This chapter develops a software reliability growth model based on the non-homogenous Poisson process incorporating the amount of test-effort expanded during the software testing phase. The time dependent behavior of test-effort expenditures is described by a Burr Type III curve.

SRGMs proposed by most researchers incorporate the effect of testing effort in the software reliability growth and the software development effort can be described by the traditional Rayleigh, Weibull or Exponential curve. However, in much software testing environment it is difficult to describe the testing-effort function by the above three consumption curves. Here, we will show that a Burr type III testing-effort function can be expressed as a software development/test effort curve. Experiments have been performed based on real test/debug data set. The results show that the SRGMs with a Burr type III testing-effort function can estimate the number of initial faults better than previous approaches.

4.1 Introduction

A computer system consists of two major components: Hardware and Software. Although extensive research has been done in the area of hardware reliability, research has also been conducted to study the software reliability of computer systems since 1970. Software reliability is the probability that a given software will be functioning without failure in a given environment during a specified period of time. Hence, software
reliability is a key factor in software development process and software quality. The testing phase is an important and expensive part during the software development process which includes the following four phases: specification, design, programming and test-and-debug. Many resources are consumed by a software development project. It is assumed that the consumption rate of testing resource expenditures during the testing phase is a constant or even do not consider such testing effort. In reality software reliability models should be developed by incorporating different testing-effort functions. Yamada et al. (1986, 1993, 1990 and 1987) and Musa et al. (1987) proposed a new and simple software reliability growth model which describes the relationship among the calendar testing, the amount of testing-effort, and the number of software errors detected.

Software reliability has been often studied in terms of software reliability growth models, based on observed software error data during the software testing phase. Software reliability growth models are concerned with the relation between the cumulative number of errors detected by software testing and the time span of the software testing. Software reliability growth models can estimate the expected initial error content of a software system, the expected number of remaining errors at an arbitrary testing time point, the software reliability, and so on. Several software reliability growth models have been proposed and investigated. For example, Goel & Okumoto (1979), Jelinski & Moranda (1972), Littlewood (1980), Moranda (1979), Musa (1980) and Yamada et al. (1984).

In general appreciable testing resources are spent on software testing in software development. The consumption curve of testing resources over the testing period can be thought of as a testing-effort curve. Testing-effort is
measured by: the number of executed test cases, the amount of man-power, and the CPU time spent during the testing phase, and so on. However existing software reliability growth models do not consider such testing-effort; that is, they assume that testing-effort is constant over the testing period. We should consider the effect of testing-effort on software reliability growth in order to develop more realistic software reliability growth models. Parameters are estimated by least square and maximum likelihood estimation method.

4.2 Burr Type III testing-effort function

Since actual testing-effort data express various expenditure patterns, sometimes the testing-effort expenditures are difficult to be described by only a Exponential or Rayleigh curve. Although the Weibull-type curve can fit the data well under the general software development environment, it will have an apparent peak phenomenon when the shape parameter \( m > 3 \). To overcome with this difficulty consider Burr type III test-effort function. The Burr type III testing-effort function has the following form:

The cumulative testing-effort consumption in time \((0, t]\) is

\[
W(t) = a\left[1 + (\beta t)^{-\delta}\right]^{-\alpha}
\]

\[
\alpha > 0, \beta > 0
\]

and the current testing effort consumption curve is given as

\[
w(t) = \alpha \beta m \delta (\beta t)^{-\delta-1} \left[1 + (\beta t)^{-\delta}\right]^{-\alpha-1}
\]

\[
\alpha > 0, \beta > 0, m > 0, \delta > 0
\]

Where \( \alpha, \beta, m, \) and \( \delta \) are constant parameters, \( \alpha \) is the total amount of test-effort expenditure, \( \beta \) is the scale parameter, and \( m \) and \( \delta \) are shape parameters.

The divergence between the Weibull-type curve and \( W(t) \) is concentrated in the earlier stages of software development where progress is often least
visible and formal accounting procedures for recording the amount of testing effort applied may not have been instituted. It is possible for us to judge between these models using some statistical test of their relative ability to fit actual failure data such as adjusting the origin and scales linearly (Parr (1980)).

4.2.1 Least square estimation of parameter

The parameters $\alpha, \beta, m$ and $\delta$ in the Burr type III testing-effort functions defined by eqn. (4.1) or eqn. (4.2) can be estimated by least-squares method discussed by Draper & Smith (1981). The estimators for $\alpha, \beta, m$ and $\delta$ are investigated for testing-effort $w_k$ spent at testing time $t_k$ ($k = 1, 2, \ldots, n$). Then, based on the usual procedures, the least-squares estimators $\hat{\alpha}, \hat{\beta}, \hat{m}$ and $\hat{\delta}$ be obtained by minimizing the following equation:

Minimize, $S(\alpha, \beta, m, \delta) = \sum_{k=1}^{n} \left[ \log W_k - \log \alpha + m \log \left[ 1 + (\beta t_k)^{-\delta} \right] \right]^2$

Differentiating the above equation with to $\alpha, \beta, m$ and $\delta$, we have the following non-linear equations:

\[
\frac{\partial S}{\partial \alpha} = \sum_{k=1}^{n} 2 \left[ \log W_k - \log \alpha + m \log \left[ 1 + (\beta t_k)^{-\delta} \right] \right] \left( -\frac{1}{\alpha} \right) = 0
\]

\[
\sum_{k=1}^{n} \log W_k - n \log \alpha + m \sum_{k=1}^{n} \log \left[ 1 + (\beta t_k)^{-\delta} \right] = 0
\]

\[
\therefore n \log(\alpha) = \sum_{k=1}^{n} \log W_k + m \sum_{k=1}^{n} \log \left[ 1 + (\beta t_k)^{-\delta} \right] \tag{4.3}
\]

\[
\frac{\partial S}{\partial \beta} = \sum_{k=1}^{n} 2 \left[ \log W_k - \log \alpha + m \log \left[ 1 + (\beta t_k)^{-\delta} \right] \right] \left( m \frac{1}{1 + (\beta t_k)^{-\delta}} \right) \left[ -\delta(\beta t_k)^{-\delta-1} t_k \right] = 0
\]

\[
\therefore \sum_{k=1}^{n} \left[ \log W_k - \log \alpha + m \log \left[ 1 + (\beta t_k)^{-\delta} \right] \right] \left[ \frac{t_k^{-\delta}}{1 + (\beta t_k)^{-\delta}} \right] = 0 \tag{4.4}
\]
These non-linear equations can be solved numerically to estimate $\hat{\alpha}$, $\hat{\beta}$, $\hat{m}$ and $\hat{\delta}$.

### 4.3 Software reliability growth model

A number of SRGMs have been proposed on the subject of software reliability. Among these models, Goel and Okumoto (1979) used an NHPP as the stochastic process to describe the fault process, Lyu (1996), Huang et al. (2002), Berman et al. (1998) and Boehm (2000) modify the G-O model and incorporate the concept of testing-effort in an NHPP model to get a better description of the software fault detection phenomenon. We also propose a new SRGM with the Burr type III testing-effort function to predict the behavior of failure occurrences and the fault content of a software product. Based on our past experimental results, this approach is suitable for estimating the reliability of software application during the development process.

#### 4.3.1 Model Description

Based on the assumptions given below, if the number of detected errors due to the current testing-effort expenditures is proportional to the number of remaining errors using the Burr type III test effort function in equation (1),
we have the following differential equation:

\[
\frac{\partial m(t)}{\partial t} w(t) = r[a - m(t)] , \quad a > 0, \ 0 < r < 1
\]  \(4.7\)

Where \(a\) is the initial error content in the system & \(r\) is the error detection rate per error (per unit test-effort expenditures at testing time \(t\)). Solving the differential eqn. (4.7), we get

\[
m(t) = a\left(1 - e^{-rW(t)}\right)  \quad 4.8\]

Substituting (4.1) for \(W(t)\) in eqn.(4.8), the equation (4.8) can be explicitly rewritten as:

\[
m(t) = a\left(1 - e^{-r,a[[a^{x(\beta, 1)-a}]^m]}\right)  \quad 4.9\]

From equation (4.8), we have the following important relationship between \(m(t)\) & \(W(t)\)

\[
W(t) = \frac{1}{r} \log\left(\frac{a}{a - m(t)}\right) \quad 4.10
\]

For stochastic modeling of a software-error detection phenomenon, let \([N(t), t \geq 0]\) be a counting process representing the cumulative number of errors detected by testing time \(t\). Defining the mean value of \(N(t)\) based on an NHPP by \(m(t)\) in (4.8) yields a software reliability growth model incorporating the Burr type III test-effort function under the assumptions of Goel and Okumoto (1979) and Yamada (1991) by an NHPP as:

\[
\Pr\{N(t) = n\} = \frac{[m(t)]^n e^{-m(t)}}{n!} \quad n=0,1,2,\ldots\ldots
\]

\[
= Poi(m(n; m(t)))  \quad 4.11
\]

where \(m(t)\) is called mean value function of the NHPP (Yamada et al. (1985, 1993)) and \(Poi(n; m(t))\) is a Poisson pmf with parameter \(m(t)\). The intensity function of the NHPP is given by:

\[
\lambda(t) = \frac{\partial m(t)}{\partial t} = a.r.w(t)e^{-rW(t)} \quad 4.12
\]
which means the instantaneous error detection rate. From equation (4.11), we can show that the limit distribution of \( N(t) \) is a Poisson distribution with the following mean:

\[
m(\infty) = a(1 - e^{-\alpha t})
\]

The equation (4.13) implies that even if a software system is tested during an infinitely long duration, all errors remaining in the system can not be detected (Yamada et al. (1986, 1993)). Thus the mean number of undetected errors \( d(t) \) is a test is applied for an infinite amount of time is:

\[
a - m(\infty) = a - a(1 - e^{-\alpha t})
\]

\[
\Rightarrow d(t) = ae^{-\alpha t}
\]

**Assumptions**

1. The error removal process follows the Non Homogeneous Poisson Process (NHPP).
2. The software system is subject to failures at random times caused by errors remaining in the system.
3. The mean number of errors detected in the time interval \((t, t + \Delta t]\) by the current test-effort is proportional to the mean number of remaining errors in the system.
4. The proportionality is a constant over time.
5. The consumption curve of testing-effort function is described by Burr Type III.
6. Each time a failure occurs, the error which caused it is immediately removed, and no new errors are introduced.

### 4.3.2 Software reliability measures

Let \( N(t) \) represent the number of errors remaining in the system of testing time \( t \). Based on the NHPP model with \( m(t) \), given by in equation (4.8), two
quantitative measures for software reliability assessment can be derived from Goel & Okumoto (1979) and Yamada (1991). The expectation of $\overline{N}(t)$ and its variance are given by:

$$
r(t) = E[\overline{N}(t)] = E[(\infty - N(t)) = E[N(\infty)] - E[N(t)]
$$

$$
= m(\infty) - m(t) = a[e^{-rW(t)} - e^{-rW(\infty)}]
$$

$$
= \text{Var}[\overline{N}(t)]
$$

The software reliability representing the probability that a software failure does not occur in the time interval $t, t+x$ is given by:

$$
R = R(x/t) = e^{-[m(t+x)-m(t)]} = e^{-[e^{-rW(t+x)}-e^{-rW(t)}]} \tag{4.15}
$$

The instantaneous mean time between failures (MTBF) at arbitrary testing time can be defined as a reciprocal of the instantaneous error detection rate in equation (4.12) Yamada (1985). Then, the instantaneous MTBF is given by:

$$
\text{MTBF}(t) = \frac{1}{\bar{\lambda}(t)} = \frac{1}{a x w(t) e^{-rW(t)}}
$$

$$
= \frac{1}{\alpha x \beta m \delta (\beta t)^{-\delta} \Gamma(\delta-\tau) \beta^\tau \alpha^{\delta-\tau} (1 + \beta t)^{-\delta}} \tag{4.16}
$$

### 4.4 Maximum likelihood estimations

The reliability growth parameters $a$ and $r$ in the NHPP model with $m(t)$ in eqn. (4.7) can be estimated by the method of maximum-likelihood estimation. Let the estimated parameters $\hat{\alpha}, \hat{\beta}, \hat{m}$ and $\hat{\delta}$ in the Burr type III test-effort function in eqn. (4.1) have been obtained by the method of least-squares estimation. The $\hat{a}$ and $\hat{r}$ are determined for then observed data pairs $(t_k, y_k)(k = 1, 2, \ldots, n)$. Then, the joint pmf, the log-likelihood function, for the unknown parameters $a$ and $r$ in the NHPP model with $m(t)$ in eqn. (4.7), is given by:
\[
\ln L = \sum_{k=1}^{n} (y_k - y_{k-1}) \ln a + \sum_{k=1}^{n} (y_k - y_{k-1}) \ln (\exp[-rW(t_{k-1})] - \exp[-rW(t_k)])
\]

\[-a(1 - \exp[-rW(t_{k-1})]) - \sum_{k=1}^{n} \ln [(y_k - y_{k-1})] \]

\(t_0 = 0 \text{ and } y_0 = 0\)

The usual calculus methods for an interior maximum result in

\[y_n = a f_n,\]  
\[a g_n = \sum_{k=1}^{n} \frac{(y_k - y_{k-1})(g_k - g_{k-1})}{(f_k - f_{k-1})},\]  
\[f_k = 1 - \exp[-rW(t_k)],\]  
\[g_k = W(t_k), \exp[-rW(t_k)] \quad (k = 1,2,...,n)\]

which can be solved numerically to estimate value of \(\hat{a}\) and \(\hat{r}\).

If the sample size \(n\) of the observed data is sufficiently large, the maximum-likelihood estimates (MLE) \(\hat{a}\) and \(\hat{r}\) asymptotically follow a bivariate s-normal distribution (Nelson (1982) Okamoto & Goel (1980)).

\[
\left(\hat{a}, \hat{r}\right) \sim \text{BVN}\left[\left(\hat{a}, \hat{r}\right), \Sigma\right] \quad (n \to \infty)
\]

The variance-covariance matrix \(\Sigma\) in the asymptotic properties of (4.21) is useful in qualifying the variability of the estimated parameters \(\hat{a}\) and \(\hat{r}\) Nelson (1982), and is the inverse of \(F\)

\[
F = \begin{bmatrix}
E \left( \frac{\partial^2 \ln L}{\partial a^2} \right) & E \left( \frac{\partial^2 \ln L}{\partial a \partial r} \right) \\
E \left( \frac{\partial^2 \ln L}{\partial a \partial r} \right) & E \left( \frac{\partial^2 \ln L}{\partial r^2} \right)
\end{bmatrix}
\]

\[
= \begin{bmatrix}
\frac{f_n}{a} & g_n \\
g_n & a \sum_{k=1}^{n} \frac{(g_k - g_{k-1})^2}{(f_k - f_{k-1})}
\end{bmatrix}
\]

116
Where,
\[ g_k = W(t) e^{-rW(t)} \]  \hspace{1cm} \text{(4.23)}

and
\[ f_k = 1 - e^{-rW(t)} \]

Substitute the value of \( a \) and \( r \) in (4.23) and calculate \( F^{-1} \). The estimated asymptotic variance-covariance matrix is:
\[
\hat{\Sigma} = F^{-1} = \begin{pmatrix}
\text{Var}(\hat{a}) & \text{Cov}(\hat{a}, \hat{r}) \\
\text{Cov}(\hat{a}, \hat{r}) & \text{Var}(\hat{r})
\end{pmatrix}
\]

\( \Sigma \) is useful in quantifying the variability of the estimated parameters.

4.5 Real Software Data Analysis & Comparison Criteria

4.5.1 Comparison Criteria

To check the performance of our software reliability growth model and to make their comparison with the other existing SRGM, we use two types of comparison criteria:

(1) The Accuracy of Estimation Musa et al. (1987), Hou et al. (1994), Goel et al. (1979) and Kuo et al. (2001).
\[
(AE) = \frac{|M_a - m|}{M_a} \]
\hspace{1cm} \text{(4.24)}

Where \( M_a \) is the actual cumulative number of detected errors during the test and after the test, and \( m \) is the estimated parameter.

(2) The mean of square fitting Errors
\[
(MSE) = \frac{\sum_{k=1}^{n} [m(t_k) - m_k]^2}{k} \]
\hspace{1cm} \text{(4.25)}

The lower MSE indicates less fitting errors and better performance (Kapur & Garg (1996)).
4.5.2 Actual Data Analysis

The set of real data used is given in Table 1. In this paper we use System TI data of the Rome Air Development Center (RADC) projects and cited from Musa et al. (1987). The number of object instructions for the system TI which is used for a real-time command and control application. In this case, the size of the software is approximately 21,700 object instructions. The software was tested for 21 weeks with 9 programmers. During the test phase, about 25.3 CPU Hours were used and 136 faults were detected.

In order to estimate the parameters $\alpha, \beta, m$ and $\delta$ of the Burr Type III distributed function; we fit the actual testing-effort data into equations (4.1) and (4.2) and solve it by using the method of least squares. That is, we will minimize the sum of squares given in equation (4.3). Hence, we can find the estimates only through numerical procedures. The estimated values of parameters of the Burr Type III testing effort function are:

\[ \hat{\alpha} = 27.54330, \hat{\beta} = 0.0528876, \hat{m} = 0.405062, \text{ and } \hat{\delta} = 13.56046 \]

The estimated Burr Type III test-effort functions are

\[ \hat{w}(t) = 27.5433 \times 0.0528876 \times 0.405062 \times 13.56046^{0.0528876t - 13.56046^{-1}} \]

\[ \hat{W}(t) = 27.5433[1 + (0.0528876)^{-13.56046}]^{-0.405062} \]

Figure 1 and Figure 2 shows the fitting of the estimated testing-effort by using Equation (4.27) and (4.28). Here, the fitted curves are shown as a dotted line and solid line is actual software data. Using the estimated parameters $\alpha, \beta, m$ and $\delta$, the other parameters $a, r$ in (4.8) can be solved MLE method.

For these estimates, the optimality was checked numerically.
These estimated parameters are 
\[ a = 134.14062 \text{ and } r = 0.153455 \]

Table 4.2 summarizes the experimental results of estimated parameters with their standard errors and 95% confidence bound. The estimated mean value function is

\[ \hat{m}(t) = 134.14061 \left(1 - e^{0.153455 \cdot 27.5433 \cdot \left[1 + (0.05288 \cdot t)^{13.5604}\right]^{0.40562}}\right) \quad 4.29 \]

Where \( \hat{W}(t) = 27.54330 \left[1 + (0.05288 \cdot t)^{13.5604}\right]^{0.40562} \)

Table 4.3 and 4.4 shows regression analysis depends on test-effort and number of failure respectively. Similarly, we plotted a fitted curve of the estimated mean value function with the actual software data in Figure 4.3. Intensity function also given in Figure 4.4 fitted well in this experiment.

Also a comparison Table of the estimates of our model along with other models with initial faults \( a \) and MSE is given in Table 5. From Figures (4.1), (4.2), (4.3) and (4.4) and the comparison criteria in Table 4.5 shows that our SRGM is better fit than the other models for debugging data. Kolmogorov Smirnov goodness-of-fit test shows that our proposed SRGM described by an NHPP with \( \hat{m}(t) \) in (4.29) fits pretty well at the 5% level of significance.)
Table 4.1: Software Failure Data (system TI)

<table>
<thead>
<tr>
<th>Times of Observation (in week)</th>
<th>Current execution time (in CPU hr)</th>
<th>Cumulative execution time</th>
<th>Number of failure</th>
<th>Cumulative number of failure</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.00917</td>
<td>0.00917</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>0.01000</td>
<td>0.01917</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>0.00300</td>
<td>0.02217</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>4</td>
<td>0.02300</td>
<td>0.04517</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>5</td>
<td>0.04100</td>
<td>0.08617</td>
<td>1</td>
<td>4</td>
</tr>
<tr>
<td>6</td>
<td>0.00400</td>
<td>0.09017</td>
<td>2</td>
<td>6</td>
</tr>
<tr>
<td>7</td>
<td>0.02500</td>
<td>0.11517</td>
<td>1</td>
<td>7</td>
</tr>
<tr>
<td>8</td>
<td>0.30200</td>
<td>0.41717</td>
<td>9</td>
<td>16</td>
</tr>
<tr>
<td>9</td>
<td>0.97300</td>
<td>1.39017</td>
<td>13</td>
<td>29</td>
</tr>
<tr>
<td>10</td>
<td>0.02000</td>
<td>1.41017</td>
<td>2</td>
<td>31</td>
</tr>
<tr>
<td>11</td>
<td>0.45000</td>
<td>1.86017</td>
<td>11</td>
<td>42</td>
</tr>
<tr>
<td>12</td>
<td>0.25000</td>
<td>2.11017</td>
<td>2</td>
<td>44</td>
</tr>
<tr>
<td>13</td>
<td>0.94000</td>
<td>3.05017</td>
<td>11</td>
<td>55</td>
</tr>
<tr>
<td>14</td>
<td>1.34000</td>
<td>4.39017</td>
<td>14</td>
<td>69</td>
</tr>
<tr>
<td>15</td>
<td>3.32000</td>
<td>7.71017</td>
<td>18</td>
<td>87</td>
</tr>
<tr>
<td>16</td>
<td>3.56000</td>
<td>11.27017</td>
<td>12</td>
<td>99</td>
</tr>
<tr>
<td>17</td>
<td>2.66000</td>
<td>13.93017</td>
<td>12</td>
<td>111</td>
</tr>
<tr>
<td>18</td>
<td>3.77000</td>
<td>17.70017</td>
<td>15</td>
<td>126</td>
</tr>
<tr>
<td>19</td>
<td>3.40000</td>
<td>21.10017</td>
<td>6</td>
<td>132</td>
</tr>
<tr>
<td>20</td>
<td>2.40000</td>
<td>23.50017</td>
<td>3</td>
<td>135</td>
</tr>
<tr>
<td>21</td>
<td>1.80000</td>
<td>25.30017</td>
<td>1</td>
<td>136</td>
</tr>
</tbody>
</table>
### Table-4.2 Experiment result of different parameters

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Estimate</th>
<th>Standard error</th>
<th>95% confidence interval</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>lower</td>
</tr>
<tr>
<td>α</td>
<td>27.54330</td>
<td>1.670201</td>
<td>24.01948</td>
</tr>
<tr>
<td>β</td>
<td>0.05288</td>
<td>0.000088</td>
<td>0.0510222</td>
</tr>
<tr>
<td>m</td>
<td>0.0405062</td>
<td>3.75230</td>
<td>5.642788</td>
</tr>
<tr>
<td>δ</td>
<td>13.56046</td>
<td>0.138939</td>
<td>0.0119295</td>
</tr>
<tr>
<td>a</td>
<td>134.1406112</td>
<td>5.4235</td>
<td>122.70776</td>
</tr>
<tr>
<td>r</td>
<td>0.153455</td>
<td>0.01866</td>
<td>0.144393</td>
</tr>
</tbody>
</table>

### Table-4.3 Regression analysis depends on test-effort

<table>
<thead>
<tr>
<th>Source</th>
<th>Degree of freedom</th>
<th>Sum of squares</th>
<th>Mean squares</th>
</tr>
</thead>
<tbody>
<tr>
<td>Regression</td>
<td>4</td>
<td>2369.05331</td>
<td>592.26333</td>
</tr>
<tr>
<td>Residual</td>
<td>17</td>
<td>2.94670</td>
<td>0.17334</td>
</tr>
<tr>
<td>Uncorrected total</td>
<td>21</td>
<td>2372.0004</td>
<td>-</td>
</tr>
<tr>
<td>(corrected total)</td>
<td>20</td>
<td>1447.35951</td>
<td>-</td>
</tr>
</tbody>
</table>

\[ R^2 = 1 - \text{Residual SS/ corrected SS} = 0.99803 \]
Table-4.4 Regression analysis depends on Number of failures

<table>
<thead>
<tr>
<th>Source</th>
<th>Degree of freedom</th>
<th>Sum of squares</th>
<th>Mean squares</th>
</tr>
</thead>
<tbody>
<tr>
<td>Regression</td>
<td>2</td>
<td>112046.03571</td>
<td>56023.0178</td>
</tr>
<tr>
<td>Residual</td>
<td>19</td>
<td>1331.96429</td>
<td>70.10338</td>
</tr>
<tr>
<td>Uncorrected total</td>
<td>21</td>
<td>213378.000</td>
<td>-</td>
</tr>
<tr>
<td>(corrected total)</td>
<td>20</td>
<td>51709.23810</td>
<td>-</td>
</tr>
</tbody>
</table>

\[ R^2 = 1 - \frac{\text{Residual SS}}{\text{corrected SS}} = 0.97424 \]

Table-4.5 Comparison results

<table>
<thead>
<tr>
<th>Model</th>
<th>(a)</th>
<th>(r)</th>
<th>MSE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Equation (4.9) of Burr Type III Model</td>
<td>134.140612</td>
<td>0.153455</td>
<td>63.427</td>
</tr>
<tr>
<td>G-O Model (Ohba(1984))</td>
<td>142.32</td>
<td>0.1246</td>
<td>2438.3</td>
</tr>
<tr>
<td>Exponential Model (Musa et. al (1987))</td>
<td>137.2</td>
<td>0.156</td>
<td>3019.66</td>
</tr>
<tr>
<td>Equation (4.8) with Rayleigh function</td>
<td>866.94</td>
<td>0.00962</td>
<td>89.2409</td>
</tr>
<tr>
<td>Delayed s-shaped Model (Huang &amp; Kuo (1997))</td>
<td>237.196</td>
<td>0.0963446</td>
<td>245.246</td>
</tr>
</tbody>
</table>
Figure 4.1 Observed/Estimated Current Testing-Effort Function vs. Time

Figure 4.2 Observed/Estimated Cumulative Test-Effort Function vs. Time
Figure 4.3 Observed/Estimated Cumulative Number of Failure vs. Time

Figure 4.4 Graph of Estimated Intensity Function
4.6 Conclusion

In this paper, test effort function representing the time dependent behavior of test effort spent during software testing phase have been described by Burr Type-III curve. In Burr Type-III testing effort function the proposed software reliability growth model fits the real software data set fairly well and in could give us reasonable description of resource consumption behavior. Comparison study shows that Burr Type-III model gives minimum error percentage rather than, G-O, Exponential Rayleigh and Delayed S-shaped model. From figure we also conclude that our model fit better as compare to other models.