Indian Journal of Science and Technology, Vol 8(14), DOI: 10.17485/ijst/2015/v8i14/68342, July 2015
ISSN (Print) : 0974-6846
ISSN (Online) : 0974-5645

The Infinite NHPP Software Reliability Model based on Monotonic Intensity Function

Tae-Hyun Yoo*

Department of Science in Taxation, Namseoul University, South Korea; yth@nsu.ac.kr

Abstract

Software testing (debugging) in order to reduce costs in terms of changes in the software reliability and testing costs, need to know in advance is more efficient. In this paper, proposes the reliability model with monotonic intensity functions (Power-Law, Musa-Okumoto and Comperz model), which made out efficiency application for software reliability. Algorithm to estimate the parameters used to maximum likelihood estimation and bisection method, model selection based on mean square error and coefficient of determination, for the sake of efficient model, were employed. Analysis of failure using real data set for the sake of proposing monotonic intensity function was employed. This analysis of failure data compared with monotonic intensity function. In order to insurance for the reliability of data, Laplace trend test was employed. In this study, the proposed monotonic intensity function is more efficient in terms of reliability in this area. Thus, monotonic intensity function can also be used as an alternative model. From this paper, software developers have to consider the growth model by prior knowledge of the software to identify failure modes which can be able to help.

Keywords: Laplace Trend Test, Monotonic Intensity Function, NHPP, Software Reliability

1. Introduction

Software failures caused by failure of computer systems in our society can lead to huge losses. Thus, software reliability in the software development process is an important issue. These issues of the user requirements meet the cost of testing. Software testing (debugging) in order to reduce costs in terms of changes in the software reliability and testing costs, need to know in advance is more efficient. Thus, the reliability, cost, and consideration of release time for software development process are essential. Eventually the software to predict the contents of a defect in the product development model is needed. Until now, many software reliability models have been proposed. Non-Homogenous Poisson Process (NHPP) models rely on an excellent model[1,2] in terms of the error discovery process, and if a fault occurs, immediately remove the debugging process and the assumption that no new fault has occurred.

In this field, enhanced non-homogenous Poisson Process model was presented by Gokhale and Trivedi[1]. Goel and Okumoto[2] proposed an exponential software reliability models. In this model, the total number of defects have S-shaped or exponential-shaped with a mean value function was used. The generalized model relies on these models, delayed S-shaped reliability growth model and inflection S-shaped reliability growth model were proposed by Yamada and Ohba[3]. Zhao[4] proposed a software reliability problems in change point and Shyur[5] using the generalized reliability growth models proposed. Pham and Zhang[6] testing measured coverage, the stability of model, with software stability can be evaluated presented.

Relatively recently, Huang[7], generalized logistic testing-effort function and the change-point parameter by incorporating efficient techniques to predict software reliability, were present. Kuei-Chen[8] can explain the learning process that software managers to become familiar with the software and test tools for S-type model. In addition, Kim[9] studied about the comparative study of NHPP delayed S-shaped and extreme value distribution software reliability model using the perspective of learning effects

*Author for correspondence

and Shin and Kim[10] studied about the comparative study of software optimal release time based on NHPP software reliability model using exponential and log shaped type for the perspective of learning effect.

In this paper, proposes the reliability model with monotonic intensity functions (Power-Law, Musa-Okumoto[13] and Comperz model), which made out efficiency application for software reliability. Algorithm to estimate the parameters used to maximum likelihood estimator and bisection method, model selection based on mean square error and coefficient of determination, for the sake of efficient model, was employed. Analysis of failure using real data set for the sake of proposing monotonic intensity function was employed. This analysis of failure data compared with monotonic intensity function. The purpose of this study is to analyze NHPP software reliability model with monotonic intensity function.

2. Related Works

2.1 NHPP Model

This is a class of time domain[1,11] software reliability models which assume that software failures display the behavior of a Non-Homogeneous Poisson process (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 at time t and $m(t)$ denote its expectation. Then $m(t) = E[N(t)]$ and the failure intensity $\lambda(t)$ is related as follows[9,11]:

$$m(t) = \int_0^t \lambda(s)ds \tag{1}$$

And,

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

$N(t)$ was known to have Poisson probability density function(PDF) with parameter $m(t)$, that is:

$$p(N(t) = n) = \frac{[m(t)]^n}{n!} e^{-m(t)}, n = 0, 1, \cdots, \infty \tag{3}$$

These time domain models for the NHPP process can be described by the probability of failure are possible. This model is the failure intensity function (failure occurrence rates per fault) $\lambda(t)$ expressed differently, also mean value the function $m(t)$ will be expressed differently.

The NHPP models can be further classified into finite failure and infinite failure categories. Finite failure NHPP models assume that the expected number of faults detected given infinite amount of testing time will be finite, whereas the infinite failures models assume that an infinite number of faults would be detected in infinite testing time[11]. Thus, using General Order Statistics (GOS) has become failure NHPP[12]. On the other hand, using Record Value Statistics (RVS) has become infinite NHPP.

Typically, the Generalized Order Statistics (GOS) model has N defects and any of the N defects from the probability density function (PDF); generated n point according to the order statistic is the time point of failure.

In this model, at the time of each repair, a new defect is assumed not to occur. However, the actual situation at the point of repair new failure may occur.

So as to add for this situation, the Record Value Statistics (RVS) model can be used NHPP model and mean value function was as follows: say[11].

$$m(t) = -\ln(1 - F(t)) \tag{4}$$

Where, $\exp(-m(t)) = 1 - F(t)$ and $F(t)$ is cumulative the distribution function and $f(t)$ is the probability density function. Therefore, from Equation (4) using the related equations of NHPP in Equation (1), intensity function can be the hazard function ($h(t)$[1,11]. In other words,

$$\lambda(t) = m'(t) = f(t)/(1 - F(t)) = h(t) \tag{5}$$

Let θ denote the expected number of faults that would be detected given finite failure NHPP models. Then, the mean value function of the finite failure NHPP models can also be written as[9,10]:

$$m(t) = \theta F(t) \tag{6}$$

Note that $F(t)$ is Cumulative Distribution Function (CDF). In Equation (4), the (instantaneous) failure intensity function $\lambda(t)$ in case of the finite failure NHPP models is given by:

$$q(t) = \theta F'(t) \tag{7}$$

This can be re-written as:

$$\lambda(t) = [\theta - m(t)]\frac{F'(t)}{1 - F(t)} [\theta - m(t)]\, h(t) \tag{8}$$

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 course of testing process. The quantity $[\theta - m(t)]$ denotes the expected number of faults remaining in the software at time t. Since $[\theta - m(t)]$ is a monotonically non-increasing function of time (actually $[\theta - m(t)]$ should decrease as more and more faults are detected and removed[1]), the nature of the overall failure intensity, $\lambda(t)$ is governed by the nature of failure occurrence rate per fault $h(t)$, from Equation (6). The failure occurrence rate per fault $h(t)$ can be a constant or increasing, decreasing. In this section, we describe some of the finite failure NHPP models along with their hazard functions[1].

Let $\{t_n, n = 1, 2, \ldots\}$ denote the sequence of times between successive software failures. Then t_n denote the time between $(n-1)^{st}$ and n^{th} failure. Let x_n denote failure time n, so that:

$$x_n = \sum_{i=1}^{n} t_i \tag{9}$$

The joint density or the likelihood function of $x_1, x_2, \ldots, x_n$ can be written as[1,10]:

$$f\, X_1, X_2, \ldots, X_n(x_1, x_2, \ldots, x_n) = e^{-m(x_n)} \prod_{i=1}^{n} \lambda(x_i) \tag{10}$$

For a given sequence of software failure times $(x_1, x_2, \ldots, x_n)$, that are realizations of the random variables $(X_1, X_2, \ldots, X_n)$, the parameters of the software reliability growth models are estimated using the Maximum Likelihood Method (MLE)[10].

As a result, the conditional reliability $\hat{R}(\delta | x_n)$ is known as follows[3,11]:

$$\hat{R}(\delta | x_n) = e^{-\int_{x_n}^{x_n+\delta} \lambda(\tau) d\tau} = \exp[-\{m(\delta + x_n) - m(x_n)\}] \tag{11}$$

Where δ denote mission time and x_n is the last failure time.

2.2 Model Comparison with Real Dataset

In order to investigate the effectiveness of the model, the comparison criteria[8], using mean square errors and coefficient of determination are described as follows:

2.2.1 Mean Square Errors (MSE)

$$MSE = \frac{\sum_{i=1}^{n} (m(x_i) - \hat{m}(x_i))^2}{n-k} \tag{12}$$

Where $m(x_i)$ is the total cumulated number of errors observed within time $0(x_i)$, $\hat{m}(x_i)$ is estimated cumulated number of errors at time x_i obtained from the fitting mean value function, n is the number of observations and k is the number of parameters.

2.2.2 Coefficient of Determination (R^2)

R-Square (R^2) can measure how successful the fit is in explaining the variation of the data. It is defined as follows:

$$R^2 = 1 - \frac{\sum_{i=1}^{n} [m(x_i) - \hat{m}(x_i)]^2}{\sum_{i=1}^{n} [m(x_i) - \sum_{j=1}^{n} m(x_j)/n]^2} \tag{13}$$

3. Software Reliability NHPP Model using Considering Monotonic Intensity Functions

3.1 Power Law Intensity Function

It is known, as follows: Power law intensity function[12].

$$\lambda_{P-L}(t) = \alpha \beta t^{\beta - 1} \tag{14}$$

Note that $\alpha(> 0)$ scale parameter and $\beta(> 0)$ is shape parameter $t \in (0, \infty)$.

The mean value function, using the intensity function Equation (14), is derived as follows:

$$m_{P-L}(t | \Theta) = \int_0^t \lambda_{P-L}(s)\, ds = \alpha t^{\beta} \tag{15}$$

Note that $\Theta = (\alpha, \beta)$ is parameter space.

Therefore, for infinite NHPP failure model using Equation (6), the likelihood function as follows:

$$L_{NHPP}(\Theta | \underline{x}) = \left(\prod_{i=1}^{n} \alpha \beta x_i^{\beta - 1} \right) \exp\left[-\alpha x_n^{\beta} \right] \tag{16}$$

Note: $\underline{x} = (x_1, x_2, \cdots, x_n)$.

The maximum likelihood estimation method of the parameter estimation method was used. Log-likelihood function of Equation (14) is the expression derived as follows.

$$\ln L_{NHPP}(\Theta | \underline{x}) = n \ln \alpha + n \ln \beta - (\beta - 1) \sum_{i=1}^{n} \ln x_i - \alpha x_n^{\beta} \tag{17}$$

Using the expressions (17), $\hat{\alpha}_{MLE}$ and $\hat{\beta}_{MLE}$ satisfy the following equation for the maximum likelihood estimate of each parameter.

$$\frac{\partial \ln L_{NHPP}(\Theta \mid \underline{x})}{\partial \alpha} = \frac{n}{\alpha} - x_n^{\beta} = 0 \quad (18)$$

$$\frac{\partial \ln L_{NHPP}(\Theta \mid \underline{x})}{\partial \beta} = \frac{n}{\beta} - \sum_{i=1}^{n} \ln x_i - \alpha x_n^{\beta} \ln x_n = 0 \quad (19)$$

Using the expressions (10), the reliability is derived as follows:

$$\hat{R}(\delta \mid x_n) = \exp[-\alpha(\delta + x_n)^{\beta} + \alpha x_n^{\beta}] \quad (20)$$

Note that δ is mission time.

3.2 Musa-Okumoto Intensity Function

Mean value function of Musa-Okumoto model is known as a logarithmic function[13,14]. The mean value function is as follows:

$$m_{M-O}(t \mid \Theta) = \alpha \ln(1 + \beta t) \quad (21)$$

Note that $\alpha(> 0)$ is scale parameter, $\beta(> 0)$ is shape parameter and $\Theta = (\alpha, \beta)$ is parameter space. $t \in (0, \infty)$.

The intensity function, using mean value function Equation (21), is derived as follows:

$$\lambda_{M-O}(t) = m'_{M-O}(t) = \frac{\alpha\beta}{(1 + \beta t)} \quad (22)$$

Therefore, for infinite NHPP failure model using Equation (6), the likelihood function as follows

$$L_{NHPP}(\Theta \mid \underline{x}) = \left(\prod_{i=1}^{n} \frac{\alpha\beta}{(1 + \beta x_i)} \right) \exp[-\alpha \ln(1 + \beta x_n)] \quad (23)$$

Note: $\underline{x} = (x_1, x_2, \cdots, x_n)$.

The maximum likelihood estimation method of the parameter estimation method was used. Log-likelihood function of Equation (22) is the expression derived as follows.

$$\ln L_{NHPP}(\Theta \mid \underline{x}) = n \ln \alpha + n \ln \beta - \sum_{i=1}^{n} \ln(1 + \beta x_i) - \alpha \ln(1 + \beta x_n) \quad (24)$$

Using the expressions (24), $\hat{\alpha}_{MLE}$ and $\hat{\beta}_{MLE}$ satisfy the following equation for the maximum likelihood estimate of each parameter.

$$\frac{\partial \ln L_{NHPP}(\Theta \mid \underline{x})}{\partial \alpha} = \frac{n}{\alpha} - \ln(1 + \beta x_n) = 0 \quad (25)$$

$$\frac{\partial \ln L_{NHPP}(\Theta \mid \underline{x})}{\partial \beta} = \frac{n}{\beta} - \left(\sum_{i=1}^{n} \frac{x_i}{(1 + \beta x_i)} \right) - \frac{\alpha\beta}{1 + \beta x_n} = 0 \quad (26)$$

Similarly, the reliability is derived as follows:

$$\hat{R}(\delta \mid x_n) = \exp[-\{m(\delta + x_n) - m(x_n)\}] \quad (27)$$

Note that δ is mission time. $m(x_n) = \alpha \ln(1 + \beta x_n)$, $m(\delta + x_n) = \alpha \ln[1 + \beta(x_n + \delta)]$, $m(\delta + x_n) = \alpha \ln[1 + \beta(x_n + \delta)]$.

3.3 Gompertz Intensity Function

In probability and statistics, the Gompertz distribution[15] is a continuous probability distribution. The Gompertz distribution is often applied to describe the distribution of adult lifespans by demographers and actuaries. Related fields of science such as biology and gerontology also considered the Gompertz distribution for the analysis of survival. More recently, computer scientists have also started to model the failure rates of computer codes by the Gompertz distribution. In marketing science, it has been used as an individual-level model of customer lifetime.

The probability density function and the cumulative distribution function for Gompertz distribution using various fields of industry distribution used are as follows:

$$f(t \mid \alpha, \beta) = \alpha\beta e^{\beta t} e^{\alpha} \exp(-\alpha e^{\beta t}) \quad (28)$$

$$F(t \mid \alpha, \beta) = 1 - \exp(-\alpha(e^{\beta t} - 1)) \quad (29)$$

Note that $\alpha(> 0)$ is shape parameter and $\beta(> 0)$ is scale parameter. $t \in (0, \infty)$.

In infinite failure NHPP, hazard function and intensity are same form. The hazard function, using Equation (28) and Equation (29), is derived as follows:

$$h(t) = \frac{F'(t)}{1 - F(t)} = \frac{f(t)}{1 - F(t)} = \alpha\beta e^{\beta t} = \lambda_{G-P}(t) \quad (30)$$

Note that $\alpha(> 0)$ is scale parameter, $\beta(> 0)$ is shape parameter.

The mean value function, using the intensity function Equation (30), is derived as follows:

$$m_{G-P}(t \mid \Theta) = \int_0^t \lambda_{G-P}(s) ds = \alpha(e^{\beta t} - 1) \quad (31)$$

Similarly, for the Gompertz model, the likelihood function as follows:

$$L_{NHPP}(\Theta \mid \underline{x}) = \left(\prod_{i=1}^{n} \alpha\beta e^{\beta x_i} \right) \exp[-\alpha(e^{\beta x_n} - 1)] \quad (32)$$

Note: $\underline{x} = (x_1, x_2, \cdots, x_n)$.

The maximum likelihood estimation method of the parameter estimation method was used. Log-likelihood

function of Equation (31) is the expression derived as follows.

$$\ln L_{NHPP}(\Theta \mid \underline{x}) = n\ln\alpha + n\ln\beta - \beta\sum_{i=1}^{n}\ln x_i - \alpha(e^{\beta x_n} - 1) \quad (33)$$

Using the expressions (33), $\hat{\alpha}_{MLE}$ and $\hat{\beta}_{MLE}$ satisfy the following equation for the maximum likelihood estimate of each parameter.

$$\frac{\partial \ln L_{NHPP}(\Theta \mid \underline{x})}{\partial\alpha} = \frac{n}{\alpha} - (e^{\beta x_n} - 1) = 0 \quad (34)$$

$$\frac{\partial \ln L_{NHPP}(\Theta \mid \underline{x})}{\partial\beta} = \frac{n}{\beta} + \sum_{i=1}^{n} x_i - \alpha x_n e^{\beta x_n} = 0 \quad (35)$$

Similarly, the reliability is derived as follows:

$$\hat{R}(\delta \mid x_n) = \exp[-\{m(\delta + x_n) - m(x_n)\}] \quad (36)$$

Note that δ is mission time. $m(x_n) = \alpha(e^{\beta x_n} - 1)$, $m(\delta + x_n) = \alpha\left(e^{\beta(\delta + x_n)} - 1\right), m(\delta + x_n) = \alpha\left(e^{\beta(\delta + x_n)} - 1\right)$.

4. Software Failure Time Data

In this section, want to analyze the proposed reliability model using software failure time data[16]. This failure data is listed in Table 1.

Table 1. Failure time data

Failure Number	Failure Time (hours)	Failure Time	Failure Number	Failure Time (hours)	Failure Time
1	0.479	0.0479	16	10.771	1.0771
2	0.745	0.0745	17	10.906	1.0906
3	1.022	0.1022	18	11.183	1.1183
4	1.576	0.1576	19	11.779	1.1779
5	2.61	0.261	20	12.536	1.2536
6	3.559	0.3559	21	12.973	1.2973
7	4.252	0.4252	22	15.203	1.5203
8	4.849	0.4849	23	15.64	1.564
9	4.966	0.4966	24	15.98	1.598
10	5.136	0.5136	25	16.385	1.6385
11	5.253	0.5253	26	16.96	1.696
12	6.527	0.6527	27	17.237	1.7237
13	6.996	0.6996	28	17.6	1.76
14	8.17	0.817	29	18.122	1.8122
15	8.863	0.8863	30	18.735	1.8735

In order to present a reliability model and analyze the trend for first data should be preceded trend test[17]. In general, the Laplace trend test analysis is used. As a result of this test in this Figure 1, as indicated in the Laplace factor is between 2 and –2, reliability growth shows the properties. Thus, using this data, it is possible to estimate the reliability[9].

In this paper, numerical conversion data (Failure time (hours) × 0.1) in order to facilitate the parameter estimation was used. The result of parameter estimation has been summarized in Table 2. These calculations, solving numerically, the initial values given to 0.001 and 3 and tolerance value for width of interval (10^{-5}) given using C-language checking adequate convergent, were performed iteration of 100 times.

The result of Mean Square Error (MSE) and coefficient of determination (R^2) are has been summarized in Table 2. For software model comparison in Table 2, MSE (which measures the difference between the actual value and the

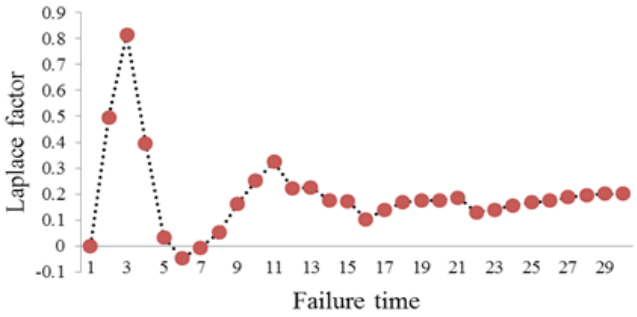

Figure 1. Laplace trend test.

Table 2. Parameter estimation, MLE and MSE

Model	*MLE*	Model Comparison	
		MSE	R^2
Power-Law	$\hat{\alpha}_{MLE} = 2.5250$ $\hat{\beta}_{MLE} = 3.9422$	92.945	0.78
Musa-Okumoto	$\hat{\alpha}_{MLE} = 21.3144$ $\hat{\beta}_{MLE} = 1.6471$	12.037	0.85
Gompertz	$\hat{\alpha}_{MLE} = 219.8915$ $\hat{\beta}_{MLE} = 0.0683$	1.746	0.91

Note. MLE: Maximum likelihood estimation;
MSE: Mean square error;
R^2: Coefficient of determination

predicted value) show that Gompertz NHPP model than Musa-Okumoto and power-law NHPP has small value. Therefore, the Gompertz NHPP is appreciably better than other infinite NHPP model.

Also, R^2 (which means that the predictive power of the difference between predicted values) show that Gompertz NHPP than other model has high value. Thus, Gompertz infinite model is the utility model. Eventually, in terms of MSE and R^2, Gompertz infinite NHPP regard as best model because MSE is the smallest and R^2 is the highest than other models.

In terms of comparison of reliability in Figure 2, case of reliability for assumed mission time, Musa-Okumoto model than other models has shown high reliability. Eventually, the reliability has sensitive for the mission time.

The result of mean value functions are has been summarized in Figure 3. In figure, patterns of mean value function have the tendency of non-decreasing form. Also, in this figure show that Gompertz NHPP model than any models is the utility model because Gompertz infinite NHPP model than any model estimates close to the true value from mean value functions. And, the result of intensity functions are has been summarized in Figure 4. In this figure, intensity functions of power-law model have the tendency of non-increasing and Musa-Okumoto model have non-decreasing form. The other hand, Gompertz NHPP model have the tendency of nearly constant form.

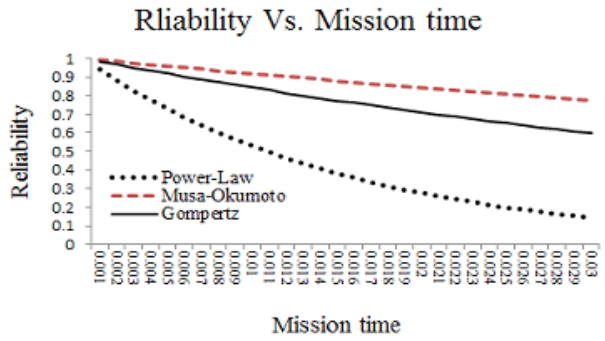

Figure 2. Reliability of each model.

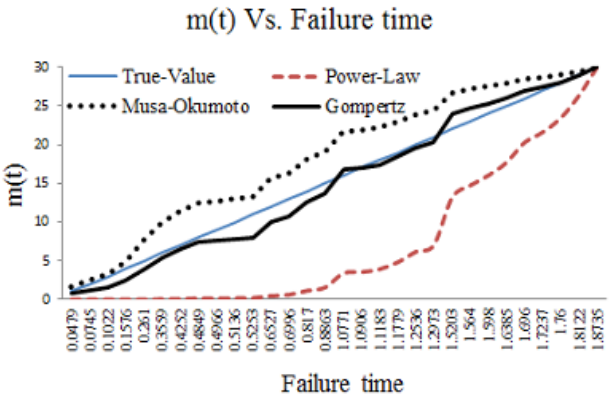

Figure 3. Mean value function of each model.

5. Conclusion

Software reliability growth model can estimate the optimal software release time and the cost of the testing efforts. More accurate model is needed to decrease the testing cost and increase the profit of releasing software. The use of software cost model can help predict the optimal software release time accurately. Compared with previous models, the proposed model takes into account the total number of faults discovered by users during the software operation period or software maintenance after its release, rather than simply assume that residual faults that are not detected will be all found by the user. It can be seen that the cost of actual fault debugging is lower than the cost of removing all remaining faults in the operation phase. So the optimal software release time is ahead of time, and it is more realistic. In further studies, we need to check the validity and effectiveness of our proposed software reliability growth model and the software cost model under the modeling framework by using much actual failure data. And will consider other parameters which can also determine the software testing cost, which is not considered in this paper. Then under this condition, will analyze the new cost model to constantly minimize the cost of software development and get more profits when releasing the software. Generally, when learning factor is the highest and autonomous errors-detected factor is the lowest, model was effective

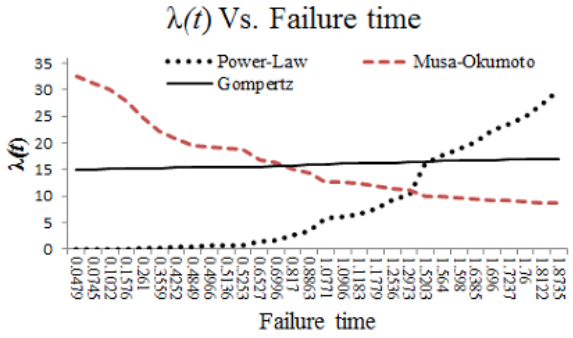

Figure 4. Intensity function of each model.

depressing. Therefore, in this paper, the proposed model can be used as an alternative model in this field. As an alternative to this area feel that the content is a valuable research.

6. Acknowledgment

Funding for this paper was provided by Namseoul University.

7. References

1. Gokhale SS, Trivedi KS. A time/structure based software reliability model. Annals of Software Engineering. 1998; 8:85–12.
2. Goel AL, Okumoto K. Time Dependent Error - Detection Rate model for software reliability and other performance measure. IEEE Trans Reliability. 1979; R-28(3):206–11.
3. Yamada S, Ohba H. S-shaped software reliability modeling for software error detection. IEEE Trans Reliability. 1983; 32:475–84.
4. Zhao M. Change-point problems in software and hardware reliability. Communication Stat Theory Methods. 1993; 22(3):757–68.
5. Shyur H-J. A stochastic software reliability model with imperfect debugging and change-point. J Syst Software. 2003; 66:135–41.
6. Pham H, Zhang X. NHPP software reliability and cost models with testing coverage. Eur. J Oper Res. 2003; 145:445–54.
7. Huang C-Y. Performance analysis of software reliability growth models with testing-effort and change-point. J Syst Software. 2005; 76:181–94.
8. Kuei-Chen C, Yeu-Shiang H, Tzai-Zang L. A study of software reliability growth from the perspective of learning effects. Reliab Eng Syst Saf. 2008; 93:1410–21.
9. Hee-Cheul K. The comparative study of NHPP delayed S-shaped and extreme value distribution software reliability model using the perspective of learning effects. International Journal of Advancements in Computing Technology (IJACT). 2013; 5(9):1210–8.
10. Hyun-Dai S, Hee-Cheul K. The comparative study of software optimal release time based on nhpp software reliability model using exponential and log shaped type for the perspective of learning effects. International Journal of Advancements in Computing Technology (IJACT). 2013; 5(12):120–9.
11. Kuo L, Yang TY. Bayesian computation of software reliability. J Am Stat Assoc. 1996; 91:763–73.
12. Available from: http://www.itl.nist.gov/div898/handbook/apr/section1/apr172.htm
13. Musa JD, Okumoto K. A logarithmic Poisson execution time model for software reliability measurement. ICSE'84. Proceedings of 7th International Conference on Software Engineering; 1984. p. 230–8.
14. Venkata SRK, Raveendra Babu B. A log based approach for software reliability modeling. Int J Adv Res Comput Sci Software Eng. 2014; 4(2):49–51.
15. Available from: http://en.wikipedia.org/wiki/Gompertz_distribution
16. Hayakawa Y, Telfar G. Mixed poisson-type processes with application in software reliability. Math Comput Model. 2000; 31:151–6.
17. Kanoun K, Laprie JC. Handbook of software reliability engineering. In: Lyu MR, editor. Chapter Trend Analysis. New York, NY: McGraw-Hill; 1996. p. 401–37.