ISSN 2394-3777 (Print)
ISSN 2394-3785 (Online)
Available online at www.ijartet.com

International Journal of Advanced Research Trends in Engineering and Technology (IJARTET)
Vol. 8, Special Issue 1, August 2021

A NEW SOFTWARE RELIABILITY GROWTH MODEL::SIZE BIASED INVERSE MAXWELL DISTRIBUTION

Pushpa Latha Mamidi[1], R. Subba Rao[2] Sri Lakshmi Alla[3]
[1]Assoc. Prof. of Mathematics, Vishnu Institute of Technology, Bhimavaram, A.P., India
pushpamamidi@gmail.com
[2] Professor of Mathematics, Sagi Rama Krishnam Raju Engineering College, Bhimavaram, A.P., India
rsubbarao9@gmail.com
[3]Asst. Professor of Mathematics, Mallareddy Institute of Technology and Science, Hyderabad, Telangana, India
Srilakshmialla1984@gmail.com

Abstract: In the Recent development of computer technology, the role of software reliability plays a vital role in almost all the fields of industry, military, pharmaceutical and business etc,, In the management of software, the software cannot fail during its use. Therefore, the software must be rigorously tested by experienced testers before use to minimize errors and malfunctions in the software. There are several software reliability enhancement models for assessing the reliability of software systems. Finding a reliability function for time domain data based on a Non-Homogeneous Poisson Process (NHPP) is essential. The main objective of this paper is to present the Size Biased Inverse Maxwell Distribution (SBIMD) as a software reliability enhancement model and to obtain expressions for the reliability function that enables the reliability of the product to be calculated. The parameters are estimated using the maximum likelihood estimation method. Live data set is analyzed and results are displayed.

Keywords: Software reliability, NHPP, Non-Homogeneous Poisson Process, Maximum Likelihood Estimation.

I. Introduction

Software reliability is the probability failure of free operation of software in a specified environment over a specified period of time. Many models have been proposed over the past 4 decades to gain access to the reliability of the software system, for example Goel & Okumoto (1979), Musa (1980), Crow & Basu (1988), Wood (1996), Pham (2005), Kantam & Subbarao (2009), Sethi & Rana (2011), Vara Prasad *et al.* (2013), Subbarao *et al.* (2015), Subbarao *et al.* (2017), Pushpa Latha *et al.* (2019), Geeta Reddy & Prashant (2019), Hui & Liu (2020). The goal of such models is to improve software performance. These models are related to estimating future system performance from the failure data collected during the testing phase of the software product. Most models assume that the time between failures will follow the exponential distribution with the parameter changing with the number of errors remaining in the software system. The software system is likely to have product and defects of human work. The accuracy of software reliability enhancement models varies significantly when verified using the very few data sets available and despite the existence of many models. None of them can be recommended unreservedly for potential users.

ISSN 2394-3777 (Print)
ISSN 2394-3785 (Online)
Available online at www.ijartet.com

International Journal of Advanced Research Trends in Engineering and Technology (IJARTET)
Vol. 8, Special Issue 1, August 2021

This paper provides an SBIMD to analyze the reliability of the software system. Our goal is to develop a parsimonious model whose parameters have a physical description and can provide quantitative measurements for software performance evaluation. The layout of the paper is as follows: Section II describes the SBIM model and the description of the mean value function for the underlying NHPP. Section III discusses the parametric assessment of SBIMD based on the time between failure data. Section IV describes the methods used for software failure data analysis for live data and contains Section V conclusion.

II Size Biased Inverse Maxwell Distribution Development

Software reliability is best assessed using the model Non-Homogeneous Poisson process. Non-Homogeneous is a computational process that is used to determine the appropriate mean value function m(x), where m(x) represents the expected number of software failures.

The pdf, cdf and mean value function of SBIMD are respectively given by

$$f(t)=\frac{2}{\lambda^2 t^3}e^{\frac{-1}{\lambda^2 t^2}},\ t>0,\lambda>0 \tag{2.1}$$

$$F(t)=e^{\frac{-1}{\lambda^2 t^2}},t>0,\lambda>0 \text{ and} \tag{2.2}$$

$$m(t)=a\,e^{\frac{-1}{\lambda^2 t^2}} \tag{2.3}$$

where ‘λ’ is the scale parameter and m(t)/a is the cdf of SBIMD.

Software reliability models can be classified according to probability estimates. When the Markov process refers to a failure process, the resulting model is called the Markovian model. The second is the fault counting model, which describes the failure phenomenon through a random process such as Homogeneous Poisson Process (HPP), Non Homogeneous Poisson Process (NHPP) and compound poison process. Most failure calculation models are based on the NHPP described in the following lines.

A software system is subject to failures at random times caused by errors present in the system. Let $\{N(t),t>0\}$ be a counting process representing the cumulative number of failures by time ‘t’. Since there are no failures at time t = 0 we have N(0)=0. It is assume that the number of software failures during non overlapping time intervals do not affect each other. In other words, for any finite collection of times $t_1<t_2<....<t_n$ the ‘n’ random variables $N(t_1),\{N(t_2)-N(t_1)\},....\{N(t_n)-N(t_{n-1})\}$ are independent. This implies that the counting process $\{N(t),t>0\}$ has independent increments.

Let m(t) indicate the number of software failures at time ‘t’. Since the number of errors remaining in the system at any time is finite, m(t) is bounded, non decreasing function of ‘t’ with the following boundary condition.

ISSN 2394-3777 (Print)
ISSN 2394-3785 (Online)
Available online at www.ijartet.com

International Journal of Advanced Research Trends in Engineering and Technology (IJARTET)
Vol. 8, Special Issue 1, August 2021

$$m(t) = 0, t = 0$$
$$= a, t \to \infty$$

Where 'a' is the expected number of software errors to be eventually detected.

Suppose N(t) is known to have a Poisson probability mass function with parameter m(t) i.e.,

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

Then N(t) is called NHPP. Thus the random behaviour of the software failure phenomenon can be explained by the N(t) process. Different time domain patterns have appeared in the literature that describes the random failure process by NHPP, which differs in the mean value function m (t).

In this paper we consider m(t) as given by

$$m(t) = a\, e^{\frac{-1}{\lambda^2 t^2}}$$

$$P\{N(t) = n\} = \frac{[m(t)]^n . e^{-m(t)}}{n!}$$

$$Lim_{n \to \infty}\ P\{N(t) = n\} = \frac{.e^{-a} . a^n}{n!}$$

which is also a poisson model with mean 'a'

Let N(t) be the number of errors remaining the system at time 't'

$$N(t) = N(\infty) - N(t)$$

$$E[N(t)] = E[N(\infty)] - E[N(t)]$$
$$= a - m(t)$$
$$= a - a\, e^{\frac{-1}{\lambda^2 t^2}}$$
$$= a\left[1 - e^{\frac{-1}{\lambda^2 t^2}}\right]$$

Let $s_k$the time between $(k-1)^{th}$ and k^{th} failure of the software product. Let x_k be the time up to k^{th} failure. Let us find out the probability that time between $(k-1)^{th}$ and k^{th} failures, exceeds a real number 's' given that the total time up to the $(k-1)^{th}$ failure is equal to x, is

ISSN 2394-3777 (Print)
ISSN 2394-3785 (Online)
Available online at www.ijartet.com

International Journal of Advanced Research Trends in Engineering and Technology (IJARTET)
Vol. 8, Special Issue 1, August 2021

$$P\{S_k / X_{k-1}(s/x)\} = e^{-[m(x+s)-m(s)]} \tag{2.4}$$

This expression is called Software Reliability.

III Modified Maximum Likelihood Estimation of SBIMD

The constants found in the mean value function are 'a', 'λ' and are therefore called the parameters of the model in NHPP and other expressions. Software reliability should be estimated from 'a', 'λ' or they should be estimated from software failure data. Suppose we have a 'n' time instant of experiencing the first, second, third .... n failures of the software. In other words, if x_k is the total time for k^{th} failure, then x_k is an observation of the random variable x_k and such failures are recorded as 'n' respectively. The joint probability of such failure time realizations $x_1, x_2, ...x_n$ is

$$L = e^{-m(x_n)} \prod_{k=1}^{n} m'(x_k) \tag{3.1}$$

The function given in equation (3.1) is called the Likelihood function of the given failure data. Values of 'a', 'λ' that would maximise L are called Maximum Likelihood Estimators (MLEs) and the method is called Maximum Likelihood (ML) method of estimation.

$$L = e^{-\left(ae^{\frac{-1}{\lambda^2 x_n^2}}\right)} \prod_{i=1}^{n} \left[\frac{2a}{\lambda^2 x_i^3} e^{\frac{-1}{\lambda^2 x_i^2}}\right] \tag{3.2}$$

Taking logarithms on both sides of L, to get the log-likelihood equation and the unknown values of 'a' and 'λ' are obtained

$$\log L = -ae^{\frac{-1}{\lambda^2 x_n^2}} + n\log 2a - n\log\lambda^2 + \sum_{i=1}^{n}\log\left(\frac{1}{x_i^3}\right) - \sum_{i=1}^{n}\frac{1}{\lambda^2 x_i^2} \tag{3.3}$$

The unknown values of 'a', 'λ' would be solutions of the equations

$$\frac{\partial \log L}{\partial a} = 0, \quad \frac{\partial \log L}{\partial \lambda} = 0$$

$$\frac{\partial \log L}{\partial a} = 0 \Rightarrow \hat{a} = \frac{n}{e^{-1/\lambda^2 x_n^2}} \tag{3.4}$$

ISSN 2394-3777 (Print)
ISSN 2394-3785 (Online)
Available online at www.ijartet.com

International Journal of Advanced Research Trends in Engineering and Technology (IJARTET)
Vol. 8, Special Issue 1, August 2021

$$\frac{\partial \log L}{\partial \lambda} = 0 \Rightarrow n - \sum_{i=1}^{n} \frac{1}{z_i} + \frac{a}{z_n} e^{\frac{-1}{z_n}} = 0 \tag{3.5}$$

Let us approximate the following expressions in the LHS of equation (3.5) by linear functions in the neighbourhoods of the corresponding variables.

$$\frac{e^{-1/z_n}}{z_n} = \alpha_i + \beta_i zi,\ i = 1,2....n-1 \tag{3.6}$$

Where α_i is the slope, β_i is the intercept, $z_i = \lambda^2 x_i^2$ and $z_n = \lambda^2 x_n^2$

With such values equation (3.6) when used in equation (3.5) would give an approximate MLE for 'λ' as

$$\hat{\lambda}^2 = \frac{(n + a\alpha_n) + \sqrt{(n + a\alpha_n)^2 + 4a\beta_n x_n^2 \left(\sum_{i=1}^{n} \frac{1}{x_i^2} \right)}}{2a\beta_n x_n^2} \tag{3.7}$$

We suggest the following method to get the slope and intercept in the RHS of equation (3.6).

Let $p_i = \frac{i}{n+1}, i = 1,2.....n,\ q_i = 1 - p_i$

Let z_i^*, z_i^{**} be the solutions of

$F(z_i^*) = p_i^*,\ F(z_i^{**}) = p_i^{**}$

Where

$$p_i^* = p_i - \sqrt{\frac{p_i q_i}{n}} \tag{3.8}$$

$$p_i^{**} = p_i + \sqrt{\frac{p_i q_i}{n}} \tag{3.9}$$

Given a natural number 'n' we can get the values of z_i^* and z_i^{**} by inverting the equations (3.8) and (3.9) through the function F(z) the LHS of equation (3.6) we get

ISSN 2394-3777 (Print)
ISSN 2394-3785 (Online)
Available online at www.ijartet.com

International Journal of Advanced Research Trends in Engineering and Technology (IJARTET)
Vol. 8, Special Issue 1, August 2021

$$\beta_i = \frac{\left(\frac{e^{\frac{-1}{z_i^{**}}}}{z_i^{**}}\right) - \left(\frac{e^{\frac{-1}{z_i^{*}}}}{z_i^{*}}\right)}{z_i^{**} - z_i^{*}} \tag{3.10}$$

$$\alpha_i = \left(\frac{e^{\frac{-1}{z_i^{*}}}}{z_i^{*}}\right) - \beta_i z_i^{*} \tag{3.11}$$

It can be seen that the evaluation of x_n, α_i are based on only a specified natural number 'n' and can be computed free from any data. Given the data observations and sample size using these values along with the sample data in equation (3.11), (3.7) we get an approximate MLE of 'λ'. Equation (3.4) gives approximate MLE of 'a'.

IV. Illustration

In this section, we present an analysis of NTDS software failure data taken from Jelinski and Moranda (1972). The data were first released to the U.S. The Navy Fleet comes from the Computer Programming Center, and in real-time software development is flawed, it is a multi-computer complex that forms the core of Naval Tactical Data Systems (NTDS). NTDS software consists of 38 different modules. Each module has to follow three steps; Production phase, testing phase and consumer phase. The data is based on one of the larger modules referred to as the A-module, the trouble reports or the 'software anomaly reports'. The times (days) between software failures and additional information for this module are summarized in the table below.

Table 1: NTDS Software Failure Data

Error Number n	Time between Errors S_k days	Cumulative Time $x_n = \sum s_k$ days
Production (check out) phase		
1	9	9
2	12	21
3	11	32
4	4	36
5	7	43

ISSN 2394-3777 (Print)
ISSN 2394-3785 (Online)
Available online at www.ijartet.com

International Journal of Advanced Research Trends in Engineering and Technology (IJARTET)
Vol. 8, Special Issue 1, August 2021

6	2	45
7	5	50
8	8	58
9	5	63
10	7	70
11	1	71
12	6	77
13	1	78
14	9	87
15	4	91
16	1	92
17	3	95
18	3	98
19	6	104
20	1	105
21	11	116
22	33	149
23	7	156
24	91	247
25	2	249
26		250
Test Phase		
27	87	337
28	47	384
29	12	396
30	9	405
31	135	540
User Phase		
32	258	798
Test Phase		
33	16	814
34	35	849

The data set contains 26 failures in 250 days. 26 software errors were found during the production phase and five additional errors during the testing phase. An error has been detected in the user phase and two more errors have been detected in the next test phase, indicating that the module's network occurred after the user error was detected.

Considered a random sample of size 20 and are assumed to follow SBIMD with 'λ' as a parameter. At λ=1 this assumption is confirmed by the method of QQ plot correlation. The ML estimates so obtained for standard SBIMD are

$\hat{a} = 20.00184$

ISSN 2394-3777 (Print)
ISSN 2394-3785 (Online)
Available online at www.ijartet.com

International Journal of Advanced Research Trends in Engineering and Technology (IJARTET)
Vol. 8, Special Issue 1, August 2021

$\hat{\lambda} = 0.03401$

The estimator of the reliability function from the equation (2.4) at any time x beyond 105 days is given by

$$RS_k / X_{k-1}(s/x) = e^{-[m(x+s)-m(s)]}$$

$$RS_{21} / X_{20}(105/50) = e^{-[m(50+105)-m(105)]}$$
$$= 0.44849$$

V. Conclusion

In this paper we have presented the SBIM software reliability enhancement model with a mean value function. It provides an acceptable explanation of the software failure phenomenon. This is a simple method for model validation and is very convenient for software reliability practitioners.

References

[1] B. Vara Prasad Rao, K. Gangadhara Rao, B. Srinivasa Rao, Inverse Rayleigh Software Reliability Growth Model, International Journal of Computer Applications, 75(6): 1-5, 2013.
[2] Crow, H, and Basu, A.P, Reliability Growth Estimation with Missing Data-II, Proceeding Annual Reliability and Maintainability Symposium, 26-28, 1988.
[3] Goel, A.L., Okumoto, K., Time-dependent error-detection rate model for software reliability and other performance measures. IEEE Trans. Reliab. R-28, 206-211, 1979.
[4] Gunjan Sethi, Poonam Rana, Optimum Software Reliability: A Literature Review, Journal of Business Management and Information Technology, 1(2): 125-129, 2011.
[5] Musa, J.D, The Measurement and Management of Software Reliability, Proceeding of the IEEE, 68(9):1131-1142, 1980.
[6] Musa, J.D, Software Reliability Engineering MCGraw-Hill, 1998.
[7] Pushpa Latha Mamidi, R. Subba Rao, R.N.V. Jagan Mohan, Software Reliability Growth Model Based on Inverse Half Logistic Distribution, International Journal of Recent Technology and Engineering, 8(1S3): 298-301, 2019.
[8] R. Subba Rao, Pushpa Latha Mamidi, R.R.L. Kantam, Estimation through failure censored sample: Size Biased Inverse Maxwell Distribution, International Journal of Computational Science, Mathematics and Engineering, 2(8): 129-133, 2015.
[9] R. Subba Rao, Pushpa Latha Mamidi, R.R.L. Kantam, K. M. Ganesh, Parametric Estimation for Inverse Half Logistic Distribution Based on Type II Censored Samples, Global Journal of Pure and Applied Mathematics, 13(4): 1-6, 2017.
[10] Pham. H, A Generalised Logistic Software Reliability Growth Model, Opsearch, 42(4):332-331, 2005.

ISSN 2394-3777 (Print)
ISSN 2394-3785 (Online)
Available online at www.ijartet.com

[11] R.R.L. Kantam and R. Subbarao, Pareto Distribution: A Software Reliability Growth Model, International Journal of Performability Engineering, 5(3):275-281, 2009.
[12] Wood, A. Predicting Software Reliability, IEEE computer, 2253-2264, 1996.
[13] Y. Geetha Reddy, Y. Prasanth, Maximized Composite Functions Based Optimized Software Reliability Growth Function for Reliability Prediction, International Journal of Scientific & Technology Research, 8(12):910-919, 2019.
[14] Ziqing Hui, Xiaoyan Liu, Research on Software Reliability Growth Model Based on Gaussian New Distribution, Procedia Computer Science, 166: 73-77, 2020.