A Sequential Bayesian Generalization of the Jelinski Moranda Software Reliability Model

Washburn, Alan
A Sequential Bayesian Generalization of the Jelinski–Moranda Software Reliability Model

Alan Washburn

Operations Research Department, Naval Postgraduate School, Monterey, CA 93940, USA

Received 30 August 2004; revised 25 July 2005; accepted 14 August 2005
DOI 10.1002/nav.20148
Published online 9 March 2006 in Wiley InterScience (www.interscience.wiley.com).

Abstract: The Jelinski–Moranda model of software reliability is generalized by introducing a negative-binomial prior distribution for the number of faults remaining, together with a Gamma distribution for the rate at which each fault is exposed. This model is well suited to sequential use, where a sequence of reliability forecasts is made in the process of testing or using the software. We also investigate replacing the Gamma distribution with a worst-case assumption about failure rates (the worst-case failure rate in models such as this is not infinite, since faults with large failure rates are immediately discovered and removed).


Keywords: reliability; software; Bayes; sequential

1. INTRODUCTION

The point of view taken in this paper is that a given computer program (software) initially contains an unknown number of faults, each of which will eventually be discovered and instantly eliminated as the software is tested or used. As the faults are eliminated, the software becomes more reliable because there are fewer faults remaining to be discovered—we assume that no new faults are introduced in the process of fixing the old ones. The problem is to plan testing and forecast reliability in the face of uncertainties about the initial number of faults, as well as the chances of exposing the faults in testing. We take a Bayesian approach, postulating prior distributions for the important quantities and updating them based on evidence from testing.

We assume throughout that each fault has a random failure time (time at which it will be discovered) with an exponential distribution; that is, we assume that the failure rate (sometimes called “hazard rate” [21]) of each fault is constant in time. These failure rates are one of the primary objects of Bayesian analysis.

In reality, some faults are more serious than others, but all faults will be assumed equivalent here. The definitions of “fault” and “time” must be standardized and borne in mind when making Bayesian estimates of the initial number of faults and their associated failure rates. Software will not fail during periods when it is not in use, but only when it is executing. One might measure time as execution time \( e \), or equivalently as calendar time \( t \) if \( e = ft \) and \( f \) is a known fraction of the time during which the software is executing. If \( k \) multiple, parallel tests are made, one might also assume that \( e = kft \). Although there may be substantial uncertainty about conversion factors such as \( k \) and \( f \), we will not deal with that uncertainty here, referring only to time in the sequel.

Our methods enable a natural, sequential approach to reliability tracking and estimation. First one performs some tests, observes and eliminates some faults, and estimates reliability. Based on that estimation, one might perform further tests before submitting the software to an operational phase. Although faults discovered in the operational phase might be particularly expensive to eliminate, they are treated here just like faults discovered during testing. Bayesian parameters are updated in either case, and reliability forecasts are based on them.

2. BACKGROUND

One of the earliest models of software reliability is that of Jelinski and Moranda [6], who suppose that there are \( N \) faults, the \( i \)th of which has a failure time \( T_i \). All of these failure times are supposed to be independent random variables, given \( N \), and all have the same unknown but constant
failure rate \( \phi \). Testing over some time interval \( t_0 \), in addition to eliminating all faults for which \( T_i \leq t_0 \), provides data from which \( N \) and \( \phi \) can be estimated. They find maximum likelihood estimates (MLE) of \( N \) and \( \phi \), after which those estimates are simply “plugged in” as if there were no further uncertainty about either \( N \) or \( \phi \). Littlewood [10] takes a Bayesian approach to the same fundamental model, assuming that each fault has a failure rate with a Gamma(\( \alpha, \beta \)) distribution, independent of the other faults. Conditional on testing for a time \( t_0 \), each of the surviving faults has a Gamma(\( \alpha, \beta + t_0 \)) distribution; that is, testing provides evidence that the failure rates of surviving faults tend to be small. This evidence influences forecasts, as does the number of faults removed by testing. Estimation of \( N \) is still by MLE. The current paper can best be regarded as a further generalization where \( N \), as well as \( \phi \), is subjected to Bayesian analysis.

There have been other Bayesian approaches. Littlewood and Verrall [9] account for the possibility that removing one fault can introduce others. Their system failure rate is assumed to be Gamma distributed, but with parameters that depend on the failure index in such a manner that the times to failure are stochastically decreasing. Meinhold and Singpurwalla [13] incorporate a prior distribution for the number of faults, as well as for the common failure rate.

Several software reliability models are based on the idea that the number of failures encountered up to time \( t \) is a Non-homogeneous Poisson Process (NHPP), with various assumptions about the rate function. Crow (1974) [2] shows that Duane’s model of reliability growth [3] can be regarded as a model of this type. The models of Musa [15], Schniedewind [22], and Musa and Okumoto [16] are also of this type. See Musa, Iannino, and Okumoto [17] for a comprehensive survey, or AIAA [1] or Lyu [12]. NHPP-based models are widely applied on account of their simplicity and lack of a requirement for prior distributions, although assumptions about the rate function are still needed. Bayesian models such as the one described here nonetheless deserve consideration when there is no reason to suspect bias in the selection of priors. This is particularly true when testing and use are repetitive and sequential.

Singpurwalla and Wilson [23] are concerned with finding a unifying framework for the large variety of software reliability models that have been developed over the past several decades, it being their contention that most such models are special cases of Self Exciting Point Processes (SEPP). The current model is no exception: it is a SEPP with zero memory, just like the Jelinski–Moranda model. It is also a Doubly Stochastic Exponential Order Statistic model according to Miller [14]. Since failure times have a Pareto distribution when failure rates have a Gamma distribution, it also corresponds to Kuo and Yang’s [8] Generalized Order Statistic model with Pareto failure times.

The present paper differs from previous work in that the prior distribution for \( N \) is taken to be negative binomial and also in that worst-case failure rates are considered an alternative to Bayesian assumptions. The worst-case failure rate in models such as this is not infinite, since faults with high failure rates are discovered and removed almost immediately. Neither is it zero, since faults that never happen cannot be embarrassing. Worst-case failure rates are considered in Section 6.

3. THE BAYESIAN MODEL

We suppose that the software has an unknown number of faults \( N \), each with an unknown failure rate \( L_i \), \( i = 1, \ldots, N \). Given \( N \), the failure rates are assumed to be independent, identically distributed random variables. Each fault \( i \) has a failure time \( T_i \) that is exponentially distributed with its own failure rate, independently of the others. One can imagine that the fault failure times are determined or simulated by the following three-step process:

- determine \( N \) as a sample from some prior discrete distribution,
- given \( N \), determine the failure rates \( L_1, \ldots, L_N \) as independent samples from some prior failure rate distribution, and finally
- given \( L_i \), sample the failure time \( T_i \) as an independent, exponential random variable with rate \( L_i \), \( i = 1, \ldots, N \).

During a testing period of length \( t_0 \), faults for which \( T_i \leq t_0 \) are discovered and eliminated, with no new faults being introduced in the process. After the testing period, we are interested in the probability \( R(t) \) that no additional faults will be discovered in the period \([t_0, t_0 + t]\). \( R(t) \) is by definition the reliability of the software over a time period of length \( t \).

It would be natural to suppose that \( N \) has a Poisson distribution, since that distribution characterizes situations where there are many independent trials with a small probability of success per trial. However, there is often uncertainty about the meaning and measure of “trials” (modules?, characters?, lines of code?) and the probability of “success” per trial. Experience shows that the mean of \( N \) depends on the type of code, the conditions of its creation, and, above all, its length. AIAA [1], for example, suggests that the fault probability per line of computer code is highly variable, ranging from 0.001 in a highly disciplined programming environment to 0.01 in a routine environment. Let \( M \) be the mean of \( N \). To accommodate uncertainties of the type just described, we suppose that \( M \) itself is a random variable having a Gamma(\( r, b \)) distribution, where \( r \) is a shape parameter and \( b \) is a scale parameter; \( r > 0 \) and \( b > 0 \). The
coefficient of variation (the ratio of the standard deviation to the mean) for such a distribution is $r^{-1/2}$, so $r$ might be on the order of 1 or 2 if the fault density were as variable, as implied by AIAA. Parameter $r$ depends on the code production process and the project’s novelty, but not on the scale of the project, which is incorporated in $b$. Since $E(N) = r/b$, $b$ might be on the order of 0.01 for a 10,000-line computer program.

These distributional assumptions are analytically convenient, as well as flexible enough to accommodate realistic uncertainty about conditions of creation and use. The doubly stochastic assumptions about $N$ (Poisson with a mean that is Gamma) result in a negative binomial distribution [7] for $N$. Specifically, $N$ has a negative binomial (NB) distribution with parameters $r$ and $b/(b+1)$. Figure 1 shows the NB distribution with $r = 1.5$ and $b = 0.01$.

As will be seen in the next section, the NB distribution is also a conjugate prior for $N$ in the sense that evidence from testing simply revises the parameters $r$ and $b$ through Bayes theorem, thus permitting a simple, sequential view of testing.

Like Littlewood [10], we suppose that each fault has a constant failure rate $L_i$ and that $L_i$ has a Gamma($\alpha$, $\beta$) distribution; $i = 1, \ldots, N$. Given $N$, the failure rates of individual faults are assumed to be identically distributed random variables selected independently from that distribution. An alternative would be to assume that the same failure rate applies to all faults [13], but the independence assumption better captures the idea that some faults are harder to find than others. For brevity in the sequel, we will simply state that $L$ has a Gamma($\alpha$, $\beta$) distribution and that $L$ is independent of $N$, but the exact meaning of that statement is as given above. Since all failure rates have the same distribution, the symbol $L$ is used to refer to the typical failure rate.

The distributions of $N$ and $L$ are to be jointly updated based on the results of testing and then used to forecast reliability over some future period.

### 4. Bayesian Analysis

Section 8 is an appendix that deals with the Bayesian updating of the distributions of $L$ and $N$ based on the evidence that $k$ faults are removed in the process of testing over the period $[0, t_0]$. The essential result is that the number of remaining faults and $L$ remain independent with distributions that are NB and Gamma, respectively, except that the parameters are updated based on testing results.

The posterior distribution of $L$ turns out to be Gamma($\alpha'$, $\beta'$), where

$$\alpha' = \alpha \quad \text{and} \quad \beta' = \beta + t_0. \quad (4.1)$$

The probability that a particular fault will survive the testing period is

$$\rho(t_0) = E(\exp(-Lt_0)) = (\beta/(\beta + t_0))^b. \quad (4.2)$$

Equation (4.2) is essentially the characteristic function [19] of a Gamma($\alpha$, $\beta$) random variable.

Equations (4.1) and (4.2) can be used repeatedly if there are multiple testing periods. It is not significant if a testing period is partitioned into several parts. If $[0, t_0]$ is partitioned into $[0, x]$ and $[x, t_0]$, the probability of surviving the first part is $(\beta/(\beta + x))^b$, the probability of surviving the second is $(\beta + x)/(\beta + t_0))^b$, and the probability of surviving both is the product of the two, i.e., (4.2).

Let $N'$ be the number of faults remaining after testing. According to the theorem in the Appendix, with $p = b/(b + 1)$, $N'$ has the distribution $\text{NB}(r + k, 1 - \rho(t_0)(1 - p))$, which is the same as $\text{NB}(r', b'/(b' + 1))$ if we set

$$r' = r + k \quad \text{and} \quad b' = (b + 1)/\rho(t_0) - 1, \quad (4.3)$$

with $\rho(t_0)$ as given by (4.2). Furthermore, conditional on $N'$, the failure rates of the remaining faults are all independent Gamma random variables with distribution given by (4.1).

Thus, the NB and Gamma distributions are conjugate priors for $N$ and $L$ in the sense that, conditional on knowing the number of faults observed and eliminated, the distributions of the number and failure rate of the surviving faults are still NB and Gamma, respectively, with updated parameters.

Equation (4.3) seems not to have been exploited before in the software reliability literature. As a point of interest, the equation has had some application in minefield clearance theory [24], where the problem is to estimate the number of mines remaining after an imperfect clearance effort. In both software reliability and minefield clearance, the testing process removes all faults (mines) that are found, as well as
providing evidence about the number remaining. The NB distribution is a useful prior in Bayesian approaches to either subject. It is a two-parameter distribution that permits the variance to be large relative to the mean.

One might have hoped for more than (4.3), since the times at which the faults occur are observed, as well as their number. However, it turns out (see Appendix) that the fault times are irrelevant in making Bayesian updates of this kind; it is sufficient to note \( t_0 \) and the number of faults discovered. The simplicity of this feature is in a sense welcome, but the corresponding cost is that the shape parameter \( \alpha \) of the failure rate distribution of surviving faults is never updated. Contrast this with the results of Meinhold and Singpurwalla [13], who describe a related Bayesian model where all failure rates are identical, albeit unknown. In that case the shape parameter is observable, which is welcome, but the corresponding cost is that the posterior independence of \( N \) and \( L \) is lost.

On account of the sequential nature of testing, it would suffice to develop reliability forecasts at time 0, since dealing with later times is simply a matter of substituting different values for \( \alpha, \beta, r, \) and \( b \). Instead, we deal with some time \( t_0 \) at which a test has just been completed, recording \( k \) failures in the process. The probability that each fault remaining at \( t_0 \) will survive for an additional time \( t \) can be obtained from (4.2), with \( \alpha' \) and \( \beta' \) from (4.1) substituted for \( \alpha \) and \( \beta \). This probability is \( q = ((\beta + t_0)/(\beta + t_0 + t))^\alpha' \). Since \( R(t) \) is the probability that all \( N' \) remaining faults survive, we have

\[
R(t) = E(q^{N'}) = \left( \frac{b'}{b' + 1 - q} \right)^r \left( \frac{b + 1 - (\beta/(\beta + t_0))^\alpha}{b + 1 - (\beta/(\beta + t_0 + t))^\alpha} \right)^{r+k}, \tag{4.4}
\]

a result obtained from the characteristic function of a NB\((r', b'(b' + 1))\) random variable, with \( r' \) and \( b' \) obtained from (4.3). Equation (4.4) provides a reliability forecast that is based on testing results achieved so far.

Reliability over any period is thus determined in a simple manner by the initial distributions of \( N \) and \( L \), the length and results of the testing program, and the length of the subsequent operational period \( t \). Note that the limit of \( R(t) \) for large \( t \) is not 0, but rather \( P(N' = 0) \). “Time to failure” is therefore a defective random variable; the MTBF is undefined (infinite).

Inspection of (4.4) reveals that reliability is a decreasing function of \( k \). This may be counterintuitive, since one might expect reliability to increase with the number of faults found and removed. Certainly it cannot hurt to remove faults, but a countervailing tendency is that a large number of faults removed implies that a large number probably remain. This tendency turns out to be dominant when the initial number of faults is NB.

In the limit where \( r \) and \( b \) both become very large while the ratio \( r/b \) remains fixed at \( m \), the initial distribution of \( N \) becomes Poisson with mean \( m \). In that case formula (4.4) should be replaced by its limiting version,

\[
R(t) = \exp(-m((\beta/(\beta + t_0))^\alpha - (\beta/(\beta + t + t))^\alpha)), \tag{4.5}
\]

and the update formula (4.3) should be replaced by \( m' = \rho(t_0)m \), that is, the number of residual faults is Poisson with a mean unrelated to the number found. Similar update formulas are available when the initial distribution of \( N \) is binomial [24], in which case reliability increases with the number of faults removed. The Poisson case is thus intermediate between binomial and NB.

For purposes of determining the testing period, one might also want to explore the impact of testing for a time \( t_0 \), without obtaining any information about the as yet unknown number of faults found in testing. There is no change in the update formula (4.1), since it does not depend on the number of faults found. The distribution of the number of faults that remain at the end of \( t_0 \), without knowing the number found during that period, is still NB, with revised parameters \( r'' = r \) and \( b'' = b/(\rho(t_0)) \). The reliability over any subsequent period can be obtained from (4.4), provided current parameter values as of time \( t_0 \) are substituted into (4.2) and (4.4). Equivalently, let \( T \) be the failure time of a fault. The probability of the event \( t_0 \leq T \leq t_0 + t \) is \( \rho(t_0) - \rho(t_0 + t) \). This is the only embarrassing event—the fault will be eliminated if it occurs early and is not significant if it occurs late. The probability that there will be no embarrassing events is then

\[
RR(t_0, t) = E((1 - \rho(t_0) + \rho(t_0 + t))^N) \left( \frac{b}{b + \rho(t_0) - \rho(t_0 + t)} \right)^r. \tag{4.6}
\]

Equation (4.6) can be the basis of determining how much testing should be done in order to achieve a given reliability, on average. However, the ultimate reliability forecast based on that testing should be done with (4.4), which will depend on how many faults are found.

EXAMPLE: Suppose \((r, b) = (1.5, 0.01)\) and \((\alpha, \beta) = (0.3, 1 \text{ day})\), in which case \( E(N) = 150 \) faults and \( E(L) = 0.3 \) per day. A testing program is proposed for 10 days, after which it is hoped that the system will function for an additional 5 days with no further faults. Substituting into (4.6), we find that the expected reliability is 0.065. This is disappointingly small, so we decide to increase the length of the test to \( t_0 = 100 \) days. Suppose that the number of
should not be surprising, since the expected number of faults remaining decreases sharply, with a corresponding decrease in reliability (Eq. (4.4)). The forecast reliability eventually begins a slow increase after about 20 tests. The initial guess at the number of faults is \( r/b = 150 \), but the expected number remaining increases rapidly over the first several 10-second tests. After 100 tests in which 111 faults are discovered and removed, \( E(N) \) is 1944, considerably larger than 150 and all previous estimates in the literature. Estimates of residual values for \( E(N) \) are sensitive to initial parameter values, particularly \((\alpha, \beta)\). This should not be surprising, since \( N \) and \( L \) are nearly confounded in a reliability experiment; the product \( NL \) ("total failure rate") is more significant than either factor by itself. Reliability estimates are less sensitive to initial parameter choices. When each of the four parameters \((\alpha, \beta, r, b)\) is individually doubled, for example, the reliability estimate after 1000 seconds of testing changes from 0.436, as shown in Figure 2, to \((0.431, 0.495, 0.410, 0.473)\).

Our contention is that reliability models ought to be compared on the basis of reliability forecasts, rather than on estimates of \( E(N) \) or \( E(L) \). How can the quality of reliability forecasts be tested? Consider the following procedure. Let \( R_i(t) \) be the \( i \)th reliability forecast, let \( T_i \) be the time to the next failure, and let \( H_i \) be the history of previous observations, i.e., \( H_i = (T_1 = t_1, \ldots, T_{i-1} = t_{i-1}) \). In other words, \( R_i(t) = P(T_i > t|H_i) \). Also define random variables \( U_i = R_i(T_i) \). The events \( (U_1 < R_1(t)) \) and \( (T_1 > t) \) are equal, so \( P(U_1 < R_1(t)|H_i) = P(T_1 > t|H_i) = R_1(t) \). Since this is true for all \( t \), it follows that \( P(U_1 < u|H_i) = u \) for all \( u \) in \([0, 1]\), i.e., \( U_i \) is uniform on the interval \([0, 1]\), given \( U_1, \ldots, U_{i-1} \). Thus, \( U_1, U_2, \ldots \) is (or should be) a sequence of independent, standard uniform random variables and should therefore pass a Kolmogorov–Smirnov (KS) test.

To make a KS test for the sequential Bayesian procedure applied to the Musa data, sequentially process the first 100 failures, one at a time (the theorem in the Appendix permits stopping times of the form "stop at the next failure"). After each failure there is a forecast of the reliability function of the next failure time, with associated statistics \( U_1, \ldots, U_{100} \). The resulting KS statistic is 0.083. Based on this small value, the null hypothesis of uniformity cannot be rejected. According to the KS test, the reliability forecasts are accurate.

Of course, one would like to know how many faults are truly present, but that number can never be known for real software. Monte Carlo simulation experiments offer the advantage of having “true” values of \( N \) and other random variables available for comparison. The reader may wish to download and experiment with an Excel workbook SofRel.xls (http://diana.cs.nps.navy.mil/~arwashbu/) that performs Monte Carlo experiments for Bayesian software reliability problems such the one described above. The user sets \((\alpha, \beta, r, b)\), after which the initial number of faults \( N \) is sampled, together with a failure rate \( L \), and then a failure time \( T_i \) for each fault; \( i = 1, \ldots, N \). The sequential algorithm is then applied, after which true and estimated values can be compared.

Ideally, one would discover after experimenting with such simulations that forecasts and estimates are robust to errors made in initially estimating the four parameters, since such errors can be expected in practice. Unfortunately, this is not the case. Reliability estimates are more robust than estimates of \( L \) or especially \( N \), but even reliability estimates have a disappointing tendency to depend on the accuracy of initial assumptions. Given this lack of robustness, it is natural to consider worst-case assumptions about failure rates. This is the subject of the next section.

5. EXPERIMENTS

Musa [15] reports and analyzes the first 136 failure times of a command-and-control system. These same data are also analyzed by Meinhold and Singpurwalla [13], Goel [4], Okumoto [1985] [18], Raftery [20], and Littlewood [11]. We process the same data using the Bayesian algorithm outlined above, using an arbitrary initial parametric starting point except that the two scale parameters \( b \) and \( \beta \) are Maximum Likelihood Estimates. The solid curve in Figure 2 shows the forecast reliability over the next 10-second testing period after each of the first 100 testing periods. The first several tests reveal large numbers of faults, with the result that the estimated number of faults remaining increases sharply, with a corresponding decrease in reliability (Eq. (4.4)). The forecast reliability eventually begins a slow increase after about 20 tests.

The initial guess at the number of faults is \( r/b = 150 \), but the expected number remaining increases rapidly over the first several 10-second tests. After 100 tests in which 111 faults are discovered and removed, \( E(N) \) is 1944, considerably larger than 150 and all previous estimates in the literature. Estimates of residual values for \( E(N) \) are sensitive to initial parameter values, particularly \((\alpha, \beta)\). This should not be surprising, since \( N \) and \( L \) are nearly confounded in a reliability experiment; the product \( NL \) ("total failure rate") is more significant than either factor by itself. Reliability estimates are less sensitive to initial parameter choices. When each of the four parameters \((\alpha, \beta, r, b)\) is individually doubled, for example, the reliability estimate

Figure 2. Reliability forecasts over a 10-second interval for data reported by Musa, as a function of test index. Each test consists of counting the number of faults in a 10-second interval. Initial \((\alpha, \beta, r, b) = (0.3, 6000, 1.5, 0.01)\). The dotted line shows \( R^*(10) \), a lower bound introduced in Section 6.
### Table 1. The function \( REL(b, t_0/t) \) for selected values of \( b \) (columns) and \( t_0/t \) (rows).

<table>
<thead>
<tr>
<th>( t_0/t )</th>
<th>( 0.125 )</th>
<th>( 0.25 )</th>
<th>( 0.5 )</th>
<th>( 1 )</th>
<th>( 2 )</th>
<th>( 4 )</th>
<th>( 8 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.6667</td>
<td>0.7236</td>
<td>0.7887</td>
<td>0.8536</td>
<td>0.9082</td>
<td>0.9472</td>
<td>0.9714</td>
</tr>
<tr>
<td>2</td>
<td>0.7878</td>
<td>0.8265</td>
<td>0.8693</td>
<td>0.9107</td>
<td>0.9447</td>
<td>0.9684</td>
<td>0.9830</td>
</tr>
<tr>
<td>4</td>
<td>0.8768</td>
<td>0.9002</td>
<td>0.9256</td>
<td>0.9496</td>
<td>0.9690</td>
<td>0.9824</td>
<td>0.9905</td>
</tr>
<tr>
<td>8</td>
<td>0.9329</td>
<td>0.9460</td>
<td>0.9600</td>
<td>0.9731</td>
<td>0.9835</td>
<td>0.9907</td>
<td>0.9950</td>
</tr>
<tr>
<td>16</td>
<td>0.9649</td>
<td>0.9718</td>
<td>0.9792</td>
<td>0.9860</td>
<td>0.9915</td>
<td>0.9952</td>
<td>0.9974</td>
</tr>
</tbody>
</table>

*Note. This function must be raised to an appropriate power to obtain a reliability.*

6. **WORST-CASE ANALYSIS**

Different faults have different failure rates, and failure rates are notoriously variable. One reaction to this situation is a Bayesian approach involving random failure rates, as described above. A more conservative approach is to base reliability calculations on worst-case failure rate assumptions. If all failure rates were 0, then of course the reliability of the software would be 1. As long as there is an initial testing program, the same statement is true if the failure rates are very large, since all faults will be discovered and removed in the testing program. The worst possible failure rate will be found somewhere in the middle. In this section we abandon the Bayesian approach in favor of assuming that \( L \) is an unknown, adversely selected constant, the same for every fault. We will continue to assume that \( N \) has a negative binomial distribution.

When \( L \) is deterministic, the survival probabilities \( p(t_0) \) (used in (4.3) to update \( b \)) and \( q \) (used in (4.4) to calculate \( R(t) \)) should be replaced by \( \exp(-L t_0) \) and \( \exp(-Lt) \), respectively. In that case (4.4) becomes

\[
R(t) = E(q^n) = \left( \frac{b'}{b' + 1 - q} \right)^r \quad = \left( \frac{b + 1 - \exp(-L t_0)}{b + 1 - \exp(-L(t_0 + t))} \right)^{i + k}. \tag{6.1}
\]

Here \( b \) and \( r \) are the initial parameters of the distribution of \( N \), \( t_0 \) is the total amount of time on test, and \( k \) is the total number of faults detected and eliminated in that time. Equating to 0 the derivative of \( R(t) \) with respect to \( L \) produces a transcendental equation, so there is no closed-form solution for the minimizing \( L \). However, the minimum depends only on the two parameters \( b \) and \( t_0/t \) and is not difficult to find using numerical techniques. Let

\[
REL(b, t_0/t) = \min_L \left( \frac{b + 1 - \exp(-L t_0)}{b + 1 - \exp(-L(t_0 + t))} \right). \tag{6.2}
\]

The aforementioned workbook SofRel.xls includes VBA code for calculating this function, using Newton’s method to find the point where the derivative is 0. Table 1 gives some typical values for the function \( REL(\cdot) \).

The minimized \( R(t) \) is \( R^*(t) = REL(b, t_0/t)^{i + k} \). Figure 2 shows \( R^*(t) \) along with \( R(t) \) for \( t = 10 \) in 100 sequential tests of the Musa data. If the results of testing were held fixed while all possible values (\( \alpha, \beta \)) were used to generate a family of reliability curves, the lower envelope of that family would be \( R^*(t) \).

An analytic lower bound can be derived from (6.1). Since the function \((1 + x) \exp(-x)\) is decreasing, it follows that \((1 + x)/(1 + y) \geq \exp(y - x)\) as long as \( x \leq y \). Therefore, letting \( x = (1 - \exp(-L t_0))/b \) and \( y = (1 - \exp(-L(t_0 + t)))/b \),

\[
\left( \frac{b + 1 - \exp(-L t_0)}{b + 1 - \exp(-L(t_0 + t))} \right) \geq \exp\left( -\left[ \exp(\exp(-L t_0)) - \exp(-L(t_0 + t)) \right]/b \right). \tag{6.3}
\]

The quantity enclosed in \( \{ \} \) in (6.3) is the probability that any given fault will prove embarrassing in the sense of failing after \( t_0 \), but before \( t_0 + t \). Let \( z \) be that quantity, let \( \phi \) be the value of \( L \) that minimizes \( z \), and let \( z^* \) be the minimized embarrassment probability. By equating the derivative of \( z \) with respect to \( L \) to 0 and solving the resulting equation, we find that \( \phi = \ln(1 + t (t_0/t))/t \). Substituting this for \( L \) in the expression for \( z \), we find that the minimal value of \( z \) is

\[
z^*(x) = x^r(1 + x)^{1+r}, \quad \text{where} \quad x = t_0/t. \tag{6.4}
\]

Substituting \( z^*(t_0/t) \) into (6.3) and then (6.3) into (6.1), we find that

\[
R(t) = R^*(t) \geq R^{**}(t) = \exp(-r + k)z^*(t_0/t)/b. \tag{6.5}
\]

As usual, the initial number of faults is assumed to be NB(\( r, b(t_0 + 1) \)), and \( k \) faults are found in testing to \( t_0 \).

The function \( z^*(x) \) is shown in Figure 3. Since \( z^*(x) \) decreases rapidly when \( x \) is small, \( R^{**}(t) \) will be small unless \( t_0 \) is large relative to \( t \).

The functions \( R^*(t) \) and \( R^{**}(t) \) are both bounds on \( R(t) \) that require no assumptions about failure rates, other than...
constancy in time. $R^*(t)$ is the better bound, but $R^{**}(t)$ is easier to compute. Table 2 shows the logarithm of the ratio, except for the exponent $r + k$. It should be apparent that $R^{**}(t)$ is at its best when $b$ is large. Indeed, in the Poisson limit where $b$ and $r$ both become large while $r/b = m$, both bounds approach $\exp(-mz^*(t_0/t))$.

In the example of Section 4, $b$ was very small (0.01), while $t_0/t = 20$. In that case $R^{**}(t)$ is essentially zero, while $R^*(t) = 0.114$ and $R(t) = 0.784$. The great disparity between the latter two is explained by the observation that the worst-case failure rate is 0.0013 per day, whereas the failure rate distribution lying behind $R(t)$ has a mean of 0.0030 per day, with considerable variance. In such cases the worst-case approach is, indeed, pessimistic. To construct an example where $R(t)$ and $R^*(t)$ are approximately equal, it would suffice to posit a prior distribution for $\lambda$ with a small variance (large $\sigma$) and a mean that happens to agree with the worst-case failure rate.

The function $z^*(\cdot)$ has another use in forecasting the amount of testing time needed to achieve a given reliability. When the failure rate $L$ is not random, Eq. (4.6) becomes

$$RR(t_0, t) = \left( \frac{b}{b + \exp(-Lt_0) - \exp(-Lt_0 + t)} \right)^r.$$  \hspace{1cm} (6.6)

The difference of exponentials is $z$, and $z$ cannot exceed $z^*(t_0/t)$. Therefore,

$$RR(t_0, t) \equiv RR^*(t_0, t) = \left( \frac{b}{b + z^*(t_0/t)} \right)^r.$$  \hspace{1cm} (6.7)

Given a reliability goal and a forecasting period $t$, Eq. (6.7) enables the conservative adjustment of the testing period $t_0$ to meet the goal. To continue the example of Section 4, suppose $(r, b) = (1.5, 0.01)$, and that the goal is to achieve a reliability of 0.9 over a 5-day period. Setting $t_0$ to 2535 days will accomplish this. This testing period is long not because of approximations made in deriving it (there are none), but because it is difficult to guarantee that software that initially has about 150 faults in it will function reliably over a period of 5 days, while making worst-case assumptions about failure rates. As in the case of (4.6), it should be borne in mind that the quantity being manipulated is only the average reliability. Once the tests have been completed, a better informed forecast can be made by taking account of the number of failures observed, i.e., by computing $R^*(t)$ as described above.

7. SUMMARY

The basic problem considered in this paper is one where a system has an unknown number of faults, each of which is discovered and instantly eliminated in the process of testing or using the system. At any given time, it may be necessary to forecast the reliability of such a system, by which is meant the probability that no further faults will be encountered over some specified future interval of time. We have found that these forecasts can be made in the context of a state-based, Bayesian system where both the number of faults $N$ and the failure rates $L$ are assumed to have probability distributions of a specified two-parameter type, with the four parameters constituting the state. The specific operations covered are as follows.

Update the parameters to account for testing. The relevant formulas are (4.1), which updates the parameters for $L$, and (4.3), which updates the parameters for $N$.

Forecast reliability given test results. The relevant formula is (4.4).

Forecast reliability without test results. The relevant formula is (4.6).

<table>
<thead>
<tr>
<th>$t_0/t$</th>
<th>$b$</th>
<th>0.125</th>
<th>0.25</th>
<th>0.5</th>
<th>1</th>
<th>2</th>
<th>4</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1.5945</td>
<td>0.6765</td>
<td>0.2626</td>
<td>0.0917</td>
<td>0.0288</td>
<td>0.0083</td>
<td>0.0022</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>0.9467</td>
<td>0.4020</td>
<td>0.1562</td>
<td>0.0546</td>
<td>0.0171</td>
<td>0.0049</td>
<td>0.0013</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>0.5239</td>
<td>0.2225</td>
<td>0.0865</td>
<td>0.0302</td>
<td>0.0095</td>
<td>0.0027</td>
<td>0.0007</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>0.2770</td>
<td>0.1177</td>
<td>0.0458</td>
<td>0.0160</td>
<td>0.0050</td>
<td>0.0014</td>
<td>0.0004</td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>0.1426</td>
<td>0.0606</td>
<td>0.0236</td>
<td>0.0082</td>
<td>0.0026</td>
<td>0.0007</td>
<td>0.0002</td>
<td></td>
</tr>
</tbody>
</table>

Figure 3. A graph of formula 6.4: $z^*$ versus $x$. 

The testing program can be sequential; that is, the fundamental operations can be employed multiple times for a given system.

In addition, we have considered the possibility that forecasts might be based on worst-case assumptions about $L$. Depending on Bayesian assumptions about failure rates, worst-case estimates of reliability can be similar to, or much smaller than, the corresponding Bayesian estimates.

8. APPENDIX

THEOREM: Suppose that the number of faults $N$ has a negative binomial distribution with parameters $r$ and $p$, with $q = 1 - p$, and that each fault $i$ has an associated failure rate $L_i$ that is independently selected from a Gamma distribution with parameters $\alpha$ and $\beta$. Given the failure rates, the failure time $T_i$ for fault $i$ is independently selected from the density $\exp(-L_i t)$, i.e., the failure times are conditionally independent and exponential. Let $T_0$ be any proper stopping time with respect to the stochastic process that consists of the order statistics of the failure times. For $i > 0$, $T_i$ is observed and fault $i$ is removed if and only if $T_i \leq T_0$. Let $M$ be the number of faults not removed, and let $E$ be the event that the observed failure times are $t_1, \ldots, t_k$, and that the stopping time is $T_0$.

- for $i > 0$, $T_i$ has a Pareto distribution, that is, $P(T_i > t) = (\beta(\beta + t))^{q \alpha} = \rho(t)$;
- conditional on $E$ being given, $M$ has a negative binomial distribution with parameters $r + k$ and $1 - q \rho(t_0)$;
- conditional on $E$ and $M$ being given, the remaining $M$ faults have failure rates that are all independent, each having a Gamma distribution with parameters $\alpha$ and $\beta + t_0$.

PROOF: The first claim has been proved by Harris [5]. To prove the other two, we must consider the joint probability of $N$, $E$, and the remaining failure rates. Let vector $L$ be the failure rates of the $n - k$ remaining faults, and define vectors $T = (T_0, T_1, \ldots, T_k)$ and $t = (t_0, t_1, \ldots, t_k)$. Also let $g(k)$ be the Gamma density of each failure rate and $\text{pmf}(n)$ be the probability mass function of $N$:

\[
\text{pmf}(n) = \left( \frac{r + n - 1}{n} \right) p^r q^n; \quad n \geq 0. \tag{8.1}
\]

Then

\[
P(N = n, E, L = \lambda) = \text{pmf}(n) \frac{n!}{(n-k)!} \left( P(T = t) \right) \times \prod_{i=1}^{k} \exp(-\lambda_i t_i) g(\lambda_i) d\lambda_i. \tag{8.2}
\]

The factor $n!/(n-k)!$ in (8.2) is the number of permutations of $n$ things taken $k$ at a time. It is needed because the event $E$ does not specify the indexes of the observed faults, but only that $k$ of them are observed at the stated failure times. The factor $\exp(-\lambda_i t_i)$ is present because each of the surviving faults must have a failure time that exceeds the stopping time.

Equation (8.2) is the product of the three factors shown in (8.1). Call the first $A(k, n)$, the second $B(t)$, and the third $C(\lambda_1, \ldots, \lambda_n)$. Integrating (8.2) with respect to $\lambda_k+1, \ldots, \lambda_n$, we find that

\[
P(N = n, E) = A(k, n) B(t) \rho(t_0)^{k-1}. \tag{8.3}
\]

It follows upon dividing (8.2) by (8.3) that

\[
P(L = \lambda|N = n, E) = \prod_{i=k+1}^{n} \left( \exp(-\lambda_i t_0) g(\lambda_i) \rho(t_0) \right) d\lambda_i. \tag{8.4}
\]

When $E$ is given, the event $(N = n)$ is the same as the event $(M = n - k)$. Thus, (8.4) establishes the third claim, since each factor in the product is a Gamma density with parameters $\alpha$ and $\beta + t_0$ [10].

Equation (8.3) can be rewritten

\[
P(M = m, E) = A(k, m + k) B(t) \rho(t_0)^m. \tag{8.5}
\]

To find $P(E)$, we substitute (8.1) for $\text{pmf}(m + k)$ and sum (8.5) over all possible values for $m$:

\[
P(E) = B(t) \sum_{m=0}^{\infty} \frac{(r + m + k - 1)!}{(m + k)! (r - 1)! \rho(t_0)^m} \frac{m! k^m}{m!} \rho(t_0)^m. \tag{8.6}
\]

Equation (8.6) can be rearranged to be

\[
P(E) = B(t) \frac{p^r q^k (r + k - 1)!}{k! (r - 1)!} \sum_{m=0}^{\infty} \frac{(r + m + k - 1)!}{m! (r + k - 1)!} \rho(t_0)^m. \tag{8.7}
\]

The negative binomial distribution sums to 1, so the sum in (8.7) must be $(1 - \rho(t_0)q)^{(r+k)}$. Taking the ratio of (8.5) to (8.7), we finally find

\[
P(M = m|E) = \left( \frac{r + m + k - 1}{m} \right) (1 - \rho(t_0)q)^{m-k} \rho(t_0)^m; \quad m \geq 0. \tag{8.8}
\]

Equation (8.8) establishes the second claim and the theorem. □

REFERENCES


Naval Research Logistics DOI 10.1002/nav


