Journal of Systems Engineering and Electronics
Vol. 33, No. 2, April 2022, pp.360 – 369

Grey-based approach for estimating software reliability under nonhomogeneous Poisson process

LIU Xiaomei[1,2] and XIE Naiming[1,*]

1. College of Economics and Management, Nanjing University of Aeronautics and Astronautics, Nanjing 211106, China;
2. College of Science, Jiujiang University, Jiujiang 332005, China

Abstract: Due to the randomness and time dependence of the factors affecting software reliability, most software reliability models are treated as stochastic processes, and the non-homogeneous Poisson process (NHPP) is the most used one. However, the failure behavior of software does not follow the NHPP in a statistically rigorous manner, and the pure random method might be not enough to describe the software failure behavior. To solve these problems, this paper proposes a new integrated approach that combines stochastic process and grey system theory to describe the failure behavior of software. A grey NHPP software reliability model is put forward in a discrete form, and a grey-based approach for estimating software reliability under the NHPP is proposed as a nonlinear multi-objective programming problem. Finally, four grey NHPP software reliability models are applied to four real datasets, the dynamic R-square and predictive relative error are calculated. Comparing with the original single NHPP software reliability model, it is found that the modeling using the integrated approach has a higher prediction accuracy of software reliability. Therefore, there is the characteristics of grey uncertain information in the NHPP software reliability models, and exploiting the latent grey uncertain information might lead to more accurate software reliability estimation.

Keywords: software reliability model, stochastic process, uncertainty system, non-homogeneous Poisson process, grey system theory.

DOI: 10.23919/JSEE.2022.000038

1. Introduction

With the rapid development of information technology and the increasing scale of software system, the research of software reliability becomes necessary. Unlike hardware reliability systems, there is no physical mechanism for software failure. The development and design of software are all made by people, software inevitably occurs failure. In order to ensure the quality of the software, large numbers of carefully selected testing cases are used to expose the faults as quickly and effectively as possible. How much testing can make the software achieve acceptable quality depends on the evaluation technology of software reliability. Therefore, it is very important to evaluate software reliability for software engineer.

Software reliability evaluation is usually completed by software reliability modeling, which is to estimate software reliability according to the observed failure data by the statistical method. The research on software reliability modeling started from the Jelinski-Moranda (J-M) model [1]. Since the 1970s, software reliability modeling has developed rapidly, and many well-known models of different viewpoints have emerged. Stochastic process is one of the main frameworks for developing software reliability models. Nelson data domain model [2], Musa execution time model [3], and the Goel-Okumoto (G-O) failure counting model [4], were all presented based on the non-homogenous Poisson process (NHPP). Among these models, the G-O model is the most famous model. It assumes that the fault detection rate is a constant and the debugging is perfect. Although this hypothesis deviates from the real testing process greatly, it is an important reference for the follow-up research. Many software reliability models are proposed based on the G-O model by considering various elements that affect software reliability, such as incomplete debugging, fault detection rate, testing-effort, and change point [5−8]. Markov processes are also used to software reliability modeling by setting the transition probability between states as the input parameter. Jelinski et al. [9] first used Markov process to software reliability estimation. Kremer [10] studied software reliability as birth and death process. Qu [11] analyzed software reliability in the air traffic control system using

Manuscript received December 28, 2020.
*Corresponding author.
This work was supported by the National Natural Science Foundation of China (71671090), the Fundamental Research Funds for the Central Universities (NP2020022), the Qinglan Project of Excellent Youth or Middle-Aged Academic Leaders in Jiangsu Province.

Markov processes. Another family of software reliability models is Bayesian model. The first Bayesian inference software reliability model is Littlewood-Verrall model [12]. Other different Bayesian inference software reliability models were proposed under different assumptions of prior distribution [13–15]. Software reliability models based on time-series forecasting method are also widely studied. Singpurwda [16] first used time series analysis for software reliability estimation. Ho et al. [17] studied reliability prediction of repairable system based on the auto-regressive moving average (ARMA) model. Besides, combination models with other methods are presented to improve the prediction. Aly et al. [18] used genetic programming (GP) to build models to predict the number of errors. Jayadeep et al. [19] used a hybrid method composed of auto-regressive integrated moving average (ARIMA) and neural network to predict the software reliability. In recent years, the research on software reliability has developed to machine learning. Karunanithi et al. [20] first used neural network to research software reliability. Now, neural network has been applied in software reliability estimation for nearly 30 years, and numerous different machine learning algorithms have been explored. Qi [21] evaluated software reliability based on the back-propagation neural network. Kiran et al. [22] created a software reliability model from a heterogeneous ensemble of several different neural networks. Roy et al. [23] proposed an artificial neural network for software reliability modeling by a novel particle swarm optimization algorithm. Chaos theory is also used in software reliability modeling. Zou et al. [24] studied three well known software reliability models by extracting a fractal dimension. Shao [25] created a software reliability model to find reliable estimates of chaotic invariants. Yazdanbakhsh et al. [26] employed the nonlinear time series analysis to test for chaotic behavior in software reliability models.

Above reviews show the software reliability research methods are varied, and there has been no consensus on the mechanism of software failure. In general, the software reliability models based on the NHPP are the most widely studied because of the simple mathematical structure and the easily implementation. However, software internal structure is highly complex, and the NHPP software reliability models sometimes occur a serious bias from the actual value. Cai [27–29] has conducted meaningful research on the problems of the NHPP software reliability models. He verified that the software reliability behavior does not follow the NHPP in a statistically rigorous manner and pointed out the modeling framework of software reliability should not only consider the stochastic as a form of uncertainty, but also consider other forms of uncertainty. Thus, he proposed a fuzzy software reliability model by setting software failure time as a fuzzy variable.

The viewpoint of Cai does not adopt any assumptions related to the probability distribution and is to use another uncertainty theory (fuzzy set) instead of randomness to construct the software reliability model framework. Considering that a single uncertain system theory may not be enough to describe the failure behavior of software, it may be better to estimate software reliability with various uncertainty theories at the same time. This paper investigates the characteristics of grey uncertain information in NHPP software reliability models and proposes an integrated approach combining stochastic process and grey system theory to estimate software reliability.

The main work of this paper lies in the following two aspects:

(i) Several grey NHPP software reliability models are proposed;

(ii) A grey-based approach for estimating software reliability under the NHPP is presented.

This paper is organized as follows: Section 2 reviews software reliability estimation methods based on the NHPP. Section 3 introduces grey system theory and the related work on its applications in software reliability estimation. Section 4 proposes several grey NHPP software reliability models. Section 5 presents a grey-based approach for estimating software reliability under the NHPP. Section 6 tests the effectiveness of the grey-based estimating approach under the NHPP through four real datasets. Section 7 makes conclusions.

2. Software reliability estimation based on the NHPP

Software reliability model based on the NHPP describes software failure by a counting process and assumes the occurrence of failures obeys the nonhomogeneous Poisson process.

Denote $\{N(t), t \geqslant 0\}$ as the actual number of failures by time t, a counting process $\{N(t), t \geqslant 0\}$ is called the NHPP if the following assumptions are satisfied [30].

(i) $N(0)=0, \{N(t), t \geqslant 0\}$ is an independent incremental process.

(ii) $P\{N(t+\Delta t)-N(t) \geqslant 2\}=o(\Delta t)$.

(iii) $P\{N(t+\Delta t)-N(t)=1\}=\lambda(t)\cdot\Delta t+o(\Delta t)$.

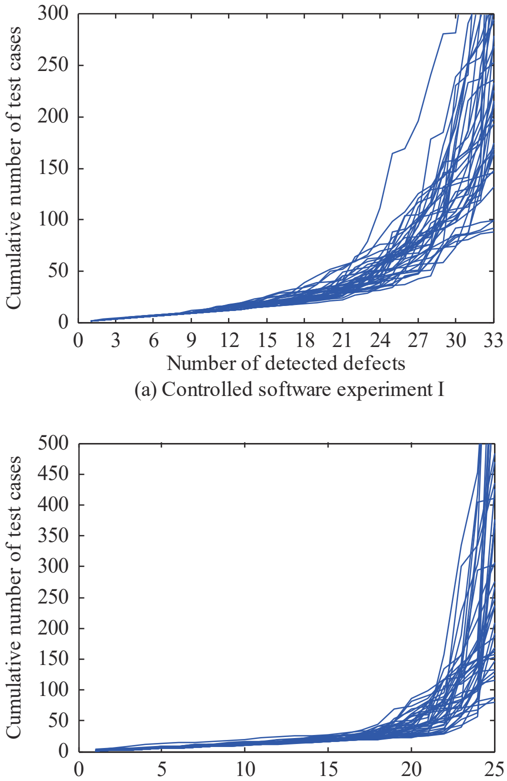
(iv) $P\{N(t+s)-N(s)=n\}=\dfrac{[m(t+s)-m(s)]^n}{n!}\mathrm{e}^{-(m(t+s)-m(s))}$,

where $m(t)$ is called the cumulative number of failures and expressed as $m(t)=\int_0^t \lambda(s)\mathrm{d}s$, $\lambda(t)$ is the number of failures of unit time at t.

According to (iv), the reliability in $[t, t+x)$ can be calculated by

$$R(x|t) = P\{N(t+x) - N(t) = 0\} = e^{-(m(t+x)-m(t))}. \quad (1)$$

Equation (1) shows the software reliability in specified time can be estimated with the cumulative function of failures $m(t)$. Therefore, an important step in the NHPP software reliability models is to establish the cumulative function of failures $m(t)$ by the failure data collected in the testing process. However, the failure data collected is only to represent a sample path, and the differences between different sample paths are significant. We adopt the simulation results of 40 trails of Cai's two controlled software experiments to describe this phenomenon. The detailed data can be seen in [27]. As shown in Fig. 1, the sample paths are very close to each other in the early stage, but become more and more dispersed in the later stage.



(a) Controlled software experiment I

(b) Controlled software experiment II

Fig. 1 Analysis of 40 trials from Cai's control experiments

He et al. [31] has pointed out software reliability can only be evaluated when software producers think software products are stable and mature, that is to say, the real sample paths of software failure are scattered with each other. Besides, the collected failure data is discrete, and the accuracy of extrapolation prediction cannot be guaranteed by pure simulation of failure data by using the continuous function $m(t)$. Therefore, it may not be able to describe the failure behavior of software only according to the NHPP software reliability model. In view of this, we introduce grey system theory to the NHPP software reliability modeling and not to pursue higher fitting accuracy. We assume software failure approximately follows the NHPP, and want to know whether there is the grey uncertain information in software system reliability.

3. Grey system theory and its related works on software reliability estimation

Grey system theory, like probability statistics, fuzzy mathematics and rough set theory, is an approach for studying uncertain systems. It was first proposed by Chinese scholar Professor Deng [32]. Different from other uncertainty theories, grey system theory focuses on the uncertainty problems of small data sets and poor information. One of the main tasks of grey system theory is to uncover the mathematical relationships or the change laws of system variables based on the available data, in which each stochastic variable is regarded as a grey quantity changing within a fixed region or within a certain time frame.

In essence, grey system theory is to seek an organic equilibrium between quantitative predictions and qualitative analyses. The main adopted method is to construct a suitable accumulated operator to preprocess the data to eliminate the influence of data sequence distortion before quantitative modeling analysis. For example, as shown in Fig. 2, the original data sequences do not show any definite regularity, and after processing by different accumulated operators, they show different change trend.

Up to now, grey system theory has been applied widely and has produced more effectiveness by combining with other approaches [33−36]. Meanwhile, grey information modeling is also used to estimate software reliability [37−43]. Gao et al. [37] first presented a software reliability model based on the GM(1,1) model of grey system theory. Ye et al. [38] and Mei [39] used Gao's method to evaluate software reliability. Zhang et al. [40] evaluated software reliability by combing GM(1,1) model and neural network. Zhang et al. [41] proposed a software reliability model by combining support vector regression model and GM(1,1) model. Ma et al. [42] constructed a series of grey Markov chain models by combining GM(1,1) model and Markov chain. Huang [43] studied grey fitting problems of software failure data and proposed the multistep prediction algorithm. These related works show two common points. One is that the adopted technology of grey system theory is only limited to the GM(1,1) model. The other is that the presented combination models are only to modify the residual errors produced by the GM(1,1) model.

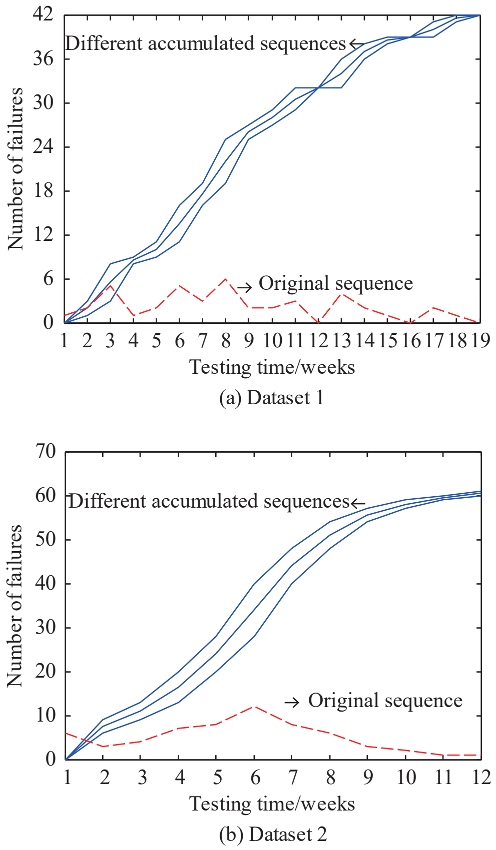


Fig. 2 Trend analysis of grey accumulated sequences

However, the analytical solution of GM(1,1) model is the monotonous exponential function, and it cannot simulate the software failure data with obvious nonlinear characteristics. Therefore, to better describe the failure behavior of software, it is necessary to extend the GM(1,1) model based on the characteristics of software failure data.

4. Grey software reliability models based on the NHPP

Due to the discreteness of data and the incompleteness of information, grey system model is established in the form of discrete difference equation representing approximately the continuous model. Therefore, we first proposed several grey software reliability models by discretizing the NHPP models based on the grey modeling mechanism. For illustrations, four traditional NHPP software reliability models are applied here, including the delayed S-shaped model [44], the inflection S-shaped model [45], the Yamada exponential model, and the Yamada Rayleigh model [46]. These models are widely used and have a great influence on the development of software reliability research.

4.1 Grey delayed S-shaped software reliability model

The cumulative function of failures $m(t)$ of the delayed S-shaped software reliability model is expressed as

$$m(t) = a(1-(1+bt)\mathrm{e}^{-bt}),\ \ a>0; b>0. \tag{2}$$

Its differential equation can be represented as

$$\frac{\mathrm{d}m(t)}{\mathrm{d}t} = \frac{b^2 t}{1+bt}(a-m(t)). \tag{3}$$

According to the grey modeling mechanism and using the undetermined grey background value coefficient θ, the discrete delayed S-shaped software reliability model is defined as

$$m(t_{k+1}) - m(t_k) = \frac{b^2 t_k}{1+bt_k}(a-(\theta m(t_k)+(1-\theta)m(t_{k+1}))). \tag{4}$$

Assume $m(t_k) = n_k (k=1,2,\cdots,N)$, and n_k denotes the actual defects found by time t_k, N denotes the total number of observed intervals. Thus, (4) becomes the following form:

$$n_{k+1} - n_k = \frac{b^2 t_k}{1+bt_k}(a-(\theta n_k+(1-\theta)n_{k+1})). \tag{5}$$

Transform (5) to obtain the following discrete equations:

$$a\cdot\frac{b^2 t_k}{1+bt_k} - (\theta n_k+(1-\theta)n_{k+1})\cdot\frac{b^2 t_k}{1+bt_k} = n_{k+1}-n_k. \tag{6}$$

Equation (6) is called the grey delayed S-shaped software reliability model.

4.2 Grey inflection S-shaped software reliability model

The cumulative function of failures $m(t)$ of the inflection S-shaped software reliability model is expressed as

$$m(t) = a\frac{1-\mathrm{e}^{-bt}}{1+c\mathrm{e}^{-bt}},\ \ a>0; b>0; c>0. \tag{7}$$

Its differential equation can be represented as

$$\frac{\mathrm{d}m(t)}{\mathrm{d}t} = \frac{b}{1+c\mathrm{e}^{-bt}}(a-m(t)). \tag{8}$$

Similar to Subsection 4.1, the discrete inflection S-shaped software reliability model is defined as

$$m(t_{k+1}) - m(t_k) = \frac{b}{1+c\mathrm{e}^{-bt_k}}(a-(\theta m(t_k)+(1-\theta)m(t_{k+1}))) \tag{9}$$

Transform (9) to obtain

$$a\cdot\frac{b}{1+c\mathrm{e}^{-bt_k}} - (\theta n_k+(1-\theta)n_{k+1})\cdot\frac{b}{1+c\mathrm{e}^{-bt_k}} = n_{k+1}-n_k. \tag{10}$$

Equation (10) is called the grey inflection S-shaped software reliability model.

4.3 Grey Yamada exponential software reliability model

The cumulative function of failures $m(t)$ of the Yamada exponential software reliability model is expressed as

$$m(t)=a(1-\mathrm{e}^{-b(1-\mathrm{e}^{-ct})}),\ \ a>0;b>0;c>0. \tag{11}$$

Its differential equation can be represented as

$$\frac{\mathrm{d}m(t)}{\mathrm{d}t}=bc\mathrm{e}^{-ct}(a-m(t)). \tag{12}$$

Similar to Subsection 4.1, the discrete Yamada exponential software reliability model is defined as

$$m(t_{k+1})-m(t_k)=bc\mathrm{e}^{-ct_k}(a-(\theta m(t_k)+(1-\theta)m(t_{k+1}))). \tag{13}$$

Transform (13) to obtain

$$a\cdot bc\mathrm{e}^{-ct_k}-(\theta n_k+(1-\theta)n_{k+1})\cdot bc\mathrm{e}^{-ct_k}=n_{k+1}-n_k. \tag{14}$$

Equation (14) is called the grey Yamada exponential software reliability model.

4.4 Grey Yamada Rayleigh software reliability model

The cumulative function of failures $m(t)$ of the Yamada Rayleigh software reliability model is expressed as

$$m(t)=a(1-\mathrm{e}^{-b(1-\mathrm{e}^{-ct^2/2})}),\ \ a>0;b>0;c>0. \tag{15}$$

Its differential equation can be represented as

$$\frac{\mathrm{d}m(t)}{\mathrm{d}t}=bct\mathrm{e}^{-ct^2/2}(a-m(t)). \tag{16}$$

Similar to Subsection 4.1, the discrete Yamada Rayleigh software reliability model is defined as

$$m(t_{k+1})-m(t_k)=bct_k\mathrm{e}^{-\frac{ct_k^2}{2}}(a-(\theta m(t_k)+(1-\theta)m(t_{k+1}))). \tag{17}$$

Transform (17) to obtain

$$a\cdot bct_k\mathrm{e}^{-\frac{ct_k^2}{2}}-(\theta n_k+(1-\theta)n_{k+1})\cdot bct_k\mathrm{e}^{-\frac{ct_k^2}{2}}=n_{k+1}-n_k. \tag{18}$$

In the same way, (18) is called the grey Yamada Rayleigh software reliability model.

Equations (6), (10), (14), and (18) are examples of several grey software reliability models proposed in this paper. It can be seen that they are discrete difference equations and approximately equal to the original continuous NHPP software reliability models. Therefore, these grey software reliability models still remain some characteristics of the NHPP, and the uniqueness is that they may simulate small and discrete data more realistically.

5. Grey-based approach for estimating software reliability under NHPP

Software failures may be highly correlated in the early stage of testing, while the number of failures would be closer to a constant in the later stages. So a single method of uncertain systems may not be enough to deal with the software failure process. This paper proposed a different approach than previous and describe the failure behavior of software based on multiple uncertainty theories of stochastic process and grey system. Specifically, it is assumed that the failure data not only has the characteristics of stochastic process, but also has the behavior characteristics of grey system.

In addition, with the development of software testing, the ultimate number of remaining failures will be reduced, and the amount of data will not be adequate. Such that, for the NHPP software reliability models, the maximum likelihood estimate (MLE) may not work well. Hirose [47] presented a typical example in which the predicted total failures emerge serious deviations. Such phenomena also occurred in other cases [48,49]. In view of this, the model parameters are estimated by the nonlinear least square principle in this paper. Furthermore, different models and different training data may result in different grey background value coefficients, so the grey background value coefficient θ is also set as a variable in the interval $[0,1]$ to search for the best coefficient.

Next, we will take the grey Yamada Rayleigh model as an example to describe the computational steps of the grey-based approach for estimating software reliability under the NHPP, and the other three grey models are similar.

The grey-based estimating approach under the NHPP can be treated as a nonlinear multi-objective programming problem, and the grey Yamada Rayleigh software reliability modeling under the NHPP can be expressed as follows:

$$\begin{cases}\min\displaystyle\sum_{k=1}^{N}(\hat{m}(t_k)-n_k)^2\\ \min\displaystyle\sum_{k=1}^{N-1}(\hat{x}(t_k)-(n_{k+1}-n_k))^2\end{cases}$$

$$\text{s.t.}\begin{cases}\hat{m}(t_k)=a(1-\mathrm{e}^{-b(1-\mathrm{e}^{-\frac{ct_k^2}{2}})})\\ \hat{x}(t_k)=abct_k\mathrm{e}^{\frac{-ct_k^2}{2}}-(\theta n_k+(1-\theta)n_{k+1})\cdot bct_k\mathrm{e}^{-\frac{ct_k^2}{2}}\\ a>0,b>0,c>0,0\leqslant\theta\leqslant 1\end{cases} \tag{19}$$

where n_k denotes as the actual defects found by time t_k. There are four parameters a,b,c,θ that need to be estimated. We will estimate these four parameters in two stages. The computational steps can be summarized as follows:

Step 1 Input the failure data into the grey Yamada Rayleigh software reliability model (i.e., (18)).

Step 2 Calculate the symbolic solutions of parameters a,b according to the least square principle.

Step 3 Substitute the symbolic solutions of parameters a, b into the Yamada Rayleigh software reliability model (i.e., (15)).

Step 4 Calculate the optimal solutions of parameters c, θ according to the nonlinear least square estimate.

Step 5 Return to (18) to find the optimal solutions of parameters a, b using the parameters c, θ estimated in Step 4.

Step 6 Substitute all parameters into the Yamada Rayleigh software reliability model to calculate the related software reliability measurements.

The nonlinear optimization in Step 4 needs to be done by some computer software. In this paper, the optimization is realized using the genetic algorithm (GA) by programming in Matlab.

6. Numerical experiments

This section is to test the effectiveness of the grey-based approach for estimating software reliability under the NHPP. Four numerical experiments are to predict dynamically the total number of failures in the software testing process using four datasets come from Wood [50]. Meanwhile, the single NHPP software reliability model is estimated by the MLE and is used for comparisons. R-square is calculated dynamically to judge the fitting accuracy, and the relative error of prediction is calculated dynamically to compare the forecasting performance. Moreover, the predictive validity is judged by plotting the relative error of the normalized test time.

The relative error of prediction is defined as follows: Assume the actual number of defects observed is M by the end of test time T. Use the failure data up to time $t_k (t_k \leqslant T)$ to get the predicted number of failures $m(T)$ by T, then the ratio $(m(T)-M)/M$ is called the relative error. This procedure repeated with various t_k yields the dynamical relative error.

Dataset 1 There are 19 test weeks and the actual number of defects observed is 42 (that is, $T=19$, $M=42$). The predicted numbers of defects and the R-square under different test time are calculated by the single delayed S-shaped model and by the grey delayed S-shaped modeling under the NHPP. Table 1 listed these results, and Fig. 3 plots the relative error of prediction and the R-square against the different percentage of data points.

As shown in Table 1, both the single delayed S-shaped model and the grey delayed S-shaped modeling under the NHPP have high R-square, the minimum R-square is over 0.953, and most of them are over 0.97, they can all simulate Dataset 1 well. However, the prediction obtained by the single delayed S-shaped model is poor at the beginning, especially at the eighth week. It is not until the 16th week that the prediction is stable. On the contrary, the prediction obtained by the grey delayed S-shaped modeling under the NHPP is relatively more stable, and converged after the ninth week. It is clearly seen from Fig. 3 that the R-square curves of the single delayed S-shaped model and the grey delayed S-shaped modeling under the NHPP are straight and close to 1, and the relative error of the grey delayed S-shaped modeling under the NHPP approaches 0 after 45% of data points. However, the relative error of the single delayed S-shaped model fluctuates and tends to 0 until 85% of the data points.

Table 1 Comparison results for different estimates of delayed S-shaped model on Dataset 1

Test weeks	Defects found	Delayed S-shaped		Grey delayed S-shaped	
		Predicted	R^2	Predicted	R^2
1	1	–	–	–	–
2	3	–	–	–	–
3	8	–	–	–	–
4	9	–	–	–	–
5	11	15.9835	0.9616	17.8443	0.9608
6	16	38.1267	0.9533	33.1945	0.9579
7	19	37.1073	0.9732	35.3610	0.9740
8	25	61.9431	0.9717	44.2515	0.9720
9	27	47.4860	0.9827	42.7439	0.9791
10	29	42.8805	0.9852	40.4875	0.9815
11	32	44.7539	0.9893	42.1101	0.9870
12	32	38.6972	0.9846	40.6727	0.9874
13	36	44.1493	0.9902	41.8980	0.9898
14	38	44.4800	0.9917	42.5610	0.9913
15	39	43.3726	0.9929	42.1856	0.9918
16	39	41.4597	0.9917	41.1419	0.9907
17	41	42.7744	0.9939	41.9332	0.9930
18	42	42.8261	0.9945	42.0690	0.9938
19	42	42.0000	0.9941	41.6576	0.9934

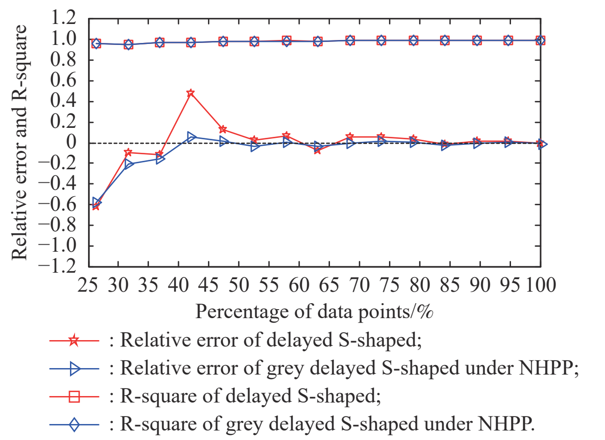


Fig. 3 R-square and relative error curves of different estimates of delayed S-shaped model on Dataset 1

Dataset 2 There are 20 test weeks and the actual number of defects observed is 100 (that is, $T = 20$, $M = 100$). The predicted numbers of defects and the R-square estimated by the single inflection S-shaped model and by the grey inflection S-shaped modeling under the NHPP for various test time are shown in Table 2. Similarly, Fig. 4 plots the relative error of prediction and the R-square against the different percentage of data points.

Table 2 Comparison results for different estimates of inflection S-shaped model on Dataset 2

Test weeks	Defects found	Inflection S-shaped		Grey inflection S-shaped	
		Predicted	R^2	Predicted	R^2
1	16	–	–	–	–
2	24	–	–	–	–
3	27	–	–	–	–
4	33	–	–	–	–
5	41	66.3894	0.9159	55.1628	0.9141
6	49	85.7874	0.9220	71.1501	0.9304
7	54	87.0057	0.9501	75.5131	0.9518
8	58	86.0565	0.9650	77.0651	0.9644
9	69	114.1563	0.9561	98.8765	0.9653
10	75	116.6969	0.9674	102.6451	0.9686
11	81	119.2928	0.9747	100.8727	0.9718
12	86	118.8128	0.9803	102.5417	0.9741
13	90	116.7822	0.9841	102.5982	0.9773
14	93	113.7481	0.9859	101.2632	0.9783
15	96	112.0970	0.9870	101.4722	0.9809
16	98	108.6041	0.9872	101.1054	0.9824
17	99	104.7258	0.9869	99.7897	0.9815
18	100	103.0323	0.9869	100.0392	0.9836
19	100	101.0997	0.9861	101.1259	0.9877
20	100	100.0000	0.9856	100.9800	0.9878

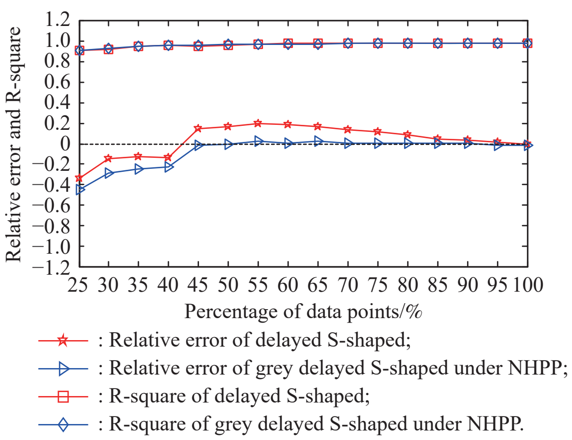


Fig. 4 R-square and relative error curves of different estimates of inflection S-shaped model on Dataset 2

Table 2 shows both the single inflection S-shaped model and the grey inflection S-shaped modeling under the NHPP still have high R-square, the minimum R-square is over 0.9141, and most of them are over 0.96. However, the prediction of the single inflection S-shaped model is approximately equal to the actual defects until the 19th week. In contrast, the prediction obtained by the grey inflection S-shaped modeling under the NHPP is much more accurate, and is almost stable after the ninth week. Similarly, Fig. 4 shows the R-squares of the single inflection S-shaped model and the grey inflection S-shaped modeling under the NHPP are all close to 1, and the relative error of the grey inflection S-shaped modeling under the NHPP approaches 0 after 45% of data points. However, the relative error of the single inflection S-shaped model seriously deviates from 0, and tends to 0 until 90% of the data points. Finally, the grey inflection S-shaped modeling under the NHPP converges earlier than the single inflection S-shaped model by 45% of data points.

Dataset 3 There are 19 test weeks and the actual number of defects observed is 120 (that is, $T = 19$, $M = 120$). The predicted numbers of defects and the R-square estimated by the single Yamada exponential model and by the grey Yamada exponential modeling under the NHPP are shown in Table 3. Meanwhile, Fig. 5 plots the relative error and the R-square against the different percentage of data points.

Table 3 Comparison results for different estimates of Yamada exponential model on Dataset 3

Test weeks	Defects found	Yamada exponential		Grey Yamada exponential	
		Predicted	R^2	Predicted	R^2
1	13	–	–	–	–
2	18	–	–	–	–
3	26	–	–	–	–
4	34	–	–	–	–
5	40	78.2824	0.9792	64.1298	0.7320
6	48	100.7217	0.9837	86.3526	0.6750
7	61	138.7355	0.9625	92.0219	0.7907
8	75	154.2108	0.9310	138.5518	0.8463
9	84	159.0036	0.9534	159.2504	0.8833
10	89	153.6955	0.9760	125.8609	0.9551
11	95	147.0843	0.9798	121.6978	0.9720
12	100	142.6860	0.9824	119.1437	0.9807
13	104	136.9456	0.9818	117.2671	0.9857
14	110	136.8509	0.9845	122.8409	0.9763
15	112	129.7455	0.9809	120.7351	0.9895
16	114	125.2981	0.9777	119.5513	0.9912
17	117	123.8234	0.9778	120.8430	0.9915
18	118	120.6101	0.9742	120.1751	0.9927
19	120	119.5196	0.9743	120.8021	0.9928

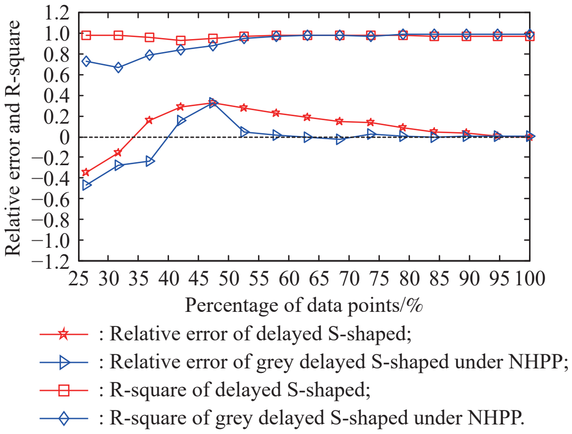


Fig. 5 R-square and relative error curves of different estimates of Yamada exponential model on Dataset 3

Table 3 indicates the R-square of the single Yamada exponential model is still high, and the prediction converges after the 17th week. By contrast, the R-square of the grey Yamada exponential modeling under the NHPP is not high at the beginning, but after the 11th week, both the R-square and the prediction accuracy all converge. The above analysis is clear in Fig. 5. The R-square curve of the single Yamada exponential model is a straight line close to 1, where its relative error seriously deviates from 0, and tends to 0 until the last data points. The R-square and the relative error of the grey Yamada exponential modeling under the NHPP have the same features, and the relative error converges after 50% of data points, earlier than the single Yamada exponential model by 40% of data points.

Dataset 4 There are 12 test weeks and the actual number of defects observed is 61 (that is, $T = 12$, $M = 61$). The predicted numbers of defects and the R-square estimated by the single Yamada Rayleigh model and by the grey Yamada Rayleigh modeling under the NHPP are shown in Table 4. Similarly, Fig. 6 plots the relative error and the R-square against the different percentage of data points.

In this dataset, the test week is relative small. Table 4 shows the R-square of the single Yamada Rayleigh model is high, and the predicted total defects is equal to the actual values after the ninth week. The difference is, the R-square of the grey Yamada Rayleigh modeling under the NHPP is not high at the beginning, but after the seventh week, the predicted number of failures is very close to the actual values, and the R-square also becomes high. Fig. 6 clearly shows the above results. The R-square curve of the single Yamada Rayleigh model is a straight line close to 1, and its relative error seriously deviates from 0. It is very obvious that the relative error of the grey Yamada Rayleigh modeling under the NHPP converges much earlier than that of the single Yamada Rayleigh model.

Table 4 Comparison results for different estimates of Yamada Rayleigh model on Dataset 4

Test weeks	Defects found	Yamada Rayleigh		Grey Yamada Rayleigh	
		Predicted	R^2	Predicted	R^2
1	6	–	–	–	–
2	9	–	–	–	–
3	13	–	–	–	–
4	20	–	–	–	–
5	28	44.0892	0.9253	39.4451	0.6206
6	40	83.8606	0.9556	66.1143	0.7546
7	48	76.2715	0.9767	61.0024	0.8990
8	54	70.4148	0.9841	63.3178	0.9516
9	57	63.9641	0.9850	61.6199	0.9739
10	59	62.0392	0.9847	61.4499	0.9816
11	60	60.8038	0.9838	61.1707	0.9856
12	61	60.4684	0.9857	61.5068	0.9872

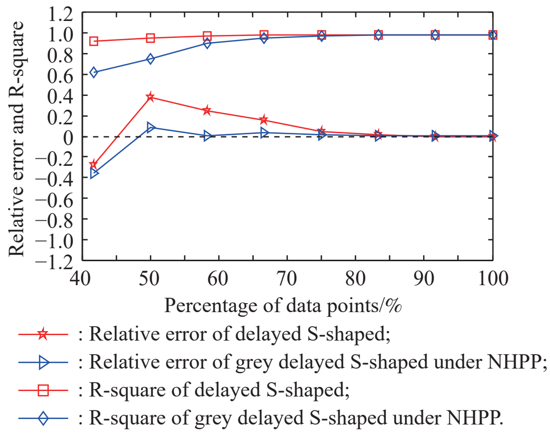


Fig. 6 R-square and relative error curves of different estimates of Yamada Rayleigh model on Dataset 4

7. Conclusions

Neither the single stochastic process model nor the combined model for correcting residual errors can completely solve the problems in software reliability estimation. In this paper, we investigate the characteristics of grey uncertain information in the NHPP software reliability models. A new grey-based approach for estimating software reliability under the NHPP is presented and is applied to predict the total number of failures in the software testing process. The results of numerical experiments indicate the predictive relative error of these single NHPP software reliability models having high R-square fluctuated greatly in the early and middle testing stages, and sometimes even in the last testing stage. In contrast, the new grey-based estimating approach under the NHPP effectively improves this problem and can predict the final stage much earlier although without higher fitting accuracy. Results also show that the new grey-based estimating approach under the NHPP can almost offer a more accurate prediction of the future total number of failures at 50% of the test process, and the shorter the software testing time, the more obvious the effectiveness (for example, the experiment of Dataset 4). Therefore, this paper verifies that the modeling method of combining randomness and greyness might be better than that of single randomness in software reliability estimation.

In addition, the grey-based approach for estimating software reliability can be applicable to all NHPP software reliability models. Therefore, the grey-based approach for estimating software reliability under the NHPP is of practical significance in predicting the behavior of software future failures in the early testing process.

References

[1] JELINSKI Z, MORANDA P. Statistical Computer Performance Evaluation, Rhode Island: Academic Press, 1972: 465–484.
[2] NELSON E. Estimating software reliability from test data. Microelectronics Reliability, 1978, 17(1): 67–73.
[3] MUSA J D. A theory of software reliability and its application. IEEE Press, 1975, SE-1(3): 312–327.
[4] GOEL A, OKUMOTO K. A time dependent error detection model for software reliability and other performance measures. IEEE Trans. on Reliability, 1979: 206–211.
[5] PHAM H. A general imperfect-software-debugging model with S-shaped fault-detection rate. IEEE Trans. on Reliability, 1999, 48: 169–175.
[6] HSU C J. Enhancing software reliability modeling and prediction through the introduction of time-variable fault reduction factor. Applied Mathematical Modelling, 2011, 35: 506–521.
[7] CHIU K C, HUANG Y S, HUANG I C. A study of software reliability growth with imperfect debugging for time-dependent potential errors. International Journal of Industrial Engineering-Theory Applications and Practice, 2019, 26(3): 376–393.
[8] CHATTERJEE S, SHUKLA A, PHAM H. Modeling and analysis of software fault detectability and removability with time variant fault exposure ratio, fault removal efficiency, and change point. Proc. of the Institution of Mechanical Engineers Part O-Journal of Risk and Reliability, 2019, 233(2): 246–256.
[9] MORANDA P B, JELINSKI Z. Software reliability predictions. International Federation of Automatic Control, 1975, 8(1): 218–224.
[10] KREMER W. Birth-death and bug counting. IEEE Trans. on Reliability, 1983, 32(1): 37–47.
[11] QU J. Software reliability analysis in air traffic control system. Proc. of the Integrated Communication in Navigation and Surveillance Conference, 2015.
[12] LITTLEWOOD B, VERRALL J L. A Bayesian reliability growth model for computer software. Journal of the Royal Statistical Society Series C, 1973, 22(3): 332–346.
[13] SINGPURWALLA N D. Determining an optimal time interval for testing and debugging software. IEEE Trans. on Software Engineering, 1991, 17: 313–319.
[14] PHAM L, PHAM H. A Bayesian predictive software reliability model with pseudo-failures. IEEE Trans. on Systems, Man, and Cybernetics, Part A: Systems and Humans, 2001, 31: 233–238.
[15] CAI B P, KONG X D, LIU Y H, et al. Application of Bayesian networks in reliability evaluation. IEEE Trans. on Industrial Informatics, 2019, 15(4): 2146–2157.
[16] SINGPURWDA N D. Estimating reliability growth (or deterioration) using time series analysis. Naval Research Logistics Quarterly, 1978, 25(1): 1–14.
[17] HO S L, XIE M. The use of ARIMA models for reliability forecasting and analysis. Computers & Industrial Engineering, 1998, 35(2): 213–216.
[18] ALY W, ALWESHAH M, ALDABBAS H. Evolution of software reliability growth models: a comparison of auto-regression and genetic programming models. International Journal of Computer Applications, 2015, 125(3): 8875–8887.
[19] JAYADEEP P, SHUKLA K K. A hybrid technique for software reliability prediction. Proc. of the India Software Engineering Conference, 2015: 139–146.
[20] KARUNANITHI N, MALAIYA Y K, WHITLEY D. Prediction of software reliability using neural networks. Proc. of the IEEE International Symposium on Software Reliability Engineering, 1991: 124–130.
[21] QI Y D. A BP neural network based hybrid model for software reliability prediction. Proc. of the International Conference on Computer Application and System Modeling, 2010.
[22] KIRAN N R, RAVI V. Software reliability prediction by soft computing techniques. Journal of Systems and Software, 2008, 81: 576–583.
[23] ROY P, MAHAPATRA G S, DEY K N. Forecasting of software reliability using neighborhood fuzzy particle swarm optimization based novel neural network. IEEE-CAA Journal of Automatica Sinica, 2019, 6(6): 1365–1383.
[24] ZOU F Z, LI C X. A chaotic model for software reliability. Chinese Journal of Computers, 2001, 3: 281–291. (in Chinese)
[25] SHAO C. Recovering chaotic properties from small data. IEEE Trans. on Cybernetics, 2014, 44: 2545–2556.
[26] YAZDANBAKHSH O, DICK S, REAY I, et al. On deterministic chaos in software reliability growth models. Applied Soft Computing, 2016, 49: 1256–1269.
[27] CAI K Y, HU D B, BAI C G. Does software reliability growth behavior follow a non-homogeneous Poisson process. Information & Software Technology, 2008, 50(12): 1232–1247.
[28] CAI K Y, WEN C Y, ZHANG M L. A critical review on software reliability modeling. Reliability Engineering & System Safety, 1991, 32(3): 357–371.
[29] CAI K Y, WEN C Y, ZHANG M L. A novel approach to software reliability modeling. Microelectronics Reliability, 1993, 33(15): 2265–2267.
[30] ROSS S M. Stochastic processes. Hoboken: Wiley, 1996.
[31] HE G W, WANG W. Software reliability. Beijing: National Defense Industry Press, 1998. (in Chinese)
[32] DENG J L. Grey system theory course. Wuhan: Huazhong University of Science and Technology Press, 1990. (in Chinese)
[33] LIU S F, YANG Y J, JEFFREY F. Grey data analysis: methods, models and applications. London: Springer, 2016.
[34] LIU X M, XIE N M. A nonlinear grey forecasting model with double shape parameters and its application. Applied Mathematical and Computation, 2019, 360: 203–212.
[35] SU B T, XIE N M, YANG Y J. Hybrid genetic algorithm based on bin packing strategy for the unrelated parallel workgroup scheduling problem. Journal of Intelligent Manufactur-

ing, 2021, 32(4): 957–969.
[36] XIE N M, SU B T, CHEN N L. Construction mechanism of whitenization weight function and its application in grey clustering evaluation. Journal of Systems Engineering and Electronics, 2019, 30(1): 121–131.
[37] GAO J Z, XU F Z, LIU B L. A new software reliability prediction model based on grey theory. Journal of Beijing University of Light Industry, 1989, 2: 29–33. (in Chinese)
[38] YE H, TANG Y H, ZHOU Z D, et al. Software reliability modeling by grey system theory. Measurement Technology and Intelligent Instruments, 1993, 2101: 1218–1221.
[39] MEI D H. Software reliability estimation in grey system theory. Proc. of the IEEE International Conference on Grey System and Intelligent Services, 2007: 299–302.
[40] ZHANG K, LI T, YU Y. The research of software reliability model based on the grey theory and neural network. Computer Applications and Software, 2009, 26(12): 34–36. (in Chinese)
[41] ZHANG D P, WANG S. Hybrid predication model for software reliability based on empirical mode decomposition method. Computer Engineering and Applications, 2013, 49(17): 24–29. (in Chinese)
[42] MA X F, WANG W B. Software detection algorithm robustness analysis based on the combination of correction model. Bulletin of Science and Technology, 2014, 30(2): 83–85. (in Chinese)
[43] HUANG X B. Grey fitting and multi-step prediction algorithm of software failure time series data. Intelligent Computer and Applications, 2017, 7(6): 9–13. (in Chinese)
[44] YAMADA S, OHBA M, OSAKI S. S-shaped reliability growth modeling for software error detection. IEEE Trans. on Reliability, 1983, 32(5): 475–484.
[45] YAMADA S, OHBA M, OSAKI S. S-shaped software reliability growth models and their applications. IEEE Trans. on Reliability, 1984, 33(4): 289–292.
[46] YAMADA S, OHTERA H, NARIHISA H. Software reliability growth models with testing-effort. IEEE Trans. on Reliability, 1986, 35(1): 19–23.
[47] HIROSE H. Estimation of the number of failures in the Weibull model using the ordinary differential equation. European Journal of Operational Research, 2012, 223: 722–731.
[48] HUANG C Y. Performance analysis of software reliability growth models with testing-effort and change-point. Journal of Systems and Software, 2005, 76: 181–194.
[49] KAPUR P K, PHAM H, ANAND S, et al. A unified approach for developing software reliability growth models in the presence of imperfect debugging and error generation. IEEE Trans. on Reliability, 2011, 60(1): 331–340.
[50] WOOD A. Predicting software reliability. IEEE Computer, 1996, 11: 69–71.

Biographies

LIU Xiaomei was born in 1981. She received her B.S. and M.S. degrees from Nanchang University in 2003 and 2006, respectively. Now, she is a lecturer at Jiujiang University. She has been pursuing her Ph.D. degree in Nanjing University of Aeronautics and Astronautics since 2018. Her research interests include grey information modeling and reliability system modeling.
E-mail: liuxiaomeijju@126.com.

XIE Naiming was born in 1981. He received his B.S., M.S., and PH.D. degrees from Nanjing University of Aeronautics and Astronautics in 2002, 2005 and 2008, respectively. Now, he is a full professor from Institute of Grey System Studies, and serves as Vice-Dean of College of Economics and Management at Nanjing University of Aeronautics and Astronautics. His major research interests include grey information modelling, dynamic system modelling and data analytics in smart manufacturing.
E-mail: xienaiming@nuaa.edu.cn