A Bootstrapping Approach for Software Reliability Measurement Based on a Discretized NHPP Model

Shinji Inoue, Shigeru Yamada

Department of Social Management Engineering, Graduate School of Engineering, Tottori University, Tottori, Japan.
Email: ino@sse.tottori-u.ac.jp, yamada@sse.tottori-u.ac.jp

Received December 14th, 2012; revised January 17th, 2013; accepted January 26th, 2013

ABSTRACT

Discrete software reliability measurement has a proper characteristic for describing a software reliability growth process which depends on a unit of the software fault-detection period, such as the number of test runs, the number of executed test cases. This paper discusses discrete software reliability measurement based on a discretized nonhomogeneous Poisson process (NHPP) model. Especially, we use a bootstrapping method in our discrete software reliability measurement for discussing the statistical inference on parameters and software reliability assessment measures of our model. Finally we show numerical examples of interval estimations based on our bootstrapping method for the several software reliability assessment measures by using actual data.

Keywords: Software Reliability Measurement; Discretized NHPP Model; Nonparametric Bootstrapping Method; Regression Analysis; Bootstrap Confidence Intervals

1. Introduction

It is very important to measure reliability of a software product with accuracy in the final stage of software development process for shipping a highly reliable software product. A software reliability growth model [1-4] is known as one of the useful mathematical tools for quantitative measurement or assessment of software reliability. Generally in an actual testing-phase, we observe a software reliability growth process, in which software faults are detected and removed and the number of faults remaining in the software system is decreasing along with the test-execution time. The software reliability growth model describes the software reliability growth process, and measures the software reliability quantitatively by using software reliability assessment measures, which are derived by the software reliability growth model. A huge number of software reliability growth models were proposed so far for accurate software reliability assessment. Especially, there are discretized nonhomogeneous Poisson process (discretized NHPP) models, which have good fitting and prediction performance in software reliability assessment [5,6] because the discretized NHPP models have consistency with fault counting data, which are obtained by collecting information on the frequency of the software failure-occurrence or the number of detected faults during each constant testing-period. Estimating parameters in the discretized NHPP model from actual data is conducted by using the regression analysis based on a regression equation derived from a difference equation of the discretized NHPP model. After the parameter estimation, software reliability assessment is performed based on the software reliability assessment measures derived from the discretized NHPP model. This approach is based on the point estimation, which is better to use when we have enough number of data.

In recent years, it is very difficult to obtain enough number of data for the point estimation method due to the quick delivery of software development. In such case, it is better to conduct interval estimation for considering the uncertainty of the estimators being related to the model parameters and software reliability assessment measures. We often use asymptotic approximation approaches [7] for the interval estimation. However, we have some difficulty in mathematical manipulation for conducting the interval estimation even if we use the asymptotic approximation approach. For overcoming the problem above, the bootstrap method [8] was proposed. The bootstrapping method is known as one of the useful Monte Carlo methods for obtaining probability distributions for estimators by a resampling method. Recently,
the bootstrapping method is applied not only to software reliability analysis [9-11] but also optimal checkpoint replacement for hardware system [12].

In this paper, we discuss an interval estimation method for parameters and software reliability assessment measures of a discretized exponential software reliability growth model, which is one of the discretized NHPP models and has the simplest model-structure, by the bootstrapping method. And, we discuss several kinds of bootstrap confidence intervals for the interval estimations. Finally, we show numerical examples for our bootstrapping method for software reliability assessment based on the discretized exponential software reliability model and the bootstrap confidence intervals by using actual data.

2. Discretized Exponential NHPP Model

2.1. The Model

We briefly discuss the aspect of the discretized NHPP model by showing a discretized exponential software reliability growth model [5,6], which has the simplest mathematical structure. Now we define a discrete counting process \( \{N_n, n = 0, 1, 2, \ldots \} \) representing the cumulative number of faults detected up to \( n \)-th testing-period. And we can say that the discrete counting process \( \{N_n, n = 0, 1, 2, \ldots \} \) follows a discrete-time NHPP [13] if the process has the following property:

\[
\Pr \{ N_n = x \mid N_0 = 0 \} = \frac{(\Lambda_x)^x}{x!} \exp[-\Lambda_x] \quad (n, x = 0, 1, 2, \ldots),
\]

which is derived based on a continuous-time NHPP. In Equation (1), \( \Pr \{ A \} \) means the proba of event \( A \). \( \Lambda_x \) is a mean value function of the discrete-time NHPP. The mean value function, \( \Lambda_x \), represents the expected cumulative number of faults detected up to \( n \)-th testing-period.

The discretized exponential software reliability growth model is a discrete analog of the original (continuous-time) exponential software reliability growth model. Let \( H_n \) denote the mean value function of the original continuous-time exponential software reliability growth model. The discretized exponential software reliability growth model is given as

\[
H_{n+1} - H_n = \delta \beta (\omega - H_n),
\]

from the basic assumptions of the original exponential software reliability growth model. In Equation (2), \( \delta \) represents the constant time-interval, \( \omega \) the expected total number of potential faults to be detected in an infinitely long duration or the expected initial fault content, and \( \beta \) the fault detection rate per fault. Regarding the discretization method, we use the Hirota’s bilinearization methods [14] for conserving the property of the continuous-time NHPP model. Solving the integrable difference equation in Equation (2), we can obtain an exact solution \( H_n \) as

\[
H_n = \omega \left[ 1 - (1 - \delta \beta)^{n} \right] \quad (\omega > 0, \beta > 0).
\]

As \( \delta \to 0 \), Equation (3) converges to an exact solution of the original continuous-time exponential software reliability growth model, which is derived by the differential equation.

The discretized exponential software reliability growth model in Equation (3) has two parameters, \( \omega \) and \( \beta \), which have to be estimated by using actual data. The parameter estimations of \( \omega \) and \( \beta \), \( \hat{\omega} \) and \( \hat{\beta} \), can be obtained by the following procedure using the method of least-squares. First of all, if we observed fault counting data \( \{n, y_n\} \) \( (n=1,2,\ldots,N) \), where \( y_n \) represents the cumulative number of faults detected up to \( n \)-th testing-period, we derive the following regression equation from Equation (2):

\[
C_n = \alpha_0 + \alpha_1 D_n,
\]

where

\[
\begin{align*}
C_n &= H_{n+1} - H_n \\
D_n &= H_n \\
\alpha_0 &= \hat{\delta} \hat{\omega} \beta \\
\alpha_1 &= -\hat{\beta}.
\end{align*}
\]

Based on the regression analysis, we can estimate \( \hat{\alpha}_0 \) and \( \hat{\alpha}_1 \), which are the estimations of \( \alpha_0 \) and \( \alpha_1 \) in Equation (4). Then, the parameter estimations, \( \hat{\omega} \) and \( \hat{\beta} \), can be obtained as

\[
\begin{align*}
\hat{\omega} &= -\hat{\alpha}_0 / \hat{\alpha}_1 \\
\hat{\beta} &= -\hat{\alpha}_1 / \delta.
\end{align*}
\]

\( C_n \) in Equation (4) is independent of \( \delta \) because \( \delta \) is not used in calculating \( C_n \) as showing Equation (5). Hence, we can obtain the same parameter estimates \( \hat{\omega} \) and \( \hat{\beta} \), respectively, when we choose any value of \( \delta \) [5,6,15,16].

2.2. Software Reliability Assessment Measures

Software reliability assessment measures are useful in quantitative software reliability assessment. This paper discusses the expected number of remaining faults and the software reliability function, which are well-known software reliability assessment measures. The expected number of remaining faults, \( M_n \), represents the expected number of undetected faults in the software system at arbitrary testing-period. Then, we have

\[
M_n = \mathbb{E}[N_n - N_s] = \omega - \Lambda_n = \omega (1 - \delta \beta)^{n},
\]

if we assume that \( N_n \) follows a discrete-time NHPP.
with mean \( \Lambda_n \) in Equation (3). The software reliability function, \( R(n, h) \), is defined as the probability that a software failure does not occur in the time-interval \( (n, n+h] \) \((h = 1, 2, \cdots)\) given that the testing has been going Up to the nth testing-period. Then, we have

\[
R(n, h) = \Pr\{N_{n+h} - N_n = 0 | N_n = x\} = \exp\left[-\left(\Lambda_{n+h} - \Lambda_n\right)\right]
\]

(8)

\[= \exp\left[-H_n (1-\delta\beta)^x\right].\]

3.3. Software Reliability Assessment Based on Bootstrapping Method

Ordinarily, the parameters of the discretized NHPP models are estimated by using the regression analysis based on the regression equation derived from the difference equation of the discretized NHPP models. However, it is difficult to discuss the statistical inference on software reliability assessment in the existing estimation approach because it is very difficult or complex to identify the probability distribution function for the estimators of parameter analytically. For overcoming a problems above, Kimura and Fujiwara [9,10] applied non-parametric bootstrap software reliability assessment methods for an incomplete gamma function-based software reliability growth model. Kaneishi and Dohi [11] discussed a parametric bootstrap method for software reliability assessment based on continuous-time NHPP models. In this paper, we apply a non-parametric bootstrap method to the discretized NHPP model for estimating model parameters and software reliability assessment measures. Especially in this paper, we discuss five types of bootstrap confidence intervals for interval estimation of the model parameters and software reliability assessment measures.

3.3.1. Our Bootstrapping Method

As an example for discussing our bootstrapping method based on the discretized NHPP model, we apply the discretized exponential software reliability growth model. Our bootstrap method for software reliability assessment follows the following procedure:

Step 1: Estimate \( \alpha_0 \) and \( \alpha_i \) in Equation (4) based on the linear regression scheme by using fault counting data. We indicate \( \hat{\alpha}_0 \) and \( \hat{\alpha}_i \) as \( \hat{\alpha}_{(0)} \) and \( \hat{\alpha}_{(i)} \), respectively.

Step 2: Calculate the residual errors, \( \hat{d}_i \), at each observation point by

\[\hat{d}_i = C_i - \left(\hat{\alpha}_{(0)} + \hat{\alpha}_{(i)} D_i\right) \quad (i = 1, 2, \cdots, N-1).\]

Step 3: Construct an empirical distribution function \( \hat{F} \) by assuming the residual errors \( \hat{d}_i \) follows the independent and identically probability distribution and putting mass 1/(\(N-1\)) at each ordered point \( \{\hat{d}_1, \hat{d}_2, \cdots, \hat{d}_{N-1}\} \).

Step 4: Set the total number of iteration \( B \) and let \( b (b = 1, 2, \cdots, B) \) be the iteration count.

Step 5: Generate a bootstrap sample for the residual errors,

\[\hat{d}_b^* = \{\hat{d}_{b,1}^*, \hat{d}_{b,2}^*, \cdots, \hat{d}_{b,N-1}^*\}\]

by sampling with replacement from \( \hat{F} \).

Step 6: Generate a bootstrap sample for

\[
z_b^* = \{y_1, C_{b,1}, y_2, C_{b,2}, \cdots, y_{N-1}, C_{b,N-1}\}
\]

by

\[C_{b,i} = \hat{\alpha}_{(0)} + \hat{\alpha}_{(i)} D_i + \hat{\alpha}_{b,i}^*.
\]

Step 7: Estimate \( \alpha_{(0)}^* \) and \( \alpha_{(i)}^* \) from the bootstrap sample \( z_b^* \).

Step 8: Calculate parameters of the discretized exponential software reliability growth model by the following equation:

\[
\hat{\alpha}_{(b)} = -\frac{\hat{\alpha}_{(b)}}{D_b} \quad \hat{\beta}_{(b)} = -\frac{\hat{\alpha}_{(b)}}{\delta}.
\]

Step 9: Calculate software reliability assessment measures.

Step 10: Let \( b = b + 1 \) and go back to Step 5 if \( b < B \).

Step 11: We have \( B \) samples for \( \hat{\alpha}, \hat{\beta} \) and a software reliability assessment measures.

We can calculate the mean and the standard deviation for the model parameters and software reliability assessment measures by the Monte Carlo approximation, respectively.

3.3.2. Bootstrap Confidence Intervals

We discuss the following three typical bootstrap confidence intervals [17]: basic, standard normal, and percentile bootstrap confidence intervals. Further we discuss bootstrap-t and BCa methods [17,18] for deriving bootstrap confidence intervals considering with the asymmetric property and the bias and the skewness of the estimator of the parameter. Let \( \theta \) be parameter of interest.

The basic bootstrap confidence interval is derived by using the quantile of the distribution of \( \hat{\theta} - \hat{\theta} \); where \( \hat{\theta} \) is the bootstrap statistic. We can approximate the \( \alpha \) and \( (1-\alpha) \) quantile denoting \( \theta_{\alpha} \) and \( \theta_{1-\alpha} \), respectively, of the distribution of \( \hat{\theta} - \hat{\theta} \) by \( \hat{\theta}_{\alpha} - \hat{\theta} \) and \( \hat{\theta}_{1-\alpha} - \hat{\theta} \). Then,
1 - 2\alpha = \Pr \left\{ v_{i-a} \leq \hat{\theta} - \theta \leq v_{1-a} \right\}
= \Pr \left\{ \hat{\theta} - v_{1-a} \leq \theta \leq \hat{\theta} - v_{a} \right\}
= \Pr \left\{ 2 \hat{\theta} - \hat{\theta}_{[n_{1-a}]} \leq \theta \leq 2 \hat{\theta} - \hat{\theta}_{[a]} \right\}.

Thus, the 100(1 - 2\alpha)\% basic bootstrap confidence interval is given by
\[ \left[ 2 \hat{\theta} - \hat{\theta}_{[n_{1-a}]} \right], \left[ 2 \hat{\theta} - \hat{\theta}_{[a]} \right]. \tag{9}\]

The standard normal bootstrap confidence interval is derived by assuming that the distribution of \( \hat{\theta} - \theta \) can be approximated by the distribution of \( \hat{\theta} - \hat{\theta} \) and \( \hat{\theta} - \hat{\theta} \sim N\left(0, SD[\hat{\theta}]^{2}\right) \). That is,
\[ 1 - 2\alpha = \Pr \left\{ w_{a} \leq \frac{\hat{\theta} - \hat{\theta}}{SD[\hat{\theta}]} \leq w_{1-a} \right\}. \]

Thus, we have the 100(1 - 2\alpha)\% standard normal bootstrap confidence interval as
\[ \left[ \hat{\theta} - w_{1-a} SD[\hat{\theta}], \hat{\theta} - w_{a} SD[\hat{\theta}] \right]. \tag{10}\]
where \( w_{a} = \Phi^{-1}(\alpha) \), which is the \( \alpha \) quantile of the standard normal distribution. For example,
\[ w_{0.025} = -w_{0.025} = 1.96. \]

The percentile bootstrap confidence interval is calculated from the empirical cumulative probability distribution function, which consists of the bootstrap iteration values: \( \hat{\theta}_{(1)}, \hat{\theta}_{(2)}, \ldots, \hat{\theta}_{(B)} \). Then, the 100(1 - 2\alpha)\% percentile bootstrap confidence interval is calculated by
\[ \left[ \hat{\theta}_{[n_{1-a}]}, \hat{\theta}_{[a]} \right]. \tag{11}\]
where \( \hat{\theta}_{[a]} \) represents the \( \alpha \) quantile of the empirical cumulative probability distribution function.

Further, we discuss a bootstrap-\( t \) method, which enables us to take into consideration the variance of \( \hat{\theta} \) by deriving \( T = \frac{\hat{\theta} - \theta}{\hat{\sigma}} \), where \( \hat{\sigma} \) is the variance of \( \hat{\theta} \). Letting \( u_{a} \) and \( u_{1-a} \) are the \( \alpha \) and \( 1 - \alpha \) quantile of \( T \), we have
\[ 1 - 2\alpha = \Pr \left\{ u_{a} \leq \frac{\hat{\theta} - \theta}{\hat{\sigma}} \leq u_{1-a} \right\}
= \Pr \left\{ \hat{\theta} - \hat{\sigma} u_{1-a} \leq \theta \leq \hat{\theta} - \hat{\sigma} u_{a} \right\}.
\]

In above equation, we substitute \( u_{a} \) and \( u_{1-a} \), which are \( \alpha \) and \( 1 - \alpha \) quantile of \( T \), into \( T_{[n_{1-a}]} \) and \( T_{[a]} \), which are the \( \alpha \) and \( 1 - \alpha \) quantile of \( T^{*} = \left( \hat{\theta}^{*} - \hat{\theta} \right)/\hat{\sigma}^{*} \). Then, the 100(1 - 2\alpha)\% bootstrap-\( t \) confidence interval is derived as
\[ \left[ \hat{\theta} - \hat{\sigma} T_{[n_{1-a}]}^{*}, \hat{\theta} - \hat{\sigma} T_{[a]}^{*} \right]. \tag{12}\]

In Equation (12), \( \hat{\sigma} \) is the standard deviation of \( \hat{\theta}_{(b)} \) and \( \hat{\sigma}^{*} \) is estimated by the bootstrap-\( t \) statistics \( T^{*} \).

And we also discuss a BCa method for getting better bootstrap confidence interval with the asymmetric property, the bias, and the skewness of the probability distribution of the estimator. The BCa confidence interval can be given as
\[ \left[ \hat{\theta}^{* - 1} \left( \Phi \left( z(\alpha) \right) \right), \hat{\theta}^{* - 1} \left( \Phi \left( z(1 - \alpha) \right) \right) \right], \tag{13}\]
where
\[ z(\alpha) = z_{0} + \frac{z_{a} + z_{0}}{1 - a(z_{a} + z_{0})}. \]

In above equation, \( z_{0} = \Phi^{-1} \left( \hat{\Phi}(\hat{\theta}) \right) \), in which \( \hat{\Phi}(\cdot) \) is the bootstrap distribution for the estimator. And \( a \) is the acceleration constant derived as
\[ a = \frac{1}{6} \sum_{i=1}^{n} \left( \hat{\theta}_{(i)} - \hat{\theta}_{(i)} \right)^{3} n^{1/2}, \]
where \( \hat{\theta}_{(i)} \) is a jackknife iteration value, which is estimated by using the data, removed \( i \)th data and \( \hat{\theta}_{(i)} = \sum_{i=1}^{n} \hat{\theta}_{(i)} \).

4. Numerical Examples

We show numerical examples for our bootstrap software reliability assessment method based on the discretized exponential software reliability growth model.

We apply fault counting data: \( (n, y_{n}) \)
\( (n = 1, 2, \ldots, 25; y_{25} = 136) \) [1] and we set the total number of iteration \( B = 2000 \).

We first obtain
\[ \hat{\alpha}_{(0)} = 139.9564 \quad \text{and} \quad \hat{\alpha}_{(1)} = 0.1133109 \]
by the linear regression scheme from the actual data. Following to our bootstrapping method, we have 2000 bootstrap samples \( \{\hat{\alpha}_{(1)}, \hat{\alpha}_{(2)}, \ldots, \hat{\alpha}_{(2000)}\} \). Then, we obtain bootstrap samples for \( \alpha \) and \( \beta \). Figures 1 and 2 show histograms for the bootstrap samples \( \hat{\alpha}_{(b)} \) and \( \hat{\beta}_{(b)} \) to see bootstrap distributions of \( \alpha \) and \( \beta \), respectively. And we have bootstrap samples for the software reliability assessment measures, such as the expected number of remaining fault at the termination time of the testing, \( \hat{M}_{25} \), and the software reliability, \( \hat{R}(25,1) \), respectively. These bootstrap samples of \( \hat{M}_{25} \) and \( \hat{R}_{25}(25,1) \) are calculated by
\[ \hat{M}_{25} = \frac{\hat{\alpha}_{(b)}}{\hat{\alpha}_{(b)}} \left( 1 + \hat{\alpha}_{(b)} \right)^{25}, \tag{14}\]
\[ \hat{R}_{25}(25,1) = \exp \left[ \frac{\hat{\alpha}_{(b)}}{\hat{\alpha}_{(b)}} \left( 1 - \left( 1 + \hat{\alpha}_{(b)} \right) \right) \left( 1 + \hat{\alpha}_{(b)} \right) \right]. \tag{15}\]
A Bootstrapping Approach for Software Reliability Measurement Based on a Discretized NHPP Model

Figure 1. Bootstrap distribution of $\hat{\omega}$.

Figure 2. Bootstrap distribution of $\hat{\beta}$.

Figure 3. Bootstrap distribution of the expected number of remaining faults at $n = 25, \hat{M}_{25}$.

Figure 4. Bootstrap distribution of software reliability, $\hat{R}(25, 1)$.

From Equations (7) and (8), respectively. Figures 3 and 4 show histograms of the bootstrap samples for $\hat{M}_{25}^{(b)}$ and $\hat{R}^{(b)}(25, 1)$, respectively. Further, Table 1 indicates the mean and the standard deviations of the estimators of $\hat{\alpha}_0, \hat{\alpha}_1, \hat{\omega}, \hat{\beta}, \hat{M}_{25}$, and $\hat{R}(25, 1)$, respectively. And, Figure 5 shows the estimated discretized exponential software reliability growth model, $\hat{H}_n$, in which we use the means of the bootstrap samples of $\hat{\alpha}_0^{(b)}$ and $\hat{\beta}^{(b)}$ as the point estimations, respectively. The means of $\hat{\alpha}_0^{(b)}$ and $\hat{\beta}^{(b)}$, which are denoted by $\bar{\omega}$ and $\bar{\beta}$, are calculated by

$$\bar{\omega} = \frac{1}{B} \sum_{b=1}^{B} \hat{\alpha}_0^{(b)},$$

(16)

respectively.

Further, Table 2 shows the results of interval estimations based on the basic, standard normal, percentile, bootstrap-$t$, and BCa methods, respectively, with the 5% significance level ($\alpha = 0.025$). From Table 2, we can see that the inappropriate confidence intervals for $\hat{M}_{25}$ are estimated in the basic, standard normal, and bootstrap-$t$ confidence intervals because the number of re-
A Bootstrapping Approach for Software Reliability Measurement Based on a Discretized NHPP Model

5. Conclusions

This paper discussed a bootstrap software reliability assessment method based on a discretized exponential software reliability growth model. And we discussed five types of bootstrapping confidence intervals for interval estimations of model parameters and several software reliability assessment measures.

In our numerical examples, we confirmed that our bootstrap approach gives probability distributions of each parameters and software reliability assessment measures numerically even if we do not derive these probability distributions analytically, and that we can obtain useful information in software reliability assessment, such as results of interval estimations on the model parameters, the number of remaining faults, and software reliability. This approach is very useful for the case that we cannot collect enough number of data and we have to conduct interval estimation for complex estimators. However, regarding bootstrap confidence intervals, we encountered a problem that we could not get appropriate interval estimations in the basic, standard normal, and bootstrap-t

Table 1. Quantities of the bootstrap distribution.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Mean</th>
<th>Standard Deviation</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \hat{\alpha}_0 )</td>
<td>15.79443</td>
<td>1.500837</td>
</tr>
<tr>
<td>( \hat{\alpha}_1 )</td>
<td>-0.1127263</td>
<td>0.01422163</td>
</tr>
<tr>
<td>( \hat{\omega} )</td>
<td>140.7685</td>
<td>6.609685</td>
</tr>
<tr>
<td>( \hat{\beta} )</td>
<td>0.1127263</td>
<td>0.01422163</td>
</tr>
<tr>
<td>( \hat{M}_{25} )</td>
<td>7.761994</td>
<td>3.59502</td>
</tr>
<tr>
<td>( \hat{R}(25,1) )</td>
<td>0.4514124</td>
<td>0.1072053</td>
</tr>
</tbody>
</table>

Table 2. Results of interval estimations based on bootstrap confidence intervals.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Lower</th>
<th>Upper</th>
</tr>
</thead>
<tbody>
<tr>
<td>Basic</td>
<td>124.0544</td>
<td>150.3849</td>
</tr>
<tr>
<td>Standard Normal</td>
<td>127.0015</td>
<td>152.9114</td>
</tr>
<tr>
<td>Percentile</td>
<td>129.528</td>
<td>155.8585</td>
</tr>
<tr>
<td>Bootstrap-t</td>
<td>122.9817</td>
<td>152.0545</td>
</tr>
<tr>
<td>BCa</td>
<td>128.1697</td>
<td>153.2575</td>
</tr>
<tr>
<td>Basic</td>
<td>0.08618762</td>
<td>0.141511</td>
</tr>
<tr>
<td>Standard Normal</td>
<td>0.08543655</td>
<td>0.1411854</td>
</tr>
<tr>
<td>Percentile</td>
<td>0.08511091</td>
<td>0.144191</td>
</tr>
<tr>
<td>Bootstrap-t</td>
<td>0.08372191</td>
<td>0.144191</td>
</tr>
<tr>
<td>BCa</td>
<td>0.09073159</td>
<td>0.1481078</td>
</tr>
<tr>
<td>Basic</td>
<td>-2.835469</td>
<td>10.84297</td>
</tr>
<tr>
<td>Standard Normal</td>
<td>-0.1235227</td>
<td>13.96896</td>
</tr>
<tr>
<td>Percentile</td>
<td>3.002461</td>
<td>16.6809</td>
</tr>
<tr>
<td>Bootstrap-t</td>
<td>-5.021783</td>
<td>11.90131</td>
</tr>
<tr>
<td>BCa</td>
<td>2.300986</td>
<td>13.80817</td>
</tr>
</tbody>
</table>

Copyright © 2013 SciRes.
confidence intervals for the number of remaining faults at the termination time of the testing. This problem was solved by using other bootstrap confidence intervals, such as the percentile and the BCa confidence intervals. In the future studies, we are going to apply our bootstrap approach for estimating optimal software release time and other practical software project management issues.

6. Acknowledgements
The second author is supported in part by the Grant-in-Aid for Scientific Research (C), Grant No. 22510150, from the Ministry of Education, Culture.

REFERENCES


