FATIGUE LIFE PROGRAM
USING STRAIN-LIFE METHODS

by

MICHAEL V. SKELLY

March, 1993

Thesis Advisor: GERALD H. LINDSEY

Approved for public release; distribution is unlimited

93-16924
A user friendly program was developed to calculate fatigue life using Strain-Life equations, given either a stress history or a strain history. Additionally, the material parameters and associated stress concentration factors can be varied. Since certain material constants, such as cyclic strength coefficient (K') and cyclic strain hardening exponent (n') vary during a material’s fatigue life, the program is capable of either keeping them constant or varying them as a function of elapsed cycles. The program was then utilised to examine the effects of varying K' and n' on the calculated fatigue life of aluminum 7075-T6 under a typical flight load history.
FATIGUE LIFE PROGRAM
USING STRAIN-LIFE METHODS

by

MICHAEL V. SKELLY
Lieutenant, United States Navy
B.S., Duke University, 1984

Submitted in partial fulfillment
of the requirements for the degree of

MASTER OF SCIENCE IN AERONAUTICAL ENGINEERING

from the

NAVAL POSTGRADUATE SCHOOL
March, 1993

Author:  Michael V. Skelly

Approved by:
Gerald H. Lindsay, Thesis Advisor
Conrad F. Newberry, Second Reader

Daniel J. Collins, Chairman
Department of Aeronautics and Astronautics
ABSTRACT

A user friendly program was developed to calculate fatigue life using Strain-Life equations, given either a stress or strain history. Additionally, the material parameters and associated stress concentration factors can be varied. Since certain material constants, such as cyclic strength coefficients (K') and strain hardening exponents (n') vary during a material's fatigue life, the program is capable of either keeping them constant or varying them as a function of elapsed cycles. The program was then utilized to examine the effects of varying K' and n' on the calculated fatigue life of aluminum 7075-T6 under a typical flight load history.
# TABLE OF CONTENTS

I. INTRODUCTION ................................................. 1

II. STRAIN-LIFE COMPUTATION PROGRAM ......................... 4  
   A. GENERAL .................................................. 4  
   B. MAIN PROGRAM ............................................ 4  
   C. USER SELECTABLE OPTIONS ................................. 5  
   D. LOAD HISTORY INPUT ..................................... 10  
   E. CALCULATIONS ............................................. 10  
   F. COMPUTATIONAL SUBROUTINES ............................... 11  
   G. DATA OUTPUT ............................................... 12  
   H. CODE VERIFICATION ....................................... 12  

III. MATERIAL DATA BASE .......................................... 15  
   A. CYCLIC PROPERTIES ....................................... 15  
   B. ARCHIVAL DATA ........................................... 16  

IV. LOAD GENERATING PROGRAM LOADGEN .......................... 18  
   A. LOAD SPECTRUM CONCEPT ................................ 18  
   B. A-6 STRESS SEQUENCE GENERATION ....................... 19  

V. APPROXIMATION OF K' AND n' ................................. 22  
   A. EXPERIMENTAL EFFORTS .................................. 22  

iv
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>B.</td>
<td>ESTIMATION OF K' AND n'</td>
<td>26</td>
</tr>
<tr>
<td>VI.</td>
<td>COMPUTATIONS</td>
<td>29</td>
</tr>
<tr>
<td>A.</td>
<td>COMPARISON PROCEDURE</td>
<td>29</td>
</tr>
<tr>
<td>B.</td>
<td>DISCUSSION OF RESULTS</td>
<td>29</td>
</tr>
<tr>
<td>VII.</td>
<td>DISCUSSION/CONCLUSIONS</td>
<td>32</td>
</tr>
<tr>
<td>APPENDIX A</td>
<td>FLP PROGRAM LISTING</td>
<td>33</td>
</tr>
<tr>
<td>A.</td>
<td>MAIN PROGRAM</td>
<td>33</td>
</tr>
<tr>
<td>B.</td>
<td>SUBROUTINE BATCH</td>
<td>37</td>
</tr>
<tr>
<td>C.</td>
<td>SUBROUTINE CRUNCHER</td>
<td>42</td>
</tr>
<tr>
<td>D.</td>
<td>SUBROUTINE DATADUMP</td>
<td>44</td>
</tr>
<tr>
<td>E.</td>
<td>SUBROUTINE EQUATIONS1</td>
<td>45</td>
</tr>
<tr>
<td>F.</td>
<td>SUBROUTINE EQUATIONS2</td>
<td>45</td>
</tr>
<tr>
<td>G.</td>
<td>SUBROUTINE EQUATIONS3</td>
<td>46</td>
</tr>
<tr>
<td>H.</td>
<td>SUBROUTINE EQUATIONS4</td>
<td>47</td>
</tr>
<tr>
<td>I.</td>
<td>SUBROUTINE EQUATIONS5</td>
<td>47</td>
</tr>
<tr>
<td>J.</td>
<td>SUBROUTINE GetConfig</td>
<td>48</td>
</tr>
<tr>
<td>K.</td>
<td>SUBROUTINE HEADER</td>
<td>49</td>
</tr>
<tr>
<td>L.</td>
<td>SUBROUTINE Klaxon</td>
<td>49</td>
</tr>
<tr>
<td>M.</td>
<td>SUBROUTINE LOADER</td>
<td>50</td>
</tr>
<tr>
<td>N.</td>
<td>SUBROUTINE Loadmaterial</td>
<td>51</td>
</tr>
<tr>
<td>O.</td>
<td>FUNCTION LOG10</td>
<td>51</td>
</tr>
<tr>
<td>P.</td>
<td>SUBROUTINE MATMENU</td>
<td>52</td>
</tr>
<tr>
<td>Q.</td>
<td>SUBROUTINE NeuberKf</td>
<td>54</td>
</tr>
</tbody>
</table>
APPENDIX B. PROGRAM LOADGEN

APPENDIX C. MATLAB DATA REDUCTION PROGRAM

APPENDIX D. MATERIAL DATA BASE

LIST OF REFERENCES
LIST OF FIGURES

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Figure 1</td>
<td>Options Menu 1</td>
<td>6</td>
</tr>
<tr>
<td>Figure 2</td>
<td>Option Menu 2</td>
<td>8</td>
</tr>
<tr>
<td>Figure 3</td>
<td>Material Entry Menu</td>
<td>9</td>
</tr>
<tr>
<td>Figure 4</td>
<td>Sample output summary</td>
<td>13</td>
</tr>
<tr>
<td>Figure 5</td>
<td>Sample step by step output</td>
<td>13</td>
</tr>
<tr>
<td>Figure 6</td>
<td>Hysteresis response of copper. (From Ref. 1)</td>
<td>16</td>
</tr>
<tr>
<td>Figure 7</td>
<td>Cycle by cycle load application</td>
<td>20</td>
</tr>
<tr>
<td>Figure 8</td>
<td>Fatigue test specimen</td>
<td>22</td>
</tr>
<tr>
<td>Figure 9</td>
<td>Stress-Strain Curves</td>
<td>25</td>
</tr>
<tr>
<td>Figure 10</td>
<td>Strain-Life Deviation</td>
<td>26</td>
</tr>
<tr>
<td>Figure 11</td>
<td>$K'$ and $n'$ Functions</td>
<td>28</td>
</tr>
</tbody>
</table>
I. INTRODUCTION

There are distinctly different approaches used to calculate cumulative fatigue damage during the crack initiation and crack propagation stages. Since the United States Navy, Naval Air Systems Command considers any cracked part to be "failed", the focus of this thesis was on crack initiation.

The definition of fatigue damage during the initiation phase of a crack's life is difficult. The damage during the crack initiation phase can be related to dislocations, slip bands, microcracks, etc, but the phenomena are microscopic and are not easily correlated with macroscopic measurements. Because of this, the damage summing methods typically used to calculate crack initiation are empirical in nature. They relate damage to life consumed. Life here refers to the physical separation of a small test specimen which is subsequently used to approximate crack initiation in larger aircraft components, since larger components will have a much larger critical crack length.

The most common method for summing damage is to use the linear damage rule, also known as Miner's rule:

$$\sum \frac{n_i}{N_i} \geq 1$$  \hspace{1cm} (1)
where \( n_i \) is the number of cycles applied at a given stress level and \( N_i \) is the total fatigue life for constant amplitude loading, at that stress level.

When applying Miner’s rule to a variable load history, the life used up at each load change has to be calculated. This can be done with one of several possible strain-life equations; for instance:

\[
\text{Morrow's: } \frac{\Delta \varepsilon}{2} = \frac{\sigma' - \sigma_0}{E} (2N_f)^b + \varepsilon'_f (2N_f)^c \tag{2}
\]

\[
\text{Manson-Halford: } \frac{\Delta \varepsilon}{2} = \frac{\sigma' - \sigma_0}{E} (2N_f)^b \varepsilon'_f \left( \frac{\sigma' - \sigma_0}{\sigma'_f} \right) \frac{\varepsilon'_f}{b} (2N_f)^c \tag{3}
\]

\[
\text{Smith-Watson-Topper: } \sigma_{\text{max}} \frac{\Delta \varepsilon}{2} = \left( \frac{\sigma'_{\text{f}}}{E} \right)^2 (2N_f)^{2b} + \sigma'_{\text{f}} \varepsilon'_f (2N_f)^{b+c} \tag{4}
\]

All three of these equations require the local change in strain and either the maximum or mean local stress. These can be obtained from relating the far field stress to the local stress using Neuber’s empirical rule:

\[
K_f^2 \sigma \varepsilon = \sigma \varepsilon \tag{5}
\]

Relating stress changes to strain changes can be done with the monotonic stress-strain equation:

\[
\varepsilon = \frac{\sigma}{E} + \left( \frac{\sigma}{K} \right)^{1/n} \tag{6}
\]
or the hysteresis equation:

\[ \frac{\Delta \varepsilon}{2} = \frac{\Delta \sigma}{2E} - \left( \frac{\Delta \sigma}{2K'} \right)^{\frac{1}{n'}} \]  \hspace{1cm} (7)

[Reference 1]

Programs are available to solve strain-life equations; however, a fatigue life program was needed for research at the Naval Postgraduate School, where the user can modify the solution algorithms or have access to the program’s source code to make changes and explore various facets of the theory.

For example, to evaluate the effects of varying the cyclic strength coefficient and the cyclic strain hardening exponent throughout the duration of applied loading, a program named FLP was developed. This program solved the three strain-life equations, (Equations (2) through (4),) using either fixed or varying values for \( n' \) and \( K' \) as specified by the user.

To develop a realistic sequence of loads from a known spectrum, LOADGEN was created. It was used to develop a realistic load history for an A-6 based on the three \( \sigma \) "g" count data developed by LT Rich Walters [Reference 2]. The load sequences were processed with the program using various strain-life equations.
II. STRAIN-LIFE COMPUTATION PROGRAM

A. GENERAL

The Fatigue Life Program (FLP) was written using Microsoft's QuickBasic. QuickBasic is a relatively simple programming language, but an updated and more capable version of BASIC, which allows the program to be compiled into an .EXE file, executable on any computer using MS-DOS or PC-DOS operating system 2.1 or later.

FLP was designed to be as "user friendly" as possible, and to the greatest extent practical, menus were used to set the various options. The program was built in a modular fashion and documentation was included throughout the program to facilitate debugging and later modifications.

B. MAIN PROGRAM

The main program controls the general processing flow. It starts by establishing certain constants and variables to be used throughout the program, and gives initial values to most of the user selected options, which could be considered default values. Examples of default values would be British units and aluminum 7075, which appear as being the units and material selected when the program is first started.

The program then enters a perpetual DO loop wherein it sets all the user definable options, reads in the specified
load sequence, processes the load sequence, and outputs processed data to the designated file before returning to the user definable options. The loop is exited and the program terminated by pressing ESCAPE when the option menus are being displayed.

The main program also sets the video configuration for the program, allowing it to automatically select the best mode available, and incorporating error trapping sequences to indicate when insufficient memory is available.

C. USER SELECTABLE OPTIONS

The selection of the various options available to the user is accomplished through several menus, each of which is a subroutine, or set of subroutines. There are two menus to define user selectable options, and a third to input properties for new materials. All three of these menus print a list of key instructions at the top of the screen, and call on another subroutine to update the display screen.

The first menu to appear is shown in Figure 1 and allows the operator to:

- choose between using British/U.S. (Brit) units or Standard International (SI) units;
- choose a local stress concentration factor ($K_t$);
- choose a method to calculate the fatigue stress concentration factor, either by use of Neuber's method, Peterson's method, or manual entry;
- choose another screen mode if more than one is available;
• choose a material from the existing material data base or manually enter a new material, which may be saved in the material data base;

• review the material properties associated with the selected material.

Prior to exiting this menu, the program checks to ensure that a value has been either entered or calculated for the fatigue stress concentration factor \( (K_f) \). If a value for \( K_f \) other than zero isn’t present, the program will prompt the user for one. The program may also be terminated normally from the first options menu by pressing **ESCAPE**.
The second menu of user selectable functions, shown in Figure 2, appears when the operator exits the first menu by pressing **RETURN**. It allows the operator to:

- choose between using stress or strains as input loads;
- choose which strain-life equation will be used, either Morrow's, Manson-Halford, or Smith-Watson-Topper;
- choose between using either fixed values for the cyclic strength coefficient and cyclic strain hardening exponent or using values that are a function of elapsed cycles;
- choose between calculating the cycles to failure, the effects of a single load block or processing a preprogrammed series of calculations;
- select the name of the input file containing the load history;
- select the name of the output file;
- activate and deactivate the program's sound.

The sound function mentioned above is to assist the program's user when processing a large batch of calculations or making a long life calculation. When activated, it will beep every time the program finishes processing a load block and sound an alarm when completely finished. From the second option menu the user has the choice of returning to the first option menu by pressing **ESCAPE** or starting the program's calculation subroutines by pressing **RETURN**.
The other menu used in the program is to facilitate operator entry of a new material. Called by the first option selection menu, and shown in Figure 3, it displays and allows the operator to update the following material properties:

- ultimate strength \( (S_u) \)
- yield strength \( (S_y) \)
- fatigue yield strength \( (S_y') \)
- strength coefficient \( (K) \)
- cyclic strength coefficient \( (K') \)
- strain hardening exponent \( (n) \)
- cyclic strain hardening exponent \( (n') \)
- ductility coefficient \( (\epsilon_f) \)
- fatigue ductility coefficient \( (\epsilon_f') \)
- strength coefficient \( (\sigma_f) \)
- fatigue strength coefficient \( (\sigma_f') \)
- fatigue strength exponent \( (b) \)
- fatigue ductility exponent \( (c) \)
- endurance strength \( (S_f) \)
- modulus of elasticity \( (E) \)

Figure 3 Material Entry Menu

Upon exiting this menu, the user is prompted for the newly entered material’s name and gives the option of saving the new material in the material data base.
D. LOAD HISTORY INPUT

The load history to be processed/analyzed is read into the program from the operator designated input file. This is accomplished by the subroutine LOADER. Additionally, this subroutine determines the number of loads in the sequence and whether the first increment is increasing or decreasing. The load input file is simply a consecutive list of either stresses or strains, one per line, in a DOS compatible format.

E. CALCULATIONS

The data processing takes place in the subroutine CRUNCHER. In this subroutine the far-field stress or strain load sequence is turned into a sequence of far-field stress or strain changes, according to what was initially read in. If the input file consisted of strain loads, the monotonic stress-strain equation, (Equation (6)) and the hysteresis equation, (Equation (7),) is used to convert strain changes to stress changes. Next using Neuber's rule, (Equation (5),) the far-field stress changes are used to find the local stress changes. The local stress changes and either the mean or maximum stress are then used in one of the three available strain-life equations, (Equations (2) through (4),) to determine the number of cycles to failure at the given load levels. This life is then used in conjunction with Miner's rule, (Equation (1)) to determine how much damage has
accumulated, or how much of the fatigue life was consumed, and added to a running counter. This process is repeated until all the loads are processed, or until failure, which occurs when the damage sums to one.

F. COMPUTATIONAL SUBROUTINES

Equations (2) through (6) can not be solved directly for the needed variables and require an iterative numerical solution. Newton's method of successive approximations was chosen as the simplest and most efficient method for this application. Newton's method is an iterative method which requires the function in question be evaluated at an initial estimate. If the estimate produces excessive error, the function's derivative is evaluated at the estimated value and a new estimate is obtained by subtracting from the old estimate the error at the old estimate divided by the derivative at the old estimate. [Reference 4]

Subroutines EQUATIONS1, EQUATIONS2, EQUATIONS3, EQUATIONS4, and EQUATIONS5 utilize Newton's method to solve equations (2) through (5) and (7). In EQUATIONS2, EQUATIONS3, and EQUATIONS4, which solve the strain-life equations, Newton's method was modified slightly to ensure that variable \( \text{NNf}_i \) being solved for was never approximated as being negative. (The variable \( \text{Nnf}_i \), which is raised to a negative
power, would produce a complex number which the program cannot accept.)

To allow for cycle to cycle variation in \(K'\) and \(n'\) the functions \(xxkf\) and \(xxnf\) were created to redefine the values of \(K'\) and \(n'\) based on the number of elapsed cycles (calculated by the function \(xxnnfcount\)).

Since QuickBasic has no built in function for base 10 logarithms, the function \(LOG10\) was incorporated to generate base ten logarithms from natural logarithms with the relationship shown:

\[
Log_{10}(x) = \frac{\ln(x)}{\ln(10)}
\]  

[Reference 5].

G. DATA OUTPUT

Depending on the processing option chosen, the output will consist of either a step by step print out of each variable's value for one trip through the input load block, (Figure 4,) and a summary of the chosen options, the number of cycles and load blocks until failure, or just the output summary, (Figure 5). The output is appended to the output file so that any previous data in the file will not be lost.

H. CODE VERIFICATION

To verify the programming code, the \(FLP\) was run for several short load histories (approximately 10 cycles). The first test sequences were constant amplitude loading till
"Morrow's equation"
"Fixed n' and K'"
"Load blocks to failure"
"input file": "test1"
"output file": "OUTPUT.DAT"
"blocks": 7
"i counter": 2482
"reversal count": 35655
"life factor": 1.000275410409951

Figure 4 Sample output summary.

<table>
<thead>
<tr>
<th>Index</th>
<th>Stress (ksi)</th>
<th>Delta Stress (ksi)</th>
<th>Delta sig (ksi)</th>
<th>sig0 (ksi)</th>
<th>Delta sig0 (ksi)</th>
<th>MNF</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>45</td>
<td>45.00</td>
<td>45.00</td>
<td>22.50</td>
<td>0.004366</td>
<td>0.15D+09</td>
</tr>
<tr>
<td>2</td>
<td>25</td>
<td>25.00</td>
<td>20.00</td>
<td>32.50</td>
<td>0.002428</td>
<td>0.16D+09</td>
</tr>
<tr>
<td>3</td>
<td>40</td>
<td>40.00</td>
<td>30.00</td>
<td>40.00</td>
<td>0.001942</td>
<td>0.18D+09</td>
</tr>
<tr>
<td>4</td>
<td>50</td>
<td>50.00</td>
<td>50.00</td>
<td>50.00</td>
<td>0.001402</td>
<td>0.20D+09</td>
</tr>
<tr>
<td>5</td>
<td>50</td>
<td>50.00</td>
<td>50.00</td>
<td>50.00</td>
<td>0.001402</td>
<td>0.20D+09</td>
</tr>
<tr>
<td>6</td>
<td>45</td>
<td>45.00</td>
<td>45.00</td>
<td>25.02</td>
<td>0.005800</td>
<td>0.20D+06</td>
</tr>
<tr>
<td>7</td>
<td>20</td>
<td>20.00</td>
<td>20.00</td>
<td>20.00</td>
<td>0.003984</td>
<td>0.17D+09</td>
</tr>
<tr>
<td>8</td>
<td>40</td>
<td>40.00</td>
<td>40.00</td>
<td>40.02</td>
<td>0.002355</td>
<td>0.14D+09</td>
</tr>
<tr>
<td>9</td>
<td>130</td>
<td>130.00</td>
<td>-89.98</td>
<td>24.98</td>
<td>0.012858</td>
<td>0.18D+05</td>
</tr>
<tr>
<td>10</td>
<td>60</td>
<td>150.00</td>
<td>60.02</td>
<td>14.98</td>
<td>0.015193</td>
<td>0.59D+04</td>
</tr>
<tr>
<td>11</td>
<td>-40</td>
<td>130.00</td>
<td>-40.02</td>
<td>-4.98</td>
<td>0.016000</td>
<td>0.59D+04</td>
</tr>
<tr>
<td>12</td>
<td>100</td>
<td>100.00</td>
<td>40.02</td>
<td>40.02</td>
<td>0.009749</td>
<td>0.15D+05</td>
</tr>
<tr>
<td>13</td>
<td>70</td>
<td>70.00</td>
<td>30.02</td>
<td>-9.98</td>
<td>0.005800</td>
<td>0.20D+06</td>
</tr>
<tr>
<td>14</td>
<td>30</td>
<td>30.02</td>
<td>-0.01</td>
<td>15.00</td>
<td>0.002915</td>
<td>0.15D+09</td>
</tr>
<tr>
<td>15</td>
<td>40</td>
<td>40.00</td>
<td>40.00</td>
<td>40.02</td>
<td>0.009749</td>
<td>0.15D+05</td>
</tr>
<tr>
<td>16</td>
<td>70</td>
<td>50.01</td>
<td>50.01</td>
<td>34.99</td>
<td>0.004686</td>
<td>0.10D+09</td>
</tr>
<tr>
<td>17</td>
<td>40</td>
<td>20.01</td>
<td>20.01</td>
<td>19.99</td>
<td>0.001942</td>
<td>0.11D+09</td>
</tr>
<tr>
<td>18</td>
<td>60</td>
<td>60.00</td>
<td>60.00</td>
<td>60.00</td>
<td>0.005800</td>
<td>0.11D+09</td>
</tr>
<tr>
<td>19</td>
<td>117</td>
<td>117.00</td>
<td>86.99</td>
<td>28.49</td>
<td>0.011474</td>
<td>0.91D+04</td>
</tr>
<tr>
<td>20</td>
<td>44</td>
<td>44.00</td>
<td>42.99</td>
<td>64.99</td>
<td>0.004272</td>
<td>0.14D+09</td>
</tr>
<tr>
<td>21</td>
<td>133</td>
<td>13.00</td>
<td>55.99</td>
<td>49.49</td>
<td>0.001262</td>
<td>0.31D+09</td>
</tr>
<tr>
<td>22</td>
<td>132</td>
<td>132.00</td>
<td>76.01</td>
<td>-10.01</td>
<td>0.013078</td>
<td>0.31D+05</td>
</tr>
<tr>
<td>23</td>
<td>80</td>
<td>80.00</td>
<td>3.99</td>
<td>-36.01</td>
<td>0.007775</td>
<td>0.15D+09</td>
</tr>
<tr>
<td>24</td>
<td>9</td>
<td>9.01</td>
<td>-5.02</td>
<td>-0.52</td>
<td>0.000875</td>
<td>0.17D+09</td>
</tr>
<tr>
<td>25</td>
<td>5</td>
<td>5.02</td>
<td>-0.01</td>
<td>-2.51</td>
<td>0.004887</td>
<td>0.20D+09</td>
</tr>
</tbody>
</table>

Figure 5 Sample step by step output.

failure and later load sequences were variable amplitude. The identical load sequences processed by hand with an HP-48 handheld calculator capable of solving complex equations numerically. The results were identical to four, usually five
or six, significant figures. This however didn’t test the
programs calculation of varying strength coefficient and
varying strain hardening exponents. To verify that these were
being correctly calculated the program was run in the
QuickBasic environment before being compiled into an
executable, (.EXE) file. When the program is being run in the
QuickBasic environment, the operator can pause execution at
any time and, using what’s referred to as the Immediate
window, execute other instructions [Reference 6]. Using this
feature and a hand held calculator, it was simple to pause the
program at random intervals between 1000 and 500,000 elapsed
cycles to ensure n’ and K’ had correct values.
A. CYCLIC PROPERTIES

In most metals the stress-strain response is altered by repeated loading. Depending on the material and its initial characteristics (i.e., quenched and tempered, or annealed,) a metal might either cyclically harden, soften, or have a mixed behavior. Typically a material which has a low yield strength relative to its ultimate tensile strength will undergo strain hardening while a hard material cyclically softens. This is reflected in the difference between a material’s monotonic strength coefficient (K) and its cyclic strength coefficient (K’) as well as the strain hardening exponent (n) and the cyclic strain hardening exponent (n’). Figure (6) shows the effects of cyclic loading on copper.

Normally, when performing strain-life calculations K’ and n’ are used in all the stress-strain calculations. For low to medium cycle fatigue, (10³ - 10⁵ cycles,) the effect of the change in these material properties is undocumented. To account for the dynamic nature of these properties, it was postulated that they could be expressed as a function of the elapsed cycles, allowing their value to be continually updated from cycle to cycle.
B. ARCHIVAL DATA

The program utilizes a separate file MAT.DAT to hold the materials available to the program without manual entry by the operator. Appendix Z shows the contents of the file MAT.DAT.

Figure 6 Hysteresis response of copper. (From Ref. 1)

The first entry in the material database indicates the number of materials in the original data base. This number couldn't be changed without restructuring the data base into a series of records so a second file NEWCOUNT was created just to record the number of new materials that were added to the data base. The number of materials within the data base is needed by the program to allow the option menu to cycle through all the materials.

The material properties used in the data base came from Metal Fatigue in Engineering by H. O. Fuchs [Reference 3].
The data base contains both the SI and British/American values for all properties and when new materials are added to the data base both the SI and British/American values are recorded. For numerous materials Reference 3 doesn't include values for $K$, $K'$ and $n$. To get around this potential problem, when any of these properties are missing the subroutine `Loadmaterial`, which selects the proper material from the data base and initializes the variables in the program to the appropriate material properties, will estimate the value of the missing property using the following relations:

\[
K = \frac{\sigma_f}{(\varepsilon_f)^a} \quad K' = \frac{\sigma_f'}{(\varepsilon_f')^{a'}} \quad n = \frac{b}{c} \quad (9)
\]

[Reference 1].

Review of the selected material's properties is accomplished by pressing F1 while the first option menu is displayed. This activates the display used for entering new materials, which will display the properties of the material that is already loaded. The displayed properties can't be modified to ensure the integrity of the original data base.
IV. LOAD GENERATING PROGRAM LOADGEN

A. LOAD SPECTRUM CONCEPT

In order to evaluate the effect of varying $n'$ and $K'$ on strain-life, a realistic load sequence was desired. Readily available was the load spectrum data representing the exceedences of a 0.9973 percentile (three sigma) A-6 aircraft, Table I, [Reference 2].

<table>
<thead>
<tr>
<th>Date of Data</th>
<th>Corrected Flt. Hours</th>
<th>4G PER 1000.0</th>
<th>5G 1000.0</th>
<th>6G 1000.0</th>
<th>7G 1000.0</th>
</tr>
</thead>
<tbody>
<tr>
<td>12/31/89</td>
<td>72.7</td>
<td>165</td>
<td>72</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1/31/90</td>
<td>48.7</td>
<td>166</td>
<td>27</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2/28/90</td>
<td>34.4</td>
<td>74</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>3/31/90</td>
<td>16.3</td>
<td>74</td>
<td>18</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>4/30/90</td>
<td>40.5</td>
<td>234</td>
<td>54</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>5/31/90</td>
<td>50.3</td>
<td>90</td>
<td>18</td>
<td>0</td>
<td>10</td>
</tr>
<tr>
<td>6/30/90</td>
<td>77.9</td>
<td>32</td>
<td>18</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>7/31/90</td>
<td>33.9</td>
<td>42</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>8/31/90</td>
<td>80.4</td>
<td>127</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>9/30/90</td>
<td>44.5</td>
<td>32</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>10/31/90</td>
<td>103.6</td>
<td>90</td>
<td>18</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>11/30/90</td>
<td>42.8</td>
<td>111</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>12/31/90</td>
<td>29.5</td>
<td>27</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1/31/91</td>
<td>4.1</td>
<td>106</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2/28/91</td>
<td>77.6</td>
<td>58</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>3/31/91</td>
<td>74.6</td>
<td>42</td>
<td>18</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>4/30/91</td>
<td>85.3</td>
<td>260</td>
<td>18</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>5/31/91</td>
<td>83.0</td>
<td>228</td>
<td>72</td>
<td>47</td>
<td>0</td>
</tr>
<tr>
<td>Totals</td>
<td>1000</td>
<td>1978.0</td>
<td>333.0</td>
<td>47.5</td>
<td>10.2</td>
</tr>
</tbody>
</table>

The number of exceedences for each $g$ level was entered into the program LOADGEN. This program would select a random number and then pick a corresponding 4g, 5g, 6g, or 7g load. The random number generator returned a number between 0 and 1,
which was compared to the percentage of g occurrences greater than 7, then greater than 6, and so on, until the random number is less than or equal to the percentage of occurrences in excess of a certain g level. The algorithm checks to ensure that all the g loads at that level haven't been depleted and decrements the counter for the respective bin. If all the loads at a particular g level are depleted and a random number was generated picking a depleted g level, then another random number is called until all the exceedences have been picked from the four g levels.

As each random number was generated, and a g level exceedence chosen, the program immediately calls another random number to determine the tenths value of the load to be inserted into the sequence. For example, if a 5g load was picked to be inserted into the sequence, the second random number would determine if it were a value of 5.0, 5.1, 5.2, etc. up to 5.9. The generated values were then sequenced in an output file for temporary storage. With the assumption that the load returned to 1 g at the end of every cycle, a sequence like the one in Figure 7 was generated.

B. A-6 STRESS SEQUENCE GENERATION

After the g sequence was developed, it was converted to a stress sequence for processing by FLP. Since the exact correlation between g load and the stress it produces on the A-6 wasn't known, the standard Navy design criteria was used
to calculate the highest acceptable corresponding stress values. The Navy's design criteria is such that the yield stress can't be exceeded at the design g limit, and the ultimate stress can't be exceeded at 1.5 times the design g limit. Since the design g limit of the A-6 is +6.5, in the worst case either a g loading of 6.5 corresponds to the material's yield strength, or a g loading of 9.75 corresponds to the material's ultimate strength. Given both the yield and ultimate strengths of the material, the program selects the limiting case and uses the corresponding g to stress ratio to convert the series of g loadings into a series of stress loadings. Since the g count information only gives maximums
and no minimums, the load profile was assumed to return to 1 g between each of the peaks.

As the file of g loads was transformed into stress loads, the stresses were written sequentially into a file for use by FLIP.
V. APPROXIMATION OF K' AND n'

A. EXPERIMENTAL EFFORTS

To account for the cyclic change in K' and n', these values were experimentally determined in test specimens whose life under a given cyclic strain load had already been determined. Specimens made from aluminum 7075-T6, shown in Figure 8, were tested in cyclic strain by LT Byron Smith in an investigation of the effects of means strain on strain life [reference 7]. Under fully reversed strain loading of ±.007 in/in, test specimens were cycled to 10%, 20%, 30%, and 40% of their predetermined total life. These specimens, and two

![Figure 8 Fatigue test specimen](image_url)
specimens without any cyclic preloading, were then subjected to a uniaxial stress-strain test to determine the properties in question.

The testing was performed on the Naval Postgraduate School, Mechanical Engineering Department's MTS machine. Data correlating to load and displacement were recorded continuously throughout the tests. This data was reduced to stress and strain, plotted in Figure 9, and then processed to determine the stress coefficients and strain hardening exponents at the time of testing.

Two methods of determining the properties in question were attempted. In the first the strain hardening exponents (n) were calculated using the following relationship from reference (1):

\[
at \text{necking: } n = \ln(1 + e) \tag{10}
\]

where \( e \) is the engineering strain.

Knowing \( n \), \( K \) can be determined with the relationship:

\[
\sigma = K(\epsilon_p)^n \quad \Rightarrow \quad K = \frac{\sigma}{(\epsilon_p)^n} \tag{11}
\]

where \( \sigma \) is the true stress and \( \epsilon_p \) is the true elastic strain.

Taking into consideration the following relationships:

\[
\epsilon_p = \epsilon_{\text{total}} - \epsilon_0 \tag{12}
\]

\[
\epsilon_0 = \frac{\sigma}{E} \tag{13}
\]
\[ \epsilon_{\text{total}} = \ln(1 + \varepsilon) \]  
\[ \sigma = S(1 + \varepsilon) \]  

where \( \varepsilon \) is the true elastic strain, and \( S \) is the engineering stress, this expression can be formed:

\[ K = \frac{S(1 + \varepsilon)}{\left( \ln(1 + \varepsilon) - \frac{S(1 + \varepsilon)}{E} \right)^n} \]

The resulting values for \( n' \) and \( K' \) appear in Table 2.

**TABLE 2: DATA REDUCTION RESULTS - METHOD 1**

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>( n )</td>
<td>( K ) (ksi)</td>
</tr>
<tr>
<td>0% prestrain #1</td>
<td>.160</td>
<td>131</td>
</tr>
<tr>
<td>0% prestrain #2</td>
<td>.155</td>
<td>130</td>
</tr>
<tr>
<td>10% prestrain</td>
<td>.020</td>
<td>93</td>
</tr>
<tr>
<td>20% prestrain</td>
<td>.155</td>
<td>131</td>
</tr>
<tr>
<td>30% prestrain</td>
<td>.151</td>
<td>130</td>
</tr>
<tr>
<td>40% prestrain</td>
<td>.155</td>
<td>132</td>
</tr>
</tbody>
</table>
The experimentally derived stress coefficient and strain hardening data doesn’t lend itself to generating a function based on elapsed cycles or used up strain-life. Looking at more of LT Smith’s work with strain fatigue in aluminum 7075-T6, shows a large deviation in the test results for strain life, (Figure 10). Because of the large deviation in the life of a specimen, there are large deviations in specified percentages of the sample’s life. This indicates the need to test a large number of samples to get results for K and n of any statistical significance.

Figure 9 Stress-Strain Curves
B. ESTIMATION OF $K'$ AND $n'$

Lacking statistically significant test results, for purposes of exercising this program both the strength coefficient and the strain hardening exponent were postulated to be:

- the monotonic values when less than 1000 cycles had elapsed,
- the cyclic values when more than 500,000 cycles had elapsed, (half the endurance life,) and
- a log-normal function of the number of elapsed cycles when between 1,000 and 500,000.
To calculate these properties between 1000 and 500,000 cycles the following equations were used:

\[ n_{vari} = \left( \frac{n' - n}{\log_{10}(500000) - \log_{10}(1000)} \right) \left( \log_{10}(x) - \log_{10}(1000) \right) + n \]
(17)

\[ K_{vari} = \left( \frac{K' - K}{\log_{10}(500000) - \log_{10}(1000)} \right) \left( \log_{10}(x) - \log_{10}(1000) \right) + K \]
(18)

For aluminum 7075-T6 these equations are plotted in Figure 11.
Figure 11: $K'$ and $n'$ Functions
VI. COMPUTATIONS

A. COMPARISON PROCEDURE

To compare the strain-lives calculated by using the cyclic properties with those obtained by varying the properties, a set of input load files were generated using the three σ limits of g load data for the A-6 and the LOADGEN program. Each file contained approximately 4800 points, representing 1000 hours of flight time. The subroutine BATCH was setup to repeat the strain-life calculation for each of the four load files, using each of the three strain-life equations, and the four following calculation methods:

- fixed n' and fixed K'
- fixed n' and variable K'
- variable n' and fixed K'
- variable n' and variable K'

This amounted to 48 sets of computations, the results of which are shown in Table 3.

B. DISCUSSION OF RESULTS

The data in Table 3 show a reduction in the calculated fatigue life of 15 to 30% when both the strength coefficient and the strain hardening exponent were varied from their monotonic values to their cyclic values. It is also clear
that the strength coefficient has a much greater impact on the fatigue life than does the strain hardening exponent. Variation of the strength coefficient consistently produced a significant drop in the fatigue life, while varying the strain hardening exponent very slightly raised the calculated fatigue life.

All three of the strain-life equations used to make the calculations behaved the same, though it seemed that the respective importance of the strain hardening exponent relative to the strength coefficient varied between the equations.

The effect of varying the strength coefficient can be anticipated. From Equation (6), a reduced value of \( K' \) translates into greater strain values locally, and greater strain means reduced life. However reduced \( n' \) values also mean larger strains and consequent reductions in life.

The effect of varying only the strain hardening could also be anticipated, but its impact, compared to that of the strength coefficient was unexpected.

It should be noted that these calculations assumed constant values for \( b \) and \( c \) in Equations (2) through (4), where in reality they would change. However this example has allowed this unique segment of the program to be exercised and to gain some preliminary estimates of the effect varying these parameters has on fatigue life.
### TABLE 3: CYCLES TO FAILURE

<table>
<thead>
<tr>
<th>Equation</th>
<th>Processing Option</th>
<th>Test0</th>
<th>Test1</th>
<th>Test2</th>
<th>Test3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Morrow’s</td>
<td>Fix n’; Fix K’</td>
<td>50554</td>
<td>49570</td>
<td>49708</td>
<td>50221</td>
</tr>
<tr>
<td></td>
<td>Var n’; Var K’</td>
<td>38723</td>
<td>33665</td>
<td>33661</td>
<td>37899</td>
</tr>
<tr>
<td></td>
<td>Var n’; Fix K’</td>
<td>50783</td>
<td>49737</td>
<td>50000</td>
<td>50453</td>
</tr>
<tr>
<td></td>
<td>Fix n’; Var K’</td>
<td>38723</td>
<td>33665</td>
<td>33661</td>
<td>38169</td>
</tr>
<tr>
<td>Smith- Watson-</td>
<td>Fix n’; Fix K’</td>
<td>39018</td>
<td>38404</td>
<td>38543</td>
<td>37899</td>
</tr>
<tr>
<td>Topper</td>
<td>Var n’; Var K’</td>
<td>33985</td>
<td>29979</td>
<td>31340</td>
<td>33114</td>
</tr>
<tr>
<td></td>
<td>Var n’; Fix K’</td>
<td>39162</td>
<td>38635</td>
<td>38779</td>
<td>38847</td>
</tr>
<tr>
<td></td>
<td>Fix n’; Var K’</td>
<td>33933</td>
<td>28927</td>
<td>30310</td>
<td>32072</td>
</tr>
<tr>
<td>Manson- Halford</td>
<td>Fix n’; Fix K’</td>
<td>47428</td>
<td>46483</td>
<td>46838</td>
<td>47346</td>
</tr>
<tr>
<td></td>
<td>Var n’; Var K’</td>
<td>34516</td>
<td>32166</td>
<td>33661</td>
<td>33729</td>
</tr>
<tr>
<td></td>
<td>Var n’; Fix K’</td>
<td>47587</td>
<td>46924</td>
<td>47320</td>
<td>47429</td>
</tr>
<tr>
<td></td>
<td>Fix n’; Var K’</td>
<td>33985</td>
<td>28927</td>
<td>33661</td>
<td>33729</td>
</tr>
</tbody>
</table>
VII. DISCUSSION/CONCLUSIONS

The FLP originally conceived has been written and appears to function well. It possesses flexibility in the manner in which it performs fatigue life calculations, using any one of three documented strain-life equations which account for mean stress effects. The initial exploration done with it supports the concept that fatigue life, especially in the lower end of the fatigue spectrum, might be better calculated if changes in material properties are accounted for. To do this, however, material properties need to be well defined as functions of elapsed cycles. The algorithms used in the FLP varied only two of the material's properties as a function of elapsed cycles, $K'$ and $n'$. Other material properties could be varied in addition to the ones currently varied by the FPL. Four candidates are: the fracture strength coefficient ($\sigma_f$), the ductility coefficient ($\epsilon_f$), the fatigue strength exponent ($b$) and the fatigue ductility exponent ($c$). These coefficients have been shown to vary with elapsed cycles as well, and their effect should be explored in future tests and incorporated into FLP.
APPENDIX A  FLP PROGRAM LISTING

A. MAIN PROGRAM

DEFSNG A-Z
DEFINT I-P
DECLARE SUB BATCH ()
DECLARE SUB LOADER ()
DECLARE SUB CRUNCHER ()
DECLARE SUB EQUATIONS1 ()
DECLARE SUB EQUATIONS2 ()
DECLARE SUB EQUATIONS3 ()
DECLARE SUB EQUATIONS4 ()
DECLARE SUB EQUATIONS5 ()
DECLARE SUB OPTMENU1 ()
DECLARE SUB OPTMENU2 ()
DECLARE SUB MATMENU ()
DECLARE SUB UPDATEMENU ()
DECLARE SUB UPDATEMENU2 ()
DECLARE SUB Loadmaterial (index)
DECLARE SUB NEWMAT (Ky)
DECLARE SUB OUTPUTER ()
DECLARE SUB HEADER ()
DECLARE SUB DATADUMP ()
DECLARE SUB PetersonKf ()
DECLARE SUB NeuberKf ()
DECLARE SUB GetConfig ()
DECLARE SUB TYPIN (cstring$, valu)
DECLARE SUB TYPINSTRING (cstring$, stringname$)
DECLARE SUB Klaxon (Hi%, low%)
DECLARE FUNCTION xxnf! ()
DECLARE FUNCTION xxKf! ()
DECLARE FUNCTION xxNNfcount&
DECLARE FUNCTION Rotated (Lower, Upper AS INTEGER, present, Inc)
DECLARE FUNCTION LOG10! (value!)

REM $DYNAMIC

' Constants for best available screen mode
CONST VGA = 12
CONST MCGA = 13
CONST EGA256 = 9
CONST EGA64 = 8
CONST MONO = 10
CONST HERC = 3
CONST CGA = 1
TYPE Config
  Scrn  AS INTEGER
  Colors AS INTEGER
  Atribs AS INTEGER
  XPix AS INTEGER
  YPix AS INTEGER
  TCOL AS INTEGER
  TROW AS INTEGER
END TYPE

DIM VC AS Config

variables shared throughout the program

COMMON SHARED YD, i, top, Kt AS SINGLE, Ktf AS SINGLE, option4 AS STRING
COMMON SHARED matcount, matindex, matname AS STRING, Su, Sy, Syf, K AS SINGLE
COMMON SHARED epsf, epsff, sigf, sigff, b, c, Sf, SfSu, E, newstuff
COMMON SHARED flag1 AS INTEGER, flag2 AS INTEGER, flag3 AS INTEGER
COMMON SHARED flag4 AS INTEGER, flag5 AS STRING, batchno AS INTEGER
COMMON SHARED nKtype AS STRING, calctype AS STRING, lgNNfcount AS SINGLE
COMMON SHARED usedlife AS DOUBLE, block AS INTEGER, lasti AS INTEGER
COMMON SHARED NNfcount AS LONG, xKf AS SINGLE, xnf AS SINGLE
COMMON SHARED inputfile AS STRING, outputfile AS STRING, sigmax AS SINGLE
COMMON SHARED laststress AS SINGLE, strainmax AS SINGLE, lasteps AS SINGLE
COMMON SHARED lastsig AS SINGLE

size = 5000

DIM SHARED stress(size) AS SINGLE, strain(size), sig(size) AS SINGLE
DIM SHARED deltaeps(size) AS SINGLE, NNf(size) AS DOUBLE, sig0(size)
DIM SHARED delastress(size) AS SINGLE, deltastrain(size) AS SINGLE
DIM SHARED deltasig(size), eps(size)

TYPE Options1
  units  AS STRING * 4
  material AS INTEGER
  Kt    AS SINGLE
  Ktf   AS INTEGER
  option5 AS INTEGER
END TYPE

DIM Menu AS Options1

TYPE Options2
  inputs AS STRING * 6
  equation AS INTEGER
  option3 AS INTEGER  'nKtype selection
  option4 AS INTEGER  'calculation type
END TYPE

DIM menu2 AS Options2

Error variables to check screen type
DIM InitRows AS INTEGER, BestMode AS INTEGER, Available AS STRING
' Initialize variables

Menu.units = "Brit"
Menu.material = 21
Menu.Kt = 1
Menu.Ktf = 1
Menu.option5 = 1
option4 = "Manually Entered"
Kt = 1
Ktf = 0

menu2.inputs = "Stress"
menu2.equation = 1
menu2.option3 = 1
menu2.option4 = 1

equationname = "Morrow's equation"
nKtype = " Fixed n' and K'"
calctype = "Load blocks to failure"
inputfile = "STRESS.DAT"
outputfile = "OUTPUT.DAT"

flag2 = 1
flag3 = 0
flag4 = 0
flag5 = "OFF"  
  initialises sound on
batchno = 0
sigmax = 0
strainmax = 0
laststress = 0
lastsig = 0
lasteps = 0

  Get best configuration and set initial graphics mode to it
GetConfig
VC.Scrn = BestMode

DO   ' endless do loop for main program (exited from menu 1)
       
  Selection of material/constants
  IF flag4 = 0 THEN
    DO
      CALL OPTMENU1
      CALL OPTMENU2
      LOOP UNTIL flag2 = 1
    END IF

    IF menu2.option4 = 3 THEN
      batchno = batchno + 1
      BATCH
    END IF

    CALL LOADER
    CALL CRUNCHER
    CALL OUTPUTER

      END

35
IF flag4 = 0 THEN
  IF flag5 = "OFF" THEN PRINT "Finished Processing.....press any key"
  IF flag5 = "OFF" THEN CALL Klaxon(987, 329)
  CLS
ELSE
  PRINT USING "Finished Processing Batch ##"; batchno
END IF

SIGMA = 0
STRAINMAX = 0
LASTSTRESS = 0
LASTSTEP = 0
LASTSIG = 0

LOOP UNTIL cowscomehome
END

' Error trap to make program screen independent
VideoErr:
  SELECT CASE BestMode  ' Fall through until something works
    CASE VGA
      BestMode = MCGA
      Available = "12BD"
    CASE MCGA
      BestMode = EGA256
      Available = "12789"
    CASE EGA256
      BestMode = CGA
      Available = "12"
    CASE CGA
      BestMode = MONO
      Available = "A"
    CASE MONO
      BestMode = HERC
      Available = "3"
    CASE ELSE
      PRINT "Sorry. Graphics not available."
  END SELECT
RESUME

' Trap to detect 64K EGA
EGAErr:
  BestMode = EGA64
  Available = "12789"
RESUME NEXT

' Trap to detect insufficient memory
MemErr:
  LOCATE 22, 1
  PRINT "Out of memory"
  RESUME NEXT
' Trap to determine initial number of rows so they can be restored
RowErr:
   IF InitRows = 50 THEN
      InitRows = 43
   RESUME
   ELSE
      InitRows = 25
   RESUME NEXT
   END IF

'================= end of main program + error trapping ==================

B. SUBROUTINE BATCH

REM $STATIC
'================================== BATCH ==================================
' Subroutine to access a file for repeated runs
' (currently set to process 4 file, with 3 eqns, 4 methods
'================================== Batch===================================
SUB BATCH
SHARED menu2 AS Options2

IF batchno = 1 THEN
   inputfile = "testaa"
   nKtype = " Fixed n' and K' "
   menu2.option3 = 1
   menu2.equation = 1
   equationname = "Morrow's Equation"
ELSEIF batchno = 2 THEN
   inputfile = "testaa"
   nKtype = "Variable n' and K' "
   menu2.option3 = 2
   menu2.equation = 1
   equationname = "Morrow's Equation"
ELSEIF batchno = 3 THEN
   inputfile = "testaa"
   nKtype = "Variable n' and fixed K' "
   menu2.option3 = 3
   menu2.equation = 1
   equationname = "Morrow's Equation"
ELSEIF batchno = 4 THEN
   inputfile = "testaa"
   nKtype = "Fixed n' and variable K'"
   menu2.option3 = 4
   menu2.equation = 1
   equationname = "Morrow's Equation"
ELSEIF batchno = 5 THEN
   inputfile = "testbb"
   nKtype = " Fixed n' and K' "
   menu2.option3 = 1
   menu2.equation = 1
   equationname = "Morrow's Equation"
ELSEIF batchno = 6 THEN
inputfile = "testbb"
nKtype = "Variable n' and K'"
menu2.option3 = 2
menu2.equation = 1
equationname = "Morrow's Equation"
ELSEIF batchno = 7 THEN
inputfile = "testbb"
nKtype = "Variable n' and fixed K'"
menu2.option3 = 3
menu2.equation = 1
equationname = "Morrow's Equation"
ELSEIF batchno = 8 THEN
inputfile = "testbb"
nKtype = "Fixed n' and variable K'"
menu2.option3 = 4
menu2.equation = 1
equationname = "Morrow's Equation"
ELSEIF batchno = 9 THEN
inputfile = "testcc"
nKtype = "Fixed n' and K'"
menu2.option3 = 1
menu2.equation = 1
equationname = "Morrow's Equation"
ELSEIF batchno = 10 THEN
inputfile = "testcc"
nKtype = "Variable n' and K'"
menu2.option3 = 2
menu2.equation = 1
equationname = "Morrow's Equation"
ELSEIF batchno = 11 THEN
inputfile = "testcc"
nKtype = "Variable n' and fixed K'"
menu2.option3 = 3
menu2.equation = 1
equationname = "Morrow's Equation"
ELSEIF batchno = 12 THEN
inputfile = "testcc"
nKtype = "Fixed n' and variable K'"
menu2.option3 = 4
menu2.equation = 1
equationname = "Morrow's Equation"
ELSEIF batchno = 13 THEN
inputfile = "testdd"
nKtype = "Fixed n' and K'"
menu2.option3 = 1
menu2.equation = 1
equationname = "Morrow's Equation"
ELSEIF batchno = 14 THEN
inputfile = "testdd"
nKtype = "Variable n' and K'"
menu2.option3 = 2
menu2.equation = 1
equationname = "Morrow's Equation"
ELSEIF batchno = 15 THEN
inputfile = "testdd"
nKtype = "Variable n' and fixed K'"
menu2.option3 = 3
menu2.equation = 1
equationname = "Morrow's Equation"
ELSEIF batchno = 16 THEN
    inputfile = "testdd"
    nKtype = "Fixed n' and variable K'"
    menu2.option3 = 4
    menu2.equation = 1
    equationname = "Morrow's Equation"
ELSEIF batchno = 17 THEN
    inputfile = "testaa"
    nKtype = "Fixed n' and K'"
    menu2.option3 = 1
    menu2.equation = 2
    equationname = "Smith-Watson-Topper"
ELSEIF batchno = 18 THEN
    inputfile = "testaa"
    nKtype = "Variable n' and K'"
    menu2.option3 = 2
    menu2.equation = 2
    equationname = "Smith-Watson-Topper"
ELSEIF batchno = 19 THEN
    inputfile = "testaa"
    nKtype = "Variable n' and fixed K'"
    menu2.option3 = 3
    menu2.equation = 2
    equationname = "Smith-Watson-Topper"
ELSEIF batchno = 20 THEN
    inputfile = "testaa"
    nKtype = "Fixed n' and variable K'"
    menu2.option3 = 4
    menu2.equation = 2
    equationname = "Smith-Watson-Topper"
ELSEIF batchno = 21 THEN
    inputfile = "testbb"
    nKtype = "Fixed n' and K'"
    menu2.option3 = 1
    menu2.equation = 2
    equationname = "Smith-Watson-Topper"
ELSEIF batchno = 22 THEN
    inputfile = "testbb"
    nKtype = "Variable n' and K'"
    menu2.option3 = 2
    menu2.equation = 2
    equationname = "Smith-Watson-Topper"
ELSEIF batchno = 23 THEN
    inputfile = "testbb"
    nKtype = "Variable n' and fixed K'"
    menu2.option3 = 3
    menu2.equation = 2
    equationname = "Smith-Watson-Topper"
ELSEIF batchno = 24 THEN
    inputfile = "testbb"
    nKtype = "Fixed n' and variable K'"
    menu2.option3 = 4
    menu2.equation = 2
    equationname = "Smith-Watson-Topper"
ELSEIF batchno = 25 THEN
    inputfile = "testcc"
    nKtype = "Fixed n' and K'"
    menu2.option3 = 1
    menu2.equation = 2
    equationname = "Smith-Watson-Topper"
ELSEIF batchno = 26 THEN
  inputfile = "testcc"
  nKtype = "Variable n' and K'
  menu2.option3 = 2
  menu2.equation = 2
  equationname = "Smith-Watson-Topper"
ELSEIF batchno = 27 THEN
  inputfile = "testcc"
  nKtype = "Variable n' and fixed K'
  menu2.option3 = 3
  menu2.equation = 2
  equationname = "Smith-Watson-Topper"
ELSEIF batchno = 28 THEN
  inputfile = "testcc"
  nKtype = "Fixed n' and variable K'
  menu2.option3 = 4
  menu2.equation = 2
  equationname = "Smith-Watson-Topper"
ELSEIF batchno = 29 THEN
  inputfile = "testcc"
  nKtype = "Fixed n' and fixed K'
  menu2.option3 = 3
  menu2.equation = 2
  equationname = "Smith-Watson-Topper"
ELSEIF batchno = 30 THEN
  inputfile = "testcc"
  nKtype = "Variable n' and K'
  menu2.option3 = 2
  menu2.equation = 2
  equationname = "Smith-Watson-Topper"
ELSEIF batchno = 31 THEN
  inputfile = "testcc"
  nKtype = "Variable n' and fixed K'
  menu2.option3 = 3
  menu2.equation = 2
  equationname = "Smith-Watson-Topper"
ELSEIF batchno = 32 THEN
  inputfile = "testcc"
  nKtype = "Fixed n' and variable K'
  menu2.option3 = 4
  menu2.equation = 2
  equationname = "Smith-Watson-Topper"
ELSEIF batchno = 33 THEN
  inputfile = "testcc"
  nKtype = "Fixed n' and fixed K'
  menu2.option3 = 3
  menu2.equation = 2
  equationname = "Smith-Watson-Topper"
ELSEIF batchno = 34 THEN
  inputfile = "testcc"
  nKtype = "Variable n' and K'
  menu2.option3 = 2
  menu2.equation = 3
  equationname = "Manson-Halford"
ELSEIF batchno = 35 THEN
  inputfile = "testcc"
  nKtype = "Variable n' and fixed K'
  menu2.option3 = 3
  menu2.equation = 3
  equationname = "Manson-Halford"
ELSEIF batchno = 36 THEN
    inputfile = "testaa"
nKtype = "Fixed n’ and variable K’"
menu2.option3 = 4
menu2.equation = 3
equationname = "Manson-Halford"
ELSEIF batchno = 37 THEN
    inputfile = "testbb"
nKtype = "Fixed n’ and K’"
menu2.option3 = 1
menu2.equation = 3
equationname = "Manson-Halford"
ELSEIF batchno = 38 THEN
    inputfile = "testbb"
nKtype = "Variable n’ and K’"
menu2.option3 = 2
menu2.equation = 3
equationname = "Manson-Halford"
ELSEIF batchno = 39 THEN
    inputfile = "testbb"
nKtype = "Variable n’ and fixed K’"
menu2.option3 = 3
menu2.equation = 3
equationname = "Manson-Halford"
ELSEIF batchno = 40 THEN
    inputfile = "testbb"
nKtype = "Fixed n’ and variable K’"
menu2.option3 = 4
menu2.equation = 3
equationname = "Manson-Halford"
ELSEIF batchno = 41 THEN
    inputfile = "testcc"
nKtype = "Fixed n’ and K’"
menu2.option3 = 1
menu2.equation = 3
equationname = "Manson-Halford"
ELSEIF batchno = 42 THEN
    inputfile = "testcc"
nKtype = "Variable n’ and K’"
menu2.option3 = 2
menu2.equation = 3
equationname = "Manson-Halford"
ELSEIF batchno = 43 THEN
    inputfile = "testcc"
nKtype = "Variable n’ and fixed K’"
menu2.option3 = 3
menu2.equation = 3
equationname = "Manson-Halford"
ELSEIF batchno = 44 THEN
    inputfile = "testcc"
nKtype = "Fixed n’ and variable K’"
menu2.option3 = 4
menu2.equation = 3
equationname = "Manson-Halford"
ELSEIF batchno = 45 THEN
    inputfile = "testdd"
nKtype = "Fixed n’ and K’"
menu2.option3 = 1
menu2.equation = 3
equationname = "Manson-Halford"
ELSEIF batchno = 46 THEN
  inputfile = "testdd"
  nKtype = "Variable n' and K'"
  menu2.option3 = 2
  menu2.equation = 3
  equationname = "Manson-Halford"
ELSEIF batchno = 47 THEN
  inputfile = "testdd"
  nKtype = "Variable n' and fixed K'"
  menu2.option3 = 3
  menu2.equation = 3
  equationname = "Manson-Halford"
ELSEIF batchno = 48 THEN
  inputfile = "testdd"
  nKtype = "Fixed n' and variable K'"
  menu2.option3 = 4
  menu2.equation = 3
  equationname = "Manson-Halford"
  flag4 = 0
END IF
END SUB

C. SUBROUTINE CRUNCHER

REM $DYNAMIC
'===------------------------ CRUNCHER ===------------------------
' subroutine to calculate Strain-Life using the various methods
'===------------------------------------------------
SUB CRUNCHER

DIM lastload AS SINGLE
SHARED menu2 AS Options2

OPEN outputfile FOR APPEND AS #7

'initialize variables
usedlife = 0
block = -1
lastload = 0

DO ' loop until life is used up if single block option not chosen
  block = block + 1
  IF block = 0 THEN 'calculate delta loads and print output header if appropriate
    IF menu2.option4 = 2 THEN HEADER ' output headers
    IF menu2.inputs = "Stress" THEN
      calculation of farfield deltastress array
      FOR ii = 1 TO top
        deltaStress(ii) = ABS(stress(ii) - lastload)
        lastload = stress(ii)
      NEXT ii
      ELSE

42
calculation of far field deltastrain array

FOR ii = 1 TO top
    deltastrain(ii) = ABS(strain(ii) - lastload)
    lastload = strain(ii)
NEXT ii

END IF

END IF

load blocks processed

FOR i = 1 TO top

    set properties based on elapsed cycles

    NNfcount = xxNNfcount
    lgNNfcount = LOG10(CSNG(NNfcount))
    xKf = xxKf
    xnf = xxnf

    turn farfield strains into stresses

    IF menu2.inputs = "Strain" THEN CALL EQUATIONS5

    find local Stress associated with farfield stress

    CALL EQUATIONS1

    calculate mean local stress

    sig(i) = lastsig + ((-1) ^ (flagl + i)) * deltasig(i)
    sig0(i) = (lastsig + sig(i)) / 2
    lastsig = sig(i)

    finding epsilon and delta epsilon

    IF sig(i) > sigmax THEN

        monotonotic equation to find local strains

        eps(i) = sig(i) / E + (sig(i) / (xKf)) ^ (1 / xnf)
        deltaeps(i) = ABS(eps(i) - lasteps)
        lasteps = eps(i)
        sigmax = sig(i)

    ELSE

        hysteresis equation to find local strains

        deltaeps(i) = deltasig(i) / E + 2 * (deltasig(i) / (2 * xKf)) ^ (1 / xnf)
        eps(i) = lasteps + ((-1) ^ (flagl + i)) * deltaeps(i)
        lasteps = eps(i)

    END IF

    pick between various strain-life eqns

    SELECT CASE menu2.equation

    CASE 1
        CALL EQUATIONS2
    CASE 2
        CALL EQUATIONS3
    CASE 3
        CALL EQUATIONS4
    CASE ELSE
    END SELECT

    print running output to file

    IF menu2.option4 = 2 THEN DATADUMP

43
IF usedlife > 1 THEN
    lasti = i
    EXIT FOR
END IF
NEXT i

IF flag5 = "OFF" THEN BEEP
LOOP UNTIL usedlife >= 1 OR (menu2.option4 = 2 AND block = 1)
CLOSE #7
END SUB

D. SUBROUTINE DATADUMP

REM $STATIC
'====================================================================
' subroutine to write output data
'====================================================================
SUB DATADUMP
SHARED menu2 AS Options2
IF menu2.inputs = "Stress" THEN
    PRINT #7, USING "### ### ### ###.## ####.######.## #.###4### #.#####*#.##...";
    stress(i); deltaStress(i); deltasig(i); sig(i); sig0(i); deltaeps(i); eps(i); NNf(i)
ELSE
    PRINT #7, USING "### #.### #.### ###.### ###.######.## #.###### #.###### #.##..";
    strain(i); deltastrain(i); deltaStress(i); deltasig(i); sig(i); sig0(i); deltaeps(i);
    eps(i); NNf(i)
END IF
END SUB
E. SUBROUTINE EQUATIONS1

' SUBROUTINE TO FIND DELTA SIGMA USING NEWTON'S METHOD
' (BASED ON NEUBER'S RELATION)

DIM Yprime AS DOUBLE

Jeltasig(i) = 1
loopcount = 0

DO
    loopcount = loopcount + 1
    Y = Jeltasig(i) * (Jeltasig(i) / (2 * E) + (Jeltasig(i) / (2 * xKf)) ^ (1 / xnf)) - (Ktf * 2) * deltaStress(i) * (deltaStress(i) / (2 * E) + (deltaStress(i) / (2 * xKf)) ^ (1 / xnf))
    IF ABS(Y) > .0000001 THEN
        Yprime = Jeltasig(i) / E + ((xnf + 1) / xnf) * (Jeltasig(i) / (2 * xKf)) ^ (1 / xnf)
        Jeltasig(i) = Jeltasig(i) - Y / Yprime
    END IF
LOOP UNTIL (ABS(Y) <= .0000001) OR (loopcount = 1000)

END SUB

F. SUBROUTINE EQUATIONS2

' SUBROUTINE TO EVALUATE MORROW'S STRAIN-LIFE EQUATION

DIM Yprime AS DOUBLE, Y AS DOUBLE, loopcount AS LONG

NNf(i) = 10000
loopcount = 0

DO
    loopcount = loopcount + 1
    Y = -(deltaeps(i) / 2 + ((sigff - sig0(i)) / E) * (2 * NNf(i)) ^ b + epsff * (2 * NNf(i)) ^ c)
    IF ABS(Y) > .0000000001# THEN
        Yprime = (b * (sigff - sig0(i)) / E) * (2 ^ b) * (NNf(i)) ^ (b - 1) + c * epsff * (2 ^ c) * (NNf(i)) ^ (c - 1)
        IF (Y / Yprime < NNf(i)) THEN
            NNf(i) = NNf(i) - Y / Yprime
        ELSE
            NNf(i) = NNf(i) / 2
        END IF
    END IF
END IF

45
LOOP UNTIL (ABS(Y) <= .0000000001#) OR (NNf(i) > 100000000) OR (loopcount=10000)

usedlife = usedlife + 1 / NNf(i)

IF loopcount = 10000 THEN PRINT "**" 'indication of convergence difficulties

END SUB

G. SUBROUTINE EQUATIONS3

REM $STATIC
'
EQUATIONS3 =============================
' subroutine to evaluate Smith Watson Topper Strain-Life equation
'
EQUATIONS3 =============================
REM $DYNAMIC

DIM Yprime AS DOUBLE

NNf(i) = 100
loopcount = 0

DO
    loopcount = loopcount + 1
    Y = -(sigO(i) + deltasig(i) / 2) * (deltaeps(i) / 2) + ((sigff ^ 2) / E) * (2 * NNf(i)) ^ (2 * b + sigff * epsff * (2 * NNf(i)) ^ (b + c))

    IF ABS(Y) > .0000000001# THEN
        Yprime = (2 * b * (sigff ^ 2) / E) * (2 ^ (2 * b - 1)) * NNf(i) ^ (2 * b - 1) + (b + c) * sigff * epsff * (2 ^ (b + c)) * NNf(i) ^ (b + c - 1)
        IF (Y / Yprime < NNf(i)) THEN
            NNf(i) = NNf(i) - Y / Yprime * .5
        ELSE
            NNf(i) = NNf(i) / 2.5
        END IF
    END IF
LOOP UNTIL (ABS(Y) <= .0000000001#) OR (NNf(i) > 100000000) OR (loopcount = 10000)

usedlife = usedlife + 1 / NNf(i)

IF loopcount = 10000 THEN PRINT "**" 'indication of convergence difficulties

END SUB
H. SUBROUTINE EQUATIONS4

' ===================================================== EQUATIONS4' subroutine to evaluate Manson-Halford's Strain-Life equation
' =====================================================

SUB EQUATIONS4
REM $DYNAMIC

DIM Yprime AS DOUBLE

NNf(i) = 100
loopcount = 0

DO
loopcount = loopcount + 1
Y = -deltaeps(i) / 2 + ((sigff - sig0(i)) / E) * (2 * NNf(i)) ^ b + epsff * ((sigff - sig0(i)) / sigff) ^ (b / c) * (2 * NNf(i)) ^ c
IF ABS(Y) > .0000000001# THEN
Yprime = (b * (sigff - sig0(i)) / E) * (2 ^ b) * NNf(i) ^ (b - 1) + c * epsff * ((sigff - sig0(i)) / sigff) ^ (b / c) * (2 ^ c) * NNf(i) ^ (c - 1)
ELSE
NNf(i) = NNf(i) - Y / Yprime * .5
END IF
END IF

LOOP UNTIL (ABS(Y) <= .0000000001#) OR (NNf(i) > 100000000) OR (loopcount=1000)
u sedlife = usedlife + 1 / NNf(i)

IF loopcount = 10000 THEN PRINT "*" 'indication of convergence difficulties
END SUB

I. SUBROUTINE EQUATIONS5

REM $STATIC
' ===================================================== EQUATIONS5' subroutine to find the far-field stress associated with the
' far-field strain inputs
' =====================================================

SUB EQUATIONS5
DIM Yprime AS DOUBLE

deltaStress(i) = 0
loopcount = 0

IF strain(i) < strainmax THEN
DO
loopcount = loopcount + 1
Y = -deltastrain(i) + deltaStress(i) / E + 2 * (deltaStress / (2 * xKf)) ^ (1 / xnf)


47
IF ABS(Y) > .0001 THEN
    Yprime = 1 / E + (1 / (2 * xKf)) ^ (1 / xnf) * (2 / xnf) * deltaStress
            ^ (1 / xnf - 1)
deltaStress(i) = deltaStress(i) - Y / Yprime
END IF
LOOP UNTIL (ABS(Y) <= .0001) OR (loopcount = 1000)

' calculate far field stress from change in stress
stress(i) = laststress + ((-1) ^ (flagl + i)) * deltaStress(i)
laststress = stress(i)
ELSE
    DO
        loopcount = loopcount + 1
        Y = -strain(i) + stress(i) / E + (stress / xKf) ^ (1 / xnf)
        IF ABS(Y) > .0001 THEN
            Yprime = 1 / E + (1 / (xKf)) ^ (1 / xnf) * (1 / xnf - 1)
stress(i) = stress(i) - Y / Yprime
        END IF
        LOOP UNTIL (ABS(Y) <= .0001) OR (loopcount = 1000)
    
    ' cal. farfield stress change and reset strainmax
    strainmax = strain(i)
deltaStress(i) = ABS(stress(i) - laststress)
    laststress = stress(i)
END IF

END SUB

J. SUBROUTINE GetConfig

DEFSNG I-P

' == GetConfig ==
' Get the starting number of lines and the video adapter.
' ==
'
SUB GetConfig STATIC
    SHARED InitRows AS INTEGER, BestMode AS INTEGER, Available AS STRING

    ' Assume 50 line display and fall through error
    ' until we get the actual number
    InitRows = 50
    ON ERROR GOTO RowErr
    LOCATE InitRows, 1

    ' Assume best possible screen mode
    BestMode = VGA
    Available = "12789BCD"

    ON ERROR GOTO VideoErr
    ' Fall through error trap until a mode works
    SCREEN BestMode
    ' If EGA, then check pages to see whether more than 64K
    ON ERROR GOTO EGAErr
    IF BestMode = EGA256 THEN SCREEN 8, , 1
ON ERROR GOTO 0

' Reset text mode
SCREEN 0, , 0
WIDTH 80, 25

END SUB

K. SUBROUTINE HEADER

DEFINT I-P
,'
', HEADER
', subroutine to an appropriate output header
', HEADER
'
SUB HEADER

SHARED menu2 AS Options2

IF menu2.inputs = "Stress" THEN
   PRINT #7, "index Stress deltaStress deltasig sig sig0
   deltaeps eps NNf "
ELSE
   PRINT #7, "index strain dstrn deltaStress deltasig sig sig0
   deltaeps eps NNf "
END IF

END SUB

L. SUBROUTINE Klaxon

DEFINT H
', Klaxon
', subroutine activates a two tone alarm until a key is pressed
', Klaxon
'
SUB Klaxon (Hi%, low%) STATIC

   DO WHILE INKEY$ = ""
      SOUND Hi, 5
      SOUND low, 5
   LOOP

END SUB
M. SUBROUTINE LOADER

REM $DYNAMIC
DEFSNG H-P
'=======================================================================
' subroutine to read in the array of loads from the file "STRESS.DAT"
' or "STRAIN.DAT" as appropriate
' also -- "top" is set to the highest index in the load array and
' -- "flagl" is set to indicate if the loads are initially
' increasing or decreasing
'=======================================================================
SUB LOADER
SHARED menu2 AS Options2
OPEN inputfile FOR INPUT AS #10
j = 1
IF menu2.inputs = "Stress" THEN
    DO UNTIL EOF(10)
        INPUT #10, stress(j)
        j = j + 1
    LOOP
    set "top" to the number of loads:
    top = j - 1
    IF stress(1) > 0 THEN
        flagl = 1
    ELSE
        flagl = 0
    END IF
ELSE
    DO UNTIL EOF(10)
        INPUT #10, strain(j)
        j = j + 1
    LOOP
    set "top" to the number of loads:
    top = j - 1
    IF strain(1) > 0 THEN
        flagl = 1
    ELSE
        flagl = 0
    END IF
END IF
CLOSE #10
END SUB
N. SUBROUTINE Loadmaterial

REM $STATIC
DEFINT I-P
'http finds the selected material in the data base and loads the
'material values for the selected material
'also calculates any critical missing values it can
'SUB Loadmaterial (index)

'OPEN "A:\MAT.DAT" FOR INPUT AS #3
'OPEN "B:\MAT.DAT" FOR INPUT AS #3
OPEN "MAT.DAT" FOR INPUT AS #3

INPUT #3, count

DO UNTIL index = matindex
    INPUT #3, matindex, matname$, Su, Sy, Syf, K, Kf, n, nf, epsf, epsff, sigf, sigff, b, c, Sf, SfSu, E
LOOP

CLOSE #3

IF n = 0 AND b <> 0 AND c <> 0 THEN n = b / c
IF nf = 0 AND b <> 0 AND c <> 0 THEN nf = b / c
IF K = 0 AND n <> 0 AND epsf <> 0 THEN K = sigf / (epsf ^ n)
IF Kf = 0 AND nf <> 0 AND epsff <> 0 THEN Kf = sigff / (epsff ^ nf)

END SUB

O. FUNCTION LOG10

'http Returns the base 10 logarithm for the specified value
'FUNCTION LOG10! (value)
LOG10 = LOG(value) / LOG(10)

END FUNCTION
P. SUBROUTINE MATMENU

'======================================== MATMENU ==========================
' subroutine to update the screen when entering or viewing materials
' in the materials database
'======================================== MATMENU ==========================

SUB MATMENU

CONST ENTER = 13, ESCAPE = 27
CONST DOWNARROW = 80, UPARROW = 72, LEFTARROW = 75, RIGHTARROW = 77
CONST COL1 = 10, COL2 = 50, ROW = 7
CONST COL3 = 7, COL4 = 62
CONST Fields = 14

SHARE Menu AS Options1
DIM forceunit AS STRING, Fld AS INTEGER

CLS
IF flag3 = 0 THEN
  ' Display key instructions
  LOCATE 1, COL1
  PRINT "UP ............... Move to next field"
  LOCATE 2, COL1
  PRINT "DOWN ........ Move to previous field"
  LOCATE 3, COL1
  PRINT "LEFT or RIGHT .... Enter a new value"
  LOCATE 4, COL1
  PRINT "ENTER ... Return with current values"
  LOCATE 5, COL1
  PRINT "ESCAPE .... Quit material data entry"
ELSE 'display material’s name at top of screen
  LOCATE 2, COL1
  PRINT "ESCAPE ....to return to previous menu"
  LOCATE 5, COL1
  PRINT USING "Data base parameters for: &"; matname
END IF

IF (Menu.material <= 51) THEN
  forceunit = "ksi)"
ELSE
  forceunit = "MPa)"
END IF

' Display fields
LOCATE ROW, COL3: PRINT "Ultimate Strength
LOCATE ROW, COL4: PRINT USING "[####]"; Su;
LOCATE ROW + 1, COL3: PRINT "Yield Strength
LOCATE ROW + 1, COL4: PRINT USING "[####]"); Sy;
LOCATE ROW + 2, COL3: PRINT "Fatigue Yield Strength
LOCATE ROW + 2, COL4: PRINT USING "[####]"); Syf;

52
LOCATE ROW + 3, COL3: PRINT "Strength Coefficient (K in forceunit); K;
LOCATE ROW + 4, COL3: PRINT "Cyclic Strength Coefficient (K' in forceunit); Kf;
LOCATE ROW + 5, COL3: PRINT "Strain Hardening Exponent (n)"); n;
LOCATE ROW + 6, COL3: PRINT "Cyclic Strain Hardening Exponent (n')"); nf;
LOCATE ROW + 7, COL3: PRINT "Ductility Coefficient (epsilon f)"); epsf;
LOCATE ROW + 8, COL3: PRINT "Fatigue Ductility Coefficient (epsilon f')"); epsff;
LOCATE ROW + 9, COL3: PRINT "Strength Coefficient (sigma f in forceunit); sigf;
LOCATE ROW + 10, COL3: PRINT "Fatigue Strength Coefficient (sigma f' in forceunit); sigff;
LOCATE ROW + 11, COL3: PRINT "Fatigue strength Exponent (b)"); b;
LOCATE ROW + 12, COL3: PRINT "Fatigue Ductility Exponent (c)"); c;
LOCATE ROW + 13, COL3: PRINT "Endurance Strength (Sf in forceunit); Sf;
LOCATE ROW + 14, COL3: PRINT "Modulus of Elasticity (E in forceunit); E;

END SUB
Q. SUBROUTINE NeuberKf

'==================================== NeuberKf ==================================='
' subroutine to calculate the fatigue stress concentration factor
' based on Neuber's method
'==================================== NeuberKf ==================================='
SUB NeuberKf
CALL TYPIN("Enter notch sensitivity factor (q; 0 if unknown)", q)
IF q <> 0 THEN
    Ktf = 1 + q * (Kt - 1)
ELSE
    CALL TYPIN("Enter the notch root radius (r)", r)
    CALL TYPIN("Enter the appropriate material constant (rho)", rho)
    Ktf = 1 + (Kt - 1) / (1 + (rho / r)^.5)
END IF
END SUB

R. SUBROUTINE NEWMAT

'==================================== NEWMAT ======================================
' subroutine to manually enter material constants and
' enter new materials into the materials database
'==================================== NEWMAT ======================================
SUB NEWMAT (Ky)
CONST ENTER = 13, ESCAPE = 27
CONST DOWNARROW = 80, UPARROW = 72, LEFTARROW = 75, RIGHTARROW = 77
CONST COL1 = 10, COL2 = 50, ROW = 7
CONST COL3 = 7, COL4 = 62
CONST Fields = 14
' conversion constant from "ksi" to "MPa"
CONST ab = 6.89476

SHARED Menu AS Options1, save$
DIM forceunit AS STRING, Fld AS INTEGER

IF (Menu.material = 1) THEN
    forceunit = "ksi"
ELSE
    forceunit = "MPa"
END IF
CALL MATMENU
' Update field values and position based on keystrokes
DO
    Put cursor on field
    LOCATE ROW + Fld, COL4 + 2

54
Get a key and strip null off if it's an extended code

DO
Key$ = INKEY$
LOOP WHILE Key$ = ""

Ky = ASC(RIGHT$(Key$, 1))

SELECT CASE Ky
CASE ENTER
  ' Check for all appropriate data parameters
  ' before allowing return to main option menu
  IF b = 0 OR c = 0 OR sigff = 0 OR epsff THEN
    LOCATE 23, 10: PRINT "Insufficient material parameters were entered"
    END IF
CASE UPARROW, DOWNARROW
  ' Adjust field location
  IF Ky = DOWNARROW THEN Inc = 1 ELSE Inc = -1
  Fld = Rotated(0, Fields, Fld, Inc)
CASE RIGHTARROW, LEFTARROW
  ' Adjust field value
  IF Ky = RIGHTARROW THEN Inc = 1 ELSE Inc = -1
  SELECT CASE Fld
CASE 0
  ' Ultimate Strength
  CALL TYPIN("Enter Ultimate Strength (Su)"", Su)
  CALL MATMENU
CASE 1
  ' Yield Strength
  CALL TYPIN("Enter Yield Strength (Sy)"", Sy)
  CALL MATMENU
CASE 2
  ' Fatigue Yield Strength
  CALL TYPIN("Enter Fatigue Yield Strength (Sy')", Syf)
  CALL MATMENU
CASE 3
  ' Strength Coefficient
  CALL TYPIN("Enter Strength Coefficient (K)"", K)
  CALL MATMENU
CASE 4
  ' Cyclic Strength Coefficient
  CALL TYPIN("Enter Cyclic Strength Coefficient (K')", Kf)
  CALL MATMENU
CASE 5
  ' Cyclic Strength Coefficient
  CALL TYPIN("Strian Hardening Exponent (n)", n)
  CALL MATMENU
CASE 6
  ' Cyclic Strain Hardening Exponent
  CALL TYPIN("Cyclic Strain Hardening Exponent (n')", nf)
  CALL MATMENU
CASE 7
  ' Ductility Coefficient (epsilon f)
  CALL TYPIN("Ductility Coefficient (epsilon f)", epsf)
  CALL MATMENU

CASE 8
  ' Fatigue Ductility Coefficient (epsilon f')
  CALL TYPIN("Fatigue Ductility Coefficient (epsilon f')", epsff)
  CALL MATMENU

CASE 9
  ' Strength Coefficient (sigma f)
  CALL TYPIN("Strength Coefficient (sigma f)", sigf)
  CALL MATMENU

CASE 10
  ' Fatigue Strength Coefficient (sigma f')
  CALL TYPIN("Fatigue Strength Coefficient (sigma f')", sigff)
  CALL MATMENU

CASE 11
  ' Fatigue Strength Exponent (b)
  CALL TYPIN("Fatigue Strength Exponent (b)", b)
  CALL MATMENU

CASE 12
  ' Fatigue Ductility Exponent (c)
  CALL TYPIN("Fatigue Ductility Exponent (c)", c)
  CALL MATMENU

CASE 13
  ' Endurance Strength (Sf)
  CALL TYPIN("Endurance Strength (Sf)", Sf)
  CALL MATMENU

CASE 14
  ' Modulus of Elasticity (E)
  CALL TYPIN("Modulus of Elasticity (E)", E)
  CALL MATMENU

CASE ELSE
END SELECT

CASE ELSE
END SELECT

' exit do loop and continue with subroutine if ENTER
LOOP UNTIL (Ky = ENTER OR Ky = ESC)

CLS
LOCATE 13, 5: PRINT "Enter Material's Name (up to 30 characters)": "
LOCATE 13, 52: INPUT matname$ 

LOCATE 15, 5: PRINT "Save this material in material data base (Y/N)": "
LOCATE 15, 54: INPUT save$
program segment to save new materials in the material data base

IF save$ = "Y" OR save$ = "y" THEN
    'OPEN "A:\MAT.DAT" FOR APPEND AS #3
    'OPEN "B:\MAT.DAT" FOR APPEND AS #3
    OPEN "MAT.DAT" FOR APPEND AS #3

    matcount = matcount + 1
    IF forceunit = "ksi") THEN
        matindex = matcount
        WRITE #3, matindex, matname$, Su, Sy, Syf, K, Kf, n, nf, epsf, epsff,
            sigf, sigff, b, c, Sf, SfSu, E
        WRITE #3, matindex + 50, matname$, Su * ab, Sy * ab, Syf * ab, K * ab,
            Kf * ab, n, nf, epsf, epsff, sigf * ab, sigff * ab, b, c, Sf
            * ab, SfSu, E * ab
    ELSE
        matindex = matcount + 50
        WRITE #3, matindex, matname$, Su, Sy, Syf, K, Kf, n, nf, epsf, epsff,
            sigf, sigff, b, c, Sf, SfSu, E
        WRITE #3, matindex - 50, matname$, Su / ab, Sy / ab, Syf / ab, K / ab,
            Kf / ab, n, nf, epsf, epsff, sigf / ab, sigff / ab, b, c, Sf
            / ab, SfSu, E / ab
    END IF

CLOSE #3

'OPEN "A:\NEWCOUNT.DAT" FOR OUTPUT AS #4
'OPEN "B:\NEWCOUNT.DAT" FOR OUTPUT AS #4
OPEN "NEWCOUNT.DAT" FOR OUTPUT AS #4
newstuff = newstuff + 1
WRITE #4, newstuff
CLOSE #4

END IF

CLS
CALL UPDATEMENU
Ky = 0

END SUB

S. SUBROUTINE OPTMENU1

REM $DYNAMIC
'==================================== Option Menu 1 =====================================
' Define/swtich the user selectable functions for the material and
stress concentration factors
'====================================
SUB OPTMENU1 STATIC

SHARED VC AS Config, Menu AS Options1, Available AS STRING, Fields AS INTEGER
SHARED Fld AS INTEGER

' Constants for key codes and column positions
CONST ENTER = 13, ESCAPE = 27, F1 = 59
CONST DOWNARROW = 80, UPARROW = 72, LEFTARROW = 75, RIGHTARROW = 77
CONST COL1 = 10, COL2 = 50, ROW = 9
CONST COL3 = 7, COL4 = 42

57
'OPEN "A:\MAT.DAT" FOR INPUT AS #3
'OPEN "B:\MAT.DAT" FOR INPUT AS #3
OPEN "MAT.DAT" FOR INPUT AS #3
INPUT #3, matcount
CLOSE #3
'OPEN "A:\NEWCOUNT.DAT" FOR INPUT AS #4
'OPEN "B:\NEWCOUNT.DAT" FOR INPUT AS #4
OPEN "NEWCOUNT.DAT" FOR INPUT AS #4
INPUT #4, newstuff
CLOSE #4
matcount = matcount + newstuff
CALL Loadmaterial(Menu.material)

' Block cursor
LOCATE ROW, COL1, 1, 1, 12
CALL UPDATEMENU

' Skip field 10 if there's only one value
IF LEN(Available$) = 1 THEN Fields = 8 ELSE Fields = 10

' Update field values and position based on keystrokes
DO
' Put cursor on field
LOCATE ROW + Fld, COL4 + 2

' Get a key and strip null off if it's an extended code
DO
Key$ = INKEY$
LOOP WHILE Key$ = ""
Ky = ASC(RIGHT$(Key$, 1))

SELECT CASE Ky
CASE ESCAPE
   ' End program
   CLS : END
CASE F1
   ' review material parameters
   flag3 = 1
   CALL MATMENU
   flag3 = 0
   DO
      DO
         Key2$ = INKEY$
         LOOP WHILE Key2$ = ""
         Ky2 = ASC(RIGHT$(Key2$, 1))
         LOOP UNTIL Ky2 = ESCAPE
         CLS
         CALL UPDATEMENU
   CASE ENTER
      ' runs material parameter entry subroutine
      IF Menu.material = 1 OR Menu.material = 51 THEN NEWMAT (Ky)
   CASE UPARROW, DOWNARROW
      ' Adjust field location
      IF Ky = DOWNARROW THEN Inc = 2 ELSE Inc = -2
      Fld = Rotated(0, Fields, Fld, Inc)
CASE RIGHTARROW, LEFTARROW
' Adjust field value
IF Ky = RIGHTARROW THEN Inc = 1 ELSE Inc = -1
SELECT CASE Fld

CASE 0
' Units
IF Menu.units = "Brit" THEN
  Menu.units = "SI"
  Menu.material = Menu.material + 50
  CALL Loadmaterial(Menu.material)
  CALL UPDATEMENU
ELSE
  Menu.units = "Brit"
  Menu.material = Menu.material - 50
  CALL Loadmaterial(Menu.material)
  CALL UPDATEMENU
END IF

CASE 2
' Material Selection
IF Menu.units = "Brit" THEN
  Menu.material = Rotated(1, matcount, Menu.material, Inc)
  CALL Loadmaterial(Menu.material)
  PRINT USING "&"; matname
ELSE
  Menu.material = Rotated(51, matcount + 50, Menu.material, Inc)
  CALL Loadmaterial(Menu.material)
  PRINT USING "&"; matname
END IF

CASE 4
' Stress Concentration Factor Kt
CALL TYPIN("Enter stress concentration factor (Kt): ", Kt)
CALL UPDATEMENU

CASE 6
' Fatigue Stress Concentration Calculation Method
Menu.Ktf = Rotated(1, 3, Menu.Ktf, Inc)
SELECT CASE Menu.Ktf
  CASE 1
    ' manual entry'
    option4 = "Manually Entered"
    LOCATE ROW + 8, COL4: PRINT USING "&"; "[ --- ]
    LOCATE ROW + 6, COL4 + 2: PRINT USING "&"; option4
  CASE 2
    ' Neubers Method'
    option4 = "Neuber's Method"
    LOCATE ROW + 8, COL4: PRINT USING "&"; "[ --- ]
    LOCATE ROW + 6, COL4 + 2: PRINT USING "&"; option4
  CASE 3
    ' Peterson Method'
    option4 = "Peterson's Method"
    LOCATE ROW + 8, COL4: PRINT USING "&"; "[ --- ]
    LOCATE ROW + 6, COL4 + 2: PRINT USING "&"; option4
  CASE ELSE
END SELECT

59
CASE 8
   ' Fatigue Stress Concentration Factor displayed
SELECT CASE Menu.Ktf
   CASE 1
      ' manual entry
      CALL TYPIN("Enter fatigue stress concentration factor (Ktf): ", Ktf)
      CALL UPDATEMENU
   CASE 2
      ' Neubers Method
      CALL NeuberKf
      CALL UPDATEMENU
   CASE 3
      ' Peterson Method
      CALL PetersonKf
      CALL UPDATEMENU
   CASE ELSE
   END SELECT
CASE 10
   ' Available screen modes
   i = INSTR(Available$, HEX$(VC.Scrn))
   i = Rotated(1, LEN(Available$), i, Inc)
   VC.Scrn = VAL("&h" + MIDS(Available$, i, 1))
   PRINT USING "##'; VC.Scrn
   CASE ELSE
   END SELECT
CASE ELSE
END SELECT
CASE ELSE
END SELECT

' if clause to ensure selection of a fatigue stress concentration factor
IF Ky = ENTER AND Ktf = 0 THEN
   Fld = 8
   CALL TYPIN("Enter fatigue stress concentration factor (Ktf): ", Ktf)
   CALL UPDATEMENU
   Ky = 0
END IF

' Return to main program if ENTER
LOOP UNTIL (Ky = ENTER AND Ktf <> 0 AND matname <> "new material")

END SUB
T. SUBROUTINE OPTMENU2

' == Option Menu 2 ==
' Define/switch the user selectable functions for processing options
' ==
SUB OPTMENU2 STATIC

SHARED VC AS Config, menu2 AS Options2, Available AS STRING, Fields AS INTEGER
SHARED Fld AS INTEGER

' Constants for key codes and column positions
CONST ENTER = 13, ESCAPE = 27, F2 = 60
CONST DOWNARROW = 80, UPARROW = 72, LEFTARROW = 75, RIGHTARROW = 77
CONST COL1 = 10, COL2 = 50, ROW = 9
CONST COL3 = 7, COL4 = 42

' Return cursor to menu top
Fld = 0
CALL UPDATEMENU2

' Skip field 10 if there's only one value
IF LEN(Available$) = 1 THEN Fields = 8 ELSE Fields = 10

' Update field values and position based on keystrokes
DO
  ' Put cursor on field
  LOCATE ROW + Fld, COL4 + 2

  ' Get a key and strip null off if it's an extended code
  DO
    Key$ = INKEY$
    LOOP WHILE Key$ = ""
  END

  Ky = ASC(RIGHT$(Key$, 1))

  SELECT CASE Ky
    CASE ESCAPE
      ' End program
      CLS : END
      CLS : flag2 = 0
    CASE F2
      ' changes sound to condition show in flag5
      IF flag5 = "ON" THEN
        flag5 = "OFF"
      ELSE
        flag5 = "ON"
      END IF
      UPDATEMENU2
    CASE UPARROW, DOWNARROW
      ' Adjust field location
      IF Ky = DOWNARROW THEN Inc = 2 ELSE Inc = -2
      Fld = Rotated(0, Fields, Fld, Inc)
    CASE RIGHTARROW, LEFTARROW
      ' Adjust field value
      IF Ky = RIGHTARROW THEN Inc = 1 ELSE Inc = -1
      SELECT CASE Fld

61
CASE 0
  ' type of inputs (stress vs strain)
  IF menu2.inputs = "Stress" THEN
    menu2.inputs = "Strain"
    inputfile = "STRAIN.DAT"
  ELSE
    menu2.inputs = "Stress"
    inputfile = "STRESS.DAT"
  END IF
  UPDATEMENU2

CASE 2
  ' Equation Selection
  menu2.equation = Rotated(1, 3, menu2.equation, Inc)
  SELECT CASE menu2.equation
    CASE 1
      ' Morrow's equation
      equationname = "Morrow's equation"
    CASE 2
      ' Smith Watson Topper
      equationname = "Smith-Watson-Topper"
    CASE 3
      ' Manson Halford
      equationname = "Manson-Halford"
  END SELECT
  UPDATEMENU2

CASE 4
  ' variable/fixed n' and K'
  menu2.option3 = Rotated(1, 4, menu2.option3, Inc)
  SELECT CASE menu2.option3
    CASE 1
      ' fixed n and K
      nKtype = "Fixed n' and K'"
    CASE 2
      ' variable n' and K'
      nKtype = "Variable n' and K'"
    CASE 3
      ' variable n' and fixed K'
      nKtype = "Variable n' and fixed K'"
    CASE 4
      ' fixed n' and variable K'
      nKtype = "Fixed n' and variable K'"
    CASE 5
      ' Experimental - not used
      nKtype = "Experimental"
  END SELECT
  UPDATEMENU2

CASE 6
  ' calculation type
  menu2.option4 = Rotated(1, 3, menu2.option4, Inc)
  SELECT CASE menu2.option4
    CASE 1
      ' blocks to failure
      calctype = "Load blocks to failure"
    CASE 2
      ' single block effects
      calctype = "Single block effects"
CASE 3
  ' batch process
  calctype = "Batch Process"
  flag4 = 1
  CASE ELSE
  END SELECT
  UPDATEMENU2

CASE 8
  ' input data file name
  CALL TYPINSTRING("Input data file's name:", inputfile)
  UPDATEMENU2

CASE 10
  ' output data file name
  CALL TYPINSTRING("Output data file's name:", outputfile)
  UPDATEMENU2

CASE ELSE
  END SELECT

CASE ELSE
  END SELECT

' if clause to ensure selection of a fatigue stress concentration factor
IF Ky = ENTER AND Ktf = 0 THEN
  Fld = 8
  CALL TYPIN("Enter fatigue stress concentration factor (Ktf): ", Ktf)
  CALL UPDATEMENU
  Ky = 0
END IF

' Return to main program if ENTER
LOOP UNTIL (Ky = ENTER AND Ktf <> 0 AND matname <> "new material") OR (Ky = ESCAPE)

CLS
END SUB
U. SUBROUTINE OUTPUTER

REM $STATIC
'==================================== OUTPUTER =====================================
' subroutine writes to an output file computation parameters
' and results
'==================================== OUTPUTER =====================================
SUB OUTPUTER

OPEN outputfile FOR APPEND AS #11

WRITE #11, equationname
WRITE #11, nKtype
WRITE #11, calctype
WRITE #11, "input file:", inputfile
WRITE #11, "output file:", outputfile
WRITE #11, "blocks:", block
WRITE #11, "i counter:", lasti
WRITE #11, "reversal count:", NMfcount
WRITE #11, "life factor:", usedlife

CLOSE #11

END SUB

V. SUBROUTINE PetersonKf

DEFSNG I-P
'==================================== PetersonKf =====================================
' subroutine to calculate the fatigue stress concentration factor
' based on Peterson's method
'==================================== PetersonKf =====================================
SUB PetersonKf

CALL TYPIN("Enter notch sensitivity factor (q; 0 if unknown)", q)
IF q <> 0 THEN
    Ktf = 1 + q * (Kt - 1)
ELSE
    CALL TYPIN("Enter the notch root radius (r)", r)
    CALL TYPIN("Enter the appropriate material constant (a)", a)
    Ktf = 1 + (Kt - 1) / (1 + a / r)
ENDIF

END SUB
W. FUNCTION Rotated

DEFINT I-P

' =========================== Rotated ===========================
' Returns the present value adjusted by Inc and rotated if necessary
' so that it falls within the range of Lower and Upper.
' =========================== Rotated ===========================
FUNCTION Rotated (Lower, Upper AS INTEGER, present, Inc)

  ' Calculate the next value
  present = present + Inc

  ' Handle special cases of rotating off top or bottom
  IF present > Upper THEN present = Lower
  IF present < Lower THEN present = Upper

  Rotated = present
END FUNCTION

X. SUBROUTINE TYPIN

' =========================== TYPIN ===========================
' subroutine to update the value of a specific variable through
' keyboard entry (real number variable)
' =========================== TYPIN ===========================
SUB TYPIN (cstring$, valu)

  LOCATE 23, 10: PRINT cstring$
  LOCATE 23, 58: INPUT valu
  LOCATE 23, 10: PRINT

END SUB

Y. SUBROUTINE TYPINSTRING

' =========================== TYPINSTRING ===========================
' subroutine to update the value of a specific string variable through
' keyboard entry
' =========================== TYPINSTRING ===========================
SUB TYPINSTRING (cstring$, stringname$)

  LOCATE 23, 10: PRINT cstring$
  LOCATE 23, 58: INPUT stringname$
  LOCATE 23, 10: PRINT

END SUB
Z. SUBROUTINE UPDATEMENU

'======================================== UPDATEMENU =========================================
' subroutine to update the options menu (menu 1)
'======================================== UPDATEMENU =========================================
SUB UPDATEMENU

SHARED VC AS Config, Menu AS Options1, Available AS STRING, Fields AS INTEGER, Fld AS INTEGER

' Constants for key codes and column positions
CONST COL1 = 10, COL2 = 50, ROW = 9
CONST COL3 = 7, COL4 = 42

' Display key instructions
LOCATE 1, COL1
PRINT "UP ............... Move to next field"
LOCATE 2, COL1
PRINT "DOWN .......... Move to previous field"
LOCATE 3, COL1
PRINT "LEFT/RIGHT ..... Change field up/down"
LOCATE 4, COL1
PRINT "F1 ...... Display material's parameters"
LOCATE 5, COL1
PRINT "ENTER .... Start with current values"
LOCATE 6, COL1
PRINT "ESCAPE ............... Quit Program"

' Display fields
LOCATE ROW, COL3: PRINT "Type of units (SI or British)";
LOCATE ROW, COL4: PRINT USING "[ & ]"; Menu.units;

LOCATE ROW + 2, COL3: PRINT "Material";
LOCATE ROW + 2, COL4: PRINT USING "[ & ]"; matname;

LOCATE ROW + 4, COL3: PRINT "Stress concentration factor (Kt)";
LOCATE ROW + 4, COL4: PRINT USING "[ ##.### ]"; Kt;

LOCATE ROW + 6, COL3: PRINT "Method to calculate Kf"
LOCATE ROW + 6, COL4: PRINT USING "[ & ]"; option4

LOCATE ROW + 8, COL3: PRINT "Fatigue stress conc. factor (Kf)"
LOCATE ROW + 8, COL4: PRINT USING "[ ##.### ]"; Ktf;

LOCATE ROW + 10, COL3: PRINT "Screen Mode"
LOCATE ROW + 10, COL4: PRINT USING "[ & ]"; VC.Scrn

END SUB
AA. SUBROUTINE UPDATEMENU2

'================================================================== UPDATEMENU2 ' subroutine to update the options menu (menu 2)
'==================================================================

SUB UPDATEMENU2

SHARED VC AS Config, menu2 AS Options2, Available AS STRING, Fields AS INTEGER, Fld AS INTEGER

' Constants for key codes and column positions
CONST COL1 = 10, COL2 = 50, ROW = 9
CONST COL3 = 7, COL4 = 42

CLS

' Display key instructions
LOCATE 1, COL1
PRINT "UP ............. Move to next field"
LOCATE 2, COL1
PRINT "DOWN ........ Move to previous field"
LOCATE 3, COL1
PRINT "LEFT/RIGHT .... Change field up/down"
LOCATE 4, COL1
PRINT USING "F2 ................ Turn sound to & "; flag5
LOCATE 5, COL1
PRINT "ENTER .... Start with current values"
LOCATE 6, COL1
PRINT "ESCAPE ... Return to previous screen"

' Display fields
LOCATE ROW, COL3: PRINT "Type of inputs (stress or strain)";
LOCATE ROW, COL4: PRINT USING "[ & ]"; menu2.inputs;

LOCATE ROW + 2, COL3: PRINT "Equation";
LOCATE ROW + 2, COL4: PRINT USING "[ & ]"; equationname;

LOCATE ROW + 4, COL3: PRINT "Fixed / Varing n' and K'";
LOCATE ROW + 4, COL4: PRINT USING "[ & ]"; nKtype;

LOCATE ROW + 6, COL3: PRINT "Calculation Type ";
LOCATE ROW + 6, COL4: PRINT USING "[ & ]"; calctype;

LOCATE ROW + 8, COL3: PRINT "Input file name ";
LOCATE ROW + 8, COL4: PRINT USING "[ & ]"; inputfile;

LOCATE ROW + 10, COL3: PRINT "Output file name ";
LOCATE ROW + 10, COL4: PRINT USING "[ & ]"; outputfile;

END SUB

67
FUNCTION xxKf

' Returns the value calculated for K' based on the number of cycles executed

FUNCTION xxKf
SHARE menu2 AS Options2

SELECT CASE menu2.option3
CASE 1 'fixed K'
   xxKf = K
CASE 2 'variable K'
   IF NNfcount > 1000000 THEN
      xxKf = Kf
   ELSEIF NNfcount < 2000 THEN
      xxKf = K
   ELSE
      xxKf = ((Kf - K) / (LOG10(500000) - LOG10(1000))) * (LOG10(1000000) / 2) - LOG10(1000)) + K
   END IF
CASE 3 'fixed K'
   xxKf = Kf
CASE 4 'variable K'
   IF NNfcount > 1000000 THEN
      xxKf = Kf
   ELSEIF NNfcount < 2000 THEN
      xxKf = K
   ELSE
      xxKf = ((Kf - K) / (LOG10(500000) - LOG10(1000))) * (LOG10(1000000) / 2) - LOG10(1000)) + K
   END IF
CASE ELSE
END SELECT
END FUNCTION

FUNCTION xxnf

' Returns the value calculated for n' based on the number of cycles executed

FUNCTION xxnf
SHARE menu2 AS Options2

SELECT CASE menu2.option3
CASE 1 'fixed n'
   xxnf = nf
CASE 2 'variable n'
   IF NNfcount > 1000000 THEN
      xxnf = nf
   ELSEIF NNfcount < 2000 THEN
      xxnf = K
   ELSE
      xxnf = ((Kf - K) / (LOG10(500000) - LOG10(1000))) * (LOG10(1000000) / 2) - LOG10(1000)) + K
   END IF
CASE ELSE
END SELECT
END FUNCTION
ELSEIF NNfcount < 2000 THEN
  xxnf = n
ELSE
  xxnf = ((nf - n) / (LOG10(500000) - LOG10(1000))) *
       (LOG10(NNfcount / 2) - LOG10(1000)) + n
END IF

CASE 3
  variable n'
  IF NNfcount > 1000000 THEN
    xxnf = nf
  ELSEIF NNfcount < 2000 THEN
    xxnf = n
  ELSE
    xxnf = ((nf - n) / (LOG10(500000) - LOG10(1000))) *
            (LOG10(NNfcount / 2) - LOG10(1000)) + n
  END IF

CASE 4
  fixed n'
  xxnf = nf
CASE ELSE
END SELECT

END FUNCTION

AD. FUNCTION xxNNfcount

'============================== xxNNfcount ===============================
' Returns the number of cycles executed
'============================== xxNNfcount ===============================
FUNCTION xxNNfcount&
  'DIM xxNNfcount AS LONG
xxNNfcount = (top * block + i)
END FUNCTION
APPENDIX B. PROGRAM LOADGEN

'============================================================================'
'  Random load generation program
'  based on a typical 1000 hour block for an A-6 aircraft
'============================================================================'

DIM outfile AS STRING
CONST fourg = 1978
CONST fiveg = 333
CONST sixg = 48
CONST seveng = 10
CONST totalg = fourg + fiveg + sixg + seveng
CONST Su = 84
CONST Sy = 76
CONST gdesign = 6.5

RANDOMIZE TIMER
CLS
'LOCATE 20, 10: PRINT "Enter name for stress load output file:"'
'LOCATE 20, 50: INPUT outfile
CLS
FOR j = 1 TO 4
  ' loop to create four random files
  fourcount = fourg
  fivecount = fiveg
  sixcount = sixg
  sevencount = seveng
  totalcount = fourg + fiveg + sixg + seveng
  SELECT CASE j
    CASE 1
      OPEN "testaa" FOR OUTPUT AS #2
    CASE 2
      OPEN "testbb" FOR OUTPUT AS #2
    CASE 3
      OPEN "testcc" FOR OUTPUT AS #2
    CASE 4
      OPEN "testdd" FOR OUTPUT AS #2
    CASE ELSE
      END SELECT
    END SELECT
"g" load history greneration:

OPEN "gseries" FOR OUTPUT AS #1

WHILE totalcount > 0
  DO
    x = RND
    xx = RND
    IF x <= (seveng / totalg) THEN
      7+ "g" case
      IF sevencount = 0 THEN EXIT DO
      y = 7 + (INT(xx * 10)) / 10
      WRITE #1, y
      WRITE #1, 1
      sevencount = sevencount - 1
    ELSEIF x <= ((seveng + sixg) / totalg) THEN
      6 to 7 "g" case
      IF sixcount = 0 THEN EXIT DO
      y = 6 + (INT(xx * 10)) / 10
      WRITE #1, y
      WRITE #1, 1
      sixcount = sixcount - 1
    ELSEIF x <= ((seveng + sixg + fiveg) / totalg) THEN
      5 to 6 "g" case
      IF fivecount = 0 THEN EXIT DO
      y = 5 + (INT(xx * 10)) / 10
      WRITE #1, y
      WRITE #1, 1
      fivecount = fivecount - 1
    ELSE
      4 to 5 "g" case
      IF fourcount = 0 THEN EXIT DO
      y = 4 + (INT(xx * 10)) / 10
      WRITE #1, y
      WRITE #1, 1
      fourcount = fourcount - 1
    END IF
  END IF
  totalcount = fourcount + fivecount + sixcount + sevencount
END DO
LOOP
WEND
WRITE #1, 999
CLOSE #1
' conversion of "g" load history to a stress load history:

IF Su > (1.5 * Sy) THEN
  gotstress = Su / (1.5 * gdesign)
ELSE
  gotstress = Sy / (gdesign)
END IF

OPEN "gseries" FOR INPUT AS #3

INPUT #3, load

WHILE load < 999
  stressload = load * gotstress
  WRITE #2, stressload
  INPUT #3, load
WEND

CLOSE #2
CLOSE #3

load = 0

NEXT j

END
APPENDIX C. MATLAB DATA REDUCTION PROGRAM

A. PROGRAM CODE

\% UNCORRECTED STRAIN + LOAD DATA (for K and n)
format compact

\% Monotonic #1
\% (81 points)
\%========================================
\% 0 1 2 3 4 5 6 7 8 9
\%----------------------------------------
ex0a=[.0004; .0012; .0022; .0030; .0038; .0047; .0055; .0063; .0071; .0081;
     .0091; .0099; .0110; .0119; .0127; .0136; .0143; .0152; .0159; .0168;
     .0176; .0184; .0192; .0200; .0208; .0216; .0224; .0232; .0241; .0249;
     .0257; .0265; .0273; .0281; .0289; .0297; .0305; .0313; .0322; .0330;
     .0356; .0381; .0401; .0422; .0443; .0464; .0487; .0510; .0530; .0551;
     .0573; .0595; .0617; .0640; .0663; .0687; .0707; .0721; .0756; .0835;
     .0859; .0880; .0901; .0923; .0944; .0966; .1170; .1373; .1574; .1777;
     .2005; .2376; .2727; .3034; .3316; .3604; .4274; .4643; .5019; .5399;
     .5786];

ld0a=[ 305; 628; 998; 1332; 1650; 1979; 2307; 2627; 2975; 3350;
      3724; 4118; 4488; 4829; 5145; 5463; 5771; 6088; 6406; 6731;
      7044; 7345; 7663; 7973; 8278; 8574; 8896; 9214; 9526; 9841;
      10140; 10448; 10726; 11036; 11341; 11635; 11943; 12223; 12513; 12789;
      13738; 14374; 14874; 15312; 15695; 16003; 16278; 16515; 16867; 16839;
      16971; 17098; 17219; 17307; 17390; 17475; 17534; 17616; 17675; 17888;
      17884; 17921; 17967; 17990; 18027; 18060; 18266; 18419; 18532; 18627;
      18901; 18843; 18916; 18932; 19065; 19128; 19176; 19225; 19264; 19302;
      19326; 19339];

\% 'Monotonic #2
\% (88 points)
\%========================================
\% 0 1 2 3 4 5 6 7 8 9
\%----------------------------------------
ex0b=[.0005; .0012; .0021; .0029; .0037; .0045; .0053; .0060; .0068; .0076;
     .0085; .0093; .0103; .0112; .0120; .0128; .0135; .0143; .0151; .0159;
     .0166; .0174; .0182; .0190; .0198; .0205; .0213; .0220; .0228; .0236;
     .0244; .0251; .0259; .0267; .0274; .0282; .0290; .0298; .0305; .0313;
     .0340; .0362; .0382; .0404; .0425; .0446; .0468; .0489; .0510; .0531;
     .0553; .0574; .0596; .0618; .0641; .0664; .0684; .0704; .0725; .0791;
     .0812; .0834; .0855; .0877; .0900; .0923; .1125; .1327; .1530; .1734;
     .1934; .2134; .2336; .2538; .2742; .2943; .3145; .3350; .3550; .3755;
     .3958; .4159; .4359; .4561; .4762; .4965; .5166; .5368; .5572; .5786;]

ld0b=[ 306; 619; 962; 1303; 1619; 1958; 2275; 2590; 2911; 3242;
      3596; 3972; 4375; 4697; 5001; 5333; 5631; 5943; 6259; 6569;
      6890; 7195; 7511; 7815; 8131; 8417; 8723; 9041; 9339; 9651;
      9934; 10240; 10540; 10852; 11134; 11442; 11749; 12031; 12300; 12574;
      13536; 14209; 14795; 15374; 15854; 16297; 16676; 16931; 17147; 17273;
      17399; 17496; 17586; 17667; 17744; 17804; 17854; 17901; 17947; 18073;
      18095; 18131; 18163; 18197; 18221; 18248; 18432; 18571; 18661; 18768;
      18851; 18927; 19002; 19052; 19105; 19153; 19200; 19241; 19277; 19305;
      19224; 19258; 19381; 19404; 19417; 19433; 19438; 19449; 19452];

73
<table>
<thead>
<tr>
<th>% 10 percent: 3790 cycles</th>
</tr>
</thead>
<tbody>
<tr>
<td>(73 points)</td>
</tr>
<tr>
<td>%.................................................................</td>
</tr>
<tr>
<td>0</td>
</tr>
</tbody>
</table>
| ~x10=[0.0005; 0.0013; 0.0022; 0.0030; 0.0038; 0.0047; 0.0054; 0.0063; 0.0071; 0.0079]
| .0088; 0.0099; 0.0108; 0.0117; 0.0126; 0.0134; 0.0142; 0.0151; 0.0159; 0.0167|
| .0175; 0.0184; 0.0192; 0.0200; 0.0209; 0.0217; 0.0225; 0.0233; 0.0242; 0.0250|
| .0258; 0.0266; 0.0274; 0.0283; 0.0291; 0.0299; 0.0307; 0.0316; 0.0324; 0.0332|
| .0361; 0.0381; 0.0402; 0.0422; 0.0444; 0.0466; 0.0489; 0.0511; 0.0532; 0.0553|
| .0574; 0.0595; 0.0618; 0.0638; 0.0660; 0.0682; 0.0703; 0.0725; 0.0747; 0.0778|
| .0798; 0.0818; 0.0840; 0.0861; 0.0882; 0.0902; 0.1102; 0.1304; 0.1506; 0.1707|
| .1911; .2114; .2314; .2516; |
| 1d10=[302; 595; 934; 1252; 1561; 1864; 2160; 2469; 2770; 3098]
| 3466; 3795; 4170; 4486; 4785; 5076; 5369; 5654; 5967; 6274|
| 6571; 6856; 7161; 7450; 7748; 8026; 8334; 8659; 8911; 9211|
| 9490; 9787; 10071; 10361; 10652; 10946; 11225; 11517; 11812; 12074|
| 13059; 13751; 14478; 15148; 15875; 16591; 17315; 18004; 18577; 19073|
| 19372; 19536; 19560; 19573; 19552; 19524; 19483; 19365; 19243; 19202; 19137; 19015; 18802; |

<table>
<thead>
<tr>
<th>% 20 percent: 7580 cycles</th>
</tr>
</thead>
<tbody>
<tr>
<td>(87 points)</td>
</tr>
<tr>
<td>%.................................................................</td>
</tr>
<tr>
<td>0</td>
</tr>
<tr>
<td>~x20=[0.0007; 0.0016; 0.0028; 0.0039; 0.0049; 0.0060; 0.0069; 0.0079; 0.0089; 0.0100; 0.0122</td>
</tr>
<tr>
<td>.0133; 0.0143; 0.0153; 0.0163; 0.0173; 0.0183; 0.0193; 0.0203; 0.0213; 0.0223; 0.0233</td>
</tr>
<tr>
<td>.0243; 0.0253; 0.0264; 0.0274; 0.0284; 0.0294; 0.0304; 0.0314; 0.0324; 0.0334; 0.0344</td>
</tr>
<tr>
<td>.0354; 0.0365; 0.0375; 0.0386; 0.0396; 0.0406; 0.0416; 0.0426; 0.0436; 0.0446; 0.0456</td>
</tr>
<tr>
<td>.0515; 0.0536; 0.0557; 0.0578; 0.0599; 0.0620; 0.0641; 0.0662; 0.0683; 0.0704; 0.0725</td>
</tr>
<tr>
<td>.0745; 0.0766; 0.0788; 0.0810; 0.0832; 0.0854; 0.0876; 0.0898; 0.0920; 0.0942; 0.0964</td>
</tr>
<tr>
<td>.1031; 0.1052; 0.1073; 0.1094; 0.1115; 0.1136; 0.1157; 0.1179; 0.1200; 0.1221; 0.1242</td>
</tr>
<tr>
<td>.1263; 0.1284; 0.1305; 0.1326; 0.1347; 0.1368; 0.1389; 0.1410; 0.1431; 0.1452; 0.1473</td>
</tr>
<tr>
<td>.1494; 0.1515; 0.1536; 0.1557; 0.1578; 0.1599; 0.1620; 0.1641; 0.1662; 0.1683; 0.1704</td>
</tr>
<tr>
<td>.1725; 0.1746; 0.1767; 0.1788; 0.1809; 0.1830; 0.1851; 0.1872; 0.1893; 0.1914; 0.1935</td>
</tr>
<tr>
<td>.1956; 0.1977; 0.1998; 0.2019; 0.2040; 0.2061; 0.2081; 0.2102; 0.2123; 0.2144; 0.2164</td>
</tr>
<tr>
<td>.2185; 0.2206; 0.2226; 0.2247; 0.2267; 0.2287; 0.2307; 0.2328; 0.2348; 0.2368; 0.2388</td>
</tr>
</tbody>
</table>
| 1d20=[327; 650; 984; 1324; 1641; 1955; 2271; 2609; 2952; 3291]
| 3592; 3912; 4246; 4540; 4896; 5206; 5513; 5827; 6132; 6468|
| 6782; 7077; 7411; 7699; 8034; 8301; 8629; 8918; 9245; 9516|
| 9816; 10093; 10364; 10670; 10939; 11241; 11518; 11793; 12051; 12331|
| 13238; 13543; 13876; 14198; 14513; 14846; 15122; 15467; 15741; 16030|
| 16302; 16577; 16838; 17029; 17194; 17366; 17483; 17593; 17656; 17921|
| 17944; 18007; 18062; 18086; 18120; 18157; 18402; 18568; 18703; 18802|
| 18888; 18951; 19025; 19092; 19140; 19209; 19253; 19297; 19343; 19387|
| 19414; 19456; 19463; 19501; 19528; 19576; 19567; 19571; 19577; 19567; 19571];
% 30 percent: 11370 cycles  
% (88 points) 

<table>
<thead>
<tr>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
</tr>
</thead>
<tbody>
<tr>
<td>.0004</td>
<td>.0012</td>
<td>.0021</td>
<td>.0030</td>
<td>.0038</td>
<td>.0046</td>
<td>.0055</td>
<td>.0063</td>
<td>.0071</td>
<td>.0079</td>
</tr>
<tr>
<td>.0089</td>
<td>.0098</td>
<td>.0107</td>
<td>.0116</td>
<td>.0125</td>
<td>.0133</td>
<td>.0141</td>
<td>.0150</td>
<td>.0158</td>
<td>.0166</td>
</tr>
<tr>
<td>.0174</td>
<td>.0182</td>
<td>.0191</td>
<td>.0199</td>
<td>.0207</td>
<td>.0215</td>
<td>.0223</td>
<td>.0230</td>
<td>.0239</td>
<td>.0246</td>
</tr>
<tr>
<td>.0255</td>
<td>.0263</td>
<td>.0271</td>
<td>.0279</td>
<td>.0287</td>
<td>.0295</td>
<td>.0303</td>
<td>.0311</td>
<td>.0319</td>
<td>.0327</td>
</tr>
<tr>
<td>.0355</td>
<td>.0376</td>
<td>.0398</td>
<td>.0419</td>
<td>.0441</td>
<td>.0463</td>
<td>.0483</td>
<td>.0505</td>
<td>.0527</td>
<td>.0547</td>
</tr>
<tr>
<td>.0568</td>
<td>.0589</td>
<td>.0609</td>
<td>.0630</td>
<td>.0651</td>
<td>.0693</td>
<td>.0716</td>
<td>.0736</td>
<td>.0819</td>
<td>.0843</td>
</tr>
<tr>
<td>.0843</td>
<td>.0867</td>
<td>.0892</td>
<td>.0915</td>
<td>.0938</td>
<td>.0963</td>
<td>.1169</td>
<td>.1372</td>
<td>.1576</td>
<td>.1780</td>
</tr>
<tr>
<td>.1989</td>
<td>.2196</td>
<td>.2397</td>
<td>.2602</td>
<td>.2803</td>
<td>.3009</td>
<td>.3209</td>
<td>.3412</td>
<td>.3613</td>
<td>.3813</td>
</tr>
<tr>
<td>.4016</td>
<td>.4216</td>
<td>.4418</td>
<td>.4625</td>
<td>.4829</td>
<td>.5031</td>
<td>.5235</td>
<td>.5438</td>
<td>.5645</td>
<td></td>
</tr>
</tbody>
</table>

ld30=[ 293; 595; 939; 1266; 1583; 1912; 2225; 2539; 2871; 3196; 3546; 3901; 4302; 4632; 4917; 5237; 5562; 5871; 6176; 6486; 6812; 7112; 7423; 7719; 8129; 8599; 9050; 9520; 9809; 10117; 10402; 10719; 11009; 11293; 11599; 11905; 12193; 12473; 13473; 14261; 15044; 15811; 16554; 17289; 17942; 18590; 18954; 19085; 19101; 19093; 19114; 19104; 19111; 19088; 19098; 19053; 19081; 19132; 19181; 19239; 19255; 19297; 19332; 19365; 19389; 19438; 19486; 19489; 19498; 19501; 19493; 19498; 19487; 19503; 19510; 19498];

% 40 percent: 15160 cycles  
% (86 points) 

<table>
<thead>
<tr>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
</tr>
</thead>
<tbody>
<tr>
<td>.0002</td>
<td>.0015</td>
<td>.0031</td>
<td>.0045</td>
<td>.0057</td>
<td>.0070</td>
<td>.0082</td>
<td>.0095</td>
<td>.0107</td>
<td>.0119</td>
</tr>
<tr>
<td>.0129</td>
<td>.0144</td>
<td>.0158</td>
<td>.0173</td>
<td>.0187</td>
<td>.0205</td>
<td>.0220</td>
<td>.0236</td>
<td>.0253</td>
<td>.0269</td>
</tr>
<tr>
<td>.0286</td>
<td>.0304</td>
<td>.0320</td>
<td>.0337</td>
<td>.0354</td>
<td>.0372</td>
<td>.0389</td>
<td>.0408</td>
<td>.0425</td>
<td>.0442</td>
</tr>
<tr>
<td>.0460</td>
<td>.0479</td>
<td>.0497</td>
<td>.0515</td>
<td>.0534</td>
<td>.0554</td>
<td>.0575</td>
<td>.0593</td>
<td>.0612</td>
<td>.0631</td>
</tr>
<tr>
<td>.0697</td>
<td>.0717</td>
<td>.0738</td>
<td>.0759</td>
<td>.0780</td>
<td>.0800</td>
<td>.0820</td>
<td>.0845</td>
<td>.0866</td>
<td>.0891</td>
</tr>
<tr>
<td>.0911</td>
<td>.0934</td>
<td>.0958</td>
<td>.0983</td>
<td>.1005</td>
<td>.1028</td>
<td>.1051</td>
<td>.1075</td>
<td>.1097</td>
<td>.1119</td>
</tr>
<tr>
<td>.1214</td>
<td>.1237</td>
<td>.1261</td>
<td>.1283</td>
<td>.1306</td>
<td>.1328</td>
<td>.1352</td>
<td>.1377</td>
<td>.1393</td>
<td>.1419</td>
</tr>
<tr>
<td>.2343</td>
<td>.2545</td>
<td>.2746</td>
<td>.2950</td>
<td>.3151</td>
<td>.3355</td>
<td>.3555</td>
<td>.3758</td>
<td>.3962</td>
<td>.4165</td>
</tr>
<tr>
<td>.4365</td>
<td>.4571</td>
<td>.4772</td>
<td>.4973</td>
<td>.5175</td>
<td>.5375</td>
<td>.5581</td>
<td>.5785</td>
<td>.5990</td>
<td>.6195</td>
</tr>
</tbody>
</table>

ld40=[ 158; 450; 835; 948; 1230; 1682; 2094; 2415; 2720; 2949; 3194; 3482; 3794; 4049; 4339; 4635; 4942; 5220; 5500; 5770; 6077; 6356; 6625; 6915; 7192; 7509; 7741; 8026; 8301; 8564; 8836; 9103; 9381; 9544; 9793; 9507; 9725; 10160; 10471; 10657; 9823; 12246; 10089; 10914; 12698; 11817; 10769; 13885; 13498; 14350; 12099; 13279; 14430; 14704; 15653; 14991; 16104; 16423; 16972; 17423; 16899; 17247; 17428; 17435; 17558; 18103; 18442; 18617; 18789; 18920; 19008; 19074; 19148; 19206; 19185; 19232; 19362; 19395; 19464; 19503; 19539; 19445; 19588; 19634; 19664; 19667];

75
% adjust number:

area = .046875;

1d0a = 1d0a / 5;  stress0a = 1d0a / area;
1d0b = 1d0b / 5;  stress0b = 1d0b / area;
1d10 = 1d10 / 5;  stress10 = 1d10 / area;
1d20 = 1d20 / 5;  stress20 = 1d20 / area;
1d30 = 1d30 / 5;  stress30 = 1d30 / area;
1d40 = 1d40 / 5;  stress40 = 1d40 / area;

strain0a = ex0a * .3;
strain0b = ex0b * .3;
strain10 = ex10 * .3;
strain20 = ex20 * .3;
strain30 = ex30 * .3;
strain40 = ex40 * .3;

%!del a:\ernie.met
%!del ermie.met

plot(strain0a, stress0a), title('Monotonic #1 Stress-Strain'), grid
xlabel('Strain (in/in)'), ylabel('Stress (psi)');
%pause
%meta a:\ernie

plot(strain0b, stress0b), title('Monotonic #2 Stress-Strain'), grid
xlabel('Strain (in/in)'), ylabel('Stress (psi)');
%pause
%meta a:\ernie

plot(strain10, stress10), title('10% Life Stress-Strain'), grid
xlabel('Strain (in/in)'), ylabel('Stress (psi)');
%pause
%meta a:\ernie

plot(strain20, stress20), title('20% Life Stress-Strain'), grid
xlabel('Strain (in/in)'), ylabel('Stress (psi)');
%pause
%meta a:\ernie

plot(strain30, stress30), title('30% Life Stress-Strain'), grid
xlabel('Strain (in/in)'), ylabel('Stress (psi)');
%pause
%meta a:\ernie

plot(strain40, stress40), title('40% Life Stress-Strain'), grid
xlabel('Strain (in/in)'), ylabel('Stress (psi)');
%pause
%meta a:\ernie

plot(strain0b, stress0b, strain10, stress10, strain20, stress20, ...
strain30, stress30, strain40, stress40), title('Stress-Strain'), grid
xlabel('Strain (in/in)'), ylabel('Stress (psi)');
%pause
%meta a:\ernie
% compute true strain from engineering

tstrain0a = log(1 + strain0a);
tstrain0b = log(1 + strain0b);
tstrain10 = log(1 + strain10);
tstrain20 = log(1 + strain20);
tstrain30 = log(1 + strain30);
tstrain40 = log(1 + strain40);

% compute true stress from engineering

sig1 = stress0a .* (1 + tstrain0a);
sig2 = stress0b .* (1 + tstrain0b);
sig3 = stress10 .* (1 + tstrain10);
sig4 = stress20 .* (1 + tstrain20);
sig5 = stress30 .* (1 + tstrain30);
sig6 = stress40 .* (1 + tstrain40);

lsig1 = log10(sig1);
sig2 = log10(sig2);
sig3 = log10(sig3);
sig4 = log10(sig4);
sig5 = log10(sig5);
sig6 = log10(sig6);

E = 10300000;

x1 = log10(tstrain0a(2:82) - sig1(2:82)/E);
x2 = log10(tstrain0b - sig2/E);
x3 = log10(tstrain10 - sig3/E);
x4 = log10(tstrain20 - sig4/E);
x5 = log10(tstrain30(2:89) - sig5(2:89)/E);
x6 = log10(tstrain40(2:87) - sig6(2:87)/E);

lsig1a = lsig1(2:82);
lsig5a = lsig5(2:89);
lsig6a = lsig6(2:87);

xx1 = polyfit(x1, lsig1a, 1)
xx2 = polyfit(x2, lsig2, 1)
xx3 = polyfit(x3, lsig3, 1)
xx4 = polyfit(x4, lsig4, 1)
xx5 = polyfit(x5, lsig5a, 1)
xx6 = polyfit(x6, lsig6a, 1)
\[
\begin{align*}
K1 &= 10^\{xx1(2)\} \\
K2 &= 10^\{xx2(2)\} \\
K3 &= 10^\{xx3(2)\} \\
K4 &= 10^\{xx4(2)\} \\
K5 &= 10^\{xx5(2)\} \\
K6 &= 10^\{xx6(2)\} \\
n1 &= xx1(1) \\
n2 &= xx2(1) \\
n3 &= xx3(1) \\
n4 &= xx4(1) \\
n5 &= xx5(1) \\
n6 &= xx6(1)
\end{align*}
\]

\footnotesize

\begin{tabular}{lcc}
\textbf{xx1} & 0.4425 & 5.6106 \\
\textbf{xx2} & 0.4331 & 5.5810 \\
\textbf{xx3} & 0.6246 & 6.0503 \\
\textbf{xx4} & 0.5063 & 5.6425 \\
\textbf{xx5} & 0.4261 & 5.5662 \\
\textbf{xx6} & 0.5902 & 5.6822 \\
\end{tabular}

\[
\begin{align*}
K1 &= 4.0798e+005 \\
K2 &= 3.8107e+005 \\
K3 &= 1.1229e+006 \\
K4 &= 4.3900e+005 \\
K5 &= 3.6831e+005 \\
K6 &= 4.8105e+005 \\
n1 &= 0.4425 \\
n2 &= 0.4331 \\
n3 &= 0.6246 \\
n4 &= 0.5063 \\
n5 &= 0.4261 \\
n6 &= 0.5902
\end{align*}
\]

B. PROGRAM OUTPUT/RESULTS
### APPENDIX D. MATERIAL DATA BASE

#### A. BRITISH/AMERICAN UNITS

**Monotonic and Cyclic Strain Properties of Selected Engineering Alloys: American/British units**

<table>
<thead>
<tr>
<th>Material Description</th>
<th>Process Description</th>
<th>$S_y$ (ksi)</th>
<th>$S_y/S_y'$ (ksi/ksi)</th>
<th>$K/K'$ (ksi/ksi)</th>
<th>$n_{0.2}$</th>
<th>$e_{0.2}$</th>
<th>$e_{0.2}'$</th>
<th>$e_{0.2}''$</th>
<th>$b$</th>
<th>$c$</th>
<th>$S_r$ (ksi)</th>
<th>$S_{0.2}$ (ksi)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Steel</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1005-1009 H.R. sheet</td>
<td></td>
<td>50</td>
<td>38.33</td>
<td>72.67</td>
<td>0.16</td>
<td>0.12</td>
<td>1.60</td>
<td>12.00</td>
<td>123.03</td>
<td>-0.109</td>
<td>-0.39</td>
<td>21</td>
</tr>
<tr>
<td>1005-1009 C.D. sheet</td>
<td></td>
<td>60</td>
<td>58.36</td>
<td>76.71</td>
<td>0.049</td>
<td>0.11</td>
<td>1.02</td>
<td>11.11</td>
<td>122.28</td>
<td>-0.073</td>
<td>-0.41</td>
<td>28</td>
</tr>
<tr>
<td>1020 H.R. sheet</td>
<td></td>
<td>64</td>
<td>38.35</td>
<td>107.112</td>
<td>0.19</td>
<td>0.18</td>
<td>0.96</td>
<td>1.0410</td>
<td>103.13</td>
<td>-0.12</td>
<td>-0.51</td>
<td>22</td>
</tr>
<tr>
<td>0030* Cast steel</td>
<td></td>
<td>72</td>
<td>44.46</td>
<td></td>
<td>0.30</td>
<td>0.13</td>
<td>0.62</td>
<td>0.28</td>
<td>109.95</td>
<td>-0.082</td>
<td>-0.51</td>
<td>28</td>
</tr>
<tr>
<td>Man-Ten H.R. sheet</td>
<td></td>
<td>74</td>
<td>57.64</td>
<td>-114</td>
<td>0.20</td>
<td>0.11</td>
<td>1.02</td>
<td>0.86</td>
<td>118.17</td>
<td>-0.071</td>
<td>-0.65</td>
<td>38</td>
</tr>
<tr>
<td>1040 As forged</td>
<td></td>
<td>90</td>
<td>50.56</td>
<td></td>
<td>0.22</td>
<td>0.18</td>
<td>0.93</td>
<td>0.61</td>
<td>152.22</td>
<td>-0.14</td>
<td>-0.57</td>
<td>25</td>
</tr>
<tr>
<td>RQC 100 H.R. sheet</td>
<td></td>
<td>135</td>
<td>128.87</td>
<td>170.208</td>
<td>0.066</td>
<td>0.14</td>
<td>1.02</td>
<td>0.66</td>
<td>193.28</td>
<td>-0.07</td>
<td>-0.69</td>
<td>59</td>
</tr>
<tr>
<td>4142 Drawn at temp</td>
<td></td>
<td>154</td>
<td>152.108</td>
<td></td>
<td>-0.18</td>
<td>0.35</td>
<td>0.22</td>
<td>0.10</td>
<td>162.21</td>
<td>-0.10</td>
<td>-0.51</td>
<td>44</td>
</tr>
<tr>
<td>4142 Q &amp; T</td>
<td></td>
<td>205</td>
<td>200.120</td>
<td></td>
<td>0.051</td>
<td>0.17</td>
<td>0.66</td>
<td>0.45</td>
<td>265.26</td>
<td>-0.08</td>
<td>-0.75</td>
<td>33</td>
</tr>
<tr>
<td>4142 Q &amp; T</td>
<td></td>
<td>280</td>
<td>250.195</td>
<td></td>
<td>0.048</td>
<td>0.13</td>
<td>0.43</td>
<td>0.09</td>
<td>315.35</td>
<td>-0.081</td>
<td>-0.61</td>
<td>85</td>
</tr>
<tr>
<td>4340 H.R. and annealed</td>
<td></td>
<td>120</td>
<td>92.66</td>
<td></td>
<td>-0.18</td>
<td>0.57</td>
<td>0.45</td>
<td>0.17</td>
<td>158.14</td>
<td>-0.095</td>
<td>-0.54</td>
<td>40</td>
</tr>
<tr>
<td>4340 Q &amp; T</td>
<td></td>
<td>180</td>
<td>170.110</td>
<td>299</td>
<td>0.066</td>
<td>0.14</td>
<td>0.84</td>
<td>0.73</td>
<td>240.24</td>
<td>-0.076</td>
<td>-0.62</td>
<td>71</td>
</tr>
<tr>
<td>4340 Q &amp; T</td>
<td></td>
<td>213</td>
<td>199.120</td>
<td></td>
<td>-0.15</td>
<td>0.48</td>
<td>0.48</td>
<td>0.48</td>
<td>226.29</td>
<td>-0.091</td>
<td>-0.60</td>
<td>68</td>
</tr>
<tr>
<td>9262 Annealed</td>
<td></td>
<td>134</td>
<td>66.76</td>
<td>253.200</td>
<td>0.220</td>
<td>0.15</td>
<td>0.16</td>
<td>0.16</td>
<td>151.45</td>
<td>-0.071</td>
<td>-0.47</td>
<td>59</td>
</tr>
<tr>
<td>9262 Q &amp; T</td>
<td></td>
<td>145</td>
<td>114.91</td>
<td>-197</td>
<td>0.140</td>
<td>0.12</td>
<td>0.41</td>
<td>0.41</td>
<td>177.17</td>
<td>-0.073</td>
<td>-0.60</td>
<td>55</td>
</tr>
<tr>
<td>Aluminum</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1100.0 As received</td>
<td></td>
<td>46</td>
<td>14.9</td>
<td></td>
<td>-0.15</td>
<td>2.09</td>
<td>1.8</td>
<td>0.28</td>
<td>-0.106</td>
<td>-0.69</td>
<td>5</td>
<td>33</td>
</tr>
<tr>
<td>2024-T1</td>
<td></td>
<td>68</td>
<td>55.62</td>
<td>66.95</td>
<td>0.032</td>
<td>0.065</td>
<td>0.28</td>
<td>0.22</td>
<td>81.180</td>
<td>-0.12</td>
<td>-0.59</td>
<td>22</td>
</tr>
<tr>
<td>2024-T4</td>
<td></td>
<td>69</td>
<td>43.64</td>
<td>117</td>
<td>0.200</td>
<td>0.08</td>
<td>0.63</td>
<td>0.21</td>
<td>92.147</td>
<td>-0.11</td>
<td>-0.52</td>
<td>25</td>
</tr>
<tr>
<td>5456-H3</td>
<td></td>
<td>58</td>
<td>13.52</td>
<td></td>
<td>-0.16</td>
<td>0.42</td>
<td>0.46</td>
<td>0.11</td>
<td>76.105</td>
<td>-0.11</td>
<td>-0.67</td>
<td>18</td>
</tr>
<tr>
<td>7075-T6</td>
<td></td>
<td>84</td>
<td>68.76</td>
<td>120</td>
<td>0.110</td>
<td>0.146</td>
<td>0.41</td>
<td>0.19</td>
<td>108.19</td>
<td>-0.12</td>
<td>-0.52</td>
<td>25</td>
</tr>
</tbody>
</table>

*Note: All values are given in ksi.*
### B. SI UNITS

#### Monotonic and Cyclic Strain Properties of Selected Engineering Alloys: SI Units

<table>
<thead>
<tr>
<th>Material</th>
<th>Process Description</th>
<th>$S_y$</th>
<th>$S_p$</th>
<th>$S_{py}$</th>
<th>$K$</th>
<th>$\sigma_{mp}$</th>
<th>$\epsilon_{mp}$</th>
<th>$\epsilon_{cr}$</th>
<th>$\sigma_{cr}$</th>
<th>$\epsilon_{max}$</th>
<th>$\sigma_{mean}$</th>
<th>$\epsilon_{mean}$</th>
<th>Fatigue Crack</th>
<th>$S_y$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Steel</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1005-1009</td>
<td>H.R. sheet</td>
<td>745</td>
<td>262</td>
<td>218</td>
<td>0.16</td>
<td>0.14</td>
<td>1.6</td>
<td>0.10</td>
<td>848/644</td>
<td>-0.109</td>
<td>-0.109</td>
<td>148</td>
<td>0.43</td>
<td></td>
</tr>
<tr>
<td>1005-1009</td>
<td>C.D. sheet</td>
<td>474</td>
<td>400</td>
<td>429</td>
<td>0.049</td>
<td>0.011</td>
<td>1.02</td>
<td>0.11</td>
<td>844/538</td>
<td>-0.073</td>
<td>-0.073</td>
<td>195</td>
<td>0.47</td>
<td></td>
</tr>
<tr>
<td>1020</td>
<td>H.R. sheet</td>
<td>441</td>
<td>262</td>
<td>241</td>
<td>0.19</td>
<td>0.018</td>
<td>0.96</td>
<td>0.041</td>
<td>790/896</td>
<td>-0.12</td>
<td>-0.12</td>
<td>152</td>
<td>0.34</td>
<td></td>
</tr>
<tr>
<td>0030</td>
<td>Cast steel</td>
<td>416</td>
<td>103</td>
<td>217</td>
<td>0.30</td>
<td>0.28</td>
<td>0.62</td>
<td>0.16</td>
<td>606/653</td>
<td>-0.082</td>
<td>-0.082</td>
<td>180</td>
<td>0.36</td>
<td></td>
</tr>
<tr>
<td>Man Ten</td>
<td>H.R. sheet</td>
<td>510</td>
<td>193</td>
<td>186</td>
<td>0.20</td>
<td>0.11</td>
<td>1.02</td>
<td>0.066</td>
<td>814/807</td>
<td>-0.071</td>
<td>-0.071</td>
<td>262</td>
<td>0.51</td>
<td></td>
</tr>
<tr>
<td>1040</td>
<td>As forged</td>
<td>621</td>
<td>145</td>
<td>136</td>
<td>0.22</td>
<td>0.18</td>
<td>0.93</td>
<td>0.064</td>
<td>1050/150</td>
<td>-0.14</td>
<td>-0.14</td>
<td>173</td>
<td>0.28</td>
<td></td>
</tr>
<tr>
<td>ROQ-100</td>
<td>H.R. sheet</td>
<td>981</td>
<td>833</td>
<td>600</td>
<td>0.06</td>
<td>0.14</td>
<td>1.02</td>
<td>0.66</td>
<td>1330/1240</td>
<td>-0.07</td>
<td>-0.07</td>
<td>403</td>
<td>0.43</td>
<td></td>
</tr>
<tr>
<td>4142</td>
<td>Drawn at temp</td>
<td>1062</td>
<td>1048</td>
<td>745</td>
<td>-</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4142</td>
<td>Q &amp; T</td>
<td>1413</td>
<td>1379</td>
<td>827</td>
<td>0.05</td>
<td>0.017</td>
<td>0.66</td>
<td>0.45</td>
<td>1825/1825</td>
<td>-0.08</td>
<td>-0.08</td>
<td>593</td>
<td>0.36</td>
<td></td>
</tr>
<tr>
<td>4142</td>
<td>Q &amp; T</td>
<td>1931</td>
<td>1724</td>
<td>1344</td>
<td>0.048</td>
<td>0.013</td>
<td>0.43</td>
<td>0.09</td>
<td>2170/2170</td>
<td>-0.08</td>
<td>-0.08</td>
<td>589</td>
<td>0.31</td>
<td></td>
</tr>
<tr>
<td>4340</td>
<td>H.R. and annealed</td>
<td>827</td>
<td>634</td>
<td>455</td>
<td>-</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4340</td>
<td>Q &amp; T</td>
<td>1241</td>
<td>1172</td>
<td>768</td>
<td>0.055</td>
<td>0.014</td>
<td>0.57</td>
<td>0.45</td>
<td>1090/1200</td>
<td>-0.095</td>
<td>-0.095</td>
<td>272</td>
<td>0.33</td>
<td></td>
</tr>
<tr>
<td>4340</td>
<td>Q &amp; T</td>
<td>1489</td>
<td>1372</td>
<td>927</td>
<td>-</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9262</td>
<td>Annealed</td>
<td>924</td>
<td>455</td>
<td>524</td>
<td>0.22</td>
<td>0.15</td>
<td>0.60</td>
<td>0.16</td>
<td>1084/1046</td>
<td>-0.07</td>
<td>-0.07</td>
<td>328</td>
<td>0.58</td>
<td></td>
</tr>
<tr>
<td>9262</td>
<td>Q &amp; T</td>
<td>1000</td>
<td>586</td>
<td>468</td>
<td>-1358</td>
<td>0.14</td>
<td>0.41</td>
<td>0.41</td>
<td>1220/1220</td>
<td>-0.073</td>
<td>-0.073</td>
<td>381</td>
<td>0.38</td>
<td></td>
</tr>
<tr>
<td>Aluminum</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1100-0</td>
<td>As received</td>
<td>110</td>
<td>97</td>
<td>62</td>
<td>-</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>37</td>
<td>0.33</td>
<td></td>
</tr>
<tr>
<td>2024-T3</td>
<td>—</td>
<td>469</td>
<td>378</td>
<td>422</td>
<td>0.032</td>
<td>0.005</td>
<td>0.30</td>
<td>0.02</td>
<td>558/1100</td>
<td>-0.124</td>
<td>-0.124</td>
<td>151</td>
<td>0.32</td>
<td></td>
</tr>
<tr>
<td>2024-T4</td>
<td>—</td>
<td>476</td>
<td>503</td>
<td>441</td>
<td>0.020</td>
<td>0.008</td>
<td>0.43</td>
<td>0.021</td>
<td>614/1015</td>
<td>-0.11</td>
<td>-0.11</td>
<td>175</td>
<td>0.37</td>
<td></td>
</tr>
<tr>
<td>5456-H3</td>
<td>—</td>
<td>400</td>
<td>234</td>
<td>359</td>
<td>-0.16</td>
<td>0.42</td>
<td>0.46</td>
<td>0.25</td>
<td>524/725</td>
<td>-0.11</td>
<td>-0.11</td>
<td>124</td>
<td>0.31</td>
<td></td>
</tr>
<tr>
<td>7075-T6</td>
<td>—</td>
<td>529</td>
<td>469</td>
<td>523</td>
<td>0.11</td>
<td>0.146</td>
<td>0.49</td>
<td>0.19</td>
<td>545/1315</td>
<td>-0.126</td>
<td>-0.126</td>
<td>176</td>
<td>0.30</td>
<td></td>
</tr>
</tbody>
</table>
LIST OF REFERENCES


7. Smith, B. L., Mean Strain Effects on the Strain Life Fatigue Curve, Naval Postgraduate School, 1993.
## INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center  
   Cameron Station  
   Alexandria, Virginia 22304-6145

2. Library, Code 52  
   Naval Postgraduate School  
   Monterey, California 93943-5002

3. CDR Duym, Code 31  
   Naval Postgraduate School  
   Monterey, California 93943-5000

4. Professor Lindsey, Code AA/Li  
   Naval Postgraduate School  
   Monterey, California 93943-5000

5. Professor Newberry, Code AA/Ne  
   Naval Postgraduate School  
   Monterey, California 93943-5000

6. LT Michael V. Skelly  
   USS AMERICA Air Department  
   FPO, New York, NY 09501