

Software Reliability Growth Models Incorporating Software Project/Application's Characteristics as a Power Function with Change Point



Shinji Inoue, Abhishek Tandon, and Prarna Mehta

Abstract Software Reliability is a high-handed aspect to ascertain the quality of the system, which leads to development of tools that incorporates real time set-up. One of the real time concepts is change point, which highlights on the fact that a characteristic of the model changes during the testing time duration and it is significant to incorporate its effect in the model developed to ameliorate the reliability of the system. Moreover, faults are assumed to be independent and are incurred at any arbitrary time but, in practical world, faults may occur due to many factors like the testing environment, resource allocation, code complexity, testing team skill-set etc. This rate tends to change with time and assuming it to be constant may not reflect upon an actual output. Another conundrum that is taken care of is the release time of the software project/application. The weightage that is implied by this optimization planning is due to the fact that over testing may incur high cost to the firm whereas under testing may lead to release of a project/application with high fixing cost faults affecting the manufacturer by an elevated post-release cost. In this chapter, a framework is proposed that extends error-removal phenomenon model by encapsulating the software project/application characteristics as a parameter. A realistic software development situation is also taken in account by considering the parameter not only as a constant but a time dependent function. The suggested SRGMs are also monitored under a change point scenario, which gives real-time edge to the problem, and are then utilized to develop release time policy balancing reliability and expected cost incurred by a project/an application. The models are validated using Tandem dataset and performance measures are compared quantitatively with the standard models.

S. Inoue

Faculty of Informatics, Kansai University, 2-1-1 Ryozenji-cho, Takatsuki-shi, Osaka 569-1095, Japan

A. Tandon

Department of Management Studies, Shaheed Sukhdev College of Business Studies, University of Delhi, Delhi, India

e-mail: abhishektandon@sscbsdu.ac.in

P. Mehta (✉)

Department of Operational Research, University of Delhi, Delhi, India

© Springer Nature Switzerland AG 2022

A. G. Aggarwal et al. (eds.), *Optimization Models in Software Reliability*,

Springer Series in Reliability Engineering,

https://doi.org/10.1007/978-3-030-78919-0_2

On comparison of the results, the proposed model outperforms other extant models with and without change point.

Keywords Software reliability · Software reliability growth model · Change point · Power function · Release policy · Software testing phase

1 Introduction

The ambitious invention of a high performing and a high-defined quality software system has given software engineers the liability to envisage a more complex system. Software brings forth automation to a wide range of domain like medicine, banking system, business houses, academics, security etc. thus, becoming an indispensable element of any commercial or non-commercial field. With the increase in its application, there is a need of improvising software at a higher speed to balance with the requirements of the consumers, to keep up with the competitors in the market and most significantly to maintain quality. Thus, reliability models are entailed to assess and forecast the reliability of the software embedded in the system quantitatively, elevating system's reliability (Huang et al. 1999) and are also known as Software Reliability growth models (SRGM). A SRGM that helps in rendering a balanced amalgamation in terms of expense, reliability, productivity and performance can be considered to be generic in nature and applicable to sundry software project/application.

Many versions and forms of SRGM has been developed considering different assumptions and objective of the underlying study, under perfect or imperfect debugging conditions (Okumoto and Goel 1979). However, in the real world, there are many other factors that influence the quality of the software with time for example testing environment, complexity of the code, running environment, line of codes, testability etc. Lai et al. (2011) stated past SRGMs fail to incorporate these factors into past simulation. Thus, fault-detecting rate is influenced by external factors and it would be favorable to take them into account in order to increase the performance rate of a SRGM developed using past failure rate. In this study, these external factors are included in the model developed and coined as software application/project characteristic parameter. The given parameter is not only a function of software characteristics but can also be a time dependent power function. The parameter measuring the influence of the software project/application's characteristic may not necessarily be constant over the testing time period and is subject to change due to alteration in testing plan at some time-point. This instance of time is termed as change-point. In this chapter, SRGMs are formulated under the postulation of change point to create a real-time problem.

The main concern of the management is scheduling the testing phase and release of the project but are unaware of the latent faults remaining in the system even after the completion of testing phase (Kapur et al. 2008a). However faults in a software project/application are extensively eliminated during the testing phase of the

system but excessive testing is uneconomical and impractical, thereby making the optimal release planning as a powerful decision-making aspect in a business plan (Lai et al. 2011). SRGM provides enumeration of faults that might be informative to the management to curtail development cost or to anticipate the appropriate release time of a software project/application (Huang 2005a). To précis, software release planning is a vital managerial optimization problem that involves determination of the optimal release time of the software such that the cost is minimized subject to a reliability constraint. Kapur et al. (2009a) have discussed the significance of release planning that if a product is subject early to or late release, then it may lead to either high operational cost or high testing cost which is a loss for the business. In this study, the expected number of faults obtained through SRGMs with and without change point scenarios are employed to develop optimal release plans for the respective models.

The objective of this study is condensed in three-fold. Firstly, a model is simulated taking into account of change point concept and software project/application as a power function. Secondly, models are assessed for its performance and efficiency based on a real software fault dataset and comparison is made with some extant models from the past literature. Thirdly, an optimization scheme is planned for the release of the product using the models developed. The scheme specifies an optimal combination of cost and reliability for a given software project/application, thereby focusing on a rational release policy for the same.

This chapter is divided into sections, namely, in Sect. 2 a brief summary of the past literature has been captured, following which in Sect. 3 the suggested model has been explained, in Sect. 4 the proposed model is justified with a numerical illustration, a release plan has been discussed in Sect. 5 using the developed models and lastly, Sect. 6 concludes the given study.

2 Literature Survey

In this section, an abridgment of the related literature is furnished for each research topic alluded in this study.

2.1 *Software Reliability Growth Models*

A SRGM follows NHPP distribution that gives an estimated count of faults for both calendar as well as running timeline. In the past literature, a vast number of SRGMs are available differentiating from each other by marginal changes in assumptions set describing or tackling a testing related problem (Lai and Garg 2012). Yamada et al. (1983) proposed an S-shaped model due the non-uniformity in reliability growth that was also a succession to Okumoto and Goel (1979) model. Kareer et al. (1990) proposed a modified S-shaped model that was based on the severity level of the faults

manifesting a real time situation. Jeske and Zhang (2005) gave a realistic approach by bridging a gap between theoretical and practical application of SRGM. Many earlier SRGMs do not fit all failure datasets perfectly as stated by Chiu et al. (2008) however, incorporating learning effect into the model would enhance the results. Effects of testing skills, strategy and environment on discrete SRGMs were well captured by Kapur et al. (2008b). Kapur et al. (2009b) have presented a unified approach to evaluate a wide range of SRGMs on the basis of hazard rate. A SRGM framework demonstrated by Inoue and Yamada (2011) reflected the effect of rate of change in the testing setting. Softwares are developed in an ideal environment nonetheless, it is not the same while in the operating stage (Pham 2016). While Zhu and Pham (2018) considered single environment factor and later on taking in account the effect of multiple environmental factors on modeling and estimation (Zhu and Pham 2020).

2.2 Change Point

Zhao (1993) was motivated to apply change point concept in the field of software and hardware reliability by including it SRGM. Huang (2005b) integrated logistic testing effort and change point into the reliability modeling suggesting that the project head to invest in tools and manpower that would aid in improving the reliability of the software product. Singh et al. (2010) introduced S-shaped SRGM with change point that monitors the reliability of OSS. A discrete SRGM was modeled by Goswami et al. (2007) for different levels of bugs severity under change point. Kapur et al. (2009a) have discussed a general framework with respect to altering rate of fault detection with and without change point. In the study by Chatterjee et al. (2012), SRGM with fluctuating introduction and detection rates have been examined under change point scenario. Inoue et al. (2013) scrutinized the altering effect of testing environment on change point based SRGM. Parr-curve was deployed by Ke et al. (2014) to analyze reliability model with multiple change point. Nagaraju and Fiondella (2017) have exemplified the significance of change point modeling by assessing different SRGM with and without change point. Change point SRGMs depicts mathematically a real life situation, however, Inoue and Yamada (2018) utilized Markov process to enhance the depiction. Chatterjee and Shukla (2017) used an improvised approach to regulate the reliability assessment under change point effect.

2.3 Release Policy

SRGMs are implemented to enumerate uncertainty attached to a software system, thus aiding a software developer to achieve a highly reliable software project/application. But with increasing complexity in the software, it is hard for the software engineers to deliver fault free. At the same time, these probabilistic tools are utilized to estimate the optimal release time. The concept of optimal release

policy was firstly implemented by Okumoto and Goel (1979). Following which, many researchers started implementing the optimal release time concept along with software reliability assessment as well as balancing cost parameters (Kapur and Garg 1989). Shrivastava et al. (2020) proposed an optimization model to establish the optimal release time as well as the testing stop time. Li et al. (2010) study not only deals with multiple change point and release policies but, they have also made an effort to do a sensitive analysis of the results. The failure data being uncertain and vague, pushed the researchers to indulge in fuzzy sets. Under the fuzzy environment, a release plan for the software was given by Pachauri et al. (2013). For a discrete SRGM, a release policy was proposed by Aggarwal et al. (2015). Release time decision-making can be sometime tricky when the past records are inconsistent, conversely (Chatterjee and Shukla 2017) suggested a fuzzy based release schedule. Shrivastava and Sachdeva (2019) proposed a generalized release policy under different testing environment.

3 Methodology

In this study, a SRGM has been enhanced by considering a parameter as a power function of time integrating with a change point scenario.

3.1 Notations

Symbols	Description
X	Preliminary faults existing in the software project/application in the time period $(0, t]$
$X(t)$	For a given time period, say $(0, t]$, expected number of observed faults
p	Failure rate for a software project/application
q	Residual fault detection rate
r_1	Software project/application's characteristics measuring factor before change point
r_2	Software project/application's characteristics measuring factor after change point
k	A constant
t	Time period
τ	Change point

3.2 Assumptions

Comprehending from the past literature (Huang et al. 1999; Huang 2005a; Yamada et al. 1983; Kareer et al. 1990), in this study, SRGMs are developed, based on Non-Homogenous Poisson Process (NHPP) where, $X(t)$ is the mean value function or the expected number of faults in the time interval $(0, t]$. The proposed models are developed under the following assumptions,

- Due to the presence of the dormant faults, the software project/application is subject to fail during the operational phase.
- A fault detected in the software project/application is removed instantaneously.
- On removal of certain faults, it is assumed that some latent dependent faults are also eliminated in the process.
- The software project/application's operational phase is considered to be as one of the phases of the lifecycle.
- The failure or fault detection phenomenon incurred by the consumer or the testing team is considered to be equivalent.
- Software project/application characteristics' effect on the model is measured by a parameter, which might change with time.

3.3 Model Development

Kapur and Garg (1992) proposed a model that was based on the assumption that faults detected can lead to observing residual faults in the system without causing the system to fail. This model has been extended by considering a parameter that characterizes software project/application as a parameter namely, r_1 . Hence, the differential equation under the given assumptions for an expected number of faults in the system is given,

$$\frac{dX(t)}{dt} = r(t)\left[\left(p + \frac{X(t)}{X}\right)q(X - X(t))\right] \quad (1)$$

where, $r(t)$ can be a constant or a time-dependent function that defines the software project/application characteristics. Zhu and Pham (2020) developed a generic model including environmental factor and on the similar lines (Inoue et al. 2013) amplified fault detection rate with the help of environment factor in their cost model. Different set up for the model in Eq. (1) has been considered in the study.

Model 1a: When $r(t)$ is considered to be a constant without change point (BCP)

Let us consider,

$$r(t) = r_1 \quad (2)$$

where, r_1 is a constant value of the above defined parameter and $r_1 > 0$. Substituting (2) in (1), the differential equation for the expected number of faults in the system is given by,

$$\frac{dX(t)}{dt} = r_1 \left[\left(p + \frac{X(t)}{X} q \right) (X - X(t)) \right] \tag{3}$$

The constant r_1 implies that each fault is uniformly affected or influenced by software project/application characteristics parameter. Solving (3), we obtain the expected number of faults in the testing time interval $(0, t]$, given the initial condition, $X(0) = 0$, as,

$$X(t) = \frac{X[1 - e^{-(p+q)r_1 t}]}{[1 + \frac{q}{p} e^{-(p+q)r_1 t}]} \tag{4}$$

Model 1b: When $r(t)$ is considered to be a constant with change point (ACP)

During the testing phase, a software project/application runs in a given environment but it is not necessarily true that parameters remain uniform throughout. The instance at which a change is observed in the pattern of the failure distribution is termed as change point. In software reliability engineering, a change point occurs due to change in testing pattern observed the testing team's, change in resource allocation, improved learning skills of the system analyst, or programmed testing codes (Kapur et al. 2008a; Zhu and Pham 2020; Ke et al. 2014). In this study, software project/application characteristics are not considered to be homogenous during the testing phase, hence developing SRGM with a change point. Let the parameter be defined as,

$$r(t) = \begin{cases} r_1, & t < \tau \\ r_2, & t \geq \tau \end{cases}$$

Considering the underlying assumptions and assuming that $r(t)$ does not remain homogeneous during the testing phase, the mean value function is given as,

$$X(t) = \begin{cases} X \frac{[1 - \exp(-(p+q)r_1 t)]}{[1 + \frac{q}{p} \exp(-(p+q)r_1 t)]} & t < \tau \\ X \frac{[1 - (1 + \frac{q}{p})(1 + \frac{q}{p}) \exp(-r_1 \tau)] [\exp(-(p+q)r_1 \tau) + (-(p+q)r_2(t-\tau))]}{[1 + \frac{q}{p} \exp(-r_1 \tau)] [1 + \frac{q}{p} \exp(-r_2 t)]} & t \geq \tau \end{cases} \tag{5}$$

Model 2a: When $r(t)$ is considered as a power function BCP

Next, let us consider as a power function of testing time which is given as,

$$r(t) = r_1 t^k \tag{6}$$

The form in (6) renders the suggested models generic and adaptable in nature (Kapur et al. 2008a). Substituting (6) in (1), the differential equation is given as,

$$\frac{dX(t)}{dt} = r_1 t^k \left[p + q \frac{X(t)}{X} \right] [(X - X(t))] \quad (7)$$

where, $k \geq 0$ and the first component represents the varying relation between r_1 and t with varying value of k . If $k = 0$, the model reduces to Eq. (3). Taking the initial condition as $X(0) = 0$, we get,

$$X(t) = X \frac{[1 - \exp(-r_1(p + q) \frac{t^{k+1}}{k+1})]}{[1 + \frac{q}{p} \exp(-r_1(p + q) \frac{t^{k+1}}{k+1})]} \quad (8)$$

Model 2b: When $r(t)$ is considered as a power function ACP

The Eq. (3) has been extended by incorporating change point and power function of r .

$$X(t) = \begin{cases} X \frac{[1 - \exp(-\frac{(p+q)r_1}{k+1} t^{k+1})]}{[1 + \frac{q}{p} \exp(-\frac{(p+q)r_1}{k+1} t^{k+1})]} & t < \tau \\ X \frac{[1 - (1 + \frac{q}{p})(1 + \frac{q}{p}) \exp(-\frac{r_1}{k+1} \tau^{k+1})] \exp(-\frac{(p+q)r_1}{k+1} \tau^{k+1}) + (-\frac{(p+q)r_2}{k+1} (t-\tau)^{k+1})]}{[1 + \frac{q}{p} \exp(-\frac{r_1}{k+1} \tau^{k+1})] [1 + \frac{q}{p} \exp(-\frac{r_2}{k+1} t^{k+1})]} & t \geq \tau \end{cases} \quad (9)$$

For simplification, in this study, only one change point has been considered in both the models given by (5) and (9), however in the practical world, change points are observed more than once in a testing phase.

4 Model Validation

In this section, the SRGMs given by (3), (5), (8) and (9) have been validated on the basis of fault dataset given by Wood (1996). A change point has been observed at the 8th week of testing in the dataset and change point models given by (5) and (9) are evaluated accordingly. The parameters are estimated for each model and the performances of the respective models are recorded for comparisons.

4.1 Dataset

The proposed models have been validated using Tandem dataset (Wood 1996) that contains number of faults detected over a testing period of 20 weeks and cumulative

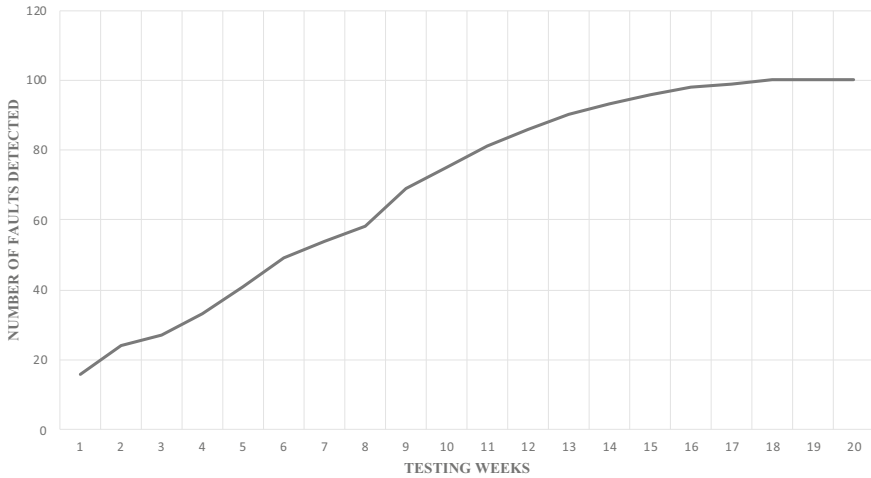


Fig. 1 A plot for Tandem Dataset

sum of number of faults being 100. This dataset has been widely implemented in the past literature for example, Park and Baik (2015), Sharma (2010), Roy et al. (2014). From the data set, a change in the number of faults detected was observed at 8th week of testing. Figure 1 depicts a graph plotting tandem data set where x-axis denote the testing weeks whereas y-axis denote cumulative number of faults detected.

4.2 Parameter Estimation

The non-linear regression option in Statistical Package for social Science (SPSS) aids in obtaining least square estimates of the parameters for a given SRGM. The parameters are valued for a given testing time in weeks denoted by t and parameters $X, p, q, k, r, r_1,$ and r_2 delineated in Sect. 3 are estimated for respective SRGM. Table 1 represents the parameter estimation for without and with change point models.

In Table 1, “-” indicates null value of the parameter because they have no role play in the respective model. 35.1% and 93.7% software project/application characteristics are explained by the parameter r in model 1a and 2a respectively. However, without

Table 1 Parameter estimations for models before and after change point scenario

Models	X	p	q	r_1	r_2	k
Model 1a	110.829	0.222	0.268	0.351	-	-
Model 1b	116.48	0.48	0.455	0.238	0.091	-
Model 2a	114.104	0.0046	0.431	0.937	-	0.377
Model 2b	112.375	0.113	0.464	0.552	0.432	0.394

Table 2 Model comparisons with respect to performance measures

Models	R ²	MSE
Model 1a	0.989	8.971
Model 1b	0.993	5.442
Model 2a	0.991	7.183
Model 2b	0.997	2.076

change point, model 1b explains 23.8% and with change point 0.91%, whereas model 2b shows the best result, combining the effect of r_1 and r_2 , 98.4% characteristics are explained.

A SRGM is said to perform well when it predicts the behavior of the system accurately by using previous fault data (Roy et al. 2014). The simulations developed were assessed with the help of performance measures, co-efficient of determination (R^2) and mean square of errors (MSE).

Looking at the R^2 and MSE values of the proposed models in Table 2, model 2b shows the best fit for the given dataset i.e. 99.7% of the variation in data is captured by the proposed model and has the lowest MSE value (2.076).

4.3 Goodness of Fit Curves

See Figs. 2 and 3.

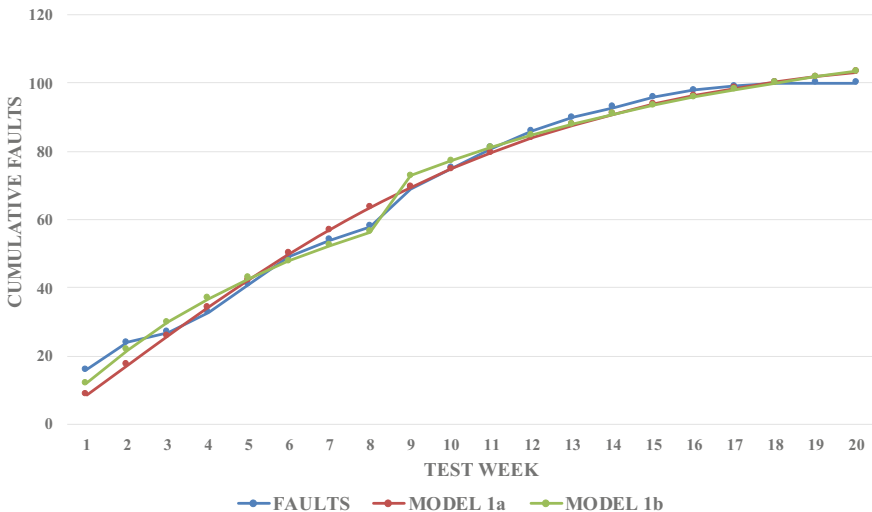


Fig. 2 Actual versus predicted values for proposed models where r is considered constant BCP and ACP

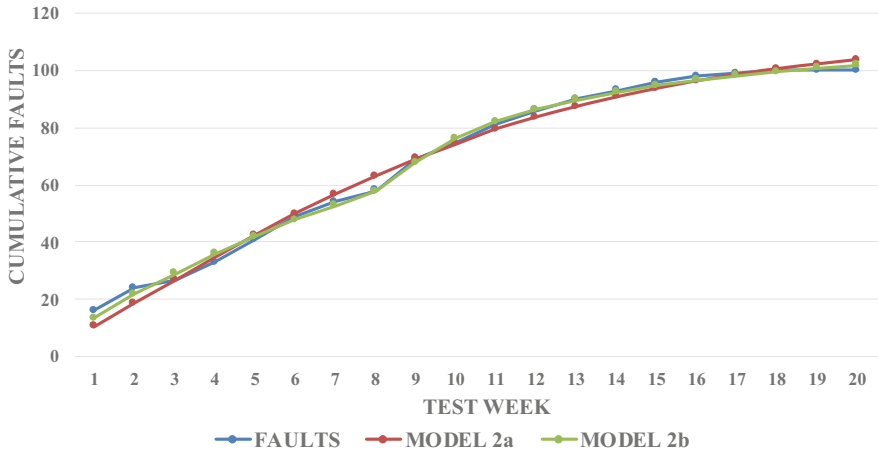


Fig. 3 Actual versus predicted values for proposed models where r is power function of time BCP and ACP

4.4 Comparison with the Previous Models

In order to validate the robustness of the proposed model, comparisons are drawn with some conventional models, namely Goel Okumoto model (Okumoto and Goel 1979), S-shaped Yamada model (Yamada et al. 1983), Kapur Garg error removal phenomenon model (Kapur and Garg 1992) respectively using the Tandem dataset. For mathematical simplifications, the proposed model 1a has been compared with the traditional models.

Table 3 depicts the comparison values for the different SRGMs with respect to change point. Nonetheless, the MSE value is lowest for the proposed model without as well as with change point than that of GO model and S-Shaped Yamada model which implies that the proposed model has the least fitting error and hence exhibit the best performance. Since the proposed model is an extension of Kapur and Garg error phenomenon model, the comparison measure is vital for this case. For the given dataset, KG Model has MSE value 18.572 and 10.937 without and with change point respectively, while those values for model 1a are nearly half of it i.e. the suggested

Table 3 Models' performance comparison with respect to coefficient of determination

	R ²		MSE	
	BCP	ACP	BCP	ACP
Model 1a	0.989	0.993	8.971	5.442
GO model	0.965	0.988	28.503	9.392
Yamada model	0.969	0.991	25.266	7.57
KG model	0.977	0.987	18.572	10.937

model has the capability to reduce the fitting error to half of that of the KG model. All in all, the extended model is a better version of KG model. Looking at the R^2 values, the proposed model has the highest (0.989 and 0.993) without and with change point cases in comparison to other models. Figures 4 and 5 reflects that the suggested model is the closest fit of the data set without and with change point respectively.

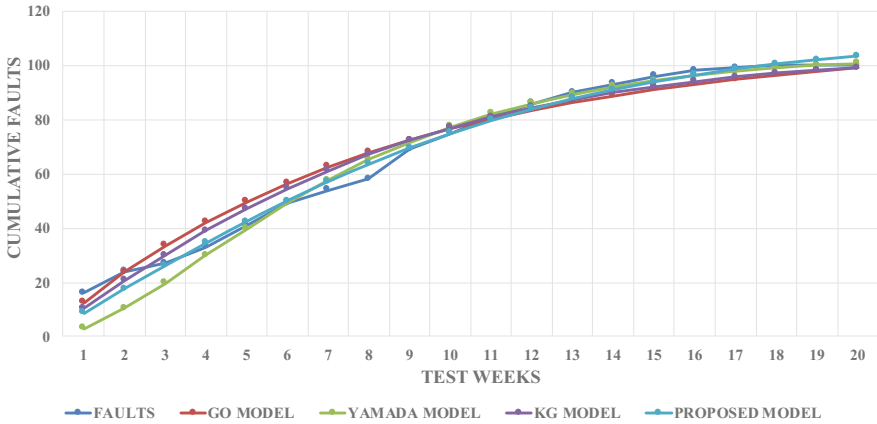


Fig. 4 A fitted curve for observed and estimated faults BCP given by proposed models and compared with some existing model

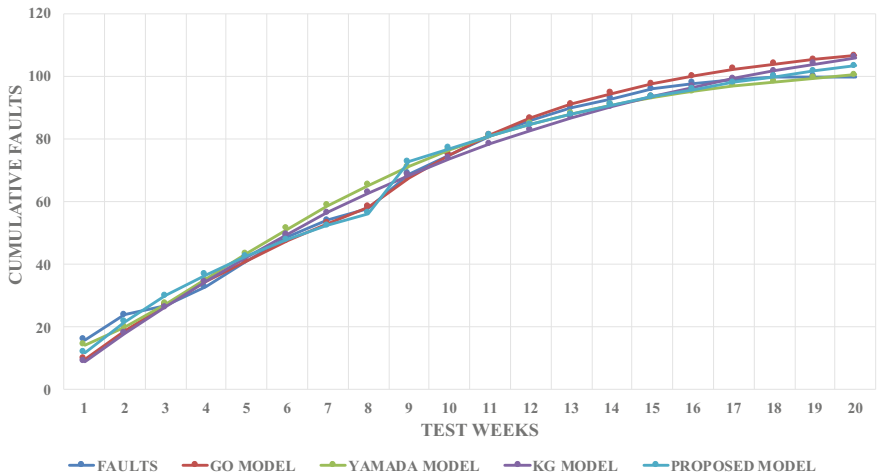


Fig. 5 A fitted curve for observed and estimated faults ACP given by proposed models and compared with some existing model

The result of the data analysis suggests that the extended version of the KG model is an effective model that enriches the data set in terms of predicting accuracy and beautifully incorporates the effect of change point under perfect debugging conditions.

5 Release Policy

With the completion of software testing phase, the software project/application is prepared for its release into the market rising to problem of optimal release policy. Protraction of testing phase implies detection of multiple faults, thus increasing the development cost, increase in the quality and delay in the release of the software project/application. The lag in release of the software may make the market restive, at the same time cause a loss to the business. On the other hand, if the testing phase is cut short, one can infer detection of a smaller number of faults implying decrease in development cost, poor quality and rapid release of the software. The accelerated release of the software may lead to loss of customers' trust towards the firm that might incur post-release high cost. In this study, one of the objectives is to minimize software outlay with respect to a desired level of reliability, thereby obtaining the optimal software release time. The outlay parameters used to illustrate release policy is given as follows,

O_1 : Outlay of fault detection BCP in the testing environment ($O_1 > 0$)

O'_2 : Outlay of fault detection ACP in the testing environment ($O'_2 > 0$)

O_2 : Outlay of fault detection in the operational environment ($O_2 > O'_2$)

O_3 : Outlay of testing at arbitrary time point

T: stopping or release time of software project/application.

The expected outlay function without change point comprises of expense acquired by the software firm in order to detect fault BCP, during the operating phase and lastly, an overall expense per unit time before the software project/application's release time, which is given by,

$$O(T) = O_1 * X(T) + O_2 * (X - X(T)) + O_3 * T \quad (10)$$

The expected outlay function with change point is composed of expense acquired by the software firm in order to detect faults BCP, ACP, during the operating phase and lastly, an overall expense per unit time before the software project/application's release time, which is given as,

$$O(T) = O_1 * X_1(\tau) + O'_2 * (X_2(T) - X_1(\tau)) + O_2 * (X - X_2(T)) + O_3 * T \quad (11)$$

where, $X_1(t)$ and $X_2(t)$ are mean value fault detection function without and with change point (τ) respectively. Henceforth, an optimization policy can be mathematically formulated as following,

$$\begin{aligned} &\text{Minimise } O(T) \\ &\text{subject to } Re(T) \geq R^* \end{aligned} \tag{12}$$

where, $Re(T)$ is the reliability function and R^* is the desiderated level of reliability at the release time of software project/application's ($0 < R^* < 1$). The software project developer establishes the desiderated reliability for the project in accordance to the market requirement.

5.1 Numerical Illustration

The formulated optimization problem is solved using MATLAB where outlay parameters are assumed on the basis of past literature (Aggarwal et al. 2015; Shrivastava and Sachdeva 2019). In order to implement the optimization problem given by (12), we have considered tandem dataset, computed parameters of the Eqs. (4), (5), (8) and (9) with the help of SPSS and given by the Table 1. The outlay parameters are assumed to be as following,

$$O_1 = 40, O'_2 = 50, O_2 = 100, O_3 = 20$$

With the help of MATLAB and Eq. (12), the expected release time and reliability is computed for the four models and given by the Table 4.

The optimal release time for model 1a is 28.02 weeks with reliability 0.9823 and expected cost is 5111.01, and with change point (model 1b), the release time is 30.28 weeks at 0.9638 reliability and total expected cost being 9116.40. For model 2a, the optimal release time is 30.53, reliability attained is 0.9758 with the expected cost being 5340.80, however with change point, the release time decreases to 28.33 weeks and reliability reached is 0.9532 with the cost of 8901.80.

Table 4 Expected release time and reliability of a software project/application

Models	T	O(T)	Re(T)
Model 1a	28.02	5111.01	0.9823
Model 1b	30.28	9116.4	0.9638
Model 2a	30.53	5340.8	0.9758
Model 2b	28.33	8901.8	0.9532

5.2 Sensitivity Analysis

The expenses involved in testing phase may vary due to various external factors like skills of developers, testing efforts, resource etc. Hence, one might be interested to know which cost parameter plays an important role to improvise the reliability for a given software project/application. Sensitivity analysis is a tool that aids in the decision making of optimal input parameters in order to obtain improvised output (Li et al. 2010). The uncertainty in failure data fails to comprehend the exact value of the outlay parameters. This methodology permits the management to evaluate the change in parameters over reliability, to assess testing strategy, to condense cost values, to attain optimal resource apportionments (Castillo et al. 2008).

For the given study, software reliability growth models given by Eqs. (4), (5), (8) and (9) have been considered, whose parameters are computed using Tandem failure data sets and with the help of SPSS. Considering a desired reliability level to be fixed at 0.95, the outlay parameters (O_1, O'_2, O_2, O_3) are altered by -10 and 10% , while monitoring its effect on the reliability, total expected cost as well as release time (given by Table 5). The sensitivity of outlay parameters is analyzed by using formula given by Li and Pham (2017). The relative changes observed with respect to varying outlay parameters is given by Table 6.

For different values of O_1, O'_2, O_2, O_3 , the altered release time, cost as well as reliability level are generated. Altering values of outlay parameters gives a guidance to the management to strike a balance between the cost and reliability and obtain the optimal release time. The sensitivity analysis of the stated release policy resulted in a very close release time, but in real time situation, the policy implemented by the management depends on the cost under consideration, SRGM adopted to describe the fault process, aim of the release plan and lastly constraints to be taken care of.

From Fig. 6a, it can be clearly observed that a 10% increase in cost O_1 , release time decreases by 1.46% , overall cost increases by 8.52% and a marginal decrease of 0.1% has been noted. However, when cost O_2 is increased by 10% , release time increases by 3.26% , that has 0.3% and 0.2% effect on total cost and reliability respectively. On the other hand, with a 10% increase in O_3 , the release time declines by 2% , increasing cost by 1% and reliability also declines by a small fraction. The respective trend was observed exactly opposite when the costs O_1, O_2, O_3 were decreased by 10% for model 1(a) as depicted by Fig. 6b.

From Fig. 6c, with an increase of 10% in cost O_1 , release time, expected cost and reliability increases by 2.34% , 2.79% and 0.2% respectively. However, with an increase of 10% in cost O'_2 , release time and reliability declines by 2.29 and 0.2% , while the cost shoots up by 6.15% . On the other hand, with an increase of 10% in cost O_2 , release time is majorly affected by 3.74% , but a marginal change of 0.4% and 0.3% on cost and reliability is observed. Lastly, with an increase of 10% in O_3 , a decline of 2.7% and 0.2% is observed in release time and reliability along with increase in cost by 0.6% . The respective trends were seen exactly opposite when cost O_1, O'_2, O_2, O_3 were decreased by 10% for model 1(b) as depicted by Fig. 6d except when cost O_1 decreased by 10% , the expected cost increased by 2.8% .

Table 5 Sensitivity analysis for SRGM with respect to outlay parameters

		Model 1a				Model 1b				Model 2a				Model 2b			
		T	O(T)	Re(T)	T	O(T)	Re(T)	T	O(T)	Re(T)	T	O(T)	Re(T)	T	O(T)	Re(T)	
O1	36	28.40	4675.27	0.9834	29.09	9372.58	0.9593	31.03	4895.10	0.9772	31.13	8656.89	0.95				
	40	28.02	5111.01	0.9823	30.28	9116.40	0.9638	30.53	5340.80	0.9758	28.33	8901.80	0.9532				
	44	27.61	5546.22	0.9811	30.99	9371.07	0.9661	30.00	5785.77	0.9742	31.13	9160.19	0.95				
O'2	45	-	-	-	30.88	8554.45	0.9658	-	-	-	32.19	8441.06	0.9536				
	50	-	-	-	30.28	9116.40	0.9638	-	-	-	28.33	8901.80	0.9532				
	55	-	-	-	29.58	9676.98	0.9613	-	-	-	31.13	9509.61	0.95				
O2	90	26.94	5089.57	0.9788	28.78	9071.12	0.958	29.14	5310.71	0.9713	31.13	8919.64	0.95				
	100	28.02	5111.01	0.9823	30.28	9116.40	0.9638	30.53	5340.80	0.9758	28.33	8901.80	0.9532				
	110	28.93	5129.10	0.9849	31.41	9156.31	0.9674	31.72	5366.52	0.979	33.24	9028.07	0.9568				
O3	18	28.64	5054.36	0.9841	30.94	9055.15	0.966	31.34	5278.92	0.978	32.31	8912.47	0.954				
	20	28.02	5111.01	0.9823	30.28	9116.40	0.9638	30.53	5340.80	0.9758	28.33	8901.80	0.9532				
	22	27.45	5166.47	0.9805	29.65	9176.29	0.9615	29.80	5401.10	0.9735	31.13	9038.08	0.95				

Table 6 Relative change observed for respective model with respect to outlay parameters

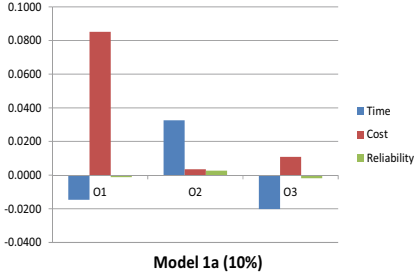
		O1		O'2		O2		O3	
		-10%	10%	-10%	10%	-10%	10%	-10%	10%
Model 1a	T	0.0137	-0.0146	0.0000	0.0000	-0.0387	0.0326	0.0223	-0.0202
	O(T)	-0.0853	0.0852	0.0000	0.0000	-0.0042	0.0035	-0.0111	0.0109
	Re(T)	0.0011	-0.0012	0.0000	0.0000	-0.0036	0.0026	0.0018	-0.0018
Model 1b	T	-0.0392	0.0234	0.0199	-0.0229	-0.0496	0.0374	0.0220	-0.0207
	O(T)	0.0281	0.0279	-0.0616	0.0615	-0.0050	0.0044	-0.0067	0.0066
	Re(T)	-0.0047	0.0024	0.0021	-0.0026	-0.0060	0.0037	0.0023	-0.0024
Model 2a	T	0.0163	-0.0173	0.0000	0.0000	-0.0456	0.0390	0.0266	-0.0239
	O(T)	-0.0835	0.0833	0.0000	0.0000	-0.0056	0.0048	-0.0116	0.0113
	Re(T)	0.0014	-0.0016	0.0000	0.0000	-0.0046	0.0033	0.0023	-0.0024
Model 2b	T	0.0988	0.0988	0.1363	0.0988	0.0988	0.1732	0.1405	0.0988
	O(T)	-0.0275	0.0290	-0.0518	0.0683	0.0020	0.0142	0.0012	0.0153
	Re(T)	-0.0034	-0.0034	0.0004	-0.0034	-0.0034	0.0038	0.0008	-0.0034

From Fig. 6e, when O_1 increases by 10%, release time decreases by 1.7%, total cost does up by 8.3% and reliability decreases by 0.2%. While O_2 increases by 10%, release time, cost and reliability also increases by 3.9%, 0.4% and 0.3% respectively. Lastly, with a 10% increase in O_3 , release time and reliability decreases by 2.3 and 0.2%, on the contrary, the expected cost increases by 1.1%. When the costs O_1 , O_2 , O_3 decreases by 10%, an exactly opposite trend is observed which is depicted by Fig. 6f for model 2a.

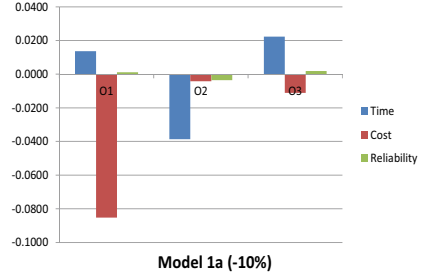
From Fig. 6g, when cost O_1 , O'_2 , O_2 , O_3 increases by 10%, release time increase significantly by 9.8%, 9.8%, 17.3%, 9.8% respectively, while the expected cost also rises by 2.9%, 6.8%, 1.4% and 1.5%, but reliability declines by 0.3% for cost O_1 , O'_2 , and O_3 , on the other hand increases by 0.3% for O_2 . From Fig. 6h, it is observed that when cost O_1 , O'_2 , O_2 , O_3 decreases by 10%, release time increases by 9.8%, 13.6%, 9.8% and 14.1% respectively, the total cost declines for O_1 and O'_2 by 2.7% and 0.5%, however increases in the case of O_2 and O_3 by 0.2% and 0.1%. Moreover, reliability decreases by 0.3% when O_1 and O_2 decreases by 10% but increases by 0.04% and 0.08%, when cost O'_2 and O_3 decrease by 10%.

6 Conclusion

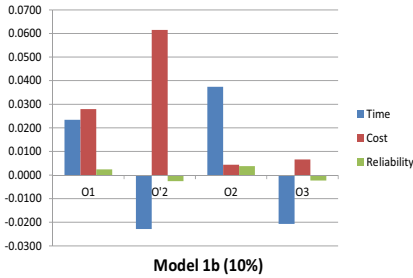
In this chapter, a software reliability growth model has been comprehended with respect to software project/application's characteristics under a perfect debugging environment. The stated characteristic has been considered as a constant as well as a power function of time along with assessing the model without and with change point. The parameters are estimated with the help of SPSS and comparison has been



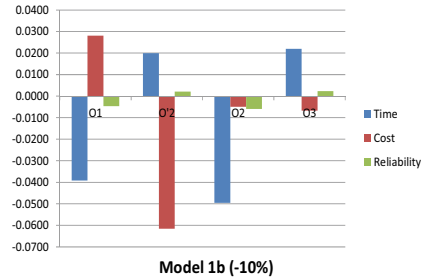
(a)



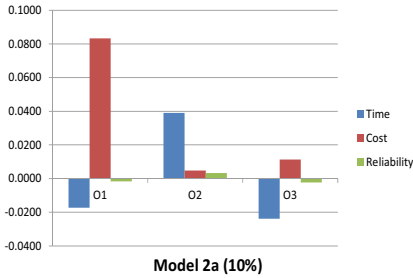
(b)



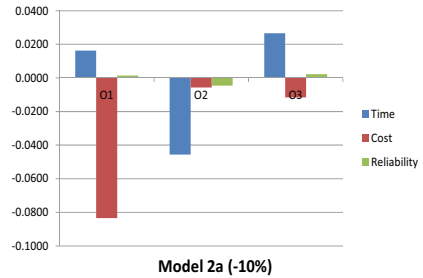
(c)



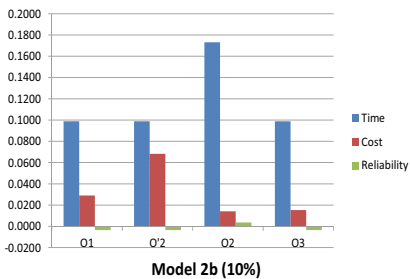
(d)



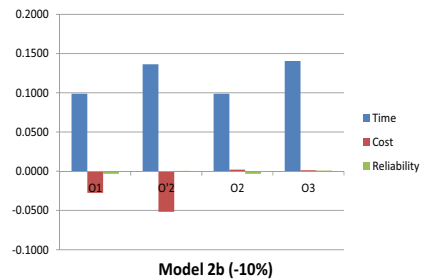
(e)



(f)



(g)



(h)

◀**Fig. 6** Relative change in release time, cost and reliability with respect to **a** Model 1a when there is 10% increase in costs O_1 , O_2 and O_3 **b** Model 1a when there is 10% decrease in costs O_1 , O_2 and O_3 **c** Model 1b when there is 10% increase in costs O_1 , O'_2 , O_2 and O_3 **d** Model 1b when there is 10% decrease in costs O_1 , O'_2 , O_2 and O_3 **e** Model 2a when there is 10% increase in costs O_1 , O_2 and O_3 **f** Model 2a when there is 10% decrease in costs O_1 , O_2 and O_3 **g** Model 2b when there is 10% increase in costs O_1 , O'_2 , O_2 and O_3 **h** Model 2b when there is 10% decrease in costs O_1 , O'_2 , O_2 and O_3

made with extant models from literature. An optimization formulation has also been proposed for all the four models with respect to desiderated reliability and with the aim of minimizing total expected outlay. Lastly, a sensitivity analysis has been provided to validate the optimal result obtained.

Looking at the results, one can state that model with change point in both cases when the software project characteristic is constant or a power function is an efficient tool in terms of providing a realistic outlook and in comparison to other previous models.

7 Future Scope

The limitation identified for the proposed model is that it has been validated on a small dataset and the under an ideal testing environment. In order to generalize the results, using large datasets would authenticate the efficiency of the proposed model. Further research work can be implemented by considering various attributes like testing effort, imperfect debugging, resource allocation etc.

References

- Aggarwal AG, Nijhawan N, Kapur P (2015) A discrete SRGM for multi-release software system with imperfect debugging and related optimal release policy. In: 2015 international conference on futuristic trends on computational analysis and knowledge management (ABLAZE). IEEE, pp 186–192
- Castillo E, Mínguez R, Castillo C (2008) Sensitivity analysis in optimization and reliability problems. *Reliab Eng Syst Saf* 93(12):1788–1800
- Chatterjee S, Shukla A (2017) An ideal software release policy for an improved software reliability growth model incorporating imperfect debugging with fault removal efficiency and change point. *Asia-Pac J Oper Res* 34(03):1740017
- Chatterjee S, Nigam S, Singh JB, Upadhyaya LN (2012) Effect of change point and imperfect debugging in software reliability and its optimal release policy. *Math Comput Model Dyn Syst* 18(5):539–551
- Chiu K-C, Huang Y-S, Lee T-Z (2008) A study of software reliability growth from the perspective of learning effects. *Reliab Eng Syst Saf* 93(10):1410–1421
- Goswami D, Khatri SK, Kapur R (2007) Discrete software reliability growth modeling for errors of different severity incorporating change-point concept. *Int J Autom Comput* 4(4):396–405

- Huang C-Y (2005a) Cost-reliability-optimal release policy for software reliability models incorporating improvements in testing efficiency. *J Syst Softw* 77(2):139–155
- Huang C-Y (2005b) Performance analysis of software reliability growth models with testing-effort and change-point. *J Syst Softw* 76(2):181–194
- Huang C-Y, Luo S-Y, Lyu MR (1999) Optimal software release policy based on cost and reliability with testing efficiency. In: Proceedings of the twenty-third annual international computer software and applications conference (Cat. No. 99CB37032). IEEE, pp 468–473
- Inoue S, Yamada S (2011) Software reliability growth modeling frameworks with change of testing-environment. *Int J Reliab Qual Saf Eng* 18(04):365–376
- Inoue S, Yamada S (2018) Markovian software reliability modeling with change-point. *Int J Reliab Qual Saf Eng* 25(02):1850009
- Inoue S, Hayashida S, Yamada S (2013) Extended hazard rate models for software reliability assessment with effect at change-point. *Int J Reliab Qual Saf Eng* 20(02):1350009
- Jeske DR, Zhang X (2005) Some successful approaches to software reliability modeling in industry. *J Syst Softw* 74(1):85–99
- Kapur P, Garg R (1989) Cost-reliability optimum release policies for a software system under penalty cost. *Int J Syst Sci* 20(12):2547–2562
- Kapur P, Garg R (1992) A software reliability growth model for an error-removal phenomenon. *Softw Eng J* 7(4):291–294
- Kapur PK, Singh VB, Anand S, Yadavalli VSS (2008a) Software reliability growth model with change-point and effort control using a power function of the testing time. *Int J Prod Res* 46(3):771–787
- Kapur P, Jha P, Singh V (2008b) On the development of discrete software reliability growth models. In: Handbook of performability engineering. Springer, pp 1239–1255
- Kapur P, Aggarwal AG, Anand S (2009) A new insight into software reliability growth modeling. *Int J Perform Eng* 5(3):267–274
- Kapur PK, Garg RB, Aggarwal AG, Tandon A (2009a) General framework for change point problem in software reliability and related release time problem. *Int J Reliab Qual Saf Eng* 16(06): 567–579
- Kareer N, Kapur P, Grover P (1990) An S-shaped software reliability growth model with two types of errors. *Microelectron Reliab* 30(6):1085–1090
- Ke S-Z, Huang C-Y, Peng K-L (2014) Software reliability analysis considering the variation of testing-effort and change-point. In: Proceedings of the international workshop on innovative software development methodologies and practices, pp 30–39
- Lai R, Garg M (2012) A detailed study of NHPP software reliability models. *J Softw* 7(6):1296–1306
- Lai R, Garg M, Kapur PK, Liu S (2011) A study of when to release a software product from the perspective of software reliability models. *JSW* 6(4):651–661
- Li Q, Pham H (2017) NHPP software reliability model considering the uncertainty of operating environments with imperfect debugging and testing coverage. *Appl Math Model* 51:68–85
- Li X, Xie M, Ng SH (2010) Sensitivity analysis of release time of software reliability models incorporating testing effort with multiple change-points. *Appl Math Model* 34(11): 3560–3570
- Nagaraju V, Fiondella L (2017) A single changepoint software reliability growth model with heterogeneous fault detection processes. In: 2017 Annual reliability and maintainability symposium (RAMS). IEEE, pp 1–6
- Okumoto K, Goel AL (1979) Optimum release time for software systems based on reliability and cost criteria. *J Syst Softw* 1:315–318
- Pachauri B, Kumar A, Dhar J (2013) Modeling optimal release policy under fuzzy paradigm in imperfect debugging environment. *Inf Softw Technol* 55(11):1974–1980
- Park J, Baik J (2015) Improving software reliability prediction through multi-criteria based dynamic model selection and combination. *J Syst Softw* 101:236–244
- Pham H (2016) A generalized fault-detection software reliability model subject to random operating environments. *Vietnam J Comput Sci* 3(3):145–150
- Roy P, Mahapatra G, Dey K (2014) An NHPP software reliability growth model with imperfect debugging and error generation. *Int J Reliab Qual Saf Eng* 21(02):1450008

- Sharma K et al (2010) Selection of optimal software reliability growth models using a distance based approach. *IEEE Trans Reliab* 59(2):266–276
- Shrivastava AK, Sachdeva N (2019) Generalized software release and testing stop time policy. *Int J Qual Reliab Manag* 37(6/7):1087–1111
- Shrivastava AK, Kumar V, Kapur PK, Singh O (2020) Software release and testing stop time decision with change point. *Int J Syst Assur Eng Manag* 11(2):196–207
- Singh V, Kapur P, Kumar R (2010) Developing S-shaped reliability growth model for open source software by considering change point. In: *Proceedings of the IASTED international conference*, vol 677, no 103, p 256
- Wood A (1996) Predicting software reliability. *Computer* 29(11):69–77
- Yamada S, Ohba M, Osaki S (1983) S-shaped reliability growth modeling for software error detection. *IEEE Trans Reliab* 32(5):475–484
- Zhao M (1993) Change-point problems in software and hardware reliability. *Commun Stat Theory Methods* 22(3):757–768
- Zhu M, Pham H (2018) A software reliability model incorporating martingale process with gamma-distributed environmental factors. *Ann Oper Res* 1–22
- Zhu M, Pham H (2020) A generalized multiple environmental factors software reliability model with stochastic fault detection process. *Ann Oper Res* 1–22