CHAPTER 4

Burr Type XII Software Reliability

4.1 Introduction

Software reliability is one of the most important characteristics and is a key part in software quality. Its measurement and management technologies employed during the software life cycle are essential for producing and maintaining quality/reliable software systems. Software reliability is the probability that given software functions without failure in a given environmental condition during a specified time. That is it is the probability of failure-free execution of the software for a specified time in a specified environment (Musa, 1998). Over the last several decades, many software reliability growth models (SRGMs) like Crow and Basu (1988), Goel Okumoto (1979, 1984), Musa (1980), Pham (2005) and several models have been developed to greatly facilitate engineers and managers in tracking and measuring the growth of reliability as software is being improved (Pham.H, 2005).

Software reliability can be improved by increasing the testing effort and by correcting detected faults. Reliability tends to change continuously during testing due to the addition of problems in new code or to the removal of problems by debugging errors. There are two important parts to provide reliability: fault detection and fault isolation. The design has to consider both aspects. Since performance requirements influence the selection of data structures and algorithms, it is important to check performance factors at the design phase.

To estimate the performance of the design, the information on usage pattern, design structure, and installation characteristics are needed. The specifications describe the level and what security looks like while design considers its implementation. So, good engineering methods can largely improve software reliability. The study of software reliability can be categorized into three parts: modelling, measurement and improvement. Software reliability modelling has matured to the point that meaningful results can be obtained by applying suitable models to the problem. There are many models exist, one of the well-known and simplest model is our Burr type XII model.
There exist several software reliability growth models which can be used during the testing phase of the software development process to estimate the software reliability. Most software reliability models contain the following parts: assumptions, factors, and a mathematical function that relates the reliability with the factors. The mathematical function is usually higher order exponential or logarithmic. Software modelling techniques can be divided into two subcategories: prediction modelling and estimation modelling. Both kinds of modelling techniques are based on observing and accumulating failure data and analyzing with statistical inference.

The content of this chapter is published in the following journal.


4.2 Burr Type XII Model Formulation

Irving W. Burr (1942) introduced twelve different forms of cumulative distribution functions for modelling data. Among those twelve distribution functions, Burr-Type X and Burr-Type XII received the maximum attention. The Burr type XII uses a wide range of skewness and kurtosis, which may be used to fit any given set of unimodal data (Tadikamalla Pandu R). The reciprocal Burr (Burr Type III) covers a wide region that includes the region covered by Burr Type XII. Burr has suggested a number of forms of cumulative distribution functions (cdf) that would be useful for fitting data (Burr, 1942).

Software reliability models can be classified according to probabilistic assumptions. When a Markov process represents the failure process; the resultant model is called Markovian Model. Second one is fault counting model which describes the failure phenomenon by stochastic process like Homogeneous Poisson Process (HPP), Non Homogeneous Poisson Process (NHPP) and Compound Poisson Process etc. A majority of failure count models are based upon NHPP described in the following lines.
A software system is subjected to failures at random times caused by errors present in the system. Let \{N(t), t>0\} be a counting process representing the cumulative number of failures by time ‘t’. Since there are no failures at t=0 we have

\[ N(t) = 0 \]

It is to assume that the number of software failures during non-overlapping time intervals do not affect each other. In other words, for any finite collection of times \( t_1 < t_2 < \ldots < t_n \). The ‘n’ random variables \( \{N(t_2) - N(t_1)\}, \ldots, \{N(t_n) - N(t_{n-1})\} \) are independent. This implies that the counting process \{N(t), t>0\} has independent increments.

Let \( m(t) \) represents the expected number of software failures by time ‘t’. The mean value function \( m(t) \) is finite valued, non-decreasing, non-negative and bounded with the boundary conditions.

\[ m(t) = \begin{cases} 0, & t = 0 \\ a, & t \to \infty \end{cases} \]

Where ‘a’ is the expected number of software errors to be eventually detected.

Suppose \( N(t) \) is known to have a Poisson probability mass function with parameters \( m(t) \) i.e.,

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

Then \( N(t) \) is called an NHPP. Thus the stochastic behavior of software failure phenomena can be described through the \( N(t) \) process. Various time domain models have appeared in the literature (Kantam and Subbarao, 2009) which describe the stochastic failure process by an NHPP which differ in the mean value function \( m(t) \).

The proposed mean value function \( m(t) \) of Burr Type XII model is given by

\[ m(t) = a \left[ 1 - (1 + t^c)^{-b} \right] \]  \hspace{1cm} (4.2.1)

Where \( \left[ m(t)/a \right] \) is the cumulative distribution function of Burr type XII distribution for the present choice.
\[ p\{N(t) = n\} = \frac{[m(t)]^n e^{-m(t)}}{n!} \]

\[ \lim_{n \to \infty} P\{N(t) = n\} = \frac{e^{-a} a^n}{n!} \]

This is also a Poisson model with mean ‘a’.

Let \( N(t) \) be the number of errors remaining in the system at time ‘t’.

\[ N(t) = N(\infty) - N(t) \]

\[ E[N(t)] = E[N(\infty)] - E[N(t)] \]

\[ = a - m(t) \]

\[ = a - a \left[ 1 - (1 + t^e)^{-b} \right] \]

\[ = a (1 + t^e)^{-b} \]

Let \( S_k \) be the time between \((k-1)^{th}\) and \(k^{th}\) failure of the software product. Let \( X_k \) be the time up to the \(k^{th}\) failure. Let us find out the probability that time between \((k-1)^{th}\) and \(k^{th}\) failures, i.e., \(S_k\) exceeds a real number ‘s’ given that the total time up to the \((k-1)^{th}\) failure is equal to \(x\).

i.e., \[ P\left[ S_k > \frac{s}{X_{k-1}} = x \right] \]

\[ R S_k/X_{k-1}(s/x) = e^{-[m(x+s) - m(s)]} \quad (4.2.2) \]

This Expression is called Software Reliability.

### 4.3 Illustrating the Maximum Likelihood Estimation

The parameters ‘a’, ‘b’ and ‘c’ are estimated by using Maximum Likelihood method and the values can be computed using iterative method for the given cumulative time domain data. Using the estimators of ‘a’, ‘b’ and ‘c’ we can compute \(m(t)\).
Mathematical derivation for parameter estimation

We propose to access the software reliability based on Burr Type XII distribution model. The Burr distribution has a flexible shape and controllable scale and location which makes it appealing to fit to data. It is frequently used to model insurance claim sizes (Hee-cheul Kim, 2013). The mean value function and intensity function of Burr Type XII NHPP model are as follows.

The Cumulative distribution function (CDF) is given by

\[ m(t) = \int_{0}^{1} \lambda(t) dt = a \left[ 1 - \left(1 + t^c\right)^{-b} \right] \]

\[ = a F(t) \]

The Probability Density Function (PDF) of Burr XII distribution are given, respectively by

\[ \lambda(t) = a \left[ \frac{cb^{c-1}}{(1 + t^c)^{b+1}} \right] = a f(t) \]

Where \( t > 0, \ a > 0, \ b > 0 \) and \( c > 0 \) denote the expected number of faults that would be detached given infinite testing time in case of finite failure NHPP models. In order to have an assessment of the software reliability, \( a, b \) and \( c \) are unknown parameters and estimated by using Newton Raphson method. Expressions are now delivered for estimating ‘a’, ‘b’ and ‘c’ for the Burr type XII model.

The Likelihood function for time domain data is given by

\[ L = e^{-m(t)} \prod_{i=1}^{n} m^*(t_i) \]  \hspace{1cm} (4.3.1)

The values of \( a, b \) and \( c \) that would maximize \( L \) are called maximum likelihood estimators (MLE’s) and the method is called Maximum Likelihood (ML) method of estimation.

Take the mean value function of Burr Type XII is of the form

\[ m(t) = a \left[ 1 - \left(1 + t^c\right)^{-b} \right], \quad t \geq 0 \]  \hspace{1cm} (4.3.2)
Substituting Equation (4.3.2) in (4.3.1), we get

\[ L = e^{-a^{(1+e)^-b}} \cdot \prod_{i=1}^{n} \frac{abct_i^{c-1}}{(1+t_i^c)^{b+1}} \]

Applying the natural logarithm on both the sides, the log likelihood function is given by

\[ \log L = \log \left[ e^{-a^{(1+e)^-b}} \cdot \prod_{i=1}^{n} \frac{abct_i^{c-1}}{(1+t_i^c)^{b+1}} \right] \]

\[ \log L = \log e^{-a^{(1+e)^-b}} + \sum_{i=1}^{n} \log \frac{abct_i^{c-1}}{(1+t_i^c)^{b+1}} \]

\[ \log L = -a + a (1+t_i^c)^{-b} + \sum_{i=1}^{n} \log \frac{abct_i^{c-1}}{(1+t_i^c)^{b+1}} \]

\[ \log L = -a + a (1+t_i^c)^{-b} + \sum_{i=1}^{n} \left[ \log a + \log b + \log c + (c-1) \log t_i - (b+1) \log (1+t_i^c) \right] \] \hspace{1cm} (4.3.3)

The log likelihood equation is used to estimate the unknown parameters ‘a’, ‘b’ and ‘c’.

The parameters ‘a’, ‘b’ and ‘c’ would be the solutions of the equations

\[ \frac{\partial \log L}{\partial a} = 0, \]

\[ \frac{\partial \log L}{\partial b} = g(b) = 0 \]

\[ \frac{\partial^2 \log L}{\partial b^2} = g'(b) = 0 \]

\[ \frac{\partial \log L}{\partial c} = g(c) = 0 \]
\[ \frac{\partial^2 \LogL}{\partial c^2} = g'(c) = 0 \]

Substituting the expressions for \( m(t) \) in the above equations, taking logarithms, differentiating the equations with respect to ‘a’, ‘b’, ‘c’ and equating to zero and simplifying the equations we get

\[ \frac{\partial \LogL}{\partial a} = -1 + (1 + t^c)^{-b} + \sum_{i=1}^{n} \frac{1}{a} \]

\[ \frac{\partial \LogL}{\partial a} = 0 \]

\[ \Rightarrow -1 + (1 + t^c)^{-b} + \sum_{i=1}^{n} \frac{1}{a} = 0 \]

\[ \frac{n}{a} = 1 - \frac{1}{(1 + t^c)^b} \]

\[ a = \frac{n(1 + t^c)^b}{(1 + t^c)^b - 1} \quad (4.3.4) \]

The parameter ‘b’ is estimated by using Newton Raphson iterative Method

\[ b_{n+1} = b_n - \frac{g(b_n)}{g'(b_n)}, \text{ which is substituted in finding ‘a’. Where } g(b) \text{ & } g'(b) \text{ are expressed as follows.} \]

\[ \frac{\partial \LogL}{\partial b} = a(1 + t^c)^{-b} \cdot \log \left( \frac{1}{1 + t^c} \right) + \sum_{i=1}^{n} \left[ \frac{1}{b} \log \left( 1 + t_i^c \right) \right] \]

\[ \frac{\partial \LogL}{\partial b} = a(1 + t^c)^{-b} \cdot \log \left( \frac{1}{1 + t^c} \right) + \frac{n}{b} - \sum_{i=1}^{n} \log \left( 1 + t_i^c \right) \]

Substituting Equation (4.3.4) in the above Equation, we get
\[ \Rightarrow n (1 + t_i)^b / (1 + t_i)^b - 1 \cdot (1 + t_i)^b \cdot \log \left( \frac{1}{1 + t_i} \right) + n / b - \sum_{i=1}^{n} \log (1 + t_i^c) \]

\[ \Rightarrow \frac{n}{(1 + t_i)^b - 1} \cdot \log \left( \frac{1}{1 + t_i^c} \right) + \frac{n}{b} - \sum_{i=1}^{n} \log (1 + t_i^c) \]

Taking c value as 1, the g(b) is obtained as follows.

\[ g(b) = \frac{\partial \log L}{\partial b} = \frac{n \log \left( \frac{1}{(t+1)^b} \right)}{(t+1)^b - 1} + \frac{n}{b} - \sum_{i=1}^{n} \log (t_i + 1) \quad (4.3.5) \]

The second order partial derivative of L with respect to the parameter ‘b’ is given by

\[ g'(b) = \frac{\partial^2 \log L}{\partial b^2} = 0 \]

\[ \frac{\partial^2 \log L}{\partial b^2} = n \log \left( \frac{1}{(t+1)} \right) \left[ \frac{-1}{(t+1)^b - 1^2} \cdot (t+1)^b \cdot \log (t+1) \right] + n \left( \frac{-1}{b^2} \right) \]

\[ g'(b) = \frac{\partial^2 \log L}{\partial b^2} = -n \left[ \log \left( \frac{1}{t+1} \right) \left( \frac{(t+1)^b \log (t+1)}{(t+1)^b - 1^2} \right) + \frac{1}{b^2} \right] \quad (4.3.6) \]

The parameter ‘c’ is estimated by iterative Newton Raphson Method using

\[ c_{n+1} = c_n - \frac{g(c_n)}{g'(c_n)} \]

Where (c) and g'(c) are expressed as follows.

\[ g(c) = \frac{\partial \log L}{\partial c} = 0 \]
\[
\frac{\partial \text{Log}L}{\partial c} = -ab(1+t^b)^{-(b+1)}t^c \log t + \frac{n}{c} + \sum_{i=1}^{n} \log t_i - \sum_{i=1}^{n} \frac{(b+1)}{(1+t_i^b)}t_i^c \log t
\]

Substituting Equation (4.3.4) in the above Equation, we get

\[
\frac{n(1+t^b)^b}{(1+t^b)^{b+1}} - \frac{-b}{(1+t^b)^b - 1} \cdot \frac{1}{(1+t^b)^{b+1}}t^c \log t + \frac{n}{c} - \sum_{i=1}^{n} \log t_i - \sum_{i=1}^{n} \frac{(b+1)t_i^c \log t}{(1+t_i^b)}
\]

\[
\Rightarrow \frac{-bn}{(1+t^b)^b - 1} \cdot \frac{1}{(1+t^b)^{b+1}}t^c \log t + \frac{n}{c} - \sum_{i=1}^{n} \log t_i - \sum_{i=1}^{n} \frac{(b+1) \log t_i}{(1+t_i^b)} \cdot \frac{t_i^c}{(1+t_i^b)}
\]

If \(b\) is known and its value is considered as ‘1’, then the following equations are obtained.

\[
\Rightarrow g(c) = \frac{\partial \text{Log}L}{\partial c} = \frac{-n}{(1+t^b)} \log t + \frac{n}{c} - \sum_{i=1}^{n} \frac{2 \log t_i}{(1+t_i^b)} + \sum_{i=1}^{n} \log t_i
\]

Second order partial derivative of \(L\) with respect to the parameter ‘\(c\)’ which is denoted by \(g'(c)\) is performed.

\[
g'(c) = \frac{\partial^2 \text{Log}L}{\partial c^2} = 0
\]

\[
\frac{\partial^2 \text{Log}L}{\partial c^2} = (-n) \cdot \frac{-1}{(1+t^b)}t^c \log t + \frac{n}{c^2} \cdot \sum_{i=1}^{n} \left( \frac{1}{(1+t_i^b)^2} \log t_i \cdot \frac{-1}{c^2} \right) - \sum_{i=1}^{n} \frac{2 \log t_i}{(1+t_i^b)} \cdot \frac{(1+t_i^b) \log t_i - t_i^c \log t_i}{(1+t_i^b)^2}
\]

\[
g'(c) = \frac{\partial^2 \text{Log}L}{\partial c^2} = \frac{nt^c \log t}{(1+t^b)^2} \log t - \frac{n}{c^2} - 2 \log t \sum_{i=1}^{n} t_i^c \log t_i \cdot \frac{1}{(1+t_i^b)^2}
\]
4.4 Illustrations/ Time Domain Failure Data Sets

The failure control charts are generated considering various failure software processes and the procedure is illustrated with few examples in this section. Table 4.1 to 4.6 indicates the time between failures of different software products presented depending on the size of the data sets.

Table 4.1: IBM online Data Entry Software Testing (Pham Hong, 2005)

Testing is performed on an on-line data entry software package developed by IBM which is reported by Ohba(1984) and the inter failure times are recorded. The table given below exemplifies both the observation time and the cumulative number of errors that were detected.

<table>
<thead>
<tr>
<th>No. of Error</th>
<th>Inter-failure time</th>
<th>Cumulative failure time</th>
<th>No. of Error</th>
<th>Inter-failure time</th>
<th>Cumulative failure time</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>10</td>
<td>10</td>
<td>9</td>
<td>22</td>
<td>125</td>
</tr>
<tr>
<td>2</td>
<td>9</td>
<td>19</td>
<td>10</td>
<td>25</td>
<td>150</td>
</tr>
<tr>
<td>3</td>
<td>13</td>
<td>32</td>
<td>11</td>
<td>19</td>
<td>169</td>
</tr>
<tr>
<td>4</td>
<td>11</td>
<td>43</td>
<td>12</td>
<td>30</td>
<td>199</td>
</tr>
<tr>
<td>5</td>
<td>15</td>
<td>58</td>
<td>13</td>
<td>32</td>
<td>231</td>
</tr>
<tr>
<td>6</td>
<td>12</td>
<td>70</td>
<td>14</td>
<td>25</td>
<td>256</td>
</tr>
<tr>
<td>7</td>
<td>18</td>
<td>88</td>
<td>15</td>
<td>40</td>
<td>296</td>
</tr>
<tr>
<td>8</td>
<td>15</td>
<td>103</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 4.2: Naval Tactical System (NTDS) Software Error (Pham Hong, 2005)

NTDS Software Failure data set was extracted from information about failures in the development of software for the real time multi-computer complex of the US Naval Fleet Computer Programming Centre of the US Naval Tactical Data Systems (NTDS) (Goel 1979a). The software consists of 38 different project modules. The time horizon is divided into four phases: Production phase, test phase, user phase, and subsequent test phase. It was
observed that 26 failures were found during the production phase, five during the test phase and the last failure was found on 4 January 1971. One failure was observed during the user phase, in September 1971, and two failures during the test phase in 1971. The table describes 26 software failures that are formed during the production phase.

<table>
<thead>
<tr>
<th>No.of Error</th>
<th>Inter-failure time</th>
<th>Cumulative failure time</th>
<th>No.of Error</th>
<th>Inter-failure time</th>
<th>Cumulative failure time</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>9</td>
<td>9</td>
<td>14</td>
<td>9</td>
<td>87</td>
</tr>
<tr>
<td>2</td>
<td>12</td>
<td>21</td>
<td>15</td>
<td>4</td>
<td>91</td>
</tr>
<tr>
<td>3</td>
<td>11</td>
<td>32</td>
<td>16</td>
<td>1</td>
<td>92</td>
</tr>
<tr>
<td>4</td>
<td>4</td>
<td>36</td>
<td>17</td>
<td>3</td>
<td>95</td>
</tr>
<tr>
<td>5</td>
<td>7</td>
<td>43</td>
<td>18</td>
<td>3</td>
<td>98</td>
</tr>
<tr>
<td>6</td>
<td>2</td>
<td>45</td>
<td>19</td>
<td>6</td>
<td>104</td>
</tr>
<tr>
<td>7</td>
<td>5</td>
<td>50</td>
<td>20</td>
<td>1</td>
<td>105</td>
</tr>
<tr>
<td>8</td>
<td>8</td>
<td>58</td>
<td>21</td>
<td>11</td>
<td>116</td>
</tr>
<tr>
<td>9</td>
<td>5</td>
<td>63</td>
<td>22</td>
<td>33</td>
<td>149</td>
</tr>
<tr>
<td>10</td>
<td>7</td>
<td>70</td>
<td>23</td>
<td>7</td>
<td>156</td>
</tr>
<tr>
<td>11</td>
<td>1</td>
<td>71</td>
<td>24</td>
<td>91</td>
<td>247</td>
</tr>
<tr>
<td>12</td>
<td>6</td>
<td>77</td>
<td>25</td>
<td>2</td>
<td>249</td>
</tr>
<tr>
<td>13</td>
<td>1</td>
<td>78</td>
<td>26</td>
<td>1</td>
<td>250</td>
</tr>
</tbody>
</table>
Table 4.3: Time between Failures of a component (Xie.M. et al., 2002)

<table>
<thead>
<tr>
<th>No. of Error</th>
<th>Inter-failure time</th>
<th>Cumulative failure time</th>
<th>No. of Error</th>
<th>Inter-failure time</th>
<th>Cumulative failure time</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>30.02</td>
<td>30.02</td>
<td>16</td>
<td>15.53</td>
<td>151.78</td>
</tr>
<tr>
<td>2</td>
<td>1.44</td>
<td>31.46</td>
<td>17</td>
<td>25.72</td>
<td>177.5</td>
</tr>
<tr>
<td>3</td>
<td>22.47</td>
<td>53.96</td>
<td>18</td>
<td>2.79</td>
<td>180.29</td>
</tr>
<tr>
<td>4</td>
<td>1.36</td>
<td>55.29</td>
<td>19</td>
<td>1.92</td>
<td>182.21</td>
</tr>
<tr>
<td>5</td>
<td>3.43</td>
<td>58.72</td>
<td>20</td>
<td>4.13</td>
<td>186.34</td>
</tr>
<tr>
<td>6</td>
<td>13.2</td>
<td>71.92</td>
<td>21</td>
<td>70.47</td>
<td>256.81</td>
</tr>
<tr>
<td>7</td>
<td>5.15</td>
<td>77.07</td>
<td>22</td>
<td>17.07</td>
<td>273.88</td>
</tr>
<tr>
<td>8</td>
<td>3.83</td>
<td>80.9</td>
<td>23</td>
<td>3.99</td>
<td>277.87</td>
</tr>
<tr>
<td>9</td>
<td>21</td>
<td>101.9</td>
<td>24</td>
<td>176.06</td>
<td>453.93</td>
</tr>
<tr>
<td>10</td>
<td>12.97</td>
<td>114.87</td>
<td>25</td>
<td>81.07</td>
<td>535</td>
</tr>
<tr>
<td>11</td>
<td>0.47</td>
<td>115.34</td>
<td>26</td>
<td>2.27</td>
<td>537.27</td>
</tr>
<tr>
<td>12</td>
<td>6.23</td>
<td>121.57</td>
<td>27</td>
<td>15.63</td>
<td>552.9</td>
</tr>
<tr>
<td>13</td>
<td>3.39</td>
<td>124.96</td>
<td>28</td>
<td>120.78</td>
<td>673.68</td>
</tr>
<tr>
<td>14</td>
<td>9.11</td>
<td>134.07</td>
<td>29</td>
<td>30.81</td>
<td>704.49</td>
</tr>
<tr>
<td>15</td>
<td>2.18</td>
<td>136.25</td>
<td>30</td>
<td>34.19</td>
<td>738.68</td>
</tr>
</tbody>
</table>
Table 4.4: SONATA Data (Ashoka, M, 2010)

<table>
<thead>
<tr>
<th>No. of Error</th>
<th>Inter-failure time</th>
<th>Cumulative failure time</th>
<th>No. of Error</th>
<th>Inter-failure time</th>
<th>Cumulative failure time</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>52.5</td>
<td>52.5</td>
<td>16</td>
<td>52.5</td>
<td>852.25</td>
</tr>
<tr>
<td>2</td>
<td>52.5</td>
<td>105</td>
<td>17</td>
<td>35</td>
<td>887.25</td>
</tr>
<tr>
<td>3</td>
<td>26.25</td>
<td>131.25</td>
<td>18</td>
<td>52.5</td>
<td>939.75</td>
</tr>
<tr>
<td>4</td>
<td>52.5</td>
<td>183.75</td>
<td>19</td>
<td>105</td>
<td>1044.75</td>
</tr>
<tr>
<td>5</td>
<td>17.5</td>
<td>201.25</td>
<td>20</td>
<td>105</td>
<td>1149.79</td>
</tr>
<tr>
<td>6</td>
<td>105</td>
<td>306.25</td>
<td>21</td>
<td>105</td>
<td>1254.75</td>
</tr>
<tr>
<td>7</td>
<td>105</td>
<td>411.25</td>
<td>22</td>
<td>105</td>
<td>1359.75</td>
</tr>
<tr>
<td>8</td>
<td>21</td>
<td>432.25</td>
<td>23</td>
<td>52.5</td>
<td>1412.25</td>
</tr>
<tr>
<td>9</td>
<td>35</td>
<td>467.25</td>
<td>24</td>
<td>52.5</td>
<td>1464.75</td>
</tr>
<tr>
<td>10</td>
<td>35</td>
<td>502.25</td>
<td>25</td>
<td>52.5</td>
<td>1517.25</td>
</tr>
<tr>
<td>11</td>
<td>52.5</td>
<td>554.75</td>
<td>26</td>
<td>52.5</td>
<td>1569.75</td>
</tr>
<tr>
<td>12</td>
<td>52.5</td>
<td>607.25</td>
<td>27</td>
<td>105</td>
<td>1674.75</td>
</tr>
<tr>
<td>13</td>
<td>105</td>
<td>712.25</td>
<td>28</td>
<td>52.5</td>
<td>1727.25</td>
</tr>
<tr>
<td>14</td>
<td>35</td>
<td>747.25</td>
<td>29</td>
<td>52.5</td>
<td>1779.75</td>
</tr>
<tr>
<td>15</td>
<td>52.5</td>
<td>799.75</td>
<td>30</td>
<td>52.5</td>
<td>1832.25</td>
</tr>
<tr>
<td>No. of Error</td>
<td>Inter-failure time</td>
<td>Cumulative failure time</td>
<td>No. of Error</td>
<td>Inter-failure time</td>
<td>Cumulative failure time</td>
</tr>
<tr>
<td>-------------</td>
<td>--------------------</td>
<td>-------------------------</td>
<td>-------------</td>
<td>--------------------</td>
<td>-------------------------</td>
</tr>
<tr>
<td>1</td>
<td>3.183</td>
<td>3.183</td>
<td>28</td>
<td>59.183</td>
<td>355.165</td>
</tr>
<tr>
<td>2</td>
<td>3.700</td>
<td>6.883</td>
<td>29</td>
<td>13.333</td>
<td>368.498</td>
</tr>
<tr>
<td>3</td>
<td>4.667</td>
<td>11.55</td>
<td>30</td>
<td>65.167</td>
<td>433.665</td>
</tr>
<tr>
<td>4</td>
<td>4.833</td>
<td>16.383</td>
<td>31</td>
<td>115.000</td>
<td>548.665</td>
</tr>
<tr>
<td>5</td>
<td>4.833</td>
<td>21.216</td>
<td>32</td>
<td>55.000</td>
<td>603.665</td>
</tr>
<tr>
<td>6</td>
<td>6.417</td>
<td>27.633</td>
<td>33</td>
<td>25.167</td>
<td>628.832</td>
</tr>
<tr>
<td>7</td>
<td>9.500</td>
<td>37.133</td>
<td>34</td>
<td>3.250</td>
<td>632.082</td>
</tr>
<tr>
<td>8</td>
<td>10.167</td>
<td>47.3</td>
<td>35</td>
<td>32.600</td>
<td>664.682</td>
</tr>
<tr>
<td>9</td>
<td>6.083</td>
<td>53.383</td>
<td>36</td>
<td>2.250</td>
<td>666.932</td>
</tr>
<tr>
<td>10</td>
<td>6.500</td>
<td>59.883</td>
<td>37</td>
<td>11.017</td>
<td>677.949</td>
</tr>
<tr>
<td>11</td>
<td>4.583</td>
<td>64.466</td>
<td>38</td>
<td>0.833</td>
<td>678.782</td>
</tr>
<tr>
<td>12</td>
<td>6.000</td>
<td>70.466</td>
<td>39</td>
<td>12.150</td>
<td>690.932</td>
</tr>
<tr>
<td>13</td>
<td>13.333</td>
<td>83.799</td>
<td>40</td>
<td>15.000</td>
<td>705.932</td>
</tr>
<tr>
<td>14</td>
<td>20.167</td>
<td>103.966</td>
<td>41</td>
<td>3.000</td>
<td>708.932</td>
</tr>
<tr>
<td>15</td>
<td>6.783</td>
<td>110.749</td>
<td>42</td>
<td>70.417</td>
<td>779.349</td>
</tr>
<tr>
<td>16</td>
<td>0.833</td>
<td>111.582</td>
<td>43</td>
<td>260.000</td>
<td>1039.349</td>
</tr>
<tr>
<td>17</td>
<td>11.000</td>
<td>122.582</td>
<td>44</td>
<td>0.000</td>
<td>1039.349</td>
</tr>
<tr>
<td>18</td>
<td>25.117</td>
<td>147.699</td>
<td>45</td>
<td>0.000</td>
<td>1039.349</td>
</tr>
<tr>
<td>19</td>
<td>10.417</td>
<td>158.116</td>
<td>46</td>
<td>5.000</td>
<td>1044.349</td>
</tr>
<tr>
<td>20</td>
<td>15.200</td>
<td>173.316</td>
<td>47</td>
<td>150.350</td>
<td>1194.699</td>
</tr>
<tr>
<td>21</td>
<td>10.633</td>
<td>183.949</td>
<td>48</td>
<td>41.983</td>
<td>1236.682</td>
</tr>
<tr>
<td>22</td>
<td>4.883</td>
<td>188.832</td>
<td>49</td>
<td>114.833</td>
<td>1351.515</td>
</tr>
<tr>
<td>23</td>
<td>20.200</td>
<td>209.032</td>
<td>50</td>
<td>55.800</td>
<td>1407.315</td>
</tr>
<tr>
<td>24</td>
<td>10.200</td>
<td>219.232</td>
<td>51</td>
<td>45.833</td>
<td>1453.148</td>
</tr>
<tr>
<td>25</td>
<td>11.250</td>
<td>230.482</td>
<td>52</td>
<td>111.250</td>
<td>1564.398</td>
</tr>
<tr>
<td>26</td>
<td>20.250</td>
<td>250.732</td>
<td>53</td>
<td>115.750</td>
<td>1680.148</td>
</tr>
<tr>
<td>27</td>
<td>45.250</td>
<td>295.982</td>
<td>54</td>
<td>131.650</td>
<td>1811.798</td>
</tr>
</tbody>
</table>
Table 4.6 AT&T Data Set (Pham Hong, 2005)

<table>
<thead>
<tr>
<th>No. of Error</th>
<th>Inter-failure time</th>
<th>Cumulative failure time</th>
<th>No. of Error</th>
<th>Inter-failure time</th>
<th>Cumulative failure time</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>5.5</td>
<td>5.5</td>
<td>12</td>
<td>14.6</td>
<td>167</td>
</tr>
<tr>
<td>2</td>
<td>1.83</td>
<td>7.33</td>
<td>13</td>
<td>11.41</td>
<td>178.41</td>
</tr>
<tr>
<td>3</td>
<td>2.75</td>
<td>10.08</td>
<td>14</td>
<td>18.94</td>
<td>197.35</td>
</tr>
<tr>
<td>4</td>
<td>70.89</td>
<td>80.97</td>
<td>15</td>
<td>65.3</td>
<td>262.65</td>
</tr>
<tr>
<td>5</td>
<td>3.94</td>
<td>84.91</td>
<td>16</td>
<td>0.04</td>
<td>262.69</td>
</tr>
<tr>
<td>6</td>
<td>14.98</td>
<td>99.89</td>
<td>17</td>
<td>125.67</td>
<td>388.36</td>
</tr>
<tr>
<td>7</td>
<td>3.47</td>
<td>103.36</td>
<td>18</td>
<td>82.69</td>
<td>471.05</td>
</tr>
<tr>
<td>8</td>
<td>9.96</td>
<td>113.32</td>
<td>19</td>
<td>0.46</td>
<td>471.51</td>
</tr>
<tr>
<td>9</td>
<td>11.39</td>
<td>124.71</td>
<td>20</td>
<td>31.61</td>
<td>503.12</td>
</tr>
<tr>
<td>10</td>
<td>19.88</td>
<td>144.59</td>
<td>21</td>
<td>129.31</td>
<td>632.43</td>
</tr>
<tr>
<td>11</td>
<td>7.81</td>
<td>152.4</td>
<td>22</td>
<td>47.6</td>
<td>680.03</td>
</tr>
</tbody>
</table>

### 4.5 Parameter Estimation

The parameters a, b and c are computed for the data sets specified in Table 4.1 to Table 4.6 using well known Newton Raphson method. The values of ‘b’ and ‘c’ obtained with Newton Raphson method are substituted in the Equation (4.3.4) to obtain the value of ‘a’. The model has been applied on several live data sets the values of unknown parameters ‘a’, ‘b’ and ‘c’ are computed for each data set and the results are exhibited in the Table 4.7. For a software system the failures are random and can take place both during design phase or analysis phase and in some cases due to insufficient testing of software. In this chapter the Burr Type XII model has been applied to the time between failures data. The distribution uses cumulative time between failure data for reliability monitoring.
## Table 4.7: Parameter Estimation for Time Domain data

<table>
<thead>
<tr>
<th>Failure Data Set</th>
<th>No. of Failures</th>
<th>Estimated Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>a</td>
</tr>
<tr>
<td>IBM (Pham Hong, 2005)</td>
<td>15</td>
<td>15.05105</td>
</tr>
<tr>
<td>NTDS (Pham Hong, 2005)</td>
<td>26</td>
<td>26.10527</td>
</tr>
<tr>
<td>Xie (Xie.M. et al., 2002)</td>
<td>30</td>
<td>30.04080</td>
</tr>
<tr>
<td>SONATA (Ashoka, M, 2010)</td>
<td>30</td>
<td>30.01639</td>
</tr>
<tr>
<td>S2 (Michael R. Lyu, 1996)</td>
<td>54</td>
<td>54.02986</td>
</tr>
<tr>
<td>AT &amp; T (Pham Hong, 2005)</td>
<td>22</td>
<td>22.03246</td>
</tr>
</tbody>
</table>

The estimator of the Reliability function from the Equation (4.2.2) at any time \( x \) is given by

\[
R_{S_k/X_{k-1}}(s/x) = e^{-[m(x+s)−m(s)]}
\]

(i) The Reliability of the IBM online Data Entry Software Testing System Test data is given by

\[
R_{S_k/X_{k-1}}(296/50) = e^{-[m(50+296)−m(296)]}
\]

\[
= e^{-[m(346)−m(296)]}
\]

\[
= e^{-[15.00734658−15.000000018]}
\]

\[
= e^{-[0.007346404]}
\]

\[
= 0.992680514
\]
(ii) The Reliability of the Naval Tactical System (NTDS) Software System Test data is given by

\[
R_{S_k/X_{k-1}}(250/50) = e^{-[m(50+250)-m(250)]}
= e^{-[m(300)-m(250)]}
= e^{-[26.0174525-26.00000008]}
= e^{-[0.01745168]}
= 0.98269972
\]

(iii) The Reliability of the Xie System Test data is given by

\[
R_{S_k/X_{k-1}}(738.68/50) = e^{-[m(50+738.68)-m(738.68)]}
= e^{-[m(788.68)-m(738.68)]}
= e^{-[30.002617-30.00000375]}
= e^{-[0.00257955]}
= 0.99742377
\]

(iv) The Reliability of the SONATA System Test data is given by

\[
R_{S_k/X_{k-1}}(1832.25/50) = e^{-[m(50+1832.25)-m(1832.25)]}
= e^{-[m(1882.25)-m(1832.25)]}
= e^{-[30.00043772-30.000000267]}
= e^{-[0.000435055]}
= 0.999565039
\]
(v) The Reliability of the S2 System Test data is given by

\[ R_{S_k/X_{k-1}}(1811.798/50) = e^{-(m(50 + 1811.798) - m(1811.798))} \]

\[ = e^{-(m(1861.798) - m(1811.798))} \]

\[ = e^{-(54.00081378 - 54.0001268)} \]

\[ = e^{-0.00068696} \]

\[ = 0.999199213. \]

(vi) The Reliability of the AT & T System Test data is given by

\[ R_{S_k/X_{k-1}}(680.03/50) = e^{-(m(50 + 680.03) - m(680.03))} \]

\[ = e^{-(m(730.03) - m(680.03))} \]

\[ = e^{-(22.00222068 - 22.0000142)} \]

\[ = e^{-0.00221257} \]

\[ = 0.997783204. \]

4.6 Method of Performance Analysis

The performance of SRGM is judged by its ability to fit the software failure data. The term goodness of fit denotes the question of “How good does a mathematical model fit to the data”? In order to validate the model under study and to assess its performance, experiments on a set of actual software failure data have been performed. The considered model fits more to the dataset whose Log Likelihood is most negative. The application of the considered distribution function and its Log Likelihood on different datasets collected from real world failure data is given in Table 4.8.
### Table 4.8 The Results on different datasets

<table>
<thead>
<tr>
<th>Dataset</th>
<th>No. of Failures</th>
<th>Reliability ($t_{n+50}$)</th>
<th>Log L</th>
</tr>
</thead>
<tbody>
<tr>
<td>IBM (Pham Hong, 2005)</td>
<td>15</td>
<td>0.99268051</td>
<td>-27.365034</td>
</tr>
<tr>
<td>NTDS (Pham Hong, 2005)</td>
<td>26</td>
<td>0.98269972</td>
<td>-38.477204</td>
</tr>
<tr>
<td>Xie (Xie.M. etal, 2002)</td>
<td>30</td>
<td>0.99742377</td>
<td>-44.402007</td>
</tr>
<tr>
<td>SONATA (Ashoka, M, 2010)</td>
<td>30</td>
<td>0.99956503</td>
<td>-46.991596</td>
</tr>
<tr>
<td>S2 (Michael R. Lyu, 1996)</td>
<td>54</td>
<td>0.99919921</td>
<td>-70.349950</td>
</tr>
<tr>
<td>AT &amp; T (Pham Hong, 2005)</td>
<td>22</td>
<td>0.99778320</td>
<td>-18.949300</td>
</tr>
</tbody>
</table>

### 4.7 Conclusion

Software reliability is an important quality measure that quantifies the operational profile of computer systems. This model is primarily useful in estimating and monitoring software reliability, viewed as a measure of software quality. In this thesis we have discussed the performances of 6 time domain datasets by using our new Burr type XII SRGM. The experiment result shows that the S2 system test data (Table 4.5) can provide a better goodness-of-fit compared with other datasets. Since, it is having the highest negative value for the log likelihood. This is a simple method for model validation and is very convenient for practitioners of software reliability. The reliability of all the data sets are given in Table 4.8. The reliability of the model over SONATA data is high among the data sets which were considered.
// Program for computing the values of ‘a’, ‘b’ and ‘c’ using Newton-Raphson method.
#include<stdio.h>
#include <conio.h>
#include <math.h>
#define N 22
double g(double b, int s[], int sn);
double gdash(double b, int s[], int sn);
double gc(double c, int s[], int sn);
double gcdash(double c, int s[], int sn);
void main()
{
    int i, j, sk;
    int s[N]={5.50,7.33,10.08,80.97,84.91,99.89,103.36,113.32,124.71,144.59,
    152.40,167.00,178.41,197.35,262.65,262.69,388.36,471.05,471.51,503.12,632.43,680.03};
    double savg, g1, g2, g3, g4, a;
    double b[25], c[25];
    double f1, f2, f3, x, y;
    clrscr();
    sk=0;
    printf("n ****************NEWTON RAPHSON METHOD ******");
    c[0]=b[0]=1;
    i=-1;
    do
    {
        i=i+1;
        g1=g(b[i], s, s[N-1]);
        g2=gdash(b[i], s, s[N-1]);
        b[i+1]=b[i]-(g1/g2);
        printf("n b[%d]=%f b[%d]=%f", i, b[i], i+1, b[i+1]);
        printf("n t\t t b[%d]=%f", i+1, fabs(b[i+1]-b[i]));

while(fabs(b[i+1]-b[i])>=0.1));
j=-1;
do{
  j=j+1;
g3=gc(c[j],s,s[N-1]);
g4=gcdash(c[j],s,s[N-1]);
c[j+1]=c[j]-(g3/g4);
printf("\n\n\nc[%d]=%fc[%d]=%f",j,c[j],j+1,c[j+1]);
printf("\n\t\tc[%d] - c[%d]=%f",j+1,j,fabs(c[j+1]-c[j]));
}while(fabs(c[j+1]-c[j])>=0.1); 
x=(double)s[N-1];
y=pow(x,c[j+1]);
f1=N*(pow(1+y,b[i+1]));
f2=pow(1+y,b[i+1])-1;
a=f1/f2;
printf("\n\n\nb[%d]=%f is the MLE of b=%f",i+1,b[i+1],b[i+1]);
printf("\n\nc[%d]=%f is the MLE of c=%f",i+1,c[j+1],c[j+1]);
printf("\n\n\t\ta=%f",a);
getch();
}
double g(double b,int s[N],int sn)
{
  int k;
  double ss,c1,c2,c3=0.0,c4,gval,p;
  p=(double)sn;
  for(k=0;k<N;k++)
  {
    ss=(double)s[k];
  }
c3 = c3 + log(ss+1);
}
c1 = (N/(pow(p+1,b)-1))*(log(1/(p+1)));
c2 = N/b;
gval = c1 + c2 - c3;
return gval;
}
double gdash(double b, int s[N], int sn)
{
    double gdval, c1, c2, c3, c4, p;
p = (double) sn;
c1 = pow(p+1, b)*log(p+1);
c2 = (pow(p+1, b)-1)*(pow(p+1, b)-1);
c4 = log(1/(p+1));
gdval = (N*c4*(c2/c3))+(N/b*b);
return gdval;
}
double gc(double c, int s[N], int sn)
{
    int k;
double gcval, c1, c2, c3 = 0.0, c4 = 0.0, c5 = 0.0, p, n, q, r, h;
p = (double) sn;
q = log(p);
for(k=0; k<N; k++) {
    //r = pow(s[k], c);
    //h = (double) s[k];
c3 = c3 + pow(s[k], c)/(1+pow(s[k], c));
    //c3 = c3 + n + log(h);
}
for(k=0; k<N; k++)

{ 
c5=c5+log(s[k]);
}
c1=(N/(pow(p,c)+1))*q;
c2=N/c;
c4=2*q*c3;
gcval=-c1+c2-c4+c5;
return gcval;
}
double gcdash(double c,int s[N],int sn)
{
int k;
double gcdval,c1,c2,c3=0.0,c4,p,d,e,g,m,l,u;
p=(double)sn;
g=pow(p,c);
for(k=0;k<N;k++)
{
  //d=pow(s[k],c);
  //u=(double)s[k];
  e=1/(1+pow(s[k],c))*(1+pow(s[k],c));
  c3=c3+(pow(s[k],c)*log(s[k])*e);
}
c1=((g*log(p)*log (p)*N)/((1+g)*(1+g)));
//l=2*log(p)*N;
//m=c1*l;
c2=N/(c*c);
c4=2*log(p);
gcdval=(c1-c2)-(c4*c3);
return gcdval;
}