SOFTWARE RELIABILITY ESTIMATION UNDER CONDITIONS OF INCOMPLETE --ETC(U)
OCT 79 C K RUSHFORTH, F L STAFFANSON

UNCLASSIFIED - UTEC-79-063
SOFTWARE RELIABILITY ESTIMATION UNDER CONDITIONS OF INCOMPLETE INFORMATION

University of Utah

C. K. Rushforth
F. L. Staffanson
A. E. Crawford

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED

ROME AIR DEVELOPMENT CENTER
Air Force Systems Command
Griffiss Air Force Base, New York 13441
This report has been reviewed by the RADC Public Affairs Office (PA) and is releasable to the National Technical Information Service (NTIS). At NTIS it will be releasable to the general public, including foreign nations.

RADC-TR-79-230 has been reviewed and is approved for publication.

APPROVED:

ALAN N. SUKERT
Project Engineer

APPROVED:

WENDALL C. BAUMAN, COLONEL, USAF
Chief, Information Sciences Division

FOR THE COMMANDER:

JOHN P. RUSS
Acting Chief, Plans Office

If your address has changed or if you wish to be removed from the RADC mailing list, or if the address is no longer employed by your organization, please notify RADC (IBCS), Griffiss AFB NY 13441. This will enable us in maintaining a current mailing list.

Do not return this copy. Retain as reference.
SOFTWARE RELIABILITY ESTIMATION UNDER CONDITIONS OF INCOMPLETE INFORMATION

C. K. Rushforth
F. L. Staffanson
A. E. Crawford

PERFORMING ORGANIZATION NAME AND ADDRESS
University of Utah
Electrical Engineering Department
Salt Lake City UT 84112

CONTROLLING OFFICE NAME AND ADDRESS
Rome Air Development Center (ISIS)
Griffiss AFB NY 13441

MONITORING AGENCY NAME AND ADDRESS (IF different from Controlling Office)
Same

MONITORING AGENCY NAME AND ADDRESS (IF different from Controlling Office)
Same

PROJECT ENGINEER: Alan N. Sukert (ISIS)

This report describes a computer program for the estimation of the reliability of large software packages. A model is developed which incorporates additional features found in realistic testing environments. The model is analyzed for deterministic data. Performance on simulated random test data is presented. The algorithm estimates the parameters of the model from software error data, and computes therefrom running estimates of the mean-time-to-failure and the number of software errors remaining.
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.0</td>
<td>Introduction</td>
<td>1.1</td>
</tr>
<tr>
<td>1.1</td>
<td>Problem Statement</td>
<td>1.1</td>
</tr>
<tr>
<td>1.2</td>
<td>Previous Work</td>
<td>1.1</td>
</tr>
<tr>
<td>2.0</td>
<td>Development and Analysis of the Model</td>
<td>2.1</td>
</tr>
<tr>
<td>2.1</td>
<td>Assumptions</td>
<td>2.1</td>
</tr>
<tr>
<td>2.2</td>
<td>The Model</td>
<td>2.4</td>
</tr>
<tr>
<td>2.3</td>
<td>Model Behavior</td>
<td>2.9</td>
</tr>
<tr>
<td>2.4</td>
<td>The Data Simulator</td>
<td>2.10</td>
</tr>
<tr>
<td>3.0</td>
<td>The Estimation Algorithm</td>
<td>3.1</td>
</tr>
<tr>
<td>3.1</td>
<td>The Estimation Problem and Method</td>
<td>3.1</td>
</tr>
<tr>
<td>3.2</td>
<td>Description of the Search Algorithm</td>
<td>3.4</td>
</tr>
<tr>
<td>3.3</td>
<td>Estimation Results</td>
<td>3.8</td>
</tr>
<tr>
<td>4.0</td>
<td>Conclusions and Recommendations</td>
<td>4.1</td>
</tr>
<tr>
<td>4.1</td>
<td>RELY I</td>
<td>4.1</td>
</tr>
<tr>
<td>4.2</td>
<td>Data Requirements</td>
<td>4.1</td>
</tr>
<tr>
<td>4.3</td>
<td>Recommendations</td>
<td>4.2</td>
</tr>
<tr>
<td>REFERENCES</td>
<td></td>
<td>5.1</td>
</tr>
<tr>
<td>APPENDIX A</td>
<td>MAIN AND SUBROUTINE DESCRIPTIONS</td>
<td>A.1</td>
</tr>
<tr>
<td>1.</td>
<td>MAIN and Subroutine Diagram</td>
<td>A.1</td>
</tr>
<tr>
<td>2.</td>
<td>MAIN</td>
<td>A.2</td>
</tr>
<tr>
<td>3.</td>
<td>Subroutine RNDTA</td>
<td>A.3</td>
</tr>
<tr>
<td>4.</td>
<td>Subroutine POISS</td>
<td>A.4</td>
</tr>
<tr>
<td>5.</td>
<td>Subroutine RANDEX</td>
<td>A.5</td>
</tr>
</tbody>
</table>
6. Subroutine KOST ........................................... A.6
7. Subroutine EIGMIN ......................................... A.9
8. Subroutine TRIDMX ........................................ A.10
9. Subroutine EIGVAL ......................................... A.11
10. Subroutine EIGVEC ........................................ A.12
11. Subroutine MARQ ........................................ A.13

APPENDIX B: RELY I GLOSSARY AND INDEX .................. B.1
APPENDIX C: PROCEDURE FOR OPERATION OF RELY I ........ C.1
APPENDIX D: RELY I PROGRAM LISTING AND SAMPLE OUTPUT . . . D.1
<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Number of errors in program as a function of time</td>
<td>2.2</td>
</tr>
<tr>
<td>2</td>
<td>Block diagram representation of the proposed model for the error-detection and the error-correction processes.</td>
<td>2.6</td>
</tr>
<tr>
<td>3</td>
<td>Simplified model.</td>
<td>2.7</td>
</tr>
<tr>
<td>4</td>
<td>Cumulative detected errors $n_d(k)$ for $0 &lt; \theta &lt; 0.5$.</td>
<td>2.11</td>
</tr>
<tr>
<td>5</td>
<td>Cumulative corrected errors $n_c(k)$ for $0 &lt; \theta &lt; 0.5$.</td>
<td>2.12</td>
</tr>
<tr>
<td>6</td>
<td>Flow diagram of the estimation algorithm.</td>
<td>3.7</td>
</tr>
<tr>
<td>7</td>
<td>Histograms of reliability parameters for Example I.</td>
<td>3.13</td>
</tr>
<tr>
<td>8</td>
<td>Histograms of reliability parameters for Example II.</td>
<td>3.14</td>
</tr>
<tr>
<td>9</td>
<td>Histograms of reliability parameters for Example III.</td>
<td>3.15</td>
</tr>
<tr>
<td>10</td>
<td>Histograms of reliability parameters for Example IV.</td>
<td>3.16</td>
</tr>
<tr>
<td>11</td>
<td>Tested software in only one version at a time, identical to neighboring versions except for the changes counted in $\Delta_c$ at the respective boundaries.</td>
<td>4.3</td>
</tr>
<tr>
<td>A.1</td>
<td>RELY I subroutine diagram</td>
<td>A.1</td>
</tr>
</tbody>
</table>
LIST OF TABLES

| 1. Estimation results.                      | 3.12 |

vi
The increased importance of software for embedded avionics systems has led to an increasing desire to insure that avionics software meets very strict reliability and quality goals. However, a significant problem in assuring such goals are met is the inability of Government personnel to accurately predict the reliability of an avionics software development project. This problem has been expressed at several Government and industry sponsored conferences, as well as in documents such as the Joint Logistics Commanders Software Reliability Working Group Report (November 1975) and the Joint Logistics Commanders Software Quality Management workshop Report (July 1979). As a result, efforts have been initiated to develop and validate mathematical models for predicting the reliability and error content of a software system. However, models developed to date have not adequately addressed the unique features of avionics software developments.

This effort was initiated in response to the need for developing software reliability prediction models applicable to avionics software developments, and fits into the goals of RADC TAC No. 5, Software Cost Reduction, in the sub thrust of Software Quality (Software Modeling). This report summarizes the development of a mathematical model for predicting the reliability and mean-time-to-failure of a software development under the assumptions of incomplete information available on error correction, and discrete versions of the software being developed. The report also describes the modified nonlinear search algorithm developed for finding model parameters and an accompanying...
computer program for operating the model. The importance of this model development is that the assumptions underlying this model more closely reflect the actual avionics software development process than prior model developments.

The theory and model algorithm developed under this effort will lead to much needed predictive measures for use by software managers of avionics software developments in adequately tracking those developments in terms of reliability and mean-time-to-failure objectives. More importantly, the measures developed under this effort will be applicable to current avionics software developments and thus help to produce the high quality, low cost avionics software needed for today's aircraft.

Alan N. Sukert
Project Engineer
1.0 Introduction

1.1 Problem Statement

As the cost and complexity of computer software continue to increase, there is a growing need for accurate determination of software reliability. Before a software package is put into operation, there is a testing period during which errors are detected and corrected. The problem with which we are concerned is the estimation of certain reliability parameters from the error data generated during the test phase. Specifically, we wish to estimate the number of errors remaining in the software package at any time, and the mean time to failure (MTTF). Accurate determination of these parameters could reduce the cost associated with excessive testing, and could increase the confidence with which the package is used.

In order to estimate software reliability, it is necessary to develop an appropriate model describing the error detection and correction processes, and to develop procedures for estimating the parameters of this model from observed error data. Our intention is to generalize certain models which have previously been used for this purpose in order to depict more accurately an actual testing environment. In addition, we will consider a somewhat different approach to the estimation of the parameters of this generalized model.

1.2 Previous Work

A substantial body of work now exists on the application of statistical modeling and estimation techniques to the determination of
software reliability. We make no attempt to describe all this work, but rather restrict ourselves to those efforts which are directly related to our own. For a comprehensive review and bibliography, see [1] or [2].

One of the most widely-used error models was developed by Jelinski and Moranda [3]. A similar model has been considered by Shooman [4] and others. The assumptions about the error-detection and error-correction processes which underlie this model are the following:

(a) The error-detection process is a Poisson process whose detection rate is constant between error detections.
(b) The error-detection rate at the time prior to the detection of the \(i\)th error is a function of \(i\); it is denoted by \(z_i\). It is commonly assumed that \(z_i\) is proportional to the number of errors in the program at detection time. This can be written as:

\[
z_i = \phi (N_0 - i + 1)
\]  
(1.1)

where \(N_0\) is the initial number of errors and \(\phi\) is a positive constant. An alternative assumption is that the detection rate forms a geometric progression

\[
z_i = \lambda a^i
\]  
(1.2)

with both \(\lambda\) and \(a\) being positive constants. It should be noted that the main justification for (1.2) is the improved
convergence of the resulting estimator equations [5].

(c) Error detection is followed by an immediate correction.
Consequently, upon detection of the $i^{th}$ error, the number of remaining errors drops to $(N_0 - i)$.

(d) The debugging process is perfect and no new errors are generated by the correction process.

These assumptions, although restrictive, were initially adopted by most of the researchers in the field. The estimation of the reliability parameters was based on the above assumptions, and employed the maximum likelihood (ML, criterion to derive the best estimates.

It is realized now that the assumptions given above are quite restrictive and unrealistic in most cases, and steps have been taken to make the model more realistic. The model assumptions have been changed to comply more closely with the real process.

Goel [6] has considered a nonideal debugging process in which the probability of correcting an error is $p$. Based on this assumption, an analysis of the resulting model is performed. Further generalization is suggested by Shooman [7], who modified both assumptions c and d above concerning the error-correction process. According to the modified model of Shooman, the correction process does not necessarily proceed identically to the detection process, and new errors may be introduced. Denote by $r_d(t)$, $r_c(t)$, and $r_g(t)$ the rates of error detection, correction, and new error generation, respectively. The models suggested by Shooman assume different relationships between these rates. The main models are:
Model 1

\[ r_c(t) = \beta r_d(t) \]  \hspace{1cm} (1.3)

\[ r_g(t) = a r_c(t) \]  \hspace{1cm} (1.4)

and

Model 2

\[ r_c(t) = b r_d(t) \]  \hspace{1cm} (1.5)

\[ r_g(t) = a n(t) r_d(t) \]  \hspace{1cm} (1.6)

where \( n(t) \) represents the number of errors in the program.

These models and others have been studied by Shooman, and the results are described [7].

Another generalization of the original model concerns the assumption that the corrections are implemented continuously. This is not consistent with actual practice in which a program is replaced by a newer version at discrete times. Between the replacement times, the program undergoing the test is the same and the number of errors in it is constant. A possible solution for this discrepancy is that rediscovery of errors should not be counted. However, this requires the analysis of the source of errors in order to determine whether the error sources are the same, and this is not always practical. A modified model in which this generalization was implemented was discussed by
Tal [5] and by Sukert [2], and estimator equations for use with this model were developed.

These generalizations, along with some additional ones, will be incorporated into a new model. The new model, we believe, more accurately describes an actual testing environment. We will first discuss the behavior of this model as a function of its parameters under the simplifying assumption that the error processes are deterministic rather than random.

After presenting certain results for deterministic processes, we will then show results using simulated random error data. We have developed a least-squares search procedure for estimating the model parameters, and will discuss its convergence behavior. Recommendations are made toward increased utility, and toward closer coupling of the algorithm to information in real test data.

The true test of the usefulness of the model will lie in its ability to describe real software tests. Thus, there remains for subsequent work the application of the model to enough real cases to draw conclusions concerning validity.

One of the difficulties encountered by researchers in the past has been the inadequacy, incompleteness, and ambiguity of available test data. We found some of these same problems with the data available to us during this work. Hence, we include comments regarding data requirements.
2.0 Development and Analysis of the Model

2.1 Assumptions

In order to develop a generalized model to describe the error detection and correction processes, we make the following assumptions:

(a) The error detection process is a Poisson process whose average rate of occurrence is proportional at any time to the number of errors present in the software package. Denoting the number of errors present at time t by $N(t)$ and the average error occurrence rate by $r_d(t)$, we have

$$r_d(t) = \phi N(t) \quad (2.1)$$

where $\phi$ is a fixed constant of proportionality.

(b) No attempt is made to correct detected errors at the time of detection. Instead, a new and corrected version of the program is provided to the testing group at discrete ("tape replacement") times $t_1$, $t_2$, ..., $t_j$, .... Thus, the number of errors present in the program at time $t$, $t_j \leq t < t_{j+1}$, is constant and equal to $N(t_j)$. This is illustrated in Fig. 1.

(c) Of the detected errors reported to the correcting group, some are corrected and some are not. In addition, new errors are generated. Denote the cumulative number of errors corrected to time t in the program being tested by $N_c(t)$, and the cumulative number of newly-generated errors by $N_g(t)$. Both $N_c$ and $N_g$ are piecewise constant because of the assumption
Fig. 1. Number of errors in program as a function of time.
that a new version of the program is provided only at the
discrete times \(t_1, t_2, \ldots\). A key feature of our model is
that many errors may be detected, corrected, and generated
between update times. At any time \(t\) during testing, we
have

\[ N(t) = N_0 - N_c(t) + N_g(t) \quad (2.2) \]

where \(N_0\) is the initial number of errors in the program.

(d) The error correction rate \(r_c(t)\) depends on both the error
detection rate \(r_d(t)\) and the error backlog \(N_b(t)\), where

\[ N_b(t) = N_d(t) - N_c(t). \quad (2.3) \]

For simplicity, we assume a linear relationship

\[ r_c(t) = \alpha r_d(t) + \beta N_b(t). \quad (2.4) \]

The addition of the second term in (2.4) represents a
generalization of the model of Shooman [7].

(e) The rate of generation of new errors is proportional to the
error-correction rate:

\[ r_g(t) = \gamma r_c(t). \quad (2.5) \]

(f) The error-detection process \(N_d(t)\) is precisely known, but
the error-correction process \(N_c(t)\) is unknown. This ap-
ppears to be a realistic assumption in view of the way
error correction is actually performed. The error generation process is also unknown.

In the above we tacitly equate software "failure" to coding "faults". In effect, we include in $\alpha$ and $\rho$ the proportionality between the two, and call them both "errors".

2.2 The Model

The model which we develop is actually a deterministic model which relates the expected values of the various random processes involved. The required connection between the observed sample functions of the random processes involved and the deterministic model is established by means of an estimation algorithm which operates on the observed data to estimate model parameters. The deterministic model will be described first, followed by a discussion of the estimation procedure.

Taking expected values of (2.1)-(2.4) yields the equations

$$r_d(t) = \phi n(t),$$

$$r_c(t) = \alpha r_d(t) + \beta n_b(t),$$

$$n(t) = N_0 - n_c(t) + n_g(t),$$

$$n_b(t) = n_d(t) - n_c(t),$$

$$n_g(t) = \gamma n_c(t),$$
where a lower-case \( n \) denotes the expected value of the process represented by the corresponding upper-case \( N \).

It follows from the relationship between \( r_d(t) \) and \( n_d(t) \) that

\[
    n_d(t) = \int_0^t r_d(t) \, du. \tag{2.11}
\]

Similarly,

\[
    n_c(t) = \int_0^t r_c(u) \, du. \tag{2.12}
\]

The model represented by the above equations can be viewed as a linear system with sampling and feedback as shown in Fig. 2. Our problem is now one of system identification: Given \( N_d(t) \), estimate the parameters of the system shown in Fig. 2. Revisions to the software are applied at time instants \( t_k \), between which times \( n(t) \) remains constant. The system therefore is treated as a discrete-time system. We employ the usual notation \( k \) in place of the argument \( t_k \).

The four system equations (2.6-2.9) can be reduced to two:

\[
    r_d(k) = \phi \left[ N_0 - (1 - \gamma) n_c(k) \right] \tag{2.13}
\]

\[
    r_c(k) = \alpha \phi \left[ N_0 - (1 - \gamma) n_c(k) \right] + \beta \left[ n_d(k) - n_c(k) \right] \tag{2.14}
\]
Fig. 2. Block diagram representation of the proposed model for the error-detection and the error-correction processes.

and the number of parameters reduced to four:

\[ r_d(k) = \phi_a [N_a - n_c(k)] \]  \hspace{1cm} (2.15)

\[ r_c(k) = \alpha \phi_a [N_a - n_c(k)] + \beta [n_d(k) - n_c(k)] \]  \hspace{1cm} (2.16)

where

\[ \phi_a = (1 - \gamma)\phi, \hspace{1cm} N_a = N_0/(1 - \gamma). \]  \hspace{1cm} (2.17)

The application of Laplace transform techniques and some algebraic manipulation similarly lead to the equivalent block diagram shown in
Fig. 3. Simplified model.

Fig. 3. The identification problem reduces to the estimation of the four parameters $N_a$, $\phi$, $\alpha$, and $\beta$. Note that $N_a$ is the sum of the initial errors $N_o$ and all errors which are subsequently generated during the correction process. Note further that the ultimately sought reliability factor, mean-time-to-failure, is:

$$MTTF = \frac{1}{r_d(k)} = \frac{1}{\phi a_n(k)} = \frac{1}{\phi a[N_a - n_c(k)].} \quad (2.18)$$

Defining the discrete state

It is noted that the dynamics of the system can be studied using the even simpler nondimensionalized three-parameter system, using ($\phi T$), ($\beta T$), and ($n/N_a$), with unit step input.
and using the usual approximation, which in our case is exact,

\[ \hat{r}(k) = \frac{\hat{n}(k+1) - \hat{n}(k)}{T(k)}, \quad T(k) = t(k+1) - t(k) \]  \hspace{1cm} (2.20)

the model becomes

\[ \hat{n}(k+1) = Ln(k) + N_a \phi_a B \]  \hspace{1cm} (2.21)

where

\[ L = \begin{pmatrix} 1 & -\phi_a T(k) \\ \beta T(k) & 1 - T(k)[\beta + \alpha \phi_a] \end{pmatrix}, \quad B = \begin{pmatrix} T(k) \\ \alpha T(k) \end{pmatrix} \]  \hspace{1cm} (2.22)

When tape replacement occurs at uniform time intervals, $T$ is constant over $k$ and the system is seen to be stationary, and the equations can be solved immediately by successive evaluation:

\[ \hat{n}(1) = N_a \phi_a B, \quad \hat{n}(0) = 0 \]
\[ \hat{n}(2) = N_a \phi_a (L + 1)B \]
\[ \hat{n}(3) = N_a \phi_a \left( L^2 + L + 1 \right)B \]
\[ \vdots \]
\[ \hat{n}(k) = N_a \phi_a \sum_{j=0}^{k-1} L^j B \]
and applying the familiar procedure for the geometric sum,
\[ L_n(k) - n(k) = N_a \phi_a \left( L^k B - B \right) \]
which gives for the state at the \( k^{th} \) tape replacement time,
\[ \hat{n}(k) = N_a \phi_a (L - 1)^{-1} \left( L^k - 1 \right) B \]
(2.23)
The increment \( \delta(k) = \hat{n}(k) - \hat{n}(k-1) \) at the \( k^{th} \) tape replacement time is given by:
\[ \delta(k) = N_a \phi_a (L - 1)^{-1} \left( L^k - L^{k-1} \right) B \]
\[ = N_a \phi_a L^{k-1} B \]
(2.24)

2.3 Model Behavior

Note from the discrete state equations above that the parameter \( N_a \) is simply a scale factor on the state \( \hat{n} \). Recall also that the initial slope of \( n_d(k) \) is \( N_a \phi_a \), and that of \( n_c(k) \) is \( \alpha N_a \phi_a \), regardless of the value of \( \beta \). Furthermore, for \( \beta = 0 \), \( n_d(k) \) and \( n_c(k) \) maintain the constant ratio \( n_c(k)/n_d(k) = \alpha < 1 \), and, of course, coincide as \( \alpha \to 1 \).

The effect of \( \beta > 0 \) is to increase the error correction rate, and therefore increase \( n_c(k) \), especially for the larger differences \( n_d(k) - n_c(k) \) (backlog) which tend to occur later in the test program. The resulting decrease in remaining errors \( N_a - n_c(k) \) causes the detected
error curve \( n_d(k) \) to be bent downward. Thus the effect of \( \beta \) is to draw the two curves together. Figures 4 and 5 display this effect for \( 0 \leq \beta \leq 0.5 \). The "bending" of the curves due to \( \beta \), together with the effects of the discrete nature of the model, are expected to occur in real data.

2.4 The Data Simulator

RELY I contains a data simulator for the purposes of study and experimentation. The simulator is an optional source of input data to the estimator (see Appendix C). The simulator reads from input cards the nominal parameter values, \( \phi, \beta, \phi_a, N_a \), the time interval \( T \), the number of test intervals \( K \), and an input initial random number (RRR), and computes the associated software test history \( \Delta_d(k) \). The random number RRR is changed by the investigator when he wishes a different sample of the random data set \( \Delta_d(k) \) (see Appendix A, RNDTA, RANDEX).
Fig. 4. Cumulative detected errors $n_d(k)$ for $0 < \beta < 0.5$. 

Parameters:
- $a = 0.7$
- $\beta = 0.5$
- $\phi_a = 0.02$
- $N_a = 200$
- $T = 1.5$

$200 - 175 - 150 - 125 - 100 - 75 - 50 - 25 - 0$

$n_d$

$7.5 - 15.0 - 22.5 - 30.0 - 37.5 - 45.0 - 52.5 - 60.0 - 67.5 - 75.0$
Fig. 5. Cumulative corrected errors $n_c(k)$ for $0 \leq \beta \leq 0.5$. 

\[
\begin{align*}
\alpha &= 0.7 \\
\beta &= 0.5 \\
\phi_a &= 0.02 \\
N_a &= 200 \\
T &= 1.5
\end{align*}
\]
3.0 The Estimation Algorithm

3.1 The Estimation Problem and Method

Having described the model, we turn to the parameter estimation algorithm which estimates the values of the model parameters corresponding to a given set of real test data. The resulting parameter estimates provide the reliability information sought regarding the tested software package.

Though the model is linear in the sense that the equations are linear in the state $\hat{n}$, the model equations are nevertheless nonlinear in the parameters $\hat{\xi}$ (i.e., in $a, b, \phi_a, N_a$). Determining the parameter values corresponding to a given set of real test data $\tilde{d}(k)$ is then a nonlinear estimation problem.

Nonlinear parameter estimation methods, in general, are iterative procedures in which the estimate is approached from some initial guess for the parameter values, in steps which successively decrease a cost functional $J$. Since our purpose is to determine the parameter values $\hat{\xi}$ for which the solution $\delta_d(k)$ of the model equations approximates the measured function (or sequence) $\Delta_d(k)$, we choose the cost functional $J$ to be the sum of the squares of the residuals, $\delta_d(k) - \Delta_d(k)$, viz.,

$$ J = \sum_{k=1}^{K} [\delta_d(k) - \Delta_d(k)]^2 \quad (3.1) $$

Minimizing this cost functional, then, minimizes the difference between the observed function $\Delta_d(k)$ and its expected value $\delta_d(k)$ in the least squares sense.
Of the numerous methods described in the literature, both direct search methods (Fletcher [9]) and gradient methods (Bard [8]), the gradient methods are generally preferred when the computation of the derivatives of $J$ is not prohibitive. Gradient methods, in principle, step from one point $\mathbf{\delta}_i$ in parameter space to the next $\mathbf{\delta}_{i+1}$ according to

$$
\mathbf{\delta}_{i+1} = \mathbf{\delta}_i - \tau_i R_i \mathbf{\delta}_i
$$

(3.2)

where $\mathbf{\delta}_i$ is the gradient of $J$ evaluated at $\mathbf{\delta}_i$, $R_i$ is some matrix which operates on the gradient to define the $i$th step direction $R_i \mathbf{\delta}_i$, and $\tau_i$ is a scalar which determines the step size. The methods differ in what each employs for $R_i$, i.e., in the step direction each takes relative to the gradient. The method of steepest descent, for example, uses the identity matrix for $R_i$, so that the step direction is opposite to that of the gradient. This is "the steepest way down" locally but tends to be less efficient and therefore less desirable than methods which use second order information about the surface $J(\mathbf{\delta})$.

The Newton-Raphson method uses for $R_i$ the inverse of the Hessian, the matrix of the second partial derivatives,

$$
H_{mn} = \frac{\partial^2 J}{\partial \mathbf{\delta}_m \partial \mathbf{\delta}_n},
$$

(3.3)

of the cost functional.

Notice that the Taylor series expansion of $J$ to second order terms,
\[ J = J_i + \nabla^T_i \left( \bar{\mathbf{g}} - \mathbf{g}_i \right) + \frac{1}{2} \left( \mathbf{g} - \mathbf{g}_i \right)^T H_i \left( \mathbf{g} - \mathbf{g}_i \right) \]

has an extremum,

\[ \frac{\partial J}{\partial \mathbf{g}_i} = \mathbf{g}_i + H_i \left( \mathbf{g} - \mathbf{g}_i \right) = 0, \]

at

\[ \mathbf{g} = \mathbf{g}_i - H_i^{-1} \mathbf{g}_i \quad (H_i \text{ nonsingular}) \]

so if \( R_i = H_i^{-1}, \rho_i = 1, \) and \( J \) is quadratic, then \( \mathbf{g}_{i+1} \) coincides with the extremum. The Newton-Raphson method in this case converges in a single iteration. This method is quite efficient even for nonquadratic \( J \), but only if \( H_i \) is positive definite. This latter condition is the principal weakness of the method. The Marquardt method meets this weakness by guaranteeing positive definiteness in \( R_i \) by adding to \( H_i \) (or to some convenient approximation of \( H_i \)) a variable amount of a positive definite matrix \( C_i^2 \):

\[ R_i = \left( H_i + \lambda_i C_i^2 \right)^{-1} \]

and suggests \( C_i^2 \) be a matrix of the diagonal elements of \( H_i \), viz.,

\[ C_{i,ss}^2 = |H_{i,ss}|. \]

For sufficiently large \( \lambda_i \), \( R_i \) then is positive definite, even when \( H_i \) is not. The Marquardt method behaves as the Newton-Raphson for small
\[ \lambda_i \text{, but where larger } \lambda_i \text{ is necessary it steps nevertheless in some acceptable (downward) direction. A step is said to be acceptable if it decreases } J. \text{ If } \lambda_i \text{ is large and } H_i \text{ has low condition number (eigenvalues of near-equal magnitude), the method approximates that of steepest descent. The Marquardt method varies from step to step according to } \lambda_i, \text{ between the behavior of the Newton-Raphson method and that of steepest descent.} \]

3.2 Description of the Search Algorithm

The program, RELY I, uses the above Marquardt R., i.e.,

\[ \theta_{i+1} = \theta_i - \tau_i (H_i + \lambda_i C_i^2)^{-1} g_i \]  \hspace{1cm} (3.6)

and selects \( \tau_i \) or \( \lambda_i \) from step to step according to the procedure described below. Essentially the program progresses in one or the other of two modes. In mode A, \( \lambda_i \) is fixed while the largest \( \tau_i \) (0.0001 < \( \tau_i \) < 1) is sought which results in an acceptable step size. If the sought \( \tau_i \) is found, the program continues in mode A preferring smaller and smaller values of \( \lambda \) (more nearly Newton-Raphson). If at any point insufficient progress is being made in mode A, the routine moves to mode B, in which \( \tau_i \) is initially fixed, and \( \lambda_i \) is successively increased until an acceptable step direction is reached. In mode B, when a sufficiently large \( \lambda_i \) is reached, then the program steps in that direction until \( J \) begins to increase, or until, for large \( J \), \( J \) has decreased more than 10 percent, at which point the routine returns to mode A. In short, when progress is slow in mode A, the program resorts to mode B to move to a different
"locality". "Progress" in mode B is deliberately restricted for large J due to experience which indicates that mode B for large J tends to settle into local minima. The program terminates when J becomes less than a predetermined value (ERR), or upon a time limit for machine computation.

More specifically, the estimator proceeds as follows: Given initial guess \( \theta_1 \) and \( \lambda_1 = 1, i = 0 \):

Mode A

1. Compute cost \( J_i \) and step direction \( R_i \).

2. If \( J_i < \text{ERR} \) terminate, otherwise determine an acceptable step size in the following way:
   a. Compute \( \tau_1 \) such that twice the associated step causes \( \beta = 5 \); i.e.,
      \[
      \tau_1 = \frac{(5 - \beta_i)}{2|R_i \theta_i|}
      \]
   b. If such a step causes \( \phi_a > 0.2 \), choose instead
      \[
      \tau_1 = \frac{(0.2 - \phi_{a1})}{2|R_i \theta_i|}
      \]
   c. If the resulting \( \tau_1 > 1 \), set \( \tau = 1 \).
   d. If the resulting \( \tau_1 < 0.0001 \), jump to mode B.
   e. Limit \( 0 \leq \beta \leq 10 \). Compute \( J_{i+1} \).
   f. If \( J_{i+1} \geq J_i \), jump to item 4 below.

3. Accept \( \theta_{i+1} \) and reduce \( \lambda \); i.e.,
   a. Set \( \theta_i = \theta_{i+1}, \lambda_i = \lambda_i / 10 \).

- 3.5 -
b. If J has decreased less than 1 percent in more than five iterations (reductions of λ in item 3.a) since passing through mode B, jump to mode B. Otherwise continue in mode A (jump to item 1 above).

4. Reduce $\tau_1$ by a factor of 10. If the resulting $J_{i+1} < J_i$ jump to item 3 above, otherwise repeat item 4 above up to five times (according to counter INDEX). If J does not decrease with five reductions of $\tau_1$, jump to mode B. If $J_{i+1} < \text{ERR}$, terminate.

**Mode B**

5. Fix $\tau = 0.1$, set $\lambda = 0.01$.

6. Increase $\lambda$ by a factor of 10, increment the count ICLAM, determine the corresponding step direction $\left( H_1 + \lambda C_1^2 \right)^{-1} g_1$, parameter set $\theta_{i+1}$, and $J_{i+1}$. If $J_{i+1} > J_i$, repeat item 6.

7. Accept $\theta_{i+1}$ (i.e., set $\theta_i = \theta_{i+1}$) and set $J_\lambda = J_{i+1}$.

8. Increase $\tau_1$ by a factor of $5^k$, $k = \text{ICLAM}$.

9. Try $\bar{\theta}_{i+1} = \theta_i - \tau_1 \left( H_1 + \lambda C_1^2 \right)^{-1} g_1$, if $J_{i+1} > J_i$ or if $J_i > 50$ and $J_{i+1}/J_\lambda < 0.9$, accept $\bar{\theta}_i$ and return to item 1 above, otherwise accept $\theta_{i+1} + \theta_i$ and repeat item 8.

The algorithm is depicted in the flow diagram of Fig. 6.
Read error data, $N$, ERR.
Read initial guess for $\theta_1$.

$\lambda_1 = 1$

Compute $J_1$

If $J_1 < ERR$ Yes

No

Compute $(H_1 + \lambda_1 C_1^T)_{11}^{-1} g_1$

Select $\tau_1$ as in step 2

$\tau_1 \leq .0001$

Yes

No

$\theta_{i+1} = \theta_i - \tau_i (H_i + \lambda_i C_i^T)_{11}^{-1} g_i$

$J_{i+1} \leq J_i$

No

Yes

$\theta_{i+1} = \theta_{i+1}$

$\lambda_i = \lambda_i + 1$

$J_{i+1} \leq J_i$

No

Yes

$\theta_i = \theta_i - \tau_i (H_i + \lambda_i C_i^T)_{11}^{-1} g_i$

$J_{i+1} \leq J_i$

No

Yes

$\theta_i = \theta_i + 1$

$J_{i+1}$

No

Yes

$\theta_i = \theta_i - \tau_i (H_i + \lambda_i C_i^T)_{11}^{-1} g_i$

$J_{i+1} > J_i$

No

Yes

$\theta_i = \theta_i - \tau_i (H_i + \lambda_i C_i^T)_{11}^{-1} g_i$

$J_{i+1} > J_i$

or

$J_{i+1} \geq 50$ and $J_{i+1}/\text{PCOST} < .9$

No

$\theta_i = \theta_i - \tau_i (H_i + \lambda_i C_i^T)_{11}^{-1} g_i$

$J_{i+1} \leq J_i$

$J_{i+1} \leq J_i$

Stop

Has step size been reduced five times?

Yes

No

$\tau_i = \tau_i / 10$

Select $\tau_i$ as in step 2

$\tau_1 \leq .0001$

Yes

No

$\theta_{i+1} = \theta_i - \tau_i (H_i + \lambda_i C_i^T)_{11}^{-1} g_i$

$J_{i+1} \leq J_i$

No

Yes

$\theta_i = \theta_i - \tau_i (H_i + \lambda_i C_i^T)_{11}^{-1} g_i$

$J_{i+1} \leq J_i$

No

Yes

$\theta_i = \theta_i - \tau_i (H_i + \lambda_i C_i^T)_{11}^{-1} g_i$

$J_{i+1} \leq J_i$

No

Yes

$\theta_i = \theta_i - \tau_i (H_i + \lambda_i C_i^T)_{11}^{-1} g_i$

$J_{i+1} \leq J_i$

No

Yes

$\theta_i = \theta_i - \tau_i (H_i + \lambda_i C_i^T)_{11}^{-1} g_i$

$J_{i+1} \leq J_i$

No

Yes

$\theta_i = \theta_i - \tau_i (H_i + \lambda_i C_i^T)_{11}^{-1} g_i$

$J_{i+1} \leq J_i$

Stop

Fig. 6. Flow diagram of the estimation algorithm.

- 3.7 -
Experience during development of RELY I proved $\alpha$ to be insufficiently independent of the other parameters to warrant a fourth degree of freedom in the search process. The computer program therefore was modified to accept a priori the estimate of $\alpha$, and to search in three dimensions for the values of $\beta$, $\phi_a$, and $N_a$. From these the estimated number of remaining errors,

$$n_a = n_a(k) = N_a - n_c(k), \quad k = 1, 2, 3, \ldots K$$

and mean time to failure

$$\text{MTTF} = \frac{1}{\phi_a n_a}$$

are computed. The latter two computed quantities, the sought software reliability factors, were found to be essentially insensitive to reasonable a priori estimates of $\alpha$. Results below include cases of correct and incorrect fixed $\alpha$.

3.3 Estimation Results

Results are tabulated and displayed in histograms below for several simulated random data examples. Examples I and II differ in the selection of $K$ and $T$ to vary the number of errors, $N_R$, remaining in the software. Example I uses test interval length $T = 1.5$ and 60 intervals which leaves about 30 remaining of the initial 200 software errors. Example II uses longer test intervals, $T = 8$, and fewer intervals, $K = 20$, to leave about 5 errors remaining. Example III corresponds to a
larger software system having a considerably larger number of initial errors ($N_a = 1000$), smaller error detection rate ($\phi_a = 0.01$), but a better correction rate ($\alpha = 0.8$), and the longer test intervals ($T = 8.0$). Finally, the fourth example demonstrates the insensitivity to the fixed value of $\alpha$. Example IV essentially is Example I with $\alpha$ fixed at 0.5 instead of the "true" value, 0.7.

Before examining the results, we anticipate the nature of the distributions by analytically determining the mean and variance for the simple single interval ($K = 1$) case. Let the observed number of detected errors $N_d$ be Poisson with mean and variance $pN_a$, where $p = \phi_a T$. We minimize the squared error $J$,

$$J = (pN_a - N_d)^2$$

$$\frac{\partial J}{\partial N_a} = 2p (pN_a - N_d) = 0$$

to obtain an estimate

$$\hat{N}_a = \frac{N_d}{p}$$

which has mean

$$E\left\{\hat{N}_a\right\} = \frac{E\left\{N_d\right\}}{p} = N_a$$

and variance

$$-3.9-$$
\[ E \left\{ \left( \hat{N}_a - N_a \right)^2 \right\} = E \left\{ \left( \hat{N}_a \right)^2 \right\} - 2 E \left\{ \hat{N}_a N_a \right\} + E \left\{ N_a^2 \right\} \]

\[ = E \left\{ \left( \hat{N}_d \right)^2 \right\} \left( 2 N_a E \left\{ \hat{N}_a \right\} + N_a^2 \right) \]

\[ = \frac{\text{var} \left( N_d \right) + \hat{N}_d^2}{\rho^2} - \frac{N_a^2}{\rho^2} \]

\[ = \frac{\rho N_a + \rho^2 \hat{N}_a^2 - \rho^2 N_a^2}{\rho^2} \]

\[ = \frac{N_a}{\rho} \]

The number of remaining errors, \( N_R = N_a - N_d \) is estimated

\[ \hat{N}_R = \hat{N}_a - N_d = N_d \left( \frac{1 - \rho}{\rho} \right) \]

with unbiased mean

\[ E \left\{ \hat{N}_R \right\} = E \left\{ \hat{N}_a - N_d \right\} = N_a - N_d \]

and with root mean squared difference from its true value,

\[-3.10-\]
\[ E \left\{ \left[ N_d \left( \frac{1-\rho}{\rho} \right) - (N_a - N_d) \right]^2 \right\}^{1/2} = E \left\{ \left( \frac{N_a - N_d}{\rho} \right)^2 \right\}^{1/2} \]

\[ = \left[ \frac{2N_a}{\rho} E \left\{ N_d \right\} + \frac{E\left\{ N_d^2 \right\}}{\rho^2} \right]^{1/2} = \left[ N_a^2 - 2N_a^2 + \frac{\rho N_a^2 + \rho N_a}{\rho^2} \right]^{1/2} = \sqrt{\frac{N_a}{\rho}} \]

Notice that the value of the latter quantity corresponding to:

a. Example I: Let \( N_R = N_a - \rho N_a \), or \( (\rho = 1 - 31/200 = .845) \) is

\[ \sqrt{\frac{200}{.845}} = 15 \]

b. Example II: \( (\rho = 1 - 5/200 = .975) \) is

\[ \sqrt{\frac{200}{.975}} = 14 \]

c. Example III: \( (\rho = 1 - 185/1000 = .815) \) is

\[ \sqrt{\frac{N_a}{.815}} = \sqrt{\frac{1000}{.815}} = 35 \]

One would expect these values to approximate the standard deviations \( \sigma(N_R) \) for the respective multiple-interval cases (though perhaps with less validity when \( N_R/N_a \) is small). The \( \sigma(N_R) \) indicated below for the four examples then are of the magnitude to be expected. Table 1 lists numerical information from the four examples. Examination of results from the four simulated examples indicates that the estimator produces reasonable estimates of the reliability parameters \( N_R \) and MTTF.
Table 1. Estimation results.

<table>
<thead>
<tr>
<th>Item</th>
<th>Example: I</th>
<th>II</th>
<th>III</th>
<th>IV</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;True&quot; Values</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$\alpha$</td>
<td>.700</td>
<td>.700</td>
<td>.800</td>
<td>.700</td>
</tr>
<tr>
<td>$\beta$</td>
<td>.100</td>
<td>.100</td>
<td>.100</td>
<td>.100</td>
</tr>
<tr>
<td>$T_\alpha$</td>
<td>.020</td>
<td>.020</td>
<td>.010</td>
<td>.020</td>
</tr>
<tr>
<td>$N_a$</td>
<td>200</td>
<td>200</td>
<td>1000</td>
<td>200</td>
</tr>
<tr>
<td>$N_R$</td>
<td>31.0</td>
<td>4.93</td>
<td>185</td>
<td>30.6</td>
</tr>
<tr>
<td>MTTF</td>
<td>1.61</td>
<td>10.1</td>
<td>0.538</td>
<td>1.64</td>
</tr>
<tr>
<td>Time Interval</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$T$</td>
<td>1.5</td>
<td>8.0</td>
<td>8.0</td>
<td>1.5</td>
</tr>
<tr>
<td>Number of Intervals</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$K$</td>
<td>60.0</td>
<td>20.0</td>
<td>20.0</td>
<td>60.0</td>
</tr>
<tr>
<td>A Priori $\alpha$</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$\alpha'$</td>
<td>.700</td>
<td>.700</td>
<td>.800</td>
<td>.500</td>
</tr>
<tr>
<td>Initial Guess</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$\beta$</td>
<td>.300</td>
<td>.300</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$T_\alpha$</td>
<td>.010</td>
<td>.010</td>
<td>.02</td>
<td>.010</td>
</tr>
<tr>
<td>$N_a$</td>
<td>300</td>
<td>300</td>
<td>1500</td>
<td>300</td>
</tr>
<tr>
<td>Estimated Values</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$N_R$</td>
<td>26.3</td>
<td>5.03</td>
<td>195</td>
<td>29.7</td>
</tr>
<tr>
<td>$\sigma (N_R)$</td>
<td>11.2</td>
<td>3.07</td>
<td>40.4</td>
<td>10.8</td>
</tr>
<tr>
<td>MTTF</td>
<td>2.19</td>
<td>12.6</td>
<td>.543</td>
<td>1.75</td>
</tr>
<tr>
<td>$\sigma$(MTTF)</td>
<td>.65</td>
<td>7.43</td>
<td>.093</td>
<td>.33</td>
</tr>
</tbody>
</table>
Fig. 7. Histograms of reliability parameters for Example 1.
Fig. 8. Histograms of reliability parameters for Example II.
Fig. 9. Histograms of reliability parameters for Example III.
Fig. 10. Histograms of reliability parameters for Example IV.
4.0 Conclusions and Recommendations

4.1 RELY I

We have developed and displayed a model which we believe more accurately describes an actual testing environment of a large software package. This new generalized model has been incorporated in an estimation algorithm for the purpose of discerning reliability of the software from its test data. The first version of the algorithm RELY I described here converges in a given region of interest of the model parameters. RELY I is applicable to software test cases where tape replacement (software revision) occurs at uniform intervals of time, and where sufficiently reliable information is available concerning the number of errors detected during each of the successive intervals.

4.2 Data Requirements

Data required for RELY I are simple, viz., the time interval $T$ between software revisions (tape replacements), and the sequence $\Delta_d(k)$ during each of the $K$ successive versions of the software, where $k = 1, 2, \ldots, K$. Secondly, the data must be from a process of the type upon which the assumptions of the model were based, viz., the testing of large-scale software packages such as that in the F-16 control system.

There must be an identifiable single continuous line of software package identity throughout the test process. The package passes successively through a sequence of versions $k$, $k = 1, 2, \ldots, K$. At any given time during test, the software is in only one of the versions,
"all" of which software version is being tested. Each version, \( k \), is identical to the preceding version, \( k - 1 \), and succeeding version, \( k + 1 \), except for the software corrections "counted" in \( \Delta_c(k) \) and \( \Delta_c(k+1) \), respectively. Figure 11 indicates the time relationship of the several sequential quantities. The requirement is that \( \Delta_d(k) \) be precisely known for each version \( k \), where all versions are identified and satisfy this single and continuous identity as described. This requirement is violated if a major untested version is suddenly introduced midstream, or if an alternate part of the software package simultaneously being tested suddenly is adopted. The generated error feature can accommodate a minor amount of this kind of violation, but generated errors are modeled as occurring as a constant proportion of the correction rate.

Errors are usually classified into certain arbitrary categories, ranging from those obviously to be counted, to those of doubtful pertinence (obviously "repeated" errors, errors associated purely with erroneous test conduct, etc.). Suffice it here to suggest that the criterion for counting a given error or not will be related to its likelihood of occurrence, and its interpretation as a "failure", under operational conditions.

4.3 Recommendations

Recommendations toward improved interfacing with information in a real test process (thus taking greater advantage of inherent features

---

1 That is, all the parts of the software package are being exercised in a manner representing that for which the reliability factors, e.g., MTTF, are to be applied later.
Fig. 11. Tested software in only one version at a time, identical to neighboring versions except for the changes counted in $\Delta_c$ at the respective boundaries.
of the underlying new model) include the following further work:

1. Revise the KOST subroutine to accommodate nonuniform test intervals $T(k)$, by using a finite difference technique for the solution of $\hat{\phi}(k, \hat{\theta})$. The increased utility is expected to far outweigh the lesser analytic tractability of the resulting system and the possible increase in required computation time.

2. Apply the algorithm to real data. Available data should be gathered, studied, and adapted, by interpretation and transformations, to the requirements of RELY. Residual functions over a variety of cases will indicate how well the model represents the real test process. Experience will lead to further recommendations concerning data requirements, and to possible improvements in RELY such as provisions for using information in real data concerning error correction and error generation.

For example, the quantity $n_a(k)$,

$$n_a(k) = N_a - n_c(k) = \frac{N_0 - n_c(k) + n_g(k)}{1 - \gamma} = \frac{n(k)}{1 - \gamma}$$

is the augmented number of errors remaining in the tested software. That is, $n_a(k)$ is the number of errors which would be detected henceforth if the testing process were to continue indefinitely, including those generated after time $k$. The number of errors remaining in the software, excluding
those yet to be generated in the correction process, is

\[ n(k) = (1 - \gamma) n_a(k) \]

The parameter \( \gamma \) is assumed not observable in the present implementation of the model. If, however, among the detected errors, generated errors are distinguishable from original errors, then the additional quantity \( n_{gd}(k) \), the number of generated errors detected, is available. The model state is easily augmented to include \( n_{gd}(k) \). The model remains unchanged but, to the extent that the additional information is available in test data, the model parameter \( \gamma \) becomes observable.

Experience in the development of RELY I suggests further investigation of the nature of the \( J(\hat{\theta}) \) surface. Such investigation should include also the surface associated with the alternative cost functional using cumulative functions \( N(k) \), rather than the incremental number of errors \( \Delta(k) \). Convergence properties in certain regions of parameter space may be significantly improved using \( N(k) \) rather than their derivatives \( \Delta(k) \). Indeed, parallel computation using each, respectively, may prove both feasible and advantageous. Another gradient type parameter estimation method, such as the Fletcher-Powell deflected gradient method, may also prove more efficient with the alternative functional.
REFERENCES


APPENDIX A

MAIN AND SUBROUTINE DESCRIPTIONS

1. MAIN and Subroutine Diagram

The subroutines of RELY I are indicated in Fig. A.1. Internal subroutines EIGMIN and MARQ are shown, as well as the UNIVAC MATH-PACK library subroutines RANDXX, TRIDMX, EIGVAL, and EIGVEC. A brief description of MAIN and its subroutines follows.

Fig. A.1. RELY I subroutine diagram.
2. **MAIN**

   MAIN reads input data, executes the estimation algorithm (see Sec. 3.2), and prints output. It also computes simulated random data \( \Delta_d(k) \) under the ISIM = 1 option (see Appendix C).
3. Subroutine RNDTA (A, B, P, NA, T, V, RR, JJJ)

From α, β, φ_a, N_a, T, the input initial random numbers and L (corresponding to RNDTA variables A, B, P, NA, T, RR, JJJ, respectively, and corresponding to the MAIN variables ALPH, BETA, PHI, NA, TD, RRR, NN, respectively), RNDTA computes the sequence Δ_d(k), k = 1, 2, ... L (the RNDTA variable V, and MAIN variable S). The resulting sequence Δ_d(k) is used as simulated data, the (incremental) number of errors detected successively in each software test interval. The initial random number RRR is passed through POISS to RANDEX.

RNDTA, at each interval k, integrates the system equations:

\[ n_d(k) = N_d(k - 1) + \alpha \delta_a T (N_a - n_c(k - 1)) \]

\[ n_c(k) = n_c(k - 1) + \beta \delta_a T (N_a - n_c(k - 1)) + \beta T [N_d(k - 1) - n_c(k - 1)] \]

\[ N_d(0) = n_c(0) = 0 \]

using the cumulative (random) number of errors detected \( N_d(k - 1) \), to obtain \( n_c(k) \) for use in POISS. Subroutine POISS generates the random integer \( \Delta_d(k) \) according to the mean detection rate \( r_d(k) = \phi_a [N_a - n_c(k)] \).
4. Subroutine POISS (DD, ZZ, PP, TT, RRRR, KKK)

From \( n_a (N_a - n_c) \), \( \phi_a \), T, the input initial random number RRR, and \( k \) (POISS variables DD, PP, TT, RRRR, and KKK, respectively, corresponding to RNDTA variables D, RP, RT, RQ, and K), POISS computes the value \( \Delta_d(k) \), according to

\[
\sum_{i=1}^{m} C(i) < T, \quad \Delta_d(k) = m
\]

The random sequence \( C(i), i = 1, 2, ... 100 \), with exponential distribution function \( 1 - e^{-r_d C^r} \), \( r_d = \phi(N_a - n_c) \), is generated by RANDEX. The starting random number required by RANDEX in \( C(1) \) is the input initial random number RRR for \( k = 1 \), and is the preceding random number \( R(2^{25}) \) for \( k > 1 \), where \( R \) is the value \( C(100) \) previously computed for the \( (k - 1)^{th} \) pass.
5. Subroutine RANDEX (C, 100, U)

Reference: UNIVAC Large Scale System MATH-PACK, Programmer's Reference, UP-7542, Rev. 1.

RANDEX produces a set of 100 pseudo-random numbers C with exponential distribution

\[ 1 - C^{-UC} \]

by operating on a uniformly distributed variate X, according to the inverse transform method

\[ C = \frac{-\ln(1 - X)}{U} \].

RANDEX uses two other UNIVAC MATH-PACK subroutines RANDU and RANDN. RANDU generates X, 0 ≤ X < 1, for which computation it calls RANDN for random integers 0 ≤ I < 2^{35}. RANDEX requires an initial value, 0 ≤ C(1) < 2^{35}, different integer parts of which produce different output sequences.
Given values of the parameters $\delta$, time interval $T$, number of intervals $K$, test data $\Delta_d(k)$, KOST computes the incremental error sequences (see Sec. 2.2)

$$ \delta_d(k) = N_a \phi_a (1 \ 0) L^{k-1} B $$

$$ \delta_c(k) = N_a \phi_a (0 \ 1) L^{k-1} B $$

where $(1 \ 0)$ and $(0 \ 1)$ are the transposes of the vectors $(1 \ 0)$ and $(0 \ 1)$, respectively, and

$$ L = \begin{pmatrix} 1 & -\phi_a^T \\ \beta_T & 1 - T(\beta + a\phi_a) \end{pmatrix}, \quad B = \begin{pmatrix} T \\ a_T \end{pmatrix} $$

KOST further computes the cost scalar (see Sec. 3.1)

$$ J = \sum_{k=1}^K \left[ \delta_d(k) - \Delta_d(k) \right]^2 $$

the gradient vector components

$$ \frac{\partial J}{\partial \theta_m} = 2 \sum_{k=1}^K \delta_d(k) \frac{\partial \delta_d(k)}{\partial \theta_m} $$

and the Hessian matrix elements

$$ \frac{\partial^2 J}{\partial \theta_n \partial \theta_m} = 2 \sum_{k=1}^K \left[ \delta_d(k) \frac{\partial^2 \delta_d(k)}{\partial \theta_n \partial \theta_m} + \frac{\partial \delta_d(k)}{\partial \theta_n} \frac{\partial \delta_d(k)}{\partial \theta_m} \right] $$

- A.6 -
KOST also computes the associated estimate of total errors corrected

\[ n_c = \sum_{k=1}^{K} \delta_c(k) \]

the number of errors remaining

\[ n_{R} = N_{a} - n_c \]

and the mean time to failure

\[ \text{MTTF} = \frac{1}{\Phi_a n_{R}} \]

The first derivatives above are given by:

\[ \frac{\partial \delta_d(k)}{\partial \theta_m} = N_{a} \phi_{a} (1 \ 0) L^{k-2} \left[ (k-1) \frac{\partial L}{\partial \theta_m} B + L \frac{\partial B}{\partial \theta_m} \right] \]

\[ + \delta_d(k) \left[ \frac{1}{\phi_{a}} \frac{\partial \phi_{a}}{\partial \theta_m} + \frac{1}{N_{a}} \frac{\partial N_{a}}{\partial \theta_m} \right] \]

where

\[ \frac{\partial \theta_{i}}{\partial \theta_{m}} = \begin{cases} 1 & i = m \\ 0 & i \neq m \end{cases} \]

and

- A.7 -
The second derivatives are:

\[
\frac{\partial^2 \delta_\phi(k)}{\partial \theta_n \partial \theta_m} = N^a \left( \begin{array}{c} 1 \\ 0 \\ 0 \end{array} \right) \quad \left\{ \begin{array}{l} (k-2) L^{k-3} \frac{\partial L}{\partial \theta_n} \\ (k-1) \frac{\partial L}{\partial \theta_m} B + L \frac{\partial B}{\partial \theta_m} \end{array} \right\}
\]

\[
+ L^{k-2} \left( \begin{array}{c} \frac{\partial^2 L}{\partial \theta_n \partial \theta_m} B + \frac{\partial L}{\partial \theta_m} \frac{\partial B}{\partial \theta_n} \\ \frac{\partial^2 L}{\partial \theta_n \partial \theta_m} \end{array} \right) + \frac{\partial^2 B}{\partial \theta_n \partial \theta_m}
\]

\[
+ L \left( \delta_d(k) \left\{ \begin{array}{c} 1 \\ \phi \frac{\partial a}{\partial \theta_m} \end{array} \right\} + \frac{3N_a}{\partial \theta_m} \right)
\]


-A.8-
7. Subroutine EIGMIN (HH, CORR, GGJ, DDAM, EH, EV)

Given the Hessian HH(4, 4) and the gradient GGJ(4) of the cost functional $J(\bar{\beta})$, and the current Marquardt parameter $\lambda$ DDAM, EIGMIN computes the eigenvalues EV(4) and eigenvectors EGV(4, 4), and the outer product EH(4, 4, 4), for $J$. EIGMIN then computes $\lambda_1$ such that the Marquardt matrix $(H_1 + \lambda_1 C_1^2)$ is positive definite, and the corresponding parameter correction factors CORR(4) $= (H_1 + \lambda_1 C_1^2)^{-1} \frac{\bar{\beta}}{\bar{\beta}_1}$. 
8. **Subroutine TRIDMX (N, NM, A, D, B)**

Reference: UNIVAC Large-Scale System MATH-PACK, Programmer's Reference, UP-7542, Rev. 1, Sec. 9, p. 1.

TRIDMX transforms a real symmetric matrix, B(4, 4), to tridiagonal form using Householder's method, where D(4) are the resulting diagonal elements and B(4) are the off-diagonal elements. Input integers N and NM are equal to the order, 4, of B.
9. **Subroutine EIGVAL** (LP, E, A, B, W, F)

   Reference: UNIVAC Large-Scale Systems MATH-PACK, Programmer's Reference, UP-7542, Rev. 1, Sec. 9, p. 8.

   EIGVAL evaluates the eigenvalues of a symmetric tridiagonal matrix, using Sturm sequences. A(4) are the diagonal elements and B(4) are the off-diagonal elements of the matrix. The eigenvalues E(4) are stored in descending order of absolute value.
10. Subroutine EIGVEC (LP, NM, R, A, B, E, V, P, Q)

Reference: UNIVAC Large-Scale Systems MATH-PACK, Programmer's Reference, UP-7542, Rev. 1, Sec. 9, p. 15.

EIGVEC evaluates the eigenvectors of a real symmetric tridiagonal matrix using Wilkinson's method. A(4) are the diagonal elements and B(4) are the off-diagonal elements of the matrix. E(4) are the eigenvalues, and V(4, 4) are the eigenvectors.
11. Subroutine MARQ (EEH, EEV, DDLAM, CCORR, GGGJ, HHH)

Given the outer products EEH(4, 4, 4) of the eigenvectors of the
Hessian, the eigenvalues EEV(4), the gradient GGGJ(4), and Marquardt
parameter (λ₁) DDLAM, MARQ computes the step CCORR(4), (H₁ + λ₁C₁)⁻¹ S₁,
in parameter space.
APPENDIX B

RELY I GLOSSARY AND INDEX

(Library subroutines RANDEX, RANDU, RANDN, TRIDMX, EIGVAL, EIGVEC are not included here. See UNIVAC MATH-PACK references given in Appendix A for detailed information.)

MAIN (including internal subroutines EIGMIN and MARQ)

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
<th>Line Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALPH</td>
<td>Value for $a$ in simulation mode</td>
<td>18, 22, 25, 30</td>
</tr>
<tr>
<td>$B(4, 4)$</td>
<td>Normalized Hessian matrix in EIGMIN</td>
<td>162, 167, 170, 172</td>
</tr>
<tr>
<td>BETA</td>
<td>Value for $b$ in simulation mode</td>
<td>18, 22, 25, 30</td>
</tr>
<tr>
<td>CNC</td>
<td>Cumulative values for $n_c$ in simulation mode</td>
<td>24, 33, 34</td>
</tr>
<tr>
<td>CND</td>
<td>Cumulative values for $n_d$ in simulation mode</td>
<td>23, 32, 34</td>
</tr>
<tr>
<td>CORR(4)</td>
<td>Vector of corrections to parameters</td>
<td>3, 48, 57, 58, 61-63, 72, 78, 86-88, 95, 114, 115-117, 127, 137, 160, 163, 201</td>
</tr>
<tr>
<td>COST</td>
<td>Most recently computed value for the mean-squared error</td>
<td>25, 30, 43, 46, 49, 50, 56, 65, 67, 69, 70, 73, 76, 91, 93, 94, 100, 120, 122, 124-127, 129, 130, 135, 145, 147</td>
</tr>
<tr>
<td>COSTI</td>
<td>Cost for currently accepted parameter values used in mode B to determine whether the new estimate for the parameters reduces the cost</td>
<td>46, 130, 145</td>
</tr>
<tr>
<td>DDD</td>
<td>Denominator used to normalize the matrix $(H + \lambda I)^{-1}$ in EIGMIN</td>
<td>194, 195</td>
</tr>
<tr>
<td>Variable</td>
<td>Description</td>
<td>Line Number</td>
</tr>
<tr>
<td>----------</td>
<td>-------------</td>
<td>-------------</td>
</tr>
<tr>
<td>DDDD</td>
<td>Denominator used to normalize the matrix ((H + \lambda I)^{-1}) in MARQ</td>
<td>223, 224</td>
</tr>
<tr>
<td>DIA(4)</td>
<td>Diagonal entries of the tridiagonalized Hessian matrix used in EIGMIN to compute eigenvalues</td>
<td>162, 170-172</td>
</tr>
<tr>
<td>DLAM</td>
<td>Value for (\lambda)</td>
<td>45, 48, 72, 74, 78, 95, 103, 111, 114, 128, 137</td>
</tr>
<tr>
<td></td>
<td>(in EIGMIN: DDAM)</td>
<td>160, 188, 205, 217</td>
</tr>
<tr>
<td></td>
<td>(in MARQ: DDLAM)</td>
<td>182</td>
</tr>
<tr>
<td>DND(300)</td>
<td>Incremental values in (n_d)</td>
<td>4, 25, 32, 43, 67, 76, 91, 120, 135</td>
</tr>
<tr>
<td>EGV(4, 4)</td>
<td>Eigenvectors for the Hessian matrix</td>
<td>162, 172, 176</td>
</tr>
<tr>
<td>EH(4, 4, 4)</td>
<td>Outer products of the eigenvectors for the Hessian matrix used to compute ((H + \lambda I)^{-1})</td>
<td>3, 48, 72, 78, 95, 114, 128, 137, 160, 162, 176, 188</td>
</tr>
<tr>
<td></td>
<td>(in MARQ: EEH)</td>
<td>205, 207, 217</td>
</tr>
<tr>
<td>ENC(300)</td>
<td>Incremental values in (n_c)</td>
<td>4, 25, 33, 43, 67, 76, 91, 120, 135</td>
</tr>
<tr>
<td>ERR</td>
<td>Value for termination criterion</td>
<td>5, 7, 56, 93, 126</td>
</tr>
<tr>
<td>EV(4)</td>
<td>Eigenvalues for the Hessian matrix</td>
<td>3, 48, 72, 78, 95, 114, 128, 137, 160, 162, 172, 181, 182, 185, 188</td>
</tr>
<tr>
<td></td>
<td>(in MARQ: EEV)</td>
<td>205, 207, 217</td>
</tr>
<tr>
<td>GJ(4)</td>
<td>Gradient vector for the cost functional</td>
<td>3, 25, 43, 48, 67, 72, 76, 91, 95, 114, 120, 127, 135, 137</td>
</tr>
<tr>
<td></td>
<td>(in EIGMIN: GGJ)</td>
<td>160, 163, 201</td>
</tr>
<tr>
<td></td>
<td>(in MARQ: GGGJ)</td>
<td>205, 207, 229</td>
</tr>
<tr>
<td>H(4, 4)</td>
<td>Hessian matrix for the cost functional</td>
<td>3, 25, 43, 48, 67, 72, 76, 78, 91, 95, 114, 120, 127, 135, 137</td>
</tr>
<tr>
<td></td>
<td>(in EIGMIN: HH)</td>
<td>160, 162, 167, 194</td>
</tr>
<tr>
<td></td>
<td>(in MARQ: HHH)</td>
<td>205, 207, 223</td>
</tr>
<tr>
<td>Variable</td>
<td>Description</td>
<td>Line Number</td>
</tr>
<tr>
<td>----------</td>
<td>-------------</td>
<td>-------------</td>
</tr>
<tr>
<td>I</td>
<td>Index for various loops</td>
<td></td>
</tr>
<tr>
<td>ICLAM</td>
<td>Index which counts the number of times that $\lambda$ is increased in mode B</td>
<td>104, 113, 144</td>
</tr>
<tr>
<td>IFLAG</td>
<td>Index that counts the number of times that an iteration of mode A reduces the cost by less than 1 percent</td>
<td>47, 70, 71, 106</td>
</tr>
<tr>
<td>INDEX</td>
<td>Index that counts the number of times that the step size has been reduced</td>
<td>51, 80, 81</td>
</tr>
<tr>
<td>ISIM</td>
<td>Indicates whether the run is a simulation (ISIM = 1) or an estimation with real data</td>
<td>15, 17</td>
</tr>
<tr>
<td>J</td>
<td>Index for various loops</td>
<td></td>
</tr>
<tr>
<td>JDEX</td>
<td>JDEX = 1 indicates the first time that changing $\lambda$ has been successful in a given iteration of mode B</td>
<td>105, 143, 144</td>
</tr>
<tr>
<td>KK</td>
<td>Index used for DO loop in EIGMIN for computing outer products of eigenvectors</td>
<td></td>
</tr>
<tr>
<td>KL</td>
<td>Index used for DO loop in EIGMIN and MARQ for computing $(H + \lambda I)^{-1}$</td>
<td>184, 185, 188, 214, 217</td>
</tr>
<tr>
<td>LMBEX</td>
<td>LMBEX = 1 indicates that $\lambda$ was changed in mode B</td>
<td>112, 122, 123</td>
</tr>
<tr>
<td>NA</td>
<td>Value for $N_a$ in simulation mode</td>
<td>2, 18, 22, 25, 30</td>
</tr>
<tr>
<td>NJ</td>
<td>Number of test intervals</td>
<td>10, 25, 43, 67, 76, 91, 120, 135</td>
</tr>
<tr>
<td>NN</td>
<td>Number of tape versions</td>
<td>5, 7, 10, 11, 22, 25, 31, 43, 67, 76, 91, 120, 135</td>
</tr>
<tr>
<td>Variable</td>
<td>Description</td>
<td>Line Number</td>
</tr>
<tr>
<td>-----------</td>
<td>-------------</td>
<td>-------------</td>
</tr>
<tr>
<td>OFDI(4)</td>
<td>Off-diagonal entries in tridiagonalized Hessian matrix</td>
<td>162, 170-172</td>
</tr>
<tr>
<td>PCOST</td>
<td>Current minimum value for the cost functional</td>
<td>50, 69, 70, 73, 94, 124, 127, 129</td>
</tr>
<tr>
<td>PHI</td>
<td>Value of $\phi_a$ in simulation mode</td>
<td>18, 22, 25, 30</td>
</tr>
<tr>
<td>R(4, 4)</td>
<td>Matrix $(H + \lambda I)^{-1}$ computed in EIGMIN</td>
<td>163, 166, 188, 195, 201</td>
</tr>
<tr>
<td>RR(4, 4)</td>
<td>Matrix $(H + \lambda I)^{-1}$ computed in MARQ</td>
<td>207, 211, 217, 224, 229</td>
</tr>
<tr>
<td>REFF</td>
<td>Estimated number of errors remaining at the end of test period</td>
<td>26, 30, 44, 49, 68, 77, 92, 100, 121, 122, 125, 136, 147</td>
</tr>
<tr>
<td>RMTTF</td>
<td>Estimated mean time to failure</td>
<td>25, 30, 43, 49, 67, 76, 91, 100, 120, 122, 125, 135, 147</td>
</tr>
<tr>
<td>RRR</td>
<td>Randomization value in simulation mode</td>
<td>18, 19, 22</td>
</tr>
<tr>
<td>S(300)</td>
<td>Error data</td>
<td>3, 12, 22, 25, 38, 43, 67, 76, 91, 120, 135</td>
</tr>
<tr>
<td>T(300)</td>
<td>Tape version replacement times</td>
<td>3, 13, 25, 43, 67, 76, 91, 120, 135</td>
</tr>
<tr>
<td>TAU</td>
<td>Step size</td>
<td>57-63, 79, 86-88, 102, 115-117, 144</td>
</tr>
<tr>
<td>TD</td>
<td>Length of each test interval</td>
<td>5, 7, 13, 22, 25, 43, 67, 76, 91, 120, 135</td>
</tr>
<tr>
<td>TEMP1(4)</td>
<td>Vectors which are used temporarily in the computation of the eigenvalues and eigenvectors of Hessian matrix</td>
<td>163, 171, 172</td>
</tr>
<tr>
<td>TEMP2(4)</td>
<td></td>
<td>163, 171, 172</td>
</tr>
<tr>
<td>Variable</td>
<td>Description</td>
<td>Line Number</td>
</tr>
<tr>
<td>----------</td>
<td>-------------</td>
<td>-------------</td>
</tr>
<tr>
<td>ZA</td>
<td>Value for $a$ in simulation and estimation mode</td>
<td>40, 43, 49, 52, 67, 82, 91, 96, 100, 107, 120, 122, 125, 131, 135, 139, 147</td>
</tr>
<tr>
<td>ZB</td>
<td>Value for $\beta$ in simulation and estimation mode</td>
<td>40, 43, 49, 53, 57, 61, 64, 66-67, 83, 86, 89-91, 97, 100, 108, 115, 118-120, 122, 125, 132, 135, 140, 147</td>
</tr>
<tr>
<td>ZN</td>
<td>Value for $N_a$ in simulation and estimation mode</td>
<td>40, 43, 49, 55, 63, 67, 85, 88, 91, 99, 100, 110, 117, 120, 122, 125, 134, 135, 142, 147</td>
</tr>
<tr>
<td>ZNC</td>
<td>Estimated value for $n_c$ at the end of the test period</td>
<td>26, 30, 44, 49, 68, 77, 92, 100, 121, 122, 125, 136, 147</td>
</tr>
<tr>
<td>ZP</td>
<td>Value for $\phi_a$ in simulation and estimation mode</td>
<td>40, 43, 49, 54, 58, 62, 67, 84, 87, 91, 98, 100, 109, 116, 120, 122, 125, 133, 135, 141, 147</td>
</tr>
<tr>
<td>ZZA</td>
<td>Currently accepted value for $a$</td>
<td>52, 76, 82, 96, 107, 137, 139</td>
</tr>
<tr>
<td>ZZB</td>
<td>Currently accepted value for $\beta$</td>
<td>53, 76, 83, 97, 108, 115, 132, 140</td>
</tr>
<tr>
<td>ZZN</td>
<td>Currently accepted value for $N_a$</td>
<td>55, 76, 85, 99, 110, 117, 134, 142</td>
</tr>
<tr>
<td>ZZP</td>
<td>Currently accepted value for $\phi_a$</td>
<td>54, 76, 84, 98, 109, 116, 133, 141</td>
</tr>
</tbody>
</table>

**Subroutine RNDTA**

<table>
<thead>
<tr>
<th>A (dbl)</th>
<th>1, 7, 19</th>
</tr>
</thead>
<tbody>
<tr>
<td>Correction rate parameter $a$</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>B (dbl)</th>
<th>1, 8, 19</th>
</tr>
</thead>
<tbody>
<tr>
<td>Correction rate parameter $\beta$</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>D</th>
<th>4, 21, 18</th>
</tr>
</thead>
<tbody>
<tr>
<td>Estimated number of remaining errors $n_a$</td>
<td></td>
</tr>
</tbody>
</table>

- B.5 -
<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
<th>Line Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>E (dbl)</td>
<td>D</td>
<td>17, 18</td>
</tr>
<tr>
<td>JJJ</td>
<td>Number L of intervals (software versions)</td>
<td>1, 15</td>
</tr>
<tr>
<td>K</td>
<td>Index corresponding to k(^{th}) interval</td>
<td>15, 21, 27</td>
</tr>
<tr>
<td>NA (dbl)</td>
<td>Initial number (N_a) ((\gamma)-augmented) of software errors</td>
<td>1, 3, 10, 16, 17, 19</td>
</tr>
<tr>
<td>P (dbl)</td>
<td>Detection rate parameter (\phi_a)</td>
<td>1, 9, 16, 19</td>
</tr>
<tr>
<td>RA</td>
<td>A</td>
<td>4, 7</td>
</tr>
<tr>
<td>RB</td>
<td>B</td>
<td>4, 8</td>
</tr>
<tr>
<td>RNA</td>
<td>NA</td>
<td>4, 10</td>
</tr>
<tr>
<td>RF</td>
<td>P</td>
<td>4, 9, 21</td>
</tr>
<tr>
<td>RQ</td>
<td>RR</td>
<td>5, 6, 21</td>
</tr>
<tr>
<td>RR</td>
<td>Storage place for MAIN input initial random number for RANDN. Contains first exponential random number from RANDEX upon return.</td>
<td>1, 6</td>
</tr>
<tr>
<td>RT</td>
<td>T</td>
<td>4, 11, 21</td>
</tr>
<tr>
<td>RZ</td>
<td>Poisson random (\Delta_d(k)) returned by POISS</td>
<td>4, 21, 22, 27</td>
</tr>
<tr>
<td>T (dbl)</td>
<td>Time interval T</td>
<td>1, 11, 16, 19</td>
</tr>
<tr>
<td>V(300) (dbl)</td>
<td>RZ, Poisson random sequence (\Delta_d(k)) returned by RNDTA</td>
<td>1, 12, 27</td>
</tr>
<tr>
<td>Variable</td>
<td>Description</td>
<td>Line Number</td>
</tr>
<tr>
<td>-----------</td>
<td>------------------------------------------------------------------------------</td>
<td>---------------------</td>
</tr>
<tr>
<td>X(2) (dbl)</td>
<td>Temporary memory for current cumulative random M</td>
<td>12-14, 16, 17, 19, 20, 22, 25</td>
</tr>
<tr>
<td>Y(2) (dbl)</td>
<td>Temporary memory for current cumulative estimate M</td>
<td>12, 16, 19, 20</td>
</tr>
</tbody>
</table>

Subroutine POISS

C(100)   2, 4, 6, 8, 11
Set of uniformly distributed random numbers generated by RANDEX to be used by POISS as the sequence of times between successive detected errors

DD       1, 7
Number of remaining errors nₐ

K        10, 11, 13
Counter of successive detected errors

KKK      1, 3
Number L of test intervals

PP       1, 7
Parameter value a

Q        9, 11, 12
Cumulative time Σ₁Cᵢ during the test interval, accumulated until it exceeds T

RRRR     1, 4
Initial random number for starting RANDN. Its value for k = 1 is input by MAIN. Subsequent values are set by POISS, RRRR = 2²⁵ C(100).

TT       1, 12
Test interval T

U        7, 8
Mean frequency φₐnₐ of the error detection rₜ

ZZ       1, 13, 16
Poisson random number of detections Δₜ(k) generated by POISS for the kth interval. It is Δₜ such that:

\[
\Delta_t \sum_{i=1}^{\infty} C_i \leq T, \quad \Delta_t \leq 100
\]

- B.7 -
The program listing (FORTRAN V, Appendix D) accompanying this report consists of MAIN with internal subroutines EIGMIN and MARQ, and external double-precision subroutines RNDTA, POISS, KOST, TRIDMX, EIGVAL, and EIGVEC. The subroutines which call single-precision library functions have the necessary coding for converting between double- and single-precision variables. Certain double-precision library functions are used, viz., DABS, DEXP, and DSQRT.

The input deck depends on whether data are to be simulated or are to be read from input cards.

**INPUT DECK (to simulate data and estimate)**

Card 1: TD, NN, ERR, [F5.2, 2X, I4, 2X, F8.6]
Card 2: ISIM [12] (must be unity)
Card 3: ALPH, BETA, PHI, NA [4(G14.6, 2X)]
Card 4: RRR [16X, G14.1] (input initial random number)
Card 5: ZA, ZB, ZP, ZN [4(G14.6, 2X)]

**INPUT DECK (to estimate using punched card data)**

Card 1: TD, NN, ERR, [F5.2, 2X, I4, 2X, F8.6]
Card 2: ISIM [12] (must be zero)
Card 3: ZA, ZB, ZP, ZN [4(G14.6, 2X)]
Card 4-(k+3): S(i), i = 1, 2, ..., K [G10.1]

The integer part of any real number RRR, $0 \leq RRR < 2^{35}$.
determines a unique repeatable random sequence $\Delta_d(k)$ from the simulator.

Initial guesses $\hat{\theta}_0$ for the model parameters are recommended as follows:

- $0 < ZA < 1$. (typically 0.8)
- $0 < ZB < 1$. (typically 0.0)
- $0 < ZP < 0.2$
- $0 < ZN$

To produce different simulated random data, the operator must change the input initial random number RRR, $0 < RRR < 2^{35}$.

Though $J < \text{ERR}$ is the internal stopping criterion, experience in random cases proved maximum pages of printed output (say, 10) to be as practical a stopping criterion as any.

The program prints out the current accepted parameter estimates $\hat{\theta}_i = a, \beta, \gamma, N_a$, together with running estimates of the reliability parameters $N_R$ and MTTF, and selected auxiliary quantities, at each step $i$. However, the label "CHANGING LAMBDA" indicates only tentative parameter values produced in mode B (see Sec. 3.2). Therefore, these tentative values must not be taken as values which minimize the cost functional $J$. The final estimates of the reliability parameters are those associated with the last accepted iteration $i$. 

- C.2 -
RELY I PROGRAM LISTING AND SAMPLE OUTPUT

HELTERELY(1), MAIN

1 10 PLICII REAL*8(A-H#E#Z)
2 REAL*8 NA
3 IMESIUSH T(300) S(300) H(4,4) GJ(4) CORR(4) EH(4,4) EV(4)
4 IMESIUSI D(300) E(300)
5 CLAUS(5007) T6 N=ERR
6 WRITE(6,88) T6 N=ERR
7 WRITE(6,88) T6 N=ERR
8 WRITE(6,88) T6 N=ERR
9 WRITE(6,88) T6 N=ERR
10 WRITE(6,88) T6 N=ERR
11 WRITE(6,88) T6 N=ERR
12 WRITE(6,88) T6 N=ERR
13 WRITE(6,88) T6 N=ERR
14 WRITE(6,88) T6 N=ERR
15 WRITE(6,88) T6 N=ERR
16 WRITE(6,88) T6 N=ERR
17 WRITE(6,88) T6 N=ERR
18 WRITE(6,88) T6 N=ERR
19 WRITE(6,88) T6 N=ERR
20 WRITE(6,88) T6 N=ERR
21 WRITE(6,88) T6 N=ERR
22 WRITE(6,88) T6 N=ERR
23 WRITE(6,88) T6 N=ERR
24 WRITE(6,88) T6 N=ERR
25 WRITE(6,88) T6 N=ERR
26 WRITE(6,88) T6 N=ERR
27 WRITE(6,88) T6 N=ERR
28 WRITE(6,88) T6 N=ERR
29 WRITE(6,88) T6 N=ERR
30 WRITE(6,88) T6 N=ERR
31 WRITE(6,88) T6 N=ERR
32 WRITE(6,88) T6 N=ERR
33 WRITE(6,88) T6 N=ERR
34 WRITE(6,88) T6 N=ERR
35 WRITE(6,88) T6 N=ERR
36 WRITE(6,88) T6 N=ERR
37 WRITE(6,88) T6 N=ERR
38 WRITE(6,88) T6 N=ERR
39 WRITE(6,88) T6 N=ERR
40 WRITE(6,88) T6 N=ERR
41 WRITE(6,88) T6 N=ERR
42 WRITE(6,88) T6 N=ERR
43 WRITE(6,88) T6 N=ERR
44 WRITE(6,88) T6 N=ERR
45 WRITE(6,88) T6 N=ERR
46 WRITE(6,88) T6 N=ERR
47 WRITE(6,88) T6 N=ERR
48 WRITE(6,88) T6 N=ERR
49 WRITE(6,88) T6 N=ERR
50 WRITE(6,88) T6 N=ERR
51 WRITE(6,88) T6 N=ERR
52 WRITE(6,88) T6 N=ERR
53 WRITE(6,88) T6 N=ERR
54 WRITE(6,88) T6 N=ERR
55 WRITE(6,88) T6 N=ERR
56 WRITE(6,88) T6 N=ERR

-D.1-
TAU=(5-..b)/(2*DABS(CORR(2)))

IF (TAU<UT.1) TAU=1
IF (TAU=L.T.0.0001) GO TO 74
LB=LB+CORR(2)*TAU
LF=LP+CORR(3)*TAU
LZ=LZ+CORR(4)*TAU
IF (LB.LT.0.0) LB=0.
IF (COST+GT.10) DLAM=1.
IF (LB+1.12B=16.)
CALL KOST(ZA+ZB+2B+ZNP+5+LW.TW+TD+HD+GD*DOM+COST*ENC+RMTTF,
  141+GRENK)

IF (COST+6E,PCOST1) GO TO 76
IF (FLAU+GT.99) IFLA=IFLAG+1
IF (FLAU+GT.4) GU TO 74
CALL EMIN(H+COKK+GD+DLAM+Eh,EV)
65 COST=COST
DLAM=DLAM//10
GU TO 76
CALL KOST(ZA+ZB+ZNP+5+SN+TW+TD+HD+GD*DOM+COST*ENC+RMTTF.
77 141+GRENK)
CALL EMIN(H+COKK+GD+DLAM+Eh,EV)

TAU=TAU/10
60 GE=GE+SE+1
IF (GE.LT.5) GU TO 74
71=72
74=75
78=79
80=81
82=83
84=85
85=86
86=87
87=88
88=89
89=90
90=91
91=92
92=93
93=94
94=95
95=96
96=97
97=98
98=99
99=100
100=101
101=102
102=103
103=104
104=105
105=106
106=107
107=108
108=109
109=110
110=111
111=112
112=113
113=114

- D.2 -
CALL MAHL(EHV,ULAM,CORK,GNJ)

76 &zzcE=CHRR(2)*TAU
76 &zzcE=CHRR(3)*TAU
77 IF(Z1=Z2)Z1=Z2
78 IF(LME(X,EQ.1))WHITE(6.190)Z2=Z2
79 IF(COST(5)=COST(5))GO TO 72
80 IF(COST(5)<COST(5))GO TO 91
81 Z2=Z2
82 &zzCZB
83 &zzCZB
84 &zzCZB
85 &zzCZB
86 &zzCZB
87 &zzCZB
88 &zzCZB
89 &zzCZB
90 &zzCZB
91 &zzCZB
92 &zzCZB
93 &zzCZB
94 &zzCZB
95 &zzCZB
96 &zzCZB
97 &zzCZB
98 &zzCZB
99 &zzCZB
100 &zzCZB
101 &zzCZB
102 &zzCZB
103 &zzCZB
104 &zzCZB
105 &zzCZB
106 &zzCZB
107 &zzCZB
108 &zzCZB
109 &zzCZB
110 &zzCZB
111 &zzCZB
112 &zzCZB
113 &zzCZB
114 &zzCZB
115 &zzCZB
116 &zzCZB
117 &zzCZB
118 &zzCZB
119 &zzCZB
120 &zzCZB
121 &zzCZB
122 &zzCZB
123 &zzCZB
124 &zzCZB
125 &zzCZB
126 &zzCZB
127 &zzCZB
128 &zzCZB
129 &zzCZB
130 &zzCZB
131 &zzCZB
132 &zzCZB
133 &zzCZB
134 &zzCZB
135 &zzCZB
136 &zzCZB
137 &zzCZB
138 &zzCZB
139 &zzCZB
140 &zzCZB
141 &zzCZB
142 &zzCZB
143 &zzCZB
144 &zzCZB
145 &zzCZB
146 &zzCZB
147 &zzCZB
148 &zzCZB
149 &zzCZB
150 &zzCZB
151 &zzCZB
152 &zzCZB
153 &zzCZB
154 &zzCZB
155 &zzCZB
156 &zzCZB
157 &zzCZB
158 &zzCZB
159 &zzCZB
160 &zzCZB
161 &zzCZB
162 &zzCZB
163 &zzCZB
164 &zzCZB
165 &zzCZB
166 &zzCZB
167 &zzCZB
168 &zzCZB
169 &zzCZB
170 &zzCZB

- D.3 -
CALL EUVAL(EV, EVLIA, OFGl, TEMPl, TEMP2)
CALL EUVEC(EV, EVLIA, OFGl, EV, EVG, TEMPl, TEMP2)
DO 706 w=1.4
DO 709 =1.4
DE(1+K)(K+1)=EGV(J*1)*EGV(K+1)

710 CONTINUE
709 CONTINUE
708 CONTINUE
UL 713 =1.4
IF(EV(I).GT.0)GO TO 713
IF(DAM.LT.EV(I))DAM=DAM-DAU(EV(I))+1
713 CONTINUE
UL 702 AL=1.4
IF(DAMSL(EV(KL)).LT.00001)EV(KL)=1
700 UL 704 =1.4
707 UL 704 =1.4
K(I,J)=K(I,J)+HE(KL+I,J)/(EV(KL)+DDAM)

704 CONTINUE
703 CONTINUE
702 CONTINUE
UL 705 =1.4
UL 706 =1.4
704 UL=LSh(T(DAU(MM(I,1)*MM(J,J)))
705 K(I,J)=K(I,J)/ShL

706 CONTINUE
705 CONTINUE
UL 711 =1.4
709 CORSK(I)=
200 UL 712 =1.4
201 CORSK(I)=CORSK(I)+K(I,J)*G6G(J)

712 CONTINUE
711 CONTINUE
704 RETURN
205 SUBROUTINE MARK(I,CHI;EV;CLLM;CCORF;GHHJ;HHH)

600 IMPLICIT REAL(A-H,Z)
607 DIMENSION EHH(4,4,4)EEV,ECCOR,F;G6G(4);RHN(4,4,4);HHH(4,4)
208 UL 714 =1.4
209 CORHK(I)=0
210 UL 725 =1.4
211 HR(I,J)=

725 CONTINUE
714 CONTINUE
713 CONTINUE
UL 716 AL=1.4
215 UL 717 =1.4
210 UL 718 =1.4
217 HR(I,J)=HR(I,J)+LH(KL+I,J)/(EEV(KL)+DDAM)

718 CONTINUE
717 CONTINUE
716 CONTINUE
211 UL 719 =1.4
224 UL 720 =1.4
223 ULDOCSM(T(DAB(MM(I,1)+MM(J,J)))
244 HR(I,J)=HR(I,J)/ULDO
225 720 CONTINUE
220 719 CONTINUE
227 UL 721 =1.4

- D4 -
268   LU 722 J=1+4
269   LCCORR(I)=CCORR(I)+RH(I,J)*GGGJ(J)
270    722 CONTINUE
274    721 CONTINUE
276       RETURN
278       END

***

WHAT'S RELT(I), HOST
IMPLICIT REAL*8(A%-40-Z)

IF(ISVU(SS(300)) .GT. 100)
COST = 1.0
CLL11 = 1
CLL12 = 1
CLL13 = 1
CLL14 = 1
CLL15 = 0
CLL21 = 1
CLL22 = 1
CLL23 = 1
CLL24 = 1
CLL25 = 1
CLL31 = 1
CLL32 = 1
CLL33 = 1
CLL34 = 1
CLL35 = 1
CLL41 = 1
CLL42 = 1
CLL43 = 1
CLL44 = 1
CLL45 = 1
CLL51 = 1
CLL52 = 1
CLL53 = 1
CLL54 = 1
CLL55 = 1
CLL61 = 1
CLL62 = 1
CLL63 = 1
CLL64 = 1
CLL65 = 1
CLL71 = 1
CLL72 = 1
CLL73 = 1
CLL74 = 1
CLL75 = 1
CLL81 = 1
CLL82 = 1
CLL83 = 1
CLL84 = 1
CLL85 = 1
CLL91 = 1
CLL92 = 1
CLL93 = 1
CLL94 = 1
CLL95 = 1
CLL101 = 1
CLL102 = 1
CLL103 = 1
CLL104 = 1
CLL105 = 1

12 CONTINUE

13 CONTINUE

14 C = (1.0-AA) +DT

15 LH = (LH+AA) +DT

16 LH = (LH+AA) +DT

17 LH = (LH+AA) +DT

18 LH = (LH+AA) +DT

19 LH = (LH+AA) +DT

20 LH = (LH+AA) +DT

21 LH = (LH+AA) +DT

22 LH = (LH+AA) +DT

23 LH = (LH+AA) +DT

24 LH = (LH+AA) +DT

25 LH = (LH+AA) +DT

26 LH = (LH+AA) +DT

27 LH = (LH+AA) +DT

28 LH = (LH+AA) +DT

29 LH = (LH+AA) +DT

30 LH = (LH+AA) +DT

31 LH = (LH+AA) +DT

32 LH = (LH+AA) +DT

33 LH = (LH+AA) +DT

34 LH = (LH+AA) +DT

35 LH = (LH+AA) +DT

36 LH = (LH+AA) +DT

37 LH = (LH+AA) +DT

38 LH = (LH+AA) +DT

39 LH = (LH+AA) +DT

40 LH = (LH+AA) +DT

41 LH = (LH+AA) +DT

42 LH = (LH+AA) +DT

43 LH = (LH+AA) +DT

44 LH = (LH+AA) +DT

45 LH = (LH+AA) +DT

46 LH = (LH+AA) +DT

47 LH = (LH+AA) +DT

48 LH = (LH+AA) +DT

49 LH = (LH+AA) +DT

50 LH = (LH+AA) +DT

51 LH = (LH+AA) +DT

52 LH = (LH+AA) +DT

53 LH = (LH+AA) +DT

54 LH = (LH+AA) +DT

55 LH = (LH+AA) +DT

56 LH = (LH+AA) +DT

57 LH = (LH+AA) +DT

58 LH = (LH+AA) +DT

59 LH = (LH+AA) +DT

60 LH = (LH+AA) +DT
Lio3=iNA*f-T*

C LLK

I

AA.

L12)

FP *NN A

*DT sK*

(-AA)

*rT*

bo

19

6,1r4O()tPLT*(LLK11,AA*ZLL12)

ou

IF (K.I.E. 1) GO TO 3u

125x614

178x614

125x578

178x578

125x548

151x548

184x548

223x548

122x531

146x531

178x531

125x515

178x515

125x506

174x506

125x498

178x498

125x464

183x464

125x448

179x448

10

189x440

189x440

125x398

180x398

125x389

183x389

125x369

183x369

125x349

178x349

125x332

178x332

125x324

178x324

125x316

178x316

125x307

163x307

125x300

177x300

125x293

177x293

125x285

177x285

125x278

177x278

125x270

177x270

125x263

177x263

125x256

177x256

125x249

177x249

125x242

177x242

125x235

177x235

125x228

177x228

125x221

177x221

125x214

177x214

125x207

177x207

125x200

177x200

125x193

177x193

125x186

177x186

125x179

177x179

125x172

177x172

125x165

177x165

125x158

177x158

125x151

177x151

125x144

177x144

125x137

177x137

125x130

177x130

125x123

177x123

125x116

177x116

125x109

177x109

125x102

177x102

125x95

177x95

125x88

177x88

125x81

177x81

125x74

177x74

125x67

177x67

125x60

177x60

125x53

177x53

125x46

177x46

125x39

177x39

125x32

177x32

125x25

177x25

125x18

177x18

125x11

177x11

125x4

177x4

125x0

177x0

10 CONTINUE

97 RETURN

90 END

<==>

6PhT's HLT(1).Sub2
SUBROUTINE PUSS (UD, ZZ, PP, TT, RRRR, KKK)

DIMENSION C(100)

IF (KKK.GT.1) GO TO 555

C(1) = RRRR

GO TO 600

555 C(1) = C(1) * 2 ** 25

CALL FRA...EX(C, 100, U)

U = U0

U = U0 * 1 + U0

C = C + C(K)

IF (U.LT.5) GO TO 100

L = L0 + 1

GO TO 200

100 CONTINUE

L = L0

END
SUBROUTINE TRIDNA (N,MTA,D,A)

C TRIANGULARIZATION OF REAL SYMMETRIC MATRIX.

C DIMENSION A(1:M,1:N),D(1:N)

C SAVE ORIGINAL DIAGONALS IN ARRAY D.

C FOR N-2 RETURN WITHOUT COMPUTING.

C IF(N-1) RETURN WITHOUT COMPUTING.

C SUM CONTAINS THE SUM OF THE SQUARE ELEMENTS
C OF A COLUMN, EXCEPT THE FIRST K-2 ELEMENTS.

C THE J ARRAY CONTAINS THE BETA VALUES.

C IF(B(J=1)) 24,46,29

C THE COMPONENTS OF THE COLUMN VECTOR W ARE
C STORED IN THE POSITIONS OF THE ANNIHILATED

C THE BETAS ARE FORMED ONE BY ONE.
C WHEN R OF THE N HAVE BEEN FORMED.
C IF AN L CONTAINS ONLY (N-1) ELEMENTS
C THE J ARRAY IS USED TO STORE SUCCESSIVE
C P'S AND Q'S.

C U(J)=A(I)

C IF(J(J-I))+D(J)=D(J)

C 1 SUBROUTINE TRIDNA (N,MTA,D,A)
C 2 IMPLICIT REAL*4(A-H,Z-D)
C 3 C TRIANGULARIZATION OF REAL SYMMETRIC MATRIX.
C 4 C DIMENSION A(1:M,1:N),D(1:N)
C 5 C SAVE ORIGINAL DIAGONALS IN ARRAY D.
C 6 C FOR N-2 RETURN WITHOUT COMPUTING.
C 7 C IF(N-1) RETURN WITHOUT COMPUTING.
C 8 C SUM CONTAINS THE SUM OF THE SQUARE ELEMENTS
C 9 C OF A COLUMN, EXCEPT THE FIRST K-2 ELEMENTS.
C 10 C THE J ARRAY CONTAINS THE BETA VALUES.
C 11 C IF(B(J=1)) 24,46,29
C 12 C THE COMPONENTS OF THE COLUMN VECTOR W ARE
C 13 C STORED IN THE POSITIONS OF THE ANNIHILATED
C 15 C THE BETAS ARE FORMED ONE BY ONE.
C 16 C WHEN R OF THE N HAVE BEEN FORMED.
C 17 C IF AN L CONTAINS ONLY (N-1) ELEMENTS
C 18 C THE J ARRAY IS USED TO STORE SUCCESSIVE
C 19 C P'S AND Q'S.
C 20 C U(J)=A(I)
C 21 C IF(J(J-I))+D(J)=D(J)
TRANSFORM ALL ELEMENTS OF A, EXCEPT PIVOTAL ROW AND COLUMN.

RESHAPE ORIGINAL DIAGONALS OF A, STORE DIAGNALS OF TRANSFORMED MATRIX IN ARRAYS D.

- D.11 -
SUBROUTINE EIGVAL(LP, E, A, B, F, W)

IMPLICIT REAL*8(A-H,O-Z)

C LP IS THE SIZE OF ARRAY A.
C X IS A VECTOR OF LP ELEMENTS WHICH WILL HOLD
C THE EIGENVALUES IN DESCENDING ABSOLUTE ORDER.
C A IS A VECTOR OF LP ELEMENTS GIVING THE DIAGONAL
C ELEMENTS OF THE TRIDIAGONAL MATRIX.
C B IS A VECTOR OF LP ELEMENTS, THE LAST LP - 1
C IS A VECTOR OF LP ELEMENTS USED FOR TEMPORARY
C STORAGE.
C F IS A VECTOR OF LP ELEMENTS USED FOR TEMPORARY
C STORAGE.

DIMENSION E(LP), A(LP), B(LP), W(LP)

C THIS ABSOLUTE BOUND FOR THE EIGENVALUES

C THIS METHOD FORCES THE EIGENVALUES TO LIE BETWEEN:
C PLUS AND MINUS ONE, THE E AND X VECTORS ARE
C RESPECTIVELY LOW AND HIGH ESTIMATES TO ALL
C THE EIGENVALUES.

A(1) = (1)/UL
B(1) = 1/UL
L(1) = 1.0
W(1) = 1.0
UL = 50 * A(1)

C THIS IS THE K-TH EIGENVALUE. ALSO LOW AND HIGH
C ESTIMATES FOR THE K+1-ST TO LP-TH EIGENVALUES
C ARE IMPROVED, THE EIGENVALUES ARE FOUND IN
C ASCENDING ORDER.
C THE K-TH EIGENVALUE IS CONSIDERED FOUND IF THE
C HIGH AND LOW PLACES AGREE TO SEVEN DEIMAL
C PLACES.

D IF ((A(K) - E(K)) / MAX(ABS(A(K)) + GAMS(E(K))) < 1.E-29) = TRUE

C X IS A GUESS FOR THE K-TH EIGENVALUE, COMPUTE
C THE NUMBER OF EIGENVALUES EQUAL OR EXCEEDING X BY
C USING STURM SEQUENCE (OPTEGA'S METHOD).

C IF (I) = 1023, 104K*K

- D.12 -
D7

a.

SI=-1.0

EVAL

BO NO

EVAL

Z9

GO TO lub EVAL

IU4 b1--1.0

EVAL

, . NI,=

EVAL

l"a

I0

120

[=2LP

EVAL

u3 IF(U|I)1 106-113010t)

EVAL

b*

A.Uu IF (l6(-1,)) 107-114, 107 EVAL

oa) 1J7

IF

(1A)FS (I-)) Lj6BS(F (1-')I 1 11e12

EVAL

C IF

iE

PREVLU TWO

TERk S

OF THE

STUr%

SLOuFrjCE

EVAL

bEL

VERY

SMALL' THEY APE FORCED

TO

U

CLOSER

EVAL

C

TO ONE IN MAGNITUDE TO AVOID UNDERFLOW PROBLEMS. EVAL

7J

-------------

71

11 F([1]=1*[-1]=1.15

EVAL

7c

F(=2)=[1]*1.15

EVAL

73

F(1)=[(1)-x]*1+1)=1*1)1*31[1]+11-2)

EVAL

7d

uU Tu 120

EVAL

7j

112 F([1]=A([1)])=81

EVAL

7l

gu Tu 125

EVAL

7f

114 F([1]=A([1]-x)*1(1=1)15111(1)+11)51)

EVAL

7j

52211

EVAL

7v

116 IF(F(1))110+17+12

EVAL

7w

118 S1S16(51f (1))

EVAL

7x

1 If (51f=117-120+117)

EVAL

117

EVAL

7a

140 CUNITU

EVAL

7b

C

NO LET N BE THE NUMBER OF

EVAL

7c

EIGENVALUES SMALLER THAN X.

EVAL

7u

uU NUP=1.

EVAL

7v

IF(NLT,N) uu Tu 20

EVAL

7w

C

X BECOMES AN UPPER LIMIT FOR THE

EVAL

7x

K=TH TO N=TH EIGENVALUES.

EVAL

7y

uu

13 uu 13 uu

EVAL

7z

14 14 14

EVAL

8u

k1

EVAL

8v

l16

EVAL

97

C

IF ALL THE EIGENVALUES ARE SMALLER THAN X.

EVAL

98

TEST METHOD WE HAVE CONVERGED TO THE

EVAL

9u

K=TH EIGENVALUE.

EVAL

lu

k 1

IF LP,L1,N) Gu Tu b

EVAL

124

Gu 20 20LP

EVAL

EVAL

100

C

IF X IS LARGER THAN PREVIOUS LOWP

EVAL

100

C

LOWP INCREASE THE LOWER BOUND.

EVAL

114

k

(1- L(J) )Bred

EVAL

119

k

EVAL

134

b

CONTINUE

EVAL

115

C

RESTORE INPUT AND SCALE EIGENVALUES.

EVAL

- D.13 -
C
U0 60 I=1:LP
A(I)=A(I)+B(I)
B(I)=B(I)*B
W(I)=W(I)+L(I)*40*0.5
C
* SORT EIGENVALUES IN ABSOLUTE DESCENDING ORDER *
J=L
K=1
DO 80 I=1:LP
IF(DABS(A(I))=DABS(W(I)))E3+63+65
E3 E(I)=E(J)
J=J+1
GO TO 80
E8 L(I)=E(K)
K=K+1
CUNDI/L
RETURN
END
SUBROUTINE EIGVCL(LP, NM, R, A, B, EV, P, Q) EVEC

IMPLICIT REAL*4(A-H, I-O)

C R IS THE GIVEN MATRIX.
C LP IS DIMENSION OF THE MATRIX R.
C NM IS THE MAXIMUM DIMENSION OF R AND V.
C A AND THE DIAGONAL ELEMENTS OF THE TRIANGULARIZED R.
C B AND THE OFF-DIAGONAL ELEMENTS OF TRIANGULARIZED R.
C E ARE THE EIGENVECTORS OF R.
C V WILL HOLD THE EIGENVECTORS STORED COLUMNWISE.
C P AND Q ARE VECTORS FOR TEMPORARY STORAGE.
C D, Vائع(coefficients of the linear equations which
clet compute the eigenvectors.)
C
C DIMENSION R(LP, NM, A(LP), B(LP), E(LP), V[NM, LP], P(LP), Q(LP)) EVEC
C
C C GENERATE COEFFICIENTS TO COMPUTE THE VX=TM
C C EIGENVECTOR, FIRST PICK LAST PIVOT ELEMENT.
C
C C

LU 10 IF(A(I))=1.E-4
10 4 IF(J(I))=1.E-4
20 4 IF(LP(I))=1.E-4
10 4 IF(L(I))=1.E-4

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C
```plaintext
57  GO TO 20
58  20  V(LP,IX)=1,UL10
59  GO TO 20
60  30  X=R50/2/IX)
61  DO 31,2,1+LP
62  31  V(IX)=(V(IX)+/A)
63  C
64  C  TRANSFORM LGEVECTOR FOR THE TRIDAGONAL
65  C  MATRIX TO AN EIGENVector OF THE ORIGInAL MAtrix.
66  C  -----------------------------------------------
67  C
68  60  IF(LP.EQ.2) GO TO 50
69  70  UG 42 K=2+LP
70  71  K = LP + K + 1
71  72  100
72  73  30  T=BYV(I,IX)*K(IX-1)
73  74  UG 40 I=1+LP
74  75  40  V(I,IX)=1+K(I,K-1)
75  76  40  UG 40 I=1+LP
76  77  50  CONTINUE
77  78  RETURN
78  79  END

***

MULTI-BLOCK
INTERVAL LENGTH = 1.50  NUMBER OF INTERVALS = 60  TERMINATION CRITERION = .100000

- D.16 -
<table>
<thead>
<tr>
<th>ALPHA</th>
<th>BETA</th>
<th>PHI</th>
<th>NA</th>
<th>COST</th>
<th>EST.NC</th>
<th>MTTF</th>
<th>NA-MC</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.809</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
<td>200.00</td>
<td>1.634555</td>
<td>30.55</td>
<td></td>
</tr>
<tr>
<td>MINIMIZATION WITH ALPHA FIXED</td>
<td>DATE 06/11/79</td>
<td>PAGE 4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-------------------------------</td>
<td>---------------</td>
<td>-------</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Step Size Values</td>
<td>Limiting Lambda</td>
<td>Steepest Descent Values</td>
<td>Parameter Values</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>------------------</td>
<td>-----------------</td>
<td>-------------------------</td>
<td>------------------</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.0000</td>
<td>11.0000</td>
<td>0.0206</td>
<td>0.0000</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.0000</td>
<td>11.0000</td>
<td>0.0206</td>
<td>0.0000</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.0000</td>
<td>11.0000</td>
<td>0.0206</td>
<td>0.0000</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.0000</td>
<td>11.0000</td>
<td>0.0206</td>
<td>0.0000</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.0000</td>
<td>11.0000</td>
<td>0.0206</td>
<td>0.0000</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.0000</td>
<td>11.0000</td>
<td>0.0206</td>
<td>0.0000</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.0000</td>
<td>11.0000</td>
<td>0.0206</td>
<td>0.0000</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.0000</td>
<td>11.0000</td>
<td>0.0206</td>
<td>0.0000</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.0000</td>
<td>11.0000</td>
<td>0.0206</td>
<td>0.0000</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.0000</td>
<td>11.0000</td>
<td>0.0206</td>
<td>0.0000</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.0000</td>
<td>11.0000</td>
<td>0.0206</td>
<td>0.0000</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.0000</td>
<td>11.0000</td>
<td>0.0206</td>
<td>0.0000</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.0000</td>
<td>11.0000</td>
<td>0.0206</td>
<td>0.0000</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**DATE 06/11/79**

**PAGE 5**
The above sample output listing is given here only to illustrate the format as programmed in RELY I at the time of delivery. The values shown are of no significance relative to the content of the report (though they are those of a sample from Example IV, Sec. 3.3), nor are they expected to be repeatable exactly with implementation of RELY I at a different computer facility. The sample is offered as an aid to users, showing format and exemplary behavior in a given random case.