Far East Journal of Theoretical Statistics
Volume 39, Number 1, 2012, Pages 67-77

Published by Pushpa Publishing House, Allahabad, INDIA

MODELING AND STATISTICAL INFERENCE ON GENERALIZED INVERSE EXPONENTIAL SOFTWARE RELIABILITY GROWTH MODEL

K. M. Manjunatha and K. Harishchandra

Department of Statistics
Bangalore University
Bangalore-560 056, India
e-mail: manjustat@gmail.com; harish.jbc@gmail.com

Abstract

Finite failure non-homogeneous Poisson process models proposed in the literature exhibit either constant, monotonic increasing or decreasing failure occurrence rates per fault and are inadequate to describe the failure process underlying both increasing/decreasing failure rates. In this article, we propose generalized inverse exponential software reliability growth model, which can capture both increasing/decreasing nature of failure occurrence rate per fault. The system parameters are estimated by means of the maximum likelihood estimators and the properties of the estimators are discussed. The experimental results of real data show that our proposed model performs better.

1. Introduction

In last three decades, we have seen formulation of several software reliability growth models (SRGMs) to predict the reliability of software

---

2010 Mathematics Subject Classification: 60K20.

Keywords and phrases: software reliability, failure detection process, hazard rate function, mean value function, maximum likelihood estimation.

Received February 29, 2012

systems. These models are concerned with forecasting future system operability from the failure data collected during the testing phase of a software product. A plethora of SRGMs have appeared in the literature. However, many existing finite failure category models describe the failure behaviour as of constant, increasing and decreasing; see Goel and Okumoto [3], Ohba [6], Yamada et al. [8], Goel [2] and Gokhale et al. [4]. For further information regarding this, one can refer to Lyu [5] and Pham [7]. But in most of the situations, these may not be true because in early stages of testing, the testers are new to software, so they need time to adjust, it implies less detection of failure caused by faults at beginning. As testing progresses, testers will get good exposure to software implies increase in detection of failures and it implies detection of faults increases in this period of time and when most of software faults are eliminated, then failure of software decreases as time increases. This phenomenon of increasing/decreasing failure behaviour is not included in the existing finite failure models. This idea leads us to a development of new model taking into account of both increasing/decreasing behaviours of the software failures by its hazard rate function.

In this paper, we describe finite failure non-homogeneous Poisson process (NHPP) class of SRGMs and offer a decomposition of the mean value function (MVF) of the finite failure NHPP models, which enables us to attribute the nature of failure intensity of the software to the hazard function and we propose a SRGM which describes the increasing/decreasing nature of the failure occurrence rate per fault. The unknown model parameters are estimated using maximum likelihood method.

2. Finite Failure NHPP Models

This is a class of time-domain SRGMs, which assume that software failures display the behaviour of a NHPP. The parameter of the stochastic process, $\lambda(t)$ which denotes the failure intensity of the software at time t, is time dependent. Let $N(t)$ denote the cumulative number of faults detected by time t, and $m(t)$ denote its expectation. Then $m(t) = E[N(t)]$, and the failure intensity $\lambda(t)$ is related as follows:

$$m(t) = \int_0^t \lambda(s)ds \quad \text{and} \quad \lambda(t) = \frac{\partial m(t)}{\partial t}, \tag{1}$$

$N(t)$ is known to have a NHPP probability mass function with MVF $m(t)$, that is,

$$P\{N(t) = n\} = \frac{[m(t)]^n}{n!} e^{-m(t)}, \quad n = 0, 1, ..., \infty. \tag{2}$$

Various time-domain models have appeared in the literature which describe the stochastic failure process by an NHPP. These models differ in their failure intensity function $\lambda(t)$, and hence $m(t)$. The NHPP models can be further classified into finite failure and infinite failure categories; see Lyu [5]. Finite failure NHPP models assume that the expected number of faults detected given infinite amount of testing time will be finite, whereas the infinite failure models assume that an infinite number of faults would be detected in infinite testing time. Let a denote the expected number of faults that would be detected in given infinite testing time in case of finite failure NHPP models. Then the mean value function of the finite failure NHPP models can also be written as

$$m(t) = aF(t), \tag{3}$$

where $F(t)$ is a distribution function.

From equation (3), the instantaneous failure intensity $\lambda(t)$ in case of the finite failure NHPP models is given by

$$\lambda(t) = aF'(t), \tag{4}$$

which can be re-written as

$$\lambda(t) = [a - m(t)]\frac{F'(t)}{1 - F(t)} = [a - m(t)]h(t), \tag{5}$$

where $h(t)$ is the failure occurrence rate per fault of the software, or the rate at which the individual faults manifest themselves as failures during testing. The quantity $[a - m(t)]$ denotes the expected number of faults remaining in the software at time t and since it is monotonically non-increasing function of time, the nature of the overall failure intensity $\lambda(t)$ is governed by the nature of failure occurrence rate per fault $h(t)$, from (5).

2.1. Existing models

The failure occurrence rate per fault $h(t)$ can be a constant, increasing, decreasing, or increasing/decreasing. Here we describe some of the existing finite failure NHPP models along with their hazard functions. The Goel-Okumoto [3] (GO) model has had a strong influence on software reliability modeling. Table 2.1 gives the expressions for $m(t)$, $\lambda(t)$ and $h(t)$ for the GO model. In this model, failure occurrence rate per fault is constant, i.e., $h(t)$ is time independent; however since the expected number of remaining faults decreases with time, the overall software failure intensity decreases with time. The software quality continues to improve as testing progresses. However, in most real-life testing scenarios, the software failure intensity increases initially and then decreases. Goel [2] proposed a generalized GO or Weibull model which captures the increasing/decreasing nature of the failure intensity and the nature of failure occurrence rate per fault is determined by the parameter γ, and is increasing for $\gamma < 1$ and decreasing for $\gamma > 1$. Refer to Table 2.1 for expressions for $m(t)$, $\lambda(t)$ and $h(t)$. *S*-shaped SRGM proposed by Yamada et al. [8] captures the software error removal phenomenon in which there is a time delay between the actual detection of the fault and its reporting. The testing process in this case can be seen as consisting of two phases: fault detection and fault isolation. The *S*-shaped model has an increasing failure occurrence rate per fault and its expressions of $m(t)$, $\lambda(t)$ and $h(t)$ are presented in Table 2.1.

Table 2.1. NHPP models $m(t)$, $\lambda(t)$ and $h(t)$

Coverage function	$m(t)$	$\lambda(t)$	$h(t)$
GO	$a(1-e^{-bt})$	abe^{-bt}	b
Weibull	$a(1-e^{-bt^{\gamma}})$	$ab\gamma t^{\gamma-1}e^{-bt^{\gamma}}$	$b\gamma t^{\gamma-1}$
***S*-shaped**	$a[1-(1+bt)e^{-bt}]$	ab^2te^{-bt}	$\frac{b^2t}{1+bt}$

We now present the graph of a data set which led us to the development of generalized inverse exponential SRGM. Data set is from the U. S. Navy Feet Computer Programming Center and is given by Goel and Okumoto [3]. The data set consists of 26 failures in 250 days. The hazard rates for the GO, generalized GO and *S*-shaped models are shown in Figure 2.1.

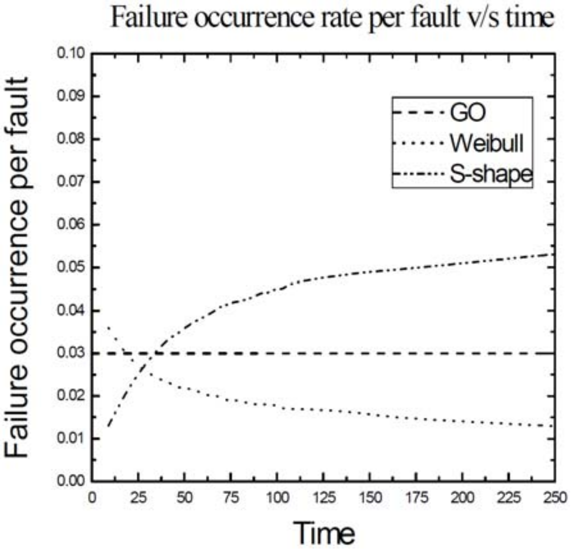

Figure 2.1. Hazard for existing NHPP models.

2.2. Proposed model

In this section, we develop a SRGM in which increasing/decreasing behaviour of failure occurrence rate per fault can be captured by nature of hazard function of generalized inverted exponential distribution given in

Abouammoh and Alshingiti [1]. The hazard rate function $h(t)$ of generalized inverse exponential SRGM is given as

$$h(t) = \frac{\alpha b}{t^2}\left(\frac{e^{-b/t}}{1 - e^{-b/t}}\right). \tag{6}$$

The corresponding MVF $m(t)$ and failure intensity function $\lambda(t)$ are

$$m(t) = a[1 - (1 - e^{-b/t})^{\alpha}], \tag{7}$$

$$\lambda(t) = \frac{a\alpha b}{t^2} e^{-b/t}(1 - e^{-b/t})^{\alpha - 1}. \tag{8}$$

Figure 2.2 shows the hazard of the generalized inverse exponential model. The parameters of the model are set up in such a way that there are 26 failures in 250 time units.

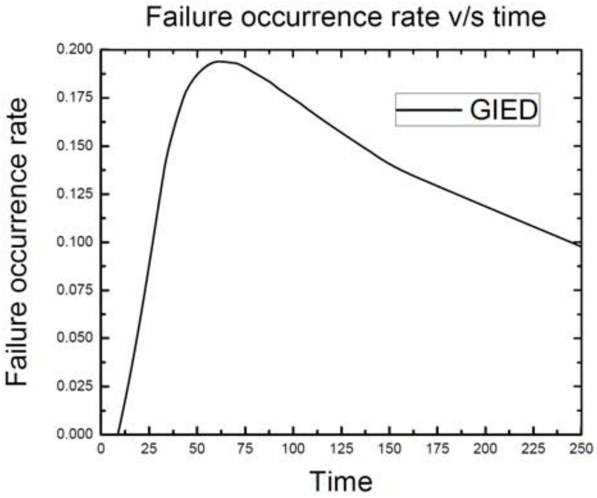

Figure 2.2. Hazard for generalized inverse exponential model.

2.3. Software reliability

Let S_i $(i = 1, 2, ...)$ be a random variable representing the *i*th software failure occurrence time. Then we have $X_i = S_i - S_{i-1}, i = 1, 2, ...; S_0 = 0,$ which is a random variable representing the time-interval between $(i-1)$th and *i*th software failure occurrences. The conditional probability that the *i*th software failure does not occur between $(t, t + x], (x \geq 0)$ on the condition that the $(i - 1)$th software failure has occurred at testing time *t*, is given by

$$R(x \mid t) = \Pr\{X_i > x \mid S_{i-1} = t\} = \exp[-\{m(x + t) - m(t)\}], \; t \geq 0, \; x \geq 0. \quad (9)$$

Substituting (7) into (9), we have the software reliability for finite failure generalized exponential model as

$$R(x \mid t) = \exp\{a[(1 - e^{-b/(t+x)})^{\alpha} - (1 - e^{-b/t})^{\alpha}]\}. \quad (10)$$

To predict future reliability, we have to estimate unknown parameters involved in above expression. In order to do so, we have carried maximum likelihood method in next section.

3. Estimation of Parameters

We estimate unknown system parameters of the finite failure NHPP model by using method of maximum likelihood. We obtain estimates by considering two types of data.

3.1. Interval domain data

Suppose that *n* data pairs $(t_i, y_i)(i = 1, 2, ..., n; 0 < t_1 < t_2 < \cdots < t_n)$ are observed during the testing phase where the observed number of software faults detected up to testing time t_i is y_i. The simultaneous probability mass function of $\{N(t_1) = y_1, N(t_2) = y_2, ..., N(t_n) = y_n\}$, i.e., the likelihood function for the interval domain data is

$$L = \prod_{i=1}^{n} \frac{\{m(t_i) - m(t_{i-1})\}^{(y_i - y_{i-1})}}{(y_i - y_{i-1})!} \exp[-\{m(t_i) - m(t_{i-1})\}], \quad (11)$$

where $t = 0$ and $y_0 = 0$. Taking the natural logarithm of (11) yields log-likelihood function, that is

$$\ln L = \sum_{i=1}^{n} (y_i - y_{i-1}) \ln[m(t_i) - m(y_{i-1})] - m(t_n). \quad (12)$$

Substituting mean value function $m(t)$ given in (7) into (12), we get log-likelihood function

$$\ln L = \sum_{i=1}^{n} (y_i - y_{i-1}) \ln[a(1 - e^{-b/t_i})^{\alpha} - a(1 - e^{-b/t_{i-1}})^{\alpha}]$$

$$- a(1 - (1 - e^{-b/t_n})^{\alpha}). \tag{13}$$

Taking partial derivative of expression (13) w.r.t. a, b and α and then equating them to zero, we get likelihood equations, which are

$$a = \frac{\sum_{i=1}^{n} (y_i - y_{i-1})}{[1 - (1 - e^{-b/t_n})^{\alpha}]}, \tag{14}$$

$$\sum_{i=1}^{n} (y_i - y_{i-1}) \left[\frac{\frac{\alpha}{t_{i-1}} e^{-b/t_{i-1}} (1 - e^{-b/t_{i-1}})^{\alpha-1} - \frac{\alpha}{t_i} e^{-b/t_i} (1 - e^{-b/t_i})^{\alpha-1}}{(1 - e^{-b/t_{i-1}})^{\alpha} - (1 - e^{-b/t_i})^{\alpha}} \right]$$

$$+ \frac{a\alpha}{t_n} e^{-b/t_n} (1 - e^{-b/t_n})^{\alpha-1} = 0, \tag{15}$$

$$\sum_{i=1}^{n} (y_i - y_{i-1}) \left[\frac{(1 - e^{-b/t_{i-1}})^{\alpha} \log[1 - e^{-b/t_{i-1}}] - (1 - e^{-b/t_i})^{\alpha} \log[1 - e^{-b/t_i}]}{(1 - e^{-b/t_{i-1}})^{\alpha} - (1 - e^{-b/t_i})^{\alpha}} \right]$$

$$+ a(1 - e^{-b/t_n})^{\alpha} \log[1 - e^{-b/t_n}] = 0. \tag{16}$$

We do not get closed form expressions for the maximum likelihood estimates (MLEs) of a, b and α. However, the MLEs can be obtained by iterative procedure. Let $\hat{a}, \hat{b}$ and $\hat{\alpha}$ be the MLEs of parameters a, b and α, respectively. We can then obtain MLEs of $m(t)$ and $\lambda(t)$ by replacing a, b and α by its MLEs $\hat{a}, \hat{b}$ and $\hat{\alpha}$ in expressions (7) and (8), respectively.

3.2. Time domain data

Suppose that the data set on *n* software failure-occurrence times s_i, i.e., realization of S_i $(i = 1, 2, ..., n; 0 \le s_1 \le s_2 \le \cdots \le s_n)$ is observed during the testing phase. The simultaneous probability density function, i.e., the likelihood function for the software failure-occurrence time data is

$$L = \prod_{i=1}^{n} \lambda(s_i) \exp\left[-\int_{s_{i-1}}^{s_i} \lambda(x)\,dx\right] = \exp[-m(s_n)] \prod_{i=1}^{n} \lambda(s_i), \tag{17}$$

where $s_0 = 0$. Taking the natural logarithm of (17) yields log-likelihood equation, that is

$$\ln L = \sum_{i=1}^{n} \ln \lambda(s_i) - m(s_n). \tag{18}$$

Substituting $m(t)$ and $\lambda(t)$ given in (7) and (8), we obtain logarithmic likelihood functions for time domain data

$$\ln L = \sum_{k=1}^{n} \left\{ \log(ab\alpha) - 2\log s_i - \frac{b}{s_i} + (\alpha - 1)\log[1 - e^{-b/s_i}] \right\}$$

$$- a[1 - (1 - e^{-b/s_n})^{\alpha}]. \tag{19}$$

Taking partial derivative of expression (19) w.r.t. *a*, *b* and α and then equating them to zero, we get likelihood equations, which are

$$a = \frac{n}{1 - (1 - e^{-b/s_n})^{\alpha}}, \tag{20}$$

$$\frac{n}{b} - \sum_{i=1}^{n} \frac{1}{s_i} + (\alpha - 1)\sum_{i=1}^{n} \frac{e^{-b/s_i}}{s_i(1 - e^{-b/s_i})} + \frac{a\alpha}{s_n} e^{-b/s_n}(1 - e^{-b/s_n}) = 0, \tag{21}$$

$$\frac{n}{\alpha} + \sum_{i=1}^{n} \log[1 - e^{-b/s_i}] + a(1 - e^{-b/s_n})^{\alpha} \log[1 - e^{-b/s_n}] = 0. \tag{22}$$

Again here also we do not get closed form expressions for MLEs of a, b and α. However, the MLEs can be obtained by iterative solution procedure. Let $\hat{a}$, $\hat{b}$ and $\hat{\alpha}$ be MLEs of a, b and α, respectively. Then MLEs of $m(t)$ and $\lambda(t)$ are obtained by replacing a, b and α by its MLEs $\hat{a}$, $\hat{b}$ and $\hat{\alpha}$ in expressions (7) and (8), respectively.

4. Numerical Example

We now present a numerical example for finite failure SRGMs based on actual testing-data. The data set consists of 26 software failure-occurrence time data s_k (days; $k = 1, 2, ..., 26$) cited by Goel and Okumoto [3]. Here we obtain the estimation results and AIC value for existing and our proposed finite failure SRGMs are summarized in Table 4.1.

Table 4.1. Maximum likelihood and AIC estimates

Model	**GO**	**Weibull**	***S*-shaped**	**GIED**
Estimates	$\hat{a} = 34.0152$ $\hat{b} = 0.0058$	$\hat{a} = 29.7786$ $\hat{b} = 0.0014$ $\hat{\gamma} = 1.3653$	$\hat{a} = 27.5041$ $\hat{b} = 0.0186$	$\hat{a} = 99.7259$ $\hat{b} = 48.2155$ $\hat{\alpha} = 0.1735$
AIC	161.3803	156.9831	157.836	156.6431

From above table, we observe that our proposed model performs better compared to other models.

5. Conclusions

In this paper, we have proposed finite failure generalized exponential SRGM which was motivated by the fact that the existing finite failure NHPP models were inadequate to describe failure process underlying increasing/ decreasing phenomenon. We use decomposition of the MVF of finite failure NHPP model which enables us to attribute the nature of the failure intensity to the failure occurrence rate per fault. The model parameters are estimated by MLE method and numerical example is illustrated for estimation technique. Finally, model comparison study is carried using Akaike's information criteria and it shows that our proposed model performs better compared to previous finite failure category models.

References

[1] A. M. Abouammoh and A. M. Alshingiti, Reliability estimation of generalized inverted exponential distribution, J. Stat. Comput. Simul. 79(11) (2009), 1301-1315.

[2] A. L. Goel, Software reliability models: assumptions, limitations, and applicability, IEEE Trans. Software Engineering SE-11(12) (1985), 1411-1423.

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

[4] S. Gokhale, P. N. Marinos and K. S. Trivedi, Important milestones in software reliability modeling, Proc. 8th Intl. Conference on Software Engineering and Knowledge Engineering (SEKE'96), Lake Tahoe, 1996, pp. 345-352.

[5] M. R. Lyu, Handbook of Software Reliability Engineering, McGraw-Hill, New York, 1996.

[6] M. Ohba, Software reliability analysis models, IBM J. Res. Develop. 28(4) (1984), 428-442.

[7] H. Pham, System Software Reliability, Springer-Verlag, 2006.

[8] S. Yamada, M. Ohba and S. Osaki, S-shaped reliability growth modeling for software error detection, IEEE Trans. Reliability R-32(5) (1983), 475-484.