



OPEN

Software belief reliability growth model incorporating change point and imperfect debugging based on uncertain differential equation approach

Neelam Sharma¹, Vijay Kumar^{1✉}, Arfat Ahmad Khan², Prashant Johri³, Farman Ali^{4✉} & Ahmad Ali AlZubi⁵

Many software reliability growth models (SRGMs) have been proposed by researchers within the context of probability theory to estimate software reliability, remaining number of faults and optimal release time. The Fault Detection Rate (FDR) may vary because of changes in testing strategies. Due to lack of knowledge of software code, the testing team might be unable to rectify the detected faults thereby introducing new faults during the fault correction process. The debugging process is imperfect due to factors like human error, insufficient testing and complex codes resulting in epistemic uncertainty. In this paper, we have proposed a new software belief reliability growth model (SBRGM) using uncertain differential equations to deal with epistemic uncertainty effectively. We have incorporated imperfect debugging and change point based on the approach of belief reliability theory, making this model more accurate as compared to some of the previously developed models. Model parameters estimation methodology is derived using the least square method and Python version 3.10. Calculation of change point is done using empirical data analysis based on the First principle of Derivatives. Three real data sets have been used to validate the proposed model. This research contributes to being more flexible and realistic in dealing with epistemic uncertainty effectively as compared to conventional approaches.

Keywords Software belief reliability, SBRGM, Uncertainty theory, Change point, Imperfect debugging, Uncertain differential equations

In today's digitalised era, software makes a substantial contribution to the advancements of digital infrastructure from data analytics to customer relationship management systems, and e-commerce platforms to mobile applications. In a heavily competitive market, companies often face immense pressure to keep up with the set standards. Hence, in the Software Development Life Cycle (SDLC), testing phase becomes important and essential. Software engineers are usually supposed to perform their tasks within the allocated time to maintain a competitive edge. Software failures are the result of software errors that could lead to serious aftermaths in our day-to-day activities. A few examples from the past are as described¹: One notable incident occurred on September 17, 1991, when 10 million telephone users were affected for nearly 9 h due to a power failure at AT&T's switching centre in New York. The deletion of three bits of code in a software upgrade and failed testing were the primary reasons for the software failure before its release in the market. On October 26, 1992, a similar incident occurred when London's Ambulance Service's computer-based dispatch system, which had earlier been used to handle 5000 requests per day in emergencies, malfunctioned right after its deployment. Many critically ill patients experienced intense repercussions due to software breakdown, highlighting the risk of system failures.

¹Department of Mathematics, Amity Institute of Applied Sciences, Amity University Uttar Pradesh, Noida, India.

²Department of Computer Science, College of Computing, Khon Kaen University, Khon Kaen 4002, Thailand.

³School of Computer and Technology, Galgotias University, Greater Noida, India. ⁴Department of Applied AI, School of Convergence, College of Computing and Informatics, Sungkyunkwan University, Seoul 03063, South Korea.

⁵Department of Computer Science and Engineering, College of Applied Studies, King Saud University, Riyadh, Saudi Arabia. ✉email: vijay_parashar@yahoo.com; farman0977@skku.edu

Not long ago, as mentioned in an article² on 19 July 2024, a flawed configuration upgrade patch was installed by CrowdStrike for its Falcon sensor software operating on Windows PCs and servers. This faulty upgrade forced systems to enter into a recovery mode or boot loop, resulting in invalid page faults. CrowdStrike Falcon's close integration with Microsoft Windows kernel led to a Windows system crash. The error was within a sensor configuration in CrowdStrike Falcon. Consequently, approximately 8.5 million systems collapsed and went down and could not restart and reboot effectively, resulting in a heavy economic and financial loss. Since software failure can result in monetary and financial disruptions or sometimes human casualties, the software's reliability is a necessity^{3,4}. To guarantee failure-free software and to determine optimal launch time, precise reliability estimation is needed before the software is installed. Hence, software reliability has become one of the major focuses in the current situation⁵. Therefore, SRGMs are considered a tool for discussing software reliability, which estimates and determines software reliability and other related metrics effectively⁶⁻⁸.

Many Software Reliability Growth Models have been developed by prominent researchers to evaluate software reliability with different suppositions and approaches based on probability theory⁹⁻¹². Yamada et al.¹³ for the very first time, applied a stochastic differential equation (SDE) to discuss the number of faults. Subsequently, Kapur et al.¹⁴ developed a generalized Erlang model based on SDE, featuring a logistic error detection function. Li and Pham¹⁵ incorporated factors such as learning behaviour and considered environmental uncertainties to enhance the accuracy of reliability assessments. During the testing phase, due to certain factors like resource allocation and testing knowledge, fault detection rate can be discontinuous and may change at a certain time point, which is referred to as a change point¹⁶. A generalized model has been developed incorporating a change point and logistic testing effort¹⁷.

Because of the intricacy and insufficient knowledge of software system, the testing team might not eliminate detected faults, and additional faults/bugs can also emerge in correction process known as imperfect debugging. Kapur et al.¹⁸ put forward the NHPP-based model taking into account imperfect debugging process. Chatterjee and Shukla [2020] considered imperfect debugging and change point together in the model. Several factors, like skill set, testing productivity, budget allocation, testing methodologies, and approaches, all of which may be uncertain, can affect the testing process²⁰. Probability theory is a mathematical framework to describe this uncertainty. As the software scale broadens, the faults identified during the testing period increase, and the number of bugs identified and rectified by a single debugging process becomes significantly less compared to the total number of faults at the start of the testing process. Various researchers discussed probability theory as advantageous in terms of quantification, risk assessment, and resource allocation in reliability growth models to deal with the uncertainty in the testing phase^{9,19,21,22}.

Complex software systems consist of various modules and correlations. Uncertainty can take place when faults are associated with other codes of the system. By identifying and managing this uncertainty, software developers and testers can improve fault detection, responses and increase overall software reliability. Software's fault is based on the fundamental rules of logic and code, behaviour, psychology, testing environment and many more. This uncertainty is mainly due to the cognitive capabilities of software engineers, coding, testing, usage and other situations. The classification of uncertainties includes two types: aleatory and epistemic. Epistemic uncertainty takes place when there is a lack of information, aleatory uncertainty stems from intrinsic inconsistency. Probability theory is adequate for examining aleatory uncertainty but inadequate for measuring epistemic uncertainty as it presumes that complete insights are accessible, see^{20,23-26}.

Software faults include epistemic uncertainty that cannot be illustrated accurately by probability theory^{24,25}. To handle epistemic uncertainty in software reliability models, many SRGMs have been established by various scholars²⁷⁻³⁰ under the framework of alternative approaches like Bayesian and fuzzy methods, but these methodologies lack accuracy. Bayesian methods consider the framework of beliefs about a system's reliability in its modelling process. On the other hand, Zhang et al.³² discussed that fuzzy methods describe uncertainty with fuzzy sets and linguistic variables but yield non-intuitive outcomes, for example, the sum of unreliability and reliability is not one.

To address epistemic uncertainty effectively in SRGMs, it is essential to come up with other possible methods. Uncertainty theory was introduced by Liu³² as a different approach to describe epistemic uncertainty. Uncertainty theory incorporates both epistemic and aleatory uncertainty, providing a comprehensive and systematic framework of software reliability. In comparison to traditional stochastic processes, the Liu process is a stochastic process rooted in uncertainty theory, as it considers increments as normal uncertain variables. The Liu process provides more suitable models for software reliability systems, as the increments in Liu process are independent. Researchers^{24-26,33,34} have shown that SBRGMs based on uncertain differential equations have performed better than existing famous models that were based on probability theory. Liu and Kang²⁴ proposed an SBRGM to evaluate properties like mean time between failures, belief reliable time and belief reliability of software reliability, incorporating imperfect debugging based on the uncertainty theory. Liu et al.³⁵ have presented SBRGM based on the methodology and approach of belief reliability and uncertainty theory. This article did not consider that the debugging process is imperfect and new errors can also be introduced. Liu et al.²⁵ developed SBRGM considering change-point based on uncertainty theory and also estimated the software reliability. Garg et al.²⁶ developed an SBRGM based on uncertain differential equations, considering software patching and FDR and Fault correction rate as a two-step process and also investigated belief reliability to optimize the testing cost. Huong³⁶ developed a dynamic event-triggered control method offering an LMI-based solution for uncertain neural networks with time delays. Huong³⁷ introduced a mathematical model to control an uncertain active suspension system by addressing event-triggered finite-time guaranteed cost control. Also, Huong³⁸, in his work, designed a cost controller for an uncertain polytopic fractional-order system with time delays.

Novelty

The novelty of this research is that this is the first one to propose SBRGM incorporating change point and imperfect debugging based on uncertainty theory, which produces better results as compared to probability theory.

Upon further investigation of existing literature, we discovered that previous models based on uncertainty did not consider change-point and imperfect debugging altogether. In fact, in more real-world scenarios, the software is influenced by many things, including the operating environment, testing methodology, and the allocation of resources. To bring our model more in line with reality, we have developed an SBRGM that integrates the change point with imperfect debugging based on uncertain differential equations. The paper provides primary contributions as follows:

- This paper proposes an SBRGM incorporating change point and imperfect debugging altogether.
- A methodology for estimating unknown parameters based on the least squares method is developed, and a code to estimate uncertainty variables present in the model is developed using Python version 3.10.
- A methodology for estimating change point under the framework of empirical data analysis based on the First principle of Derivatives has been proposed.
- The developed model is compared with other well-established models to show its effectiveness and robustness.

The later segments of the paper are arranged subsequently. The basic preliminaries, notations, and assumptions used in the proposed model are described in Sect. “Novelty”. The model formulation and development are represented in Sect. “Basic preliminaries”. Section “Elief reliability evaluation” discusses the techniques of Parameter estimation of unknown parameters. Section “Concept of -path” provides a numerical illustration to demonstrate the model’s applicability with accuracy and precision. The managerial impact of the proposed model is discussed in Sect. “Notations used”. Finally, the conclusion and suggestions for future research scope are summarized in Sect. “Assumptions”.

Basic preliminaries

Let us review some fundamental concepts regarding uncertainty theory³².

Definition 2.1 As discussed by Liu³², a measurable space is denoted by (Γ, \mathcal{L}) , where \mathcal{L} is representing a σ -algebra defined over Γ , and Γ being a nonempty set. An uncertain measure $\mathcal{M}: \mathcal{L} \rightarrow [0, 1]$ is a set function defined on σ -algebra that meets the below-listed fundamentals.

Axiom 1 It states that the measure of the universal set Γ is equal to one. In mathematical terms, this is.

represented as: $\mathcal{M}\{\Gamma\} = 1$. This is known as the Normality Axiom.

Axiom 2 For any event Λ , the sum of the measure of Λ and its complement Λ^c is equal to one.

Mathematically, it is: $\mathcal{M}\{\Lambda\} + \mathcal{M}\{\Lambda^c\} = 1$. This is stated as Duality Axiom.

Axiom 3 For every countable sequence of events $\Lambda_1, \Lambda_2, \dots$, mathematical expression is:

$$\mathcal{M}\left\{\bigcup_{i=1}^{\infty} \Lambda_i\right\} \leq \sum_{i=1}^{\infty} \mathcal{M}\{\Lambda_i\}$$

i.e., the measure of union of these events is less than or equal to the sum of measures of the individual events.

This axiom is referred to as Subadditivity Axiom.

Axiom 4 Let $(\Gamma_k, \mathcal{L}_k, \mathcal{M}_k)$ be uncertainty spaces for $k = 1, 2, \dots$, and Λ_k are arbitrarily.

chosen events from \mathcal{L}_k for $k = 1, 2, \dots$, respectively. Then the product uncertain measure \mathcal{M} is an uncertain measure that satisfies the inequality below:

$$\mathcal{M}\left\{\prod_{k=1}^{\infty} \Lambda_k\right\} \leq \bigwedge_{k=1}^{\infty} \mathcal{M}_k\{\Lambda_k\}$$

This expression explains that the measure of the product of events Λ_k is less than or equal to minimum of measures of individual events Λ_k . This is known as the Product Axiom.

Definition 2.2 As introduced by Liu³⁹, Liu’s process C_t is an uncertain process that addresses the following three key conditions.

1) Continuity and Initial Condition: Almost all sample paths are Lipschitz continuous and initially at.

$t=0, C_0 = 0$.

2) Increment Properties: The increments of C_t are independent, which means the behaviour of.

process over disjoint intervals is statistically independent.

3) Distribution of Increments: For any interval t , the increment $C_{l+t} - C_l$ follows a normal uncertain distribution with an expected value of zero and a variance t^2 .

Definition 2.3 Let us define an uncertain process^{32,39} as Z_t and a Liu process as C_t . For closed interval $[u, v]$, the partition is as follows, with $u = t_1 < t_2 < \dots < t_k = v$, denote $\Delta = \max_{1 \leq i \leq k} |t_{i+1} - t_i|$. Then, Liu's integral of Z_t for C_t is given by Eq. (1) and denoted as:

$$\int_u^v Z_t dC_t = \lim_{\Delta \rightarrow 0} \sum_{i=1}^k Z_{t_i} (C_{t_{i+1}} - C_{t_i}) \quad (1)$$

under the assumption that a limit exists and finite.

Definition 2.4 Liu³² defined C_t as Liu process (a fundamental concept in uncertainty theory) and denoted f and g as measurable functions. An uncertain differential equation can then be defined in terms of these functions and Liu process by Eq. (2) as:

$$dZ_t = f(t, Z_t)dt + g(t, Z_t)dC_t \quad (2)$$

with an initial value Z_0 , which indicates that the solution Z_t complies with the associated uncertain integral equation as described by Eq. (3) as:

$$Z_t = Z_0 + \int_0^t f(l, Z_l)dl + \int_0^t g(l, Z_l)dC_l \quad (3)$$

Elief reliability evaluation

The proposed model determines the count of detected faults in the testing phase. Software reliability improves/enhances as number of detected defects escalates/rises. In this section, Software belief reliability of the proposed model derived from the belief reliability theory has been discussed as presented below:

Let t_f denote the time required for total detected faults to achieve the predefined critical threshold f where $f \in (0,1)$. To determine the belief degree to which software testing could be stopped and released at the time t_f ,

$\tau_f \leq t_f$, belief reliability is given by $RB_f(t) = \mathcal{M}\{\tau_f \leq t\}$.
where,

$$\tau_f = \inf\{t \geq 0 | S(t) \geq fN\} \text{ and } \mathcal{M} \text{ is Uncertain measure}$$

As τ_f represents the first hitting of the solution of an uncertain differential equation. The definition of belief reliability is given by some researchers^{24–26} as:

$$RB_f = 1 - \inf\{\alpha | \sup_{0 \leq s \leq t} \Phi_s^{-1}(\alpha) \geq fN\} \quad (4)$$

where, Φ_s^{-1} is inverse function for belief reliability distribution. Also, as $S(t)$ increases, $\Phi_t^{-1}(\alpha)$ increases for particular α . Hence, belief reliability can be determined as:

$$RB_f = 1 - \inf\{\alpha | \Phi_s^{-1}(\alpha) \geq fN\} \quad (5)$$

where, $\Phi_t^{-1}(\alpha) = E(S(t)) + \left[\frac{\sqrt{3}SD(S(t))}{\pi} \right] \ln\left(\frac{\alpha}{1-\alpha}\right)$, $\alpha \in (0,1)$, $E(S(t))$ and $SD(S(t))$ is expectation and standard deviation of random variable, respectively.

Concept of α -path

In uncertain differential equations, the notion of α -path refers to a specific trajectory or solution curve dependent upon parameter α , symbolizing the uncertainty in equations. The α -path method investigates the behaviour of uncertain differential equations by encompassing a range of possible outcomes represented by different values of α . Each value of α corresponds to a specific path of the system of uncertain differential equations. By outlining the graph of numerical solutions for a range of values of α , we can depict the α -paths, which demonstrates how the system's behaviour changes as the level of uncertainty varies.

The α -path ($0 < \alpha < 1$) of an uncertain differential equation is given below in Eq. (6)

$$dS_t = f(t, S_t)dt + g(t, S_t)dC_t \quad (6)$$

with an initial value, S_0 , which is a deterministic function S_t^α with respect to t that satisfies the associated ordinary differential equation described in Eq. (7)

$$dS_t^\alpha = f(t, S_t^\alpha) + |g(t, S_t^\alpha)|\Phi^{-1}(\alpha)dt \quad (7)$$

where,

$\Phi^{-1}(\alpha)$: a standard normal uncertain variable that follows inverse uncertainty distribution, i.e.,

$$\Phi^{-1}(\alpha) = \frac{\sqrt{3}}{\pi} \ln \frac{\alpha}{1-\alpha}, \alpha \in (0,1)$$

Notations used

The notations used are given below:

N Total number of initial faults at time 't'.

$S(t)$ Cumulative number of detected faults till time 't'.

τ : change point.

$\lambda_1(t, \theta_1)$ Fault detection rate per remaining fault for $0 \leq t \leq \tau$.

$\lambda_2(t, \theta_2)$ Fault detection rate per remaining fault for $\tau < t \leq t$.

α_1 new fault introduction rate for $0 \leq t \leq \tau$.

α_2 new fault introduction rate for $\tau < t \leq t$.

C_t Liu process.

σ_1 Nonnegative constant denoting level of uncertainty for $0 \leq t \leq \tau$.

σ_2 Nonnegative constant denoting level of uncertainty for $\tau < t \leq t$.

Assumptions

In line with the models^{24,26,35}, we have taken into account the following presumptions:

- 1) A new bug/error can emerge during the debugging process, reflecting real-world situations where the correction of a fault can introduce new faults inadvertently²⁴. In this paper, we have considered this assumption to understand the concept of imperfect debugging in the proposed model.
- 2) The fault correction process occurs after the fault detection process, representing a sequential and fundamental aspect of software debugging³⁵. This sequential relationship is considered a core assumption in proposing SBRGMs based on uncertainty theory, where faults must be detected before correction.
- 3) The software operates within a finite lifecycle, incorporating development, testing, installation and maintenance within a fixed timeframe. According to Kapur et al.¹⁴, including this finite lifecycle into modeling frameworks aligns with practical realities and allows a more accurate interpretation of fault detection and correction. Here, the software cycle is divided into two intervals- before change point ($0 \leq t \leq \tau$) and after change point ($\tau < t \leq t$).

Model development

The software's life span is subdivided in two intervals in the model:

- Fault detection before change point, i.e. $0 \leq t \leq \tau$
- Fault detection after change point, i.e. $\tau < t \leq t$

Epistemic uncertainties are unavoidable in software faults. Therefore, we are applying the uncertain integral equation to model below equations³⁴:

$$S(t) = S(0) + \int \lambda(t, \theta) (N - S(l)) dl + \int \sigma dC_t \quad (10)$$

We can deduce the uncertain differential equation as:

$$dS(t) = \lambda(t, \theta) (N - S(t)) dt + \sigma dC_t, \quad S(0) = s(0) \quad (11)$$

Where, $\lambda(t, \theta)$ → fault detection rate till time 't'.

Now, we obtain the following integrals,

$$S(t) = \begin{cases} S(0) + \int_0^\tau \lambda_1(l, \theta_1) (N - S(l)) dl + \sigma_1 C_t, & 0 \leq t \leq \tau \\ S(\tau) + \int_\tau^t \lambda_2(l, \theta_2) (N - S(l)) dl + \sigma_2 C_t, & \tau < t \leq t \end{cases} \quad (12)$$

Now, we calculate the $S(t)$ i.e. cumulative number of detected faults and belief reliability.

In line with the assumptions^{15,25,26}, it is presumed that when detected faults are rectified at a given time 't', new faults may be introduced with the introduction rate $\alpha(t)$ given by Kapur et al.⁹:

$$\alpha(t) = \begin{cases} \alpha_1 & 0 \leq t \leq \tau \\ \alpha_2 & \tau < t \leq t \end{cases}$$

Taking into consideration the change point, we suppose exponential proportional factors to be $\lambda_1(t, \theta_1) = b_1$ and $\lambda_2(t, \theta_2) = b_2$.

Following the approach of Liu integral and uncertain differential equation given by Liu (2007), we obtain the integral equations:

$$dS(t) = \begin{cases} \left(\frac{b_1}{1-\alpha_1}\right)(N - S(t))dt + \sigma_1 dC_t, & 0 \leq t \leq \tau, S(0) = 0 \\ \left(\frac{b_2}{1-\alpha_2}\right)(N - S(t))dt + \sigma_2 dC_t, & \tau < t \leq \tau, S(\tau) = s(\tau) \end{cases} \quad (13)$$

where cumulative number of detected faults till time $t=0$ is denoted by $S(t)$ and $S(0) = 0$.

The developed model is a linear uncertain differential equation. The solution for $S(t)$ is given as:

$$S(t) = \begin{cases} N - N \exp\left(\frac{-b_1 t}{1-\alpha_1}\right) + \sigma_1 \exp\left(\frac{-b_1 t}{1-\alpha_1}\right) \int_0^t \sigma_1 \exp\left(\frac{b_1 l}{1-\alpha_1}\right) dC_l, & t \leq \tau \\ U_2(t) \left[S(\tau) + N \exp\left(\frac{b_2(t-\tau)}{1-\alpha_2}\right) - N + \int_\tau^t \frac{\sigma_2}{U_2(l)} dC_l \right], & t > \tau \end{cases} \quad (14)$$

where, $U_2(t) = \exp\left(\frac{-b_2(t-\tau)}{1-\alpha_2}\right)$

Then, $S(t)$ follows a normal uncertain distribution with the expected value is given by Eq. (15) as

$$E(S(t)) = \begin{cases} N - N \exp\left(\frac{-b_1 t}{1-\alpha_1}\right), & t \leq \tau \\ N - N \exp\left(\frac{-b_1 t}{1-\alpha_1} - \frac{b_2(t-\tau)}{1-\alpha_2}\right), & t > \tau \end{cases} \quad (15)$$

and the standard deviation as

$$SD(S(t)) = \begin{cases} (1 - \alpha_1) \frac{\sigma_1}{b_1} (1 - \exp\left(\frac{-b_1 t}{1-\alpha_1}\right)), & t \leq \tau \\ (1 - \alpha_1) \frac{\sigma_1}{b_1} \exp\left(-\frac{b_2(t-\tau)}{1-\alpha_2}\right) (1 - \exp\left(\frac{-b_1 t}{1-\alpha_1}\right)) + (1 - \alpha_2) \frac{\sigma_2}{b_2} (1 - \exp\left(\frac{-b_2(t-\tau)}{1-\alpha_2}\right)), & t > \tau \end{cases} \quad (16)$$

Correspondingly, the belief reliability distribution represents the number of detected faults till time 'τ' is less than x and is described as

$$\Phi_t(x) = \left(1 + \exp\left(\frac{\pi(E(S(t)) - x)}{\sqrt{3}SD(S(t))}\right)\right)^{-1} \quad (17)$$

$$\Phi_t^{-1}(\alpha) = E(S(t)) + \left[\frac{\sqrt{3}SD(S(t))}{\pi}\right] \ln\left(\frac{\alpha}{1-\alpha}\right), \quad \alpha \in (0,1) \quad (18)$$

where, $E(S(t))$ and $SD(S(t))$ are given in the Eqs. (15) and (16) respectively.

Belief reliability is given as:

$$RB_f(t) = \left(1 + \exp\left(\frac{\pi(fN - E(S(t)))}{\sqrt{3}SD(S(t))}\right)\right)^{-1} \quad (19)$$

With the help of Eqs. (6), (7), (13) and (14), the equations of α-paths for proposed model are given below:

$$S_t^\alpha = s(0) \exp\left(\frac{b_1 t}{1-\alpha_1}\right) + \left(1 + \frac{\sigma_1(1-\alpha_1)}{b_1} \cdot \frac{\sqrt{3}}{\pi} \ln\left(\frac{\alpha}{1-\alpha}\right)\right) \left(1 - \exp\left(\frac{-b_1 t}{1-\alpha_1}\right)\right), \quad t \leq \tau \quad (20)$$

and

$$S_t^\alpha = S(\tau) \exp\left(\frac{b_2(t-\tau)}{1-\alpha_2}\right) + \left(1 + \frac{\sigma_2(1-\alpha_2)}{b_2} \cdot \frac{\sqrt{3}}{\pi} \ln\left(\frac{\alpha}{1-\alpha}\right)\right) \left(1 - \exp\left(\frac{-b_2(t-\tau)}{1-\alpha_2}\right)\right), \quad t > \tau \quad (21)$$

Parameter estimation

We devised a two-step method for estimating parameters derived from the least square method and the moment estimation. This section discusses estimating unknown parameters by merging the concept of least squares and moment estimations.

Let us suppose $S(t)$ at a time, $t_1 < t_2 < \dots < t_n$ as $s(t_1), s(t_2), \dots, s(t_n)$, respectively. Also, assume that a change point occurs at a particular detection time.

To estimate unknown parameters N , θ_1 , and θ_2 by using the least squares method for each possible change point t_k , where, $k = 1, 2, \dots, n$, respectively. In other words, estimates N^* , θ_{k1}^* , θ_{k2}^* for unknown parameters N , θ_1 , and θ_2 . Optimal solutions are determined by minimizing objective functions

$$\min_{N, \theta_1, \theta_2} RSS_k = \sum_{i=1}^n (s(t_i) - E(S(t_i)))^2 \quad (22)$$

where,

$$E(S(t)) = \begin{cases} N - N \exp\left(\frac{-b_1 t}{1-\alpha_1}\right), & t \leq \tau_j \\ N - N \exp\left(\frac{-b_1 t}{1-\alpha_1} - \frac{b_2(t-\tau_j)}{1-\alpha_2}\right), & t > \tau_j, \end{cases} \quad (23)$$

$j = 1, 2, \dots, n$ respectively.

As per the definition, we obtain the difference forms of equations of SRGGM as

$$S(t_i) - S(t_{i-1}) = \frac{b_1}{1-\alpha_1} (N - S(t_{i-1})) (t_i - t_{i-1}) + \sigma_1 (C_{t_i} - C_{t_{i-1}}) , \quad t \leq \tau \quad (24)$$

and

$$S(t_i) - S(t_{i-1}) = \frac{b_2}{1-\alpha_2} (N - S(t_{i-1})) (t_i - t_{i-1}) + \sigma_2 (C_{t_i} - C_{t_{i-1}}) , \quad t > \tau \quad (25)$$

for $i = 1, 2, \dots, n-1$.

Unknown parameters σ_1 and σ_2 can be estimated as:

$$\sigma_1^2 = \frac{\sum_{j=1}^{n-1} (S(t_j) - S(t_{j-1}) - \left(\frac{b_1}{1-\alpha_1}\right) (N - S(t_j)) (t_j - t_{j-1}))^2}{\sum_{j=1}^{n-1} (t_j - t_{j-1})^2}, \quad t \leq \tau \quad (26)$$

and

$$\sigma_2^2 = \frac{\sum_{j=1}^{n-1} (S(t_j) - S(t_{j-1}) - \left(\frac{b_2}{1-\alpha_2}\right) (N - S(t_j)) (t_j - t_{j-1}))^2}{\sum_{j=1}^{n-1} (t_j - t_{j-1})^2}, \quad t > \tau \quad (27)$$

Change point calculation

Identifying the change point from a dataset includes a comprehensive analysis of fault detection trends over time in SRGGMs. The conventional approach to identifying the change point involves visually examining the cumulative count of detected faults across time, employing statistical approaches such as time series analysis or regression analysis, change point detection algorithms such as the Bayesian change point detection algorithm etc. The testing team generally possesses the knowledge of the change point in real software testing, represented as τ but in our given dataset, we do not have adequate information regarding the accurate value of τ . To address this, we have applied empirical data analysis for more exact information about τ . The failure increasing rate is given as shown below⁴⁰:

$$z'(t) = \lim_{\delta t \rightarrow 0} \frac{z(t + \delta t) - z(t)}{\delta t} \quad (28)$$

where,

- $z'(t)$: fault detection rate.
- $z(t)$: observed cumulative number of detected faults by time t .
- $z(t + \delta t)$: observed cumulative number of detected faults by time $t + \delta t$.

Numerical illustration

Within this segment, we validate our developed model on three real datasets and a comparative analysis is also done with existing well-known models.

Selection of SRGGMs

To calculate or predict the effectiveness of the developed model and techniques, a comprehensive literature review was carried out focusing on the mechanism and categorization of SRGGMs. The categorization was mainly centred on categories like failure rate models, change point models and Non-Homogeneous Poisson process. At last, 11 SRGGMs were taken into consideration as enumerated below in Table 1.

Comparison criteria

A model could be chosen as per its capability to reintegrate the predicted output and the observed failure data can be employed to estimate the software's future result. Within this segment, we validate our developed model on three real datasets and compare it with other existing models based on the sum of Mean Square Error (MSE)

No.	Model name	Mean value function
1	Delayed S-Shaped ⁴¹	$m(t) = a(1 - (1 + bt)e^{-bt})$
2	GO model ⁴²	$m(t) = a(1 - e^{-bt})$
3	Inflection S-shaped ⁴³	$m(t) = \frac{a(1 - e^{-bt})}{a(1 - \beta e^{-bt})}$
4	S-shaped change-point ⁹	$m(t) = \begin{cases} a(1 - b_1 t)e^{-b_1 t}, & 0 \leq t \leq \tau \\ a \left(1 - \frac{(1+b_1\tau)}{(1+b_2\tau)} (1 + b_2 t)e^{-b_1\tau - b_2(t-\tau)} \right) & \tau < t \end{cases}$
5	Pham-Zhang ⁴⁴	$m(t) = \frac{1}{(1 - \beta e^{-bt})} ((c + a) (1 - e^{-bt}) - \frac{ab}{b - \alpha} (e^{-\beta t} - e^{-bt}))$
6	Pure error generation ¹⁰	$m(t) = \left(\frac{a}{1 - \alpha} \right) (1 - e^{-b(1 - \alpha)t})$
7	Roy-Mahapatra-Dey model ⁴⁵	$m(t) = a\alpha (1 - e^{-bt}) - \frac{ab}{b - \beta} (e^{-\beta t} - e^{-bt})$
8	Yamada Exponential ⁴¹	$m(t) = a(1 - e^{-\gamma \alpha (1 - e^{-\beta t})})$
9	Yamada Imperfect 1 ⁴⁶	$m(t) = \frac{ab}{\alpha + b} (e^{\alpha t} - e^{-bt})$
10	Yamada Imperfect 2 ⁴¹	$m(t) = a(1 - e^{-bt}) \left(1 - \frac{\alpha}{b} \right) + \alpha at$
11	Yamada Rayleigh model ⁴⁶	$m(t) = a(1 - e^{-\gamma \alpha (1 - e^{-\beta t^2/2})})$
12	Proposed Model	$S(t) = \begin{cases} N - N \exp \left(\frac{-b_1 t}{1 - \alpha} \right) + \sigma_1 \exp \left(\frac{-b_1 t}{1 - \alpha} \right) \int_0^t \sigma_1 \exp \left(\frac{b_1 l}{1 - \alpha} \right) dC_l, & t \leq \tau \\ U_2(t) [S(\tau) + N \exp \left(\frac{b_2(t - \tau)}{1 - \alpha} \right) - N + \int_{\tau}^t \frac{\sigma_2}{U_2(l)} dC_l], & t > \tau \end{cases}$

Table 1. $m(t)$ i.e., mean value function of selected SRGMs.

and R^2 . However, the efficiency of SRGMs can be discussed by analyzing models proposing a set of comparison criteria as outlined below:

- The mean square error (MSE) gives the measure of divergence between the calculated values with the actual values of data as given below^{9,47}:

$$MSE = \frac{1}{N - n} \sum_{i=1}^n (y_i - \hat{m}(t_i))^2$$

where,

- N - Number of observations.
- y_i - Total number of faults detected up to time t_i .
- $m(t_i)$ - Estimated value of cumulative fault number up to time t_i derived from the mean value function, where $i = 1, 2, \dots, n$.
- n - represents number of parameters included in the model.

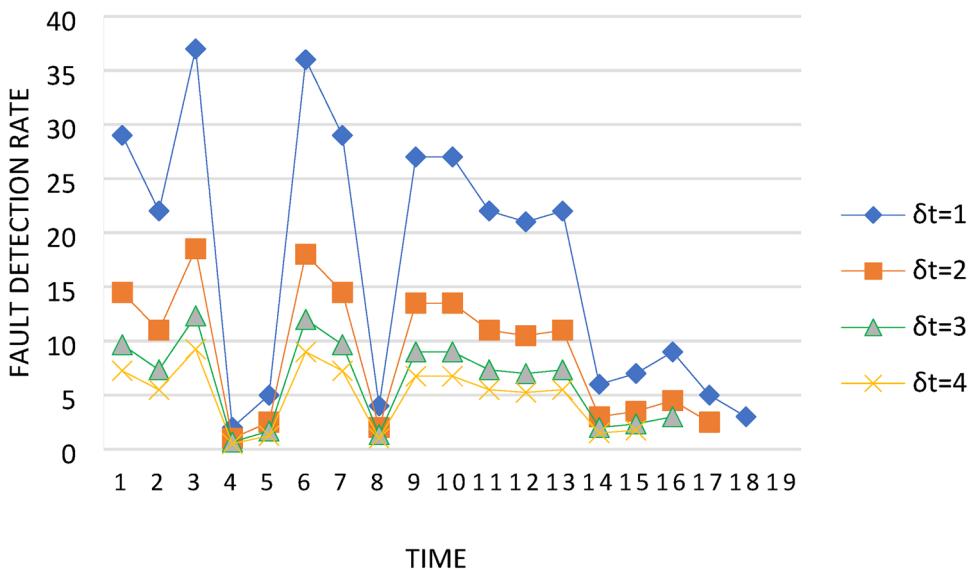
Consequently, a lesser value of MSE gives better goodness-of-fit.

- The second criterion implemented to compare the SRGMs is the correlation index of the regression curve equation (R^2) determined as^{9,47}:

$$R^2 = 1 - \frac{\sum_{i=1}^n (y_i - \bar{m}(t_i))^2}{\sum_{i=1}^n (y_i - \bar{y})^2}$$

where,

Weeks	Cumulative execution time (CPU hours)	Cumulative number of detected faults	Weeks	Cumulative execution time (CPU hours)	Cumulative number of detected faults
1	2.45	15	11	26.23	233
2	4.9	44	12	27.6	255
3	6.86	66	13	30.93	276
4	7.84	103	14	34.77	298
5	9.52	105	15	38.61	304
6	12.89	110	16	40.91	311
7	17.1	146	17	42.67	320
8	20.47	175	18	44.66	325
9	21.43	179	19	47.65	328
10	23.35	206			

Table 2. DS-I (19 weeks)- (Failure data from PL/I database application).**Fig. 1.** Failures increasing rate for different δt values for DS-I.

- $\bar{y} = \frac{1}{n} \sum_{i=1}^n y_i$. Hence, the greater value of R^2 indicates better model performance.

Dataset 1/DS-I

The DS-I is used to calculate and estimate unknown parameters⁴³. The software, during the testing phase of 19 weeks, took 47.65 CPU hours, and the data set has 328 detected faults. It is shown in Table 2.

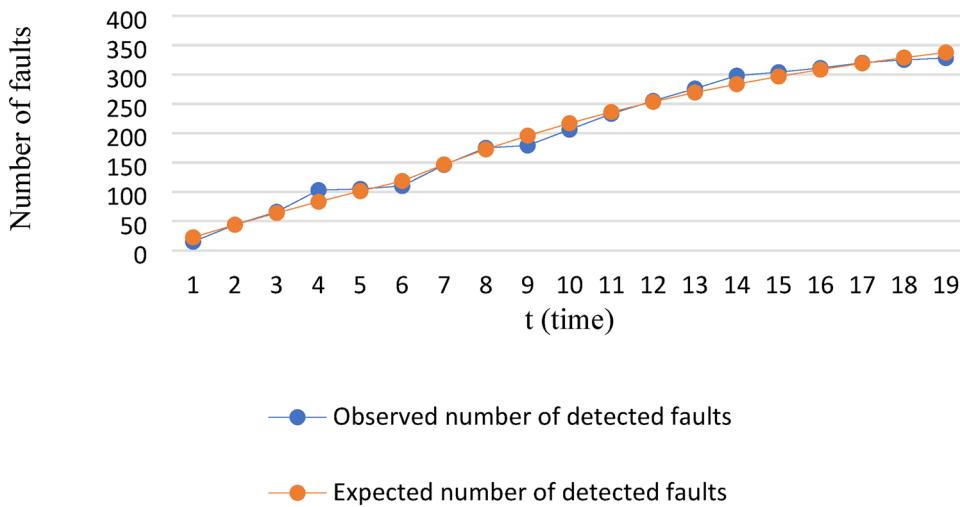
In the beginning, with software failure increasing rate, $y'(t)$ remains relatively stable. This stage aligns with the early phases of testing when testers are trying to know the algorithm and software's failure mode. After an initial stable stage, $y'(t)$ begins to showcase a significant increase. This rise is attributed to the ability of testers to detect and address faults more effectively. The increase in failure rate touches its peak at a certain point in time, representing the time when the FDR is at its best. Following this peak, $y'(t)$ gradually decreases, implying that testers have resolved almost all faults, and the software is becoming more stable. Eventually, $y'(t)$ stabilizes at a lower rate, suggesting that the software has attained a relatively fault-free state. After employing empirical data analysis, we have identified that the change point for the given DS-I is at 6 weeks as shown in Fig. 1.

Additionally, we have used SPSS version 20.0 to estimate the unknown parameters of the developed model namely $(N, b_1, b_2, \alpha_1, \alpha_2)$ and we have also developed a code in Google Colab using Python version 3.10 to estimate the unknown parameters which represent uncertainty in the proposed model namely σ_1, σ_2 . Additionally, we also used Mathematica version 13.2 to plot graphs to show belief reliability distribution and belief reliability.

The estimated values of unknown parameters are catalogued in Table 3. The expected number of detected faults is more in terms of number than the actual faults given in the data. The graph of the observed and estimated number of detected faults is presented in Fig. 2.

The uncertain differential equation is

Parameters	Values
Total number of faults present in software at time t (N)	423.3682502
Fault detection rate for $0 \leq t \leq \tau$ (b_1)	0.054629933
Fault detection rate for $\tau < t \leq t$ (b_2)	0.042641026
new fault introduction rate for $0 \leq t \leq \tau$ (α_1)	0.001999209
new fault introduction rate for $\tau < t \leq t$ (α_2)	0.004978238
uncertainty level for $0 \leq t \leq \tau$ (σ_1)	0.24701832
uncertainty level for $\tau < t \leq t$ (σ_2)	0.0584511
change point (τ)	6

Table 3. Estimated values for unknown parameters for DS-I.**Fig. 2.** Observed and expected number of detected faults for DS-I.

$$dS(t) = \begin{cases} (21.15 - 0.05S(t))dt + 0.24dC_t, & t \leq 6, \\ (17.05 - 0.04S(t))dt + 0.05dC_t, & t > 6, \end{cases} \quad (29)$$

As illustrated in Fig. 3, all the observed values lie within the region between the 0.000000124-path and the 0.996-path of the uncertain differential equation, hence the estimated values are valid and permissible.

Table 4 represents the comparison of the proposed model with other existing models based on MSE and R^2 criteria. As evident, the proposed model performs better as compared to other existing models.

Now, we plot the graph of belief reliability distribution before and after the change point, i.e. $\tau=6$, as represented in Fig. 4. The belief reliability distribution at time $t=5$ is depicted in Fig. 5. Take into account an example, we can see that $\phi_{10}(95)=0.9053$, which indicates that the belief degree that the number of detected faults is less than 95 till $t=10$ is 0.9053. Additionally, we outline the graph of belief reliability before and after the change point, i.e. $\tau=6$, which is displayed in Fig. 6. For given belief distribution function $f=0.05$, belief reliability is displayed in Fig. 7.

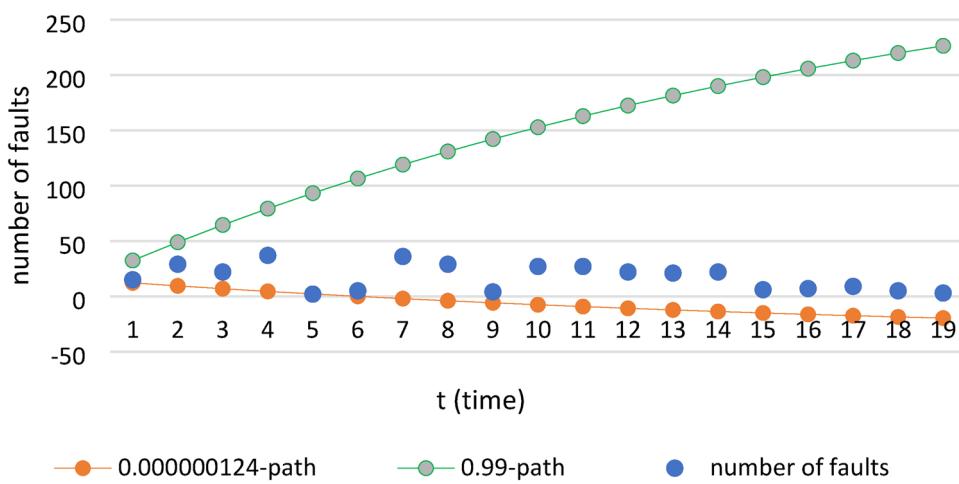
Dataset 2/DS-II

The DS-II is a widely used dataset⁴⁸. It was observed for 22 weeks, during which a sum of 86 faults were identified. The testing effort, determined in CPU hours, accounted for 93 h. We have estimated the change point based on the above-mentioned empirical data analysis for the given DS-II and found it to be 13 weeks. The analysis through graphical representation is shown in Fig. 8.

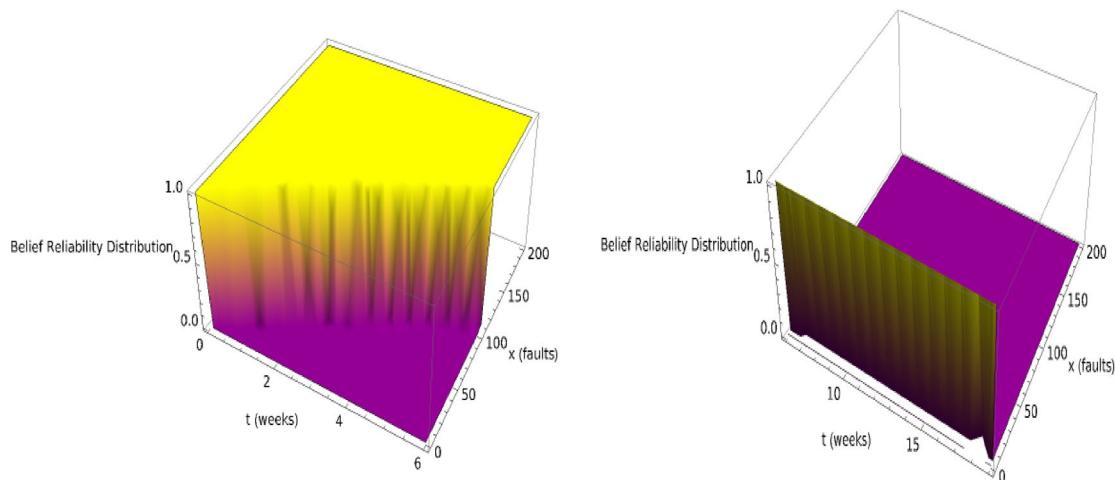
Table 5 presents the values of estimated unknown parameters. The graph of the observed and estimated number of detected faults is given in Fig. 9, which also indicates that the expected number of detected faults is greater than the observed faults.

Hence, the uncertain differential equation is

$$dS(t) = \begin{cases} (6.38 - 0.042S(t))dt + 0.298dC_t, & t \leq 13, \\ (0.14 - 0.001S(t))dt + 0.463dC_t, & t > 13, \end{cases} \quad (30)$$

**Fig. 3.** α -paths for DS-I.

No.	Model Name	Estimated Parameters	Comparison criteria	
			MSE	R^2
1	Delayed S-Shaped	$a = 374, b = 0.19$	200.2	0.983
2	GO model	$a = 750, b = 0.17$	166	0.986
3	Inflection S-shaped	$a = 380, b = 0.17, \beta = 2.88$	105.7	0.991
4	S-shaped change-point	$a = 393, b_1 = 0.19, b_2 = 0.17$	219.2	0.984
5	Pham-Zhang	$a = 330, b = 0.28, c = 21.03, \alpha = 9.49, \beta = 10.20$	141.6	0.990
6	Pure error generation	$a = 52.4, b = 0.46, \alpha = 0.93$	177.0	0.986
7	Roy-Mahapatra-Dey model	$a = 180, b = 0.05, \alpha = 2.99, \beta = 0.22$	155.4	0.988
8	Yamada Exponential	$a = 1238, \alpha = 5.29, \beta = 0.01, \gamma = 0.26$	192.7	0.986
9	Yamada Imperfect 1	$a = 682, b = 0.03, \alpha = 0.002$	178.6	0.986
10	Yamada Imperfect 2	$a = 589, b = 0.04, \alpha = 0.001$	313.7	0.976
11	Yamada Rayleigh model	$a = 423, \alpha = 1.73, \beta = 0.01, \gamma = 0.96$	362.9	0.974
12	Proposed Model	$N = 423, b_1 = 0.05, b_2 = 0.04, \alpha_1 = 0.001, \alpha_2 = 0.004, \sigma_1 = 0.247, \sigma_2 = 0.058$	105.3	0.992

Table 4. Model comparisons: MSE and R^2 for different models for DS-I.**Fig. 4.** Belief reliability distribution for the proposed model before and after the change point at $t=6$.

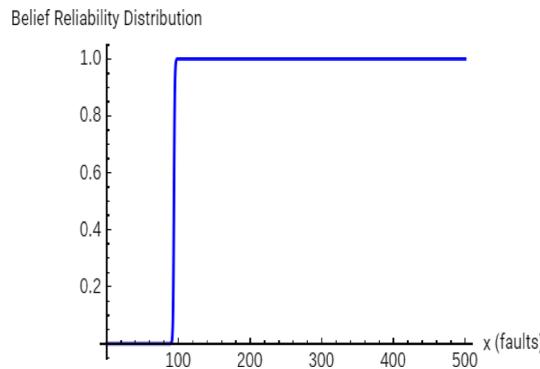


Fig. 5. Belief reliability distribution at time $t=5$.

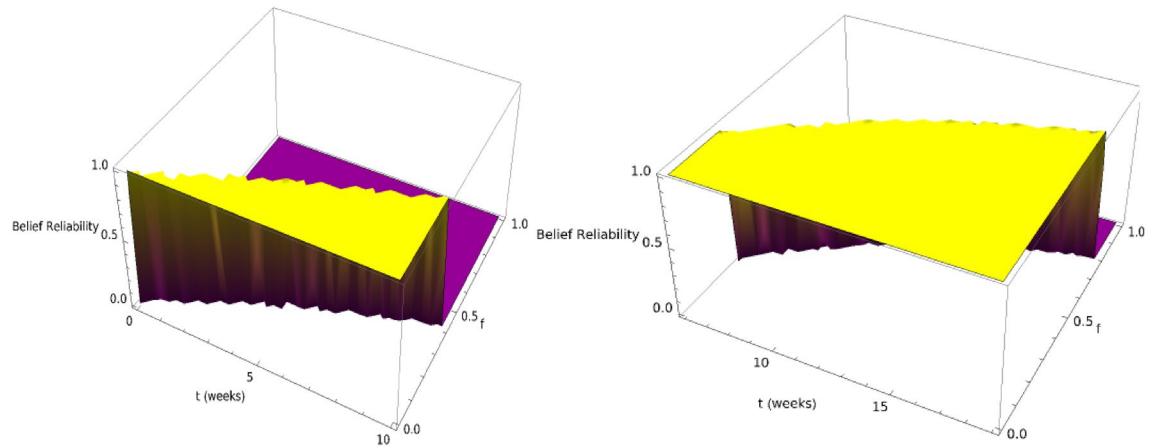


Fig. 6. Belief reliability for the proposed model before and after the change point at $t=6$.

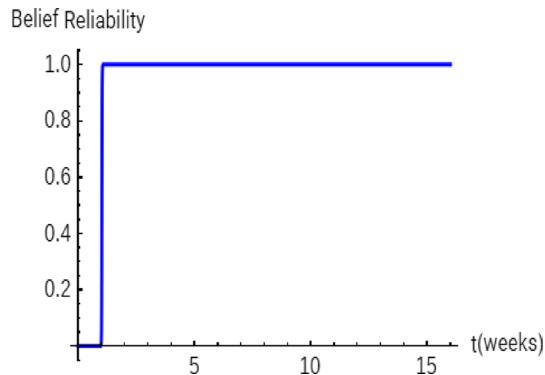


Fig. 7. Belief reliability at belief distribution function $f=0.05$.

As demonstrated in Fig. 10, all the observed data lies within the region within the 0.35-path and 0.99-path of the uncertain differential equation, the estimated values are valid and acceptable.

Table 6 depicts the comparison of our proposed model with other well-established models based on the criteria mentioned above and we can conclude that our proposed model outperforms as compared to mentioned existing models.

Similarly, as in DS-I, we plot the graph belief reliability distribution and belief reliability before and after the change point i.e. $\tau=13$ which is represented in Figs. 11 and 12 respectively. At a given time, $t=20$, the belief reliability distribution is presented in Fig. 13. For particular $f=0.5$, belief reliability is represented in Fig. 14.

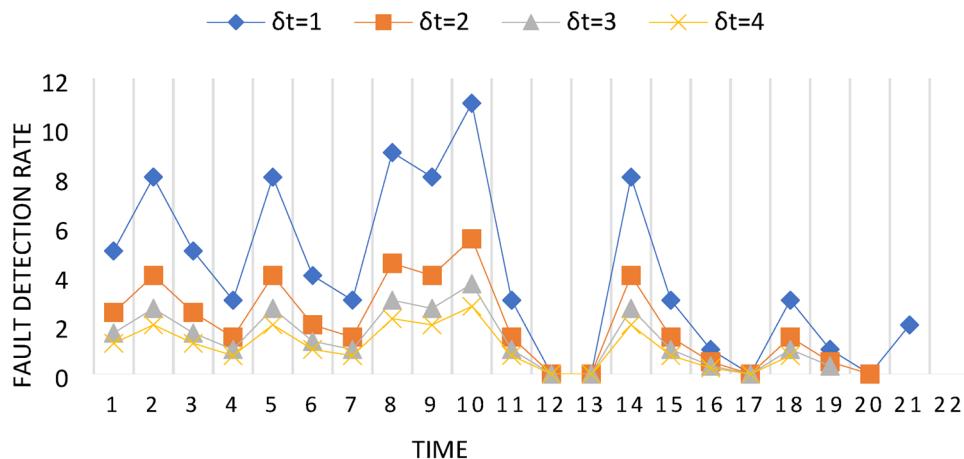


Fig. 8. Failures increasing rate for different δt values for DS-II.

Parameters	Values
Total number of faults present in software at time t (N)	149
Fault detection rate for $0 \leq t \leq \tau$ (b_1)	0.0426
Fault detection rate for $\tau < t \leq T$ (b_2)	0.001
new fault introduction rate for $0 \leq t \leq \tau$ (α_1)	0.0057
new fault introduction rate for $\tau < t \leq T$ (α_2)	0.001
uncertainty level for $0 \leq t \leq \tau$ (σ_1)	0.2985
uncertainty level for $\tau < t \leq T$ (σ_2)	0.463
change point (τ)	13

Table 5. Estimated values of unknown parameters for DS-II.

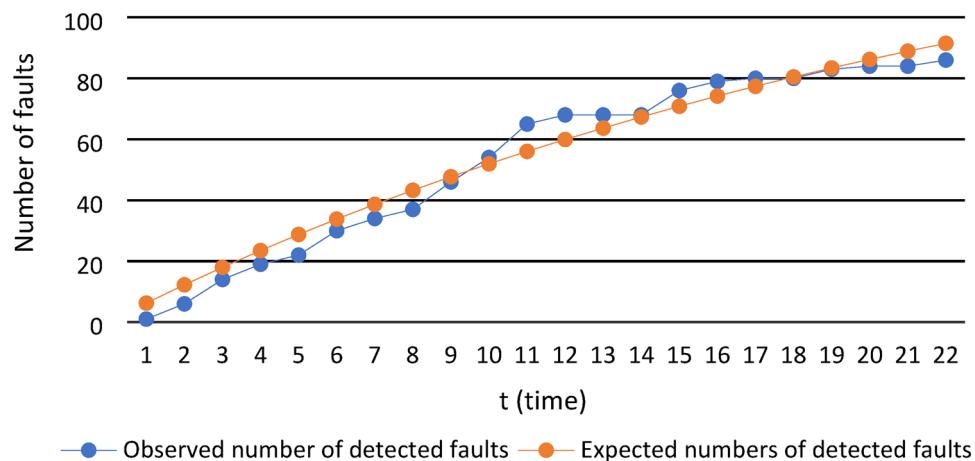


Fig. 9. Observed and expected number of detected faults for DS-II.

Dataset 3/DS-III

The data set (DS-III) is extracted from a web-based integrated accounting ERP system (Web ERP) hosted on SourceForge.net⁴⁹, dated from August 2003 to July 2008, where the time unit is in months. A sum of 146 bugs were detected in 60 months. We have identified that the change point using the above-mentioned empirical data analysis for the given DS-III is 56 months. The analysis is shown in Fig. 15.

Table 7 gives optimized values of unknown parameters. The expected number of detected faults exceeds the observed number of faults given in the data as shown in Fig. 16.

Hence, the uncertain differential equation is

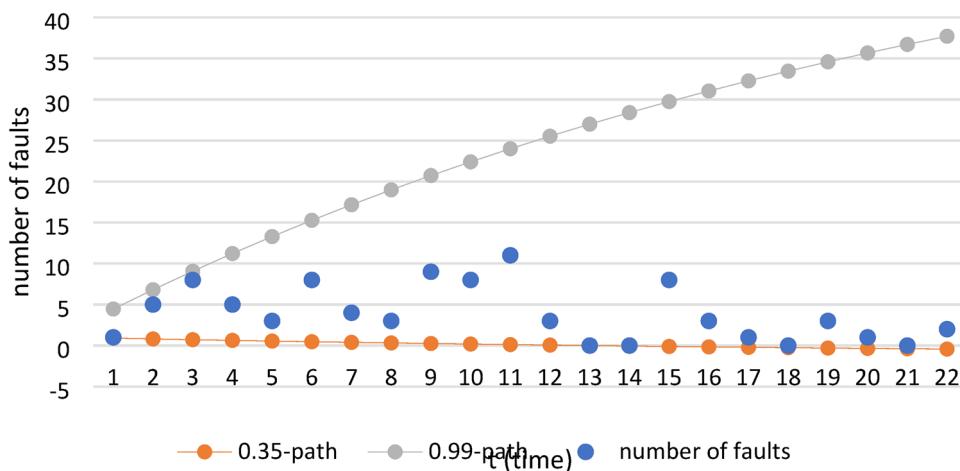


Fig. 10. α -paths for DS-II.

No.	Model name	Estimated parameters	Comparison criteria	
			MSE	R^2
1	Delayed S-Shaped	$a = 101, b = 0.19$	32.8	0.963
2	GO model	$a = 155, b = 0.04$	37.2	0.969
3	Inflection S-shaped	$a = 150, b = 0.04, \beta = 0.07$	32.3	0.967
4	S-shaped change-point	$a = 121, b_1 = 0.16, b_2 = 0.10$	56.4	0.943
5	Pham-Zhang	$a = 79, b = 0.21, c = 21.88, \alpha = 0.19, \beta = 0.16$	36.8	0.965
6	Pure error generation	$a = 115, b = 0.05, \alpha = 0.27$	34.7	0.963
7	Roy-Mahapatra-Dey model	$a = 149, b = 0.20, \alpha = 0.97, \beta = 0.18$	38.4	0.961
8	Yamada Exponential	$a = 211, \alpha = 1.98, \beta = 0.01, \gamma = 1.10$	32.3	0.969
9	Yamada Imperfect 1	$a = 113, b = 0.05, \alpha = 0.01$	32.6	0.968
10	Yamada Imperfect 2	$a = 6, b = 1.12, \alpha = 0.73$	55.7	0.941
11	Yamada Rayleigh model	$a = 176, \alpha = 0.35, \beta = 0.01, \gamma = 2.10$	37.4	0.965
12	Proposed Model	$N = 149, b_1 = 0.04, b_2 = 0.001, \alpha_1 = 0.005, \alpha_2 = 0.001, \sigma_1 = 0.298, \sigma_2 = 0.463$	31.9	0.970

Table 6. Model comparisons: MSE and R^2 for different models for DS-II.

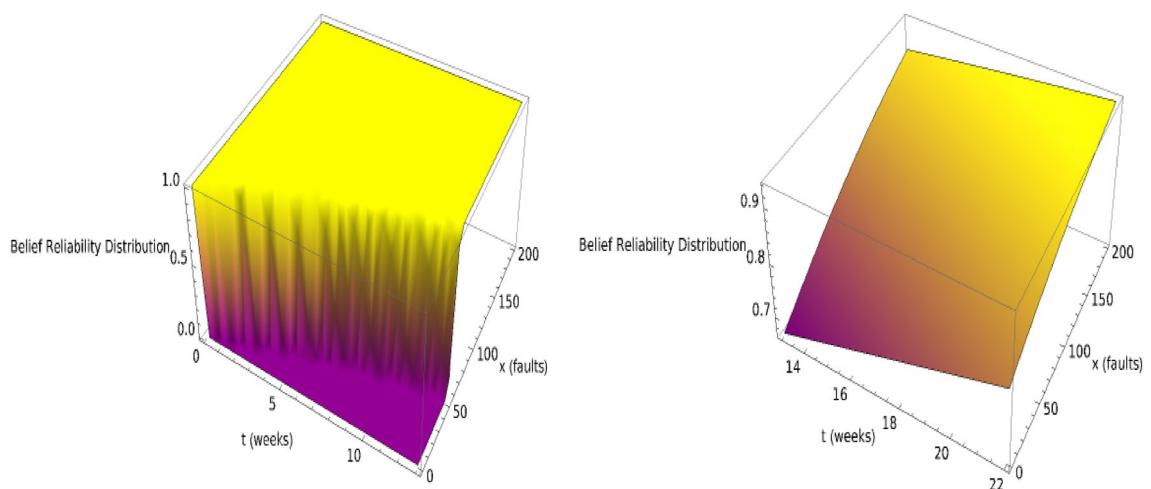


Fig. 11. Belief reliability distribution for the proposed model before and after the change point at $t = 13$.

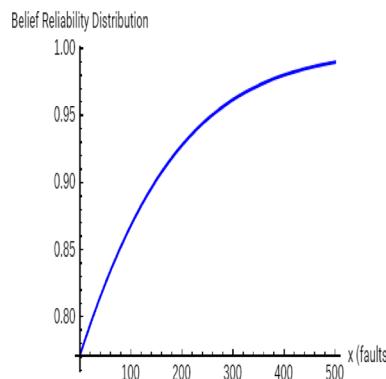


Fig. 12. Belief reliability for the proposed model before and after the change point at $t = 13$.

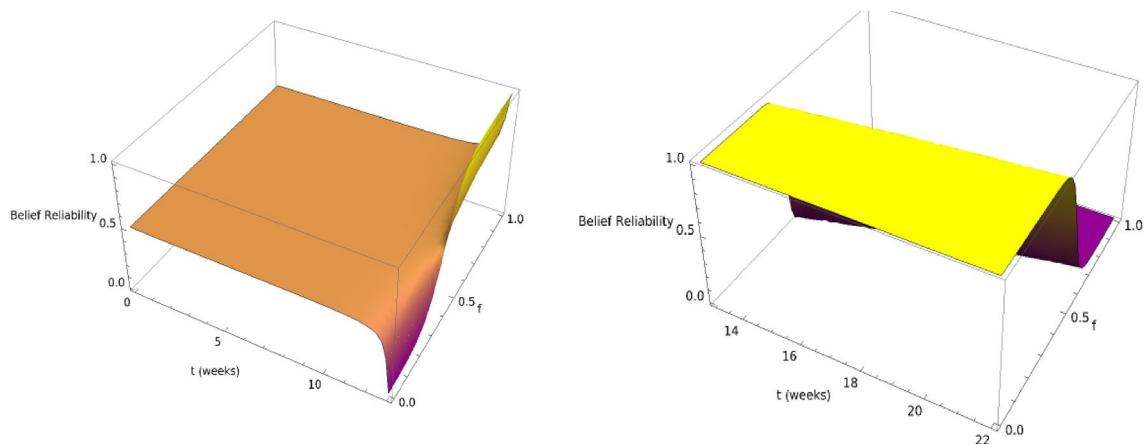


Fig. 13. Belief reliability distribution at time $t = 20$.

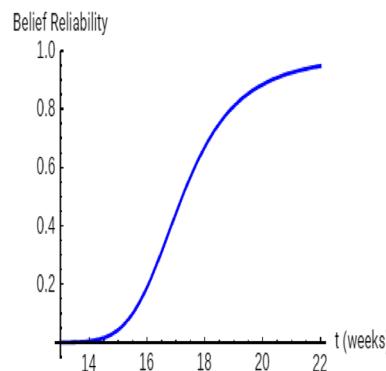


Fig. 14. Belief reliability at belief distribution function $f = 0.5$.

$$dS(t) = \begin{cases} (1.48 - 0.01(t))dt + 0.621dC_t, & t \leq 56, \\ (2060 - 14.01S(t))dt + 1.64dC_t, & t > 56, \end{cases} \quad (31)$$

All the observed data lie in the area within 0.18-path and 0.99-path of uncertain differential equation as shown in Fig. 17. Therefore, all the estimated values are acceptable.

Table 8 represents the comparison of our proposed model with other models based on the criteria mentioned above and we can conclude that our proposed model performs better than other models.

Similarly, as in DS-I, we plot the graph of belief reliability distribution before and after the change point, i.e. $\tau = 56$, which is represented in Fig. 18. At a given time, the belief reliability distribution is similar to the way we

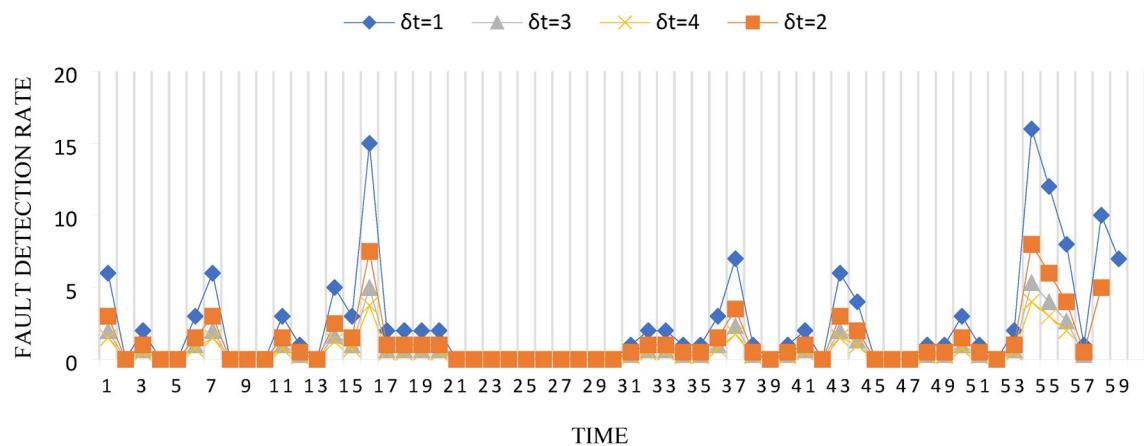


Fig. 15. Failures increasing rate for different δt values for DS-III.

Parameters	Values
Total number of faults at time ' t ' (N)	146.9334
Fault detection rate for $0 \leq t \leq \tau$ (b_1)	0.01702
Fault detection rate for $\tau < t \leq t$ (b_2)	8.27857
new fault introduction rate for $0 \leq t \leq \tau$ (α_1)	0.0001
new fault introduction rate for $\tau < t \leq t$ (α_2)	0.4017
uncertainty level for $0 \leq t \leq \tau$ (σ_1)	0.6212
uncertainty level for $\tau < t \leq t$ (σ_2)	1.6472
change point (τ)	56

Table 7. Estimated values of unknown parameters for DS-III.

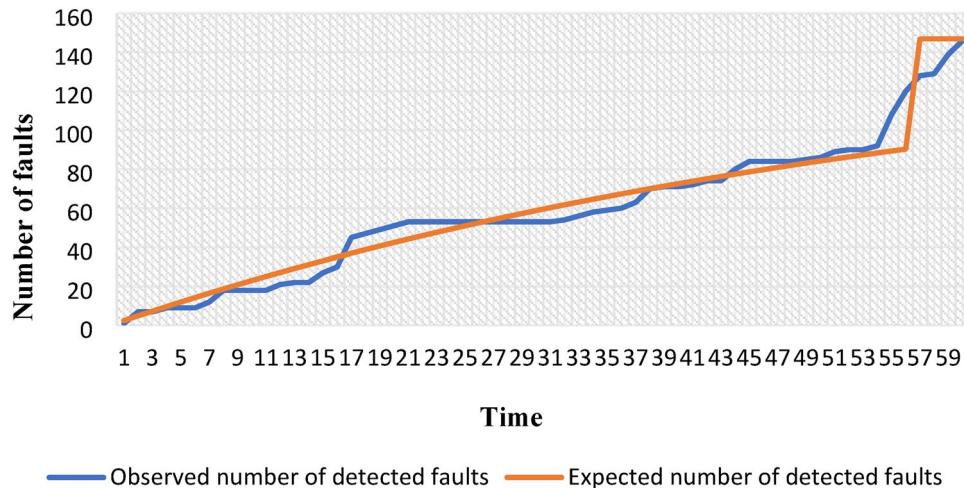
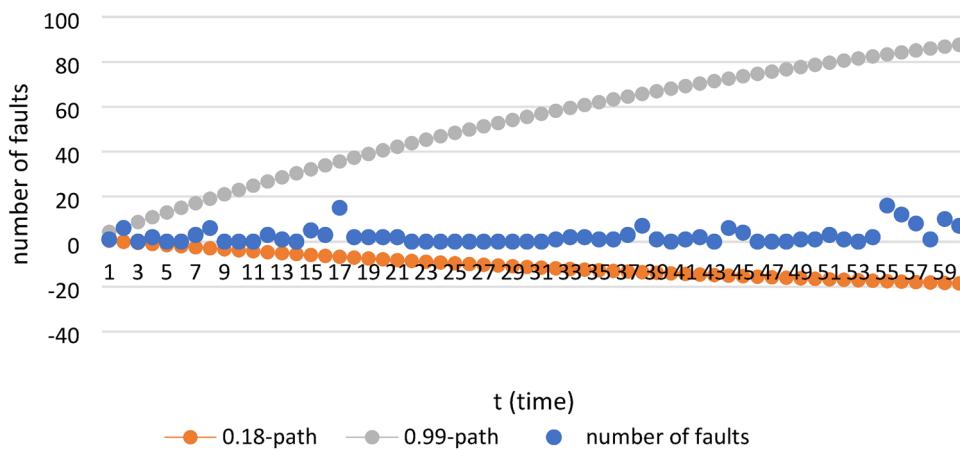


Fig. 16. Observed and expected number of faults for DS-III.

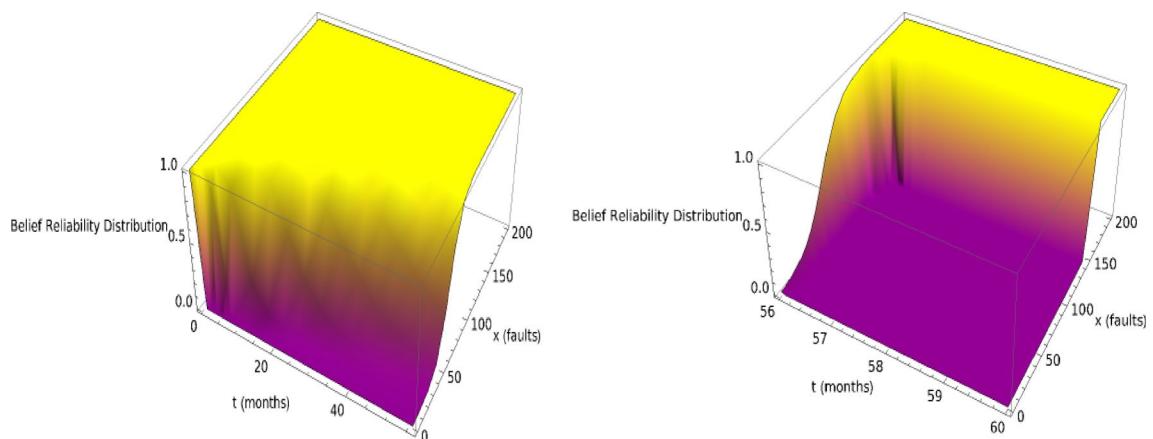
calculated in DS-I. Further, we plot the belief reliability before and after the change point, i.e. $\tau = 56$, which is shown in Fig. 19.

Managerial implications

The research paper introduces a new approach to estimate software reliability under uncertainty framework, including factors change point and imperfect debugging. The incorporation of change point and imperfect debugging altogether is the new approach adopted in this paper that identifies significant changes in fault

Fig. 17. α -paths for DS-III.

No.	Model name	Estimated parameters	Comparison criteria	
			MSE	R^2
1	Delayed S-Shaped	$a = 153, b = 0.04$	138	0.889
2	GO model	$a = 617, b = 0.003$	95.40	0.923
3	Inflection S-shaped	$a = 905, b = 0.009, \beta = 3.96$	86.02	0.932
4	S-shaped change-point	$a = 131, b_1 = 0.04, b_2 = 2.16$	77.97	0.939
5	Pham-Zhang	$a = 627, b = 0.03, c = 14.12, \alpha = 1.02, \beta = 34.4$	92.72	0.929
6	Pure error generation	$a = 12, b = 0.14, \alpha = 1.03$	85.07	0.933
7	Roy-Mahapatra-Dey model	$a = 176, b = 0.001, \alpha = 7.16, \beta = 0.02$	91.88	0.929
8	Yamada Exponential	$a = 962, \alpha = 7.68, \beta = 0.0002, \gamma = 0.96$	96.06	0.925
9	Yamada Imperfect 1	$a = 612, b = 0.002, \alpha = 0.007$	84.61	0.933
10	Yamada Imperfect 2	$a = 6, b = 0.23, \alpha = 0.31$	98.78	0.922
11	Yamada Rayleigh model	$a = 161, \alpha = 0.79, \beta = 0.0007, \gamma = 1.99$	177.4	0.86
12	Proposed Model	$N = 147, b_1 = 0.01, b_2 = 8.27, \alpha_1 = 0.0001, \alpha_2 = 0.401, \sigma_1 = 0.6212, \sigma_2 = 1.647$	6.03	0.995

Table 8. Model comparisons: MSE and R^2 for different models with for DS-III.Fig. 18. Belief reliability distribution for the proposed model before and after the change point at $t = 56$.

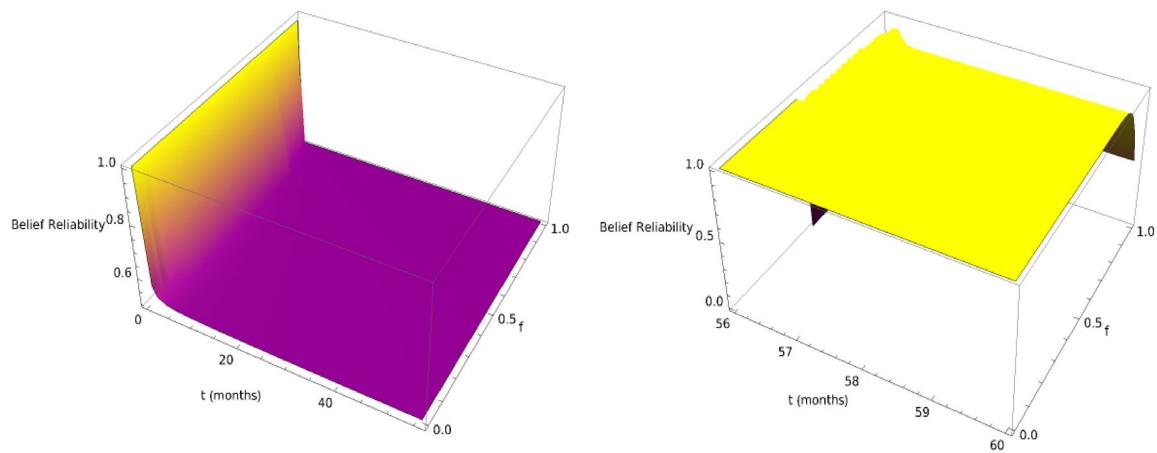


Fig. 19. Belief reliability for the proposed model before and after the change point at $t=56$.

detection rate. Managers can pinpoint these key points in the software lifecycle where changes (like feature additions or major fixes) might impact reliability, and adjust source code, resource allocation, or testing strategies as required. By involving uncertainty directly in the equations, managers can gain better insights into risk factors, including uncertainty, and make well-informed decisions related to software. The integration of uncertainty theory in the model suggests that software managers must be prepared for unexpected changes in system reliability due to external factors, unpredictable bugs, or imperfections in the debugging process. By incorporating this model, managers can develop more reliable risk management strategies, including alternative plans, which deal with the inherent uncertainty in software reliability. This could also result in more anticipatory decision-making when challenges arise. By understanding the impact of change points, imperfect debugging, and the uncertainties inherent in software systems, managers can optimize their workflows, improve resource allocation, and enhance software reliability in an uncertain environment. These characteristics enhance risk assessment and project management. For instance, if the developed model indicates that if debugging phase after the change point is resulting in the reintroduction of faults, then managers can opt to pause the testing, refactor the code, or upskill the staff. Such decisions enhance more informed decisions about optimal release time, cost estimation, and resource allocation. This could result in developing uncertain differential equations with time delays.

Conclusion and future scope

In this research, we have developed a novel software belief reliability growth model incorporating change point and imperfect debugging based on the mathematical approach of uncertainty theory that deals with epistemic uncertainties. This article is the first to integrate change point and imperfect debugging using uncertain differential equations based on belief reliability theory. Using principles of uncertainty theory, our proposed model discusses a more flexible and adaptable approach for estimating software reliability over time. Through numerical illustrations, we showcase the effectiveness of our model in handling the complexities of software development lifecycles and represent a significant contribution to advancing software reliability modelling, offering researchers an important methodology under the framework of uncertain theory. The newly developed model has been validated on three real data sets to generate more effective results. The methodology for estimating the unknown parameters is derived using the least squares method, and the change point is also estimated using empirical data analysis. The proposed model is compared with many other well-known existing SRGMs, and the results conclude that our model performs better than those models. Further, this research can be continued and explored by taking into account aspects like testing effort, multiple change points, time delay, patching and error generation. Other methodologies for estimating parameters can be developed. In future scope of research, time delays can be incorporated into the proposed model.

Data availability

We confirm that the data supporting this study's findings are cited and accessible within the article.

Received: 23 February 2025; Accepted: 1 December 2025

Published online: 12 December 2025

References

1. Zhang, X. & Pham, H. An analysis of factors affecting software reliability. *J. Syst. Softw.* **50** (1), 43–56. [https://doi.org/10.1016/S0164-1212\(99\)00075-8](https://doi.org/10.1016/S0164-1212(99)00075-8) (2000).
2. Por, L. Y., Dai, Z., Leem, S. J., Chen, Y., Yang, J., Binbeshr, F., ... & Ku, C. S. A systematic literature review on the methods and challenges in detecting zero-day attacks: Insights from the recent CrowdStrike incident. *IEEE Access*. <https://doi.org/10.1109/ACCESS.2024.3455410> (2024).

3. Singh, V., & Kumar, V. Revisiting multi-release software reliability growth model under the influence of multiple software patches. *Life Cycle Reliability and Safety Engineering*, **14**(2), 275–288 <https://doi.org/10.1007/s41872-025-00309-6> (2025).
4. Fernandez, A., & Stol, N. Economic, dissatisfaction, and reputation risks of hardware and software failures in PONs. *IEEE/ACM Trans. Networking*, **25** (2), 1119–1132. <https://doi.org/10.1109/TNET.2016.2619062> (2016).
5. Kumar, V., & Pham, H. (Eds.). Predictive analytics in system reliability. Springer Nature. <https://doi.org/10.1007/978-3-031-05347-4> (2022).
6. Liu, Y., Xie, M., Wang, L., & Hu, Q. Reliability analysis of open source software: A Bayesian approach considering both fault detection and correction processes. In *26th European Safety and Reliability Conference, ESREL 2016*. CRC Press/Balkema. (2016), <https://doi.org/10.1201/9781315374987-360>
7. Ke, S. Z., & Huang, C. Y. Software reliability prediction and management: A multiple change-point model approach. *Qual. Reliab. Eng. Int.* **36** (5), 1678–1707. <https://doi.org/10.1002/qre.2653> (2020).
8. Kim, Y. S., Song, K. Y., Pham, H., & Chang, I. H. A software reliability model with dependent failure and optimal release time. *Symmetry* **14** (2), 343. <https://doi.org/10.3390/sym14020343> (2022).
9. Saxena, P., & Ram, M. Two phase software reliability growth model in the presence of imperfect debugging and error generation under fuzzy paradigm. *Mathematics in Engineering, Science & Aerospace (MESA)*, **13**(2) (2022).
10. Yang, J., Liu, Y., Xie, M., & Zhao, M. Modeling and analysis of reliability of multi-release open source software incorporating both fault detection and correction processes. *J. Syst. Softw.* **115**, 102–110. <https://doi.org/10.1016/j.jss.2016.01.025> (2016).
11. Kumar, V., Pradhan, S. K., Kumar, A., & Kapur, P. K. Revisiting software reliability growth model under general setup. *International Journal of System Assurance Engineering and Management*, 1–14. <https://doi.org/10.1007/s13198-025-02988-x> (2025).
12. Li, Q., & Pham, H. Modeling software fault-detection and fault-correction processes by considering the dependencies between fault amounts. *Appl. Sci.* **11** (15), 6998. <https://doi.org/10.3390/app11156998> (2021).
13. Yamada, S., Kimura, M., Tanaka, H., & Osaki, S. Software reliability measurement and assessment with stochastic differential equations. *IEICE Trans. Fundamentals Electron. Commun. Comput. Sci.* **77** (1), 109–116 (1994).
14. Kapur, P. K., Anand, S., Yadavalli, V. S. S., & Beichelt, F. A generalised software growth model using stochastic differential equation. *Communication in dependability and quality management belgrade, Serbia*, **34**, (2007).
15. Li, Q., & Pham, H. NHPP software reliability model considering the uncertainty of operating environments with imperfect debugging and testing coverage. *Appl. Math. Model.* **51**, 68–85. <https://doi.org/10.1016/j.apm.2017.06.034> (2017).
16. Zhao, M. Change-point problems in software and hardware reliability. *Commun. Stat. - Theory Methods*, **22** (3), 757–768. <https://doi.org/10.1080/03610929308831053> (1993).
17. Huang, C. Y. Performance analysis of software reliability growth models with testing-effort and change-point. *J. Syst. Softw.* **76** (2), 181–194. <https://doi.org/10.1016/j.jss.2004.04.024> (2005).
18. Kapur, P. K., Pham, H., Anand, S., & Yadav, K. A unified approach for developing software reliability growth models in the presence of imperfect debugging and error generation. *IEEE Trans. Reliab.* **60** (1), 331–340. <https://doi.org/10.1109/TR.2010.2103590> (2011).
19. Chatterjee, S., & Shukla, A. Change point-based software reliability model under imperfect debugging with revised concept of fault dependency. *Proc. Institution Mech. Eng. Part. O: J. Risk Reliab.* **230** (6), 579–597. <https://doi.org/10.1177/1748006X16673767> (2016).
20. Liu, B. Why is there a need for uncertainty theory. *J. Uncertain. Syst.* **6** (1), 3–10 (2012).
21. Kapur, P. K., Pham, H., Chanda, U., & Kumar, V. Optimal allocation of testing effort during testing and debugging phases: a control theoretic approach. *International Journal of Systems Science*, **44**(9), 1639–1650 <https://doi.org/10.1080/00207721.2012.669861> (2013).
22. Singh, O., Anand, A., Singh, J., & Kapur, P. K. Assessment of distribution based SRGM with the effect of Change-Point and imperfect debugging incorporating irregular fluctuations. *J. Pure Appl. Sci. Technol.* **2** (1), 37–50 (2012).
23. Der Kiureghian, A. (ed. O.) Aleatory or epistemic? Does it matter? *Struct. Saf.* **31** 2 105–112 <https://doi.org/10.1016/j.strusafe.2008.06.020> (2009).
24. Liu, Z., & Kang, R. Imperfect debugging software belief reliability growth model based on uncertain differential equation. *IEEE Trans. Reliab.* **71** (2), 735–746. <https://doi.org/10.1109/TR.2022.3158336> (2022).
25. Liu, Z., Wang, S., Liu, B., & Kang, R. Change point software belief reliability growth model considering epistemic uncertainties. *Chaos Solitons Fractals*, **176**, 114178. <https://doi.org/10.1016/j.chaos.2023.114178> (2023).
26. Garg, M., Kumar, V., Chaudhary, K., & Kapur, P. K. Uncertain differential equation based software belief reliability growth model (SBRGM) considering software patching. *Int. J. Syst. Assur. Eng. Manage.* 1–14. <https://doi.org/10.1007/s13198-023-02225-3> (2024).
27. Reformat, M. A fuzzy-based multimodel system for reasoning about the number of software defects. *Int. J. Intell. Syst.* **20** (11), 1093–1115. <https://doi.org/10.1002/int.20113> (2005).
28. Bai, C. G., Hu, Q. P., Xie, M., & Ng, S. H. Software failure prediction based on a Markov bayesian network model. *J. Syst. Softw.* **74** (3), 275–282. <https://doi.org/10.1016/j.jss.2004.02.028> (2005).
29. Pievatolo, A., Ruggeri, F., & Soyer, R. A bayesian hidden Markov model for imperfect debugging. *Reliab. Eng. Syst. Saf.* **103**, 11–21. <https://doi.org/10.1016/j.ress.2012.03.003> (2012).
30. Shen, Q., Lou, J., Zhang, X., & Jiang, Y. Failure prediction by regularized fuzzy learning with intelligent parameters selection. *Appl. Soft Comput.* **100**, 106952. <https://doi.org/10.1016/j.asoc.2020.106952> (2021).
31. Zhang, Q., Kang, R., & Wen, M. Belief reliability for uncertain random systems. *IEEE Trans. Fuzzy Syst.* **26** (6), 3605–3614. <https://doi.org/10.1109/TFUZZ.2018.2838560> (2018).
32. Liu, B. *Uncertainty Theory* 2nd edn (Springer, 2007).
33. Yao, K., & Chen, X. A numerical method for solving uncertain differential equations. *J. Intell. Fuzzy Syst.* **25** (3), 825–832. <https://doi.org/10.3233/IFS-120688> (2013).
34. Liu, Z., & Kang, R. Multiple error types software belief reliability growth model based on uncertain differential equation. In *2021 IEEE 21st international conference on software quality, reliability and security (QRS)* (378–387). IEEE. (2021), <https://doi.org/10.1109/QRS54544.2021.00049>
35. Liu, Z., Yang, S., Yang, M., & Kang, R. Software belief reliability growth model based on uncertain differential equation. *IEEE Trans. Reliab.* **71** (2), 775–787. <https://doi.org/10.1109/TR.2022.3154770> (2022).
36. Huong, D. C. Discrete-time dynamic event-triggered H_∞ control of uncertain neural networks subject to time delays and disturbances. *Optimal Control Appl. Methods*, **44** (4), 1651–1670. <https://doi.org/10.1002/oca.2945> (2023).
37. Huong, D. C. Event-triggered finite-time guaranteed cost control of uncertain active suspension vehicle system. *J. Control Decis.* 1–10. <https://doi.org/10.1080/23307706.2024.2423186> (2024).
38. Huong, D. C. Event-triggered guaranteed cost control for uncertain polytopic fractional-order systems subject to unknown time-varying delays. *Rend. del. Circolo Matematico Di Palermo Ser. 2* (3), 917–928. <https://doi.org/10.1007/s12215-023-00960-x> (2024). 73.
39. Liu, B. Some research problems in uncertainty theory. *J. Uncertain. Syst.* **3** (1), 3–10 (2009).
40. Zhu, M., & Pham, H. A two-phase software reliability modeling involving with software fault dependency and imperfect fault removal. *Comput. Lang. Syst. Struct.* **53**, 27–42. <https://doi.org/10.1016/j.cl.2017.12.002> (2018).
41. Yamada, S., Ohba, M., & Osaki, S. S-shaped reliability growth modeling for software error detection. *IEEE Trans. Reliab.* **32** (5), 475–484. <https://doi.org/10.1109/TR.1983.5221735> (1983).

42. Goel, A. L. & Okumoto, K. Time-Dependent Error-Detection rate model for software reliability and other performance measures. *IEEE Trans. Reliab.* **28**, 206–211. <https://doi.org/10.1109/TR.1979.5220566> (1979).
43. Ohba, M. Software reliability analysis models. *IBM J. Res. Dev.* **28** (4), 428–443. <https://doi.org/10.1147/rd.284.0428> (1984).
44. Pham, H. & Zhang, X. An NHPP software reliability model and its comparison. *Int. J. Reliab. Qual. Saf. Eng.* **4** (03), 269–282. <https://doi.org/10.1142/S0218539397000199> (1997).
45. Roy, P., Mahapatra, G. S. & Dey, K. N. An NHPP software reliability growth model with imperfect debugging and error generation. *Int. J. Reliab. Qual. Saf. Eng.* **21** (02), 1450008. <https://doi.org/10.1142/S0218539314500089> (2014).
46. Yamada, S., Tokuno, K. & Osaki, S. Imperfect debugging models with fault introduction rate for software reliability assessment. *Int. J. Syst. Sci.* **23** (12), 2241–2252. <https://doi.org/10.1080/00207729208949452> (1992).
47. Pham, H. *System Software Reliability* (Springer Science & Business Media, 2007).
48. Tohma, Y., Jacoby, R., Murata, Y. & Yamamoto, M. Hyper-geometric distribution model to estimate the number of residual software faults. In [1989] *Proceedings of the Thirteenth Annual International Computer Software & Applications Conference* (610–617). IEEE. (1989)., <https://doi.org/10.1109/CMP SAC.1989.65155>
49. SourceForge.net. An Open Source Software website. [Online]. Available: (2018). Salahshour, S., Ahmadian, A., A Pansera, B. & Ferrara, M. Uncertain inverse problem for fractional dynamical systems using perturbed collage theorem. *Commun. Nonlinear Sci. Numer. Simul.* **94**, 105553 (2021). <http://sourceforge.net>

Acknowledgments

This research was supported by the "Regional Innovation System & Education (RISE)" through the Seoul RISE Center, funded by the Ministry of Education (MOE) and the Seoul Metropolitan Government (2025-RISE-01-018-01). This work was also supported by the Ongoing Research Funding program (ORF-2025-395), King Saud University, Riyadh, Saudi Arabia.

Author contributions

Each author has made nearly equal contributions to developing the proposed model and preparing the manuscript. Vijay Kumar conceptualized the research problem, provided critical revisions to the manuscript and guided the research. Neelam Sharma led the development of the mathematical model and contributed to the theoretical as well as numerical analysis. Arfat Ahmad Khan assisted with the refinement of the graphs and figures present in the manuscript. Prashant Johri contributed to the literature review and assisted in manuscript editing. Farman Ali provided expert review and suggestions to improve the manuscript's structure and clarity. Ahmad Ali AlZubi provided insights into model implementation and contributed to the technical validation of results. The manuscript was primarily written by Neelam Sharma with input from all authors. All authors read and approved the final manuscript. All authors have reviewed and approved the final version of the manuscript and agreed to be accountable for its content.

Declarations

Competing interests

The authors declare no competing interests.

Disclosure statement

The author(s) reported no potential conflicts of interest.

Additional information

Correspondence and requests for materials should be addressed to V.K. or F.A.

Reprints and permissions information is available at www.nature.com/reprints.

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Open Access This article is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 International License, which permits any non-commercial use, sharing, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if you modified the licensed material. You do not have permission under this licence to share adapted material derived from this article or parts of it. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by-nc-nd/4.0/>.

© The Author(s) 2025