CHAPTER 4

PARAMETER ESTIMATION METHODS

4.1 INTRODUCTION

After choosing a model, there remains the problem of determining its parameters to obtain a specific form of the model. There are two methods; parameter prediction and parameter estimation. Parameter prediction tries to establish the parameters of a model from the properties of the software product and the development process. It has the advantage that, it can be applied before the software is tested that is, before failure data is available. Procedure for predicting the values of the execution time component parameters are presently developed just for the basic execution model.

There are two categories of parameters that must be predicted for all software reliability models which are based on calendar time components. The parameters are the same and have the same values for the models. There are two types of parameters namely planned and resource usage. The planned parameters are established by project objectives and available resources. The resource usage parameters related to the resources required for failure identification and correction. It is believed that ultimate values of these parameters can be determined for all software projects or for large classes of projects (John Musa 2010).

A sample model where parameter prediction can be used is the basic execution time model which has the following mean value function;
Here \( v_0 \), the total number of failures that would be experienced in infinite time and \( \lambda_0 \) the initial failure intensity, can be predicted using relation

\[
\mu(\tau) = v_0 = \left[ 1 - \exp\left( -\frac{\lambda_0}{v_0} \tau \right) \right]
\]  
(4.1)

where \( \beta \) is the fault reduction factor, \( f(w_0) \), the number of inherent faults, \( f \) is the linear execution frequency of the program and \( k \) is the fault exposure ratio of the program. It accounts for the following facts;

- Programs are not generally executed in “straight line” fashion, but have many loops and branches.
- The machine state varies and hence the fault associated with an instruction may or may not be exposed at one particular execution of the instruction.

Parameter estimation is used in subsystem or system or operational phase where failure data are available. It is a statistical method trying to estimate model parameters based on failure times or number of failures per time interval. Point estimation is used to determine the parameters and interval estimation to compute confidence limits for the parameters that are useful in evaluating the accuracy of the estimates.

Point estimation is often used in reliability analysis. Each fault injection and each failure occurrence can be treated as a sample, which is assumed to be independent of other samples.
Given a collection of \( n \) sampling outcomes, \( x_1 \ldots x_n \) of a random variable \( X \), each \( x_i \) can be considered as a realization of the random variable \( X_i \). The set \( \{X_1, X_2, \ldots X_n\} \) is called a random sample of \( X \). The function \( \hat{\theta} = \hat{\theta} \{X_1, X_2, \ldots X_n\} \), is called an estimator of \( \theta \). \( \hat{\theta} \{X_1, X_2, \ldots X_n\} \) is said to be point estimate of \( \theta \) (Michael Lyu 1996).

Interval estimation may deviate from the actual parameter value. To obtain an estimate with a high confidence, it is necessary to construct an interval estimate such that the interval includes the actual parameter value with a high probability. Given an estimator \( \hat{\theta} \), if

\[
P(0 = e_1 < \hat{\theta} < \hat{\theta} + e_2) = \beta
\]

(4.3)

The random interval \( (0 = e_1, \hat{\theta} + e_2) \) is said to be 100 \( x \) \( \beta \) percent confidence interval \( \theta \) and \( \beta \) is called the confidence coefficient.

The study of connection with software reliability growth models to the software testing and debugging process has many reasons. First, if the parameters have an interpretation, then they constitute a metric for the software test process and the software under test. Secondly, it may be possible to estimate the parameters even before testing begin. The priori values can serve as a check for the values computed at the beginning of testing and also serve as initial estimates for prediction (Yaswant k. Malaiya and Jason Denton 1997).

In this chapter two widely used parameter estimation methods, namely Maximum Likelihood estimation and Least Square estimation are given. The proposed approach for parameter estimation for Gompertz reliability model is also discussed.
4.2 MAXIMUM LIKELIHOOD ESTIMATION

The most widely used formal estimation technique is the method of Maximum Likelihood. Estimation by Maximum Likelihood is a general technique that may be applied. The Maximum likelihood method is to choose an estimator such that the observed sample is the most likely to occur among all possible samples. The method usually produces estimators that have minimum variance and consistency properties. Maximum likelihood estimation is considered for “failure data” and “grouped data”. Results for binomial and Poisson type models are given for both types of data.

The foundation of the Maximum Likelihood method is the likelihood function. The function is defined as the joint density of the observed data, \( L(\beta, Y_D) \) \( \beta \) is unknown parameter and \( Y_D \) represents a set of observations. For each data set, let \( \hat{\beta} \) be the values of the parameters that make \( L(\beta, Y_D) \) as large as possible. These maximizing values will be functions of the data. The function themselves are called the Maximum Likelihood estimators. The values of these functions take on are known as the maximum likelihood estimates. Maximum likelihood estimators possess many desirable optimum properties such as consistency, efficiency and asymptotic normality. An estimator is said to be consistent, if its variance tends to zero and if its expectation tends to the true population parameter as the sample size tends to infinity. If two different estimators have the same expectation, then the one with the smaller variance is said to be more efficient. Furthermore, an estimator is called asymptotically normal, if its distribution is almost normal for sufficient large sample size.

Estimating the unknown parameter of a specific software reliability model will usually require the optimization of a particular function. For Maximum Likelihood estimation the log-likelihood function is to maximized. Two basic methods can be considered,
• Numerical root finding procedures

• Searching schemes

Assume that the mean value function $\mu(t)$ includes $w+1$ model parameters $\beta_k$ $(k = 0, 1, 2, ..., w)$. Suppose that the data set on $m_e$ failure occurrence times is observed. The likelihood function for the $w+1$ unknown parameters in the non homogeneous Poisson process model with $\mu(t)$ at given $t = t_1, t_2, ..., t_e$ is;

$$L(\beta) = \exp(-\mu(t_e) \sum_{i=1}^{m_e} \lambda(t_i))$$ (4.4)

The logarithm of the likelihood function yields,

$$\ln L(\beta) = \sum_{i=1}^{m_e} \ln (\lambda(t_i) - \mu(t_e))$$ (4.5)

Then the maximum likelihood estimates $\hat{\beta}(k=0,1,...,w)$ can be obtained by solving the likelihood equations;

$$\frac{\partial \ln L(\beta)}{\partial \beta_k} = 0, k = 0, 1, ..., w$$ (4.6)

In Poisson type model the first parameter is given by,

$$\hat{\beta}_0 = \frac{m_e \beta_0}{\mu(t_e; \beta_0, \beta_1, ..., \beta_k)}$$ (4.7)

The other parameters must be determined by finding the root of (4.5), if the model has just two parameters, only $\beta$ is needed to determine.
Since the Gompertz Software Reliability Model has a stronger non-linearity than the existing NHPP based software reliability model, the computation procedure of Maximum Likelihood estimates is more complex. And also classical iterative algorithms such as the Newton’s method are possibly used to solve the likelihood equations numerically. Since the Newton’s method is one of unconstrained optimization algorithms, it does not often function well to compute the estimates of Gompertz SRM. Moreover the local convergence property of the Newton’s method adversely affects the estimation procedure in Gompertz SRM (Koji Ohishi et al 2008). Our work we have taken Least Mean Square estimation is selected because it is simple and easy to implement. It functions well to compute the estimates of Gompertz SRM.

Several good searching schemes using gradient information are available. This scheme is suitable for very few applications. In software reliability modeling neither of these methods displays suitable convergence in practice (Michael Lyu 1996 and Shooman 2002).

4.3 LEAST MEAN SQUARE ESTIMATION

The Least Mean Square (LMS) estimation algorithm was introduced by Widrow and Hoff in 1959 as an adaptive algorithm, which uses a gradient-based method of steepest decent. Compared to other algorithms LMS algorithm is relatively simple. It neither requires correlation function calculation nor does it require matrix inversions (Tsai et al (2004)). LMS estimation is a method for predicting or estimating the value of a single random variable ‘y’ from a single measurement ‘x’, when certain condition of linearity can be assumed. This method is based on the observed outcome sequence of software runs during software testing (Bo Yang et al (2008)). There are three steps in analysis;

- Preliminary examination of sample data
• Estimation of a regression line
• Computation of confidence limits

**Preliminary estimation of sample data:** In a linear regression analysis one must first conduct a preliminary examination of sample data to determine the validity of an assumption of linear dependence. Suppose that the data consist of a set \([(x_i, y_i), i=1,...,n]\) of \(n\) paired observations of measurement ‘\(x\)’ and measurement ‘\(y\)’, then the simplest method of examining the data is to develop a scatter diagram of the data by plotting the coordinates \([(x_i, y_i),..., (x_n, y_n)]\) of \(n\) paired measurements. The scatter diagram provides a visual display of the relationship of the data. If the points in the scatter diagram seem to fall along a line, there is an indication that values of \(y\) are on the average, linearly dependent on values of ‘\(x\)’. Hence the data are appropriate.

**Estimation of a Regression Line:** If a preliminary examination of sample data suggests that it is reasonable to assume a linear dependence of \(y\) on \(x\), postulate that the mean of \(y\) is linearly related to \(x\). Therefore, we have,

\[ E[Y] = \alpha + \beta x \]

Where \(\alpha\) and \(\beta\) are parameters to be estimated from the data. Note that \(\alpha\) and \(\beta\) represent the intercept and slope of the line respectively.

The principle use of the method of LMS is to determine the best fit of a linear function to the data\([(x_i, y_i), i=1,...,n]\). It minimizes the sum of the squares of the deviation between what we expect and what we actually observed. We determine the parameters so that the sum of squares of errors is minimized.
\[ S = \sum_{i=1}^{n} c_i \]

\[ S = \sum_{i=1}^{n} [y_i - (\alpha + \beta x_i)] \text{ is minimized} \]

Differentiating \( S \) with respect to \( \alpha \) and \( \beta \),

\[ \frac{\partial S}{\partial \alpha} = -2 \sum_{i=1}^{n} y_i - (\alpha + \beta x_i) \]

(4.8)

\[ \frac{\partial S}{\partial \beta} = -2 \sum_{i=1}^{n} y_i - (\alpha + \beta x_i) \]

(4.9)

By setting the partial derivatives equal to 0 the normal equations can be obtained. The solution of the normal equations is the Least Squares estimates.

**Computation of confidence limits:** Having obtained a least squares estimate of the regression line, the next step of regression analysis is the computation of confidence limit for the intercept, slope, or regression line that are useful in evaluating the accuracy of the estimate.

The Least Mean Square algorithm is most commonly used algorithm because of its simplicity and a reasonable performance. Since it is iterative algorithm it can be used in a highly time varying testing environment (Musa and Okumo 1984).

4.4 PROPOSED APPROACH OF LMS
The work aims to identify effective parameter estimation method in software reliability. In particular we have found a better estimation method using LMS algorithm. In LMS algorithm, number of defects and time interval are the two variables used to estimate the model parameters. But in software reliability, number of defects are always depend on number of test cases. To effectively use the reliability models and defect data during software testing we propose to additionally apply the strategy called number of test cases for the particular module so that the number of defects are considered with respect to number of test cases. This strategy has been used in this work.

The Gompertz software reliability model has unknown parameters. These unknown parameters are determined using Least mean Square estimation method. The Gompertz model equation is given as;

\[ \mu(t) = ab^c t \]  \hspace{1cm} (4.10)

Model assumptions:

- Failure observation/fault detection phenomenon is modeled by NHPP.
- Software is subject to failures during execution caused by faults remaining in the software.
- Each time a failure occurs, the fault that caused it is immediately and perfectly detected, and no new faults are introduced.

The Equation (4.10) are written with the distributed normal random variables with mean 0, 

\[ \mu(t) = ab^c t u_t \]  \hspace{1cm} (4.11)

Using logarithmic transformation we have,
\[ \log \mu(t) = \log a + c^t \log b + \log u_t \quad (4.12) \]

\[ Y_t = A + C^t B + U_t \]

Where \( Y_t = \log \mu(t), \ C = c, \ B = \log b \) and \( U_t = \log u_t \).

From the least squares technique that is by taking partial derivatives of \( \sum_{t=0}^{n} U_t^2 \) with respect to \( A, \ B \) and \( C \) and equating each on of them to zero we obtain

\[ G_{N1} = \sum_{t=1}^{n}(Y_t - A + C^t B + U_t) = 0 \quad (4.13) \]

\[ G_{N2} = \sum_{t=1}^{n}(Y_t - A - BC^t)C = 0 \quad (4.14) \]

\[ G_{N3} = \sum_{t=1}^{n}(Y_t - A - BC^t)B^tC^{-1} = 0 \quad (4.15) \]

The estimates of \( A, \ B \) and \( C \) are obtained by solving the above system of Gauss-Normal Equations (4.13), (4.14) and (4.15) using least mean squares method. The LMS is given by,

\[ \sum_{t=1}^{n}(Y_t - \bar{Y})^2 = \frac{\left[ \sum_{t=1}^{n} (Y_t - \bar{Y})^2 (c^t - \bar{c})^2 \right]}{\sum_{t=1}^{n} (c^t - \bar{c})^2} \quad (4.16) \]

Where \( \bar{Y} = \left( \frac{1}{n} \right) \sum_{t=1}^{n} Y_t \) and \( \bar{C} = \left( \frac{1}{n} \right) \sum_{t=1}^{n} C^t \)

Further, to obtain analytical expressions for \( C \), this satisfies (4.13), (4.14) and (4.15) and also minimizes (4.15). In such a way the estimates of \( A \) and \( B \) are given by

\[ \hat{A} = \bar{Y} - \bar{B} \hat{C} \quad (4.17) \]

\[ \hat{B} = \frac{\left[ \sum_{t=1}^{n} (Y_t - \bar{Y}) (C^t - \bar{C}) \right]}{\sum_{t=1}^{n} (C^t - \bar{C})^2} \quad (4.18) \]
Gauss-Newton method of Least mean squares is applied in the algorithm to obtain the estimators of A, B and C and their goodness of fit measures such as D where

\[ D = \sum_{t=1}^{n} \frac{(Y_t - \hat{Y}_t)^2}{(Y_t - \bar{Y})^2} \]  

(4.19)

Where \( \log \hat{Y}_t = \log \hat{a} + \hat{c}^t \cdot \log \hat{b} \). This Gauss–Newton method consists of taking linear expansion of \( Y_t = f_t(A, B, C) = A + B \cdot C^t \) around \( A_0, B_0 \) and \( C_0 \) and retaining the first degree terms and then using ordinary least squares method to obtain A, B and C. The initial values of \( A_0, B_0 \) and \( C_0 \) are given as follows;

\[ C_0 = \left( \frac{D_2}{D_1} \right)^{1/2} \]  

(4.20)

\[ B_0 = [(1 - C_0)/C_0] \cdot \left[ D_1^2/(D_1 - D_2)^2 \right] \]  

(4.21)

\[ A_0 = (1/3r) \left[ (s_1 + s_2 + s_3) - \left[ (D_1^2 + D_1D_2 + D_2^2)/(D_1 - D_2) \right] \right] \]  

(4.22)

Where

\[ S_1 = \sum_{t=1}^{r} Y_t, \]

\[ S_2 = \sum_{t=r+1}^{2r} Y_t, \]

\[ S_3 = \sum_{t=2r+1}^{3r} Y_t, \]

\[ D_1 = S_1 - S_2 \]

\[ D_2 = S_2 - S_3 \]

And without the loss of generality it is assumed that the sample size for calculating the initial estimates is a multiple of three. The above analytical expressions are obtained by the deterministic relation,

\[ Y_t = e^{-\lambda t} \]  

(4.23)
\[ \lambda = -d(1 - c/m) \]

Where \( d \) is number of defects, \( m \) is total number of test cases, \( c \) is number of test cases executed at the time period \( t \).

This procedure is repeated with the new testing results and procedure is terminated if the reliability percent is above the expected values. A simulation tool for estimating Gompertz software reliability model using least mean squares is built. It is based on the estimation of proposed technique. The proposed technique have been validated and evaluated on actual software reliability data cited from real software development projects and compared with existing technique. The model, data and the working methods have been discussed in Chapter 5.

4.5 FITNESS ACCURACY OF THE MODEL WITH PROPOSED LMS

This is generally assed by the coefficient of determination, \( R^2 \). Using the \( R^2 \) – value as an indicator of goodness of fit, requires a threshold for the \( R \)-value that is good enough for the model to be best than the other. We have chosen a threshold of \( R=0.98 \). Since the residuals \( U_i \) are independent and identically distributed normal variants with mean 0 and variance \( (\hat{\sigma}^2) \) and if \( \hat{\theta} \) is the final estimate of \( \theta \), then

\[
R^2 = 1 - \left( \frac{\sum_{i=1}^{n} (Y_i - \bar{Y})^2}{\sum_{i=1}^{n} (Y_i - \sum_{i=1}^{n} f_i(\hat{\theta}))} \right) \tag{4.24}
\]

\[
\hat{\sigma}^2 = \frac{1}{n} \sum_{i=1}^{n} (Y_i - f_i(\hat{\theta}))^2 \tag{4.25}
\]

Using vector notation let,
\[ \theta' = [\theta_1, \theta_2, \theta_3] = [A, B, C]. \]

\[ Y' = (Y_1, Y_2, ..., Y_n). \]

\[ f'(\theta) = (f_1(\theta_1), f_2(\theta_2), ..., f_n(\theta_n)). \]

The performance analysis of the proposed model is measured by the four common criteria; they are SSE as the sum of squared errors, R-square, Adjust R-square and RMSE for the model comparison of goodness of fit as follows;

Sum of square of Error (SSE): This statistic measures the deviation of the responses from the values of responses. A value closer to 0 indicates a better estimation. It is calculated as:

\[
\text{SSE} = \sum_{i=1}^{n} [y_i - m(t_i)]^2
\]  

(4.26)

Where \( y_i \) is the total predicted reliability of the project and \( m(t_i) \) is actual reliability of the project. Mean square of fitting error is calculated as;

\[
\text{MSE} = \frac{\sum_{i=1}^{n} [y_i - n \times t_i]^2}{n}
\]  

(4.27)

The MSE measures the distance of a model estimate from the actual data with the consideration of the number of observations in the model. RMSE is defined as the root of mean squared error and for a computed value closer to 0 it indicates a better approximation and estimation. The R-square measures (using 4.24 and 4.25) how successful the model is in explaining the variation of the data, which may be defined as the square of the correlation between the response values and the predicted response values. R-square can take on any value between 0 and 1, with the value closer to 1 indicating a better estimation.
of the model. Adjusted R-square adjusts it based on the residual degrees of freedom.

The same experimented data are used in order to analyze the goodness of fit of the existing and the proposed approach and the comparison has been made with graphical output in chapter 5.

4.6 SUMMARY

The next step after selecting a model is the parameter estimation. A detailed discussion of parameter prediction and parameter estimation is given. Maximum likelihood estimation and least square estimation methods are described. A method is proposed to estimate the parameter of the Gompertz model by considering the number of test cases in addition to the variables number of defects and time interval. The step by step procedure of the proposed method is described. The analyze of fitness accuracy of the model with the proposed approach is described. The experimental results of the proposed method are given in chapter 5.