数理解析研究所講究録
第2078巻 2018年 141-146

離散化ソフトウェア信頼性モデルに基づいた信頼性評価尺度の区間推定
Interval Estimation of Software Reliability Assessment Measures Based on a Discretized Reliability Model

関西大学・総合情報学部 井上 真二
鳥取大学・大学院工学研究科 山田 茂

Shinji Inoue (Faculty of Informatics, Kansai University)
Shigeru Yamada (Graduate School of Engineering, Tottori University)

1 Introduction

This paper discusses a Bayesian approach for conducting interval estimation of software reliability based on a discretized software reliability growth model [3]. Considering a practical situation, we encourage the software development managers to use the interval estimation method when we do not obtain a sufficient number of software reliability data. However, the interval estimation needs to derive a probability distribution function for the parameter of interest. Further, it is very difficult to derive the probability distribution functions analytically even if we use the approximation approach using the asymptotic property.

Under such background, Kimura and Fujiwara [6] discussed a bootstrap software reliability assessment method of an incomplete gamma function-based software reliability growth model for estimating standard errors of the model parameters. Kaneishi and Dohi [5] discussed a parametric bootstrap method for software reliability assessment based on continuous-time nonhomogeneous Poisson process (NHPP) models. Inoue and Yamada [4] discussed a nonparametric bootstrapping approach for interval estimations of software reliability and optimal software shipping time based on a discretized software reliability growth model. The Bayesian approach is one of the useful approaches for obtaining the probability distributions of model parameters and software reliability assessment measures. For example, Okamura et al. [8] discussed Bayesian estimation for interval estimation of optimal software release time by using Markov chain Monte Carlo (MCMC) method.

This paper discusses a Bayesian estimation method for software reliability assessment based on a discretized NHPP model [3]. The discretized NHPP model conserves the basic properties of the continuous-time NHPP model and have good prediction and fitting performance for the actual data [3] because the discretized model has consistency with discrete fault count data collection activities. We conduct interval estimation of the model parameters and software reliability assessment measures by Bayesian estimation approach. Finally, we show numerical examples of our approach in this paper by using actual fault-count data, and show the results of interval estimations for the model parameters and the software reliability assessment measures based on the notion of credible interval.

2 Discretized NHPP Model

Now we define a discrete counting process $\{N_i, i = 0, 1, 2, \cdots\}$ representing the cumulative number of faults detected up to i-th testing-period. And we can say that the discrete counting process $\{N_i, i = 0, 1, 2, \cdots\}$ follows a discrete-time NHPP, which is the discrete analog of the continuous-time NHPP [7, 9, 10], if the discrete counting process has the following property:

$$\Pr\{N_i = x \mid N_0 = 0\} = \frac{\{\Lambda_i\}^x}{x!}\exp[-\Lambda_i] \qquad (i, x = 0, 1, 2, \cdots). \tag{1}$$

In Eq. (1), $\Pr\{A\}$ means the probability of event A, Λ_i is the mean value function of the discrete-time NHPP. The mean value function, Λ_i, also represents the expected cumulative number of faults detected up to i-th testing-period.

Let H_i denote a mean value function following a discretized exponential software reliability growth model [3]. The discretized exponential software reliability growth model is derived from the following difference equation:

$$H_{i+1} - H_i = \delta b\,(a - H_i)\,, \tag{2}$$

which is the discrete analog of the differential equation of the corresponding continuous-time exponential software reliability growth model [1]. In Eq. (2), a is the expected total number of potential faults to be detected in an infinitely long duration or the expected initial fault content, and b the fault detection rate per one fault. Regarding the discretization method, we use the Hirota's bilinearization methods [2] for conserving the property of the continuous-time exponential software reliability growth model. Solving the above integrable difference equation in Eq. (2), we can obtain an exact solution H_i in Eq. (2) as

$$\Lambda_i \equiv H_i = a\left[1 - (1-\delta b)^i\right] \qquad (a > 0,\ b > 0)\,, \tag{3}$$

where δ represents the constant time-interval. As $\delta \to 0$, Eq. (3) converges to the exact solution of the original continuous-time exponential software reliability growth model.

The discretized exponential software reliability growth model in Eq. (3) has two parameters, a and δb, which have to be estimated by using actual data. In the point estimation, the parameter estimations of a and δb, $\widehat{a}$ and $\widehat{\delta b}$, can be obtained by the following procedure using the method of least-squares. Suppose we have observed fault counting data $\mathcal{D} \equiv (i, y_i)(i = 1, 2, \cdots, n)$, where y_i represents the cumulative number of faults detected up to i-th testing-period. We can derive the following regression equation from Eq. (2):

$$c_i = \alpha + \beta d_i, \tag{4}$$

where

$$\left\{\begin{array}{ll} c_i & = H_{i+1} - H_i \equiv y_{i+1} - y_i, \\ d_i & = H_i \equiv y_i, \\ \alpha & = \delta a b, \\ \beta & = -\delta b. \end{array}\right. \tag{5}$$

Based on the regression analysis, we can estimate $\widehat{\alpha}$ and $\widehat{\delta b}$, which are the estimations of α and δb in Eq. (4). Then, the parameter estimations, $\widehat{a}$ and $\widehat{\delta b}$, can be obtained as

$$\left\{\begin{array}{l} \widehat{a} \ = -\widehat{\alpha}/\widehat{\beta}, \\ \widehat{\delta b} = -\widehat{\beta}, \end{array}\right. \tag{6}$$

respectively. It is worth noting that c_i in Eq. (4) is independent of δ because δ is not used in calculating c_i as showing Eq. (5). Hence, we can obtain the same parameter estimates $\widehat{a}$ and $\widehat{b}$, respectively, when we choose any constant value of δ [3].

Regarding software reliability assessment measures, the discrete version of the expected number of remaining faults, M_i, represents the expected number of undetected faults in the software system at arbitrary testing-period. Then, we have

$$\begin{aligned} M_i \equiv \mathrm{E}[N_\infty - N_i] &= a - \Lambda_i \\ &= a(1-\delta b)^i \end{aligned} \tag{7}$$

if we assume that N_i follows the discrete-time NHPP with mean value function H_i in Eq. (3). In Eq. (7), $\mathrm{E}[N_i]$ represents the expectation of N_i. And the discrete-time software reliability function, $R(i,h)$, is defined as the probability that a software failure does not occur in the time-interval $(i, i+h]$ $(h = 1, 2, \cdots)$ given that the testing has been going up to the i-th testing-priod. Then, we have

$$\begin{aligned} R(i,h) &\equiv \Pr\{N_{i+h} - N_i = 0 \mid N_i = x\} \\ &= \exp[-\{\Lambda_{i+h} - \Lambda_i\}] \\ &= \exp[-H_h(1-\delta b)^i]. \end{aligned} \tag{8}$$

3 Bayesian Estimation

The point estimations of the parameters in Eq. (3) can be obtained by the linear regression approach as discussed in Section 2. This implies that the parameter a and δb are estimated by the method of maximum-likelihood assuming $c_i \sim N(\alpha + \beta d_i, \sigma^2)$, which indicates c_i follows the normal distribution with mean $\alpha + \beta d_i$ and standard deviation σ^2. The likelihood function for $\mathcal{D}$ is derived as

$$\begin{aligned} p(\mathcal{D}|\alpha,\beta,\sigma^2) &= \prod_{i=1}^{n} \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left[-\frac{(c_i - \alpha - \beta d_i)}{2\sigma^2}\right] \\ &\propto \exp\left[-\frac{n(\alpha - \widehat{\alpha})^2 + \sum_{i=1}^{n}(\beta - \widehat{\beta})^2 d_i^2}{2\sigma^2}\right]. \end{aligned} \tag{9}$$

Now, we derive the posterior distribution of α based on the Bayes' theorem. The Bayes' theorem gives us the following relationship between the prior and posterior:

$$p(\alpha|\beta,\sigma^2,\mathcal{D}) \propto p(\mathcal{D}|\alpha,\beta,\sigma^2)p(\alpha), \tag{10}$$

when $\mathcal{D}$, β and σ^2 are given. Assuming $\alpha \sim N(\mu_\alpha, \tau_\alpha^2)$, we can derive the posterior for α as

$$\alpha|\beta,\sigma^2,\mathcal{D} \sim N\left(\frac{n\widehat{\alpha}\tau_\alpha^2 + \sigma^2\mu_\alpha}{\tau_\alpha^2 n + \sigma^2}, \frac{\sigma^2\tau_\alpha^2}{n\tau_\alpha^2 + \sigma^2}\right). \tag{11}$$

The posterior of β given α, σ^2 and $\mathcal{D}$ is derived as

$$p(\beta|\alpha,\sigma^2,\mathcal{D}) \propto p(\mathcal{D}|\alpha,\beta,\sigma^2)p(\beta). \tag{12}$$

Then,

$$\beta|\alpha,\sigma^2,\mathcal{D} \sim N\left(\frac{\tau_\beta^2\widehat{\beta}\sum_{i=1}^{n} d_i^2 + \sigma^2\mu_\beta}{\tau_\beta^2\sum_{i=1}^{n} d_i^2 + \sigma^2}, \frac{\sigma^2\tau_\beta^2}{\tau_\beta^2\sum_{i=1}^{n} d_i^2 + \sigma^2}\right), \tag{13}$$

where the prior of β is assumed that $\beta \sim N(\mu_\beta, \tau_\beta^2)$. Regarding the posterior of σ^2, we apply an inverse gamma distribution to the prior because the inverse gamma distribution is the conjugate distribution of the variance for data following the normal distribution. The inverse gamma distribution is given by

$$IG\left(\frac{r_0}{2}, \frac{s_0}{2}\right) = \frac{(s_0/2)^{r_0/2}}{\Gamma(r_0/2)}(\sigma^2)^{-\frac{r_0}{2}+1} \exp\left[-\frac{s_0}{2\sigma^2}\right], \tag{14}$$

where $r_0/2 > 0$ and $s_0/2 > 0$. The posterior of σ^2 given α, β and $\mathcal{D}$ follows $p(\sigma^2|\alpha,\beta,\mathcal{D}) \propto p(\mathcal{D}|\alpha,\beta,\sigma^2)p(\sigma^2)$. Then, the posterior is derived as

$$\sigma|\alpha,\beta,\mathcal{D} \sim IG\left(\frac{n + r_0}{2}, \frac{\sum_{i=1}^{n}(y_i - \alpha - \beta d_i) + s_0}{2}\right) \tag{15}$$

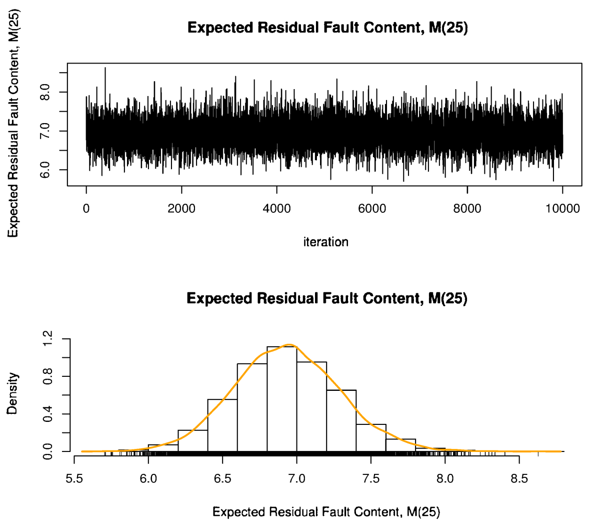

Fig 1 : The MCMC samples and posterior distribution for the expected number of remaining faults at $i = 25$, M_{25}.

because the likelihood function in terms of σ^2 is

$$p(\mathcal{D}|\alpha, \beta, \sigma^2) \propto (\sigma^2)^{-n/2} \exp\left[-\frac{\sum_{i=1}^{n}(c_i - \alpha - \beta d_i)^2}{2\sigma^2}\right]. \tag{16}$$

The Gibbs sampling method, which is one of the MCMC methods, is used for obtaining the posterior distribution of each parameter. When software fault-count data $\mathcal{D}$ is obtained, the Gibbs sampler is concretely given by the following steps:

(Step 1) Estimate $\widehat{\alpha}$ and $\widehat{\beta}$ from the observed data $\mathcal{D}$ by using the regression analysis discussed in Section 2.

(Step 2) Set $\widehat{\alpha}$, $\widehat{\beta}$ and $\sigma^2 = 1$ as $(\alpha^{(1)}, \beta^{(1)}, \sigma^{2(1)})$, which are the initial values of α, β and σ^2.

(Step 3) Generate $\alpha^{(r)}$ from $p(\alpha^{(r)}|\beta^{(r-1)}, \sigma^{2(r-1)}, \mathcal{D})$ in Eq. (11).

(Step 4) Generate $\beta^{(r)}$ from $p(\beta^{(r)}|\alpha^{(r)}, \sigma^{2(r-1)}, \mathcal{D})$ in Eq. (13).

(Step 5) Obtain $a^{(r)}$ and $\delta b^{(r)}$ by $-\alpha^{(r)}/\beta^{(r)}$ and $-\beta^{(r)}$, respectively. And calculate software reliability assessment measures.

(Step 6) Generate $\sigma^{2(r)}$ from $p(\sigma^{2(r)}|\alpha^{(r)}, \beta^{(r)}, \mathcal{D})$ in Eq. (15).

(Step 7) $r \leftarrow r + 1$, then back to **(Step 2)**.

4 Numerical Example

We show numerical examples of our Bayesian interval estimation approach for software reliability assessment based on the discretized exponential software reliability growth model. We apply the following data: $(n, y_n)(n = 1, 2, \cdots, 25;\ y_{25} = 136)$ [3]. We generated $r = 10000$ samples for all parameters and

Table 1 : Results of interval estimations based on 95% HPD interval ($\alpha = 0.05$).

	HPD Interval	
	Lower	Upper
Expected initial fault content: ω	135.76	144.06
Fault-detection rate: δb	0.1106	0.1159
Expected number of remaining faults: M_{25}	6.244	7.642
Software reliability: $R(25,1)$	0.429	0.485

software reliability assessment measures by following the steps discussed in Section 3. And the first 1,000 samples were discarded as the burn-in samples.

For examples, Figures 1 shows the MCMC samples and the posterior distribution of the expected number of remaining faults at $i = 25$, M_{25} in Eq. (7). From these posterior distributions, we can obtain the interval estimations of the parameter and the software reliability assessment measures. The interval estimation can be obtained by following the notion of the credible interval. The $100(1-\alpha)\%$ credible interval, denoted by C, satisfies

$$\int_C p(\theta|\mathcal{D})d\theta = 1 - \alpha, \tag{17}$$

where $p(\theta|\mathcal{D})$ is the posterior for the parameter of interest. The HPD (highest posterior density) interval is often used for interval estimation in Bayesian approach. The $100(1-\alpha)\%$ HPD interval, which is denoted by C_{HPD}, is obtained as $C_{HPD} = \{\theta \in \boldsymbol{\theta} \mid p(\theta|\mathcal{D}) \geq k(\alpha)\}$, where $\boldsymbol{\theta}$ is the set of the value of parameter and $k(\alpha)$ is the largest number satisfying $p(\theta|\mathcal{D}) \geq k(\alpha)$ and depends on α.

Needless to say, the posterior distributions of parameters a and δb in Eq. (3) and the software reliability in Eq. (8) can be also obtained by following the MCMC method discussed in Section 3. Table 1 shows the results of interval estimations based on the 95% HPD interval for the model parameters and the software reliability assessment measures. The bootstrapping method is based on randomly resampled data needed in the regression analysis. And the probability distribution of the parameter is obtained by the frequentist method, i.e., we need to estimate parameter repetitively by using randomly resampled data in the bootstrapping approach. On the other hand, the Bayesian interval estimation is obtained from the posterior distribution, which is updated by the likelihood for the obtained data. Further, the interval estimation in the Bayesian approach is conducted by sampling the parameter repetitively from the posterior distribution.

5 Conclusion

A Bayesian interval estimation method of a discretized NHPP model for software reliability assessment has been discussed. Concretely, we apply the MCMC method for obtaining the probability distributions of the parameters. Further, we showed numerical examples of our Bayesian interval estimation approach by applying to software fault-count data observed in an actual software testing. In our further studies, we will confirm the difference between the results of interval estimations with the bootstrapping and the Bayesian approaches. Further, we will apply our method to the interval estimation of optimal software release time.

Acknowledgement

This research was supported in part by the Grant-in-Aid for Scientific Research (C), Grant No. 16K00098, from the Ministry of Education, Culture, Sports, Science and Technology of Japan.

References

[1] A.L. Goel and K. Okumoto, "Time-dependent error-detection rate model for software reliability and other performance measures," *IEEE Transactions on Reliability*, Vol. R-28, No. 3, pp. 206–211, 1979.

[2] R. Hirota, "Nonlinear partial difference equations. V. Nonlinear equations reducible to linear equations," *Journal of the Physical Society of Japan*, Vol. 46, No. 1, pp. 312–319, 1979.

[3] S. Inoue and S. Yamada, "Discrete software reliability assessment with discretized NHPP models," *Computers & Mathematics with Applications: An International Journal*, Vol. 51, Issue 2, pp. 161–170, 2006.

[4] S. Inoue and S. Yamada, "Nonparametric bootstrapping interval estimations for software release planning with reliability objective," *Proceedings of the 24th International Symposium on Software Reliability Engineering*, Pasadena, California, U.S.A., November 4-7, 2013, pp. 81–89.

[5] T. Kaneishi and T. Dohi, "Parametric bootstraping for assessing software reliability measures," *Proceedings of the 17th IEEE Pacific Rim International Symposium on Dependable Computing*, 2010, pp. 1–9.

[6] M. Kimura and T. Fujiwara, "A study on bootstrap confidence intervals of software reliability measures based on an incomplete gamma function model," in *Advanced Reliability Modeling II*, T. Dohi and W.Y. Yun (Eds.), pp. 419–426, World Scientific, 2006.

[7] J.D. Musa, D. Iannio, and K. Okumoto, *Software Reliability: Measurement, Prediction, Application.* McGraw-Hill, New York, 1987.

[8] H. Okamura, T. Dohi, and S. Osaki, "Bayesian inference for credible intervals of optimal software release time", *Advances in Software Engineering and Its Applications*, Communications in Computer and Information Science (CCIS) 257, pp. 377-384, Springer, 2011.

[9] H. Pham, *Software Reliability.* Springer-Verlag, Singapore, 2000.

[10] S. Yamada, *Software Reliability Modeling — Fundamentals and Applications —*, Springer Japan, Tokyo, 2014.