Advances and Applications in Mathematical Sciences
Volume 19, Issue 6, April 2020, Pages 563-572

ESTIMATION OF COMPLEXITY IN SOFTWARE RELIABILITY GROWTH MODELING

PRIYANKA THAKUR and SHIV K. SHARMA

Department of Mathematics
Chandigarh University
Gharuan, Mohali, Punjab 140413, India
E-mail: thakurpriyanka109@gmail.com
shivksharma2005@gmail.com

Abstract

Software reliability growth models are a statistical interpolation of defect detection data by mathematical models. In this paper we determine the best confidence interval with less normality assumptions. Traditional models may not be appropriate in some situations. The proposed models [3] provide the consistent performance in software reliability and software complexity.

1. Introduction

Software reliability is the possibility of failure-free operations in the software for a particular time period in a particular situation. Software Reliability plays a vital role on affecting the reliability of the system. It is different from the hardware reliability in that it shows the design perfection, rather than manufacturing perfection. The modeling methods for software reliability are getting its prosperity but before using the methods, we cautiously selected the suitable model that can be excellent [5]. Many methods can be used to improve the software reliability.

Software reliability has a unique feature of reliability engineering. System reliability includes all parts of the system, including hardware, software, supporting infrastructure, operators and procedures [5].

2010 Mathematics Subject Classification: 60K10, 60K20, 62N05, 68M15.

Keywords: Software Reliability growth models, Goel-okumoto, Multi stage models, Weibull distribution, Maximum likelihood.

Received 25 November 2019; Accepted 6 January 2019

Samuel Taylor Coleridge in 1816 defined the word Reliability [12]. Reliability is the regularity of a set of dimensions or dimension instruments, in Statistics. The reliability and reliability Engineering doesn't exist earlier than the World War II. During the period of WWII, Many new electronic products were introduced.

The software reliability is given by Martin Shooman [10]. The Software failures are treated as chance which can be formed during the time axis, because of the lack of assurance by their appearance of accurate moment, so due to this we signify their reliability, not the quality, of the software [8].

Reliability means the possibility of a system will work for a particular time period with exclusive of any failure in particular working circumstances. It is considered with the time between breakdown and the breakdown rates.

The software system is thus like to the position of the breakdown method with the hardware system through reinstatement and describe by the equal collection of reliability indicators as the latter [8].

Weibull distribution is a continuous probability distribution. In 1951 Swedish mathematicians Waloddi Weibull, were described Weibull distribution in detail, though it was firstly recognized by Frechet and first time practically apply by Rosin and Rammler for describing a particle size distribution[4]. The Weibull distribution has attracted many researchers working on theories or techniques in many fields of applied statistics.

2. An Analysis of Software Reliability Growth Model

Mostly existing software reliability models improvement consider for the constant variable. Where, there are systems such business deal handing out system where the reliability should be considered as of transaction handled profitably and apart from that, there are new system i.e. command software of missile in which it is very easy to calculate the reliability and it can also measure the number of missiles successfully launched. The code coverage indicator can affect the software reliability estimation and it can also reach saturation in which there is no need of case new parts of code to be tested. So, as per this the reliability estimations which rely totally on finishing moment can be more than approximation the reliability of the plan.

Method used to evaluate the value of the models:

Predictive Validity: It shows the ability of the model to guess about the upcoming breakdown actions, all over the difficult stage prediction that we got from existing breakdown actions and from the past in the particular stage. The following steps are followed:

- Exactness (accuracy level), calculated by Maximum Likelihood.

- Inclination, calculated by the multi stage models.

- Trend, and the methodical change the capability, from little to large values of the breakdown moment, attributes calculated by the Weibull Distribution.

- Noise, calculated by the comparative transform appeared in the guess estimation of breakdown.

Capability: This means the capability of model to make estimation through the dimensions of suitable exactness that the engineers or the software user's required in the arrangement or allocating the software progress projects. The sizes includes such as the ongoing reliability, the predict meeting for attaining the reliability goal, and as well as the required price to reach to that goal.

The characteristics of the interpretation, if a assumption is build by the model can be reliable, then the trait represent the quantity to which the model is established by the genuine information; still, the hypotheses of testing is not achievable, the trait represent to the possibility from the point of view of the consistent reliability of the knowledge achievement by the engineers. The clearness and clarification of the model interpretation necessary to be determine.

Applicability: We can apply the models within various software functions. The software models can be use in various industries, and in different departments such as operations, manufacturing etc.

Simplicity: Easiness of information collection is compulsory in the outlook of customization of the model. It represents the theoretical easiness to the viewers of the model so that they can easily recognize the characteristics of the model and its hypotheses. It can also find out the applicability of the difficulty and the trait also refers to the ease of completion in a method so that the model can be a useful device of organization.

Easiness of calculating the parameters [7]. The traits take into the account the quantity of parameters are necessary by the model and the degree of complexity encountered in their inference.

G-O model based on every faults in a plan are equally free of the breakdown recognition point of view [1]. The possibility of the breakdown for faults really occurring this means detected regularly. The software failure occurs at each time, the software fault which is caused are instantly removed and no new faults can be introduced.

G-O propose that, before adapting the model the estimated number of errors in time (t) and is signify by $\sigma(t)$, where t is calendar time, execution time, or number of tests executed. According to figure, the fundamental stair about the model adopting according to the agreement not the similarity in the predicted observed and predicted values.

The methods, like the G-O, Weibull Distribution and maximum likelihood, work only when positive amount detail exit. In the software errors, then few of software breakdown arises and be observed. In case of consistent programmable devices that are in procedure, the supposition is frequently impractical. So, these techniques are improved suitably to assessing the SRGM, which addresses principally the growth stair of the life cycle of the software.

The software development is a structure that shows a set of tasks to be performed at every step in the software development process. The life cycle also shows the methodology to improve the nature of the software or the overall development process.

In first step of the software cycle is preliminary data analysis. To collect the requirements the most capable or experienced software engineers are used to do so. They are collecting the require data. After collecting the required data, the requirements get analyzed and after that check the validity and if it is feasible to reach the requirements then it includes in the system.

The requirements specification that has been studied in the previous stage, the software design that means the selection of the software is done in this phase. The selection of the model is essential and it helps in many ways. The software model selected at this step and serves input to the next step.

At this step the system designs or software model uses by the team that was received by the previous phase. The estimation of the parameter every unit is then performed. This step is the lengthy step of the life cycle.

Every units in the earlier step is tested in this step to see the product approximately same to the requirements collected in the primary step and does it is really supposed to do. In this phase many type of test that can be performed.

In this step the developed system from the earlier phase, that has been successfully tested and if it has been not successfully tested than the process again starts from the second phase that is selection of the model. After that in the next phase the prediction calculations is done. The calculations or the units which are predicted are done in this phase.

After this there are three phases. One is remaining error, in this after the prediction calculation there is some error in the software. Second one is reliability function, after the prediction calculation the developers check the software is reliable. The reliability function is checked by the software developers and in the third and the last phase the duration features are to be checked until the next failure.

2.1. Goel-Okumoto (G-O)

G-O model [1] for $\sigma(t)$ is defined as

$$\sigma(t) = p(1 - e^{-qt}),$$

where

$p =$ estimated faults

$q =$ shape factor $=$ rate approaches to the total number of defects.

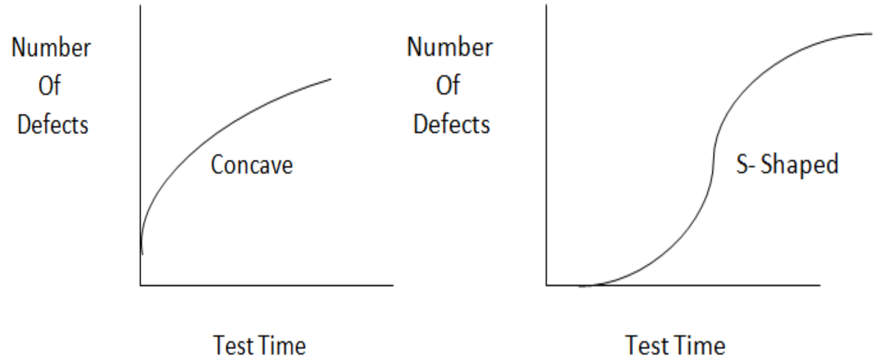

Figure 1. Concave model and S-shaped model.

The G-O model is a concave shaped model, and the parameter p can be plotted as the number of faults in figure 1. The G-O models have two parameters. For maximum models, $\sigma(t) = pF(t)$, where p is the estimation fault in the code and $F(t)$ is a cumulative distribution function.

If $F(0) = 0$, so before the test start no defects were originate, and $F(\infty) = 1$, so $\sigma(\infty) = p$ where p is estimation faults are originate after an infinite amount of testing [9].

Table1 Software Reliability Growth Model (Concave and S-shaped).

Name of the Model	Type of the Model	$\sigma(t)$	Reference
Goel-Okumoto	Concave	$p(1 - e^{-qt})$ $p \geq 0,\ q > 0$	Goel [3]
G-O S-Shaped	S-shaped	$p(1 - (1 + qt)e^{-qt})$ $p \geq 0,\ q > 0$	Yamada [13]

2.2. Multi-Stage Models

The assumptions for the models are to facilitate the arrangement of system being difficultly is unaffected all through the test period [1]. Obviously, fault reconstruct cancel the supposition, but the effects of defect reconstruct are minimum therefore the model can be excellent estimate. The important amount of original system is added during the test phase, the techniques that concede us to convert the information to account for the improved system change. The difficulty is that addition of major amount of change system should enlarge the defects in the detection rate. Therefore, D_1 defects are created in T_1 time prior in the adding of the latest system and the adding $D_2 - D_1$ defects are created in $T_2 - T_1$ time after the system addition [1]. The difficulty is to convert the information to a model $\sigma(t)$ that would be obtained if the latest system has been section of the software [10].

Let $\sigma_1(t)$ model the fault information prior to the addition of the latest system, and suppose $\sigma_2(t)$ model the fault information after that the system addition. The model $\sigma(t)$ is formed by suitably modifying the breakdown period from $\sigma(t), \sigma_1(t), \sigma_2(t)$ are all G-O models. This method can be applicable in any model including S-shaped models.

Let us assume that the model $\sigma_1(t)$ applied in the time phase $0 - T_1$ and the model $\sigma_2(t)$ applied in the time phase from $T_1 - T_2$. Now firstly to decided the parameters of the models $\sigma_1(t)$ and $\sigma_2(t)$ to get $\sigma_1(t) = p_1(1 - e^{-q_1 t})$ and $\sigma_2(t) = p_2(1 - e^{-q_2 t})$. For calculating $\sigma_1(t)$ the basic technique are used and the data in time period is $0 - T_1$ and for calculating $\sigma_2(t)$ also basic technique are used now assume that the test initiated at time T_1 and created $D_2 - D_1$ defects [1]. When calculating $\sigma_2(t)$ subtract D_1 from the cumulative defects and subtract T_1 from the "cumulative time.

2.3. Weibull Distribution

Weibull distribution has two parameters [4]:

1. Probability density function (pdf)

2. Cumulative distribution function (cdf)

1. Y is the random variable and is said to be Weibull distribution having parameters α and β where $(\alpha > 0, \beta > 0)$ is the pdf of x is

$$F(Y, \alpha, \beta) = \begin{cases} \dfrac{\alpha}{\beta^{\alpha}} Y^{\alpha - 1} e^{-(Y/\beta)^{\alpha}} & Y \geq 0 \\ 0 & Y < 0 \end{cases}$$

Where $\alpha > 0$ is shape parameter and $\beta > 0$ is scale parameter of the distribution.

The Weibull distribution is interrelated to a number of other possibility distributions, in particular, it interpolates between the exponential distribution $(\alpha = 1)$ and the Rayleigh distribution $(\alpha = 2$ and $\beta = \sqrt{2}\sigma)$.

- If $\alpha < 1$ then the breakdown rate decreases with time.
- If $\alpha = 1$ then the breakdown rate is constant with time.
- If $\alpha > 1$ then the breakdown rate increases with time.

2. Y is the random variable and is said to be Weibull distribution having parameters α and β is the cdf of Y is

$$F(Y;\ \alpha,\ \beta) = 1 - e^{-(Y/\beta)^{\alpha}}.$$

Where $\alpha > 0$ is shape parameter and $\beta > 0$ is scale parameter of the distribution.

For $Y \geq 0$, and $F(Y;\ \alpha,\ \beta) = 0$ for $Y < 0$

If $Y = \beta$ then $F(Y;\ \alpha,\ \beta) = 1 - e^{-1} \approx 0.632$ for all values of α.

When $F(Y;\ \alpha,\ \beta) = 0.632$ the value of $Y \approx \beta$.

2.4. Maximum Likelihood

A set of instantaneous equations for parameter values can be solved by maximum likelihood techniques [1]. The equation define the parameter that maximal the likelihood that the detected information came from the distribution with those parameter value. It would be satisfied a number of significant numerical situation for a finest estimator and is usually deliberated to be the most excellent numerical estimator for big sample sizes. The instantaneous equations are usually solved numerically and are very complex. The general discussion of maximum likelihood theory and equation derivation, in [9] or [11]. For the G-O model [1] the equations should be solved to give parameter estimates and confidence intervals.

G-O model expected defects are

$$\sigma(t) = p(1 - e^{-qt}),$$

where p = estimated faults

and q = shape factor = breakdown rate decreases

It is easily to see that Maximum likelihood techniques represent the most simplify version of Weibull distribution. The observations are not dependent and the probability domain is discrete which is coinciding. So, the maximum likelihood techniques are easier to find.

Conclusion

In this literature multifarious models were purposed. They have their own disadvantages. Various models have been taken some of them behave same in various cases. The history of software usages that has in its initial state of time, there is no defectively and a different familiar difficulty of these models is that they bear instantly as per reliability requirement. So solution of these problems is very difficult.

Various existing software reliability models do not take into deliberation appliance complexity. We have discussed maximum likelihood and G-O model. It requires less normality assumptions and provide best confidence interval. Further, things are complicated again because G-O model [1] works concave and S-shaped models gives the complexity for a component of a system. We discussed with Weibull distribution [4] has interpolate between exponential distribution $(\alpha = 1)$ and Rayleigh distribution $(\alpha = 2)$.

To the models that takes into consideration the software reliability and Software complexity. Find expected defects with Weibull distribution this should be taken as priority.

We use various software models together then such cases results may be improved.

References

[1] Alan Wood, Software Reliability Growth Models, Tandem Computers, 1996.

[2] S. Basu and N. Ebrahimi, Estimating the Number of Undetected Errors: Bayesian Model Selection in Proceeding of the Ninth International Symposium On Software Reliability Engineering, 1998, pp. 22.

[3] A. L. Goel and A. Kazuhira Okumoto, Time Dependent Error Detection Model for Software Reliability and Other Performance Measures, IEEE Trans. Reliability, 1979, vol R-28, pp 206-211.

[4] https://en.wikipedia.org/wiki/Weibull_distribution

[5] https://www.uniassignment.com/essay-samples/information-technology/the-history-of-software-reliability-information-technology-essay.php#reference

[6] A. Iannino, J. D. Musa, K. Okumoto and B. Littlewood, Criteria for Software Reliability Model comparisons ACM SIGSOFT Software Engineering Notes 8(3) (1983), 12-16.

[7] M. R. Lyu and A. Nikora, Applying Reliability Models More Effectively, IEEE Software Transactions 9(4) (1992), 43-52.

[8] Marian Pompiliu Cristescu, Eduard Alexandru Stoica, Laurentiu Vasile Ciovica, The Comparison of Software Reliability Assessment Models, ELSEVIER, 27 (2015), 669-675.

[9] Mood, Alexander, Franklin Graybill and Duane Boes, Introduction to the Theory of Statistics, McGraw-Hill, 1974.

[10] M. Shooman, Software Reliability: A Historical Perspective, IEEE Transactions on Reliability R-33 (1984), 48-55.

[11] Musa, John, Anthony Iannino and Kazuhira Okumoto, Software Reliability, McGraw-Hill, 1987.

[12] J. H. Saleh and Marais, Ken, Highlights from the Early (and pre-) History of Reliability Engineering, Reliability Engineering and System Safety 91(2) (2006), 249-256.

[13] Yamada, Shigeru, Mitsuru Ohba and Shunji Osaki, *S*-Shaped Reliability Growth Modeling for Software Error Detection, IEEE Trans. Reliability R-32 (1983), 475-484.