failures occur in the time interval $(t, t + \Delta t)$ given that the last failure occurred at testing time Δt, is given by:

$$R = R(\Delta t|t) = e^{-[m(t+\Delta t)-m(t)]} = e^{-a[e^{-rW(t)}-e^{-rW(t+\Delta t)}]}. \tag{11}$$

The instantaneous MTBF at arbitrary testing can be defined as a reciprocal of error detection rate in equation (8). Then, the instantaneous MTBF is given by:

$$\mathrm{MTBF}(t) = \frac{1}{\lambda(t)} = \frac{e^{(\beta t^2 + r\alpha(1-e^{-\beta t^2})^{\theta})}}{2ar\alpha\beta\theta \cdot t(1 - e^{-\beta t^2})^{\theta-1}}. \tag{12}$$

3. Parameter estimation methods

MLE and LSE techniques are used to estimate the model parameters (Musa *et al.*, 1987; Musa, 1999; Lyu, 1996). Estimation by maximum likelihood is a general technique that may be applied when the underlying distributions of the data are specified or known and is better in deriving confidence intervals and the asymptotic normal distribution for ML estimates. On the other hand, LSE is a fairly general technique which is applied in most practical situations for small or medium size data for better estimates (Musa *et al.*, 1987; Huang *et al.*, 1997; Huang and Kuo, 2002). In using LSE, one of the common assumptions is that the standard deviation of the error term is constant over all values of the predictor variable. This assumption, however, clearly does not hold in every modeling application. For example, in testing effort data, it may appear that the precision of testing effort varies as the time changes. In situation like this, when it may not be reasonable to assume every observation should be treated equally, one could use WLSE to maximize the efficiency of parameter estimation. It minimizes the weighted

sum of squares of the deviations between what we expect and what we actually observe. WLSE, like LSE, is an efficient method that makes good use of small DS.

3.1 Weighted least squares method

The parameters α, β and θ in the Burr type X TEF (1) can be estimated by the method of WLSE. These parameters are determined for n observed data pairs in the form (t_k, W_k) $(k = 1, 2, \ldots, n;\ 0 < t_1 < t_2 < \cdots < t_n)$, where W_k is the cumulative testing-effort consumed in time $(0, t_k]$. The estimators $\hat{\alpha}$, $\hat{\beta}$ and $\hat{\theta}$, which contribute the model with a greater fitting, can be obtained by minimizing (see Appendix):

$$S'(\alpha, \beta, \theta) = \sum_{k=1}^{n} l_k [W_k - W(t_k)]^2, \tag{13}$$

where l_k are weight assigned to data points according to their proper amount of influence over the parameter estimates.

3.2 Maximum likelihood method

Suppose that the estimated TEF parameters,$\hat{\alpha}$, $\hat{\beta}$ and $\hat{\theta}$ in the Burr Type X TEF have been obtained by the method of weighted least squares discussed earlier. The estimators for a and r are determined for the n observed data pairs in the form (t_k, m_k) $(k = 1, 2, \ldots, n;\ 0 < t_1 < t_2 < \cdots < t_n)$, where m_k is the cumulative number of software errors detected up to time t_k or $(0, t_k]$. Then the likelihood function for the unknown parameters a and r in the NHPP model with $m(t)$ in equation (6) is given by Kapur *et al.* (1999) and Musa *et al.* (1987) (see Appendix):

$$\begin{aligned} L'(a, r) &\equiv P\{N(t_i) = m_i, \quad i = 1, 2, \ldots, n\} \\ &= \prod_{k=1}^{n} \frac{[m(t_k) - m(t_{k-1})]^{(m_k - m_{k-1})} \cdot e^{-[m(t_k) - m(t_{k-1})]}}{(m_k - m_{k-1})!}, \end{aligned} \tag{14}$$

where $m_0 \equiv 0$ for $t_0 \equiv 0$.

4. Experimental results and comparative studies

4.1 Criteria for TEF comparison

To check the performance of Burr type X TEF and make a fair comparison with the other TEF, we describe the following comparison criteria:

- The coefficient of multiple determination is defined (Musa *et al.*, 1987; Musa, 1999) as:

$$R^2 = \frac{S(\hat{\hat{\alpha}}, 0, 1) - S(\hat{\alpha}, \hat{\beta}, \hat{\theta})}{S(\hat{\hat{\alpha}}, 0, 1)},$$

  where $\hat{\hat{\alpha}}$ is the LSE of α for the model with only a constant term, that is, $\beta = 0$ and $\theta = 1$ in equation (13). It is given by $\ln \hat{\hat{\alpha}} = (1/n)\sum_{k=1}^{n} \ln W_k$. Therefore, R^2 measures the percentage of total variation about the mean accounted for by the fitted model and tells us how well a curve fits the data. It is frequently employed to compare models and assess which model provides the best fit to the data. The best model is the one which provides the higher R^2, that is, closer to 1 (Kumar *et al.*, 2005). To investigate whether a significant trend exists in the

estimated TEF, one could test the hypotheses H_0:$\beta = 0$ and $\theta = 1$, against H_1:$\beta \neq 0$ or $\theta \neq 1$ using F-test by merely forming the ratio:

$$F = \frac{[S(\hat{\alpha}, 0, 1) - S(\hat{\alpha}, \hat{\beta}, \hat{\theta})]/2}{S(\hat{\alpha}, 0, 1)/(n-3)}.$$

If the value of F is greater than $F_\alpha(2, \ n-3)$, which is the α percentile of the F distribution with degrees of freedom 2 and $n-3$, we can be $(1-\alpha)100$ percent confident that H_0 should be rejected, that is, there is a significant trend in the TEF curve.

- Prediction error (PE_i) is defined (Huang and Kuo, 2002; Huang *et al.*, 2007; Pillai and Nair, 1997) as:

$$PE_i = \text{Actual(Observed)}_i - \text{Predicted(Estimated)}_i$$

- $$\text{Bias} = \frac{1}{n} \cdot \sum_{i=1}^{n} PE_i$$

- $$\text{Variation} = \sqrt{\frac{\sum_{i=1}^{n} (PE_i - \text{Bias})^2}{n-1}}$$

- Root mean square prediction error (RMS-PE) is:

$$\text{RMS} - \text{PE} = \sqrt{\text{Bias}^2 - \text{Variation}^2}$$

4.2 Criteria for SRGM comparison

In order to evaluate the performance of our SRGM and to make a fair comparison with the other existing SRGMs, we describe the following comparison criteria:

- The accuracy of estimation (AE) is defined (Musa *et al.*, 1987; Yamada and Osaki, 1985a; Huang and Kuo, 2002; Kuo *et al.*, 2001) as:

$$\text{AE} = \left| \frac{M_\text{a} - a}{M_\text{a}} \right|,$$

where M_a is the actual cumulative number of detected errors after the test, and a is the estimated number of initial errors. For practical purposes, M_a is obtained from software error tracking after software testing.

- The mean of squared errors (long-term predictions) is defined (Lyu, 1996; Huang and Kuo, 2002; Kuo *et al.*, 2001) as:

$$\text{MSE} = \frac{1}{k} \cdot \sum_{i=1}^{k} [m(t_i) - m_i]^2,$$

where $m(t_i)$ is the expected number of errors at time t_i estimated by a model, and m_i is the observed number of errors at time t_i. MSE gives the quantitative comparison for long-term predictions. A smaller MSE indicates a minimum

fitting error and better performance (Huang *et al.*, 1997; Kapur and Garg, 1996; Kapur *et al.*, 1999).

- The predictive validity is defined (Musa *et al.*, 1987; Musa, 1999) as the capability of the model to predict future failure behavior from present and past failure behavior. Assume that we have observed q failures by the end of test time t_q. We use the failure data up to time $t_e(\leq t_q)$ to determine the parameters of $m(t)$. Substituting the estimates of these parameters in the mean value function yields the estimate of the number of failures $\hat{m}(t_q)$ by t_q. The estimate is compared with the actually observed number q. This procedure is repeated for various values of t_e. The ratio:

$$\frac{\hat{m}(t_q) - q}{q}$$

is called the relative error. Values close to zero for relative error indicate more accurate prediction and hence a better model. We can visually check the predictive validity by plotting the relative error for normalized test time t_e/t_q.

4.3 Description of actual DS

- *DS1.* The first set of actual data is from the study by Ohba (1984). The system is PL/1 data base application software, consisting of approximately 1,317,000 lines of code. During the 19-weeks experiments, 47.65 CPU hours were consumed and about 328 software errors were removed. The study reports that the total cumulative number of detected faults after a long period of testing is 358.
- *DS2.* The second set of actual data is the pattern of discovery of errors by Tohma *et al.* (1989). The debugging time and the number of detected faults per day are reported. The cumulative number of discovered faults up to 22 days is 86 and the total consumed debugging times is 93 CPU hours. All debugging data are used in this experiment.
- *DS3.* The third set of actual data in this paper is the System T1 data of the Rome Air Development Center projects and cited from Musa *et al.* (1987) and Musa (1999). The number of object instructions for the system T1 which is used for a real-time command and control application. The size of the software is approximately 21,700 object instructions and developed by Bell Laboratories. The software was tested for 21 weeks with nine programmers. During the testing phase, about 25.3 CPU hours were consumed and 136 software errors were removed. The number of errors removed after 3.5 years of test was reported to be 188 (Huang, 2005).

4.4 Model performance analysis

In order to validate the proposed model and to compare its performance with other existing models, experiments on above three actual software failure DS will be performed.

DS1. In order to estimate the parameters α, β and θ of the Burr type X TEF; we fit the actual testing effort data into equation (1) and solve it by using the method of weighted least squares. That is, we minimize the weighted sum of squares given in equation (13) with equal weight. The estimated parameters are obtained as:

$$\hat{\alpha} = 178.35202, \quad \hat{\beta} = 0.000277, \quad \text{and} \quad \hat{\theta} = 0.5585. \tag{15}$$

Figures 1 and 2 graphically show the comparisons between the observed failure data and the estimated Burr type X TEF data. Here, the fitted curves are shown as a doted line and the solid line represents actual software data. Using the estimated parameters α, β and θ, the SRGM parameters a, r in equation (6) can be solved numerically by the MLE method. These estimated parameters are:

$$\hat{a} = 565.6733, \quad \hat{r} = 0.019639.$$

The R^2 value for proposed Burr type X TEF is 0.9957. It can therefore be observed that the Burr type X TEF is suitable for modeling the software reliability of this data set. We also observed that the fitted TEF curve is significant since the calculated value $F(= 7.9661)$ is greater than $F_{0.05}(2, 16)$ and $F_{0.01}(2, 16)$. In addition, we computed the bias, the variation, the RMS-PE and the PE and are shown in Table I. Table I shows that the Burr type X TEF has lower values of bias, variation, RMS-PE and PE. This reveals that Burr type X TEF yields a better fit for this data set.

Table II summarizes the estimated values of parameters with their standard errors and 95 percent confidence limits for the proposed model. A fitted curve of the estimated mean value function with the actual software data is shown in Figure 3.

Secondly, the selected models are compared with each other based on SRGM objective criteria. Table III lists the performance of various SRGM investigated. Kolmogorov Smirnov goodness-of-fit test shows that our proposed SRGM fits pretty well at the 5 percent level of significance. Following the work in Musa *et al.* (1987), we computed the relative error in prediction for this data set and the results are shown in Figure 4. We observed that relative error approaches zero as t_e approaches t_q and the

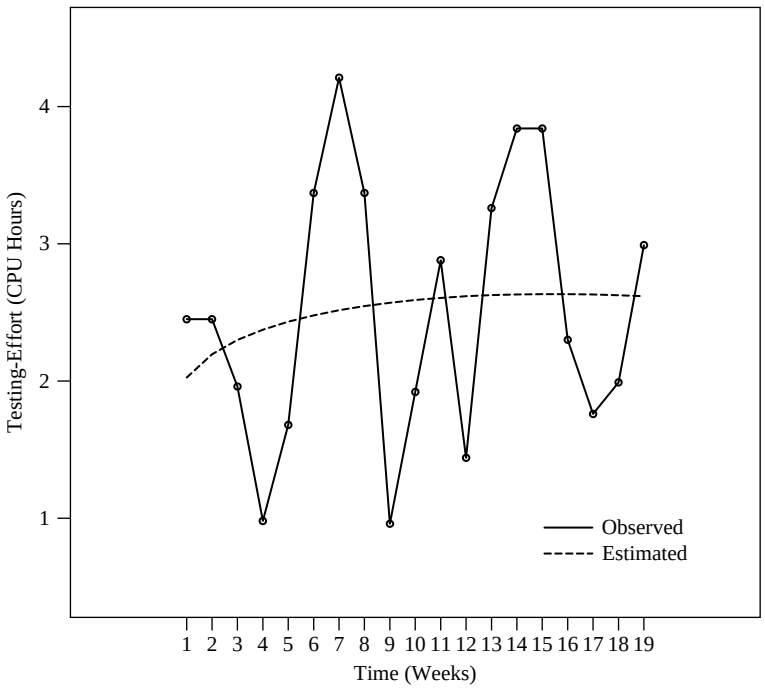

Figure 1.
Observed/estimated current testing-effort for DS1

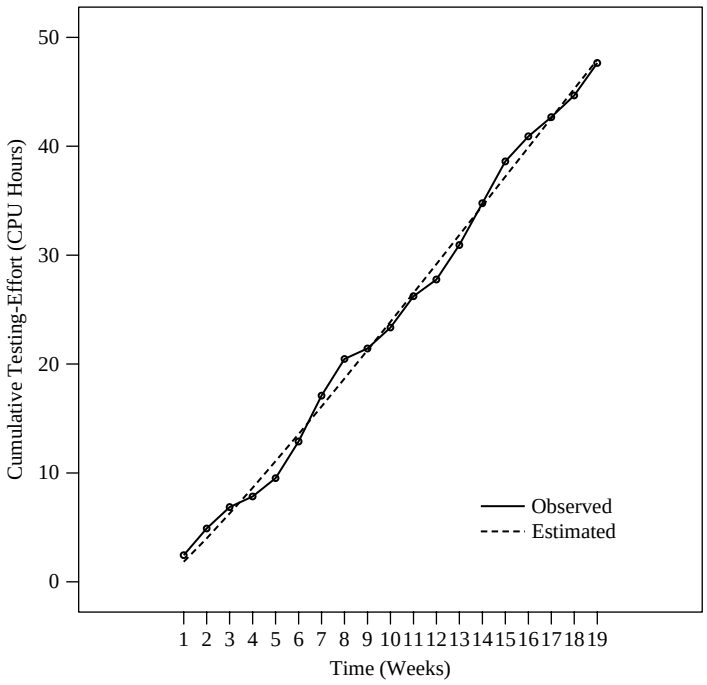

Figure 2.
Observed/estimated cumulative testing-effort for DS1

TEF	Bias	Variation	RMS-PE	$PE_{\text{end of testing}}$
Burr type X TEF	0.038	0.94	0.94	−0.30
Yamada exponential	−0.394	1.37	1.42	1.27
Yamada Rayleigh	0.830	2.17	2.02	2.50
Yamada Weibull	0.034	0.96	0.96	−0.38
Huang logistic	−0.032	1.11	1.91	1.05

Table I.
Comparison results of different TEF for DS1

Parameter	Estimate	Standard error	95 percent confidence limits Lower	Upper
a	565.6733	56.8499	445.7304	685.6162
r	0.019639	0.002807	0.01372	0.02556

Table II.
Summary of estimate of NHPP model parameters for DS1

error curve is usually within ± 5 percent. Altogether, from Figures 1-4 and Tables I-III, we can see that the proposed model has better performance and predicts the future behavior well.

In addition, substituting the estimated parameters β and θ in $t_{\max}$, the TEF reaches the maximum at time $t = 16.4626$ weeks which corresponds to $w(t) = 2.68742$ CPU hours and $W(t) = 41.1325$ CPU hours. Besides, the expected number of errors removed up to this time $t_{\max}$ is 313.4752. When t goes to infinity, the expected numbers of errors removed is 548.6367.

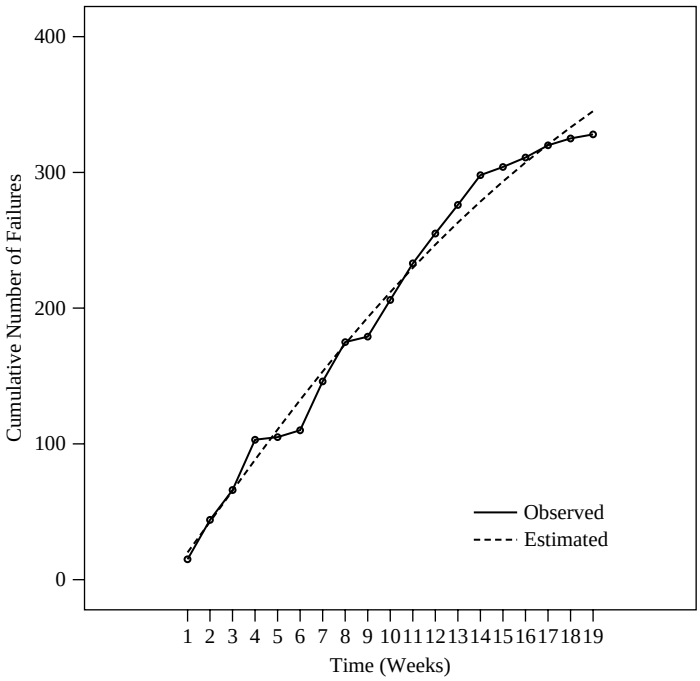

Figure 3.
Observed/estimated cumulative number of failures for DS1

Model	a	r	AE (percent)	MSE
Proposed model (equation (6))	565.673	0.01964	69.15	123.67
Yamada exponential model (equation (5) with exponential curve)	828.252	0.0117836	131.35	140.66
Yamada Rayleigh model (equation (5) with Rayleigh curve)	459.08	0.0273367	28.23	268.42
Yamada Weibull model (equation (5) with Weibull curve)	565.35	0.0196597	57.91	122.09
Huang logistic model	394.08	0.04272	10.06	118.59
Ohba exponential model	455.37	0.0267368	27.09	206.93
Inflection S-shaped model	389.1	0.0935493	8.69	133.53
Delayed S-shaped model	374.05	0.197651	4.48	168.67
G-O model	760.0	0.0322688	112.29	139.815
Delayed S-shaped model with Rayleigh	333.14	0.1004	6.93	798.49

Table III.
Comparison results of different SRGMs for DS1

DS2. similarly, this testing effort data are applied to estimate the parameters α, β and θ of the Burr type X TEF described in equation (1) by using the method of weighted least squares. The estimated values of parameters are:

$$\hat{\alpha} = 119.8385, \quad \hat{\beta} = 0.0030, \quad \text{and} \quad \hat{\theta} = 0.9338. \tag{16}$$

Figures 5 and 6 show the fitting of the estimated TEF by using above estimates. The fitted curves and the actual software data are shown by dotted and solid lines, respectively.

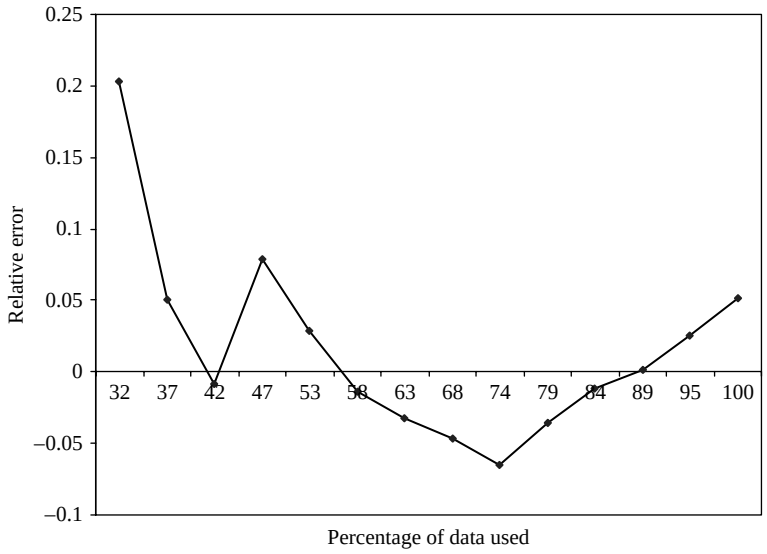

Figure 4.
Relative error curve for DS1

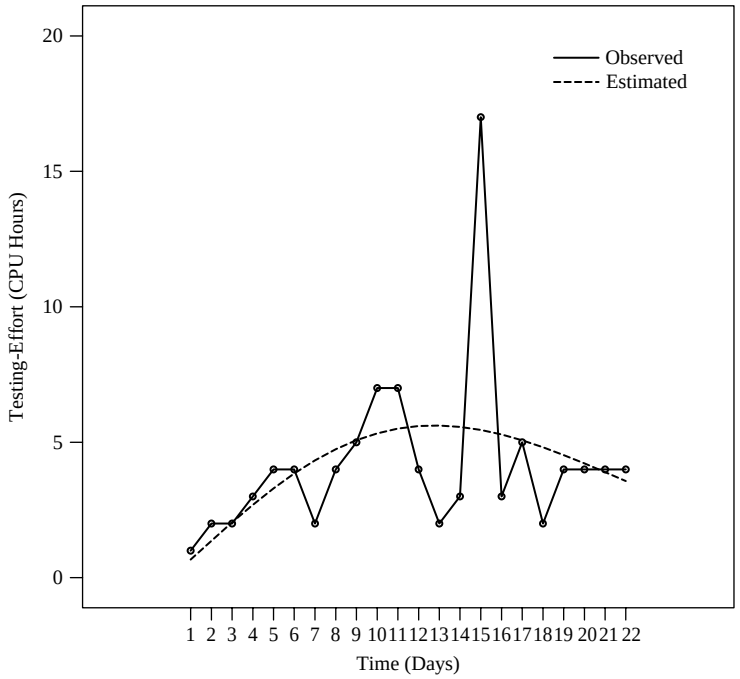

Figure 5.
Observed/estimated current testing-effort for DS2

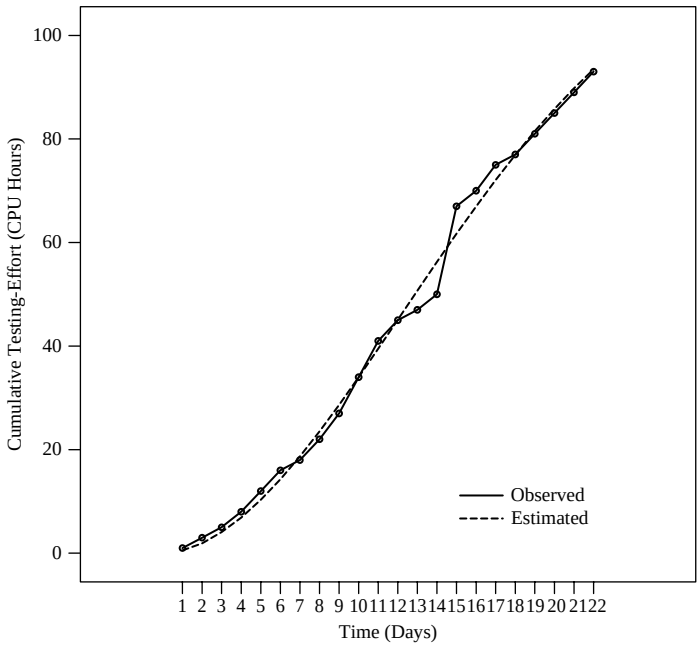

Figure 6.
Observed/estimated cumulative testing-effort for DS2

The other parameters a, r in equation (6) can be solved numerically using MLE method for these failure data. The estimators are:

$$\hat{a} = 94.6146, \quad \hat{r} = 0.25405.$$

The R^2 value for proposed Burr type X TEF is 0.9943. Therefore, we can say that the proposed curve is suitable for modeling the software reliability. The computed $F(= 9.4456)$ is greater than $F_{0.05}(2, 19)$ and $F_{0.01}(2, 19)$, which concludes that the fitted TEF curve is highly significant for this data set. Also, we computed the bias, the variation, the RMS-PE and the PE and these are shown in Table IV. On average, Table IV shows that the Burr type X TEF yields a better fit for DS2.

Table V shows the estimated values of parameters with their standard errors and 95 percent confidence limits for the proposed model. The fitted curve of the estimated mean value function with the actual software data is shown in Figure 7.

Table VI lists the comparisons of proposed model with different SRGMs which reveal that the proposed model has better performance. Kolmogorov Smirnov goodness-of-fit test shows that the proposed SRGM fits pretty well at the 5 percent level of significance. Finally, we compute the relative error in prediction of proposed model for this data set. Figure 8 shows the relative error plotted against the percentage of data used (that is, t_e/t_q). We observed that relative error approaches zero as t_e approaches t_q and the error curve is usually within ± 5 percent. Therefore, from Figures 5-8 and Tables IV-VI discussed, it can be concluded that the proposed model gets reasonable prediction in estimating the number of software errors and fits the observed data better than the others.

It has also been seen that, substituting the estimated parameters β and θ in t_{max}, the TEF reaches the maximum at time $t = 5.0024$ debug days which corresponds to $w(t) = 5.9426$ CPU hours and $W(t) = 27.6378$ CPU hours. The number of errors removed up to this time t_{max} is 94.5301 and when t goes to infinity, the numbers of errors removed is 94.6146.

DS3. Parameters α, β and θ of the Burr type X TEF for this data set can be obtained by using the method of WLSE. The estimated values are:

$$\hat{\alpha} = 33.8474, \quad \hat{\beta} = 0.00735, \quad \text{and} \quad \hat{\theta} = 6.8221. \tag{17}$$

Figures 9 and 10 show the fitting of the estimated TEF by using these estimates. The fitted curves are shown as a dotted line and the solid line is for actual software

TEF	Bias	Variation	RMS-PE	$PE_{end\ of\ testing}$
Burr type X TEF	0.155	2.35	2.35	− 0.45
Yamada Rayleigh	0.324	2.38	2.40	0.15
Yamada Weibull	0.182	2.37	2.37	− 0.44
Huang logistic	− 0.122	2.28	2.28	0.19

Table IV. Comparison results of different TEF for DS2

Parameter	Estimate	Standard error	95 percent confidence limits Lower	Upper
a	94.6146	2.5422	89.3117	99.9176
r	0.025405	0.00157	0.02212	0.02868

Table V. Summary of estimate of NHPP model parameters for DS2

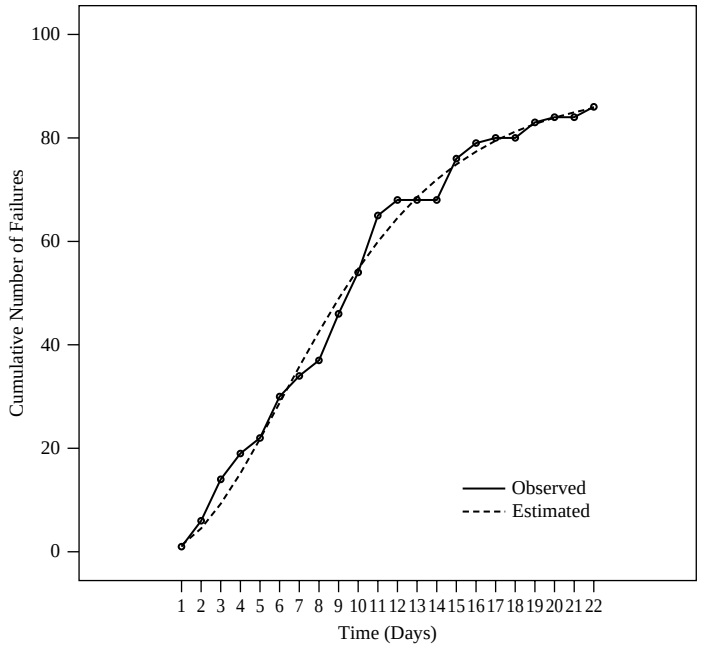

Figure 7. Observed/estimated cumulative number of failures for DS2

Model	a	r	MSE
Proposed model (equation (6))	94.6146	0.025405	6.549
Generalized exponential model (equation (5) with generalized exponential curve)	94.88	0.025	7.557
Yamada Rayleigh model (equation (5) with Rayleigh curve)	86.1616	0.0359624	3.91643
Yamada Weibull model (equation (5) with Weibull curve)	87.0318	0.0345417	7.772
Delayed S-shaped model	88.6533	0.228148	6.31268
Huang logistic model	88.8931	0.0390591	25.2279
G-O model	137.072	0.0515445	25.33
HGDM	88.30	–	33.6812

Table VI.
Comparison results of different SRGMs for DS2

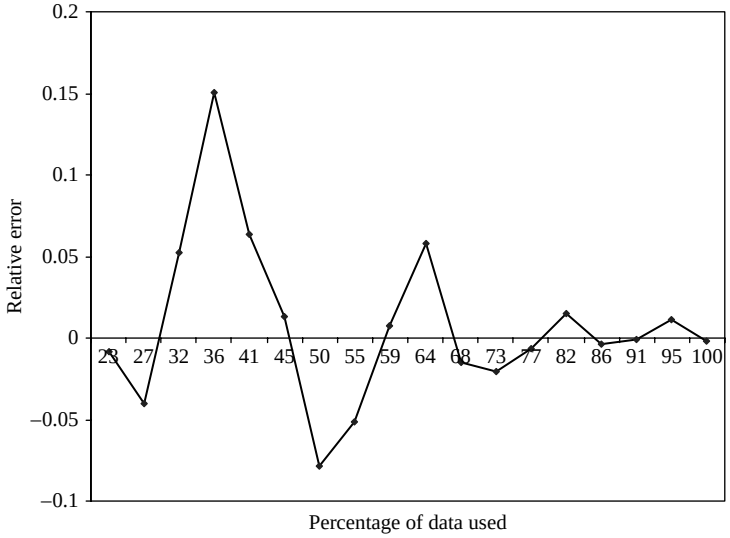

Figure 8.
Relative error curve for DS2

data in the graphs. Using the estimated parameters α, β and θ, the other parameters a, r in equation (6) can be solved numerically by MLE method. The estimates are:

$$\hat{a} = 134.1072, \quad \hat{r} = 0.15197.$$

The R^2 value for proposed Burr type X TEF curve is 0.9963 and calculated F-value is 8.9667, which is greater than $F_{0.05}(2, 18)$ and $F_{0.01}(2, 18)$. It can therefore be observed that the proposed model is suitable for modeling the software reliability and the fitted TEF curve is highly significant for this data set. In addition, the computed bias, the variation, the RMS-PE and the PE are shown in Table VII and reveal that, on average, Burr type X TEF yields a better fit.

Table VIII summarizes the experimental results of estimated parameters with their standard errors and 95 percent confidence limits of parameters for the proposed model.

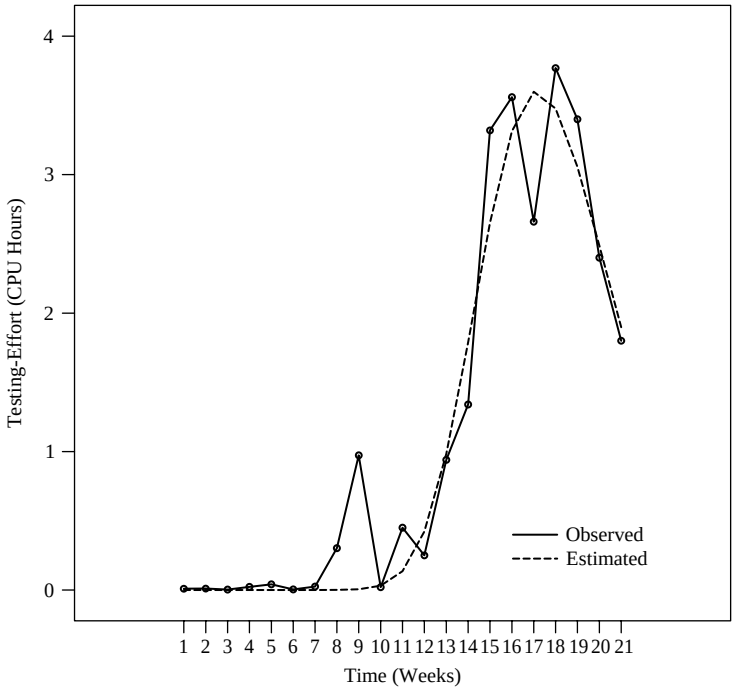

Figure 9.
Observed/estimated current testing-effort for DS3

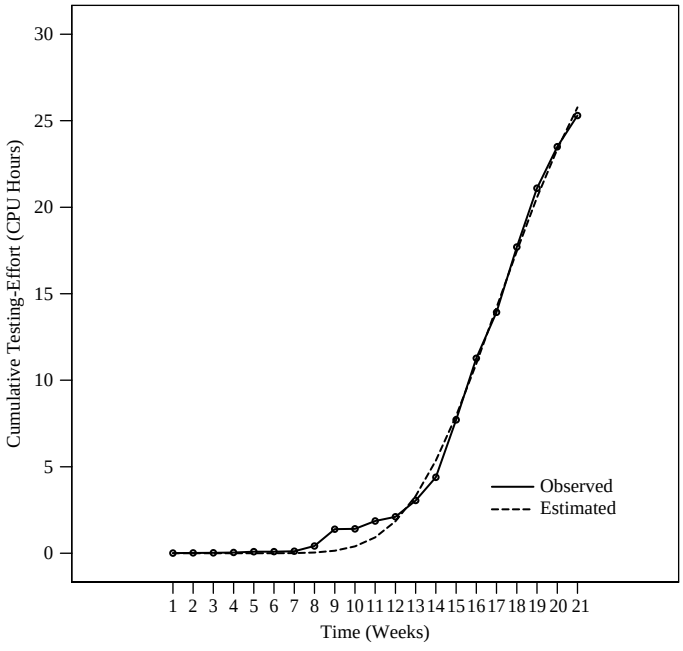

Figure 10.
Observed/estimated cumulative testing-effort for DS3

A fitted curve of the estimated mean value function with the actual software data is shown in Figure 11.

Table IX compares the performance of various SRGM for this data set. The Kolmogorov Smirnov goodness-of-fit test shows that the proposed SRGM fits pretty well at the 5 percent level of significance. Similarly, we compute the relative error in prediction for proposed model for this data set. Figure 12 shows the relative error plotted against the percentage of data used (that is, t_e/t_q). It is noted that the relative error of the proposed model approaches zero as t_e approaches t_q. Finally, Figures 9-12 and

TEF	Bias	Variation	RMS-PE	$PE_{end\ of\ testing}$
Burr type X TEF	0.155	0.50	0.47	− 0.47
Yamada exponential	− 16.53	6.35	17.71	− 13.29
Yamada Rayleigh	− 1.149	3.46	3.64	6.03
Huang logistic	0.055	0.35	0.35	− 0.10

Table VII.
Comparison results of different TEF for DS3

Parameter	Estimate	Standard error	95 percent confidence limits Lower	Upper
a	134.1072	7.884442	117.6049	150.6095
r	0.15197	0.027549	0.09431	0.20963

Table VIII.
Summary of estimate of NHPP model parameters for DS3

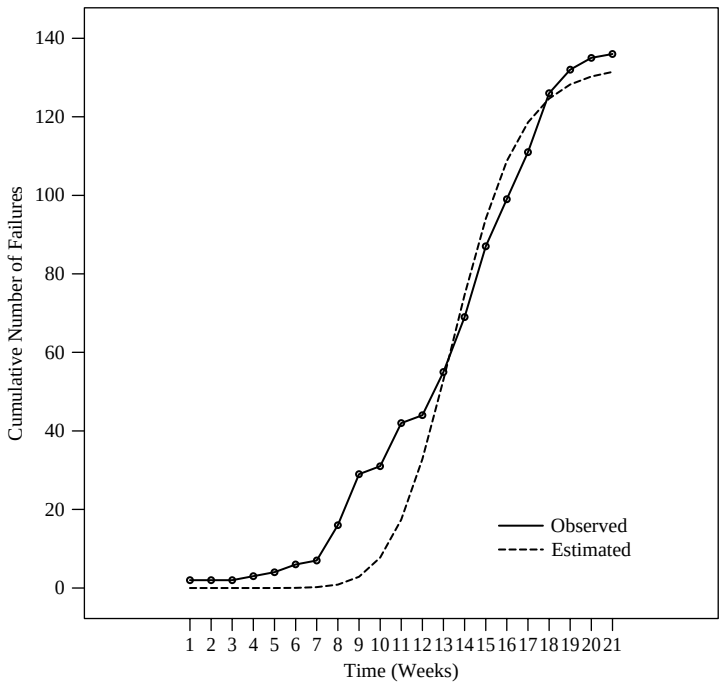

Figure 11.
Observed/estimated cumulative number of failures for DS3

Tables VII-IX reveal that the proposed model has better performance than the other models. This model fits the observed data better, and predicts the future behavior well.

Substituting the estimated parameters β and θ in equation of t_{max}, the TEF reaches the maximum at time $t = 14.82997$ debug days which corresponds to $w(t) = 2.75476$ CPU hours and $W(t) = 7.4745$ CPU hours. The number of errors removed up to this time t_{max} is 91.041. When t goes to infinity, the numbers of errors removed is 133.3246.

5. Optimal release-time determination

When the software testing is completed, software product is ready to release to users. If the length of software testing is long, it can remove many software errors in the system and its reliability increases. However, it leads to increase the testing cost and to delay software delivery. In contrast, if the length of software testing is short, a software system with low reliability is delivered and it includes many software errors which have not been removed in the testing phase. So, it is important that we have to find the solution for the optimal length of the software testing that is called optimal release time and the

Model	A	r	AE (percent)	MSE
Proposed model (equation (6))	134.1072	0.15197	28.67	123.678
Yamada Rayleigh model (equation (5) with Rayleigh curve)	866.94	0.00962	25.11	89.2409
Huang logistic model	138.026	0.145098	26.58	62.41
G-O model	142.32	0.1246	24.29	2,438.3
Inflection S-shaped model	159.11	0.0765	15.36	118.3
Delayed S-shaped model	237.196	0.096345	26.16	245.246
Ohba exponential model	137.2	0.156	27.12	3,019.66

Table IX.
Comparison results of different SRGMs for DS3

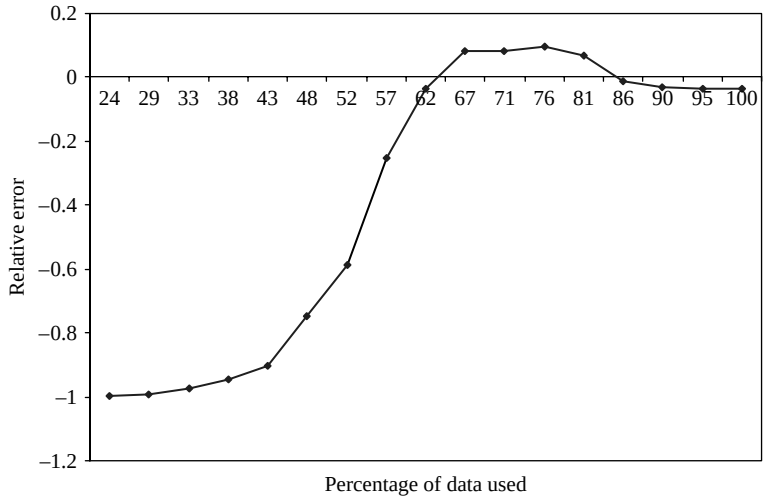

Figure 12.
Relative error curve for DS3

decision process is called an optimal software release problem (Xie, 1991; Yamada and Osaki, 1985b; Okumoto and Goel, 1980; Kapur and Garg, 1990; Kapur *et al.*, 1999).

5.1 Release time based on reliability criteria

The software-release time problem is associated with the reliability of a software system. If we know that the software reliability of this computer system has reached an acceptable reliability level, then we can determine the right time to release the software (Pham, 2000). The conditional reliability function is given in equation (11).

Differentiate equation (11) with respect to t, we observe that $(\partial R/\partial t) \geq 0$. Hence, $R(\Delta t|t)$ is a monotonic increasing function of t. Taking the logarithm on both sides of equation (11), we obtain:

$$\log R = -[m(t + \Delta t) - m(t)]. \tag{18}$$

We can easily determine the testing time needed to reach a desired R by solving equations (18) and (6). It is noted that $R(t)$ is increasing in t.

5.2 Application examples

In order to calculate the testing time that needed to reach a desired reliability, we consider the two actual DS (DS1 and DS3) described in the previous section in the following numerical examples.

DS1. In first data set, it is known that $\hat{\alpha} = 178.35202$, $\hat{\beta} = 0.000277$, $\hat{\theta} = 0.5585$, $\hat{a} = 565.6733$ and $\hat{r} = 0.019639$. Suppose the software system desires that the testing would be continued till the operational reliability is equal to 0.80 (at $\Delta t = 0.1$), from equations (18) and (6), we get $t = 47.43$ weeks. If the desired reliability is 0.85, then $t = 53.02$ weeks. If the desired reliability is 0.95, then $t = 74.42$ weeks and if the desired reliability is 0.98, then $t = 92.35$ weeks.

DS3. From the previous estimated parameters: $\hat{\alpha} = 33.8474$, $\hat{\beta} = 0.00735$, $\hat{\theta} = 6.8221$, $\hat{a} = 134.1072$ and $\hat{r} = 0.15197$, suppose the software system desires that the testing would be continued till the operational reliability is equal to 0.90 (at $\Delta t = 0.1$), from equations (18) and (6), we get testing time = 11.81 weeks. Similarly, the desired reliability is 0.95 (0.99), then $t = 13.72$ (18.56) weeks.

5.3 Release time based on cost-reliability criteria

We discuss the cost model and release time based on the cost-reliability criteria in this section. Using the total software cost evaluated by cost criterion, the cost of testing-effort expenditures during software testing and development phase, and the cost of correcting errors before and after release are given by Yamada *et al.* (1984, 1993), Yamada and Osaki (1985b), Okumoto and Goel (1980), Kapur *et al.* (1999), Kapur and Garg (1989, 1990), Huang *et al.* (1997, 1999) and Huang and Kuo (2002):

$$C(T) = C_1 m(T) + C_2[m(T_{\mathrm{LC}}) - m(T)] + C_3 \int_0^T w(x)\mathrm{d}x, \tag{19}$$

where C_1 is the cost of correcting an error during testing, C_2 is the cost of correcting an error during operation, $C_2 > C_1$, C_3 is the cost of testing per unit testing-effort expenditures and T_{LC} is the software life-cycle length.

From reliability criteria, we can obtain the required testing time needed to reach the reliability objective R_0. Our aim is to determine the optimal software release time that

minimizes the total software cost to achieve the desired software reliability. Therefore, the optimal software release problem for the proposed software reliability can be formulated as follows:

$$\begin{cases} \text{Minimize } C(T) \\ \text{Subject to } R(t+\Delta t|t) \geq R_0 \\ \text{for } C_2 > C_1 > 0, C_3 > 0, \\ \Delta t > 0, 0 < R_0 < 1. \end{cases} \tag{20}$$

The procedures to derive the determination of optimal release time for this problem are evolved step by step and are shown hereafter.

By differentiating equation (19) with respect to T and equating to zero, yields:

$$\frac{\mathrm{d}C(T)}{\mathrm{d}T} = C_1 \frac{\mathrm{d}m(T)}{\mathrm{d}T} - C_2 \frac{\mathrm{d}m(T)}{\mathrm{d}T} + C_3 w(T) = 0,$$

$$\frac{\lambda(T)}{w(T)} = \frac{C_3}{C_2 - C_1} = a \cdot r \cdot e^{-rW(T)} = r(a - m(T)). \tag{21}$$

When $T = 0$, then $m(0) = 0$ and $(\lambda(T)/w(T)) = ar$. When $T \to \infty$, then $m(\infty) = a(1 - e^{-r\alpha})$ and $(\lambda(T)/w(T)) = a \cdot r \cdot e^{-r\alpha}$. Therefore, $\lambda(T)/w(T)$ is monotonically decreasing in T. To analyze for the minimum value of $C(T)$, equation (21) is used to explore two cases of $\lambda(T)/w(T)$ at $T = 0$:

- *Case 1*:

$$\text{if } \frac{\lambda(0)}{w(0)} = a \cdot r \leq \frac{C_3}{C_2 - C_1},$$

$$\text{then } \frac{\lambda(T)}{w(T)} \leq \frac{C_3}{C_2 - C_1} \quad \text{for } 0 < T < T_{\mathrm{LC}}.$$

  It can be obtained that $(\mathrm{d}C(T)/\mathrm{d}T) > 0$ for $0 < T < T_{\mathrm{LC}}$ and the minimum of $C(T)$ can be found at $T = 0$.

- *Case 2*:

$$\text{If } \frac{\lambda(0)}{w(0)} = a \cdot r > \frac{C_3}{C_2 - C_1} > \frac{\lambda(T)}{w(T)} = a \cdot r \cdot e^{-r\alpha},$$

  there can be found a finite and unique real number:

$$T_0 = \left\{ -\frac{1}{\beta} \log \left[1 - \left(\frac{1}{r\alpha} \log \left[\frac{a \cdot r \cdot (C_2 - C_1)}{C_3} \right] \right)^{1/\theta} \right] \right\}^{1/2}$$

  satisfying equation (21):

$$\frac{dC(T)}{dT} < 0 \quad \text{for } 0 < T < T_0$$

and:

$$\frac{dC(T)}{dT} > 0 \quad \text{for } T_0 < T < T_{LC}.$$

It also can be shown that $(d^2C(T)/dT^2) > 0$ and hence $C(T)$ is a convex function. Thus, minimum of $C(T)$ is at $T = T_0$.

Furthermore, to commit the provisions of the optimal software release time for the proposed software reliability as depicted above, a finite and unique real number T_1 is determined such that $R(t + \Delta t|t) = R_0$, where $0 < R_0 < 1$.

Therefore, summarizing the above analysis and combining cost and reliability requirements, we have the following theorem.

Theorem 1. Assume $C_2 > C_1 > 0$, $C_3 > 0$, $\Delta t > 0$, and $0 < R_0 < 1$. Let T^* be the optimal software release time:

- $$\text{If } \frac{\lambda(0)}{w(0)} > \frac{C_3}{C_2 - C_1} \text{ and } \frac{\lambda(T)}{w(T)} = a \cdot r \cdot e^{-r\alpha} \leq \frac{C_3}{C_2 - C_1}, \text{ then}$$

$$T^* = \begin{cases} \max(T_0, T_1) & \text{for } R(\Delta t|0) < R_0 < 1 \\ T_0 & \text{for } 0 < R_0 < R(\Delta t|0). \end{cases}$$

- $$\text{If} \frac{\lambda(0)}{w(0)} \geq \frac{C_3}{C_2 - C_1}, \text{ then } T^* \geq \begin{cases} T_1 & \text{for } R(\Delta t|0) < R_0 < 1 \\ 0 & \text{for } 0 < R_0 < R(\Delta t|0) \end{cases}$$

- $$\text{If} \frac{\lambda(0)}{w(0)} \leq \frac{C_3}{C_2 - C_1}, \text{ then } T^* = \begin{cases} T_1 & \text{for } R(\Delta t|0) < R_0 < 1 \\ 0 & \text{for } 0 < R_0 < R(\Delta t|0) \end{cases}$$

5.4 Application example

We consider the DS1 to illustrate the determination optimal software release time. From the previously estimated parameters, we know that $\hat{\alpha} = 178.35202$, $\hat{\beta} = 0.000277$, $\hat{\theta} = 0.5585$, $\hat{a} = 565.6733$ and $\hat{r} = 0.019639$. To determine the optimal software release time, we assume the values of $C_1 = 1$, $C_2 = 50$, $C_3 = 100$, $T_{LC} = 100$, $R_0 = 0.95$, and $\Delta t = 0.1$ for the analysis. Then we get the optimal release time T_0 estimated as 33.89 based on minimizing $C(T)$ of equation (19), and T_1 is estimated as 74.42 based on satisfying the reliability criterion of $R(t + \Delta t|t) = R_0$. These values sustain the relationships of:

$$\frac{\lambda(0)}{w(0)} > \frac{C_3}{C_2 - C_1}, \frac{\lambda(T)}{w(T)} = a \cdot r \cdot e^{-r\alpha} < \frac{C_3}{C_2 - C_1} \text{ and } R(\Delta t|0) < R_0,$$

with which one could imply 1 in Theorem 1 to obtain the optimal software release time T^* as max (33.89, 74.42) = 74.42 weeks and the corresponding software cost $C(T^*)$ is 16,137.67.

6. Conclusion

In this paper, we have proposed a SRGM based on NHPP, which incorporates Burr Type X TEF. Based on different criteria, the performance of the proposed SRGM incorporating Burr Type X TEF is compared with other traditional SRGM. The results obtained show better fit and wider applicability of the proposed model to different types of actual software failure DS. We conclude that the incorporated TEF is a flexible and can be used to describe the actual expenditure patterns more faithfully during software development. We also conclude that the proposed SRGM has better performance as compare to the other SRGM and gives a reasonable predictive capability for the actual software failure data. Further, we have also discussed the optimal release-time determination based on cost and reliability criteria within our framework.

References

Ahmad, N., Bokhari, M.U., Quadri, S.M.K. and Khan, M.G.M. (2008), "The exponentiated Weibull software reliability growth model with various testing-efforts and optimal release policy: a performance analysis", *International Journal of Quality & Reliability Management*, Vol. 25 No. 2, pp. 211-35.

Bokhari, M.U. and Ahmad, N. (2006), "Analysis of a software reliability growth models: the case of log-logistic test-effort function", *Proceedings of the 17th IASTED International Conference on Modelling and Simulation (MS'2006), Montreal, Canada*, pp. 540-5.

Bokhari, M.U. and Ahmad, N. (2007), "Software reliability growth modeling for exponentiated Weibull functions with actual software failures data", *Advances in Computer Science and Engineering: Reports and Monographs*, Vol. 2, World Scientific Publications, Singapore.

Burr, I.W. (1942), "Cumulative frequency distribution", *Annals of Mathematical Statistics*, Vol. 13, pp. 215-31.

Goel, A.L. and Okumoto, K. (1979), "Time dependent error-detection rate model for software reliability and other performance measures", *IEEE Transactions on Reliability*, Vol. R-28 No. 3, pp. 206-11.

Huang, C.Y. (2005), "Performance analysis of software reliability growth models with testing-effort and change-point", *Journal of Systems and Software*, Vol. 76, pp. 181-94.

Huang, C.Y. and Kuo, S.Y. (2002), "Analysis of incorporating logistic testing-effort function into software reliability modeling", *IEEE Transactions on Reliability*, Vol. 51 No. 3, pp. 261-70.

Huang, C.Y., Kuo, S.Y. and Chen, I.Y. (1997), "Analysis of software reliability growth model with logistic testing-effort function", *Proceedings of 8th International Symposium on Software Reliability Engineering (ISSRE'1997), Albuquerque, NM, USA*, pp. 378-88.

Huang, C.Y., Kuo, S.Y. and Lyu, M.R. (1999), "Optimal software release policy based on cost, reliability and testing efficiency", *Proceedings of the 23rd IEEE Annual International Computer Software and Applications Conference (COMPSAC'99), Phoenix, AZ, USA*, pp. 468-73.

Huang, C.Y., Kuo, S.Y. and Lyu, M.R. (2000), "Effort-index based software reliability growth models and performance assessment", *Proceedings of the 24th IEEE Annual International*

Computer Software and Applications Conference (COMPSAC'2000), Taipei, Taiwan, pp. 454-9.

Huang, C.Y., Kuo, S.Y. and Lyu, M.R. (2007), "An assessment of testing-effort dependent software reliability growth models", *IEEE Transactions on Reliability*, Vol. 56 No. 2, pp. 198-211.

Kapur, P.K. and Garg, R.B. (1989), "Cost reliability optimum release policies for a software system under penalty cost", *International Journal of System Science*, Vol. 20, pp. 2547-62.

Kapur, P.K. and Garg, R.B. (1990), "Cost reliability optimum release policies for a software system with testing effort", *Operations Research*, Vol. 27 No. 2, pp. 109-16.

Kapur, P.K. and Garg, R.B. (1996), "Modeling an imperfect debugging phenomenon in software reliability", *Microelectronics and Reliability*, Vol. 36, pp. 645-50.

Kapur, P.K. and Younes, S. (1994), "Modeling an imperfect debugging phenomenon with testing effort", *Proceedings of 5th International Symposium on Software Reliability Engineering (ISSRE'1994), Monterey, CA, USA*, pp. 178-83.

Kapur, P.K., Garg, R.B. and Kumar, S. (1999), *Contributions to Hardware and Software Reliability*, World Scientific Publications, Singapore.

Kapur, P.K., Goswami, D.N. and Gupta, A. (2004), "A software reliability growth model with testing effort dependent learning function for distributed systems", *International Journal of Reliability, Quality and Safety Engineering*, Vol. 11 No. 4, pp. 365-77.

Kumar, M., Ahmad, N. and Quadri, S.M.K. (2005), "Software reliability growth models and data analysis with a Pareto test-effort", *RAU Journal of Research*, Vol. 15 Nos 1/2, pp. 124-8.

Kuo, S.Y., Hung, C.Y. and Lyu, M.R. (2001), "Framework for modeling software reliability, using various testing-efforts and fault detection rates", *IEEE Transactions on Reliability*, Vol. 50 No. 3, pp. 310-20.

Lyu, M.R. (1996), *Handbook of Software Reliability Engineering*, McGraw-Hill, New York, NY.

Musa, J.D. (1999), *Software Reliability Engineering: More Reliable Software, Faster Development and Testing*, McGraw-Hill, New York, NY.

Musa, J.D., Iannino, A. and Okumoto, K. (1987), *Software Reliability: Measurement, Prediction and Application*, McGraw-Hill, New York, NY.

Nelson, W. (1982), *Applied Life Data Analysis*, Wiley, New York, NY.

Ohba, M. (1984), "Software reliability analysis model", *IBM Journal. Research Development*, Vol. 28 No. 4, pp. 428-43.

Okumoto, K. and Goel, A.L. (1980), "Optimum release time for software system based on reliability and cost criteria", *Journal of Systems and Software*, Vol. 1, pp. 315-18.

Pham, H. (2000), *Software Reliability*, Springer, New York, NY.

Pillai, K. and Nair, V.S.S. (1997), "A model for software development effort and cost estimation", *IEEE Transactions on Software Engineering*, Vol. 4 No. 8, pp. 343-61.

Putnam, L. (1978), "A general empirical solution to the macro software sizing and estimating problem", *IEEE Transactions on Software Engineering*, Vol. 4, pp. 485-97.

Quadri, S.M.K., Ahmad, N., Peer, M.A. and Kumar, M. (2006), "Nonhomogeneous Poisson process software reliability growth model with generalized exponential testing effort function", *RAU Journal of Research*, Vol. 16 Nos 1/2, pp. 159-63.

Surles, J.G. and Padgett, W.J. (2001), "Inference for reliability and stress-strength for a scaled Burr type X distribution", *Life Data Analysis*, Vol. 7, pp. 187-200.

Surles, J.G. and Padgett, W.J. (2005), "Some properties of a scaled Burr type X distribution", *Journal of Statistical Planning and Inference*, Vol. 128, pp. 271-80.

Tohma, Y., Jacoby, R., Murata, Y. and Yamamoto, M. (1989), "Hyper-geometric distribution model to estimate the number of residual software faults", *Proceedings of COMPSAC-89, Orlando, FL, USA*, pp. 610-17.

Xie, M. (1991), *Software Reliability Modeling*, World Scientific Publications, Singapore.

Yamada, S. and Ohtera, H. (1990), "Software reliability growth models for testing effort control", *European Journal of Operational Research*, Vol. 46 No. 3, pp. 343-9.

Yamada, S. and Osaki, S. (1985a), "Software reliability growth modeling: models and applications", *IEEE Transactions on Software Engineering*, Vol. SE-11 No. 12, pp. 1431-7.

Yamada, S. and Osaki, S. (1985b), "Cost-reliability optimal release policies for software systems", *IEEE Transactions on Reliability*, Vol. R-34 No. 5, pp. 422-4.

Yamada, S., Hishitani, J. and Osaki, S. (1991), "Test-effort dependent software reliability measurement", *International Journal of Systems Science*, Vol. 22 No. 1, pp. 73-83.

Yamada, S., Hishitani, J. and Osaki, S. (1993), "Software reliability growth model with Weibull testing-effort: a model and application", *IEEE Transactions on Reliability*, Vol. R-42, pp. 100-5.

Yamada, S., Narihisa, H. and Osaki, S. (1984), "Optimum release policies for a software system with a scheduled software delivery time", *International Journal of System Science*, Vol. 15, pp. 905-14.

Yamada, S., Ohtera, H. and Narihisa, H. (1986), "Software reliability growth model with testing-effort", *IEEE Transactions on Reliability*, Vol. R-35 No. 1, pp. 19-23.

Yamada, S., Ohtera, H. and Narihisa, H. (1987), "A testing-effort dependent software reliability model and its application", *Microelectronics and Reliability*, Vol. 27 No. 3, pp. 507-22.

Appendix

Weighted least squares method

The weighted least squares estimators $\hat{\alpha}$, $\hat{\beta}$ and $\hat{\theta}$ in equation (1) can be obtained by minimizing:

$$S'(\alpha, \beta, \theta) = \sum_{k=1}^{n} l_k [W_k - W(t_k)]^2.$$

or, equivalently:

$$S(\alpha, \beta, \theta) = \sum_{k=1}^{n} l_k \left[\ln W_k - \ln \alpha - \theta \ln(1 - e^{-\beta t(2/k)})\right]^2. \tag{A1}$$

where l_k are weight assigned to data points according to their proper amount of influence over the parameter estimates.

Differentiating $S(\alpha, \beta, \theta)$ with respect to α, β and θ, equating the partial derivatives to zero, we have the following non-linear normal equations:

$$\frac{\partial S}{\partial \alpha} = 2\sum_{k=1}^{n} l_k \left\{\ln W_k - \ln \alpha - \theta \ln\left(1 - e^{-\beta \cdot t(2/k)}\right)\right\} \cdot \left(-\frac{1}{\alpha}\right) = 0.$$

Thus, the WLSE of α is given by:

$$\hat{\alpha} = e^{\frac{\left[\sum_{k=1}^{n} l_k \ln W_k - \theta \sum_{k=1}^{n} l_k \ln(1 - e^{-\beta \cdot t(2/k)})\right]}{\sum_{k=1}^{n} l_k}}. \tag{A2}$$

The WLSE of θ can be obtained as:

$$\frac{\partial S}{\partial \theta} = 2\sum_{k=1}^{n} l_k \{\ln W_k - \ln \alpha - \theta \ln(1 - e^{-\beta \cdot t(2/k)})\} \cdot \ln(1 - e^{-\beta \cdot t(2/k)}) = 0,$$

and hence:

$$\hat{\theta} = \frac{\sum_{k=1}^{n} l_k \ln W_k \ln(1 - e^{-\beta \cdot t(2/k)}) - \ln \alpha \cdot \sum_{k=1}^{n} l_k \ln(1 - e^{-\beta \cdot t(2/k)})}{\sum_{k=1}^{n} l_k (\ln(1 - e^{-\beta \cdot t(2/k)}))^2}. \tag{A3}$$

Finally, the WLSE of β can be obtained by substituting the above estimators into:

$$\frac{\partial S}{\partial \beta} = 2\sum_{k=1}^{n} l_k \left\{\ln W_k - \ln \alpha - \theta \ln(1 - e^{-\beta \cdot t(2/k)})\right\} \cdot \left(-\frac{\theta \cdot t_k^2 \cdot e^{-\beta \cdot t(2/k)}}{1 - e^{-\beta \cdot t(2/k)}}\right) = 0. \tag{A4}$$

Maximum likelihood method

The likelihood function for the unknown parameters a and r in the NHPP model with $m(t)$ in equation (6) is given by:

$$\begin{aligned} L'(a, r) &\equiv P\{N(t_i) = m_i, \quad i = 1, 2, \ldots, n\} \\ &= \prod_{k=1}^{n} \frac{[m(t_k) - m(t_{k-1})]^{(m_k - m_{k-1})} \cdot e^{-[m(t_k) - m(t_{k-1})]}}{(m_k - m_{k-1})!}, \end{aligned} \tag{A5}$$

where $m_0 \equiv 0$ for $t_0 \equiv 0$.

Taking logarithm both sides in equation (A5), we have:

$$L = \ln L' = \sum_{k=1}^{n} (m_k - m_{k-1}) \ln[m(t_k) - m(t_{k-1})]$$

$$- \sum_{k=1}^{n} [m(t_k) - m(t_{k-1})] - \sum_{k=1}^{n} \ln[(m_k - m_{k-1})!].$$

From equation (5), we know that:

$$m(t_k) - m(t_{k-1}) = a\left[e^{-rW(t_{k-1})} - e^{-rW(t_k)}\right]$$

and then we have:

$$\sum_{k=1}^{n} [m(t_k) - m(t_{k-1})] = m(t_n) = a[1 - e^{-rW(t_n)}]$$

Thus:

$$\begin{aligned} L = &\sum_{k=1}^{n} (m_k - m_{k-1}) \ln a + \sum_{k=1}^{n} (m_k - m_{k-1}) \\ &\times \ln[e^{-rW(t_{k-1})} - e^{-rW(t_k)}] - a[1 - e^{-rW(t_n)}] - \sum_{k=1}^{n} \ln[(m_k - m_{k-1})!]. \end{aligned} \tag{A6}$$

The MLE of reliability growth parameters a and r can be obtained by solving the following equations, that is:

$$\frac{\partial L}{\partial a} = \frac{\sum_{k=1}^{n}(m_k - m_{k-1})}{a} - 1 + e^{-rW(t_n)} = 0,$$

$$\frac{\partial L}{\partial r} = \sum_{k=1}^{n} \frac{(m_k - m_{k-1}) \cdot \left(-W(t_{k-1})\ e^{-rW(t_{k-1})} + W(t_k)\ e^{-rW(t_k)}\right)}{e^{-rW(t_{k-1})} - e^{-rW(t_k)}} \quad - a\ W(t_n)\ e^{-rW(t_n)} = 0.$$

After some algebraic simplification, we get:

$$\hat{a} = \frac{m_n}{1 - e^{-rW(t_n)}} = \frac{m_n}{1 - \phi_n}, \tag{A7}$$

and:

$$aW(t_n).\phi_n = \sum_{k=1}^{n} \frac{(m_k - m_{k-1})[W(t_k)\phi_k - W(t_{k-1})\phi_{k-1}]}{\phi_{k-1} - \phi_k}, \tag{A8}$$

where $\phi_k = e^{-rW(t_k)}$, for $k = 1, 2, \ldots, n$.

Which can be solved by numerical methods to get the values of $\hat{a}$ and $\hat{r}$.

Finally, if the sample size n of (t_k, m_k) is sufficiently large, then the MLE $\hat{a}$ and $\hat{r}$ asymptotically follow a bivariate s-normal distribution (Nelson, 1982; Okumoto and Goel, 1980; Yamada *et al.*, 1993; Kapur *et al.*, 1999, 2004; Huang and Kuo, 2002). That is:

$$\begin{pmatrix} \tilde{\hat{a}} \\ \hat{r} \end{pmatrix} \sim \text{BVN}\left[\begin{pmatrix} \hat{a} \\ \hat{r} \end{pmatrix}, \quad \sum\right], \text{as } n \to \infty. \tag{A9}$$

The variance-covariance matrix $\sum$ in the asymptotic properties of equation (A9) is useful in qualifying the variability of the estimated parameters $\hat{a}$ and $\hat{r}$, and is the inverse of the Fisher information matrix $\mathbf{F}$, i.e. $\sum = \mathbf{F}^{-1}$ (Nelson, 1982; Yamada *et al.*, 1986, 1993; Yamada and Osaki, 1985a), given by the expectation of the negative of the second partial derivative of L as:

$$\mathbf{F} = \begin{bmatrix} E\left(-\frac{\partial^2 L}{\partial a^2}\right) & E\left(-\frac{\partial^2 L}{\partial a \partial r}\right) \\ E\left(-\frac{\partial^2 L}{\partial r \partial a}\right) & E\left(-\frac{\partial^2 L}{\partial r^2}\right) \end{bmatrix} = \begin{bmatrix} \frac{f_n}{a} & g_n \\ g_n & \frac{a\sum_{k=1}^{n}(g_k - g_{k-1})^2}{(f_k - f_{k-1})} \end{bmatrix}, \tag{A10}$$

where $g_k = W(t) \cdot e^{-rW(t_k)}$ and $f_k = 1 - e^{-rW(t_k)}$ for $k = 1, 2, \ldots, n$.

After substituting the value of a and r in equation (A10), one can calculate $\mathbf{F}^{-1}$. The estimated asymptotic variance-covariance matrix is:

$$\hat{\sum} = \mathbf{F}^{-1} = \begin{bmatrix} \text{Var}(\hat{a}) & \text{Cov}(\hat{a}, \hat{r}) \\ \text{Cov}(\hat{r}, \hat{a}) & \text{Var}(\hat{r}) \end{bmatrix}. \tag{A11}$$

The $(1 - \alpha)100$ percent confidence limits for a and r can be obtained as (Yamada and Osaki, 1985a):

$$\hat{a} - z_{\alpha/2}\sqrt{\mathrm{Var}(\hat{a})} \leq a \leq \hat{a} + z_{\alpha/2}\sqrt{\mathrm{Var}(\hat{a})}, \tag{A12}$$

and:

$$\hat{r} - z_{\alpha/2}\sqrt{\mathrm{Var}(\hat{r})} \leq r \leq \hat{r} + z_{\alpha/2}\sqrt{\mathrm{Var}(\hat{r})}, \tag{A13}$$

where $z_{\alpha/2}$ is the $(1 - \alpha/2)$ quartile of the standard normal distribution.

Corresponding author
N. Ahmad can be contacted at: ahmad_n@usp.ac.fj