Discrete Time Software Reliability Modeling with Periodic Debugging Schedule

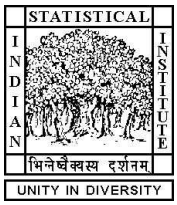
Technical Report No. ASU/2015/1
Dated: 24^{th} March, 2015

Sudipta Das
SQC and OR Unit, Indian Statistical Institute

Anup Dewanji
Applied Statistics Unit, Indian Statistical Institute

Debasis Sengupta
Applied Statistics Unit, Indian Statistical Institute

Applied Statistics Unit
Indian Statistical Institute
Kolkata 700 108



Discete Time Software Reliability Modeling with Periodic Debugging Schedule

SUMMARY. In this article, we discuss a discrete time model of software reliability in which the software is tested by test cases and detected errors are debugged periodically. We propose likelihood based inference of various parameters, including the initial number of errors and software reliability, under the assumption that all errors are equally likely to be detected at the pre-specified discrete times. We also derive some asymptotic properties of the estimated model parameters. Use of the proposed method is illustrated through the analysis of a dataset obtained from the testing of a flight control software.

KEY WORDS: Discrete time scale, Periodic debugging, Profile likelihood, Covariance matrix, Coverage probability.

1 Introduction

In the era of modern technologies, we encounter many types of systems which are supposed to provide service specified by a user over a well-defined episode of usage. For software underlying such systems, e.g transactions in an ATM machine, smart card punching system, etc., successful completion of service in a given specification of usage is a matter of greater importance than the exact epoch of a particular failure of the software. The software used in this kind of system are not used continuously, rather they are used intermittently for different sets of executions. Hence, the nature of its usage is truly discrete. A discrete time formulation is suitable for the specification of reliability of such a software, and also for collecting data to ascertain that reliability. Sometimes, even when a software is tested continuously, discretization of time occurs due to logistical issues of record-keeping. Discretely generated software fault data may also be available in situations, where the testing is done for the sake of improving the software, and assessing its reliability is not necessarily the primary objective. The well-known software models developed by Jelinski and Moranda [5], Goel and Okumoto [3], Musa et al. [7], etc., are not suitable for specifying reliability of such software, since these methods require failure data to be recorded continuously over time in some sense, e.g., calendar or execution time or staff hours. One needs a different kind of modeling by considering each run of software as one discrete unit of time and develop a model based on this discrete time scale, when inputs are selected from a suitable input space.

Different kinds of discrete time software testing models have been developed by Yamada and Osaki [10], Inoue and Yamada [4] and Kapur et al. [6], etc., among some others. However, these models assume that the errors are removed as and when they are detected. This assumption does not hold in many practical situations, when there are some pre-scheduled time points for debugging of errors, detected between two successive debugging times, and the software testing goes on till the last debugging time. Specifically, the errors are not removed at the epoch of detection; only a record of these detected

errors is maintained. These are removed, with certainty, at the subsequently scheduled debugging time (see Worwa [9]; Yang and Chao [11]). This type of debugging scheme is termed as 'periodic debugging schedule' in Dewanji et al. [2], where the authors consider a discrete time model suitable for periodic debugging schedule. Their work had been motivated by a particular dataset in which each error needs to be tagged to a particular module of the software. This model is not applicable to situations when module level information is not available, or when the error concerns integration of different modules.

In our work, we consider analysis of data arising from discrete time periodic debugging scheme, without requiring any modular structure of the software. We also estimate the initial or remaining number of errors in the software, along with the parameters associated with the failure probability. Here, we define reliability of the software as the probability of no failure in a complete run carried out with a randomly selected input. This definition is as per as the system we have discussed, where each test run is a simulation of the full execution of the software unit during an actual run of the software system. More generally, one may define reliability as the probability of no failure of a software in a fixed number of independent runs, which can be derived from the probability of no failure in a single run.

In Section 2, we consider the discrete time software reliability model for the periodic debugging schedule and the ensuing likelihood. In Section 3, we provide a computational method for estimating the number of errors. We discuss some asymptotic properties of this estimator in Section 4. In Section 5, we report results of a simulation study to investigate the properties of the estimates developed in Section 3 and studied theoretically in Section 4. We illustrate our estimation method through the analysis of a real life data set in Section 6.

2 Modeling and Likelihood

We assume that there are initially ν (finite but unknown) errors in the software. If a tester runs a test case, the software may or may not fail. If it fails, then the tester reports one or more errors along with their identities; otherwise, the tester reports no error. Suppose the probability of detection of a particular error in a test run is p. Errors are not removed/debugged at the end of each test run. Rather there are k prefixed instants when debugging takes place. Suppose that the number n_i of test runs executed during the i^{th} period (i.e., between the $(i-1)^{th}$ and i^{th} debugging instant), for $i = 1 \dots k$, are prefixed. The software testing terminates after the k^{th} period, that is after all the $n(= n_1 + \cdots + n_k)$ test runs are completed.

Let $m_{ij}(\geq 0)$ denote the number of errors (≥ 0) in the j^{th} run of the i^{th} period, for $j = 1 \dots n_i$, $i = 1 \dots k$. Since errors are not removed at the end of each test run, it is possible that the same error appears in different test runs within a period of testing. We denote the number of distinct errors appeared for the first time in the j^{th} run of the i^{th} period as m_{ij}^d, where $0 \leq m_{ij}^d \leq m_{ij}$. The total number of errors in the i^{th} period is $m_i = \Sigma_{j=1}^{n_i} m_{ij}$ and the total number of distinct errors in the i^{th} period is $m_i^d = \Sigma_{j=1}^{n_i} m_{ij}^d$.

Since errors are repeatable in a period, $m_i \geq m_i^d$, for $i = 1 \ldots k$. All the m_i^d errors are removed or debugged at the end of the i^{th} period. We assume debugging is perfect. Hence, the errors appeared in the i^{th} period do not appear in the $(i+1)^{st}$ period onwards.

Let M_i be the cumulative number of distinct errors removed on or before the i^{th} debugging instant, for $i = 1, \cdots, k$. Note that $M_i = \sum_{l=1}^{i} m_l^d$ and M_k is the number of detected errors by the last debugging instant, after the n runs. Since the detection of errors are assumed independent with probability of detection in a test run remaining the same, the relevant observation consists of $\{(m_{ij}, m_{ij}^d),\ j = 1, \ldots, n_i \text{ and } i = 1, \ldots, k\ \}$, as the identity of the detected errors do not contribute to the likelihood. Let us write $H_{ij} = \{(m_{uv}, m_{uv}^d),\ v = 1, \ldots, n_u, u = 1, \ldots i-1 \text{ and } v = 1, \ldots, j, u = i\}$ to represent the history of testing till the j^{th} test run of the i^{th} period. Note that H_{10} is empty and H_{i0}, for $i = 2 \ldots k$, is same as $H_{i-1,n_{i-1}}$.

Since no debugging takes place in the first n_1 test runs, the number m_{1j} of errors found in the j^{th} test run of the first period, given $H_{1,j-1}$, follows a $Binomial(\nu, p)$ distribution. The number of distinct errors found in the first $(j-1)$ test runs of the first period is $\sum_{l=1}^{j-1} m_{1l}^d$, which is zero for $j = 1$. The number m_{1j}^d of distinct errors found in the j^{th} test run of the first period, given $H_{1,j-1}$ and m_{1j}, follows a Hypergeometric distribution given by

$$P(m_{1j}^d | H_{1,j-1}, m_{1j}) = \frac{\binom{\sum_{l=1}^{j-1} m_{1l}^d}{m_{1j} - m_{1j}^d} \binom{\nu - \sum_{l=1}^{j-1} m_{1l}^d}{m_{1j}^d}}{\binom{\nu}{m_{1j}}},$$

for $j = 2, \ldots, n_1$, and $P(m_{11}^d | H_{10}, m_{11}) = 1$, for $m_{11} = m_{11}^d$. In general, for $i = 2, \ldots, k$, the conditional distribution of m_{ij}, given $H_{i,j-1}$, is $Binomial(\nu - M_{i-1}, p)$, for $j = 1, \ldots, n_i$. Also, for $i = 2, \ldots, k$, the conditional distribution of m_{ij}^d, given $H_{i,j-1}$ and m_{ij}, is a Hypergeometric given by

$$P(m_{ij}^d | H_{i,j-1}, m_{ij}) = \frac{\binom{\sum_{l=1}^{j-1} m_{il}^d}{m_{ij} - m_{ij}^d} \binom{\nu - M_{i-1} - \sum_{l=1}^{j-1} m_{il}^d}{m_{ij}^d}}{\binom{\nu - M_{i-1}}{m_{ij}}},$$

for $j = 2, \ldots, n_i$, and $P(m_{i1}^d | H_{i0}, m_{i1}) = 1$, since $m_{i1} = m_{i1}^d$.

Therefore, the likelihood contribution for the observation (m_{ij}, m_{ij}^d) in the j^{th} test run of the i^{th} period, given $H_{i,j-1}$, is

$$\begin{aligned} L_{ij}[m_{ij}, m_{ij}^d | H_{i,j-1}] &= \binom{\nu - M_{i-1}}{m_{ij}} p^{m_{ij}} (1-p)^{\nu - M_{i-1} - m_{ij}} \frac{\binom{\sum_{l=1}^{j-1} m_{il}^d}{m_{ij} - m_{ij}^d} \binom{\nu - M_{i-1} - \sum_{l=1}^{j-1} m_{il}^d}{m_{ij}^d}}{\binom{\nu - M_{i-1}}{m_{ij}}} \\ &= p^{m_{ij}} (1-p)^{\nu - M_{i-1} - m_{ij}} \binom{\sum_{l=0}^{j-1} m_{il}^d}{m_{ij} - m_{ij}^d} \binom{\nu - M_{i-1} - \sum_{l=0}^{j-1} m_{il}^d}{m_{ij}^d} \end{aligned}$$

$$\propto \quad p^{m_{ij}}(1-p)^{\nu-M_{i-1}-m_{ij}} \frac{(\nu - M_{i-1} - \sum_{l=0}^{j-1} m_{il}^d)!}{(\nu - M_{i-1} - \sum_{l=0}^{j} m_{il}^d)!},$$

for $j = 1, \ldots, n_i$, $i = 1 \ldots k$, with $M_0 = 0$. Hence, taking product of these successive conditional likelihood contributions, we have the total likelihood as

$$\begin{aligned} L(\nu, \lambda) &= \prod_{i=1}^{k} \prod_{j=1}^{n_i} L_{ij}[m_{ij}, m_{ij}^d | H_{i,j-1}] \\ &\propto \prod_{i=1}^{k} p^{m_i}(1-p)^{\nu n_i - M_{i-1} n_i - m_i} \frac{(\nu - M_{i-1})!}{(\nu - M_i)!} \\ &= p^m (1-p)^{\nu n - B - m} \frac{\nu!}{(\nu - M_k)!}, \end{aligned} \tag{1}$$

where $m = \sum_{i=1}^{k} m_i, n = \sum_{i=1}^{k} n_i$ and $B = \sum_{i=1}^{k} M_{i-1} n_i$.

3 Maximum Likelihood Estimation

We consider ν as the parameter of primary interest and present a profile likelihood method to obtain the MLE of ν. However, p is also estimated as a by-product of this method. Note that, when $M_k = 0$, the likelihood is $(1-p)^{\nu n}$, which gives the maximum likelihood estimate of ν as zero for any p. Therefore, we consider the case $M_k \geq 1$. The likelihood (1) gives the maximum likelihood estimate of p, for a fixed ν, as

$$\hat{p}(\nu) = \frac{m}{\nu n - B}. \tag{2}$$

Substituting $\hat{p}(\nu)$ in (1), the profile likelihood of ν is obtained as

$$L(\nu) \propto \frac{(\nu n - B - m)^{(\nu n - B - m)}}{(\nu n - B)^{(\nu n - B)}} \frac{\nu!}{(\nu - M_k)!}.$$

Now, consider

$$\frac{L(\nu+1)}{L(\nu)} = \left(\frac{\nu+1}{\nu+1-M_k}\right) \frac{(\nu n + n - B - m)^{(\nu n + n - B - m)}}{(\nu n + n - B)^{(\nu n + n - B)}} \frac{(\nu n - B)^{(\nu n - B)}}{(\nu n - B - m)^{(\nu n - B - m)}}, \text{ for } \nu \geq M_k.$$

Note that, if $\frac{L(\nu+1)}{L(\nu)} \leq 1$ for all ν, then the MLE of ν is $\hat{\nu} = M_k$. While estimating ν, we may encounter $\hat{\nu} = \infty$, for example, when $\frac{L(\nu+1)}{L(\nu)} \geq 1$ for all ν. The following proposition states a sufficient condition for finite $\hat{\nu}$.

Proposition 1:- The MLE $\hat{\nu}$ is finite if $M_k < m$ or $mn < m + n + 2B$.

Proof:- Write

$$\begin{aligned} g(\nu) &= \log\left(\frac{L(\nu)}{L(\nu+1)}\right) \\ &= \log(\nu+1-M_k) - \log(\nu+1) \\ &\quad +(\nu n - B - m)\log(\nu n - B - m) - (\nu n + n - B - m)\log(\nu n + n - B - m) \\ &\quad +(\nu n + n - B)\log(\nu n + n - B) - (\nu n - B)\log(\nu n - B) \end{aligned}$$

Treating ν as a continuous variable, we have the derivative as

$$\begin{aligned} g'(\nu) &= \frac{1}{\nu+1-M_k} - \frac{1}{\nu+1} + n\log(\nu n - B - m) - n\log(\nu n + n - B - m) \\ &\quad +n\log(\nu n + n - B) - n\log(\nu n - B) \\ &= \frac{1}{\nu+1-M_k} - \frac{1}{\nu+1} + n\left(\log(\nu n - B - m) - \log(\nu n + n - B - m) + \log(\nu n + n - B) - \log(\nu n - B)\right) \\ &= \frac{M_k}{(\nu+1)(\nu+1-M_k)} + n\log\left(\frac{(\nu n - B - m)(\nu n + n - B)}{(\nu n + n - B - m)(\nu n - B)}\right) \\ &= \frac{M_k}{(\nu+1)(\nu+1-M_k)} + n\log\left(\frac{\nu^2 n^2 + \nu n(n - m - 2B) + B^2 + Bm - Bn - mn}{\nu^2 n^2 + \nu n(n - m - 2B) + B^2 + Bm - Bn}\right) \\ &= \frac{M_k}{(\nu+1)(\nu+1-M_k)} + n\log\left(1 - \frac{mn}{(\nu n + n - B - m)(\nu n - B)}\right) \\ &< \frac{M_k}{(\nu+1)(\nu+1-M_k)} - \frac{mn^2}{(\nu n + n - B - m)(\nu n - B)} \\ &= \frac{\nu^2 n^2(M_k - m) + \nu n\left((n - 2B - m)M_k - mn(2 - M_k)\right) + \left(B(B + m - n)M_k - (1 - M_k)mn^2\right)}{(\nu+1)(\nu+1-M_k)(\nu n + n - B - m)(\nu n - B)} \end{aligned}$$

For ν large enough, the sign of $g'(\nu)$ will be controlled by the term

$$\frac{\nu^2 n^2(M_k - m)}{(\nu+1)(\nu+1-M_k)(\nu n + n - B - m)(\nu n - B)}.$$

When $m > M_k$, $g'(\nu)$ is negative, as $m \leq \nu n - B$. Therefore, for sufficiently large ν, $g(\nu)$ is decreasing. Hence, $\frac{L(\nu)}{L(\nu+1)}$ is also decreasing for large ν.

However, when $m = M_k$, the sign of $g'(\nu)$ will be controlled by the term

$$\frac{\nu n m(mn - m - n - 2B)}{(\nu+1)(\nu+1-M_k)(\nu n + n - B - m)(\nu n - B)}.$$

If $mn < m + n + 2B$, then also $\frac{L(\nu)}{L(\nu+1)}$ is also decreasing for large ν.

One can also notice that $\frac{L(\nu)}{L(\nu+1)} \to 1$ as $\nu \to \infty$. Hence, $\frac{L(\nu)}{L(\nu+1)}$ goes to 1 from above. Therefore, there exists a finite ν for which $\frac{L(\nu)}{L(\nu+1)} > 1$. It implies $\hat{\nu} < \infty$. Hence, the proof is completed. Under the conditions of Proposition 1, the MLE of ν can be obtained by $\hat{\nu} = \min\{\nu \geq M_k : L(\nu+1) \geq L(\nu)\}$. The MLE of p can be obtained as $\hat{p} = \hat{p}(\hat{\nu})$, using (2).

4 Asymptotic Results

In order to derive the asymptotic properties of the estimates, $\hat{p}$ and $\hat{\nu}$ (say) of p and ν, respectively, obtained by the method of Section 3, let the ν errors be labeled as $1, \cdots, \nu$ and let X_j denote the (unknown) observation from the jth error. If the jth error is not detected, let us write $X_j = 0$; otherwise, X_j consists of the identity l_j (say) of its detection and debugging instant (that is, this error is detected between $(l_j - 1)^{th}$ and l_j^{th} debugging instant and removed at the l_jth debugging instant with 0^{th} debugging instant being time zero) and the number m_j^* (say) of times it appears between $(l_j - 1)^{th}$ and l_j^{th} debugging. The X_j's are independent and identically distributed unobservable variables with the probability distribution given by

$$p_X(x_j; p) = \begin{cases} (1-p)^n, & \text{if } x = 0 \\ \binom{n_{l_j}}{m_j^*} p^{m_j^*} (1-p)^{\sum_{j'=1}^{l_j-1} n_{j'} + (n_{l_j} - m_j^*)}, & \text{otherwise,} \end{cases}$$

The joint distribution of $(X_1, \cdots, X_\nu)$ is, therefore, $L^*(\nu, \lambda) = \prod_{j=1}^{\nu} p_X(x_j; \lambda)$. Note that the observed data is a function of the unobserved X_j's. Hence, the observed likelihood (1) can be obtained from the joint distribution of the X_j's and is given by

$$\frac{\nu!}{(\nu - M_k)!} \prod_{j=1}^{\nu} p_X(x_j; p).$$

Following Dewanji et al. ([1]), define

$$\begin{aligned} V_{1j} &= \frac{d \ \log \ p_X(x_j; \lambda)}{d \ p} \\ &= \begin{cases} -\frac{n}{1-p}, & \text{if } x_j = 0 \\ \frac{m_j^*}{p} + \frac{m_j^*}{1-p} - \frac{N_{l_j}}{1-p}, & \text{otherwise,} \end{cases} \end{aligned}$$

where $N_{l_j} = \sum_{j'=1}^{l_j} n_{j'}$, and

$$V_{2j} = \begin{cases} 1, & \text{if } x_j \neq 0 \\ 1 - (1-p)^{-n}, & \text{otherwise,} \end{cases}$$

for $j = 1, \cdots, \nu$. Now,

$$\begin{aligned} E[V_{1j}] &= -\frac{n}{1-p}(1-p)^n + \sum_{l_j=1}^{k} \sum_{m_j^*=1}^{n_{l_j}} \left(\frac{m_j^*}{p(1-p)} - \frac{N_{l_j}}{1-p} \right) \frac{n_{l_j}!}{m_j^*!(n_{l_j} - m_j^*)!} p^{m_j^*} (1-p)^{N_{l_j} - m_j^*} \\ &= -\frac{n}{1-p}(1-p)^n \end{aligned}$$

$$
\begin{aligned}
&+\sum_{l_j=1}^{k}\frac{(1-p)^{N_{l_j-1}}}{1-p}\sum_{m_j^*=1}^{n_{l_j}}\left[n_{l_j}\binom{n_{l_j}-1}{m_j^*-1}p^{m_j^*-1}(1-p)^{n_{l_j}-m_j^*}-N_{l_j}\binom{n_{l_j}}{m_j^*}p^{m_j^*}(1-p)^{n_{l_j}-m_j^*}\right]\\
&= -\frac{n}{1-p}(1-p)^n+\sum_{l_j=1}^{k}\frac{(1-p)^{N_{l_j-1}}}{1-p}\left[n_{l_j}-N_{l_j}\left(1-(1-p)^{n_{l_j}}\right)\right]\\
&= -\frac{n}{1-p}(1-p)^n+\sum_{l_j=1}^{k}\frac{(1-p)^{N_{l_j-1}}}{1-p}\left[N_{l_j}(1-p)^{n_{l_j}}-N_{l_j-1}\right]\\
&= -\frac{N_k}{1-p}(1-p)^{N_k}+\frac{1}{1-p}\sum_{l_j=1}^{k}\left(N_{l_j}(1-p)^{N_{l_j}}-N_{l_j-1}(1-p)^{N_{l_j-1}}\right)\\
&= 0,
\end{aligned}
$$

since $n=N_k$, and

$$
\begin{aligned}
Var[V_{1j}] &= E[-\frac{\partial^2}{\partial\lambda^2}\ \log\ p_X(x_j;\lambda)]\\
&= \frac{n}{(1-p)^2}(1-p)^n\\
&\quad+\sum_{l_j=1}^{k}\sum_{m_j^*=1}^{n_{l_j}}\left(\frac{m_j^*}{p^2}-\frac{m_j^*}{(1-p)^2}+\frac{N_{l_j}}{(1-p)^2}\right)\frac{n_{l_j}!}{m_j^*!(n_{l_j}-m_j^*)!}p^{m_j^*}(1-p)^{N_{l_j}-m_j^*}\\
&= \frac{n}{(1-p)^2}(1-p)^n\\
&+\ \sum_{l_j=1}^{k}\frac{(1-p)^{N_{l_j-1}}}{(1-p)^2}\sum_{m_j^*=1}^{n_{l_j}}\left[\frac{1-2p}{p}n_{l_j}\binom{n_{l_j}-1}{m_j^*-1}p^{m_j^*-1}(1-p)^{n_{l_j}-m_j^*}+N_{l_j}\binom{n_{l_j}}{m_j^*}p^{m_j^*}(1-p)^{n_{l_j}-m_j^*}\right]\\
&= \frac{n}{(1-p)^2}(1-p)^n+\sum_{l_j=1}^{k}\frac{(1-p)^{N_{l_j-1}}}{(1-p)^2}\left[\frac{1-2p}{p}n_{l_j}+N_{l_j}\left(1-(1-p)^{n_{l_j}}\right)\right]\\
&= \frac{N_k}{(1-p)^2}(1-p)^{N_k}+\sum_{l_j=1}^{k}\left(n_{l_j}\frac{1-p}{p}+N_{l_j-1}-N_{l_j}(1-p)^{n_{l_j}}\right)\frac{(1-p)^{N_{l_j-1}}}{(1-p)^2}\\
&= \sum_{l_j=1}^{k}\frac{n_{l_j}(1-p)^{N_{l_j-1}}}{p(1-p)}+\sum_{l_j=1}^{k}\frac{\left(N_{l_j-1}(1-p)^{N_{l_j-1}}-N_{l_j}(1-p)^{N_{l_j}}\right)}{(1-p)^2}+\frac{N_k(1-p)^{N_k}}{(1-p)^2}\\
&= \sum_{l_j=1}^{k}\frac{n_{l_j}(1-p)^{N_{l_j-1}}}{p(1-p)}\\
&= I(p),\ (\text{say}).
\end{aligned}
$$

Similarly,

$$
\begin{aligned}
E[V_{2j}] &= -(1-(1-p)^{-n})(1-p)^n+(1-(1-p)^n)\\
&= 0
\end{aligned}
$$

and

$$\begin{aligned} Var[V_{2j}] &= (1-(1-p)^{-n})^2(1-p)^n + (1-(1-p)^n) \\ &= (1-p)^{-n} - 1 \end{aligned}$$

The covariance between V_{1j} and V_{2j} is

$$\begin{aligned} Cov[V_{1j}, V_{2j}] &= \frac{n}{1-p}\left((1-p)^{-n}-1\right)(1-p)^n \\ &\quad + \sum_{l_j=1}^{k}\sum_{m_j^*=1}^{n_{l_j}}\left(\frac{m_j^*}{p(1-p)} - \frac{N_{l_j}}{1-p}\right)\frac{n_{l_j}!}{m_j^*!(n_{l_j}-m_j^*)!}p^{m_j^*}(1-p)^{N_{l_j}-m_j^*} \\ &= \frac{n(1-(1-p)^n)}{(1-p)} \\ &\quad + \sum_{l_j=1}^{k}\frac{(1-p)^{N_{l_j-1}}}{1-p}\sum_{m_j^*=1}^{n_{l_j}}\left[n_{l_j}\binom{n_{l_j}-1}{m_j^*-1}p^{m_j^*-1}(1-p)^{n_{l_j}-m_j^*} - N_{l_j}\binom{n_{l_j}}{m_j^*}p^{m_j^*}(1-p)^{n_{l_j}-m_j^*}\right] \\ &= \frac{n(1-(1-p)^n)}{(1-p)} + \sum_{l_j=1}^{k}\frac{(1-p)^{N_{l_j-1}}}{1-p}\left[n_{l_j} - N_{l_j}\left(1-(1-p)^{n_{l_j}}\right)\right] \\ &= \frac{n(1-(1-p)^n)}{(1-p)} + \frac{1}{1-p}\sum_{l_j=1}^{k}\left(N_{l_j}(1-p)^{N_{l_j}} - N_{l_j-1}(1-p)^{N_{l_j-1}}\right) \\ &= \frac{n(1-(1-p)^n)}{(1-p)} + \frac{N_k(1-p)^{N_k}}{1-p} \\ &= \frac{n}{1-p} \end{aligned}$$

We write

$$\begin{aligned} u_{1\nu} &= \nu^{-1/2}\sum_{j=1}^{\nu}V_{1j} \\ &= \nu^{-1/2}\left(-\frac{n(\nu-M_k)}{1-p} + \sum_{j=1}^{M_k}(\frac{m_j*}{p(1-p)} - \frac{N_{l_j}}{1-p})\right) \\ &= \nu^{-1/2}\left(\frac{m}{p} - \nu n + (n_1+n_2+\ldots+n_k)M_k - (N_{l_1}+N_{l_2}+\ldots+N_{l_{M_k}})\right)\times\frac{1}{1-p} \\ &= \nu^{-1/2}\left(\frac{m}{\lambda} - \nu n + \sum_{j=1}^{k}M_{j-1}n_j\right)\times\frac{1}{1-p} \end{aligned}$$

and

$$u_{2\nu} = \nu^{-1/2}\sum_{j=1}^{\nu}V_{2j}$$

$$
\begin{aligned}
&= \nu^{-1/2}\left((\nu - M_k)\left(1-(1-p)^{-n}\right) + M_k\right) \\
&= \nu^{-1/2}\left(\frac{M_k - \nu + \nu(1-p)^n}{(1-p)^n}\right),
\end{aligned}
$$

so that $1 - \nu^{-1/2}u_{2\nu} = \frac{\nu - M_k}{\nu(1-p)^n}$. Thus, by the central limit theorem,

$$
\begin{pmatrix} u_{1\nu} \\ u_{2\nu} \end{pmatrix} \stackrel{d}{=} \nu^{-1/2}\sum_{j=1}^{\nu}\begin{pmatrix} V_{1j} \\ V_{2j} \end{pmatrix} \stackrel{L}{\longrightarrow} N(0, \Sigma^{-1}) \text{ as } \nu \to \infty,
$$

where

$$
\Sigma = \begin{pmatrix} I(p) & \frac{n}{1-p} \\ \frac{n}{1-p} & (1-p)^{-n} - 1 \end{pmatrix}^{-1}.
$$

Now, from (1), we have

$$
\log L(\nu, \lambda) = m\log p + (\nu n - B - m)\log(1-p) + \sum_{j=0}^{M_k - 1}\log(\nu - j).
$$

For bounded (a, b), following the technique of Dewanji et al. ([1]), we consider

$$
\begin{aligned}
&\log L(\nu + \nu^{1/2}a, p + \nu^{-1/2}b) - \log L(\nu, p) \\
= \;& m\log\left(1 + \frac{\nu^{-1/2}b}{p}\right) + (\nu n - B - m)\log\left(1 - \frac{\nu^{-1/2}b}{1-p}\right) + \nu^{1/2}an\log\left(1 - p - \nu^{-1/2}b\right) \\
& + \sum_{j=0}^{M_k-1}\log\left(1 + \frac{\nu^{1/2}a}{\nu - j}\right) \\
= \;& m\log\left(1 + \frac{\nu^{-1/2}b}{p}\right) + (\nu n - B - m)\log\left(1 - \frac{\nu^{-1/2}b}{1-p}\right) + \nu^{1/2}an\log\left(1 - p - \nu^{-1/2}b\right) \\
& + a\nu^{1/2}\sum_{j=0}^{M_k-1}\frac{1}{\nu - j} - \frac{a^2\nu}{2}\sum_{j=0}^{M_k-1}\frac{1}{(\nu-j)^2} + \frac{a^3\nu^{3/2}}{3}\sum_{j=0}^{M_k-1}\frac{1}{(\nu-j)^3} + \cdots \\
= \;& m\left(\frac{\nu^{-1/2}b}{p} - \frac{\nu^{-1}b^2}{2p^2} + \frac{\nu^{-3/2}b^3}{3p^3} - \cdots\right) + (\nu n - B - m)\left(\frac{\nu^{-1/2}b}{1-p} - \frac{\nu^{-1}b^2}{2(1-p)^2} + \frac{\nu^{-3/2}b^3}{3(1-p)^3} - \cdots\right) \\
& - \nu^{1/2}an\left((p + \nu^{-1/2}b) + \frac{(p+\nu^{-1/2}b)^2}{2} + \frac{(p+\nu^{-1/2}b)^3}{3} + \cdots\right) \\
& + a\nu^{1/2}\left(\log\nu - \log(\nu - M_k) + O(\nu^{-1})\right) - \frac{a^2}{2}\left(\frac{M_k}{\nu - M_k} + O(\nu^{-1})\right) + \frac{a^3}{3}O(\nu^{-1/2}) + \cdots \\
= \;& \nu^{-1/2}b\left(\frac{m}{p} - \frac{m}{1-p} + \frac{\nu n - B}{1-p}\right) - \frac{b^2}{2\nu}\left(\frac{m}{p^2} - \frac{m}{(1-p)^2} + \frac{\nu n - B}{(1-p)^2}\right) \\
& - \nu^{1/2}an\left(p + \frac{p^2}{2} + \frac{p^3}{3} + \cdots + \nu^{-1/2}b(1 + p + p^2 + \cdots) + O(\nu^{-1})\right)
\end{aligned}
$$

$$
\begin{aligned}
&+a\nu^{1/2}\left[-\log(1-\nu^{-1/2}u_{2\nu})(1-p)^n+O(\nu^{-1})\right]-\frac{a^2}{2}\left(\frac{1}{(1-\nu^{-1/2}u_{2\nu})(1-p)^n}-1\right)+O(\nu^{-1/2})\\
=\;& bu_{1\nu}-\frac{1}{2}b^2I(p)+\nu^{1/2}an\log(1-p)-\frac{abn}{1-p}\\
&-a\nu^{1/2}\log(1-\nu^{-1/2}u_{2\nu})-\nu^{1/2}an\log(1-p)-\frac{a^2}{2}\frac{1-(1-\nu^{-1/2}u_{2\nu})(1-p)^n}{(1-\nu^{-1/2}u_{2\nu})(1-p)^n}+O(\nu^{-1/2})\\
=\;& bu_{1\nu}-\frac{1}{2}b^2I(p)-\frac{abn}{1-p}+au_{2\nu}-\frac{a^2}{2}\frac{\frac{1-(1-p)^n}{(1-p)^n}+\nu^{-1/2}u_{2\nu}}{(1-\nu^{-1/2}u_{2\nu})}+O(\nu^{-1/2})\\
=\;& bu_{1\nu}-\frac{1}{2}b^2I(p)-\frac{abn}{1-p}+au_{2\nu}-\frac{a^2}{2}\frac{1-(1-p)^n}{(1-p)^n}+o_p(1), \qquad (3)
\end{aligned}
$$

since

$$
\begin{aligned}
I(p) &= E\left[-\nu^{-1}\frac{\partial^2}{\partial p^2}\log L(\nu,p)\right]\\
&= E\left[-\nu^{-1}\sum_{j=1}^{\nu}\frac{\partial^2}{\partial p^2}\log p_x(x_j,p)\right]\\
&= E\left[\frac{m}{p^2}-\frac{m}{(1-p)^2}+\frac{\nu n-B}{(1-p)^2}\right].
\end{aligned}
$$

In the above derivations, we have used the following identities:

$$
\begin{aligned}
\Sigma_{j=0}^{r-1}(\nu-j)^{-1} &= \log\nu-\log(\nu-r)+O(\nu^{-1}),\\
\nu\Sigma_{j=0}^{r-1}(\nu-j)^{-2} &= \frac{r}{\nu-r}+O(\nu^{-1}),\\
\nu^{3/2}\Sigma_{j=0}^{r-1}(\nu-j)^{-3} &= O(\nu^{-1/2}).
\end{aligned}
$$

Then, using the argument of Sen and Singer ([8], p207), as in Dewanji et al. ([1]), we differentiate (3) to obtain

$$
\begin{aligned}
\frac{\partial}{\partial a}\left(\log L(\nu+\nu^{1/2}a,p+\nu^{-1/2}b)-\log L(\nu,p)\right) &= -\frac{bn}{1-p}+u_{2\nu}-a((1-p)^{-n}-1)=0\\
\frac{\partial}{\partial b}\left(\log L(\nu+\nu^{1/2}a,p+\nu^{-1/2}b)-\log L(\nu,p)\right) &= u_{1\nu}-bI(p)-\frac{an}{1-p}=0.
\end{aligned}
$$

Hence, we obtain $(\hat{a_\nu},\hat{b_\nu})$ as the solution of

$$
\begin{aligned}
u_{1\nu} &= \hat{a}\frac{n}{1-p}+\hat{b}I(p)\\
u_{2\nu} &= \hat{a}\left((1-p)^{-n}-1\right)+\hat{b}\frac{n}{1-p}
\end{aligned}
$$

so that

$$
\begin{pmatrix}\hat{b}\\ \hat{a}\end{pmatrix}=\Sigma\begin{pmatrix}u_{1\nu}\\ u_{2\nu}\end{pmatrix},
$$

equivalently

$$\begin{pmatrix} \nu^{1/2}(\hat{p}-p) \\ \nu^{-1/2}(\hat{\nu}-\nu) \end{pmatrix} = \Sigma \begin{pmatrix} u_{1\nu} \\ u_{2\nu} \end{pmatrix}.$$

Now, $[u_{1\nu}, u_{2\nu}] \xrightarrow{L} N(0, \Sigma^{-1})$.

Result 1: As $\nu \to \infty$,

$$[\nu^{1/2}(\hat{p}-p), \nu^{-1/2}(\hat{\nu}-\nu)] \xrightarrow{L} N(0, \Sigma),$$

where

$$\Sigma = \begin{pmatrix} I(p) & \frac{n}{1-p} \\ \frac{n}{1-p} & (1-p)^{-n} - 1 \end{pmatrix}^{-1}$$

with $I(p) = E[-d^2 \ \log \ p_X(x_j; p)/d \ p^2]$. The covariance matrix Σ can be consistently estimated by using $\hat{p}$ in place of p and by estimating $I(p)$ by

$$\begin{aligned} I(\hat{p}) &= -\hat{\nu}^{-1}\partial^2 \log L(\hat{\nu}, \hat{p})/\partial p^2 \\ &= \hat{\nu}^{-1}\left[\frac{m}{\hat{p}^2} + \frac{\hat{\nu}n - B - m}{(1-\hat{p})^2}\right] \\ &= \frac{\hat{\nu}^{-1}}{(1-\hat{p})^2}\left[\frac{m(1-2\hat{p})}{\hat{p}^2} + \hat{\nu}n - B\right]. \end{aligned}$$

In particular, the variance of $\hat{\nu}$ can be consistently estimated by

$$\hat{\nu}\left[(1-\hat{p})^{-n} - 1 - n^2(1-\hat{p})^{-2}I^{-1}(\hat{p})\right]^{-1} \tag{4}$$

5 A Simulation Study

In order to assess the performance of the estimators of ν and p through simulation, we consider four values of ν, namely, $\nu = 100, 500, 1000$ and 5000. The total number of test cases is $n = 100$, while the number of test cases executed between successive debugging is kept identical with value 10, so that number of debugging k is equal to 10. We consider three choices of the parameter p as 0.005, 0.010 and 0.020, resulting in $4 \times 3 = 12$ different parameter configurations. For each one of the configurations, we simulate 5,000 datasets. For each dataset, excluding the cases when $M_k = m$ and $mn \geq m+n+2B$, we compute $\hat{\nu}$, $\hat{p}$ and the standard error of $\hat{\nu}$ using (4), which is denoted by $s(\hat{\nu})$.

Since ν is the parameter of primary interest, we report the simulation results regarding the performance of $\hat{\nu}$ in Table 1. In addition to the average of $\hat{\nu}$ and $s(\hat{\nu})$ over the 5,000 simulations, we also present the sample standard error of $\hat{\nu}$, denoted by $sse(\hat{\nu})$, defined as the sample standard deviation of 5,000 estimates of ν. The estimated coverage probability, denoted by CP, is computed as the proportion of times (out of 5,000 simulations) the asymptotic 95% confidence interval, obtained through the normal approximation of $\hat{\nu}$ (see Result 1) and using (4), contains the true ν. For the sake of

comparison, we also present relative bias and relative standard error defined as $(\hat{\nu}-\nu)/\nu$ and $s(\hat{\nu})/\nu$, respectively.

Note that the average standard error under $s(\hat{\nu})$ and the sample standard error under $sse(\hat{\nu})$ are similar specially for large ν and p, implying the convergence to the asymptotic variance of $\hat{\nu}$ in (4). As expected, the estimator $\hat{\nu}$ seems to perform better with respect to relative bias, relative standard error and CP with increasing ν and increasing p. The MLE is nearly unbiased in all cases. Also, the CP values are close to 0.95, specially for large ν, suggesting convergence to normality as given in Result 1. The better performance due to increasing p is possible because of larger number of failures (or, error detection) on the average leading to more informations.

Table 1: Simulation results on the MLE of ν with corresponding standard errors and estimated coverage probability for $n = 100$.

ν	p	$\hat{\nu}$	$(\hat{\nu}-\nu)/\nu$	$s(\hat{\nu})$	$sse(\hat{\nu})$	$s(\hat{\nu})/\nu$	CP
100	0.005	86.11	-0.1389	32.89	50.25	0.329	0.733
	0.010	101.51	0.0151	23.84	25.41	0.238	0.870
	0.020	99.72	-0.0028	7.19	6.84	0.072	0.906
500	0.005	517.14	0.0343	136.73	155.43	0.273	0.906
	0.010	505.43	0.0109	54.25	51.67	0.109	0.934
	0.020	499.65	-0.0007	15.13	15.00	0.030	0.937
1000	0.005	1032.56	0.0326	213.38	213.92	0.213	0.922
	0.010	1002.22	0.0022	72.68	70.41	0.073	0.934
	0.020	999.27	-0.0007	21.27	21.14	0.021	0.945
5000	0.005	5032.18	0.0064	449.87	435.46	0.090	0.941
	0.010	5001.74	0.0003	156.44	155.31	0.031	0.944
	0.020	5000.30	0.0001	48.23	47.33	0.010	0.946

To study the effect of the number and size of debugging intervals, we have also performed another simulation study with fixed $n = 100$, while the number of runs n_δ between successive debugging is varied as 10, 20, 50 and 100 with the corresponding number k of debugging being 10, 5, 2 and 1. The error probability p is also kept fixed at 0.01. The same simulation exercise as before is carried out with 5,000 repetitions and the results are presented in Table 2.

Table 2: Simulation results with varying number of debugging and $p = 0.01$ for $n = 100$.

ν	n_δ	$\hat{\nu}$	$(\hat{\nu}-\nu)/\nu$	$s(\hat{\nu})$	$s(\hat{\nu})/\nu$	CP
100	10	101.64	0.0164	25.45	0.2545	0.863
	20	103.29	0.0329	24.46	0.2446	0.888
	50	101.96	0.0196	16.61	0.1661	0.927
	100	100.35	0.0035	12.39	0.1239	0.935
500	10	504.84	0.0097	52.95	0.1059	0.941
	20	503.04	0.0061	44.81	0.0896	0.943
	50	501.43	0.0029	32.74	0.0655	0.946
	100	500.94	0.0019	26.02	0.0520	0.950
1000	10	1000.75	0.0007	70.29	0.0703	0.943
	20	1003.40	0.0034	59.80	0.0598	0.948
	50	1000.49	0.0005	45.29	0.0453	0.947
	100	1000.52	0.0005	37.14	0.0371	0.951
5000	10	5006.67	0.0013	158.56	0.0317	0.943
	20	4997.56	-0.0005	130.99	0.0262	0.953
	50	4998.45	-0.0003	106.85	0.0214	0.941
	100	4997.81	-0.0004	84.52	0.0169	0.946

The average standard error and the sample standard error found to be close, as before, and so only the $s(\hat{\nu})$'s are reported. The estimator $\hat{\nu}$ seems to perform better with respect to relative bias, relative standard error and CP with increasing ν, as before, and surprisingly with decreasing number k of debugging intervals. The latter is an interesting observation in the sense that better efficiency for smaller k seems to be counter-intuitive. In fact, with decreasing k, there are more replications of failure time due to an error resulting in more information and, hence, better efficiency. Therefore, a single debugging schedule at the end of testing seems to be the most efficient design. It can also be easily proved that the asymptotic variance of $\hat{\nu}$ for k debugging intervals, given by (See (4))

$$\nu \left[(1-p)^{-n} - 1 - \frac{n^2 p}{(1-p)\sum_{i=1}^{k} n_i (1-p)^{N_{i-1}}} \right]^{-1},$$

is minimum when $k = 1$. However, as remarked in Section 1, a schedule of more than one debugging intervals may be necessary due the market demand for software release.

6 An Example

For illustration, we consider a dataset obtained from testing of a flight control software analyzed by Dewanji et al. ([2]) using a different kind of reliability modeling without the parameter ν representing the initial number of errors in the software. The testing

was done in four phases consisting of 187, 227, 187 and 267 runs with 2, 1, 3 and 0 failures, respectively, without any repetition. Debugging was done at the end of each phase. Treating each simulation run as one unit of time for the sake of illustration, we have a periodic debugging schedule with $k = 4$ and $n_1 = 187$, $n_2 = 227$, $n_3 = 187$ and $n_4 = 267$, leading to $n = 868$. We also have $m = 6$, $M_1 = 2$, $M_2 = 3$, $M_3 = M_4 = 6$ and $B = 2617$.

Since we have $5208 = mn < m + n + 2B = 6108$, the profile likelihood method gives finite MLE of ν (See Proposition 1) as $\hat{\nu} = 9$ with standard error 6.43 (See (4)). We also obtain $\hat{p} = 0.0012$, with standard error 0.0013 using Result 1. Since ν seems to be small, the asymptotic results of Section 4, and hence the standard error in (4), may be in doubt. We, therefore, use a parametric bootstrap method to estimate the sampling distribution of $\hat{\nu}$. Based on 10,000 bootstrap samples using $\hat{p} = 0.0012$ and $\hat{\nu} = 9$, the symmetric 95% confidence interval for ν turns out to be [3,14].

The quantity of primary interest is, however, the reliability of the software for a single subsequent test run (say), which is given by $(1-p)^{\nu - M_k}$ and is estimated by $(1-\hat{p})^{\hat{\nu} - M_k}$. The estimate of this reliability is 0.9965. Application of delta method to obtain the standard error of this estimate may be questionable since one of the parameters, ν, takes discrete values and also as ν seems to be rather small. However, treating ν to be continuous and using delta method, corresponding standard error is 0.0032. The standard error is also obtained on the basis of 10,000 parametric bootstrap samples, as before, as 0.0017. Note that, in the analysis of Dewanji et al. ([2]) of the same data, the reliability for an additional unit of test run is estimated as 0.9979 with standard error 0.0026.

7 Concluding Remarks

Acknowledgments:- The authors are thankful to Shri Amitava Bandyopadhyay for many valuable comments. This work is partially sponsored by the project "Optimization and Reliability Modeling" funded by Indian Statistical Institute.

References

[1] Anup Dewanji, Tapan K. Nayak, and Pranab K. Sen. Estimating the number of components of a system of superimposed renewal processes. *Sankhyā: The Indian Journal of Statistics Series A*, pages 486–499, 1995.

[2] Anup Dewanji, Debasis Sengupta, and Ashis Kumar Chakraborty. A discrete time model for software reliability with application to a flight control software. *Applied Stochastic Models in Business and Industry*, 27:723–731, 2011.

[3] Amrit L. Goel and Kazuhira Okumoto. When to stop testing and start using software? In *Proceedings of the 1981 ACM workshop/symposium on Measurement and evaluation of software quality*, pages 131–138, 1981.

[4] Shinji Inoue and Shigeru Yamada. Generalized discrete software reliability modeling with effect of program size. *IEEE Transactions on Systems Man and Cybernetics, Part A: Systems and Humans*, 37(2):170–179, March 2007.

[5] Z. Jelinski and P. Moranda. Software reliability research. *Statistical Computer Performance Evaluation, Academic Press*, pages 465–484, 1972.

[6] P.K. Kapur, P. C. Jha, and V.B. Singh. *On the development of discrete reliability growth models in Handbook of Performability Engineering.* Springer, New York, 2008.

[7] John D. Musa, Anthony Iannino, and Kazuhira Okumoto. *Software Reliabilty.* McGraw-Hill, New York, 1987.

[8] P. K. Sen and J. M. Singer. *Large sample methods in statistics an introduction with application.* Chapman and Hall, London, 1993.

[9] K. Worwa. A discrete-time software reliability-growth model and its application for predicting the number of errors kencountered during program testing. *Control and Cybernetics*, 34(2):589–606, 2005.

[10] Shigeru Yamada and Shunji Osaki. Discrete software reliability growth models. *Applied Stochastic Models and Data Analysis*, 1(1):65–77, 1985.

[11] Mark C. K. Yang and Anne Chao. Reliability -estimation & stopping-rules for software testing, based on repeated appearances of bugs. *IEEE Transactions on Reliability*, 44(2):315–321, June 1995.