Some Flexible Software Reliability Growth Models using Two-Dimensional Approach

Ompal Singh[1,*], Jyotish N.P. Singh[2], Jyotish Kumar[3] and P.K. Kapur[4]
[1,2,3]Department of Operational Research, University of Delhi, Delhi, India
[4]Amity International Business School, Amity University, Noida, U.P., India

The concern for software reliability has grown over a period of time especially with the advent of real life systems such as satellite and shuttle control, telephone, internet and banking services. In today's life the computers are being used to monitor and control safety critical and civilian systems with a great demand for high-quality software products. So reliability is a primary concern for both software developers and software users. The new outlines of competition and collaboration that have arisen in software engineering as a result of the globalization process have an impact on the entire software process. The need is growing to estimate, risk assess, plan and manage the development of these complex software systems to ensure a smooth run of life. Therefore to capture the combined effect of testing time and testing coverage we propose a two dimensional software reliability growth model. We have used cobb-douglas production function to develop the two dimensional model incorporating the effect of testing time and testing coverage on the number of faults removed in the software system. In this paper, we present two software reliability growth models incorporating logistic distribution function and Exponentiated Exponential distribution function respectively. The model developed is validated on real data set.

Keywords: Software reliability growth models (SRGMs), Cobb–Douglas production function, Logistic distribution function, Exponentiated exponential distribution function.

1. INTRODUCTION

Software plays a key role in the modern life. Software is a functioning element in home appliances, automobiles, space ships, banking, communications, manufacture etc. This has increased our dependence on machines and its reliability. The idea of unreliable software may be unimaginable and damaging. A malfunctioning pacemaker in a heart patient or a Mars path finder that has lost contact due to a software bug or a hacker taking advantage of a bug in financial system to siphon away cash electronically in the luxury of his home portrays the problem. Inadequate testing of the delivery system of Titan IV rocket lead to two Titan rockets being lost. Expensive military equipment necessary to the U.S. Governments defence program (namely early warning satellites) were unable to be deployed. The head of the N.R.O. (National Reconnaissance Office) has attributed this error to "a misplaced decimal point" in software, which controlled the rocket. This has lead to devising methods to make the software more reliable.

The concern for software reliability has grown over a period of time especially with the advent of real life systems such as satellite and shuttle control, telephone, internet and

banking services. With the growing economy there has been a growing interest of companies to know about their competitors and have been spending a lot on strategic decision making. All these activities require complex software systems. It is important that these systems are thoroughly tested before implementation.

Software testing is the process of executing a program or system with the intent of finding errors. It is the method of raising the confidence that the software is free of flaws. But a major problem in testing software is that it cannot be made 100% bugs free. This is not because programmers are careless or irresponsible, but because the nature of software code is complex and humans have only limited ability to manage complexity. Therefore, although testing helps in assessing and improving quality, but it cannot be performed indefinitely. Hence, testing time is one of the major factors that have to be considered during the testing phase.

An SRGM is defined as a tool that can be used to evaluate the software quantitatively, develop test status, schedule status, and monitor the changes in reliability performance. Software reliability assessment and prediction is important to evaluate the performance of software system. The reliability of the software is quantified in terms of the estimated number of faults remaining in the software system. During the testing phase, the emphasis is on reducing the remaining fault content hence increasing the software reliability. A huge number of software reliability growth models (abbreviated as SRGMs) [1,2,3,4-6] which assess software reliability quantitatively have been proposed so far. However, almost all of the SRGMs have been developing under the assumption that the software reliability growth process depends only on the testing-time as the software reliability growth factor essentially. On the other hand, it is known that a software reliability growth process in a testing-phase is influenced by the following several software reliability factors: the test execution-time, the testing-skill, the testing coverage, and so forth.

Yamada et al. [7,8,9] proposed a testing-effort dependent SRGM based on a non homogeneous assumption that the fault-detection rate is proportional to the testing-effort expenditure. Fujiwara and Yamada [9] developed a testing-domain dependent SRGM which incorporates the testing-skill of test-case designers, where the testing-domain means a set of testing-paths in the software system to be influenced by executed test-cases. And Inoue and Yamada [10] discussed a testing-coverage dependent SRGM by characterizing the relationship between the testing-coverage attainment process and the software reliability growth process mathematically and developing a testing-coverage function to describe a time-dependent behaviour of a testing-coverage attainment process with the testing-skill of test-case designer. However, such SRGMs, which consider with the effect of the software reliability growth factors to the software reliability growth processes, depend only on the testing-time essentially. Then, we have doubt that the software reliability growth process depends only on the testing-time in fact. One of the solutions to the problems mentioned above is developing an SRGM which depends on the testing-time and other reliability growth factors simultaneously. Developing such SRGM would be more feasible for describing a software reliability growth process in an actual testing-phase. In recent years, Ishii and Dohi [11] proposed a two dimensional software reliability growth model and their application. They investigated the dependence of test-execution time as a testing effort on the software reliability assessment, and validate quantitatively the software reliability models with two-time scales. Inoue and

Yamada [10,12,13] also proposed two dimensional software reliability growth models. However their modeling framework was not a direct representative of using mean value functions to represent the fault removal process. They discussed software reliability assessment method by using two dimensional Weibull type SRGM. This study aims to compare the predictive capability of two popular software reliability growth models, say flexible logistic growth and exponentiated exponential growth. We present an exponentiated exponential growth model, which can capture the increasing or constant or decreasing nature of the failure occurrence rate per fault.

In this paper we discuss two dimensional SRGMs which enable us to expect more feasible software reliability assessment than the conventional software reliability measurement approach. To start with, we define one-dimensional unified approach for describing failure-occurrence or fault-detection phenomenon before discussing our two-dimensional software reliability growth modelling framework. After that, we present a unified framework for developing two-dimensional SGRM with respect to testing time and testing coverage.

Further, we conduct goodness-of-fit comparisons with several SRGMs proposed so far, and show numerical examples of software reliability assessment based on our model by using actual fault count data collected along with testing-coverage data.

2. NOTATIONS

a : Number of faults lying dormant in the software at the beginning of testing.

$m(t)$: Expected number of faults removed in the time interval (0, t).

b : Constant fault detection/isolation/correction rate.

$f_i(t)$: Probability density function.

$F_i(t)$: Probability distribution function.

α : Effect of testing parameter.

ϕ : Shape parameter for exponentiated exponential function.

φ : Scale parameter for exponentiated exponential function.

β : Constant learning parameter.

3. FINITE FAILURE SOFTWARE RELIABILITY GROWTH MODELING

The NHPP models are based on the assumption that the software system is subject to failures at random times caused by manifestation of the remaining faults in the system. Hence, NHPP are used to describe the failure phenomenon during the testing phase. Let $\{N(t)\ , t \geq 0\}$ be a counting process representing the cumulative number of software failures by time t. The counting process is assumed to be a NHPP with a mean value function $m(t)$ which represents the expected number of faults removed by time t.

Based on the NHPP assumption, it can be shown that *N (t)* has Poisson distribution with mean $m(t)$, *i.e.*,

$$\Pr\{N(t)=k\}=\frac{m(t)^k}{k!}e^{-m(t)} \quad k=0,1,2,...$$

and $m(t)=\int_0^t \lambda(s)\,ds$

The intensity function $\lambda(s)$ (or mean value function *m (t)*) is the basic building block of all NHPP models existing in software reliability engineering literature.

The proposed models are based upon the following basic assumptions:

a. The failure observation and fault removal phenomenon is modeled using a NHPP.
b. Software is subject to failures during execution caused by faults remaining in the software.
c. Each time a failure is observed, an immediate debugging effort takes place to find the cause of the failure to remove it.

Based on the above assumptions the differential equation describing the rate of change in the cumulative number of faults is given as follows:

$$\begin{aligned}\frac{dm(t)}{dt}&=\frac{f(t)}{1-F(t)}(a-m(t))\\&=h(t)(a-m(t))\end{aligned} \tag{1}$$

Here, $h(t)$ is defined as the hazard rate with which the faults are detected/corrected in the software system and (a-m(t)) denotes the expected number of faults remaining in the software at time 't'.

Solving equation (1) using the initial condition that at t=0, m(t)=0 , we get

$$m(t)=a.F(t) \tag{2}$$

Equation (2) is the Generalised NHPP SRGM model. Substituting different types of distribution functions, i.e. different values for F(t) in (2), we can obtain different mean value functions corresponding to them [2,7,14,15].

4. TWO DIMENSIONAL MODELING

Two-dimensional software reliability models have been developed to assess the software quantitatively. The need for developing a two dimensional model is an ideal solution to the problem of software reliability at the hands of software engineers. Two dimensional models are used to capture the joint effect of testing time and testing coverage on the number of faults removed in the software. The traditional one dimensional model has been dependent upon the testing time or testing effort or testing coverage. However if the reliability of a software is measured on the basis on the number of hours spent on testing the software or the percentage of software that has been covered then the results

are not conclusive. To cater the need of high precision software reliability we require a software reliability growth model which caters not only the testing time but also the testing coverage of the software i.e. the percentage of code covered of the software. For this we develop a two dimensional software reliability growth model incorporating the joint effect of testing time and testing effort on the number of faults removed in the software. The two dimensional model developed in this paper is based on the Cobb Douglas production function. The Cobb–Douglas functional form of production functions is widely used to represent the relationship of an output to inputs. It was proposed by Knut Wicksell (1851–1926), and tested against statistical evidence by Charles Cobb and Paul Douglas in (1900–1928).The Cobb-Douglas function considered a simplified view of the economy in which production output is determined by the amount of labor involved and the amount of capital invested. While there are many factors affecting economic performance, their model proved to be remarkably accurate.

The mathematical form of the production function is given as follows:

$$Y = A L^{v} K^{1-v}$$

Where *Y* = total production (the monetary value of all goods produced in a year), *L* = labor input, *K* = capital input, A = total factor productivity v is elasticity of labor. This value is constant and determined by available technology.

The assumptions made by Cobb and Douglas can be stated as follows:

a. If either labor or capital vanishes, then so will production.
b. The marginal productivity of labor is proportional to the amount of production per unit of labor.
c. The marginal productivity of capital is proportional to the amount of production per unit of capital.

The Cobb-Douglas function based on the above assumptions is very appealing. The basic characteristic of this function is linearly homogeneous with constant return to scale i.e. a proportion increase in all inputs leads to same proportion increase in output. Testing should intentionally attempt to make things go wrong to determine if things happen when they shouldn't or things don't happen when they should. It is oriented to 'detection'. The testing team has many resources of testing to make sure that software hence formed is of quality. These include software testing man hours, CPU time, testing effort testing coverage etc. Testing resources τ are given as:

$$\tau \cong s^{\alpha} u^{1-\alpha} \qquad 0 \le \alpha \le 1 \tag{3}$$

Where s : testing time, u : testing coverage and α : Effect of testing time.

Let $\{N(s,u), s \ge 0, u \ge 0\}$ be a two-dimensional stochastic process representing the cumulative number of software failures by time s and testing coverage u. A two-dimensional NHPP with a mean value function m (s,u) is formulated as:

$$\Pr(N(s,u)=n)=\frac{(m(s,u))^n}{n!}\exp((-m(s,u)) \text{ , n= 0, 1, 2…}$$

and $$m(s,u)=\int_0^s\int_0^u \lambda(\zeta,\xi)\,d\zeta\,d\xi$$

5. S-SHAPED TWO–DIMENSIONAL MODELS: UNIFIED APPROACH

In one dimensional analysis the object variable is dependent on one basic variable although the object takes on many different roles based upon its dependence on various other factors. Because a software failure is occurred when a input data hits to a software fault latent in a program, it is natural to say that important factors affecting the software reliability growth process are not only testing-time but also testing effort expenditure such as testing-coverage, the number of executed test-cases, and so on. Especially, the testing coverage is one of the important measures indicating the test adequacy and efficiency.

In this paper we develop a two dimension S-shaped model determining the combined effect of testing time and testing coverage.

Now we extend the testing time of one dimensional to a two dimensional problem considering testing resource as variable. Using the cobb-douglas production the corresponding mean value function with respect to testing resources is given as follows:

$$m(\tau)=a.F(\tau) \tag{4}$$

Equation (4) is the generalized two-dimensional SRGM model (SRGMs). Substituting different types of distribution functions, we can obtain different mean value functions corresponding to them. The cumulative number of faults removed $m(\tau)$ is dependent on testing resources τ . Here, τ is a two-dimensional variable, with testing time *s* and testing coverage *u* as its dimensions. The two-dimensional models are useful as they can show the effect of two aspects of a variable on which the result is dependent [11,16,17].

We define some additional notations as follows:

m (s,u): mean number of faults removed corresponding to coverage *u* and time s.

$$m(s,u)=a.F(s,u) \tag{5}$$

5.1. Some Flexible SRGMs

The Logistic distribution function can be used to model SRGM-1. The mean value function with respect to testing resources is given as follows:

$$m(s,u)=a.\frac{(1-\exp(-b.s^{\alpha}.u^{(1-\alpha)}))}{(1+\beta.\exp(-b.s^{\alpha}.u^{(1-\alpha)}))} \tag{6}$$

The exponentiated Exponential distribution function can be used to model SRGM-2. The mean value function with respect to testing resources is given as follows [18,19,20]:

$$m(s,u) = a.\left(1 - e^{-\phi.s^{\alpha}.u^{(1-\alpha)}}\right)^{\psi} \quad (7)$$

The above mean value functions $m(\tau)$ represent the cumulative number of faults removed dependent on testing time *s* and testing coverage *u*. Earlier many SRGMs have been discussed either it was time dependent, testing effort dependent or coverage dependent but here we develop a generalised framework for deriving two-dimensional SRGMs which show the joint effect of two metrics testing time and testing coverage to show the number of faults removed in the software.

6. DATA SET DESCRIPTION

The data sets used for the two dimensional model are coverage data set. These data sets show the combined effect of testing time and testing coverage. We have used data set in which cumulative numbers of faults found in data set 1 are 9 with execution of 796 test cases [17,21] and 95.99% of test coverage.

7. ESTIMATION OF PARAMETERS MODEL VALIDATION AND COMPARISON CRITERIA

Parameter estimation is of prime significance in software reliability prediction. Once the analytical solution for mean number of faults detected/removed by time t given by $m(t)$ that is mostly described by the non-linear functions is known for a given model, the parameters in the solution are required to be determined. Parameter estimation is achieved by extensively used estimation techniques for non-linear models method of Non-linear Least Square (NLLS). In most of the cases solving this system of equations is difficult and contrary to the linear model fitting, as we cannot express the solution of this optimization problem analytically. Moreover, it requires numerical methods and huge computation time to solve the problem, which is not favored by the management and software engineering practitioners. Statistical software packages such as SPSS, SAS, Mathematica etc. help to overcome this problem in which we can use the inbuilt software functions to solve these kinds of optimization problems to find the estimates of nonlinear models. In our study we have used the Statistical Package for Social Sciences (SPSS**).** SPSS is a comprehensive and flexible statistical analysis and data management system. It can take data from almost any type of file and use them to generate tabulated reports, charts, and plots of distributions and trends, descriptive statistics, and conduct complex statistical analysis. Regression models enables the user to apply more sophisticated models to the data using its wide range of nonlinear regression models. For the estimation of the parameters of the proposed model, nonlinear regression (NLR) modules of SPSS have been used. The modules use the iterative estimation algorithms namely sequential quadratic programming and Levenberg-Marquardt method to find the least square estimates of the parameters. In this paper we have taken three comparisons criteria as given below:

- Coefficient of Multiple Determinations (R^2).
- Bias.

- Mean Square Error (MSE).

8. RESULT OF PARAMETER ESTIMATION

The parameter estimation and comparison criteria results for data set of both models under consideration are given in Table 1 and Table 2 respectively. The fitting of the models to data set can be graphically illustrated. The testing resources are different for all the distributions as the value of α, are estimated using the failure data. SRGM-2 has low Bias and MSE, depicting less fitting error. For SRGM-2 the real-time data set as adjusted R^2 is 0.992.

Table 1: Parameter Estimated values.

Parameters	a	b	β	ϕ	φ	α
SRGM-1	9.157	0.0214	0.092	-	-	0.532
SRGM-2	9.044	-	-	0.0189	0.588	0.79

Table 2: Comparisons Criteria.

Criterion	Adj R^2	Bias	MSE
SRGM-1	0.991	0.009	0.0681
SRGM-2	0.992	-0.00771	0.0612

The curve of the estimated values of the number of faults removed in given data using the proposed modeling framework for both the SRGMs is shown graphically in Figures 1 and 2, respectively.

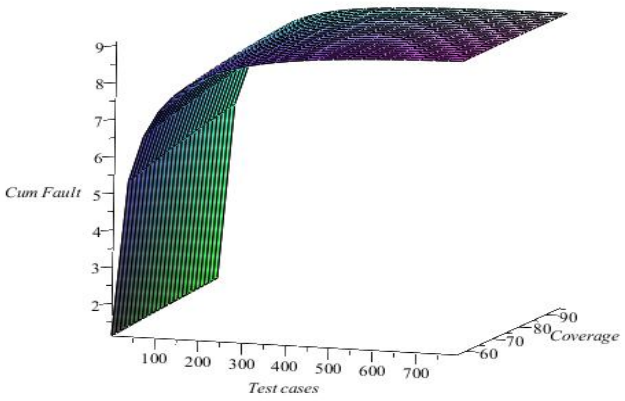

Fig. 1: Goodness of fit for SRGM-1.

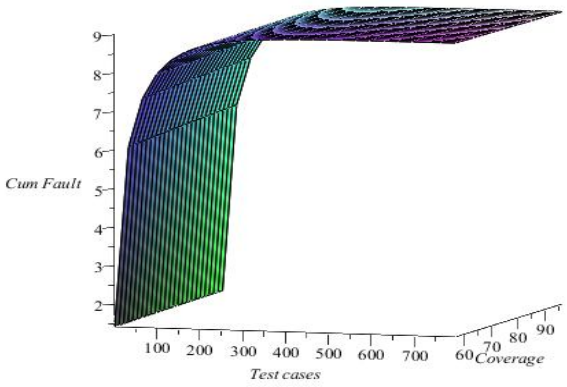

Fig. 2: Goodness of fit for SRGM-2.

9. CONCLUSION

This paper discussed a two-dimensional software reliability growth modelling framework. Especially in our discussion, we considered that an actual software reliability growth process depends not only on testing-time but also on testing-coverage expenditure.

Our model enables us to describe software reliability growth process depending on such two types of the software reliability growth factors as the two-dimensional software reliability growth process. Thus, we can say that software project managers can conduct more feasible and accurate software reliability assessment by using our two-dimensional SRGM. We have compared our two-dimensional models based on Bias and MSE.

REFERENCES

[1] P.K. Kapur, R.B. Garg and S. Kumar; “Contributions to Hardware and Software Reliability”, World Scientific Singapore, 1999.

[2] P.K. Kapur, H. Pham, Anshu Gupta, P.C. Jha; “Software reliability Assessment with OR application”, Springer London, 2011.

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

[4] S. Yamada, K. Tokuno, and S. Osaki; “Imperfect debugging models with fault introduction rate for software reliability assessment”, Int. J. Syst. Science, Vol. 23, pp. 2241-2252, 1992.

[5] S. Yamada and S. Osaki; “Software reliability growth modelling Models and applications”, IEEE Trans. Soft. Eng., Vol. SE-11(12), pp. 1431-1437, 1985.

[6] J.D. Musa, D. Iannio and K. Okumoto; “Software Reliability Measurement”, Prediction, Application, McGraw-Hill, New York, 1987.

[7] S. Yamada, H. Ohtera, and H. Narihisa; “Software reliability growth models with testing-effort”, IEEE Trans. Reliab., Vol. R-35(1), pp. 19-23, 1986.

[8] S. Yamada, J. Hishitani and S. Osaki; "Software reliability growth with a Weibull test-effort", IEEE Trans. Reliab., Vol. 42(1), pp. 100-106, 1993.

[9] T. Fujiwara and S. Yamada; "Software reliability growth modeling based on testing-skill characteristics: Model and Application", Elec. Commu. Japan (Part 3), Vol. 84(6), pp. 42-49, 2001.

[10] S. Inoue and S. Yamada; "Testing-coverage dependent software reliability growth modeling", Intern. J. Rel. Quali. Safe. Eng., Vol. 11(4), pp. 303-312, 2004.

[11] T. Ishii and T. Dohi; "Two-dimensional software reliability models and their application", (PRDC) 12th Pacific Rim International Symposium on Dependable Computing, pp.1-8, 2006.

[12] S. Inoue, and S. Yamada; "Testing-coverage dependent software reliability growth modelling", Int. J. Reliability, Quality and Safety Engineering, Vol. 11(4), pp. 303-312, 2004.

[13] S. Inoue and S. Yamada; "Two-dimensional software reliability measurement technologies", Proceedings of IEEE IEEEM, pp. 223-227, 2009.

[14] H. Pham; "Software reliability assessment: Imperfect debugging and multiple failure types in software development", EG&G-RAMM-10737, Idaho National Engineering Laboratory, 1993.

[15] H. Pham; "Software Reliability", Springer-Verlag, Singapore, 2000.

[16] S. Inoue and S. Yamada; "Two-dimensional software reliability assessment with testing coverage", Second International Conference on Secure System Integration and Reliability Improvement, pp. 150-156, 14-17 July 2008.

[17] P.K. Kapur, R.B. Garg, G.A. Aggarwal and A. Tandon; "Two-dimensional flexible software reliability growth model and release policy", Proceedings of the Fourth National Conference on Computing for Nation Development-INDIA Com-2010, 25-26, New Delhi, pp. 431-438, February 2010.

[18] R.D. Gupt, D. Kundu; "Exponentiated exponential family; an alternative to gamma and Weibull", Biometrical Journal, Vol. 43, pp. 117-130, 2001.

[19] G.S. Mudholkar and D.K. Srivastava; "Exponentiated Weibull family for analyzing bathtub failure data", IEEE Transactions of Reliability, Vol. 42, pp. 299-302, 1993.

[20] J.C. Ahuja and S.W. Nash; "The generalized Gompertz-Verhulst family of distribution", Sankhya, Vol. A(29), pp. 141-156, 1967.

[21] S. Wang, Y. Wu, M. Lu and H. Li; "Software reliability accelerated testing method based on test coverage", Proceeding of Reliability and Maintainability Symposium (RAMS), pp. 1-7, Jan 2011.