A METHODOLOGY FOR THE FATIGUE ANALYSIS OF LUG/PIN JOINTS

by

GERARD CARRIER

Department of Mechanical Engineering
McGill University
Montreal, Canada.

A Thesis submitted to the Faculty of Graduate Studies and Research in partial fulfillment of the requirements for the degree of Master of Engineering.

March, 1988

ISBN 0-315-45952-2

Abstract

Lug/pin joints are structural elements extensively used in mechanical design. Unfortunately they tend to have a low fatigue strength. This thesis presents a methodology based on advanced computational techniques for the analysis and design of metallic lug/pin joints. In the thesis, a particular lug/pin joint, typical of those found in helicopter rotors, will be studied but the methodology could be applied to different lug/pin joints. Many steps of the methodology could even be applied to the design of other metallic mechanical assemblies.

This methodology involves:

1. The construction of preprocessors that allow for 3-D geometrical design of lug/pin joints and the definition of a finite element model that considers the non- linearity induced by the clearances existing between the components.

2. The computation using the finite element method of the stress fields resulting from the application of extreme loading conditions.

3. The fatigue analysis of the lug/pin joint components using a "safe-life" approach. The probabilistic aspect of fatigue crack initiation is also considered.

In conclusion improvements to the design of the lug/pin joint under study will be proposed.

Résumé

Les joints boulonnés ("lug/pin joints" en anglais) sont des éléments structuraux très utilisés dans la conception de machines. Malheureusement, ils ont en général une faible résistance à la fatigue. Cette thèse présente une méthodologie informatisée pour l'analyse et la conception de joints boulonnés métalliques. Dans la présente thèse, la méthodologie sera appliquée à un joint particulier, typique de ceux qu'on retrouve sur les rotors d'hélicoptères. Elle pourrait être appliquée à d'autres joints. Plusieurs des étapes de la méthodologie pourraient même être appliquées à la conception d'autres assemblages mécaniques faits en métal.

La méthodologie comporte les étapes suivantes:

1. La construction de préprocesseurs qui permettent la conception d'un joint boulonné tridimensionnel et la définition d'un maillage d'éléments finis qui considère la non-linéarité due au jeu existant entre les différentes composantes du joint.
2. Le calcul par la méthode des éléments finis des contraintes résultant de l'application des conditions extrêmes de chargements.
3. L'analyse de fatigue des composantes du joint en utilisant la méthode "safe-life" (i.e. utilisation de courbes $S - N$ corrigeés). L'aspect aléatoire de l'initiation des fissures de fatigue sera aussi pris en considération.

Pour conclure, des améliorations au design du joint boulonné particulier sous étude seront proposées.

Acknowledgements

I would first like to thank Professor James W. Provan who supervised this research and whose enthusiasm was a source of motivation throughout my studies at McGill. The orientation Prof. Provan gave to the project was extremely valuable and made the investigation all more easier.

I am also grateful to Bell Helicopter Textron Canada (BHTC) that has supplied the data and the computer ressources necessary to the research. A special word of thank is given to Somen Chowdhury of BHTC who, through collaboration with Professor Provan, has made this research possible. I am also indepted to Wafaa Ishak, Alain Roquet, Guy Lambert, Edward Burczak, and many other BHTC employees for technical advices that helped me to carry out this research.

The financial assistance of the Natural Sciences & Engineering Research Council (NSERC) is gratefully acknowledged.

Contents

List of Figures

List of Tables

Chapter 1

INTRODUCTION

1.1 Motivation

"Fatigue" is a gradual deterioration of the strength of mechanical components resulting from the application of an unsteady load whose maximum value is lower, and sometimes much lower, than the static breaking load. Most mechanical failures are due to fatigue.

Some basic mechanical assemblies are particularly sensitive to fatigue. Lug/pin joints are among these and their fatigue strength may well determine the life an entire structure. In spite of this, lug/pin joints are structural elements extensively used in design because of the following advantages:

- they are easy to machine,
- they can be assembled and disassembled easily, and
- in addition, they can allow a relative rotation of the two components.

In many designs, lug/pin joints cannot be replaced by any other type of linkage.

The particularly low fatigue strength of lug/pin joints is generally due to:

1. fretting (i.e., wear due to relative alternating motion) at the lug hole (fatigue strength reduction of up to 90%), and
2. the high stress concentration factor at the lug (typically 3.5).

The present investigation has been undertaken:

- to identification more precisely the causes of the low fatigue strength of given lug/pin joint designs, and thus
- to be able to propose improvements to these designs.

1.2 Thesis Objectives

The main objective of this research is to develop a methodology for the "high cycle" i.e., long life, fatigue analysis of three dimensionnal lug/pin joints that can be adapted to allow geometrical changes (including the presence of clearance/interference fits), various loading conditions and different material properties. "Classical" methods [1,2] are limited to axially loaded bi-dimensional joints. The methodology is to be used to design more fatigue resistant lug/pin joints.

The present state of knowledge on fatigue does not permit very accurate evaluation of the endurance particularly in the high cycle range. The methodology developed in this thesis is to be used to *compare* alternative designs and select those that are worth testing. This methodology requires heavy computer work which may not be too expensive when compared to the time and expenses necessary to perform a purely experimental comparison of different lug/pin joint designs. Moreover, it can predict the most fatigue sensitive spots in a particular design and, therefore, make the establishment of an inspection procedure all more easier.

The main steps and techniques of the methodology developed in this research could be applied to the design of most mechanical metallic components.

1.3 Thesis Description

Chapter 2 of the thesis describes the particular lug/pin joint that will be analysed using the methodology developed here. It is representative of those found in helicopter rotors but should be seen as a non-restrictive example.

Chapter 3 deals with the finite element based stress analysis of the lug/pin joint under study. The "parametric drawing" concept will be explained and used. The finite element (F.E.) analysis will be carried out in two steps. A coarse three dimensional F.E. mesh will first be used to obtain the global stress state. A fine F.E. mesh will then be used to get more accurate values for the stress state in the lug which is the critical component of the assembly. Also computed are the stresses induced by the introduction of an interference fit bushing in the lug hole.

In Chapter 4, the fatigue analysis is performed using the so-called "safe life" approach whose principle is illustrated in Figure 1.1. The working $S-N$ diagrams will be established. The concept of "equivalent" stresses that are computed so as to be in a position to use the $S-N$ diagrams for multiaxial stress states will be introduced. A crack propagation criterion will also be established. The principles of cumulative damage and reliability computation will be discussed and applied to the lug/pin joint. Finally, the results of the analysis will be commented upon and possible improvements to the lug/pin joint described in Chapter 2 will be proposed.

Many programs have been written during the course of this research and are used in conjunction with the resident F.E. analysis software and pre/post processor. They are presented in the text. Their user guides and detailed codings in VAX FORTRAN are given in the Appendices.

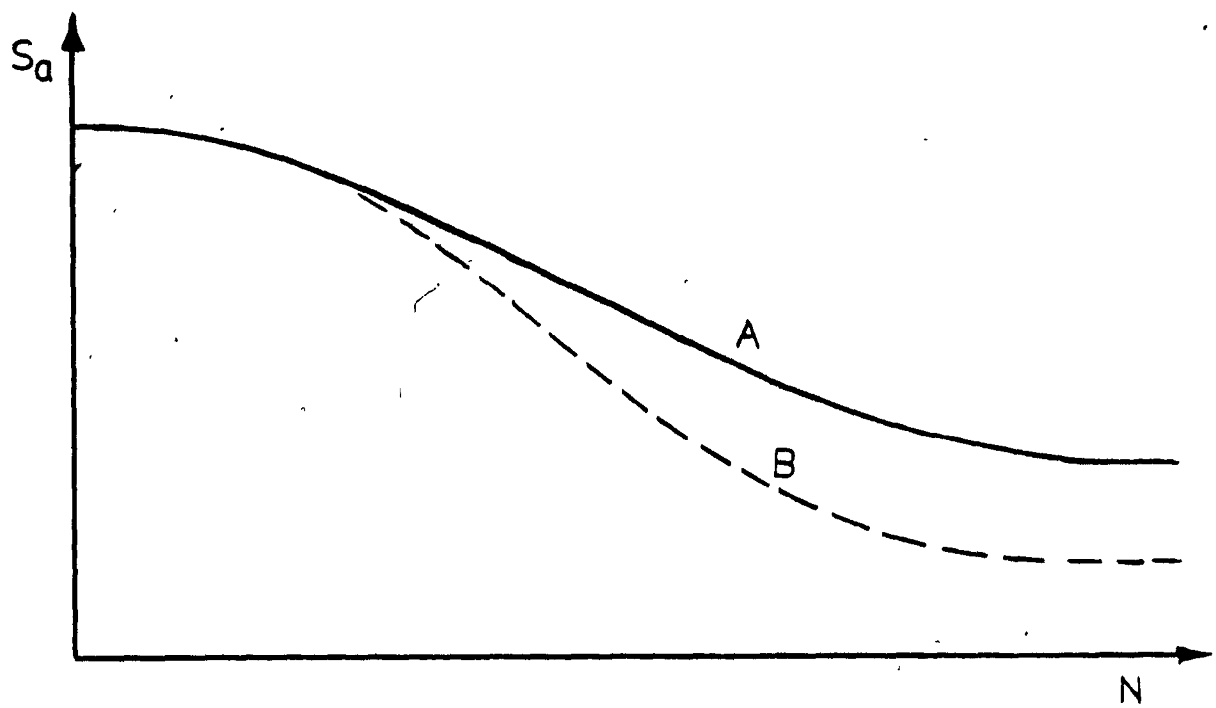


Figure 1.1: Safe-life approach

The "safe-life" approach consists in finding the number of loading cycles a component can support before failure. It is based on "$S-N$ diagrams".

curve A is the "basic $S-N$" diagram and represents the alternating stress S_a a plain specimen can sustain as a function of the "life" or "endurance" N (i.e., the number of cycles).

curve B is the "working $S-N$" diagram and represents the alternating stress an actual component can sustain once the working conditions have been taken into account.

Chapter 2

Description of the lug/pin joint assembly under study

To demonstrate the methodology that is being developed to study the fatigue of lug/pin joints, an actual lug/pin joint assembly will be analysed. As previously mentioned, this lug/pin joint is representative of those found in helicopter rotors but should be seen as a non-restrictive example. Its geometry, the components materials, and the oscillating loading conditions are again typical of those found in this industry.

2.1 Lug/pin joint geometry

Figures 2.1 to 2.3 show the geometry of this lug/pin joint. The pin geometry has been simplified by replacing its nut by a second "head" which make the pin symmetric. Moreover both "heads" of the pin are considered as cylinders.

Figure 2.4 shows the radial clearance and interference fits occurring between the components. Notice that the pin of the lug/pin joint under study is not tightened. Interference fit bushings have been introduced in the lugs to increase their fatigue resistance. Essentially, for a certain range of interference, the increase of the mean stress due to the interference fit bushing is more than compensated for by the decrease in alternating stress [3]. Moreover, the lug/bushing fretting is not as severe as the pin/lug fretting that would occur without the presence of the bushing.

All the components of the lug/pin joint under consideration have surface roughnesses of $125\mu in$ ($\approx 3\mu m$).

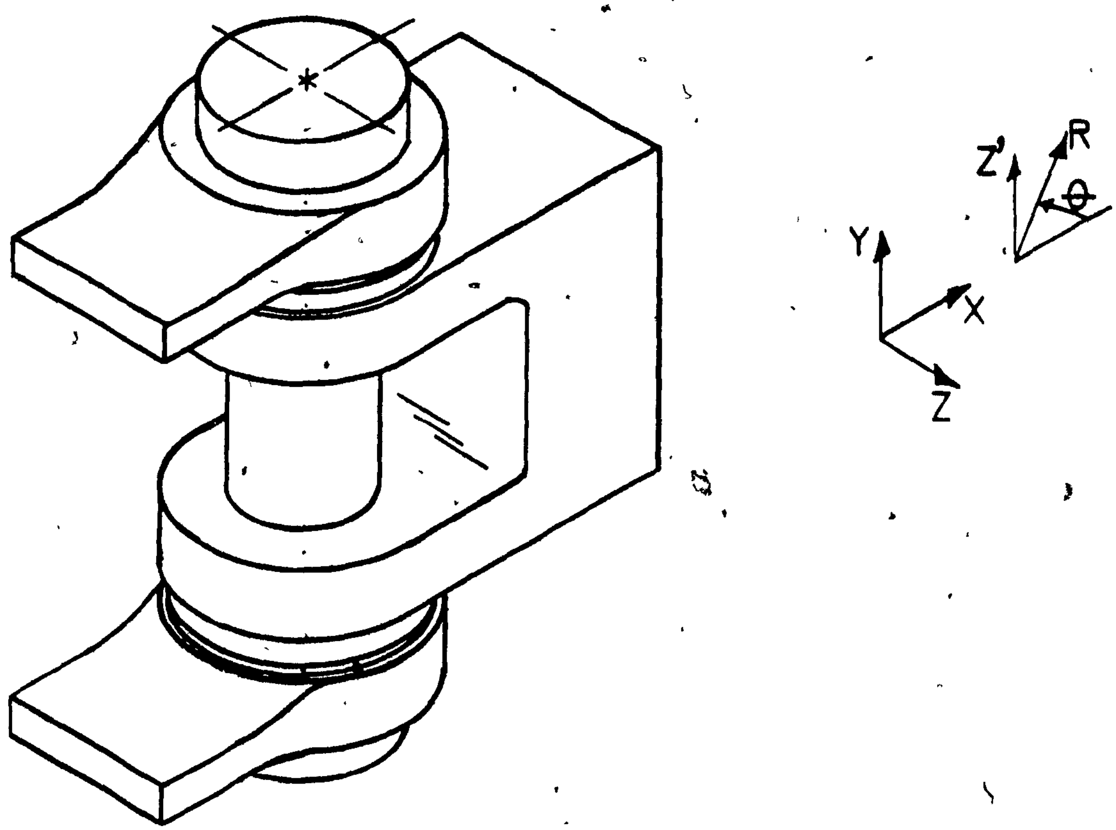


Figure 2.1: Isometric view of the lug/pin joint under study

Two coordinate systems are used in the study. The basic cartesian X, Y, Z coordinate system and the R, θ, Z' coordinate system. Their origin is at the middle of the pin axis.

The lug/pin joint has two planes of symmetry, the $X-Y$ plane and the $X-Z$ plane.

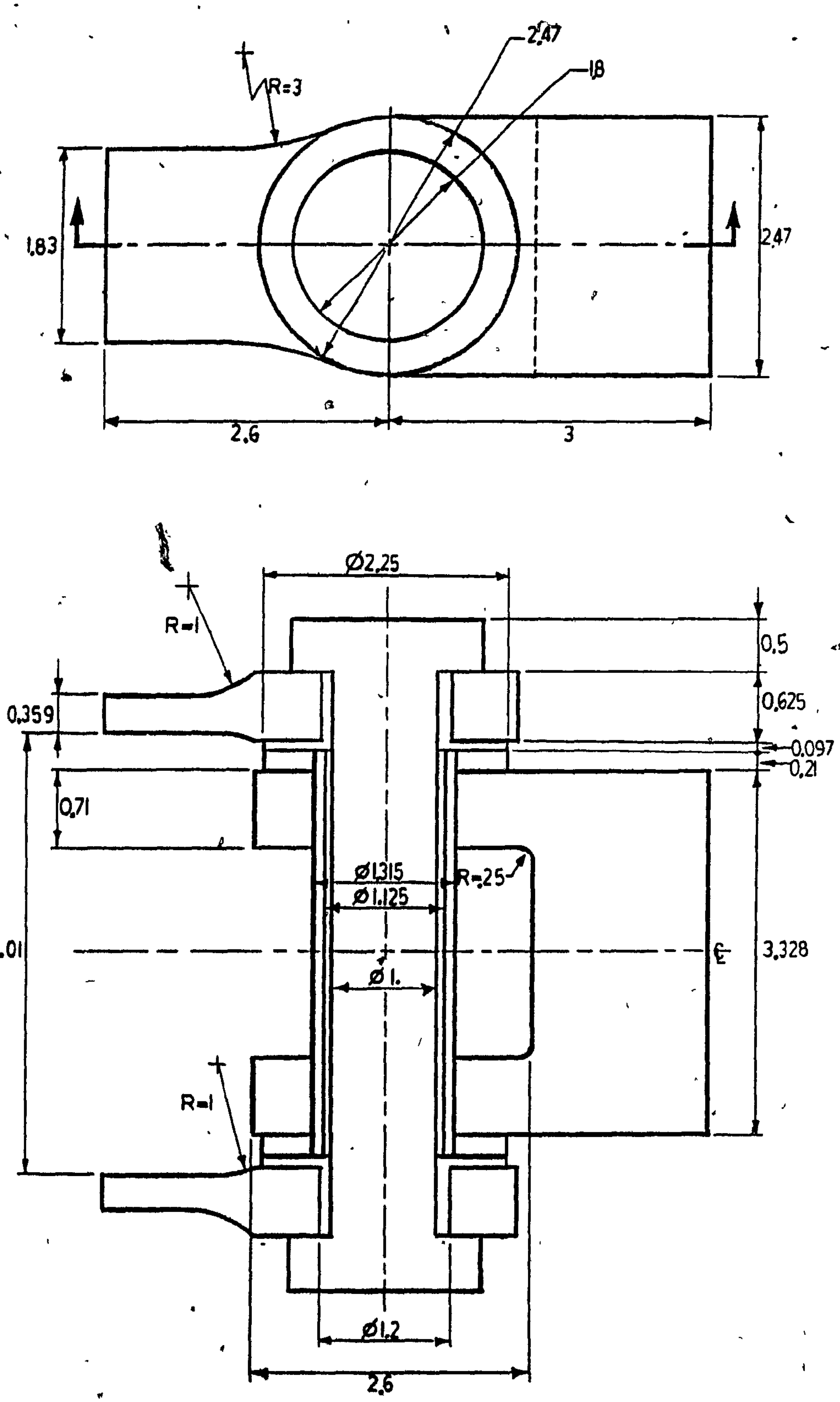


Figure 2.2: Top & section views of the lug/pin joint under study

Dimensions are in inches.

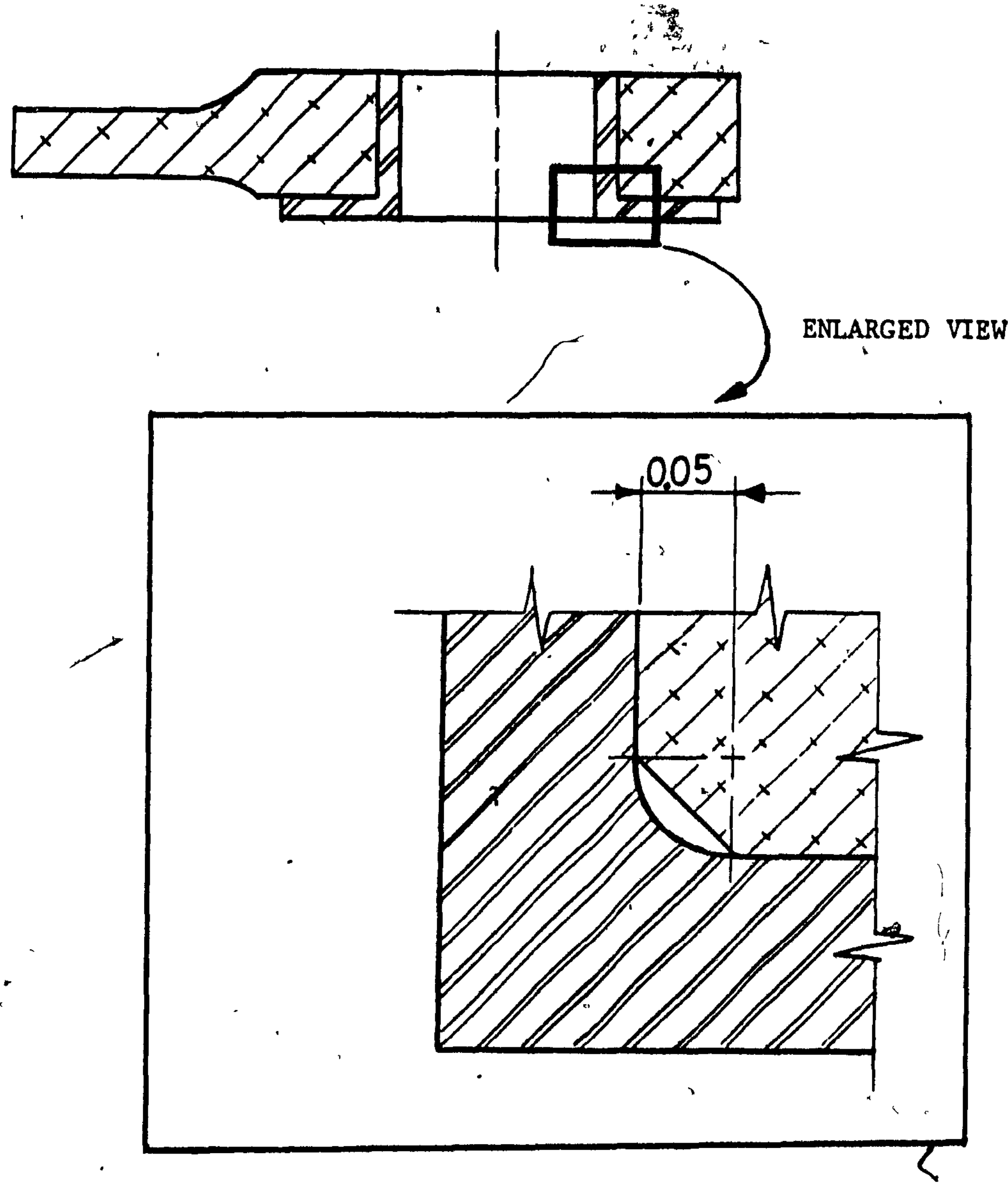


Figure 2.3: Lug & bushing geometrical detail

The bushing flange root is a stress raiser that was judged important enough to be modeled in the F.E. mesh (see Section 3.2.2). Its radius is 0.05*in*

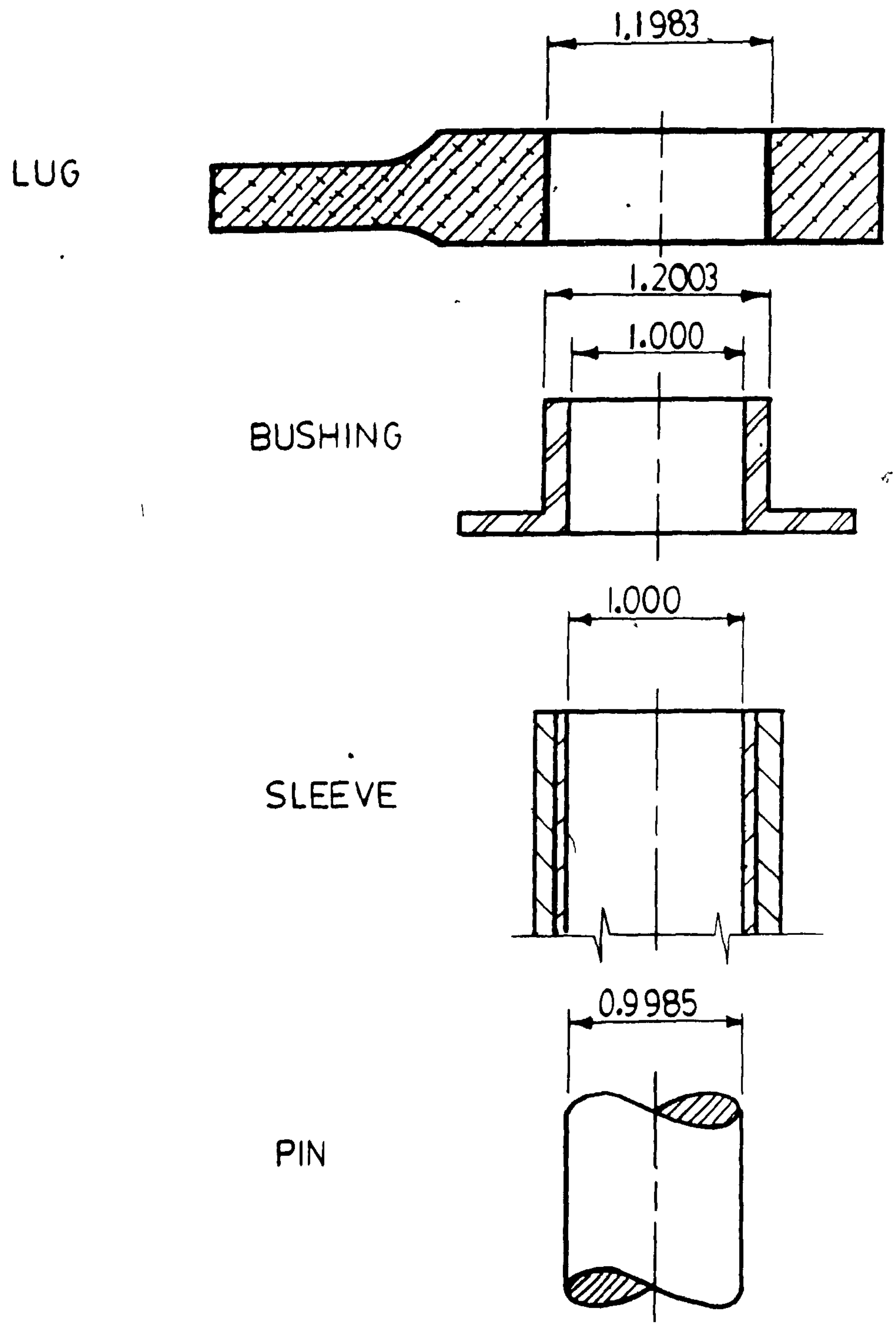


Figure 2.4: Radial clearance & interference fits between the components

The radial interference between the lug and bushing is $0.001in$. The pin/sleeve and pin/bushing radial clearance is $0.00075in$, ($0.0008in$ will be used for the computation because the machining cannot be accurate to $5/100000^{th}in$).

2.2 Lug/pin joint components materials

Table 2.1 shows the materials used to make the various components of the lug/pin joint. Notice that the blade is made of fiber glass. This material will be considered as isotropic. This approximation does not significantly modify the stress fields in the lugs which are the critical components of the assembly.

The bushes are plated with a $\approx 0.005in$ thick coating of cadmium to make the joint resistant to galvanic attack [4]. The aluminum lugs would be submitted to severe galvanic corrosion if put in direct contact with the stainless steel bushes since aluminum is anodic relative to ferric alloys.

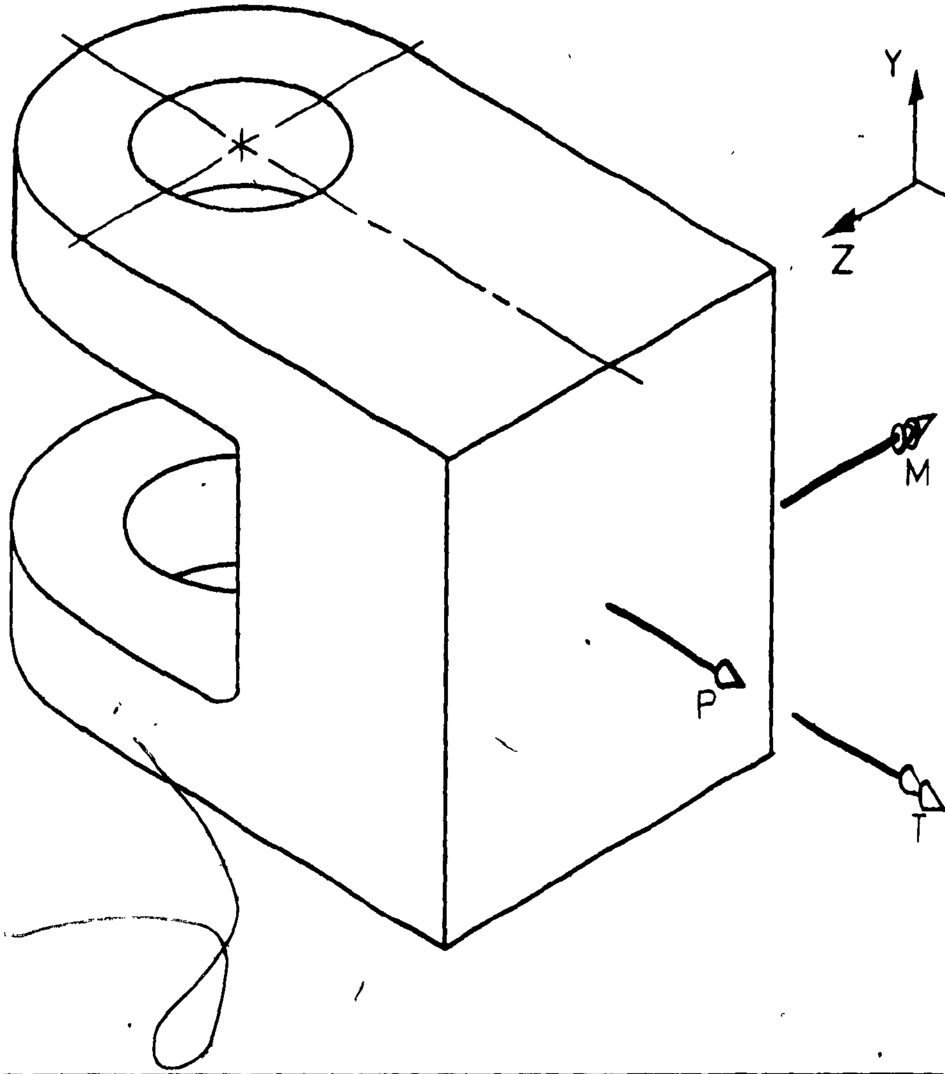
Component	Material	Young Mod. ($\times 10^3 ksi$)	Poisson Ratio	Yield Strength ($\times 10^3 ksi$)	Tensile Strength ($\times 10^3 ksi$)
BUSHING	17-4 PH Stainless S.	28.5	0.272	155.	170.
SLEEVE	15-5 PH Stainless S.	28.5	0.270	155.	170.
SLEEVE LINING	Copper	17 0	0.34	n.r.	n r.
PIN	4340 Steel	29.0	0.32	163.	180
LUG	7075-T73 Aluminum	10.0	0.33	59.	72.
BLADE	S. Fiber Glass Roving	7.3	0.3	n.r.	n r.
WASHER	15-5 PH Stainless S.	28.5	0.270	155	170

Table 2.1: Lug/pin joint component materials

n.r.: *not required for the current analysis.*

2.3 Extreme loading conditions

Figure 2.5 shows the extreme loading conditions that will be applied to the geometry under study. During the fatigue process, the loading oscillates 99.97% of the total number of cycles between the low load $LL1$ and low load $LL2$ values ("Low Amplitude Cycles"). It oscillates 0.03% of the total number of cycles between the high load $HL1$ and high load $HL2$ ("High Amplitude Cycles").



	Load ID	Tension P (lb)	Bending M ($lb \cdot in$)	Torsion T ($lb \cdot in$)
Low Amplitude Cycle	$LL1$	+16000	+12400	+1260
	$LL2$	+16000	−12400	−1260
High Amplitude Cycle	$HL1$	+17400	+23900	+2870
	$HL2$	+17400	−23900	−2870

Figure 2.5: Extreme loading conditions applied to the blade

Tension P. The force P is applied as a uniform normal stress on the blade section (see Figure 2.6).

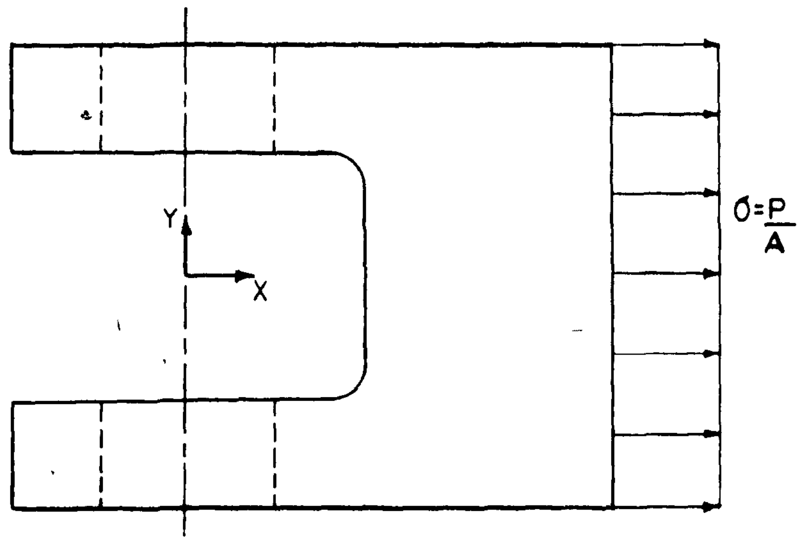


Figure 2.6: Application of the tension P on the blade

σ is the normal stress, A is the section area.

Bending Moment M. The bending moment M is applied as a linearly distributed normal stress on the blade section (see Figure 2.7).

Torsion T. Figure 2.8 shows how the torsion T has been applied. Ten punctual forces were used. Each of the punctual force applied is proportional to the local shear stress that would have resulted from the remote application of the torsion T.

Symmetric Loading Conditions.

Loading conditions $LL1$ and $LL2$ are symmetric: to know the stress field at a point located at R, θ, Z' when the lug/pin joint is submitted to $LL2$, one has only to check the stress field at the point R, θ, $-Z'$ position when the lug/pin joint is submitted to $LL1$.

Loading conditions $HL1$ and $HL2$ are symmetric too. The $HL2$ stress field can be deduced from the $HL1$ stress field.

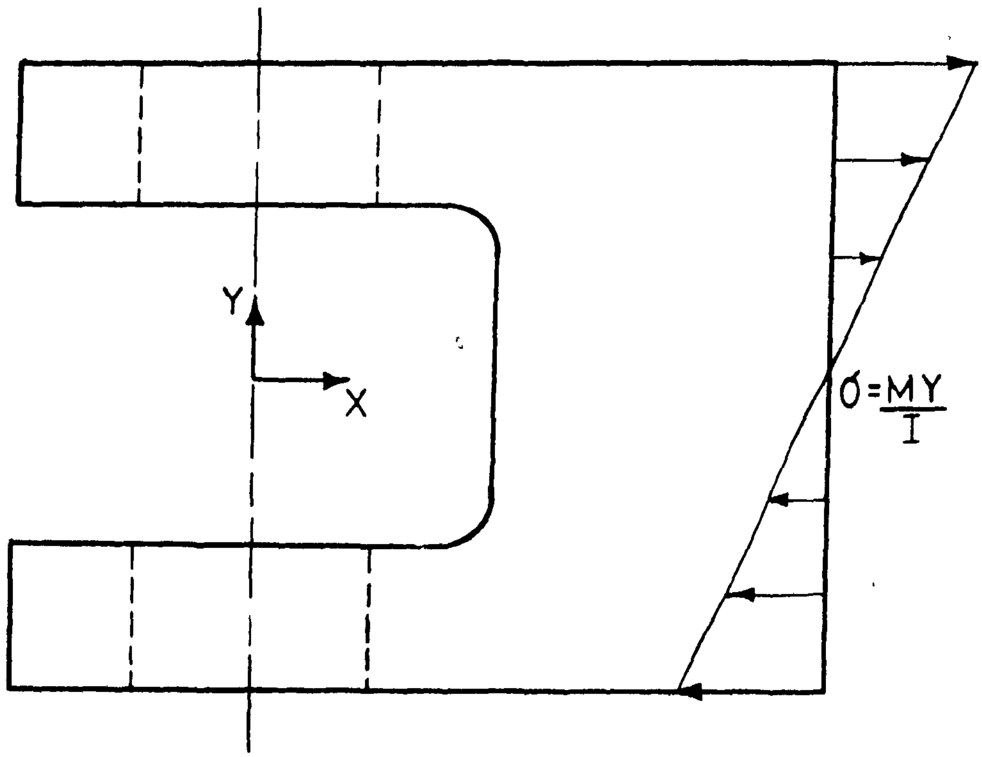


Figure 2.7: Application of the bending moment M on the blade whose section moment of inertia is I

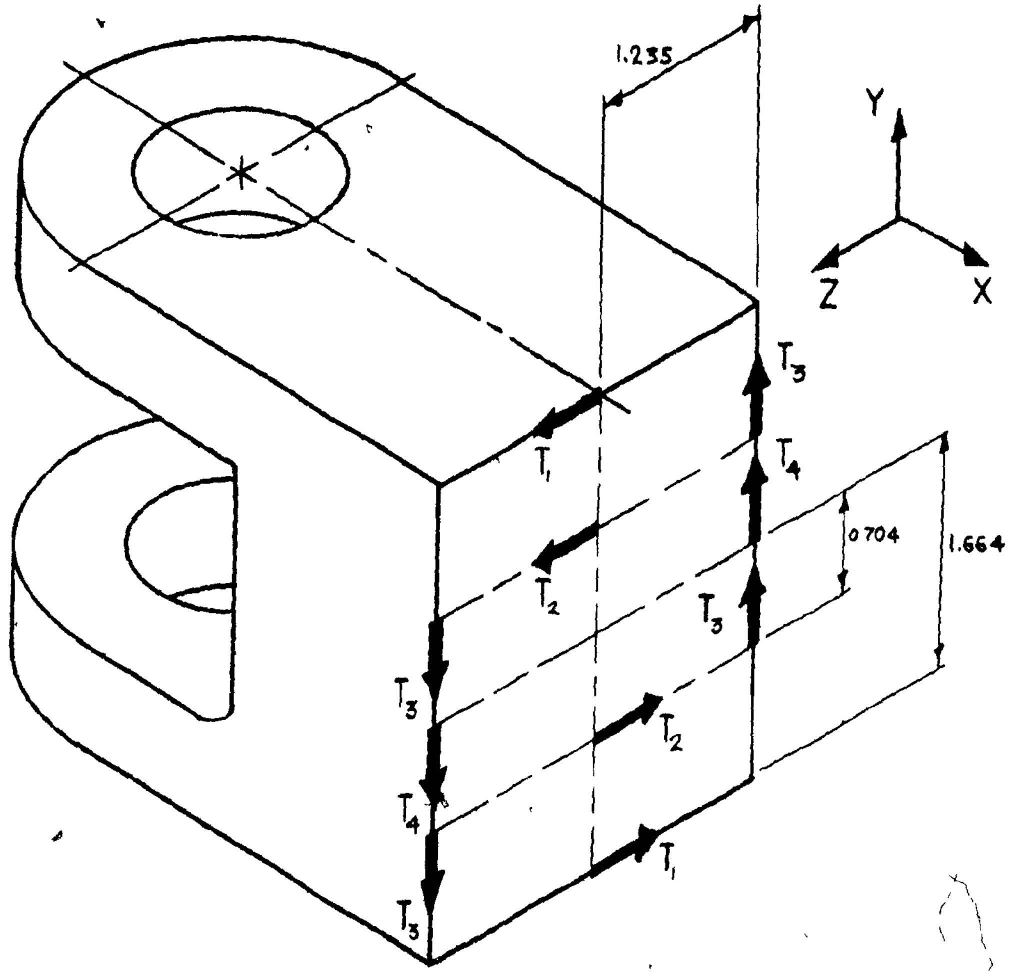


Figure 2.8: Application of the torsion T on the blade

$$
\begin{aligned}
T_1 &= 0.0873 \cdot T \\
T_2 &= 0.0235 \cdot T \\
T_3 &= 0.0878 \cdot T \\
T_4 &= 0.0981 \cdot T
\end{aligned}
$$

Chapter 3

Stress Analysis

The fatigue analysis of a component requires the knowledge of the stress fields resulting from the different applied loads. Because of the complexity of the lug/pin joint studied, a finite element method (F.E.M.) was selected to obtain a precise knowledge of the stress fields.

The NASTRAN finite element software was used throughout the analysis. The PATRAN pre & post-processor was used to define the lug/pin joint geometry, generate most of the NASTRAN inputs, and illustrate the NASTRAN outputs. The post-processing capacity of PATRAN will also be exploited to illustrate fatigue analysis results in Section 4.6.

3.1 Parametric Drawing

Improvement to a lug/pin joint design may be achieved through geometrical modifications. In this thesis, the concept of "parametric drawing" is exploited to simplify the modification of the lug/pin joint geometry. A parametric drawing is a drawing whose caracteristic dimensions are defined by parameters (Figure A.1 shows a "parametric drawing").

The program GEO (see Appendix A) uses the parametric drawing concept to write the inputs required by PATRAN to define a particular lug/pin joint geometry. As shown in Figure A.1 and Table A.1, twenty-four parameters completely define the geometry of a lug/pin joint of the type under study. Any of the 24 parameters can be modified, and with the help of the GEO program, PATRAN will be able to regenerate the new geometry, as long as the model topology is not modified (The topology is modified if 2 geometric entities, e.g., 2 lines, do or do not meet according to the values that define them).

The GEO program does not check for any inconsistent geometry or topological modification. PATRAN gives a warning or an error message if an incorrect geometry is entered in which case the user has to redefine some parts the geometry within PATRAN. Figure 3.1 shows the lug/pin joint under study while Figure 3.2 shows a *different* lug/pin joint whose topology is *similar*. Chamfers and radii smaller than $\frac{1}{8}$ inch are not considered at this stage. Neither are the clearance and interference fits. They will be when defining the finite element model. In PATRAN, the lug/pin joint component geometries are defined to avoid any discontinuity in any line or plane. This avoids any node mismatch when defining the finite element mesh. Figure 3.3 shows an example of this principle.

3.2 Finite Element Definition

The definition of the finite element mesh is performed with the help of PATRAN. Most of the solid, i.e., $3-D$ elements used in the finite element (F.E.) mesh are 8 node hexahedron elements (see Figure 3.4). To satisfy the various component geometries, some of the elements are five sided solid elements defined by 6 nodes (see Figure 3.5).

The definition of the F.E. mesh will make some nodes belonging to 2 different components coincident, i.e., located at the same position. The interactions between the various lug/pin joint components will be defined by the way their coincident nodes interact. Most of the interactions between coincident nodes are defined using non-linear "interface" elements (Section 3.2.3).

The use of $3-D$ elements with nodes at the vertices was mandatory because a non-linear approach, due to the presence of the non-linear interface elements, had to be used to compute the stress fields. NASTRAN does not accept high order elements in such analyses.

3.2.1 Finite Element Definition Program

A program named FED (Appendix B) has been written to define most of the finite elements. This simplified the analysis of different geometries that would result from the modification of the characteristic dimensions of the lug/pin joint (Section 3.1).

Upon running the FED program, files that are run as PATRAN session files are created. Tables B.1 and B.2 and Figure B.1 illustrate the use of the FED program.

```
*
*
PIN DIAMETER                          [in]    (PD) =1.
PIN HEAD DIAMETER                     [in]    (PHD)=1.8
PIN HEAD THICKNESS                    [in]    (PHT)=.5
*
SLEEVE OUT DIAMETER                   [in]    (SOD)=1.315
SLEEVE LINER DIAMETER                 [in]    (SLD)=1.125
*
WASHER THICKNESS                      [in]    (WT) =.21
*
BUSHING OUT DIAMETER                  [in]    (BOD)=1.2
BUSHING BASE OUT DIAMETER             [in]    (BBD)=2.25
BUSHING BASE THICKNESS                [in]    (BBT)=.097
*
LUG THICKNESS #1                      [in]    (LT1)=.625
LUG THICKNESS #2                      [in]    (LT2)=.359
LUG "IN" LENGTH                       [in]    (LIL)=4.01
LUG OUT DIAMETER                      [in]    (LOD)=2.47
LUG WIDTH                             [in]    (LW) =1.83
LUG REFERENCE TO PIN AXIS DISTANCE    [in]    (LRA)=2.6
LUG "IN" RADIUS                       [in]    (LIR)=1.
LUG "OUT" RADIUS                      [in]    (LOR)=1.
LUG SUPPORT RADIUS                    [in]    (LSR)=3.
*
BLADE THICKNESS                       [in]    (BT) =3.328
BLADE SUPPORT DIAMETER                [in]    (BSD)=2.47
BLADE SUPPORT THICKNESS               [in]    (BST)=.71
BLADE SUPPORT LENGTH                  [in]    (BSL)=2.6
BLADE SUPPORT FILLET RADIUS           [in]    (BSF)=.25
BLADE REFERENCE TO PIN AXIS DISTANCE  [in]    (BRD)=3.
*
```

Table 3.1: GEO program input file used to generate the lug/pin joint under study (See Figure 3.1)

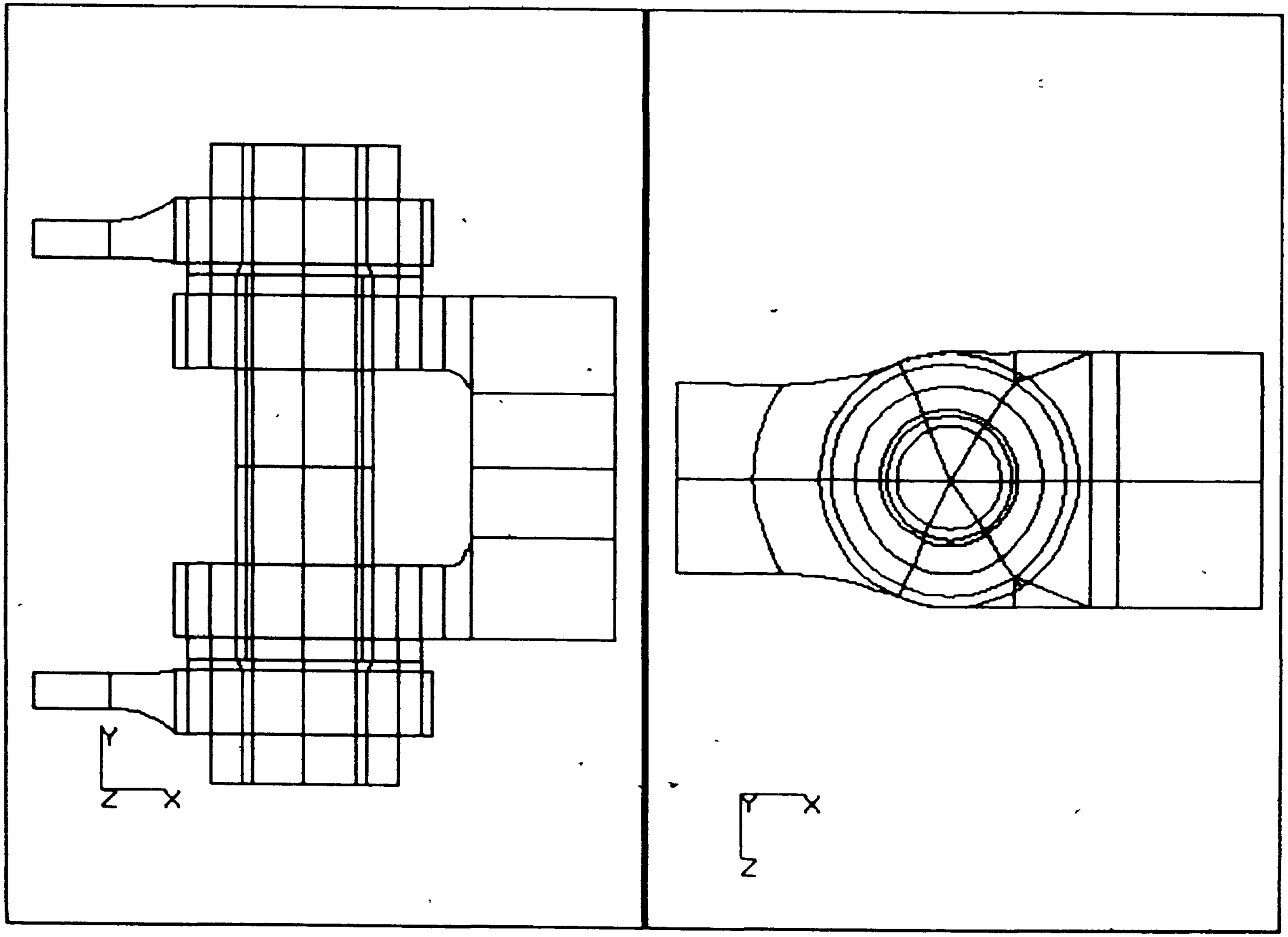


Figure 3.1: Lug/pin Joint Under Study

The dimensions are given in Table 3.1.

```
*
*
PIN DIAMETER                              [in]     (PD) =1.
PIN HEAD DIAMETER                         [in]     (PHD)=1.8
PIN HEAD THICKNESS                        [in]     (PHT)=.5
*
SLEEVE OUT DIAMETER                       [in]     (SOD)=1.315
SLEEVE LINER DIAMETER                     [in]     (SLD)=1.125
*
WASHER THICKNESS                          [in]     (WT) =.21
*
BUSHING OUT DIAMETER                      [in]     (BOD)=1.33
BUSHING BASE OUT DIAMETER                 [in]     (BBD)=2.25
BUSHING BASE THICKNESS                    [in]     (BBT)= 097
*
LUG THICKNESS #1                          [in]     (LT1)=.725
LUG THICKNESS #2                          [in]     (LT2)=.359
LUG "IN" LENGTH                           [in]     (LIL)=4.01
LUG OUT DIAMETER                          [in]     (LOD)=2.34
LUG WIDTH                                 [in]     (LW) =1.83
LUG REFERENCE TO PIN AXIS DISTANCE        [in]     (LRA)=2.6
LUG "IN" RADIUS                           [in]     (LIR)=1.
LUG "OUT" RADIUS                          [in]     (LOR)=1.
LUG SUPPORT RADIUS                        [in]     (LSR)=3.
*
BLADE THICKNESS                           [in]     (BT) =3.328
BLADE SUPPORT DIAMETER                    [in]     (BSD)=2.47
BLADE SUPPORT THICKNESS                   [in]     (BST)=.71
BLADE SUPPORT LENGTH                      [in]     (BSL)=2.6
BLADE SUPPORT FILLET RADIUS               [in]     (BSF)=.25
BLADE REFERENCE TO PIN AXIS DISTANCE      [in]     (BRD)=3.
*
```

Table 3.2: GEO program input file used to generate a lug/pin joint topologically similar to the lug/pin joint under study

The following values have been changed (refer to Table 3.1).

1. $BOD = 1.33$ instead of 1.2
2. $BBD = 2.10$ instead of 2.25
3. $LT1 = 0.725$ instead of 0.625
4. $LOD = 2.34$ instead of 2.47

(See Figure 3.2)

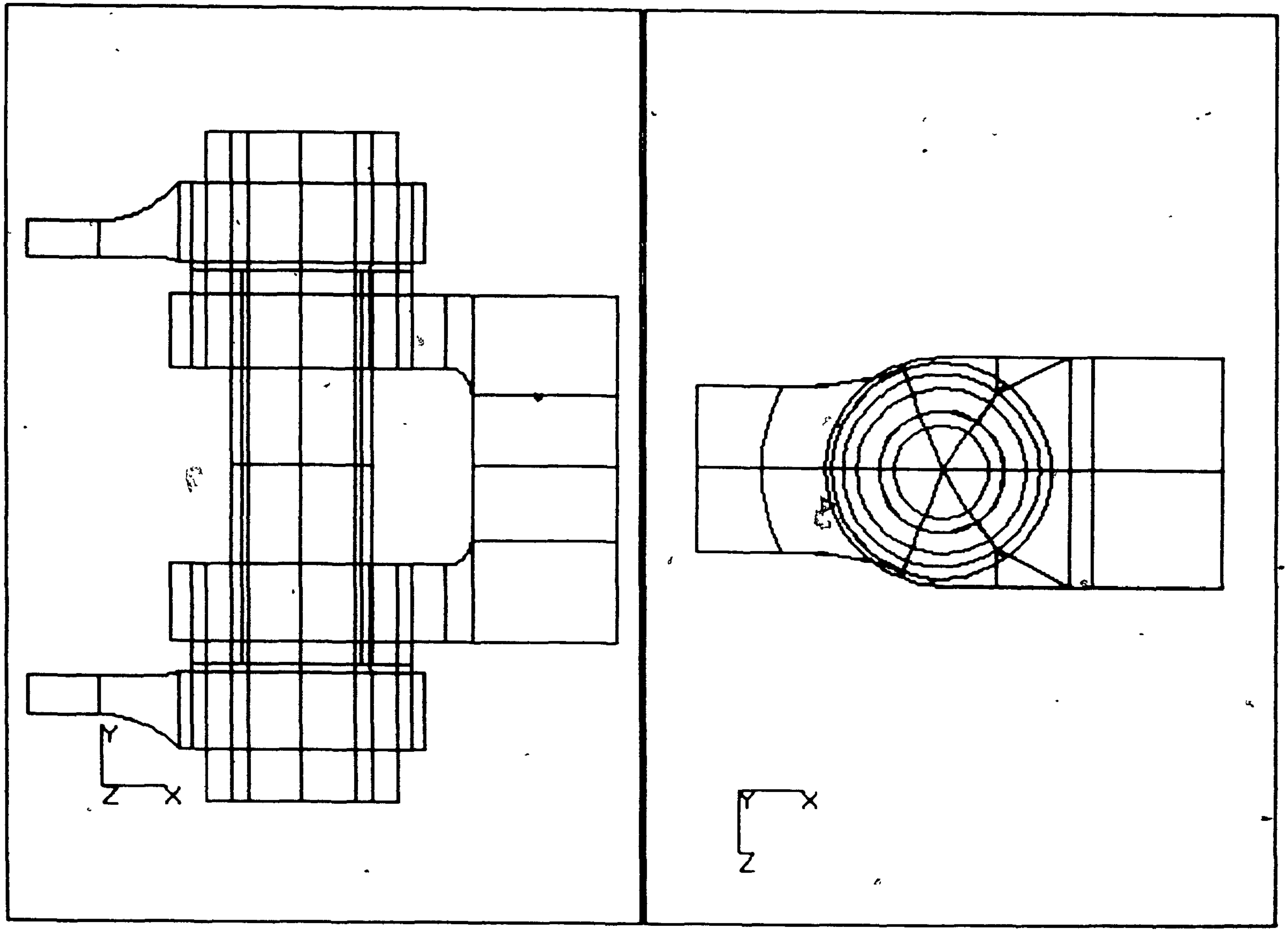
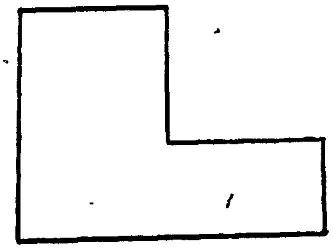


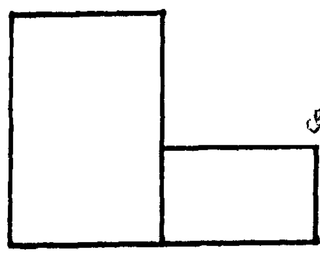
Figure 3.2: Lug/pin joint topologically similar to the lug/pin joint under study

The dimensions are given in Table 3.2.

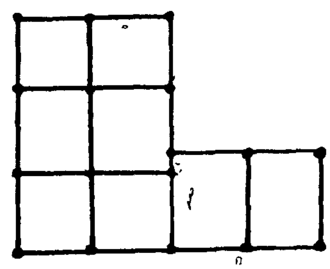


(a)

Suppose a bi-dimensional F.E. mesh has to be created on the geometry shown in Figure 3.3(a).

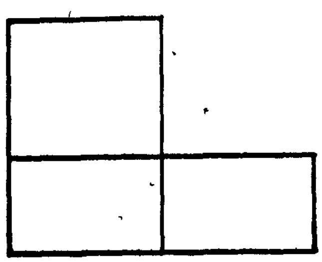


(b)

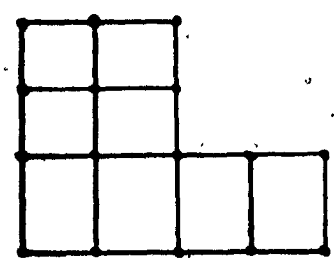


(c)

Using the preprocesser, one may define this shape using 2 rectangles (Figure 3.3(b)) and then, define the finite element mesh in each rectangle with Figure 3.3(c). If no special care is taken, the nodes on the common boundary of the rectangle will probably not match as shown in Figure 3.3(c).



(d)



(e)

However, if the shape is defined by 3 rectangles, Figure 3.3(d), the F.E. mesh will always match, Figure 3.3(e).

This principle has been applied to the definition of the lug/pin joint geometry.

Figure 3.3: Example of geometrical continuity principle

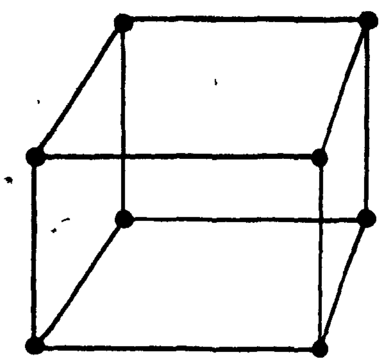


Figure 3.4: Eight node hexahedron element

This type of element is named “HEXA” in NASTRAN.

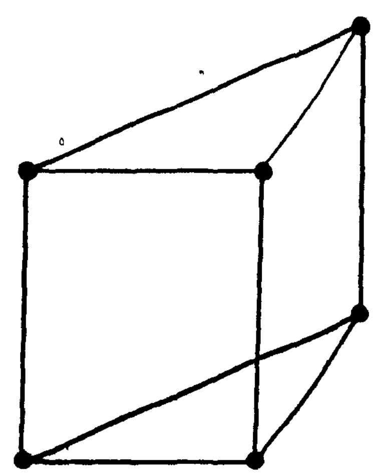


Figure 3.5: Six node five sided element

This type of element is named “PENTA” in NASTRAN.

The user may have to edit the FED output files and modify slightly the PATRAN statements to fit his needs or he may modify the F.E. mesh once the FED output files have been executed by PATRAN.

3.2.2 Finite Element Analysis Methodology

The following two step method, illustrated in Figure 3.6, has been used to obtain an accurate knowledge of the stress field in the lugs.

First step: Global analysis of the lug/pin joint using a coarse F.E. mesh.

For the *LL1* and *HL1* loadings shown in Figure 2.5, a coarse F.E. mesh modeling the complete geometry was used to find the following boundary displacement fields:

1. the radial displacement imposed by the pin on the bushings,
2. the axial (Z') displacement imposed by the washers to the bushings,
3. the axial displacement imposed by the pin heads to the lugs.

The friction between the components has been neglected. If it had not been neglected, it would have been necessary to consider the R, θ, Z' displacements at the 3 interfaces given above.

The mesh, shown in Figure 3.7, has 1625 nodes, 864 solid elements (HEXA or PENTA), and 468 gap elements used to simulate the effects of clearance/interference fits between the components (Section 3.2.3). Table 3.3 shows the inputs to the FED program that were used to generate the model. Some HEXA elements in the F.E. model generated by the FED output file had angles near 180°, they were replaced by PENTA elements (Figure 3.8). As previously mentioned, the analysis was non-linear and therefore required a large amount of computer time. As explained in Chapter 2, the stress and displacements fields induced by the loading conditions *HL2* and *LL2* could be deduced from the *HL1* and *LL1* stress and displacement fields.

Second step: Detailed analysis of the lug using a fine F.E. mesh.

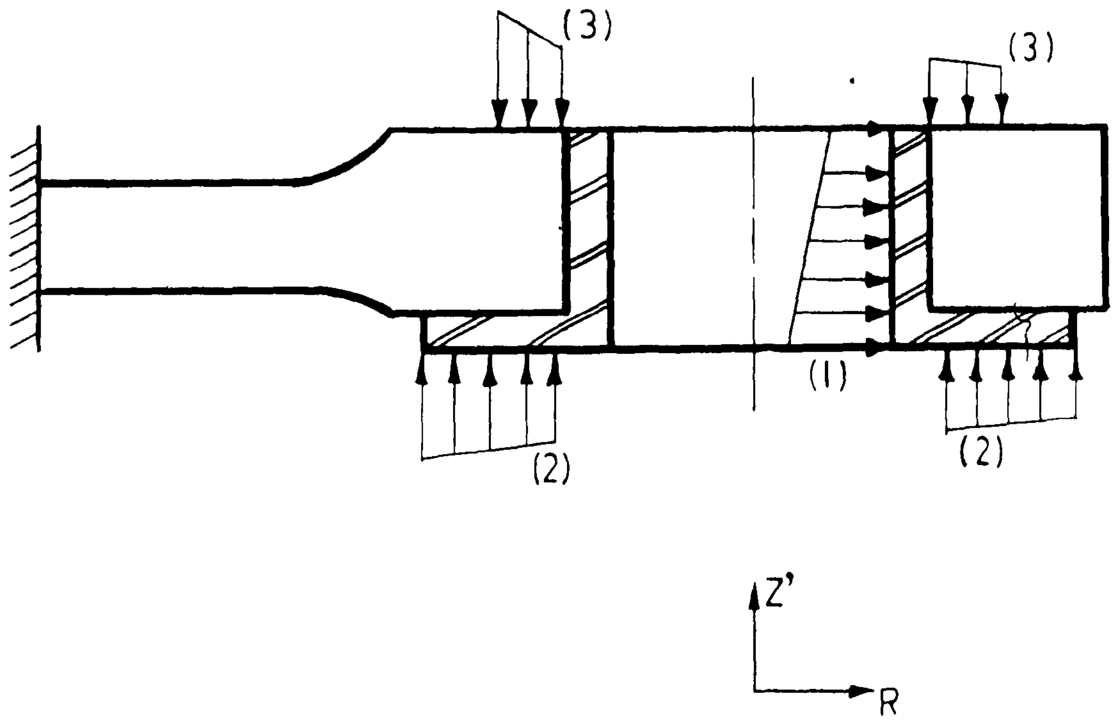


Figure 3.6: Boundary displacements applied to the lug & bushing

The first step of the method was used to get what will be called "the boundary displacements" (1), (2), (3). The second step used the boundary displacements (1), (2), (3) to accurately compute the stress fields in the bushing and the lug. A fine mesh was used.

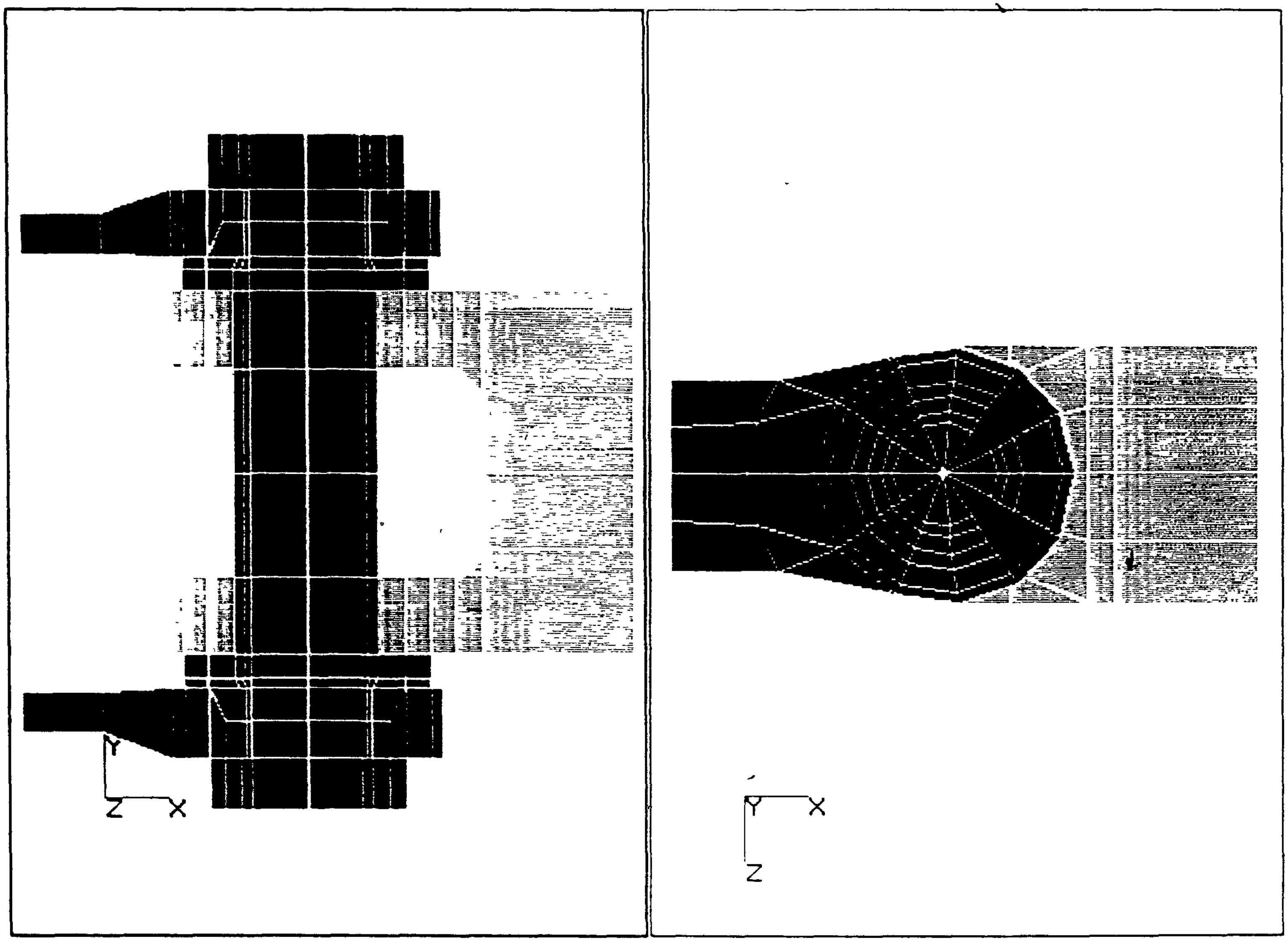


Figure 3.7: Coarse F.E. mesh used to model the lug/pin joint

```
*
*     LUG & PIN F.E. DEFINITION (refer to Figure B.1)
*
L1 =2
L21=1
L22=1
L3 =1
L4 =1
L5 =1
L6 =2
L7 =1
*
B1 =1
B2 =2
*
P1 =1
P2 =1
P21=1
P3 =1
*
W1 =1
*
D1 =1
D2 =1
D21=1
D22=1
D3 =1
D4 =1
D5 =1
D6 =1
*
S1 =1
S2 =1
*
```

Table 3.3: FED program input file used to generate the coarse F.E. mesh

The file format is explained in Appendix B. See also Figure 3.7

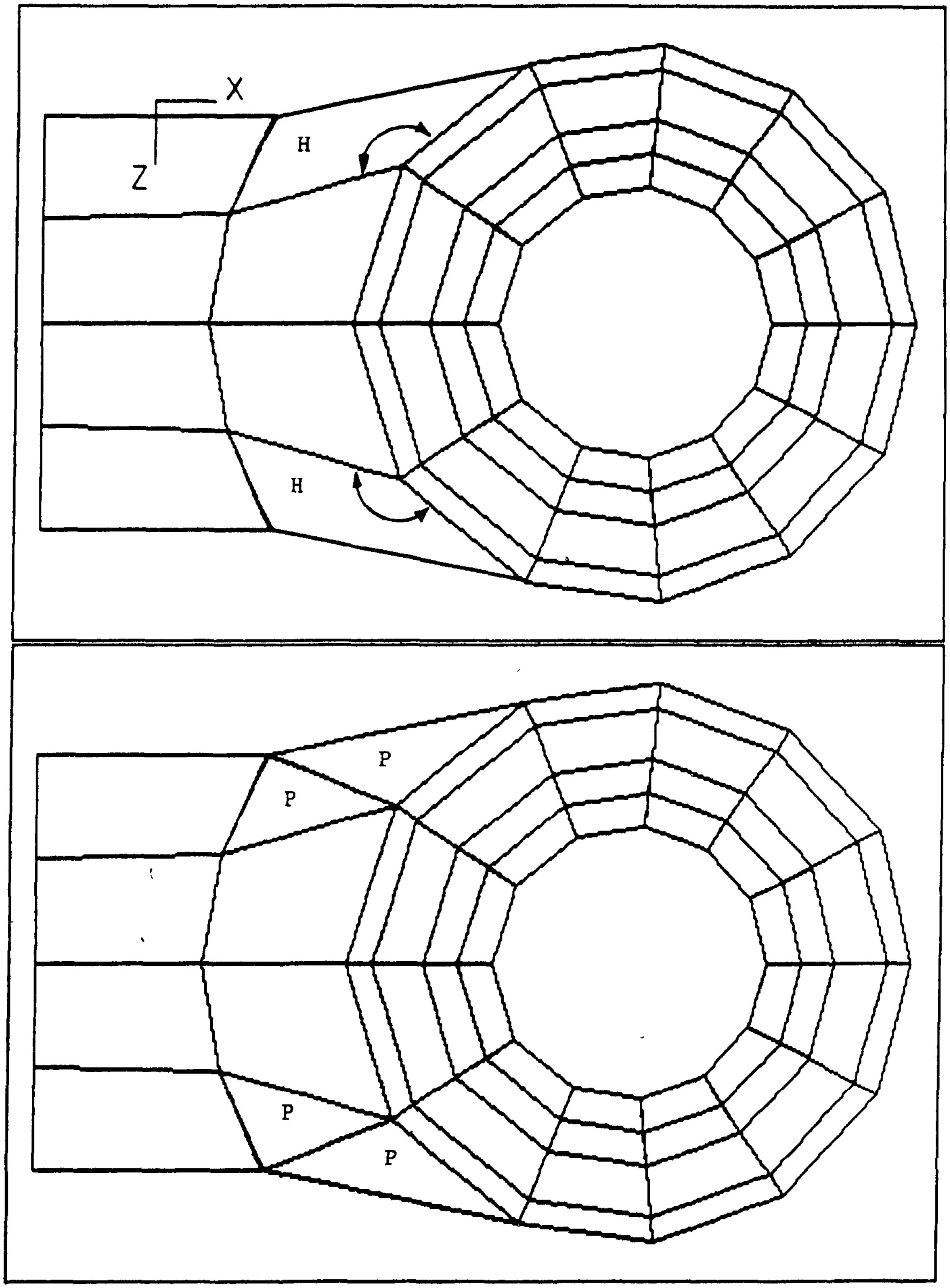


Figure 3.8: Skew elements replacement

The hexahedron elements having angles near 180° (indicated by "H") have been replaced by five sided solid elements (indicated by "P").

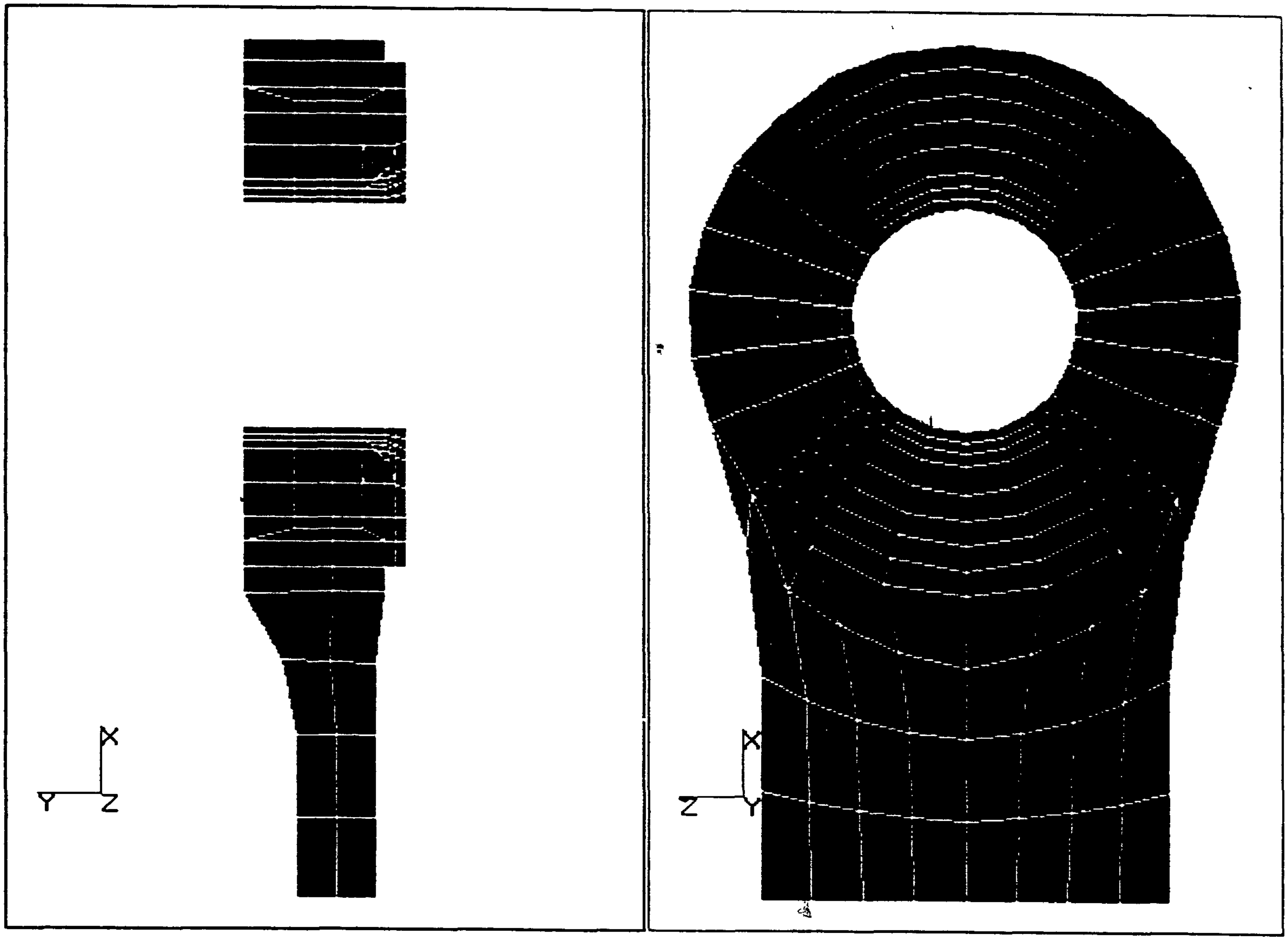


Figure 3.9: Lug & bushing fine F.E. mesh.

The boundary displacements (1), (2), (3), computed in the first step were applied to the bushing and lug model. A fine mesh that takes into account the stress concentration factors was used. The model, shown in Figure 3.9, has 1932 nodes, 1296 solid elements and 240 gap elements. The results of the second step has provided a much more accurate answer.

Table 3.4 shows the input to the FED program that were used to generate the model. Only the FED output files related to the lug and bushing were kept. Some of their instructions were modified to get a coarser mesh far from the bushing and a fine mesh at the bushing notch root (Tables 3.5, 3.6). As in the first step, some HEXA elements having angles near 180° were replaced by PENTA elements.

The OFFSET program (see Appendix C) was used to model the lug chamfer and the radius at the lug flange root (see Figures 2.3, 3.10, 3.11). Inputs and outputs to the OFFSET program that were used are shown in Tables 3.7 and 3.8. Some elements belonging to the lug had to be deleted (see Figure 3.11).

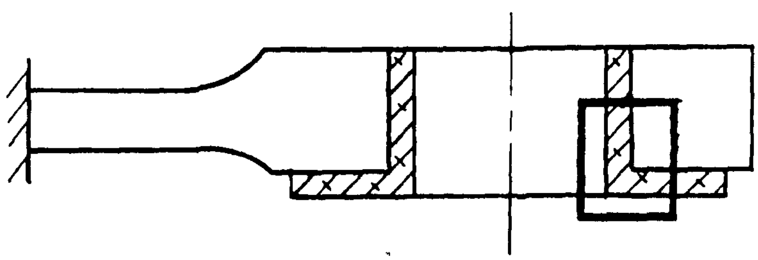


Figure 3.10: Lug chamfer and bush flange root details

(See Figure 3.11)

3.2.3 Component Interface Modeling

The lug/pin joint under study is made of 9 components. This section of the thesis explains how the interactions between these components were modeled taking into

```
*
*     LUG & BUSH F.E. DEFINITION (refer to Figure B.1)
*
L1 =4
L21=2
L22=2
L3 =1
L4 =4
L5 =2
L6 =4
L7 =2
*
B1 =2
B2 =3
*
P1 =1
P2 =1
P21=1
P3 =1
*
W1 =1
*
D1 =1
D2 =1
D21=1
D22=1
D3 =1
D4 =1
D5 =1
D6 =1
*
S1 =1
S2 =1
*
```

Table 3.4: FED program input file used to create the fine F.E. mesh

Only the output files related to the lug and bushing were kept.

```
$
$                 LUG F.E. DEFINITION
$
GF, H103/114 ,,  5/3/5/2.25,2,ED9
GF, H106/107/117/118 ,, 5/3/5/2.25,2,ED9
GF, H104/115 ,,  3/3/5/1.717071,2 ,F3
GF, H108/109/119/120 ,, 3/3/5/1.717071,2 ,F3
GF, H101 ,,3/5/3,2
GF, H102 ,,5/3/3,2
GF, H105 ,,3/5/2/1.717071,2
GF, H110/111 ,,2/3/5///1.717071,2
GF, H112 ,,5/3/3,2
GF, H113 ,,3/5/3,2
GF, H116 ,,5/3/2//1.717071,2
GF, H121/122 ,,5/2/3///1.717071,2
 CF,H101T122,HEX,,M1,1001T3999
```

Table 3.5: Modified FED output file used to generate the lug fine F.E. mesh (See Figure 3.9)

```
$
$            BUSHING F.E. DEFINITION
$
GF,H601/613 ,,3/4/5,2,ED9
GF,H605/606/617/618 ,,3/4/5,2,ED9
GF,H602/614 ,,3/3/5,2,ED9
GF,H604/616 ,,4/5/5,2,ED9
GF,H607/608/619/620 ,,3/3/5,2,ED9
GF,H611/612/623/624 ,,4/5/5//2.25,2,ED9
GF,H621/622 ,,3/5/3,2
GF,H609/610 ,,5/3/3,2
GF,H603 ,,5/3/3,2
GF,H615 ,,3/5/3,2
 CF,H601T624,HEX,,M6,6001T6999
```

Table 3.6: Modified FED output file used to generate the bushing fine F.E. mesh (See Figure 3.9)

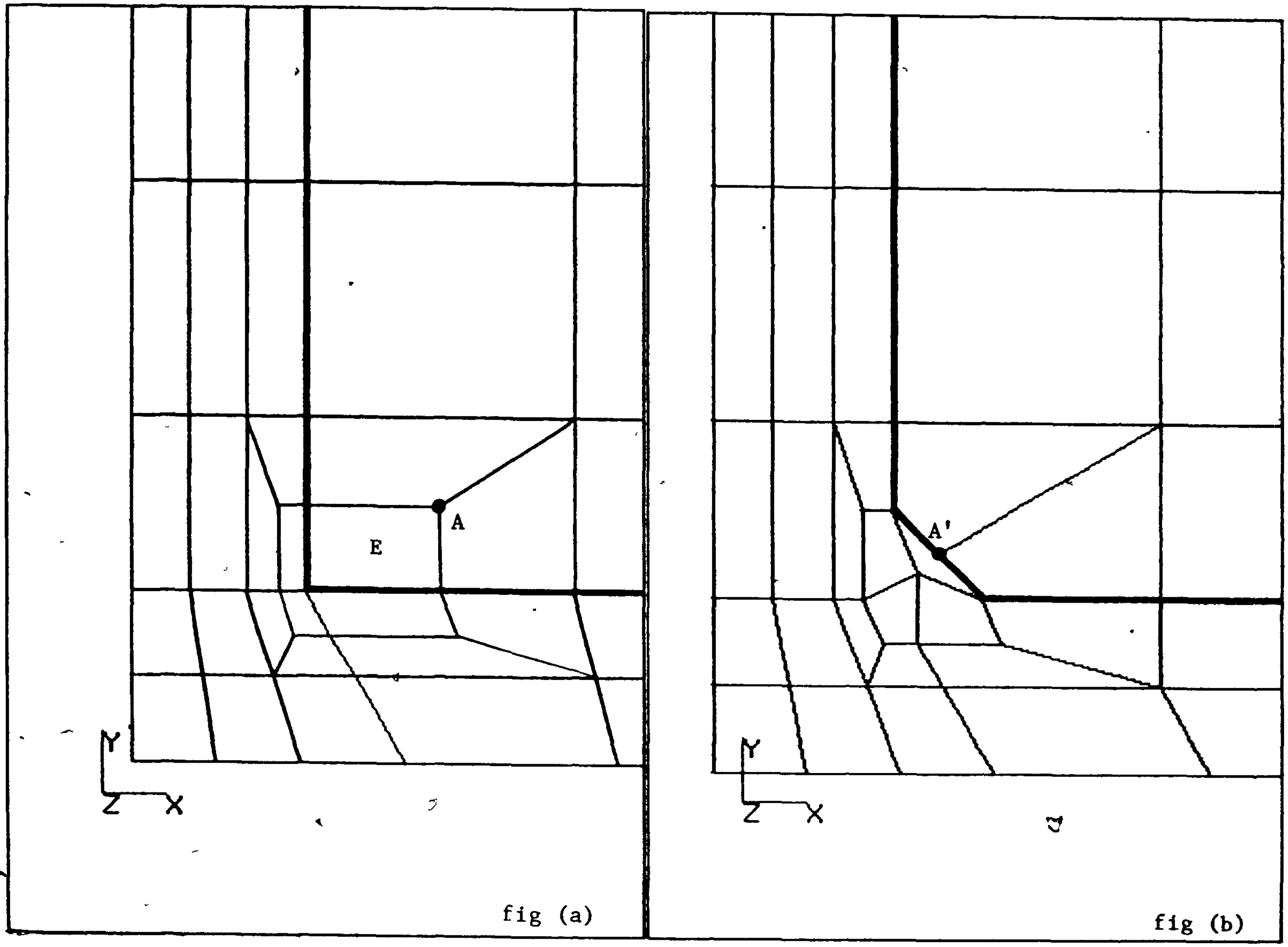


Figure 3.11: F.E. mesh at the bushing flange root and lug chamfer

Figure a) shows the mesh as produced by the files shown in Tables 3.5 and 3.6

Figure b) shows the mesh once modified by the Table 3.8 instructions. Note that the element indicated by "E" in a) has also been deleted.

```
*
*.    BUSH FLANGE ROOT MODEL
*
PATRAN.OUT FILE       :[MCGO1.F]PATRAN.OUT
*
* R1    R2    TETA1   TETA2     Z'1        Z'2     DR      DZ'
*
 .599 .601    -360.   360.   1.970      1.972 .014645 .014645
 .582 .601    -360.   360.   2.016      2.019      0.  .00358
 .674 .676    -360.   360.   1.970      1.972 -.025 0.
 .764 .767    -360.   360.   1.921      1.924 -.014375 0.
 .674 .676    -360.   360.   2.016      2.019 -.05 -.02142
 .684 .687    -360   360.   1.945 1.947 -.025 0.
```

Table 3.7: OFFSET program input file used to model the bushing flange root and the lug chamfer

See Table 3.8

```
NOD,  826,OFF,-0.005585, 0.014645,-0.013538
NOD,  894,OFF,-0.009313, 0.014645,-0.011302
NOD,  895,OFF,-0.002104, 0.014645,-0.014493
NOD,  988,OFF,-0.012171, 0.014645,-0.008145
NOD,  989,OFF, 0.001451, 0.014645,-0.014573
NOD, 1082,OFF,-0.014000, 0.014645,-0.004299
NOD, 1085,OFF, 0.004922, 0.014645,-0.013793
NOD, 1173,OFF,-0.014645, 0.014645, 0.000000
NOD, 1184,OFF, 0.008145, 0.014645,-0.012171
NOD, 1249,OFF,-0.014000, 0.014645, 0.004299
NOD, 1278,OFF, 0.010874, 0.014645,-0.009810
NOD, 1343,OFF,-0.012171, 0.014645, 0.008145
NOD, 1380,OFF, 0.012918, 0.014645,-0.006899
NOD, 1470,OFF,-0.009313, 0.014645, 0.011302
NOD, 1514,OFF, 0.014200, 0.014645,-0.003581
                .
                .
                .
NOD, 1769,OFF,-0.008402, 0.000000,-0.023546
NOD, 1773,OFF,-0.018563, 0.000000,-0.016746
NOD, 1777,OFF,-0.013904, 0.000000,-0.020777
```

Table 3.8: OFFSET program output file used to modify the F.E. mesh near the bushing flange root

The OFFSET input file is given in Table 3.7

It was executed as a PATRAN session file to get Figure 3.11(b) after the mesh shown in Figure 3.11(a) had been generated.

account clearance and interference fits where necessary.

Interactions between the components were simulated using the following means: "equivalencing", "multipoint constraints", "bar elements", and "gap elements". These actually define interactions between pairs of coincident nodes that belong to 2 different components. To insure that "coincident" nodes have exactly the same coordinates (to the last digit), these are rounded off in the NASTRAN data deck. The TRONC program was used to perform this task (see Appendix G).

There are 12 possible interactions between the lug/pin joint components. The Table 3.9 and Figure 3.12 show how they have been included in the model.

Clearance Fit Modeling

Clearance or exact fit are modeled using NASTRAN "GAP" elements [5]. Gap elements have a very high rigidity when closed and a very low rigidity (but still greater than zero) when open (Figure 3.13).

Since gap elements are non-linear, the solution of a F.E. model with gap elements is found through an iterative process. NASTRAN solution 66 "material non-linear" was used. The ITER method on the NASTRAN NLPARM card was selected [5].

The gap elements were not defined in the PATRAN preprocessor and, hence, a program was written to perform this task (Appendix D). The PATRAN to NASTRAN translator, "PATNAS", can generate GAP elements but, because of the number of interactions between the components (see Table 3.9), it is cumbersome to use. From a neutral file, PATNAS can generate only one set of GAP elements having the same property (i.e., referring to the same PGAP card). The generation of GAPs with different properties would require the generation of many PATRAN neutral files.

Interference Fit Modeling Using Bar Elements

The prestressing of components (via interference fit or the tightening of a bolt for example) can be modeled by the use of bar elements. These elements are placed between the coincident nodes of the interacting parts. Their initial deformation, set by the NASTRAN DEFORM cards [5], is equal to the interference fit length. Since they are set to a very high relative rigidity, they "un-deform" almost completely, thus inducing stress in the interacting parts. The offset parameter of the CBAR NASTRAN cards [5] is used to give a short arbitrary finite length (0.002*inch* was chosen here) to the bars. The bar elements used are constrained along their length only thereby transmiting axial stresses only (see Appendix E).

COMPONENTS	DIRECTION	INTERFACE	FIG 3.12 ID
PIN - LUG	Z'	exact fit	1
LUG - BUSHING	R	INTERFERENCE FIT: 0.001in ON RADIUS	2
LUG - BUSHING	Z'	exact fit	3
PIN - BUSHING	Z'	(a)	4
PIN - BUSHING	R	CLEARANCE FIT: 0.0008in ON RADIUS	5
WASHER - BUSHING	Z'	exact fit	6
WASHER - BLADE	Z'	(b)	7
SLEEVE - BUSHING	Z'	(c)	8
SLEEVE - BLADE	R	(d)	9
PIN - SLEEVE	R	CLEARANCE FIT: 0.0008in ON RADIUS	10
WASHER - SLEEVE	R	(e)	11
SLEEVE - LINING	R	(f)	12

(a): the pin - lug interface prevents the pin and bushing from touching axially,

(b): it is assumed that every pair of points of the washer-blade interface have the same Z' displacement,

(c): the washer - bushing interface prevents the bushing and sleeve from touching axially,

(d): the sleeve and blade are considered as tightly joined,

(e): there is a radial gap between the washer and the sleeve large enough to prevent them from touching, and

(f): the sleeve and its lining are considered tightly joined.

Table 3.9: Component interactions

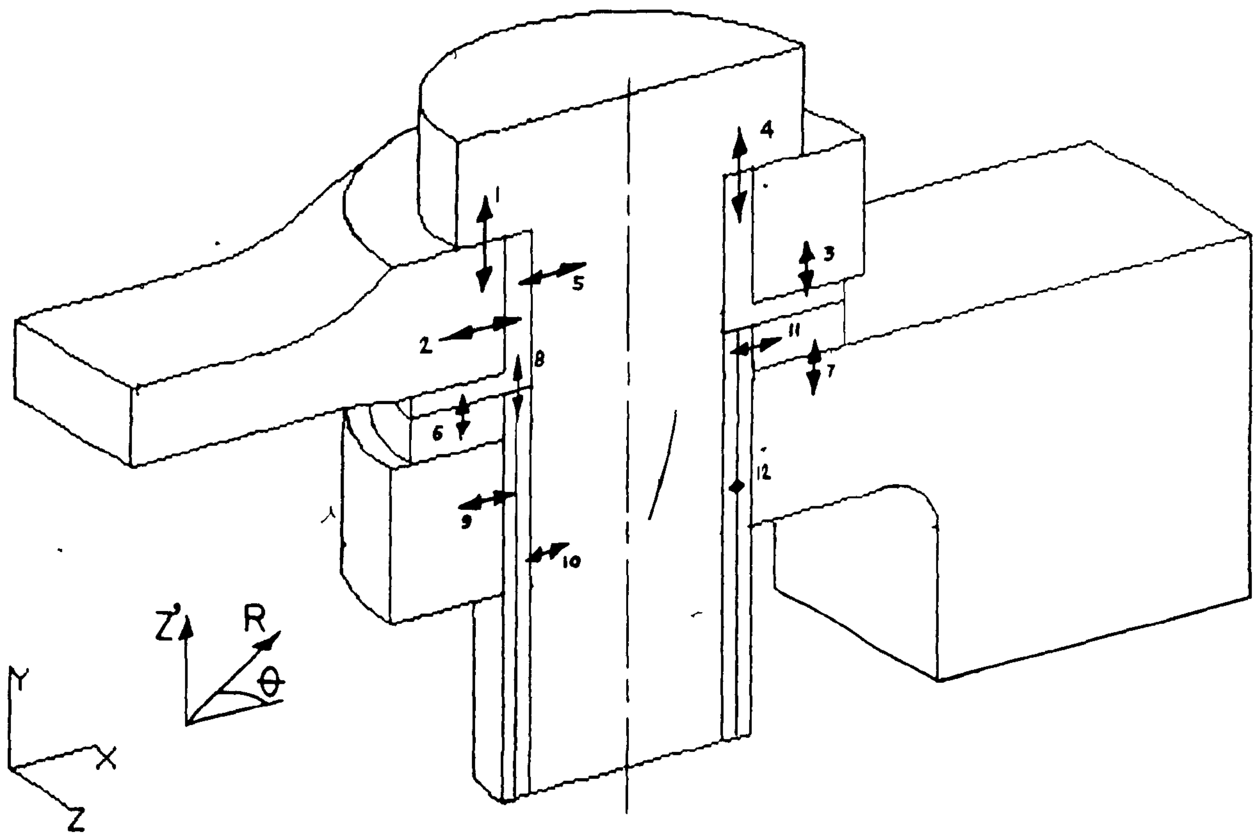


Figure 3.12: Interactions between the components

See Table 3.9.

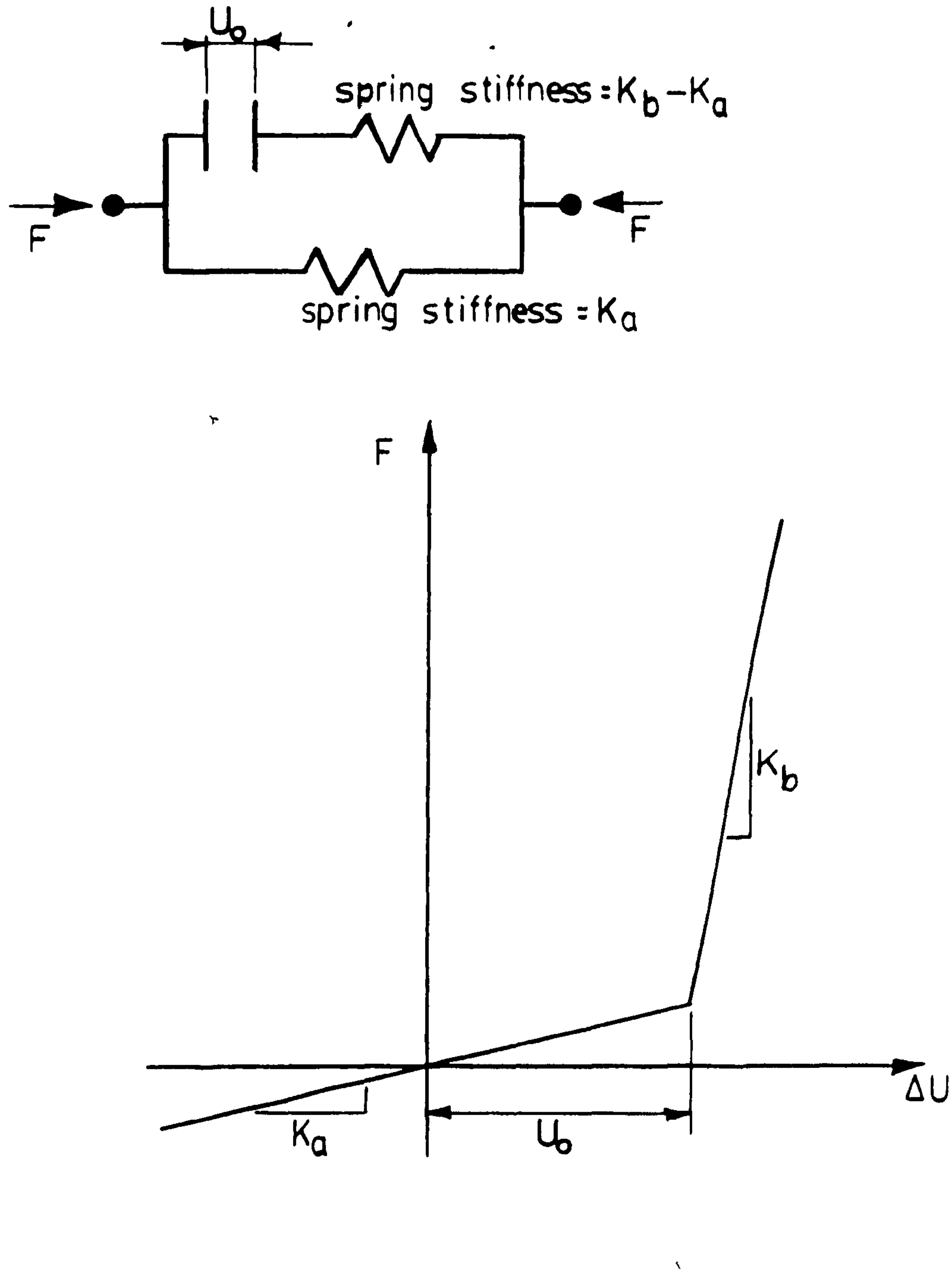


Figure 3.13: Gap Element as used for clearance fit modeling

ΔU is the distance between the interacting nodes. F is the force ($F > 0$ when in compression).

This method of computing the interference fit induced stress is linear and was used to compute the interference fit effects between the lug and the bushing when the joint is not loaded (see Section 3.3.1). Unfortunately, it could not be used in an NASTRAN non-linear analysis since NASTRAN does not accept any pre-deformation in non-linear analyses. Using an NASTRAN non-linear analysis was required when analyzing the complete problem since the clearance fit modeling is non-linear.

Interference Fit Modeling Using Gap Elements

For the reason explained in the above paragraph, the interference fit had to be modeled using gap elements in situations where non-linear analyses had to be used. Figure 3.14 shows how the interference fit can be modeled with gap elements.

Other Interfaces

The washers are assumed to always stay in contact with the blade (interface (7) in Table 3.9 and Figure 3.12). NASTRAN MPC [5] cards are used to define this type of interaction. The program IMPC (Appendix F) was used to create them.

The sleeve and its lining (interaction (12) in Table 3.9 and Figure 3.12) are tightly joined. So are the sleeve and the blade (interaction (9) in Table 3.9 and Figure 3.12). Since each pair of the interface nodes have the same R, θ, Z' displacements, they can be replaced by one node only. This task was performed by the PATRAN EQUIVALENCING instruction [6].

3.2.4 Boundary Constraints

Every component must be constrained in such a way as to avoid rigid body motion, i.e., displacement without any stress. Table 3.10 explains how every component of the lug/pin joint is constrained to avoid rigid body motion. The lug constraints were set within PATRAN. For the other components, the constraints were input directly into the NASTRAN data deck. When arbitrary nodes had to be determined to constrain the components, they were with the help of the IVOL program (Appendix H). NASTRAN MPC cards [5] were used to make the R, θ, or Z' displacement of two nodes equal.

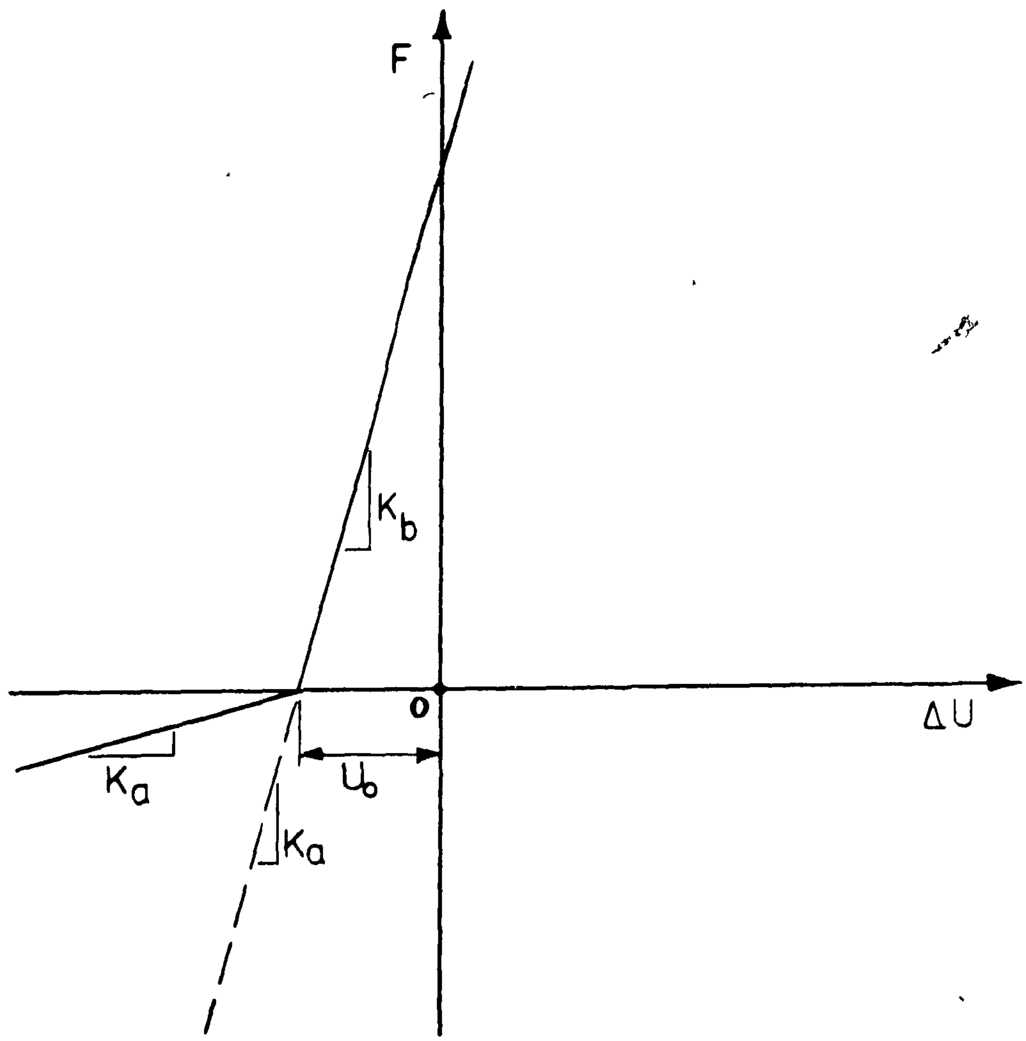


Figure 3.14: Gap Element as used for interference fit modeling

U_0 is the interference fit length.

If it is certain that, upon loading, the components in contact will not separate, the gap tensile rigidity K_a can be set equal to its compressive rigidity K_b as shown by the dashed line. The convergence will then be much faster. Unfortunately, it was not the case for the lug/pin joint under study.

Table 3.10: Boundary constraints

LUGS:	Their left ends (see (1) on Figure 3.15) are constrained in R, θ, Z'.
BUSHINGS:	The gap elements constrain the bushings in R and Z'. The θ-motion of one arbitrary node of each bushing is made equal to the θ-motion of the coincident node on the lugs (see (2) on Figure 3.15).
WASHERS:	The R-motion of one node on each washer is made equal to the R-motion of the coincident node on the blade. The θ-motion of 2 nodes on each washer are made equal to the θ-motion of the 2 coincident nodes on the blade (see (3) on Figure 3.15). The Z'-motion is constrained by the gap elements.
SLEEVE:	The sleeve is considered tightly joined to the blade. R, θ, Z' motions of the sleeve are consistent with the R, θ, Z' motions of the blade. The gap elements between the sleeve and the pin constrain the R-motion. The sleeve nodes located on the X axis are constrained in θ.
BLADE:	The blade nodes located on the X-axis are constrained in θ. The Z'-motion of the blade is constrained by gap elements. The R-motion is constrained because of the sleeve.
PIN:	The pin nodes located on the X-axis are constrained in θ. The R and Z' motions of the pin are constrained by the gap elements.

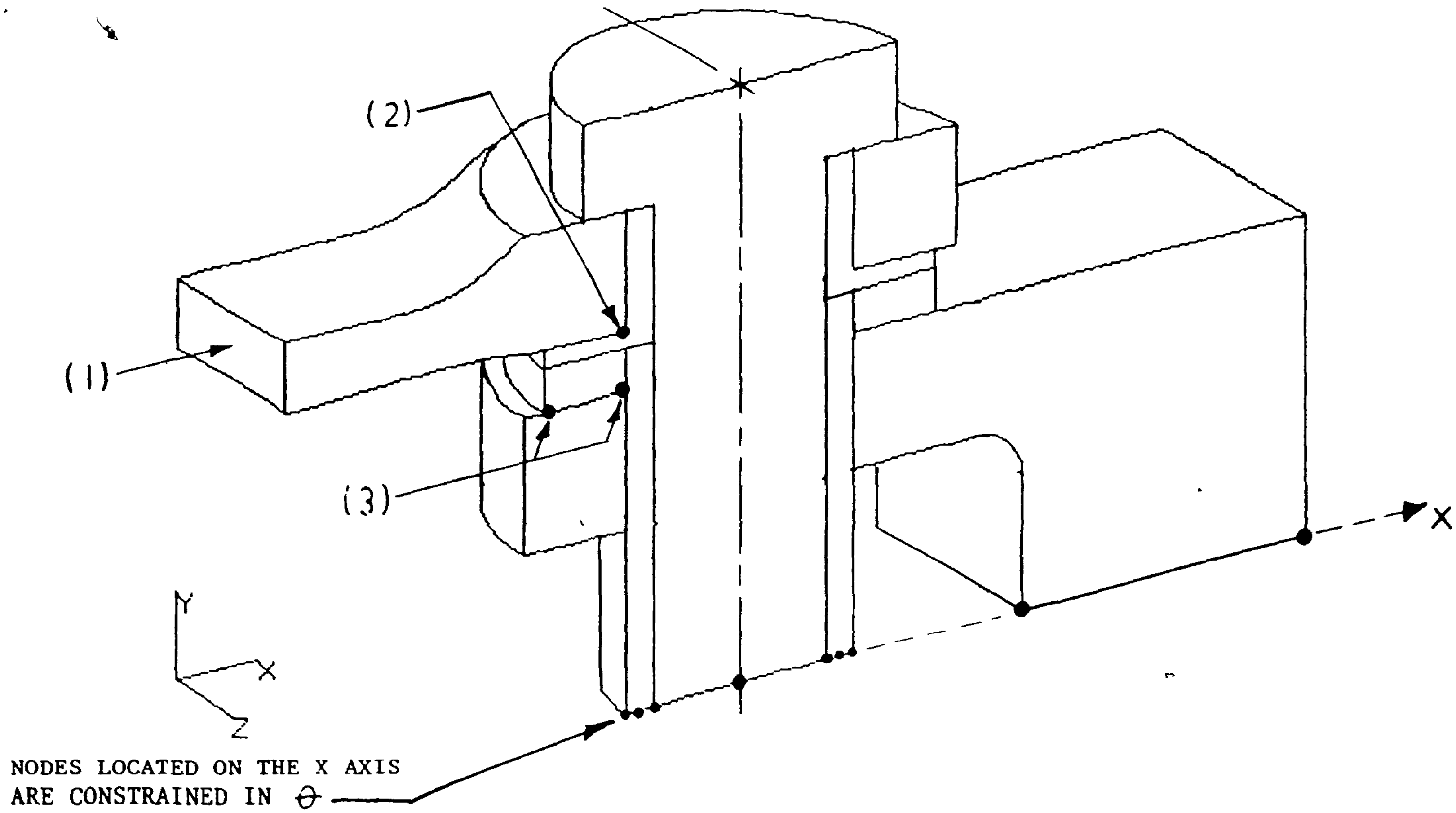


Figure 3.15: Constraints to avoid rigid body motion

Only one quarter of the model is shown here. (Refer to Table 3.10).

3.3 Lug & Bushing Interference Fit Analysis

3.3.1 Lug & Bushing Interference Fit Analysis using a F.E. Model

Table J.1 in Appendix J shows a commented version of the NASTRAN data deck used for this analysis.

The fine F.E. mesh defined in Section 3.2.2 (Figure 3.9) was used to evaluate the stresses induced on the lug by the bushing interference fit. Prestrained bar elements (see Section 3.2.3) which are defined with the help of the IBAR program (see Appendix E) were used.

A bar stiffness of $1.0E8lb/in$ was chosen. To get that value, the bar length was set to $0.002in$, the cross section to $1.00in^2$, and the material Young modulus to $20.0E4psi$. I.e.:

$$
\begin{aligned}
\text{Bar stiffness} &= \frac{\text{Young's modulus} \times \text{Bar cross section}}{\text{Bar length}} \\
&= \frac{20.0E4psi \times 1.00in^2}{0.002in} \\
&= 1.0E8lb/in
\end{aligned}
$$

Figures 3.16 to 3.19 show the stress field induced by the lug/bushing interference fit. The radial interference fit is 0.001 *inch* (Figure 2.4).

The maximum final bar deformation was found to be −0.146% of the interference fit length.

The inner face of the bushing is not constrained and,therefore the radial stress there should be zero. On Figure 3.19, it is around −1000 *psi* which is approximately 8% of the radial stress range (from −8000 to 4000*psi*).

3.3.2 Computation of Stresses using Lamé's Equations

In this section, the results obtained from the F.E. model in the previous section are compared to some theoretical results. Upper and lower bounds for the radial stress at the lug/bushing interface will be computed using the well known Lamé's eqns(3.1 & 3.2) which correspond respectively to Figures 3.20 and 3.21.

$$P = \frac{d}{\frac{D}{E_r}\left(\frac{(C^2+D^2)}{(C^2-D^2)} + \nu_r\right) + \frac{B}{E_b}\left(\frac{(B^2+A^2)}{(B^2-A^2)} - \nu_b\right)} \tag{3.1}$$

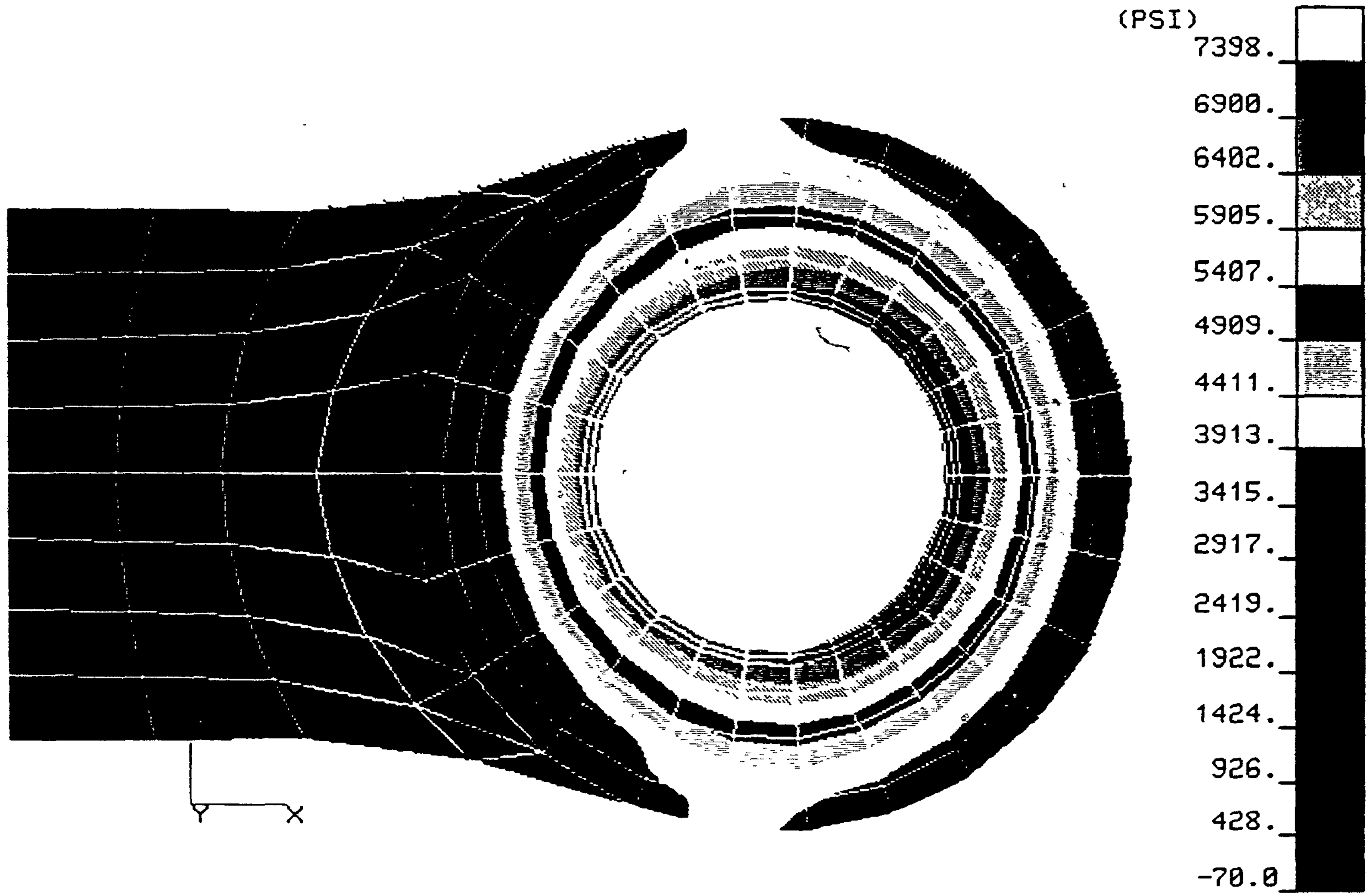


Figure 3.16: Tangential stress induced in the lug by the interference fit bushing

The lug is seen from below and the bushing is not shown.

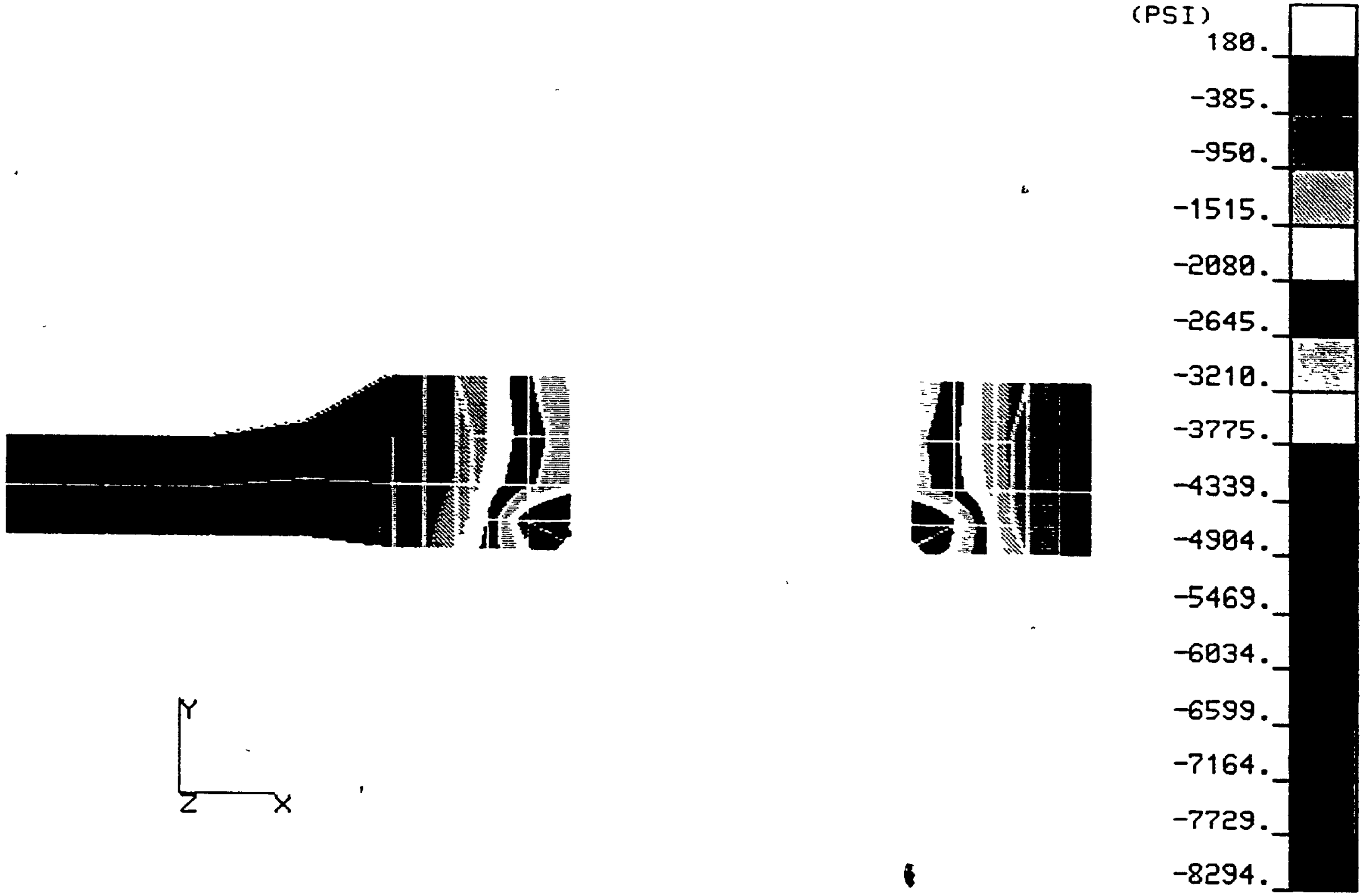


Figure 3.17: Radial stress induced in the lug by the interference fit bushing (section view)
The bushing is not shown.

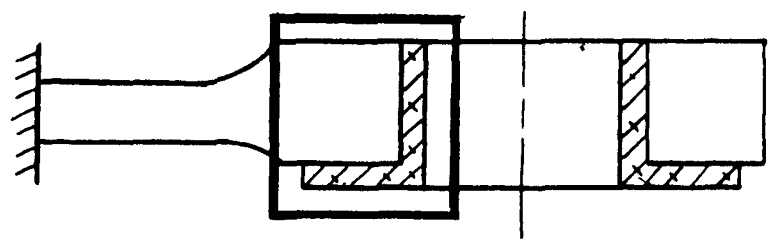


Figure 3.18: Enlarged view on next figure

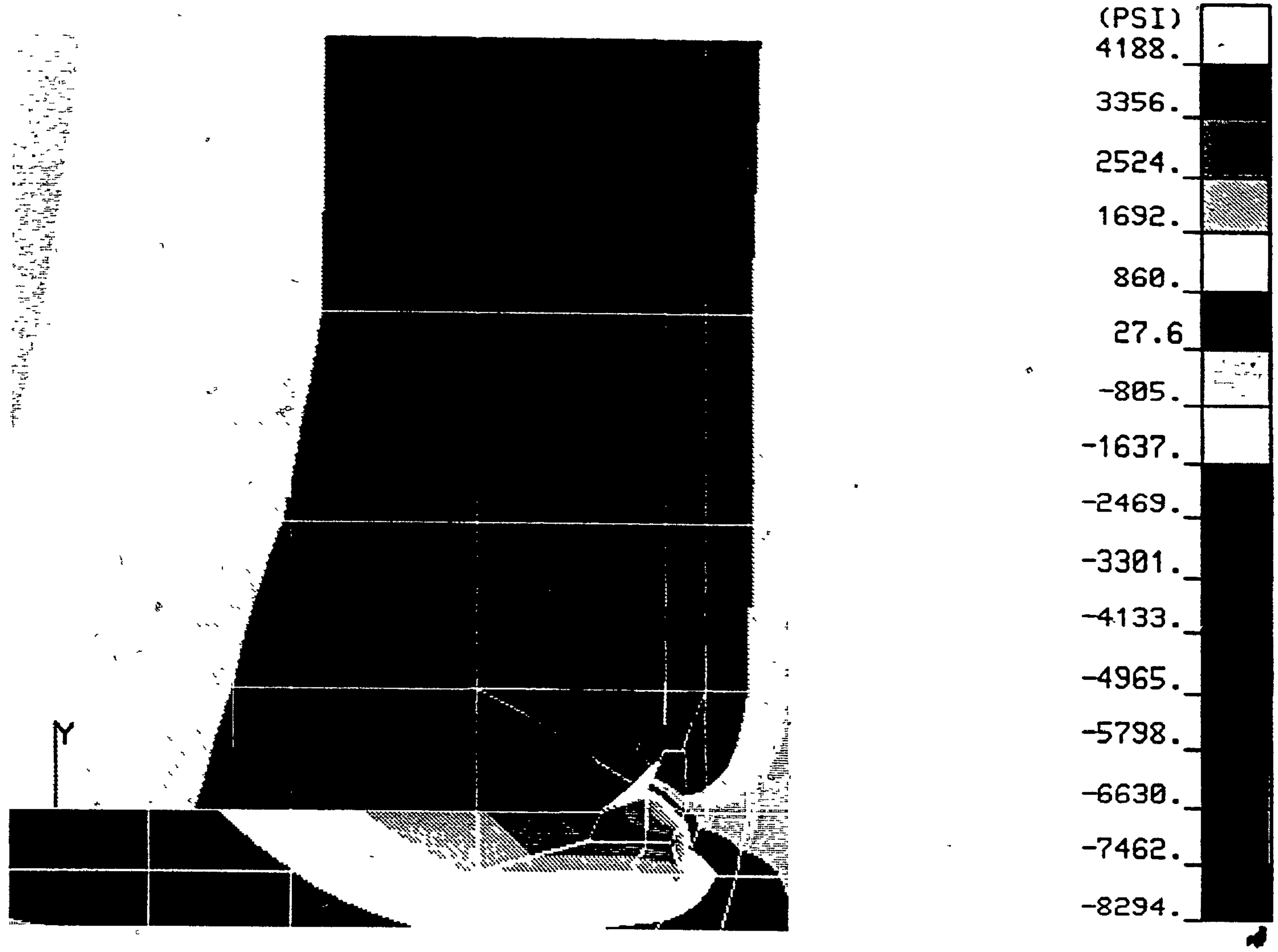


Figure 3.19: Radial stress at the lug/bushing interface

The radial stresses on both sides of the bushing/lug interface should match. They approximatively do except near the lug chamfer.

$$P = \frac{E_r d}{D(\frac{(C^2+D^2)}{(C^2-D^2)} + \nu_r)} \tag{3.2}$$

where:

- P is the radial stress at the bush/ring interface,
- A is the bushing inner radius,
- B is the bushing outer radius,
- C is the ring outer radius,
- D is the ring inner radius,
- d is the radial interference,
- E_r is the ring Young's modulus,
- E_b is the bushing Young's modulus,
- ν_r is the ring Poisson's ratio and
- ν_b is the bushing Poisson's ratio.

Here, from Table 2.1:

- $E_r = 10.0E6psi$,
- $E_b = 28.5E6psi$,
- $\nu_r = 0.33$ and
- $\nu_b = 0.272$.

Lower bound. A lower bound for the radial stress can be evaluated using eqn(3.1) (Figure 3.20) with the following geometrical values:

- $d = 0.001in$,
- $A = 0.5in$,
- $B = D = 0.6in$ and
- $C = 1.235in$.

In this case, the radial stress is $4388psi$.

Upper bound. An upper bound can be computed by considering the case illustrated in Figure 3.21. The deformation $d = 0.001in$ is completely taken by the lug. The outer radius has been chosen equal to the distance between the pin axis and the lug end, i.e., $C = 2.6in$. Knowing that inner radius is $D = 0.6in$, the radial pressure computed using eqn (3.2) is found to be $11554psi$.

Comparison & Discussion. The radial pressure at the lug/bushing interface should be between $4388psi$ and $11554psi$. From Figure 3.19, it is between ≈ 3000 and $\approx 8000psi$.

Since the minimum value obtained by the F.E. analysis is lower than the lower bound, it means the the F.E. mesh is not fine enough and, therefore, underestimates the stress concentration factor around the lug hole. This comment will also be valid for the detailed stress analysis of the loaded lug (Section 3.5).

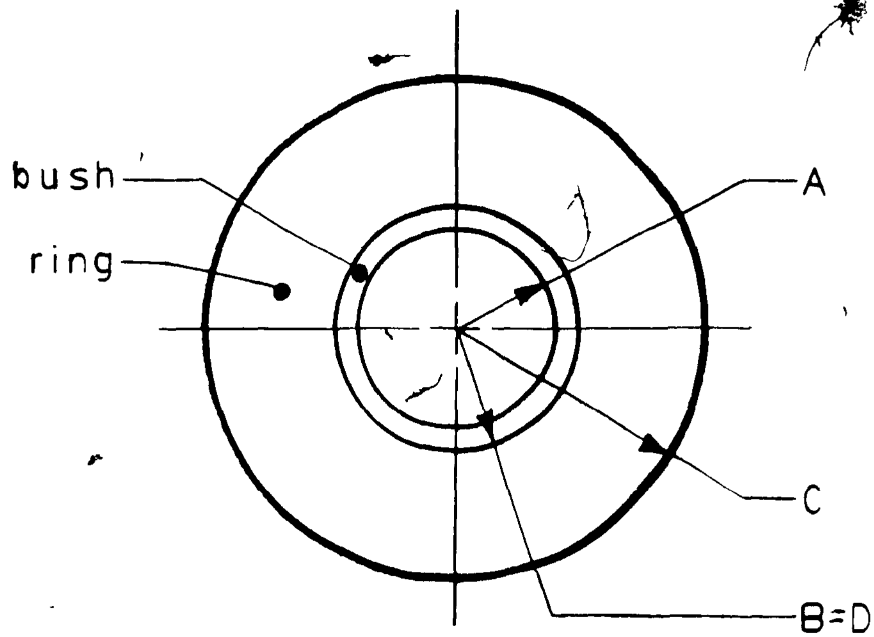


Figure 3.20: Computation of the radial stress lower bound

See eqn(3.1).

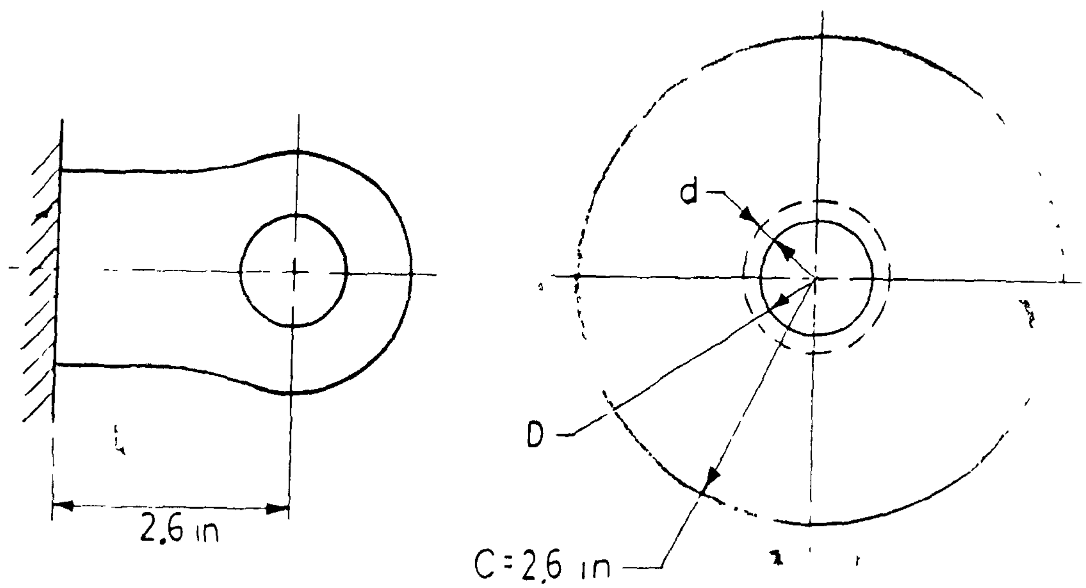


Figure 3.21: Computation of the radial stress upper bound

See eqn(3.2).

3.4 Lug/Pin Joint Global Stress Analysis Results

This section presents the results of the "first step" of the stress analysis (see Section 3.2.2). Table J.2 in Appendix J shows a commented version of the NASTRAN data deck used for this analysis. The results obtained from this analysis are shown in the following figures:

- Figures 3.22 and 3.23: section view of lug/pin joint subjected to *LL1*.
- Figures 3.24 and 3.25: section view of lug/pin joint subjected to *HL1*
- Figure 3.26: top lug, seen from below, when submitted to *HL1*

The most compressed gap element was gap #19025. It connected sleeve node #503 to pin node #1479. It was compressed to 2875lb when the lug/pin joint was submitted to *HL1*. Since the gap stiffness when closed was set to $1.0E7\ lb/in$, it means that it had "over-deformed" $0.29E-3\ in$ (Figure 3.27). The radial displacement of nodes 503 and 1479 were $-11.586E-3$ and $-10.499E-3\ inch$ respectively. So the "over- deformation" was 3% of the deformation.

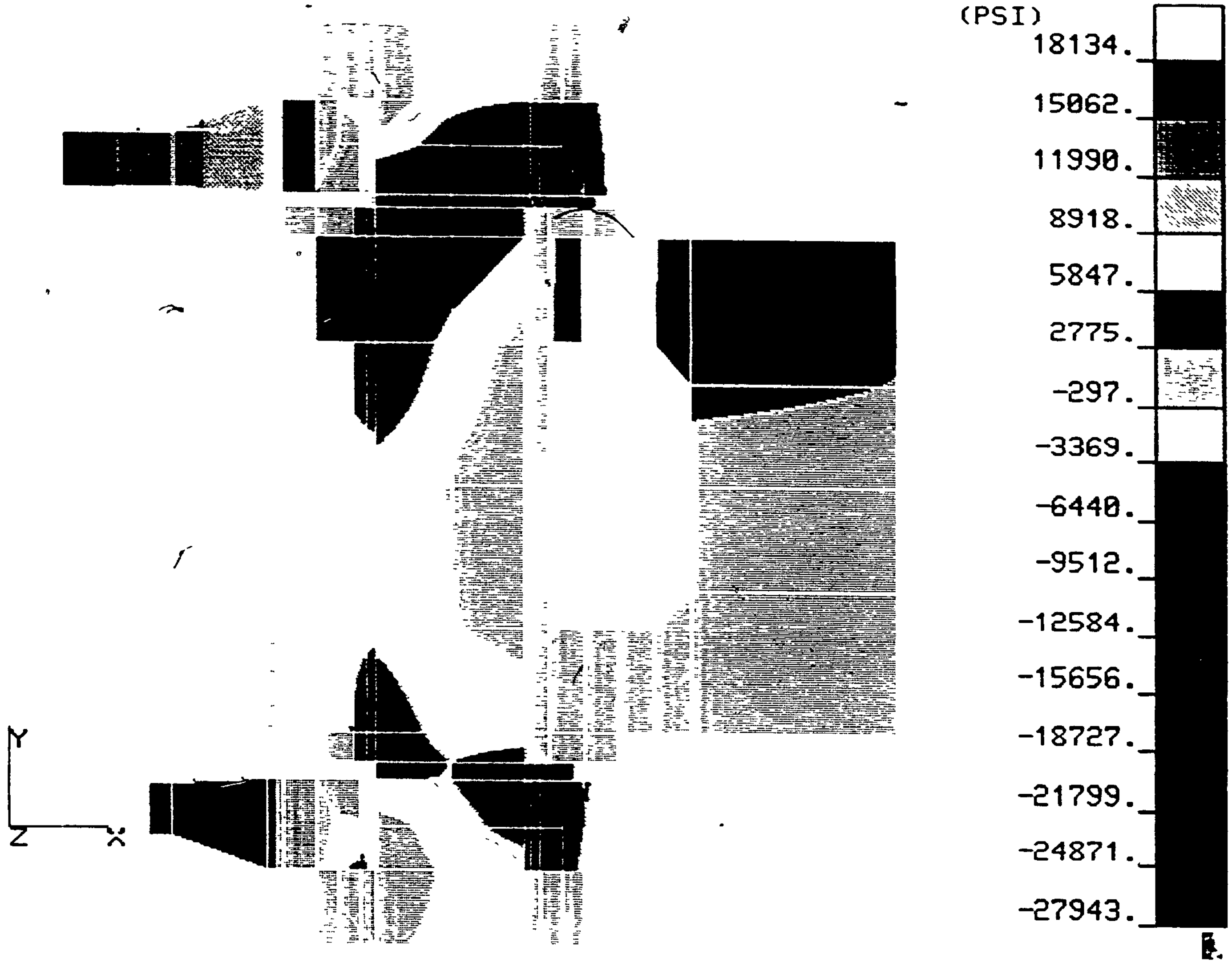


Figure 3.22: Radial stress, section view of lug/pin joint when subjected to *LL1*

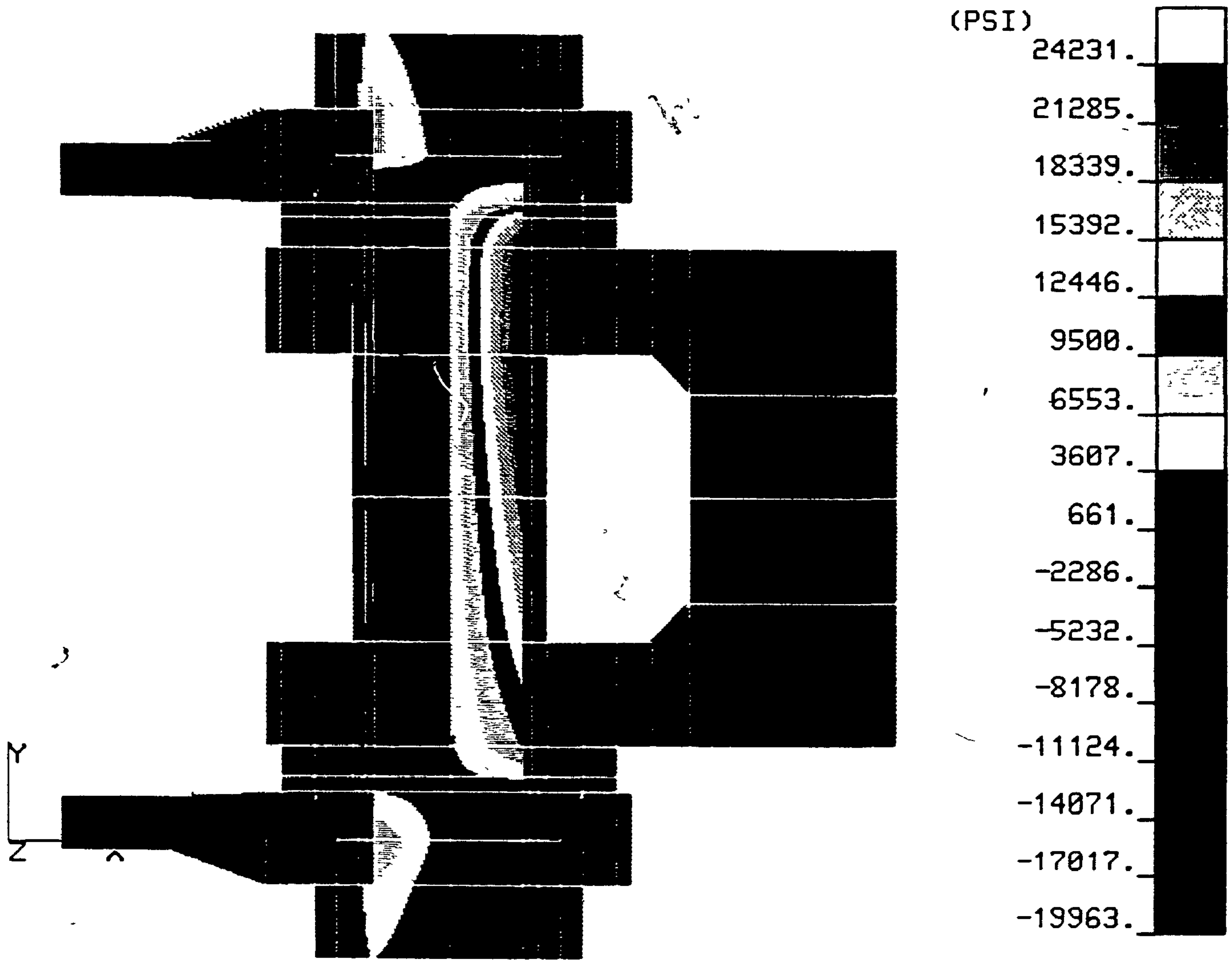


Figure 3.23: Axial stress, section view of lug/pin joint when subjected to *LL1*

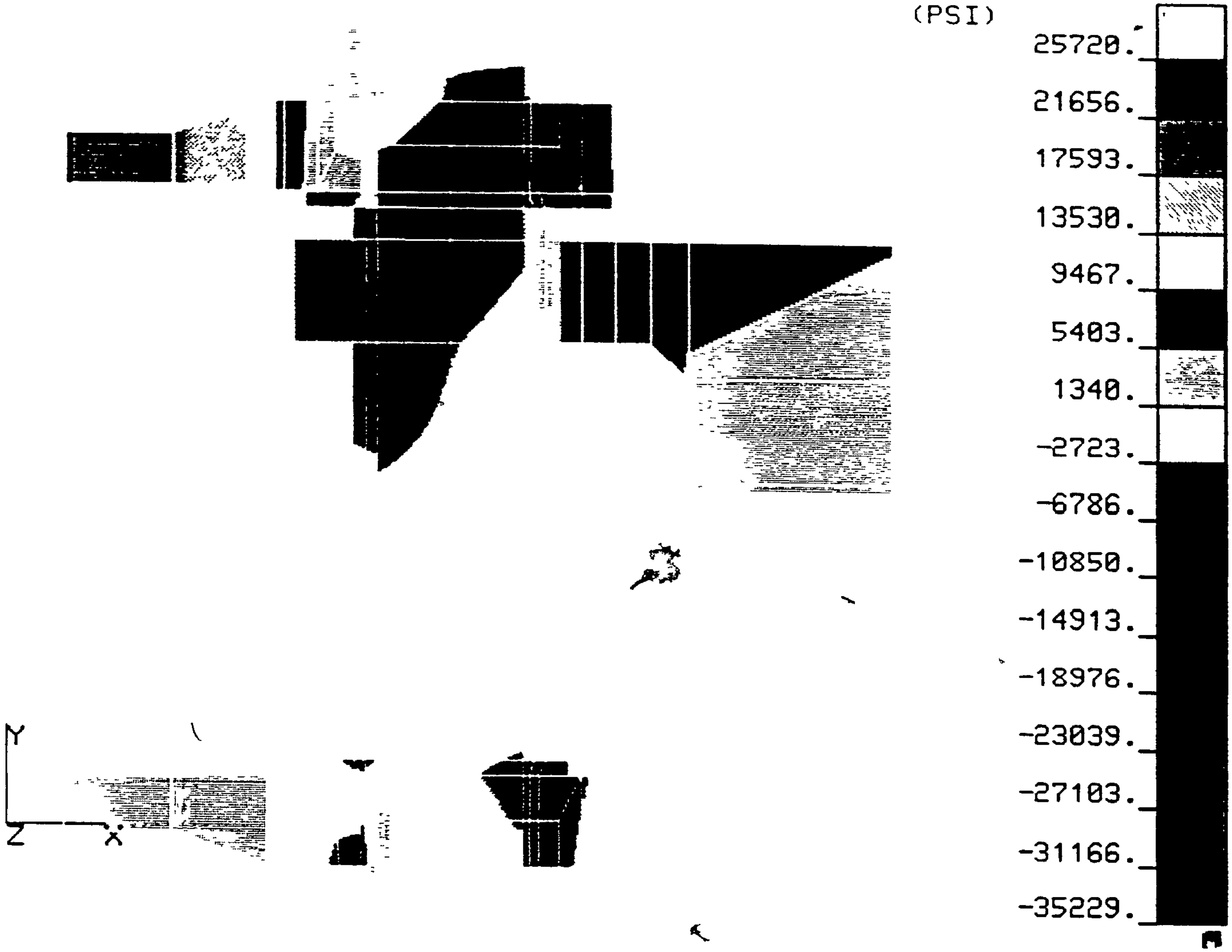


Figure 3.24: Radial stress, section view of lug/pin joint when subjected to *HL1*

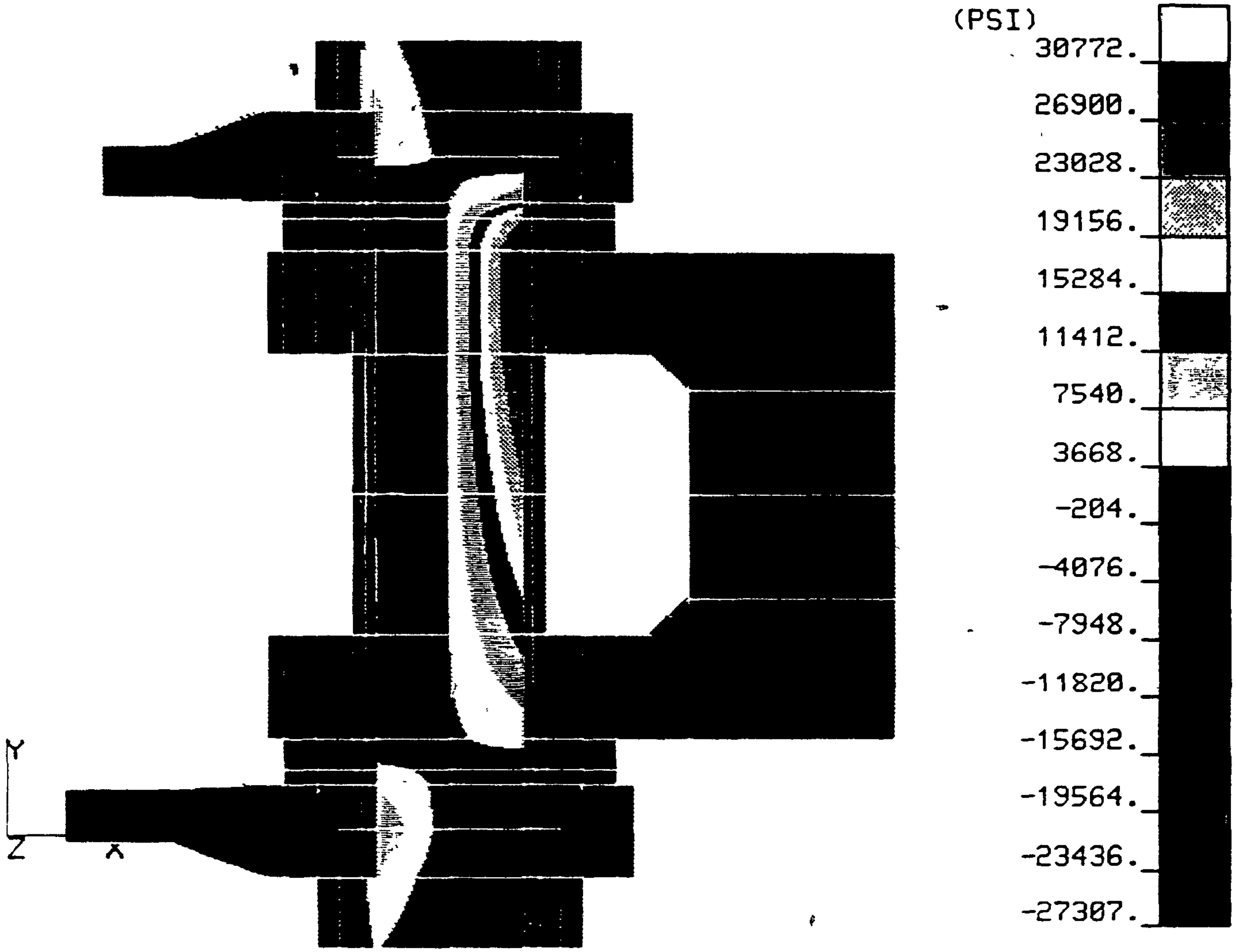


Figure 3.25: Axial stress, section view of lug/pin joint when subjected to *HL1*

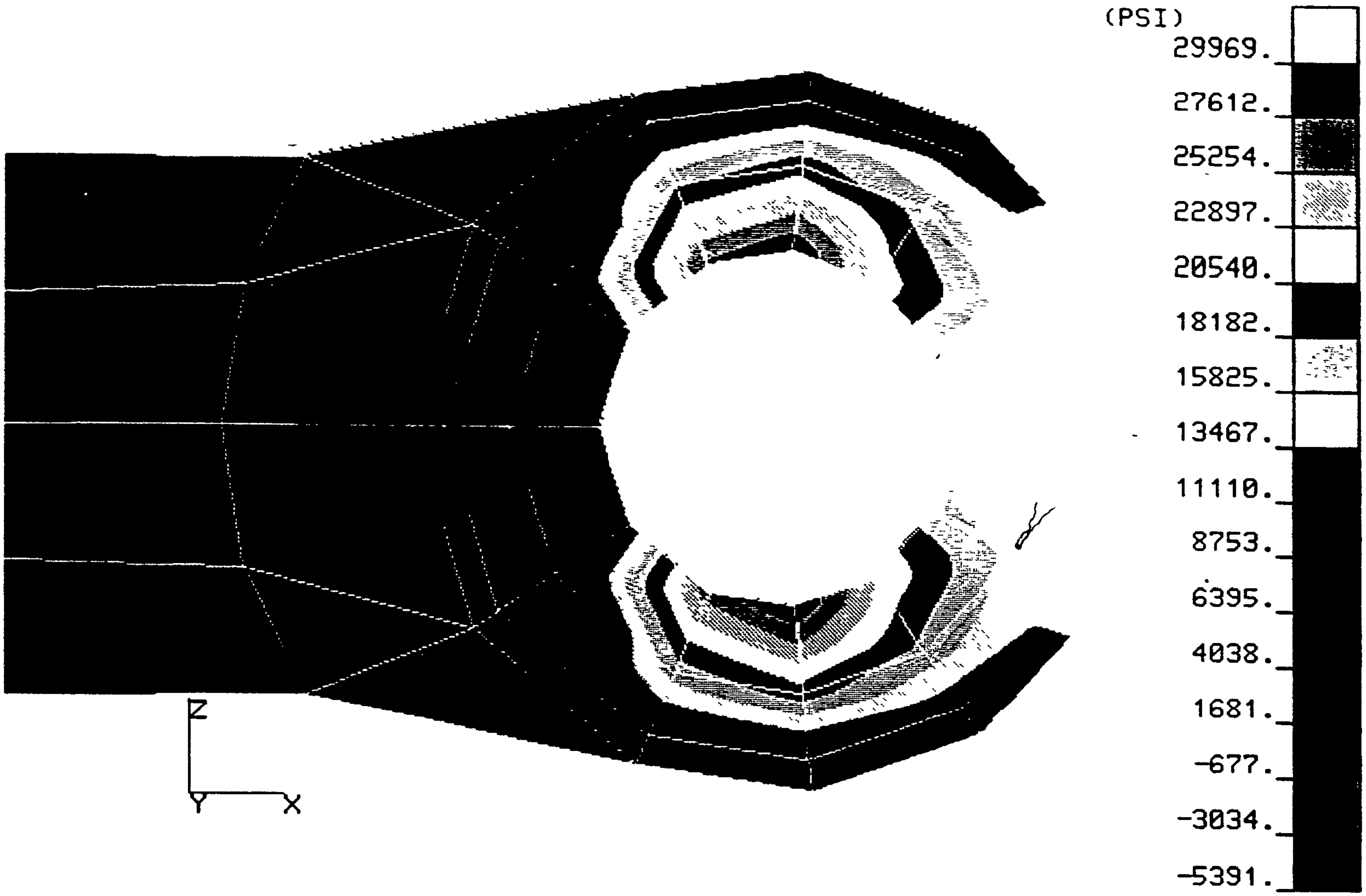


Figure 3.26: Radial stress in the top lug (seen from below), when submitted to *HL1*

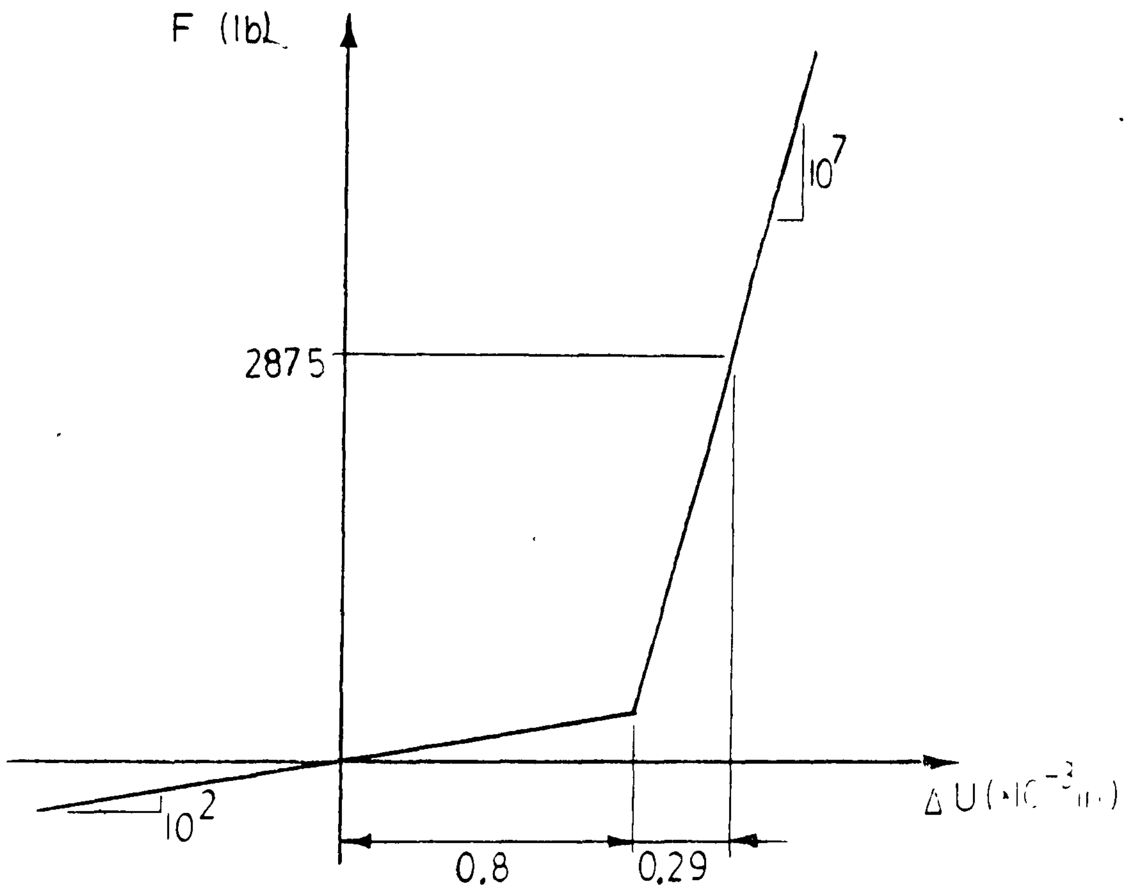


Figure 3.27: Gap Element #19025 when the lug/pin joint was submitted to *HL1*

It deformed $(0.8E-3\ +\ 0.29E-3)$ *in*.

3.5 Lug Detailed Analysis Results

Recalling that the purpose of the first step was mainly to find the boundary displacements at the lug & bushing (Figure 3.6), the boundary displacement fields applied to the fine model were computed using eqn(3.3) in which the ξ_1 and ξ_2 parameters indicate the nodes positions (Figure 3.28).

$$\begin{aligned} D(\xi_1, \xi_2) \;=\; & S_{00} + S_{01}\xi_2 + S_{02}\xi_2^2 + S_{03}\xi_2^3 + \\ & S_{10}\xi_1 + S_{11}\xi_1\xi_2 + S_{12}\xi_1\xi_2^2 + S_{13}\xi_1\xi_2^3 + \\ & S_{20}\xi_1^2 + S_{21}\xi_1^2\xi_2 + S_{22}\xi_1^2\xi_2^2 + S_{23}\xi_1^2\xi_2^3 + \\ & S_{30}\xi_1^3 + S_{31}\xi_1^3\xi_2 + S_{32}\xi_1^3\xi_2^2 + S_{33}\xi_1^3\xi_2^3. \end{aligned} \tag{3.3}$$

In eqn(3.3) the S_{ij} parameters were determined from the boundary displacement fields obtained from the coarse F.E. model analysis. Once the S_{ij} parameters were known, the boundary displacement fields applied on the fine F.E. model were computed.

The programs DF9, DF61, DF62 explained in Appendix I were used to compute the S_{ij} parameters and create the PATRAN instructions that defined the imposed displacements. Those programs were written only to make the task easier. For a different F.E. meshing, it may not be possible to use them. This is a weak part of the analysis since it is tedious and error prone to use output from a model as input to another model.

As stated in Chapter 2, the displacements fields at the top lug & bushing under the *HL2* and *LL2* loading conditions can be deduced from the displacements fields at the bottom lug & bushing under the *HL1* and *LL1* loading conditions. Figures 3.29 to 3.36 show the boundary displacement fields imposed on the top lug & bushing when the *HL2* loading conditions are applied. The *LL1*, *LL2*, *HL1* loading conditions imposed similar boundary displacement fields. The nodes located on hyperpatch sides whose gap elements have opened are set free.

Table J.3 in Appendix J shows a commented version of the NASTRAN data deck that was used for this analysis. Figures 3.37 to 3.40 show the top lug tangential stress field as seen from below when it is submitted to the 4 loading conditions (*LL*1, *LL*2, *HL*1, *HL*2). Figure 3.41 shows the radial stress field of the lug when it is submitted to the loading *HL1*.

The most compressed gap element was gap #12064. It connected bushing node #1301 to lug node #513. It was compressed to 518.49*lb* when the lug/pin joint was

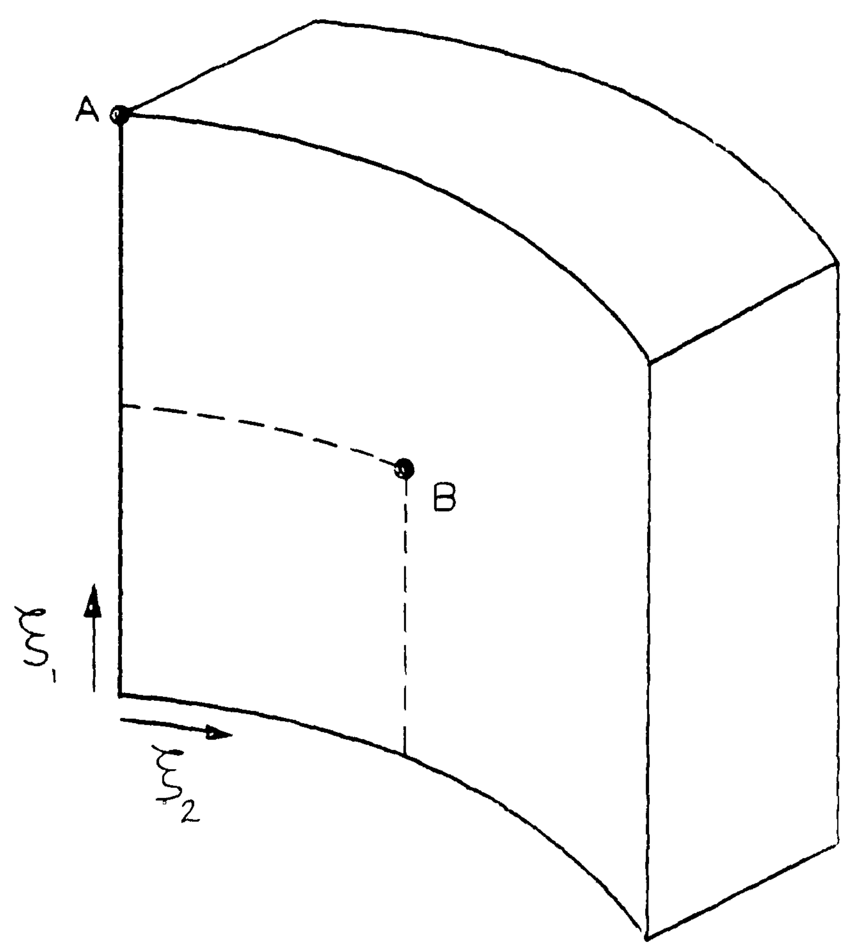


Figure 3.28: ξ_1 and ξ_2 parameters

The node position on each hyperpatch side can be defined using ξ_1 and ξ_2 parameters where $0 \leq \xi_1 \leq 1$ and $0 \leq \xi_2 \leq 1$.

For example: node A is located at $\xi_1 = 1.0$ and $\xi_2 = 0.0$, and
node B is located at $\xi_1 = 0.5$ and $\xi_2 = 0.5$.

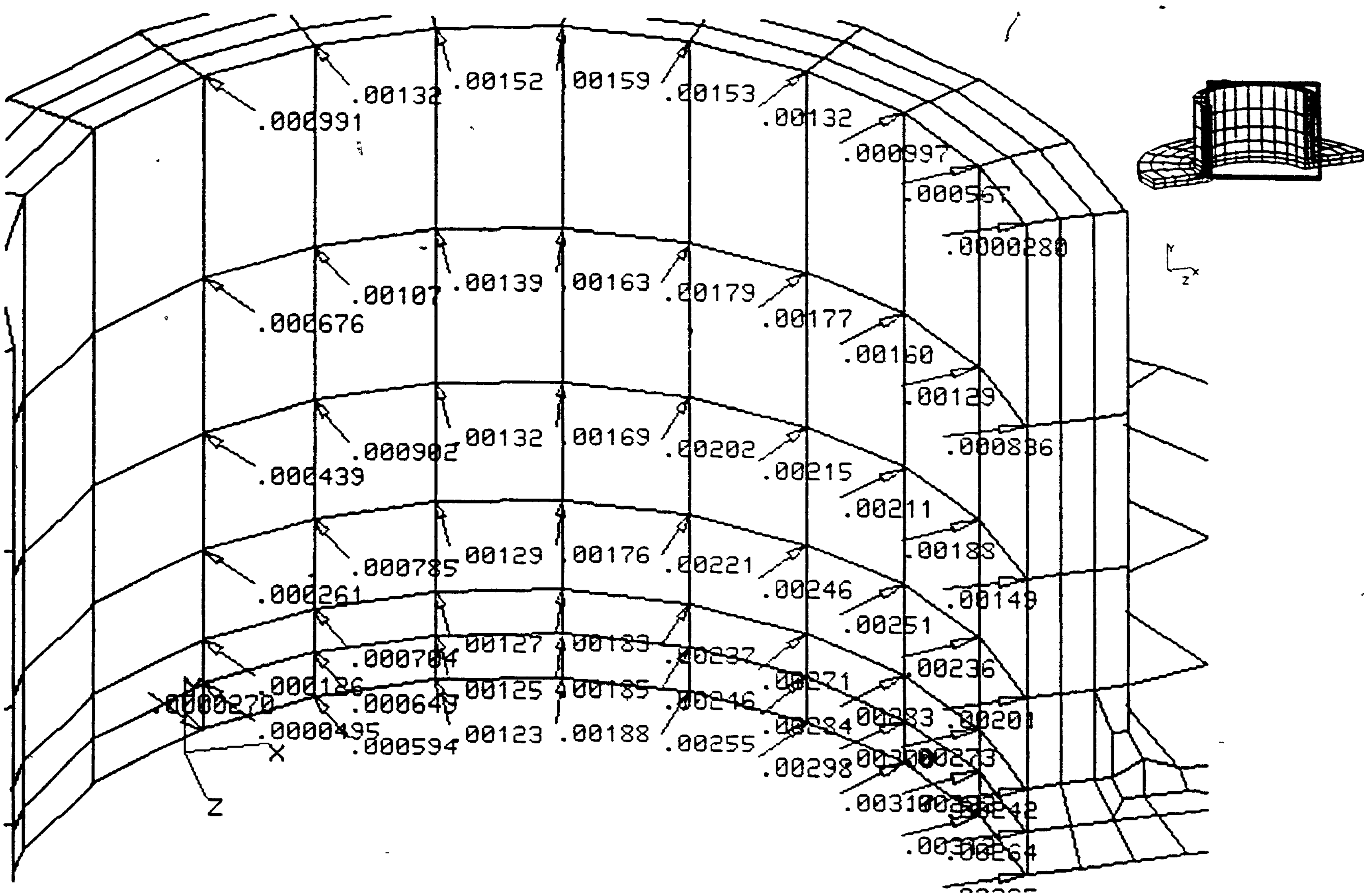


Figure 3.29: Radial displacement imposed by the pin on the bushing when the lug/pin joint is submitted to *HL2*

Values are in inches.

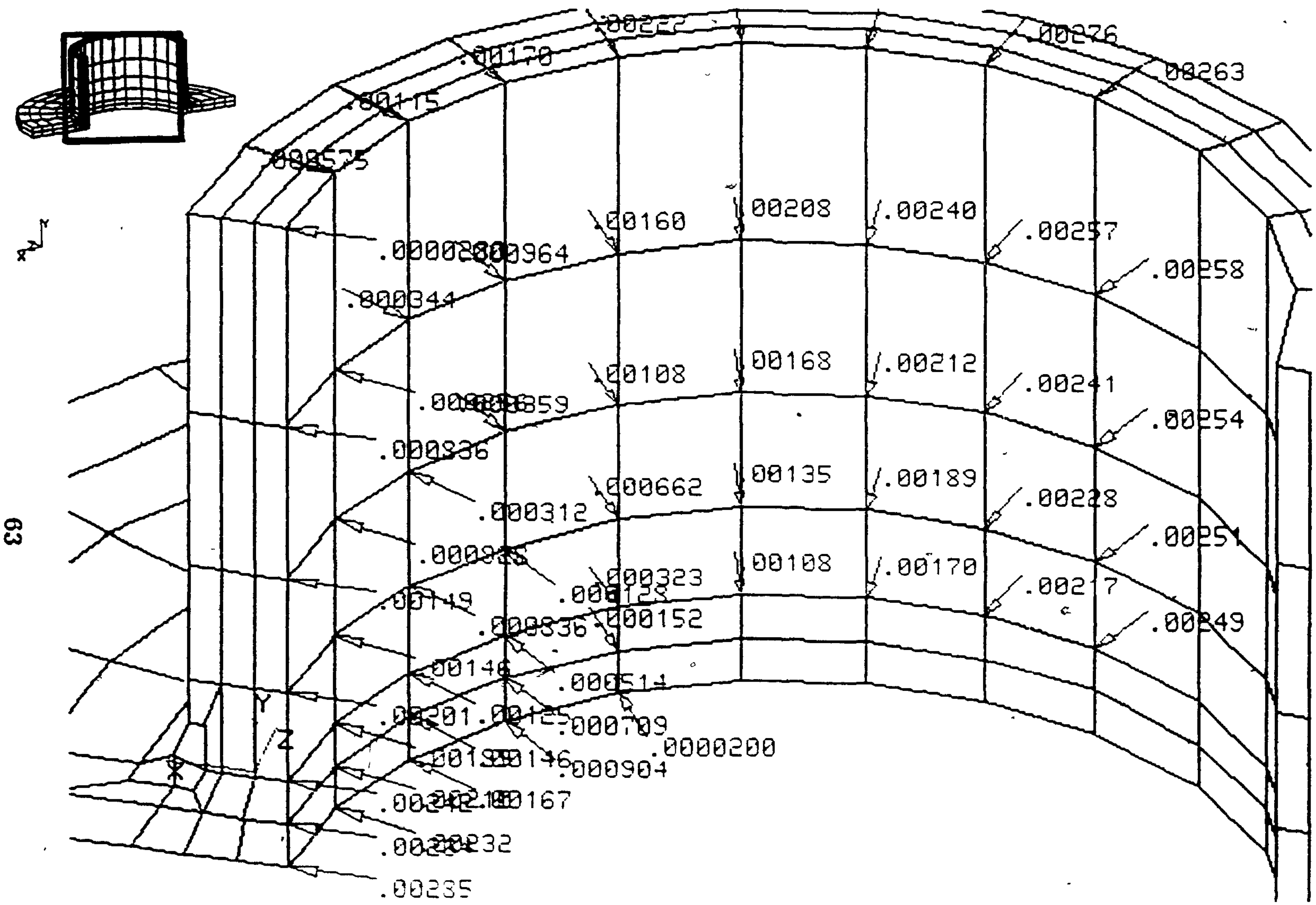


Figure 3.30: Radial displacement imposed by the pin on the bushing when the lug/pin joint is submitted to *HL2*

Values are in inches. This is the opposite half of Figure 3.29.

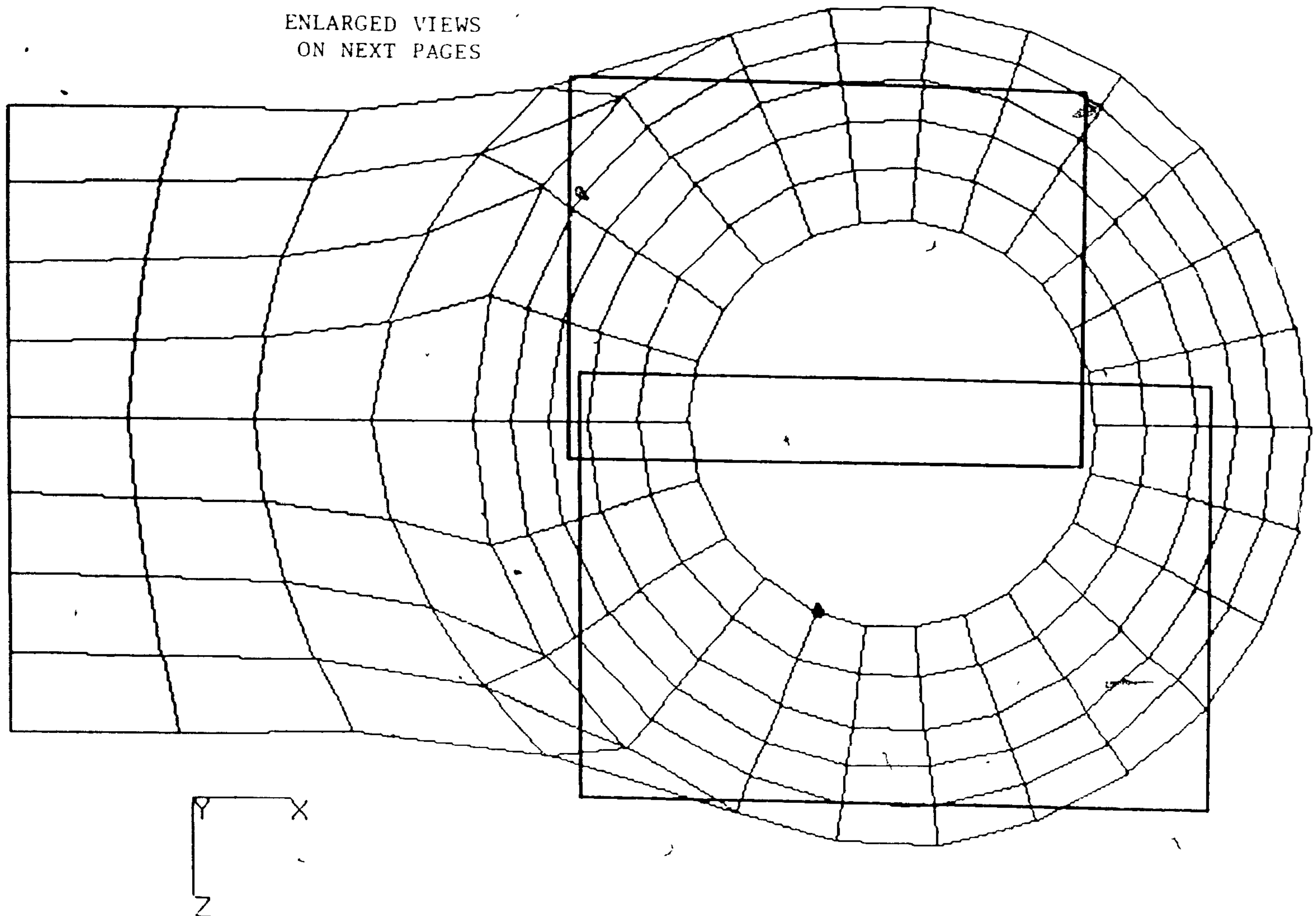


Figure 3.31 Top lug view shown in the next two figures

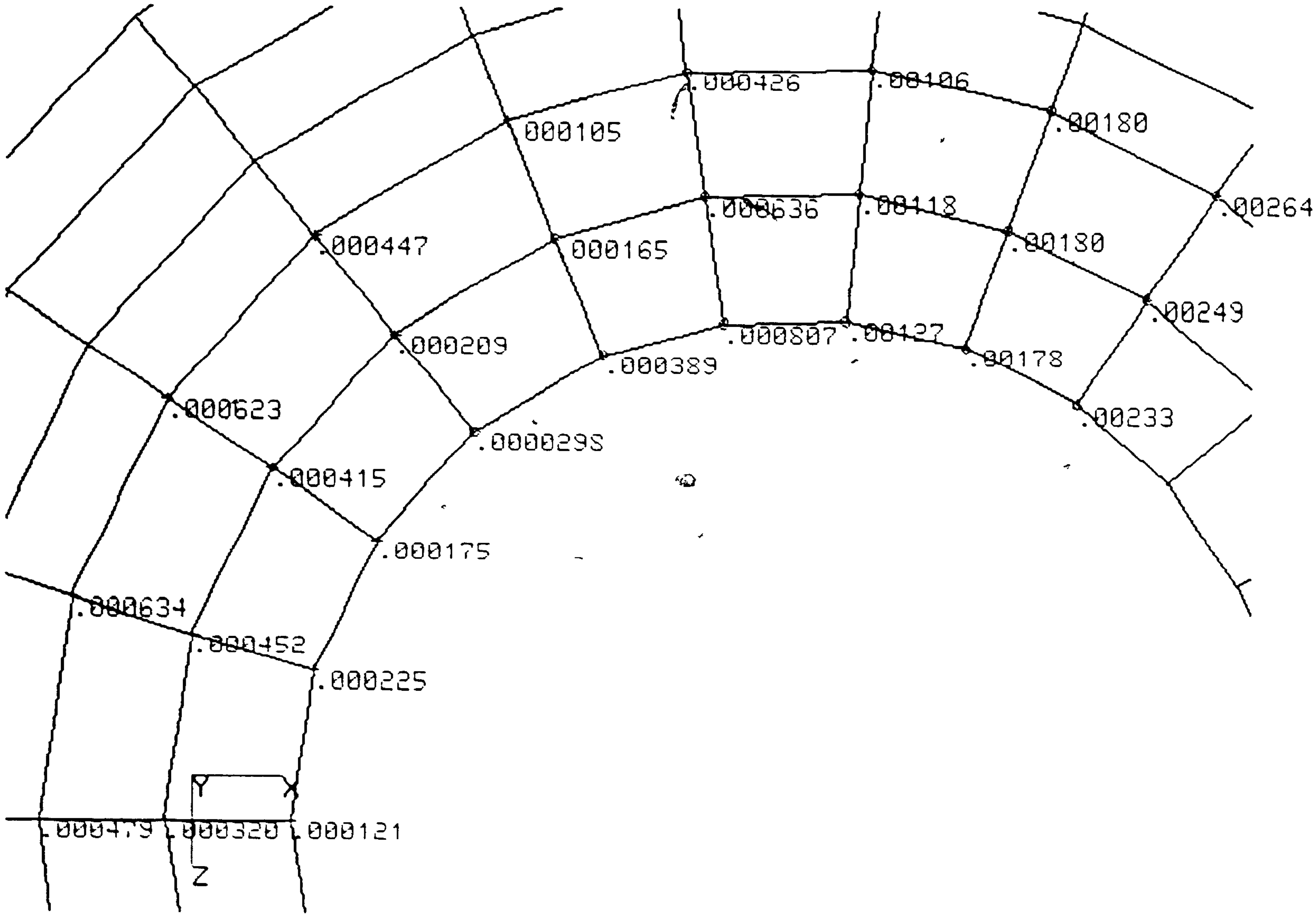


Figure 3.32: Axial (Z') displacement imposed by the pin head to the top lug when the lug/pin joint is submitted to *HL2*

Values are in inches.

(Displacement vectors indicated by "+" are entering in the figure Those indicated by "o" are coming out of the figure).

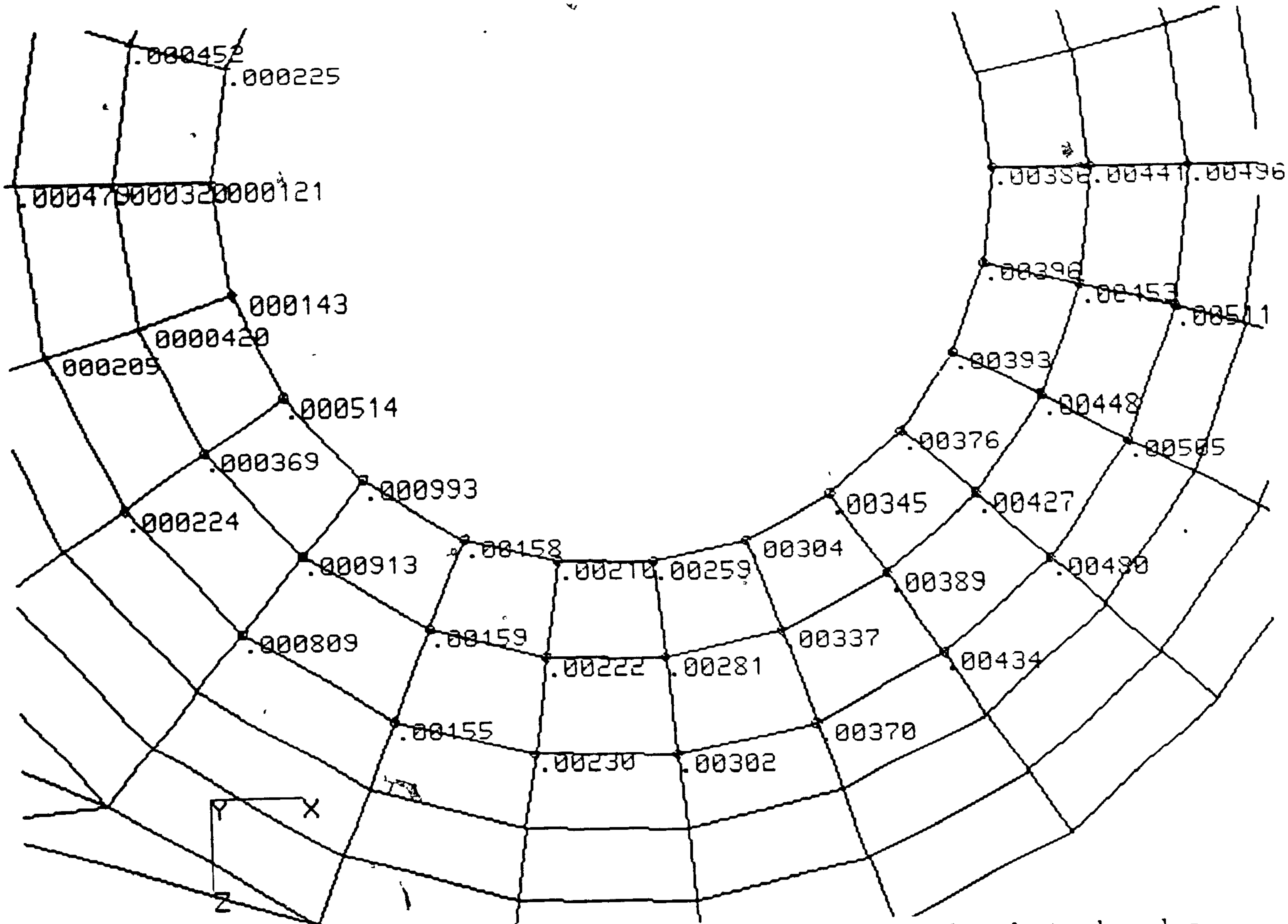


Figure 3.33: Axial (Z') displacement imposed by the pin head to the top lug when the lug/pin joint is submitted to *HL2*, (cont'd)

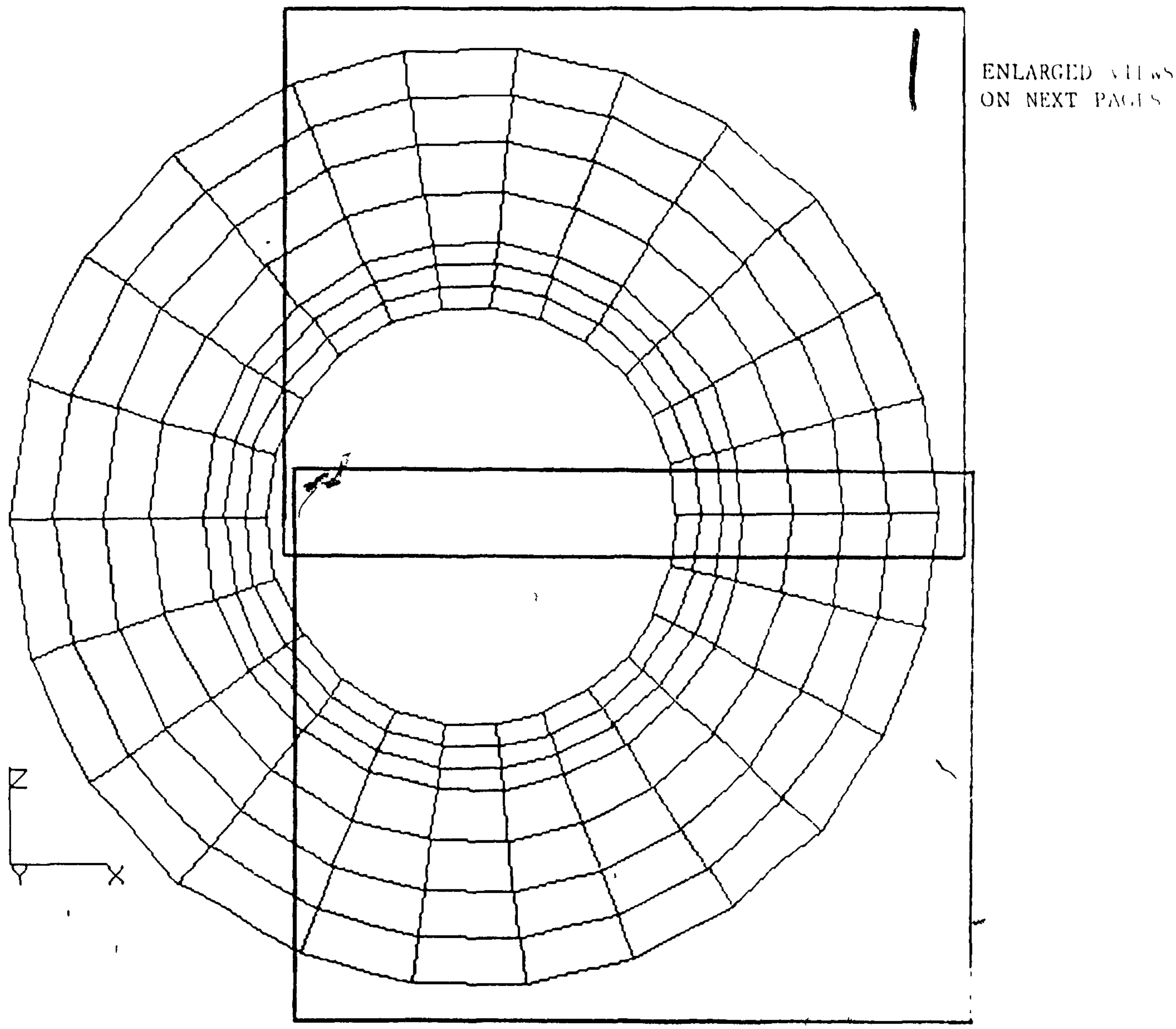


Figure 3.34: Top bushing view shown in the next two figures

It is seen from below.

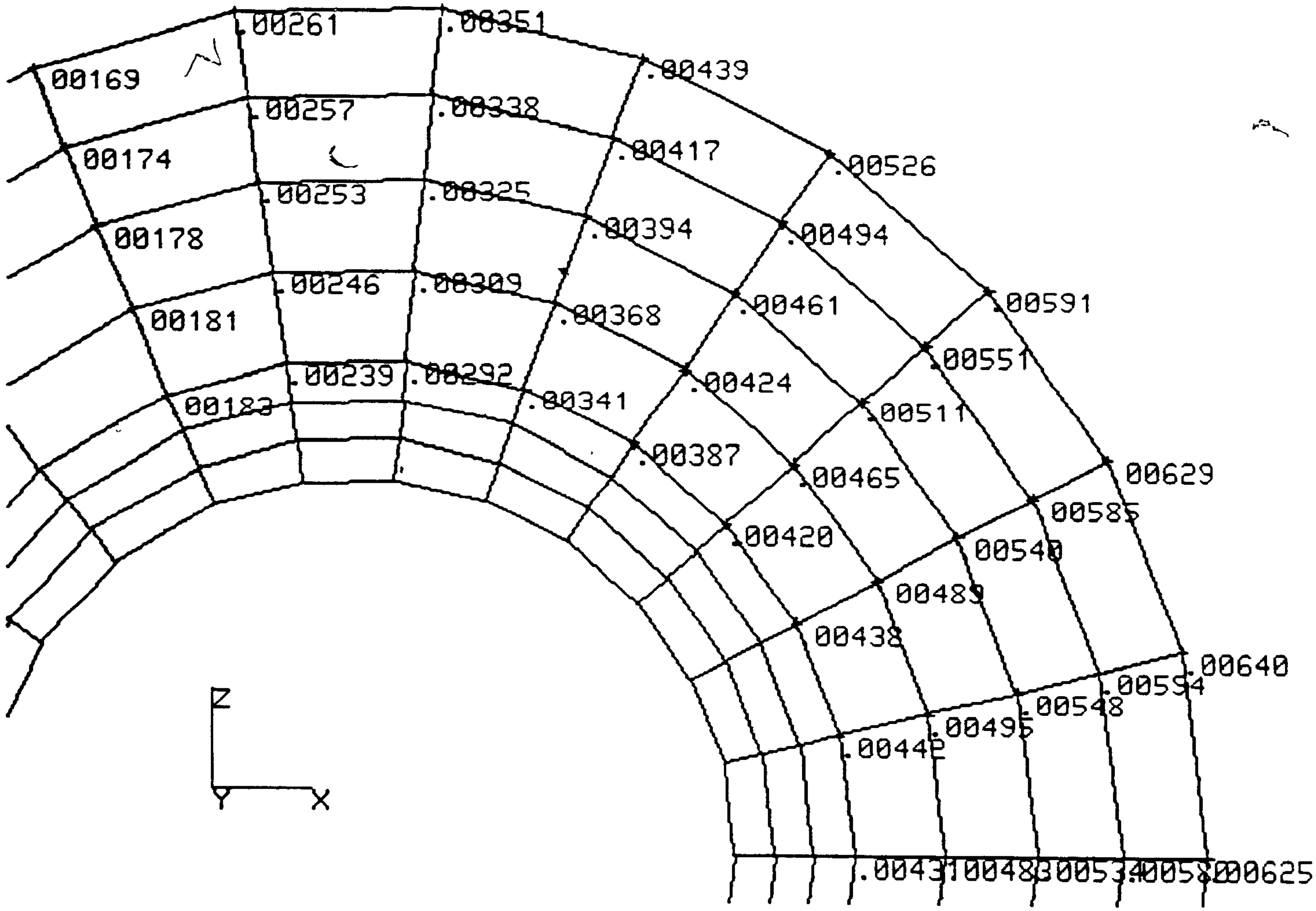


Figure 3.35: Axial (Z') displacement imposed by the washer to the top bushing when the lug/pin joint is submitted to *HL2*

Values are in inches.

Displacement vectors indicated by "+" are entering in the figure. Those indicated by "o" are coming out of the figure.

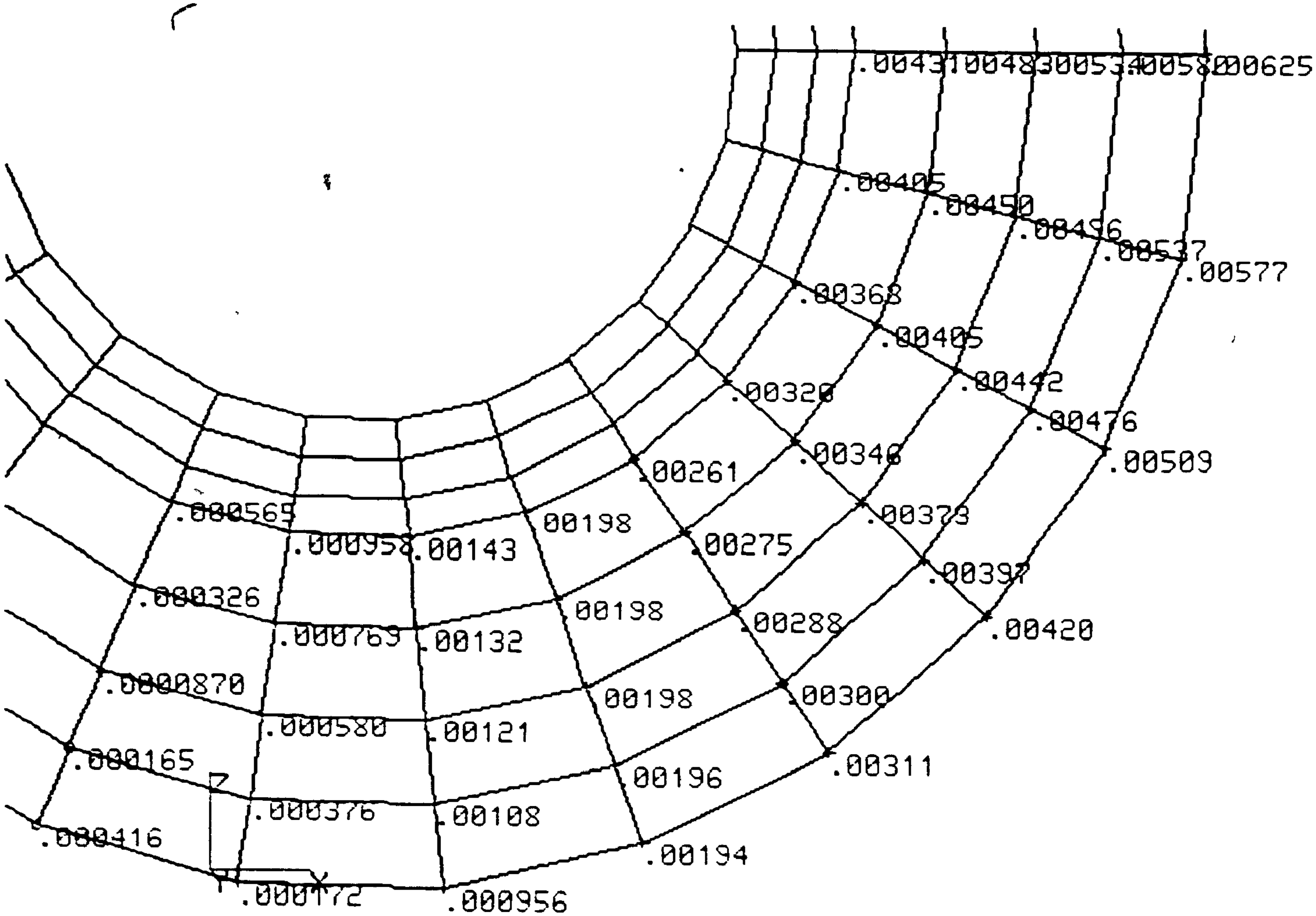


Figure 3.36: Axial (Z') displacement imposed by the washer to the top bushing when the lug/pin joint is submitted to *HL2*, (cont'd)

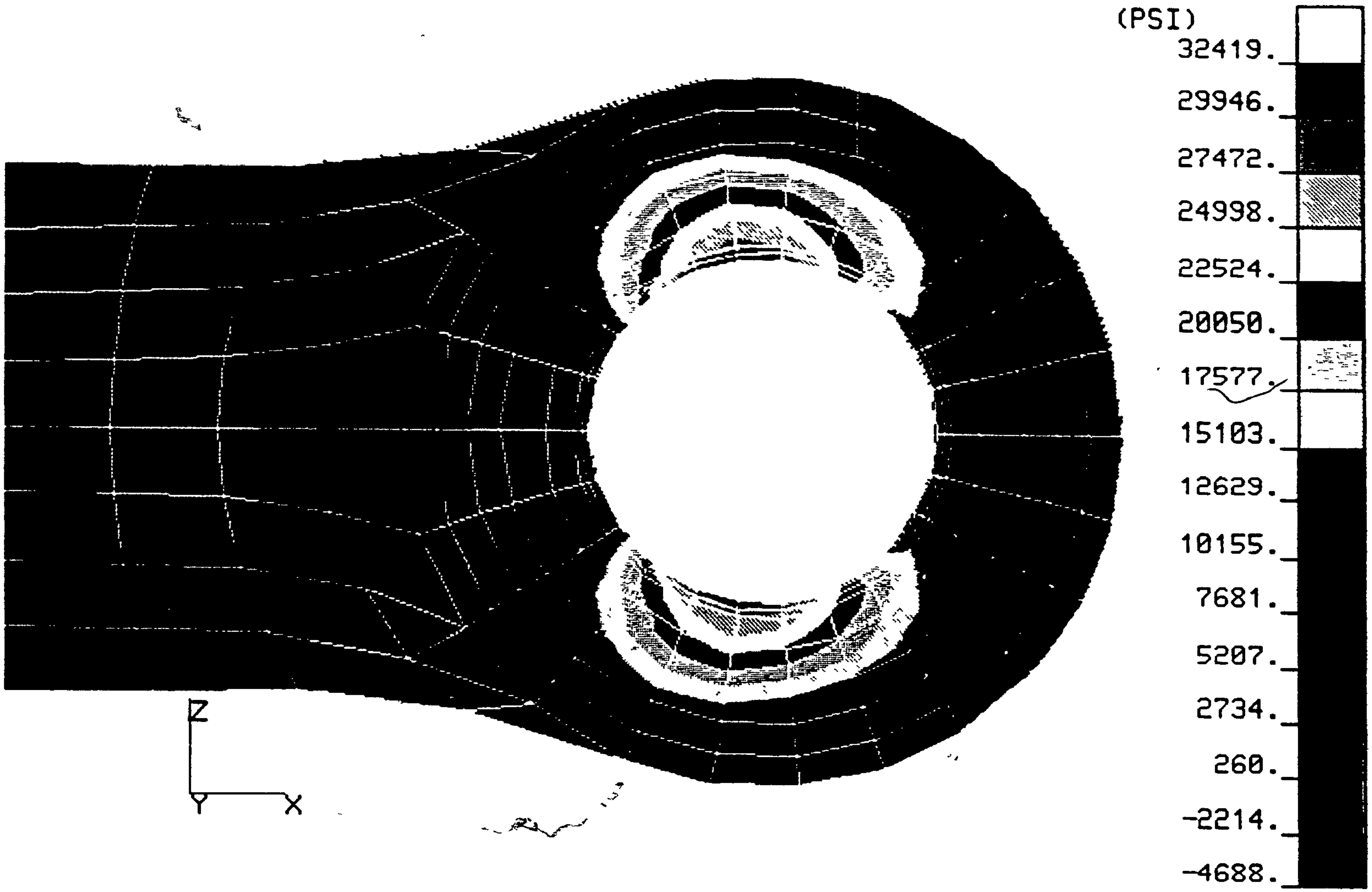


Figure 3.37: Top lug tangential stress field when submitted to *LL1*

The lug is seen from below.

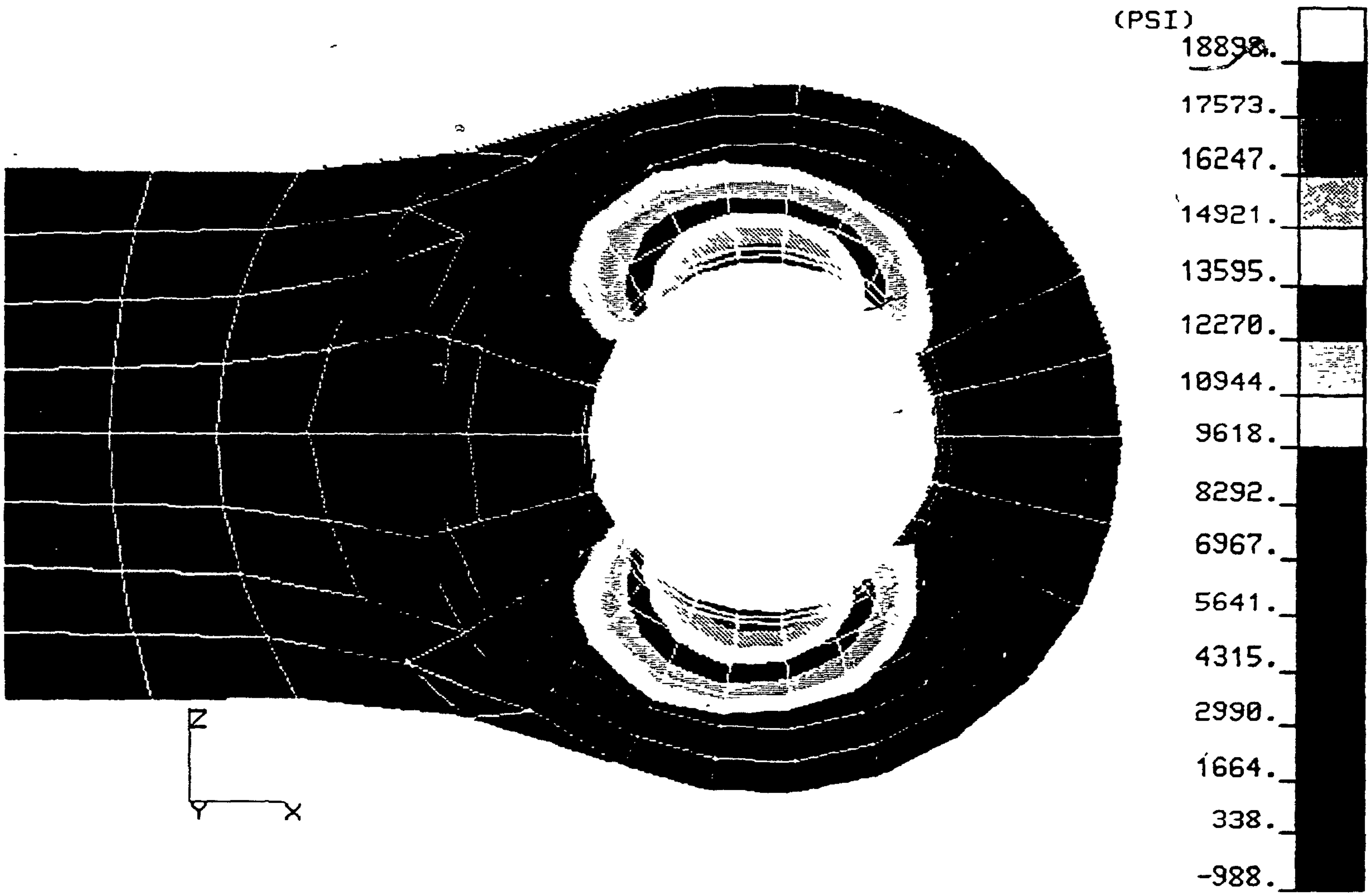


Figure 3.38: Top lug tangential stress field when submitted to *LL2*

The lug is seen from below.

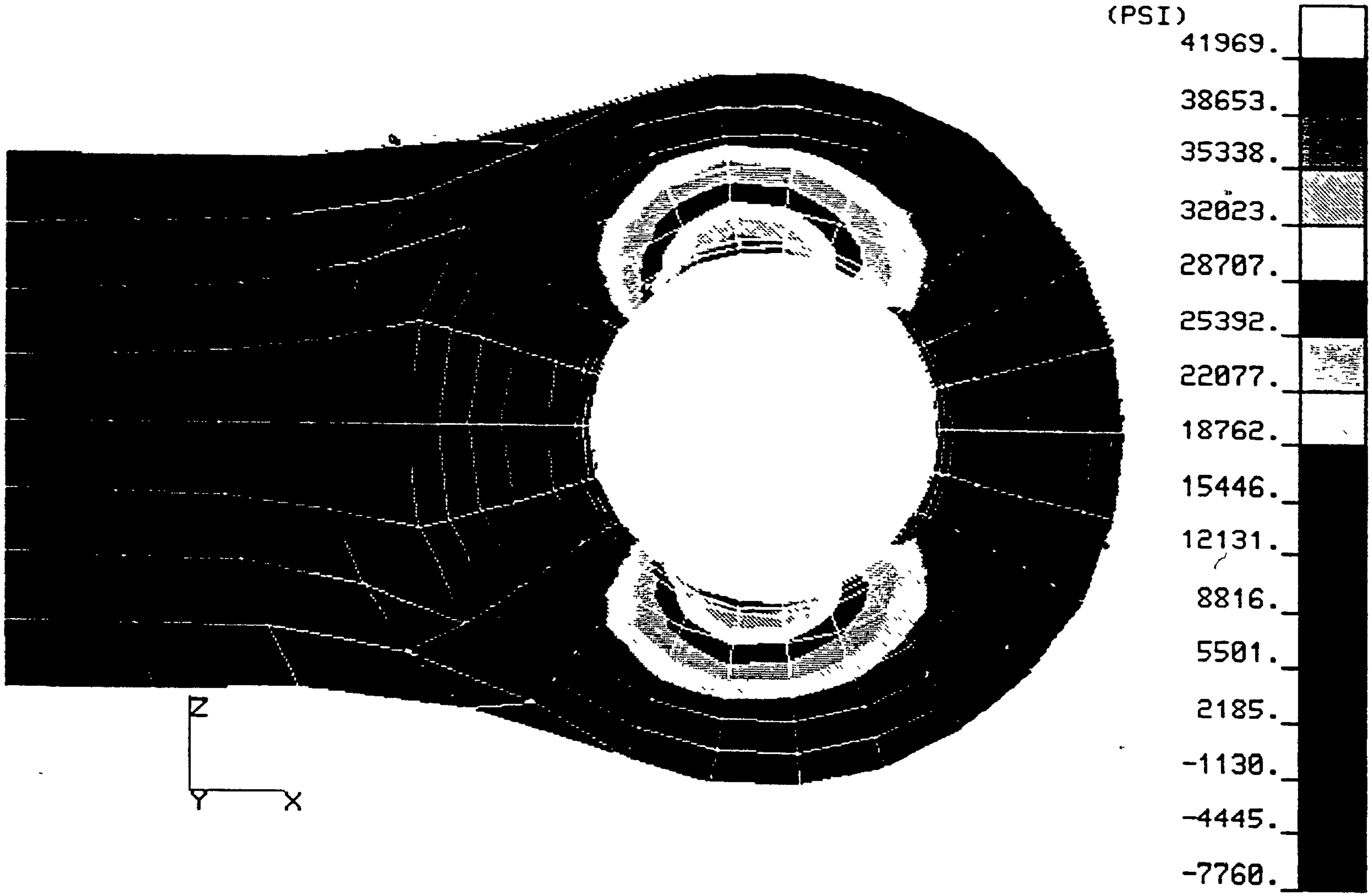


Figure 3.39: Top lug tangential stress field when submitted to *HL1*

The lug is seen from below.

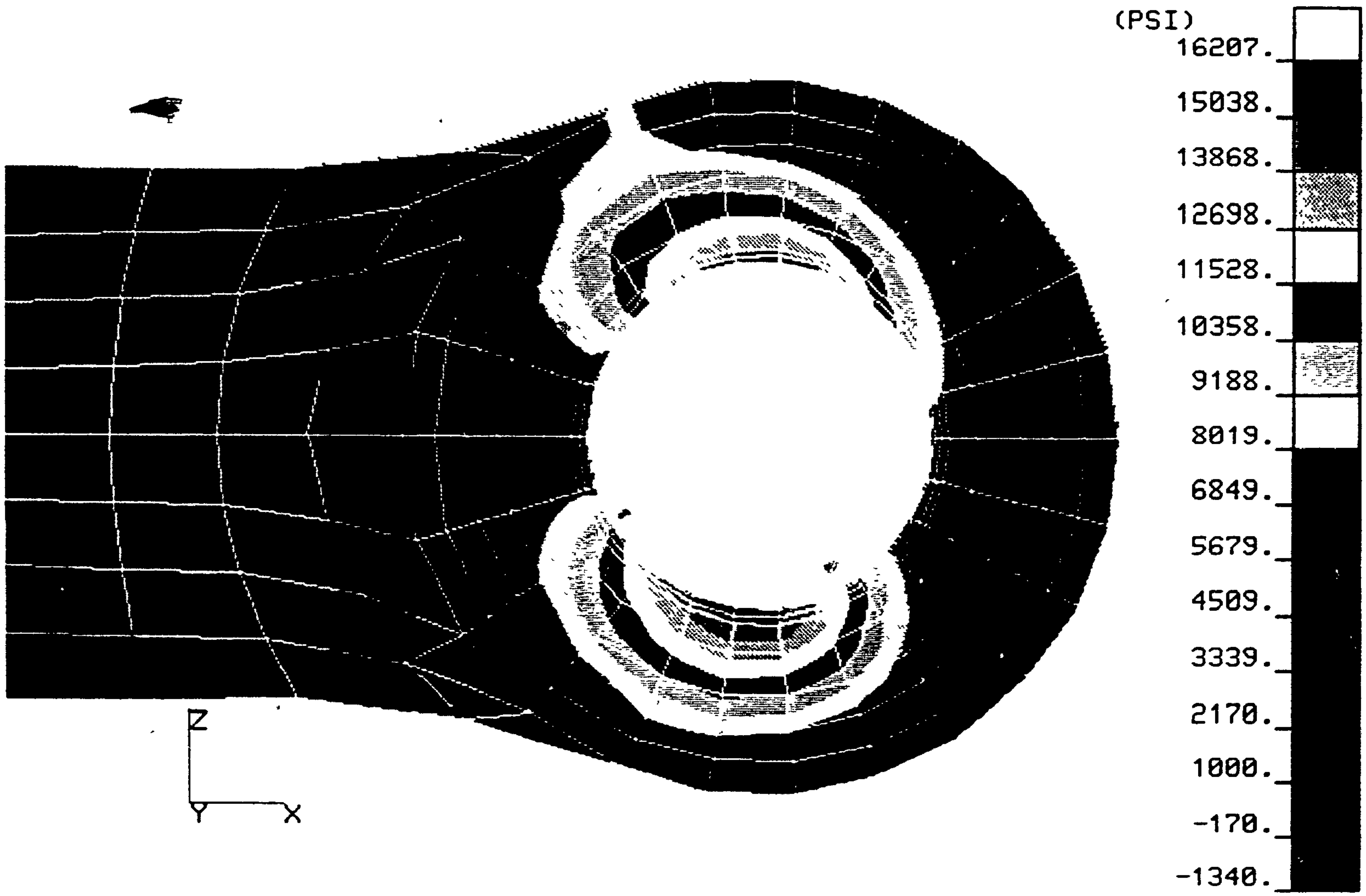


Figure 3.40: Top lug tangential stress field when submitted to *HL2*

The lug is seen from below.

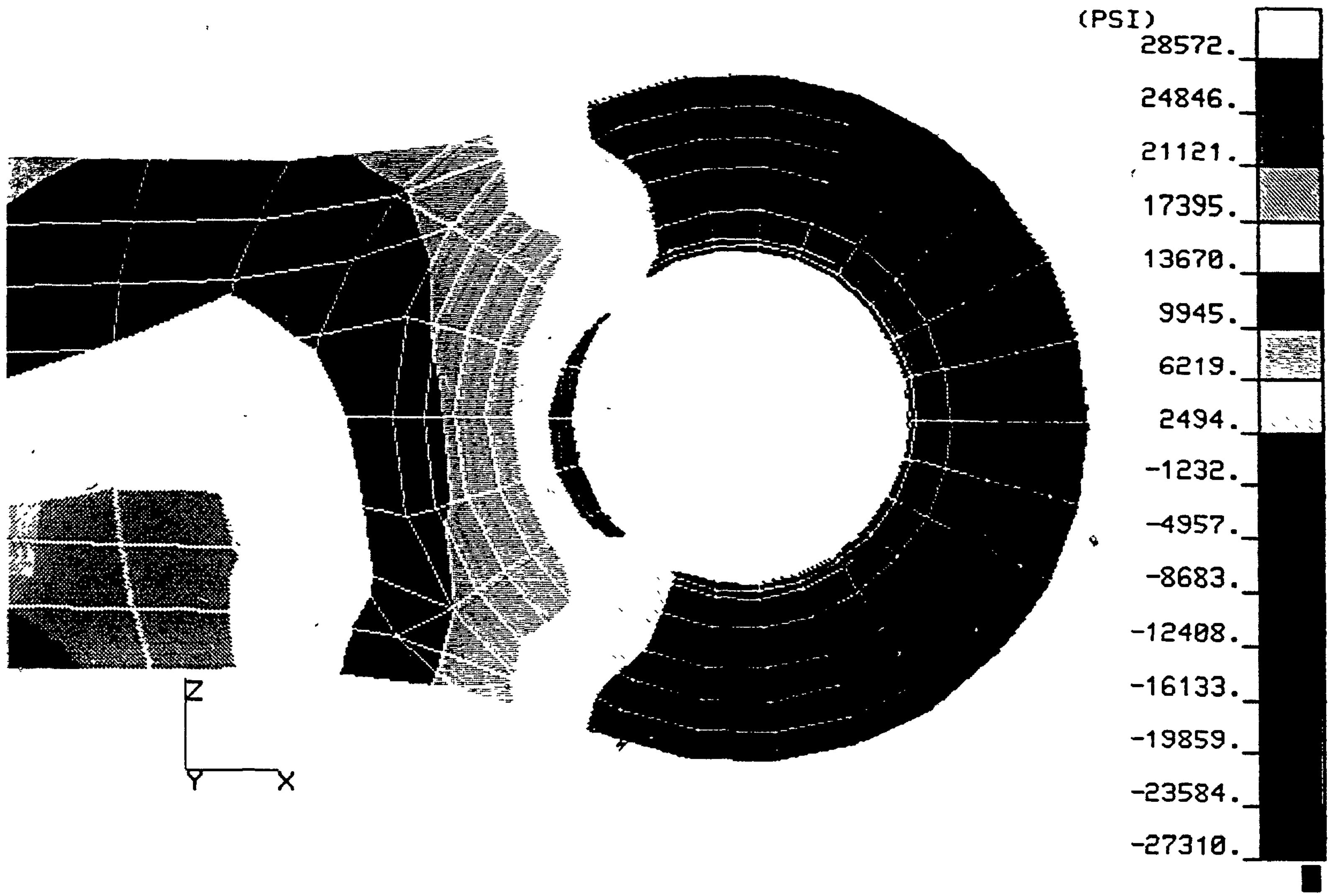


Figure 3.41: Radial stress field of the lug when submitted to the *HL1*.

submitted to *HL1*. Since the gap stiffness when closed was set to $5.0E7lb/in$, it means that it had "over-deformed" $0.01E-3in$ (Figure 3.42). The radial displacement of nodes 513 and 1301 were $3.953E-3$ and $2.963E-3inch$ respectively. So the "over-deformation" was approximately 0.3% of the deformation. This is small.

Results obtained from the coarse F.E. mesh can be compared with those obtained with the fine F.E. mesh. In particular, Figure 3.26 can be compared to Figure 3.37. It can be seen that the coarse F.E. model does underestimate the maximum hoop stress by a factor of 1.4. (For reasons that have been explained in Section 3.3.1, it is suspected that the fine mesh still underestimates the stress concentration factor at the lug hole).

In principle, the displacement of the free nodes located on the fine model boundary should be equal to the displacement of the corresponding nodes on the coarse model. Table 3.11 shows that this is indeed the case.

Table 3.11: Comparison of the displacements of 4 pairs of nodes when the lug/pin joint is submitted to *HL1*

Nodes position: $R = 0.5in, \theta = 180°, Z' = 2.596in$		
Coarse	F.E. model, node 938:	$-5.26, 2.02, -3.49\ (\times 10^{-3}in)$
Fine	F.E. model, node 1214:	$-5.16, 2.02, -3.53\ (\times 10^{-3}in)$
Nodes position: $R = 1.235in, \theta = 56.208°, Z' = 1.971in$		
Coarse	F.E. model, node 83:	$2.31, -6.29, -0.22\ (\times 10^{-3}in)$
Fine	F.E. model, node 416:	$2.35, -6.45, -0.29\ (\times 10^{-3}in)$
Nodes position: $R = 1.125in, \theta = 112.416°, Z' = 1.874in$		
Coarse	F.E. model, node 824:	$-4.56, -6.85, -2.05\ (\times 10^{-3}in)$
Fine	F.E. model, node 849:	$-4.53, -6.89, -2.05\ (\times 10^{-3}in)$
Nodes position: $R = 0.6in, \theta = 180°, Z' = 2.596in$		
Coarse	F.E. model, node 19:	$-3.37, 1.62, -4.09\ (\times 10^{-3}in)$
Fine	F.E. model, node 205:	$-3.17, 1.67, -4.09\ (\times 10^{-3}in)$

It has also been noticed that the bushing and the lug do not stay in contact

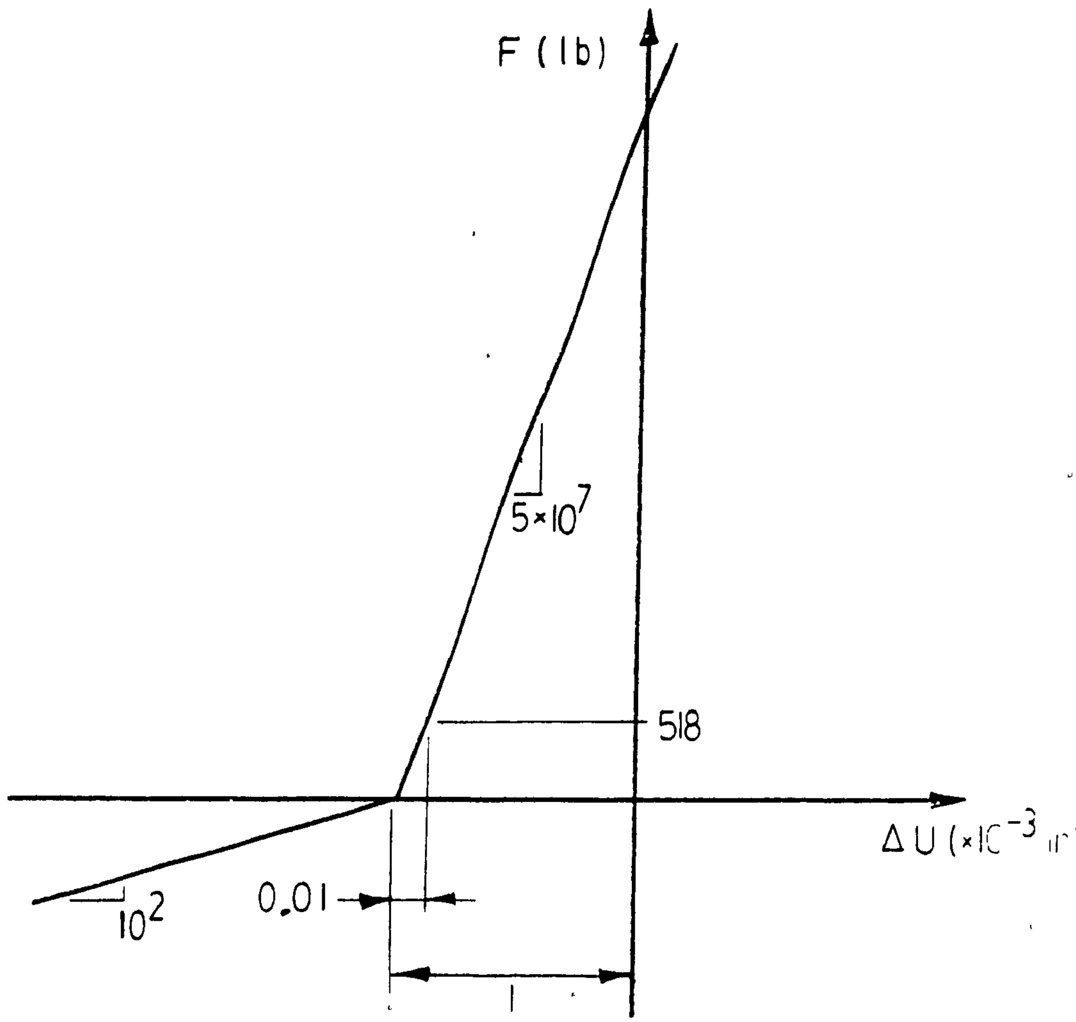


Figure 3.42: Gap Element #12064 deformation when the lug/pin joint is submitted to ***HL1***

in the R direction when the lug/pin joint is highly loaded. By checking the axial forces in the gap elements defining the lug & bushing radial interface, it can be seen that some gaps are open. It seems that the lug & bushing interference is not large enough. Figure 3.43 shows the worst case obtained. This is detrimental from a fatigue point of view since the relative motion that occurs results in fretting.

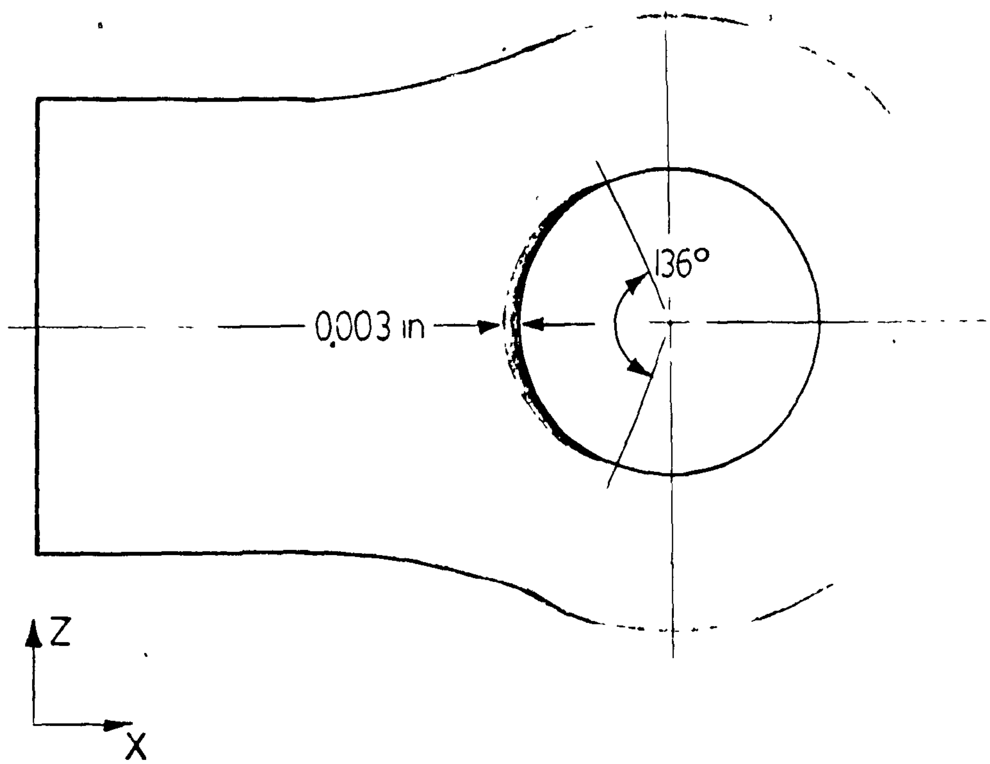


Figure 3.43: When submitted to *HL1*, the top lug and bushing lose contact on the zone indicated

Chapter 4

Fatigue Analysis

In chapter 3, the stress states corresponding to the extreme loading conditions have been found, now the fatigue analysis as such can proceed. Five components of the lug/pin joint assembly are critical and their fatigue behaviors need to be studied. They are the pin, the lugs (2) and bushings (2).

The fatigue analysis methodology developed in this thesis involves the following steps.

1. Computation of a working S-N diagrams, i.e., expressing the acceptable uniaxial alternating stress S_{aa} as a function of the number of cycles N and the uniaxial mean stress S_m:
$$S_{aa} = f(N \ , \ S_m). \tag{4.1}$$
2. Computation of the "equivalent" alternating stress field and the "equivalent" mean stress field for the critical components.
3. Checking whether crack propagation is possible and, if so, solving eqn(4.1) where N is the unknown. Cracks will initiate at the point having the shortest life which will be the life of the assembly. If a component has an infinite life its "reliability factor" (Section 4.5.1) after 10^7 cycles will be evaluated.

The ENDUR program performs the 3 steps explained above. The user guide to this program is in Appendix L.

Sections 4.1 to 4.3 of the present chapter explain the concepts used in the ENDUR program and how they have been applied to the lug/pin joint under study. Section 4.4 explains the cumulative damage concept while Section 4.5 explains the computation of reliability while Section 4.6 gives the results of the fatigue analysis of the lug/pin joint.

4.1 Determination of the Working $S-N$ curves

The fatigue analysis can be performed with the help $S-N$ curves which show the fatigue resistance of materials (see Figure 1.1).

In practice, "basic" $S-N$ curves, obtained by testing standard polished unnotched specimens, must be modify to take into account the "actual conditions" the components have to withstand. These "conditions" may be related to the environment (e.g. corrosion), the surface condition (e.g roughness), the presence of stress raisers, etc. In the present analysis however, the stress concentration factors will not be considered in the computation of the working $S-N$ curves. Stress concentrations will be considered at a later stage (Section 4.2.1).

4.1.1 Basic $S-N$ curves

"Basic" $S-N$ curves are obtained by testing standard specimens of materials usually under fully reversed axial loading. Specimens can also be tested in alternating bending, rotating bending, or in alternating torsion. Results thus obtained are different but can be correlated [7].

For steels, their is an alternating load, called the "fatigue limit", below which no fatigue failure will occur. For other materials, there is no such load. The improperly called "fatigue limit" of these materials is often taken as the maximum alternating load allowed for a finite life (10^7 cycles for example).

In actual testing of specimens under fatigue, there is an important scattering of the results which gets wider as the endurance increases. This is due to the fact that, as the alternating load decreases, the crack *initiation* stage lasts longer when compared to the crack *propagation* stage. Crack initiation depends more on the inhomogeneity (small random flaws) of the material. Crack propagation depends more on the material bulk properties. Published data usually correspond to a 50% rate of survival and this has to be considered (Section 4.5).

Many attempts have been made to express the $S-N$ curve in mathematical forms. Most of them, particularly in the high-cycle fatigue range, are not based on any theory, they are simply obtained by "curve fitting" test results. For this research, the required $S-N$ curves have been approximated using the equation given in [8], namely:

$$S_{a0} = S_f + (S_u - S_f)\exp -\alpha(\log N)^{\beta}, \tag{4.2}$$

where:

- S_f is the fatigue limit [psi],
- S_u is the tensile strength [psi],
- N the endurance, i.e. the number of cycles to failure,
- α, β are curve fitting parameters, and
- S_{a0} is the zero mean stress fatigue strength.

A less attractive alternative is

$$S_{a0} = S_f + \frac{S_0}{N^\gamma}, \tag{4.3}$$

where S_0 and γ are fatigue curve fitting parameters.

Eqn(4.2) has been preferred to eqn(4.3) because eqn(4.2) permits, in principle, the prediction of the fatigue strength for $N = 1$ to ∞. Eqn(4.3) is valid for a narrower range ($N = 10^3$ or 10^4 to ∞). Figure 4.1 compares the $S-N$ curve based on eqn(4.2) and eqn(4.3).

Restricting attention to eqn(4.2), values of α and β for the lug, pin and bushing materials are given in Table 4.1. Details on how they have been computed are given in Appendix K.

Component	α	β
Pin (4340 steel)	$1.6120E-4$	5.9962
Lug (7075 aluminum)	$4.49887E-2$	1.61941
Bushing (17-4 PH stainless steel)	$4.2884E-3$	3.6592

Table 4.1: Values of α and β for the Lug, Pin and Bushing Materials

4.1.2 The mean stress effect

For uniaxial loading, the mean stress "S_m" is defined as:

$$S_m = \frac{1}{2}(S_{max} + S_{min}). \tag{4.4}$$

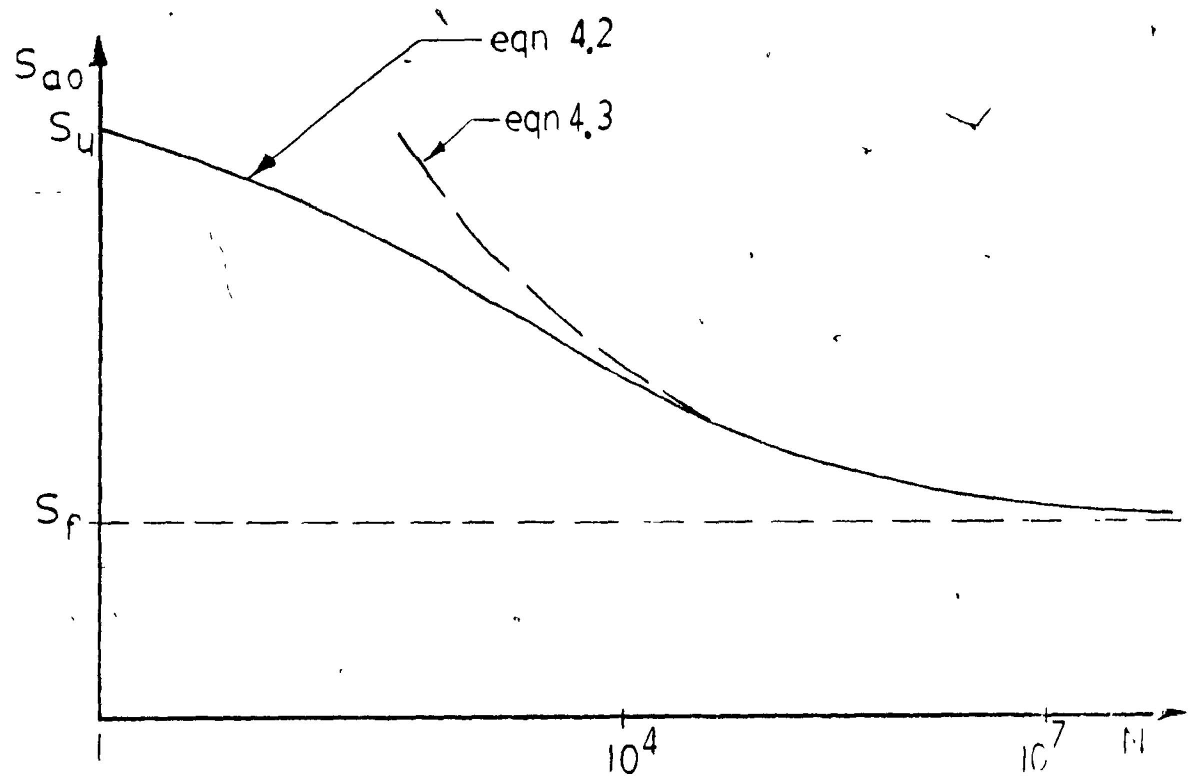


Figure 4.1: Comparison between 2 mathematical expressions used to approximate the $S - N$ curves

Eqn(4.2) has been chosen in this investigation because it gives better predictions on a wider range of N.

The alternating stress or stress amplitude "S_a" is defined as:

$$S_a = \frac{1}{2}(S_{max} - S_{min}). \quad (4.5)$$

Figure 4.2 illustrates the above definitions.

Various equations relating the acceptable alternating stress to the applied mean stress have been proposed. The best known are the Goodman's line and Gerber's parabola.

$$\frac{S_a}{S_{a0}} = 1 - \left(\frac{S_m}{S_u}\right)^2 \quad , \quad \text{Gerber's parabola,} \quad (4.6)$$

$$\frac{S_a}{S_{a0}} = 1 - \left(\frac{S_m}{S_u}\right) \quad , \quad \text{Goodman's line.} \quad (4.7)$$

Another line that has been proposed is the Sine's line:

$$S_a = S_N - \frac{m}{\sqrt{2}} S_m, \quad (4.8)$$

where S_N is the fatigue strength for a life "N". The curve fitting parameter "m" is chosen to get the "best fit" line on the S_m interval of interest (usually $0 < S_m < S_y$, S_y being the yield strength of the material). Near the fatigue limit, [9] proposes to use $m = 0.5$ if no better data are available. Unfortunately, m may change when $N < 10^7$. To overcome this problem, the following modification to eqn(4.8) is proposed:

$$\frac{S_a}{S_{a0}} = 1 - \frac{M}{\sqrt{2}} \left(\frac{S_m}{S_u}\right), \quad (4.9)$$

the parameter "M" being assumed independent of "N".

The Gerber's parabola tends to be unconservative for S_m greater than 0 and overly conservative if extrapolated for S_m lower than 0. The opposite is true for the Goodman's line (see Figure 4.3). Eqn(4.9) is valid on a narrow range of S_m. Because of this, eqn(4.10) is preferred for an evaluation of the mean stress effect:

$$\left(\frac{S_a}{S_{a0}}\right) = a_2 \left(\frac{S_m}{S_u}\right)^2 + a_1 \left(\frac{S_m}{S_u}\right) + 1, \quad (4.10)$$

where a_1 and a_2 are curve fitting parameters which are assumed independent of "N". Note that if $a_2 = 0$ and $a_1 = -1$; the eqn(4.10) becomes the Goodman's line; if $a_2 = -1$ and $a_1 = 0$, it becomes the Gerber's parabola and if $a_2 = 0$ and a_1 is set $= -\frac{M}{\sqrt{2}}$, eqn(4.10) becomes eqn(4.9).

Table 4.2 gives the values of a_1 and a_2 for the lug, pin and bushing materials. Details of their computation are given in Appendix K.

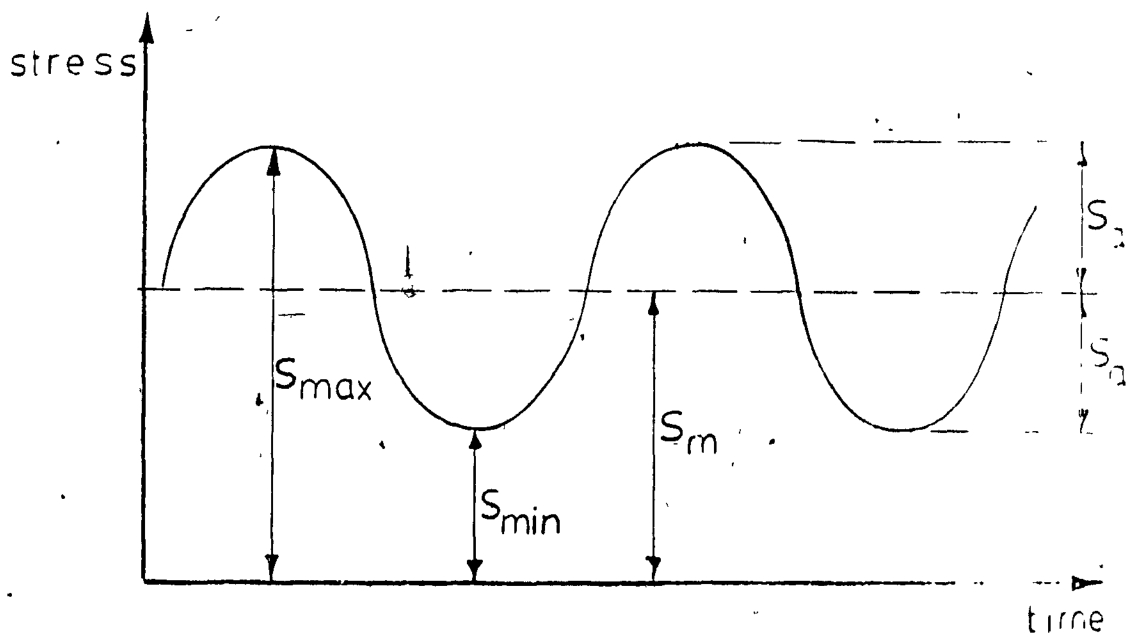


Figure 4.2: S_{max}, S_{min}, S_a, S_m definitions

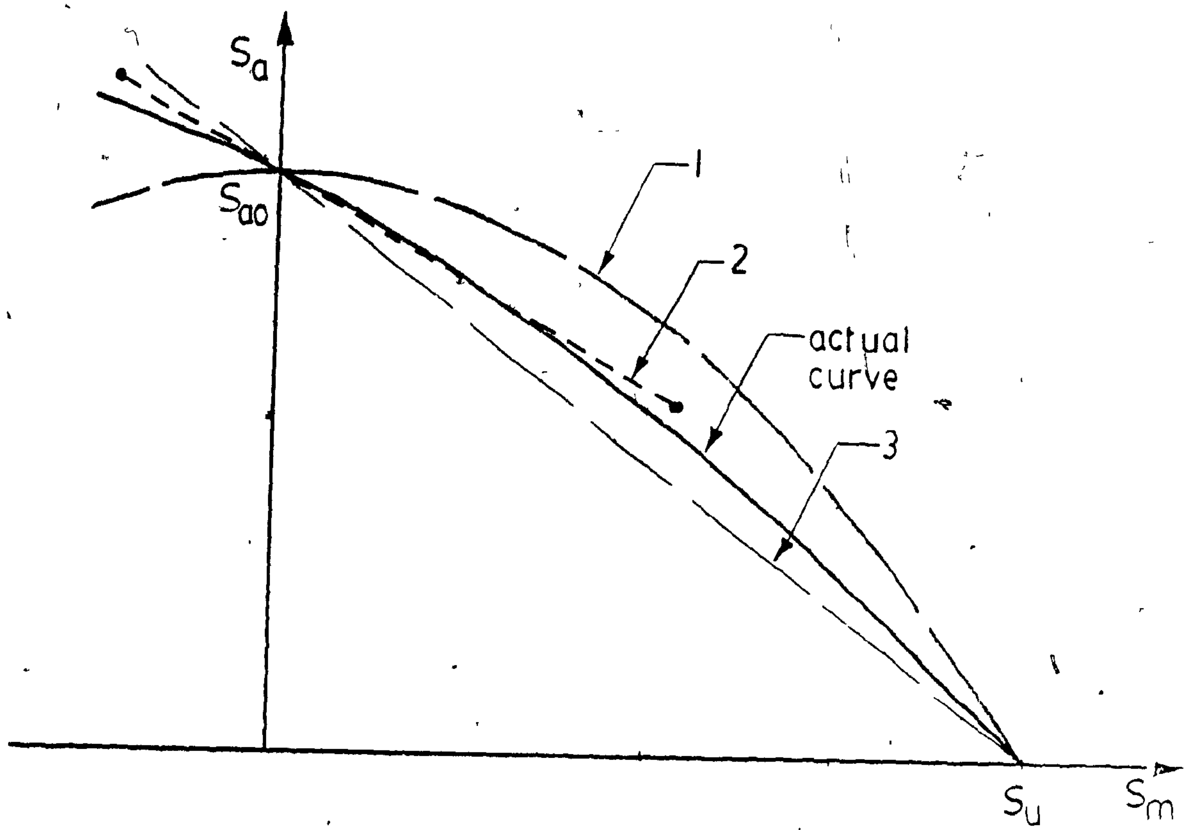


Figure 4.3: Comparison of Goodman's Line, Gerber's Parabola and Proposed Equations

1. Gerber's parabola
2. Equation 4.9
3. Goodman's line.

Equation 4.10 is close to the actual curve.

4.1.3 $S_a = f(N, S_m)$ diagram

To help visualize how the acceptable alternating stress S_a varies with N and S_m, eqns(4.2 & 4.10) have been schematically combined as shown in Figure 4.4.

4.1.4 Modifying Factors

The "working" $S-N$ curve is a $S-N$ curve once the factors that reduce the material fatigue strength have been taken into account. Table 4.3 shows the main factors that may influence the fatigue behavior of metallic components. Among these factors, only those marked (*) are relevant to the present study.

Size Effect

Size effect for steels. For steels, the fatigue limit in air decreases as the specimen size increases. The explanation of this behavior is based on the idea that it is the "weakest link" that governs the fatigue strength [9]: in a larger specimen, the chance to have a weak zone due to a small flaw in the material is greater.

The following empirical equation has been proposed [9] to relate the fatigue strength of specimens of different sizes:

$$S_f = S_{f0} \left(\frac{V}{V_0} \right)^{-0.034}, \tag{4.11}$$

where:

- V_0 is the volume submitted to a stress greater than 95 % of the maximum stress in the reference specimen,

Component	a_1	a_2
Pin (4340 steel)	−0.4544	−0.5456
Lug (7075 aluminum)	−1.0	0.0
Bushing (17-4 PH stainless steel)	−0.5 assumed	−0.5 assumed

Table 4.2: Values of a_1 and a_2 for the Lug, Pin and Bushing Materials

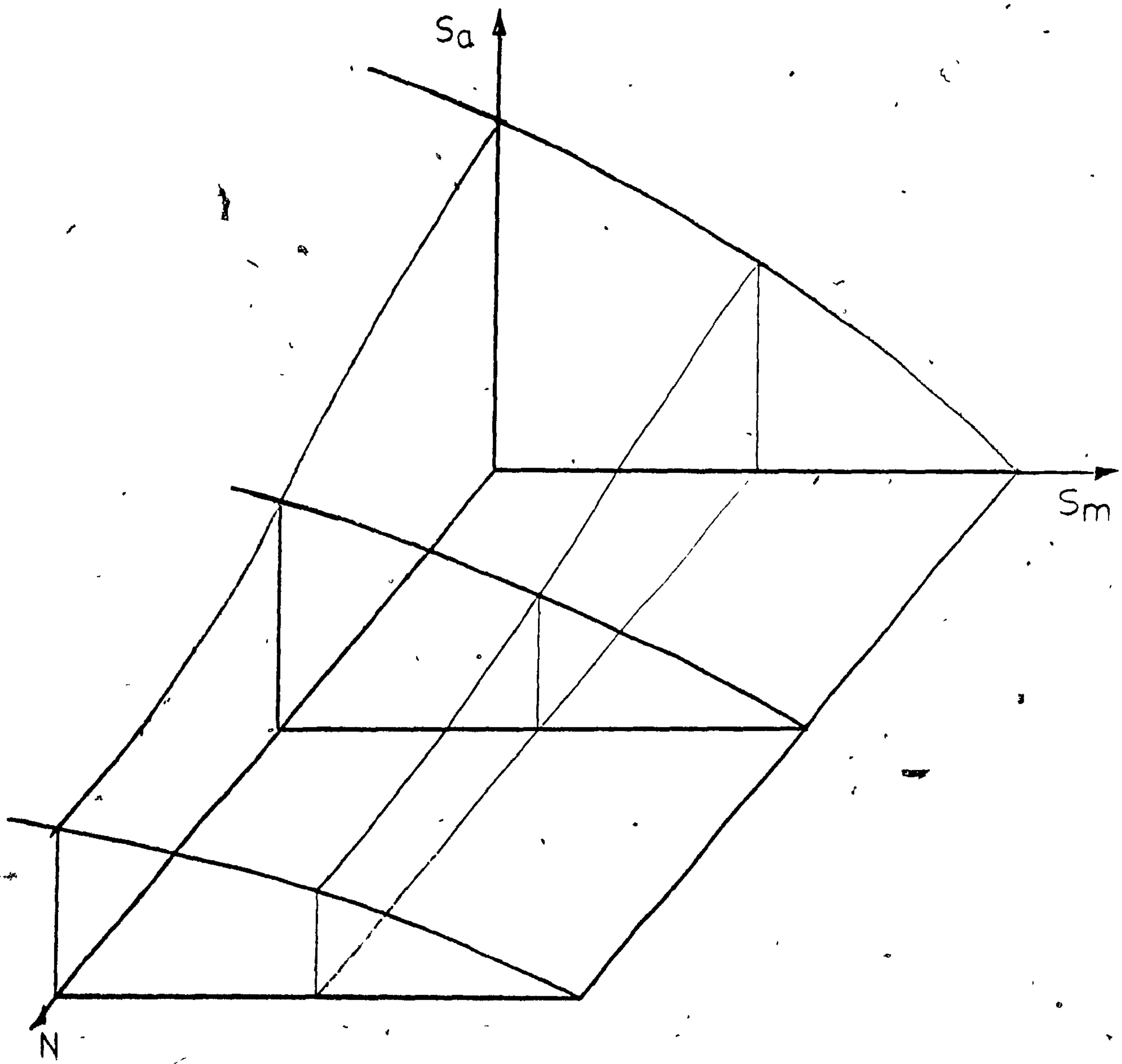


Figure 4.4: Three Dimensional Schematic Illustrating $S_a = f(S_m, N)$

- V is the volume submitted to a stress greater than 95 % of the maximum stress in the other specimen,
- S_{f0} is the fatigue limit for the reference specimen, and
- S_f is an estimate of the fatigue limit for the other specimen.

Eqn(4.11) can be applied to specimens having different shapes and submitted to different type of loadings. The size effect factor can therefore be computed using the expression:

$$K_{size} \doteq \left(\frac{V}{V_0}\right)^{-0.034} \qquad (4.12)$$

Size effect for Aluminum Alloys. For high strength aluminum alloys, the fatigue strength in air (in high cycle fatigue) can be considered as independent of the specimen size [7]. It may be because the humidity in air attacks the aluminum. The loss of fatigue strength that should result from an increased size is compensated by the fact that the environment is less detrimental for larger specimens.

Size effect Variation with N. In short life fatigue (arbitrary chosen as $N < 10^3$ here), the size is considered as having no effect. In long life fatigue (arbitrary chosen as $N > 10^7$ here), the size effect is considered constant. For intermediate life, the size effect is considered to vary linearly with $\log N$. (see Appendix L). To be conservative, the size effect is considered to reduce both the "acceptable" tensile stress AND the "acceptable" alternating stress (see Figure 4.5).

Size Effect for the Lugs. Because the lugs are made of aluminum, the size effect can be neglected.

Size Effect for the Pin. The pin volume is approximately $0.3in^3$. By checking Figures 3.22 to 3.25, the "highly stressed" (i.e., 95 % of the maximum stress) volume is < 20 % of the pin volume. As a result, the highly stressed volume is $< 0.6in^3$. The specimen size being approximately $L = 2in$, $D = 0.4in$ (Appendix K), its volume is $0.6in^3$. The size effect factor is therefore equal to 1.

Size Effect for the Bushings. The volume of each bushing is $0.5in^3$. The "highly stressed" volume for each bushing is $< 0.1in^3$ (i.e., < 20 % of the total volume). This is small enough so that the size effect can be considered as 1.

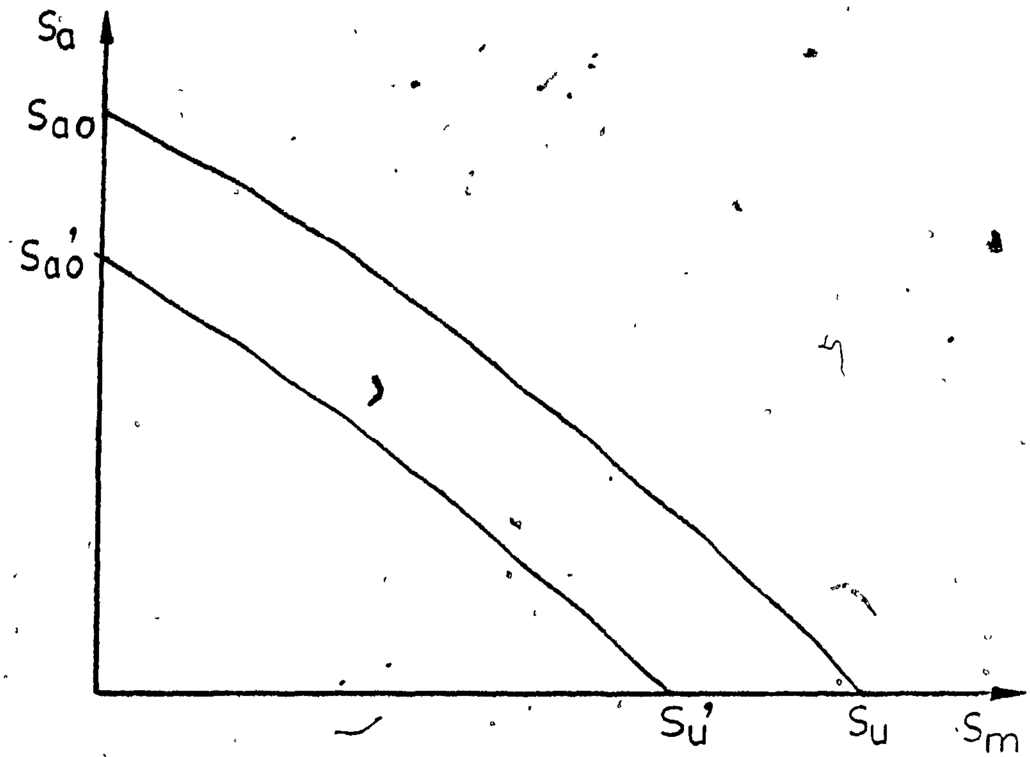


Figure 4.5: The Size Effect

$$
\begin{aligned}
S'_{a0} &= K_{size} \times S_{a0} \\
S'_{u} &= K_{size} \times S_{u} \\
K_{size} &\leq 1.
\end{aligned}
$$

Surface Effect

Almost all fatigue cracks are initiated at surfaces. The surface conditions have therefore a dominant effect on the fatigue life.

Three aspects of the surface are relevant to the lug/pin problem under study. They are:

1. fretting at the components interfaces,
2. surface roughness, and
3. residual stresses induced by shot peening the lugs.

Surface roughness need not be considered when fretting occurs.

Fretting. Fretting involves the behavior of two surfaces in contact subjected to small repeated relative motion. Fretting damage will occur in the contact zone submitted to high shear stress [1]. Some authors attribute fretting damage to the contact of the two surfaces and the rubbing action that provokes microwelding and rupture of the microwelds [7]. Oxydation may occur and, if the debris is harder than the surface, it causes abrasion. Furthermore, fretting fatigue occurs even in an inert environment but will not be as detrimental. Notice, however that fretting fatigue damage decreases with increasing humidity in the atmosphere. Once a fretting induced macrocrack is formed, fretting itself does not control its growth.

Fretting is important in high cycle fatigue but of less importance in short life components. Typically, fretting reduces the fatigue strength by 30 % for smooth specimens [10]. Reductions of up to 90 % are possible [11].

Tables 4.4 and 4.5 show the fatigue strength for various combinations of metals. The weakness of those tables is that they neglect to state some important experimental conditions, such as the normal stress and the slip amplitude. The geometry of the components submitted to fretting can also be important since it may or may not foster the accumulation of debris. Tabled values for fretting fatigue reduction factors are at best approximate.

The mean stress effect in fretting fatigue is shown in Figure 4.6. Notice that fretting does not reduce the "acceptable" tensile strength and does not affect the admissible alternating stress if the mean stress is very low.

In spite of the above limitations on the evalution of the fretting effect in fatigue, that cannot be neglected in the fatigue life evaluation of lug/pin joint components. In the present investigation, it has been considered that fretting has no effect for

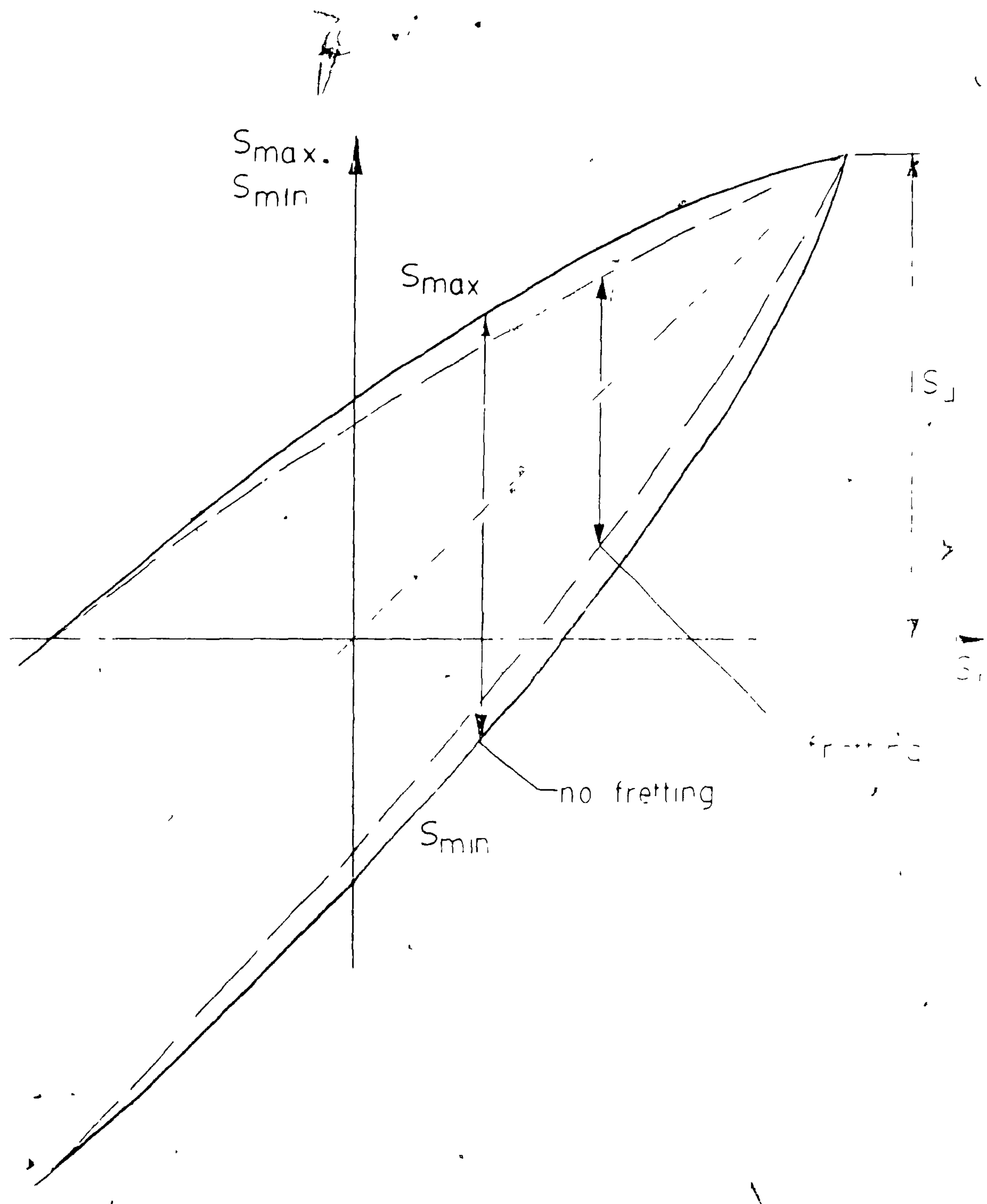


Figure 4.6: Illustration of the Relation between Fretting-Fatigue Strength and the Regular Fatigue Strength for Various Combinations of Axial Loading

From [13].

$N < 10^3$, a constant effect for $N > 10^7$, and a negative tangential mean stress decreases the fretting effect. Refer to Appendix L ("ENDUR" program) for details on how fretting effect has been incorporated into the analysis. Notice that the fretting fatigue effect coefficients computed in the ENDUR program do not take into account either the normal stress or the slip amplitude.

Fretting Effect at the Lugs. The surface of the lugs in contact with the bushings is submitted to fretting. The lugs are made of aluminum, the bushings are made of stainless steel. The fretting of aluminum on stainless steel is known to be very damaging for the aluminum. Since the thin and soft cadmium plating on the bushes may wear rapidly at some points, its presence is neglected. This is a conservative assumption. In the lug/pin joint under study, the slip amplitude at the bush/lug interfaces is very important (from the F.E. analysis, it has been determined it may go as high as $0.002in = 50\mu m$). As explained in the previous paragraph, quantifying the fretting is not easy. As an indication, we can refer to Figure 4.7. It is seen that, when the slip amplitude is larger than $7.5\mu m$, the fretting strength reduction factor is greater than 5. Unfortunately the normal stress at which the experiment was carried is not mentioned. It is probably high since factors of 2 or 3 are more usual. On the other hand, fretting of steel on aluminum is more detrimental than aluminum on aluminum. Bearing in mind those considerations, a fretting fatigue reduction factor of 4 (at zero mean tangential stress and $N = 10^7$), is considered to be reasonably conservative for the lug/pin joint under study.

Fretting Effect at the Pin. There is a copper liner in the sleeve. The fretting of copper on steel is known to be relatively undamaging for the steel. The fretting strength reduction factor for the section of the pin in contact with the sleeve has been assumed equal to 1.5 (at zero mean tangential stress and $N = 10^7$).

The fretting at the pin/bushing interface is serious because both components are made of steel and the slip amplitude between them is important. It may not be as detrimental as the bushing/lug fretting though. The fretting strength reduction factor for the sections of the pin in contact with the bushings has been assumed equal to 2.5 (at zero mean tangential stress and $N = 10^7$). Figure 4.8 shows the pin surface submitted to fretting.

Fretting Effect at the Bushings. Each bushing is in contact with the pin, a lug, and a washer. To make the computation tractable, the fretting fatigue reduction

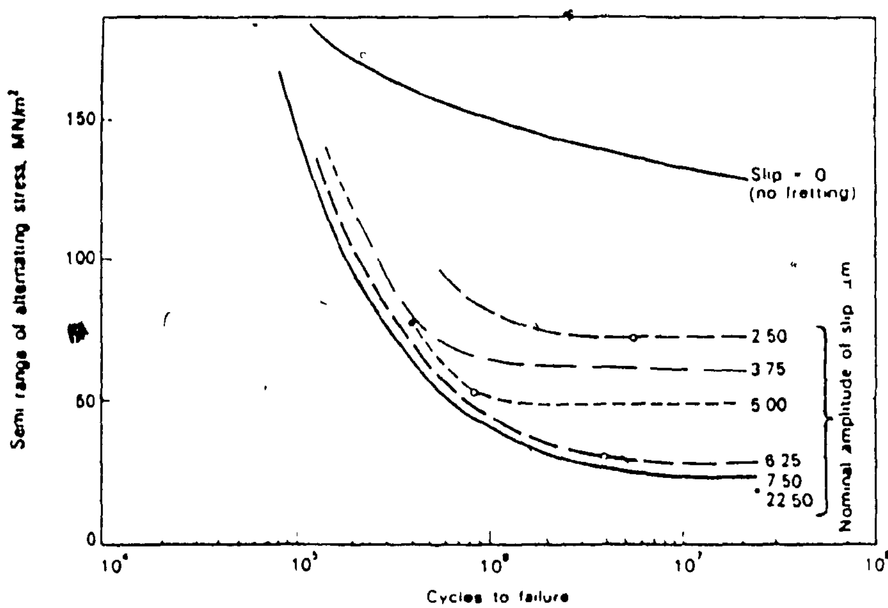


Figure 4.7: Effect of Amplitude of Slip on the Fretting Fatigue of Aluminum Alloy From [11].

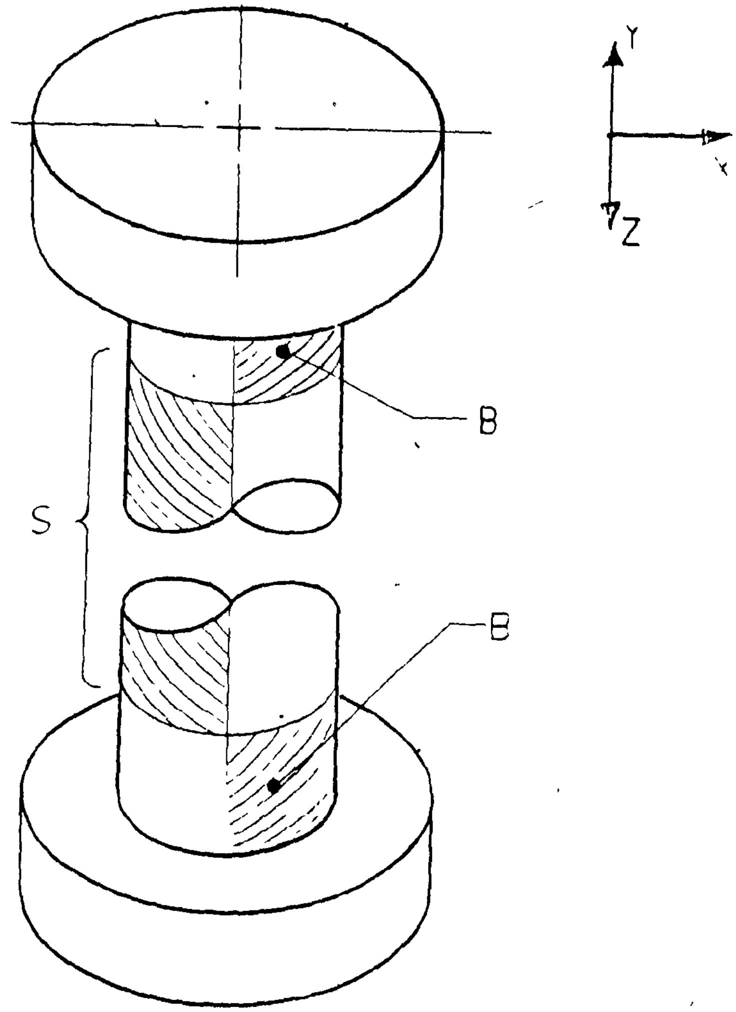


Figure 4.8: Pin surface submitted to fretting

Because of the loading conditions and the pin/sleeve & pin/bushing clearances, the pin surface is not completely submitted to fretting. Only the shaded surfaces are considered as being submitted to fretting. The surface indicated by "S" is considered in contact with the sleeve while those indicated by "B" are considered in contact with the bushings.

factor is assumed to be constant on all faces subjected to fretting and equal to 2.5 (at zero mean tangential stress and $N = 10^7$). It will be shown later that the bushings are far from being critical. Figure 4.9 shows the bushings surface submitted to fretting.

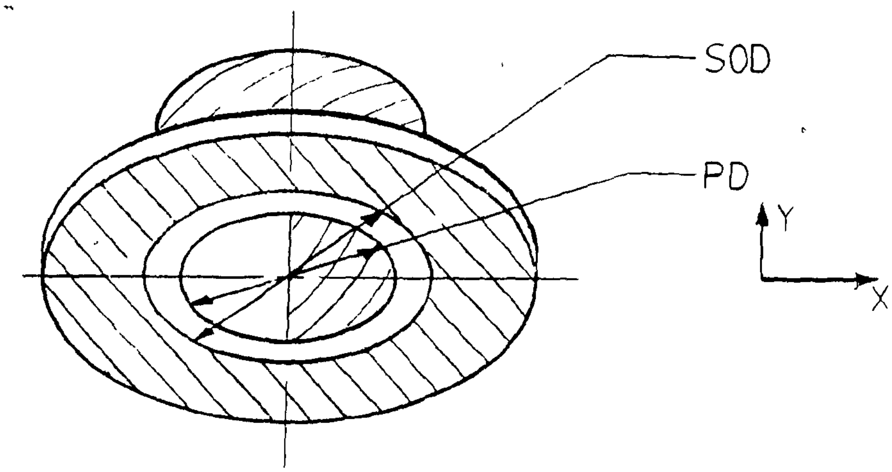


Figure 4.9: Bushing surface submitted to fretting

The shaded area is considered as being submitted to fretting. Refer to Table A.1 to know the meaning of the "SOD" and "PD".

Roughness effect. Surface irregularities can be seen as "micro" stress raisers. Components with rougher surfaces therefore show worse fatigue behaviors than similar components with smooth surfaces.

In short life fatigue ($N < 10^3$), the surface roughness is considered as having no effect. In long life fatigue ($N > 10^7$), the effect is considered constant. For intermediate life, the surface roughness effect is considered to vary linearly with $\log N$. (see Appendix L). To be conservative, the surface roughness effect is considered to reduce both the "acceptable" tensile stress AND the acceptable alternating stress. Figure 4.10 shows how the high cycle fatigue strength reduction coefficient varies with the tensile strength of steels.

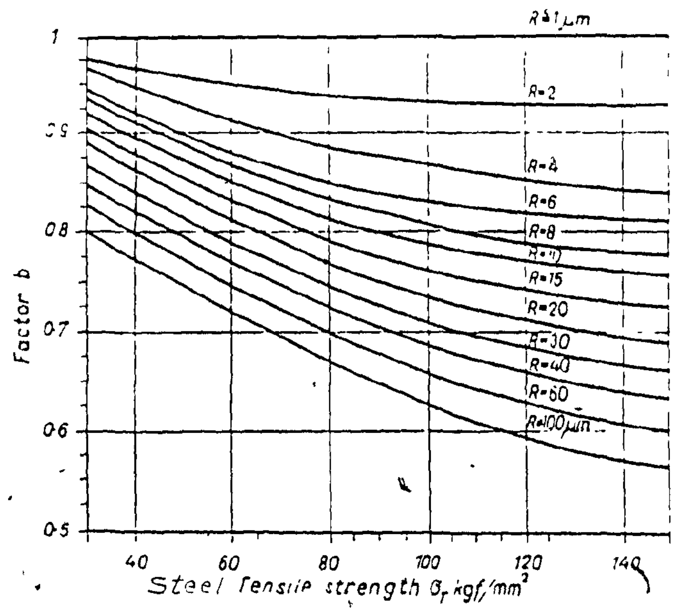


Figure 4.10: Surface Finish Factor for Bending Fatigue Strength

From [10].

As stated in Section 2.1, all the components of the lug/pin joint under study have surface roughnesses of $125\mu in$ ($\approx 3\mu m$). For the strongest material (pin material

$S_u = 180ksi = 126\frac{kgf}{mm^2}$) which is the most influenced by surface roughness, the factor from Figure 4.10 is ≈ 0.88. This factor is used on all lug/pin component surfaces that are not influenced by fretting. The results will show that none of the unfretted surface is critical.

Surface Residual Stresses. In high cycle fatigue, residual stresses can be treated as an externally induced mean stress (see Section 4.1.2 & [14,15,9,16]). In low cycle fatigue on the other hand, microplastic deformations relax the residual stresses during the fatigue process and, hence, they may be neglected.

For the lug/pin joint under study, there is no significant residual stresses on the pin and bushings. The lug hole walls, however, have been shot peened. This is considered in the following paragraph.

Shot Peening of the Lug Hole Walls. The lug hole walls have been shot peened. The stress induced by shot peening is shown in Figure 4.11.

Since it is not easy to measure, there is always an uncertainty on the value of compressive stresses induced by shot peening on an actual component. Moreover, the compressive stresses may partially relax due to fretting or high temperatures induced by friction. For these reasons, the shot peening induced stresses will be considered as 60 % of the minimum value shown in Figure 4.11, i.e., $60\% \times 50ksi = 30ksi$. The value $30ksi$ will be added to both the axial and tangential stresses, i.e., to both $\sigma_{z'z'}$ and $\sigma_{\theta\theta}$, of the nodes located on the lug hole walls.

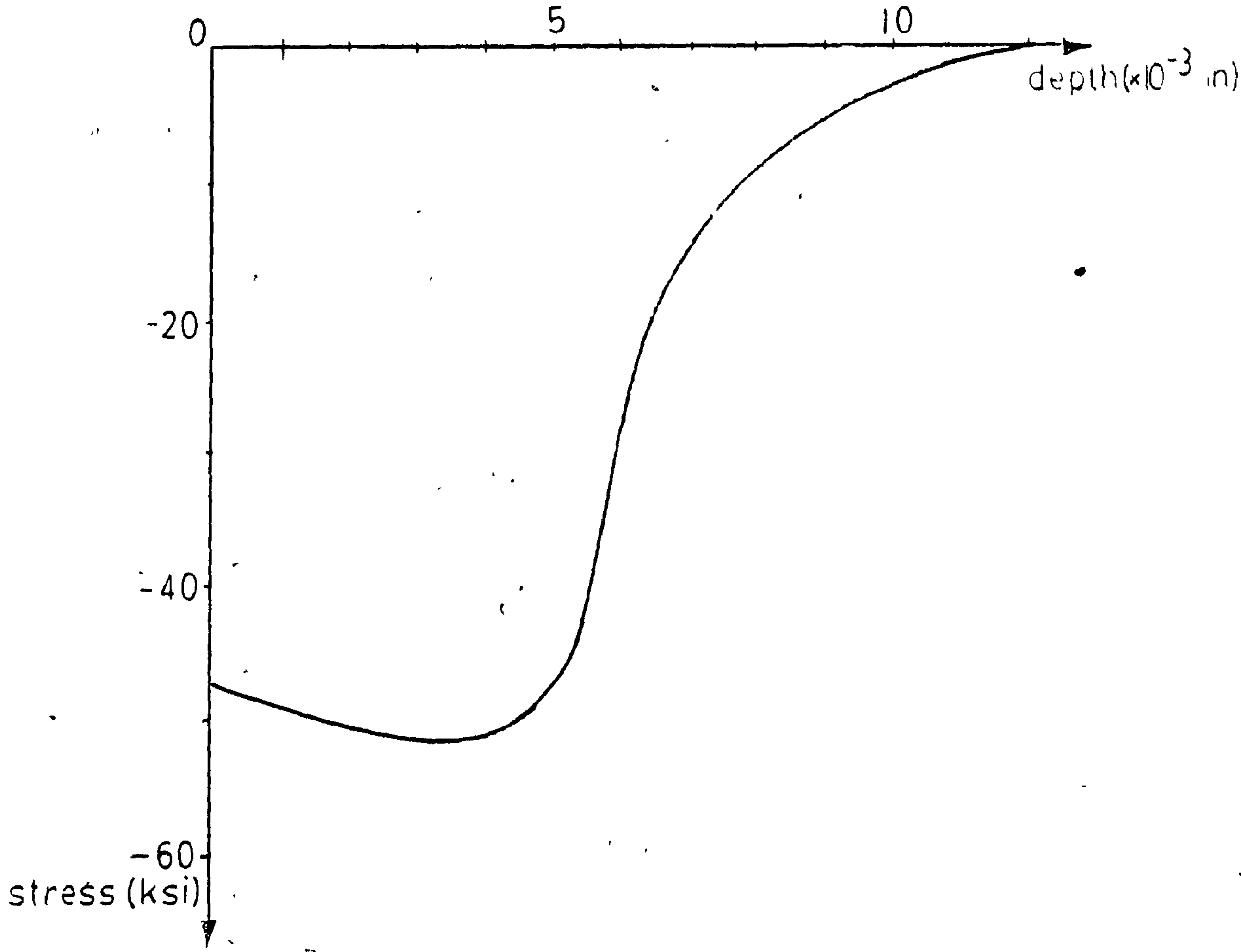


Figure 4.11: Approximative Tangential Residual Stresses Induced by Shot Peening of the Lug

(ref Metal Improvement Co.)

* Component Size	
* Surface effects:	* roughness * fretting * residual stresses
Loading Frequency	
Corrosion	
Temperature	
Metallurgical considerations:	porosity grain size grain orientation

Table 4.3: Fatigue Modifying Factors

	Specimen	Hardness VHN	Fatigue strength MN m^{-2}	Clamp	Fretting fatigue strength MN m^{-2}	Strength reduction factor (SRF)
Carbon steels	0·1 C steel	137	172	0·1 C steel brass zinc	122 95 137	1·41 1·83 1·25
	0·33 C steel	165	372	0·33 C steel	254	1·48
	0·4 C steel	420	550	0·2 C steel 0·4 C steel 70/30 brass Al 4·4 Cu 0·5 Mn 1·5 Mg	450 257 325 500	1·23 2·00 1·70 1·10
	0·7 C steel, cold-drawn 0·7 C steel, normalized	365 270	525 371	0·7 C steel, cold drawn 0·7 C steel, normalized	147 178	3·18 2·08
Alloy steels	0·25 Cr 0·25 Ni 1·0 Mn	285	372	0·1 C steel 18 Cr 8 Ni steel	294 264	1·62 1·80
	1·3 Cr 2·6 Ni 0·4 Mo	217	304	3 Si steel	241	1·19
	1·1 Cr 3·7 Ni 0·4 Mo	176	272	18 Cr 8 Ni steel aluminium	212 238	1·28 1·14
	0·6 Cr 2·5 Ni 0·5 Mo	330	542	0·6 Cr 2·5 Ni 0·5 Mo	124	4·14
	1·4 Cr 4·0 Ni 0·3 Mo	510	850	1·4 Cr 4·0 Ni 0·3 Mo	240	3·55
Aluminium alloys	Al-Cu-Mg	—	276	Al-Cu-Mg	99	2·70
	Al 4·4 Cu 0·5 Mn 1·5 Mg	140	159	Al 4·4 Cu 0·5 Mn 1·5 Mg	82	1·92
	Al 4·4 Cu 0·8 Mn 0·7 Mg	160	134	Al 4·4 Cu 0·8 Mn 0·7 Mg mild steel	49·5* 35·6*	2·72 3·78
	Al 4 Cu	117	83·5	Al 4 Cu	52·5	1·59
Copper alloys	70/30 brass	175	139	70/30 brass	93	1·50

* 193 MN m^{-2} mean stress

Table 4.4: Strength Reduction Factors Produced by the Fretting of Various Materials against Steels and Aluminum Alloys

From [11].

Rotating bending fatigue tests at 3500 cycles/min

Material	Condition	Tensile strength tons/in²	Hardness Rockwell B	Normal fatigue strength (10^7 cycles) tons/in²	Fretting fatigue strength (10^7 cycles) tons/in²
0·35% C steel	Annealed		65		8·5
	Cold worked	47	95		9·5
	Normalized		75		9
18% Cr 9% Ni Stainless steel	Annealed	47·8	89		12·5
	Cold worked	77·8			10·5
19% Cr 11% Ni Stainless steel	Annealed		77		11
	Cold worked		95		6
Aluminium Bronze (4% Al 7% Zn)	Annealed	34	77		6·5
	Cold worked	48	94		6
Duralumin (4% Cu)	Annealed	11·6			4·5
	Aged 1 hr at 175 C		61		4
	Water-quenched from 500 C		56		4
0·24% C steel	Forged and normalized	36		18	9
	Cast and normalized	34		15	11
Aluminium alloy (4·5% Cu)	Forged and heat-treated	24·5			4·5
Aluminium alloy (4% Cu)	Sand cast and heat-treated	14		3·5	3
Aluminium alloy (4·5% Cu)	Chill cast and heat-treated	16		5·5	4·5

Table 4.5: Fretting Fatigue Strengths of Various Metals

From [12].

4.2 Equivalent Stresses

In Chapter 3, the stress states of the lug/pin joint components for the various loading conditions were found. In Section 4.1, it was explained how the working $S-N$ diagram could be determined. To find the fatigue life of the components, the stress states must now be related to the working $S-N$ diagrams.

The concept of "equivalent stresses" is introduced here in order to accomplish this. Equivalent stresses are the values of alternating and mean stresses, S_a and S_m, that represent the fatigue loading and can be used in the working $S-N$ diagram to evaluate the life.

Notice that, for simple geometries and loading conditions, the concept of "equivalent stresses" is not used. The concept of "fatigue notch factors" denoted by K_f is used instead to take into account the stress raisers effect. Most of the time, K_f is introduced in the computation by reducing the acceptable alternating stress of the working $S-N$ curve. Then the nominal value of alternating stress is used in the $S-N$ diagram to find the component endurance.

In the subsection below, it will be shown that, for the critical components of the lug/pin joint under study, the fatigue notch factors can be safely assumed equal to the stress concentration factors and, therefore, the stress states computed in Chapter 3 can be used as inputs to the computation of the equivalent stresses necessary to perform the fatigue analysis.

4.2.1 Stress Concentration Factors and Fatigue Notch Factors

In a linear static analysis, the actual stress at a geometric irregularity is given by:

$$\sigma = K_t S, \tag{4.13}$$

where:

- S is the nominal stress,
- σ is the actual stress, and
- K_t is the stress concentration factor.

The stress concentration factor depends only on the component geometry and the loading conditions. It can be computed using the theory of elasticity or the finite element method. [17] gives the stress concentration factors corresponding to various geometries and loading conditions. In a static analysis, to avoid plastic deformations, the nominal stress "S" applied on a component must always be such that:

$$S < \frac{S_y}{K_t}, \tag{4.14}$$

where S_y is the yield-strength of the material in question.

Similarly, in fatigue analyses, the alternating stress "S_a" that can be applied to a component must be such that:

$$S_a < \frac{S_f}{K_f}, \tag{4.15}$$

where K_f is the high cycle ($N \geq 10^7$) fatigue notch factor. K_f is defined as the ratio of the notched specimen nominal fatigue strength to the unnotched specimen fatigue strength. For a given geometry, K_f is never greater then K_t because K_f takes into account the micro-plastic deformations that occur at the notch root.

Various equations have been established to relate K_f to K_t, the most common being as follows.

1.
$$K_f = q(K_t - 1) - 1, \tag{4.16}$$

   where q is the "notch sensitivity". The notch sensitivity is a function of the component geometry, the loading condition, the material and varies from 0 to 1.

2.
$$K_f = 1 + \frac{K_t - 1}{1 + \sqrt{\frac{a}{r}}}, \tag{4.17}$$

   where:

   - r is the notch root radius and
   - a is a material property.

3.
$$K_f = 1 + \frac{K_t - 1}{1 + \frac{\rho}{r}}, \tag{4.18}$$

   where ρ is again a material property.

The following equation has been proposed [9] to estimate "ρ" (in *inches*) for steels:

$$\rho = 10^{-3}\left(\frac{3.0E5}{S_u}\right)^{1.8}, \tag{4.19}$$

where S_u is the tensile strength, [psi]. Table 4.6 shows some values of ρ and a.

Steel	$S_u = 345MPa$ $S_u = 1750MPa$	$\rho = 10mm$ $\rho = 0.03mm$
Aluminum 7075 − $T6$	plane stress plane strain	$\rho = 0.02mm$ $\rho = 0.003mm$
Steel	$S_u = 500MPa$ $S_u = 2000MPa$	$a = 0.25mm$ $a = 0.0002mm$
Aluminum 7075 − $T6$	$S_u = 150MPa$ $S_u = 600MPa$	$a = 2.0mm$ $a = 0.4mm$

Table 4.6: Some values of ρ and a

Fatigue Notch Factor for the Lugs. Using Table 4.6, with $\rho = 0.02mm = 8.0E-4in$ and the "r" value of eqn(4.18) being equal to the lug hole radius of $0.6in$, it follows that:

$$K_f = 0.999K_t + 0.001. \tag{4.20}$$

Since $K_f \approx K_t$, the stresses computed in Chapter 3 can be applied directly to evaluate the fatigue equivalent stresses.

Fatigue Notch Factor for the Pin. The only notch on the pin is at the pin head (see Figure 4.12). This notch has not been studied since cracking at this location is not expected. If the pin had been torqued up tightly, a more careful study of this area would have been warranted. The pin nut has also been neglected in the current investigation.

Fatigue Notch Factor at the Bushings. The notch radius at the flange root (see Figure 4.13) is small and plasticity in this area may be important. However, neglecting the plasticity, i.e., assuming $K_f = K_t$ can only be conservative.

It will be shown in Section 4.6 that the bushings are not critical in the current lug/pin joint configuration.

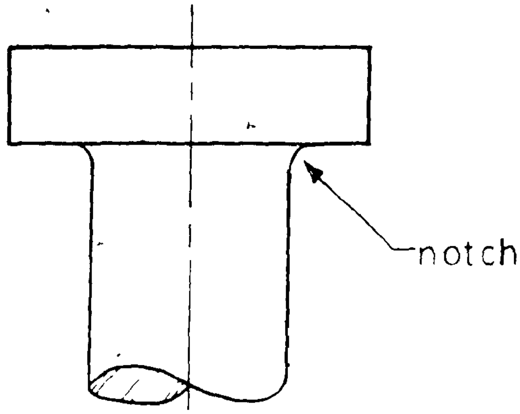


Figure 4.12. Notch at the Pin Head

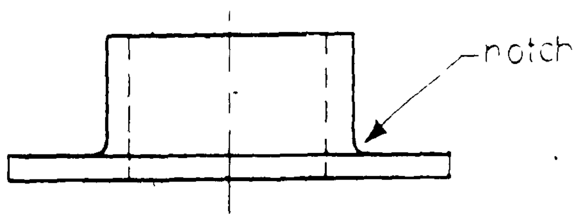


Figure 4 13: Notch at the Bushing Flange Root

The notch root radius is 0.050*in*.

4.2.2 Multiaxial stress state in fatigue

In this Section, some methods that relate the multiaxial stress states resulting from the application of oscillating loading conditions to the equivalent stresses are explained. Notice that there is no universally accepted method for the computation of multiaxial fatigue. Garud [18] has counted 24 methods that have been proposed to put multiaxial fatigue into equations. Only 2 methods that are relatively easy to use are explained here.

Computation of Equivalent Stresses for a Stress State having fixed Principal Axes

[9] has proposed the following criterion for fatigue crack initiation:

$$S_a + \frac{m}{\sqrt{2}} S_m > S_N, \qquad (4.21)$$

where:

$$S_a = \frac{K_f}{\sqrt{2}} \left[(S_{a_1} - S_{a_2})^2 + (S_{a_2} - S_{a_3})^2 + (S_{a_3} - S_{a_1})^2 \right]^{\frac{1}{2}}, \qquad (4.22)$$

$$S_m = K_f (S_{m_1} + S_{m_2} + S_{m_3}), \qquad (4.23)$$

where:

- S_m and S_a are the "equivalent" mean and alternating stresses, respectively,
- S_{a_i} and S_{m_i} are the nominal alternating and mean stress components of the principal stresses in the direction "i" ($i = 1, 2, 3$),
- S_N is the fatigue strength at life N,
- K_f is the fatigue notch factor, and
- m is the coefficient of influence of the mean stress given in eqn(4.8)

If the fatigue notch factors differ too greatly for different principal directions, different factors K_{f_i} ; $i = 1, 2, 3$ must be used [19,20] and K_f must be removed. Eqns(4.22 & 4.23) then become:

$$S_a = \frac{1}{\sqrt{2}} \left[(K_{f_1} S_{a_1} - K_{f_2} S_{a_2})^2 + \cdots \right]^{\frac{1}{2}}, \qquad (4.24)$$

$$S_m = K_{f_1} S_{m_1} + K_{f_2} S_{m_2} + K_{f_3} S_{m_3}. \qquad (4.25)$$

If K_{f_i} cannot be determined, the actual stresses; σ_{a_i} , σ_{m_i} computed using linear elasticity can be used instead of $K_{f_i}S_{a_i}$ and $K_{f_i}S_{m_i}$. This is equivalent to making the conservative assumption stating that $K_f = K_t$.

Eqn(4.21) is the equivalent of eqn(4.8) for multiaxial loadings. Since eqn(4.21) is valid for narrow ranges of S_m, it has been decided to use the parabola given in eqn(4.10).

For biaxial stress states, the crack initiation criterion eqn(4.23) is plotted on a S_2, S_1 graph as an ellipse centered at the mean stress. Figure 4.14 shows an example.

Computation of Equivalent Stresses for Stress States having moving Principal Axes

Various methods have been proposed to study fatigue when the principal axes change orientation as the component is submitted to oscillating loads [18]. The following method that has been adopted in the present investigation has been proposed by Langer [21] and modified by Fuchs [19,22].

Two "extreme" instants t_1 , t_2 during the fatigue cycle are chosen. The 6 stress tensor components computed at these two instants are designated by:

$$\sigma_{xx_1} \quad \sigma_{yy_1} \quad \sigma_{zz_1} \qquad \tau_{xy_1} \quad \tau_{yz_1} \quad \tau_{zx_1}, \tag{4.26}$$

$$\sigma_{xx_2} \quad \sigma_{yy_2} \quad \sigma_{zz_2} \qquad \tau_{xy_2} \quad \tau_{yz_2} \quad \tau_{zx_2}, \tag{4.27}$$

From these the differences D_{ij} can be computed:

$$D_{ij} = \sigma_{ij_1} - \sigma_{ij_2}, \tag{4.28}$$

which again form a symmetric second order tensor.

At this point, Langer proposed to compute the equivalent alternating stress as:

$$S_a = \frac{P_1 - P_3}{2}, \tag{4.29}$$

where P_1 and P_3 are the maximum and minimum principal values of the D_{ij} tensor.

However, in [19], it is claimed that a von Mises like method given by eqn(4.30) gives better results than Langer's.

$$S_a = \frac{1}{2\sqrt{2}}\left[(D_{xx} - D_{yy})^2 + (D_{xx} - D_{zz})^2 + (D_{yy} - D_{zz})^2 + 6(D_{xy}^2 + D_{xz}^2 + D_{zy}^2)\right]^{\frac{1}{2}}. \tag{4.30}$$

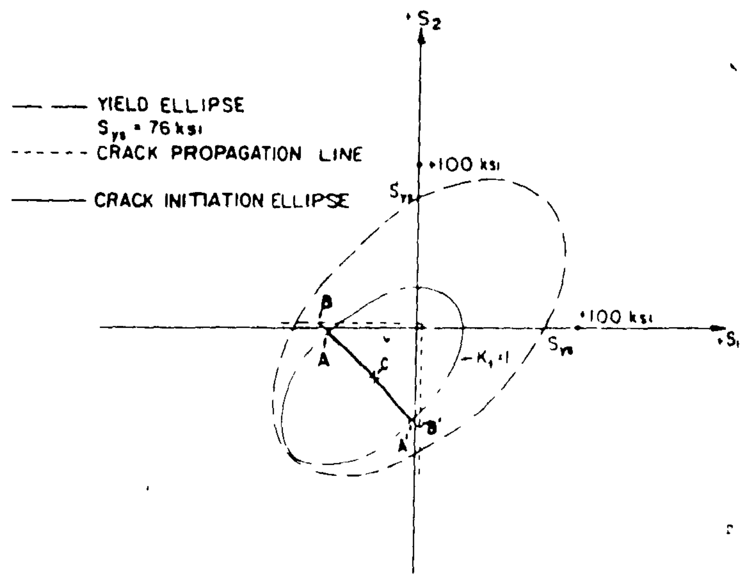


Figure 4.14: Crack Initiation Ellipse

This figure taken from [9] shows the fatigue behavior of an unnotched shot-peened torsion bar. Shot-peening induces a tangential mean stress shown by the point C. When submitted to alternating torsion, the stress state $S_1(t), S_2(t)$ may oscillate between points A and A' before crack initiation. To get crack propagation (see Section 4.3), it must oscillate between B and B'. The "Yield Ellipse" shows the von Mises yielding criterion.

In [22], it is proposed to compute S_a the following way:

$$S_a = \frac{b}{2t}(P_1 - P_3) + (1 - \frac{b}{2t})(P_1 + P_3), \tag{4.31}$$

where b and t are respectively the fatigue stength in bending and torsion. For ductile metals such as steel or aluminum, $t \approx 0.6b$ and eqn(4.31) becomes:

$$S_a = \frac{P_1}{2} - \frac{P_3}{3}. \tag{4.32}$$

In the ENDUR program (Appendix L), the user can choose either eqn(4.30) or eqn(4.32) to evaluate the equivalent alternating stress. As Figures 4.15 and 4.16 show, it does not do much difference. Notice that if the principal directions are not moving, eqn(4.30) is the same as that explained in Section 4.2.2.

The equivalent mean stress is set equal to the average, over a cycle, of the first stress tensor invariant [18]. If only the 2 extreme stress states are known, this average can be approximated as:

$$S_m = \frac{1}{2}\left(\sigma_{xx_1} + \sigma_{yy_1} + \sigma_{zz_1} + \sigma_{xx_2} + \sigma_{yy_2} + \sigma_{zz_2}\right). \tag{4.33}$$

Care must be taken in determining the "extreme" loading conditions. For example, Figure 4.17 taken from [23] shows that, when the external load does one cycle, the critical area is submitted to 2 cycles, one having a large amplitude, the other having a small amplitude. For the lug/pin joint under study though, the lugs always stay in tension (Figures 3.37 to 3.40), therefore the "extreme" stress states correspond to the "extreme" loading conditions.

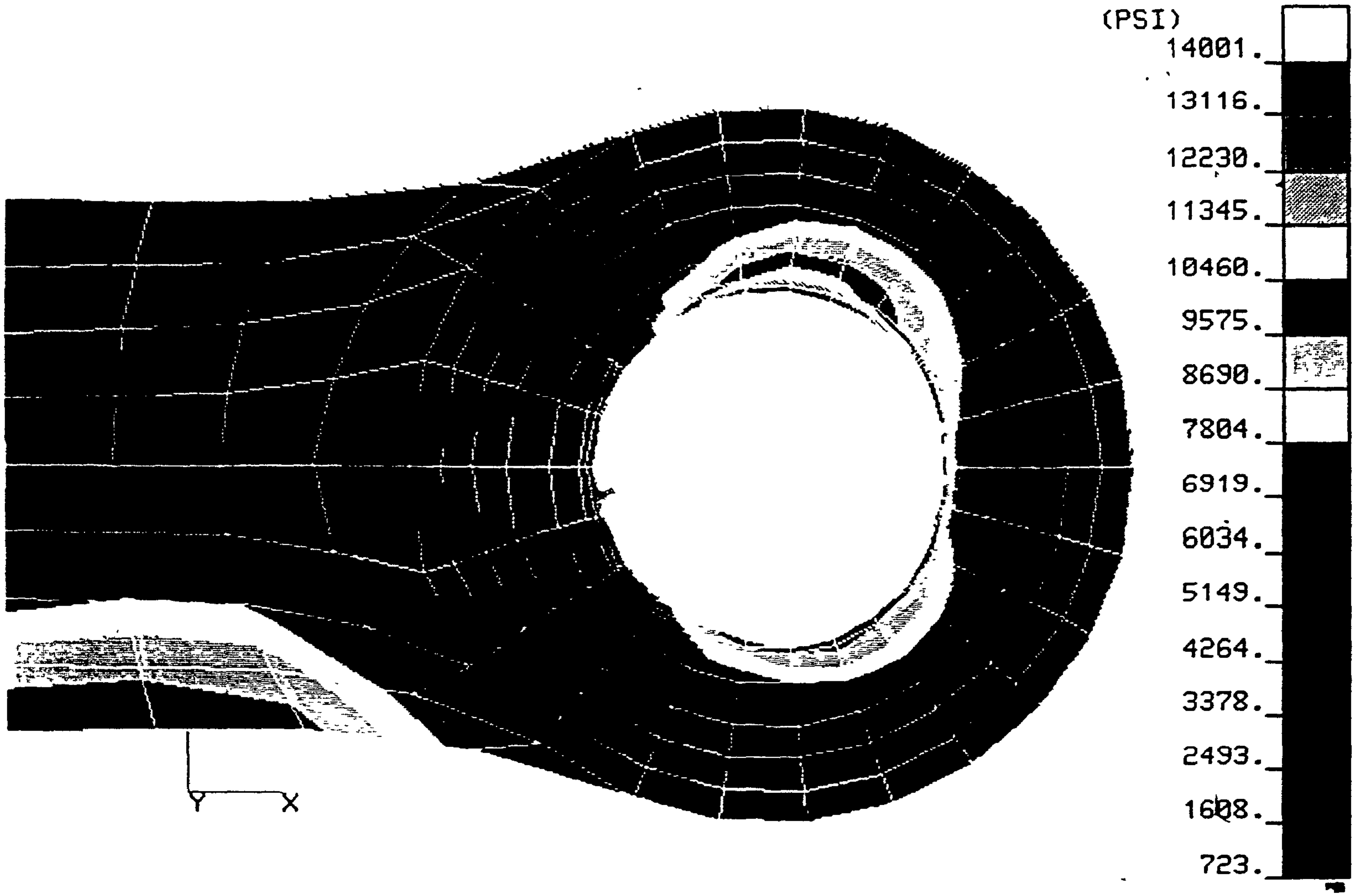


Figure 4.15: Equivalent alternating stress under "high amplitude cycle" conditions, computed using the von Mises like equation

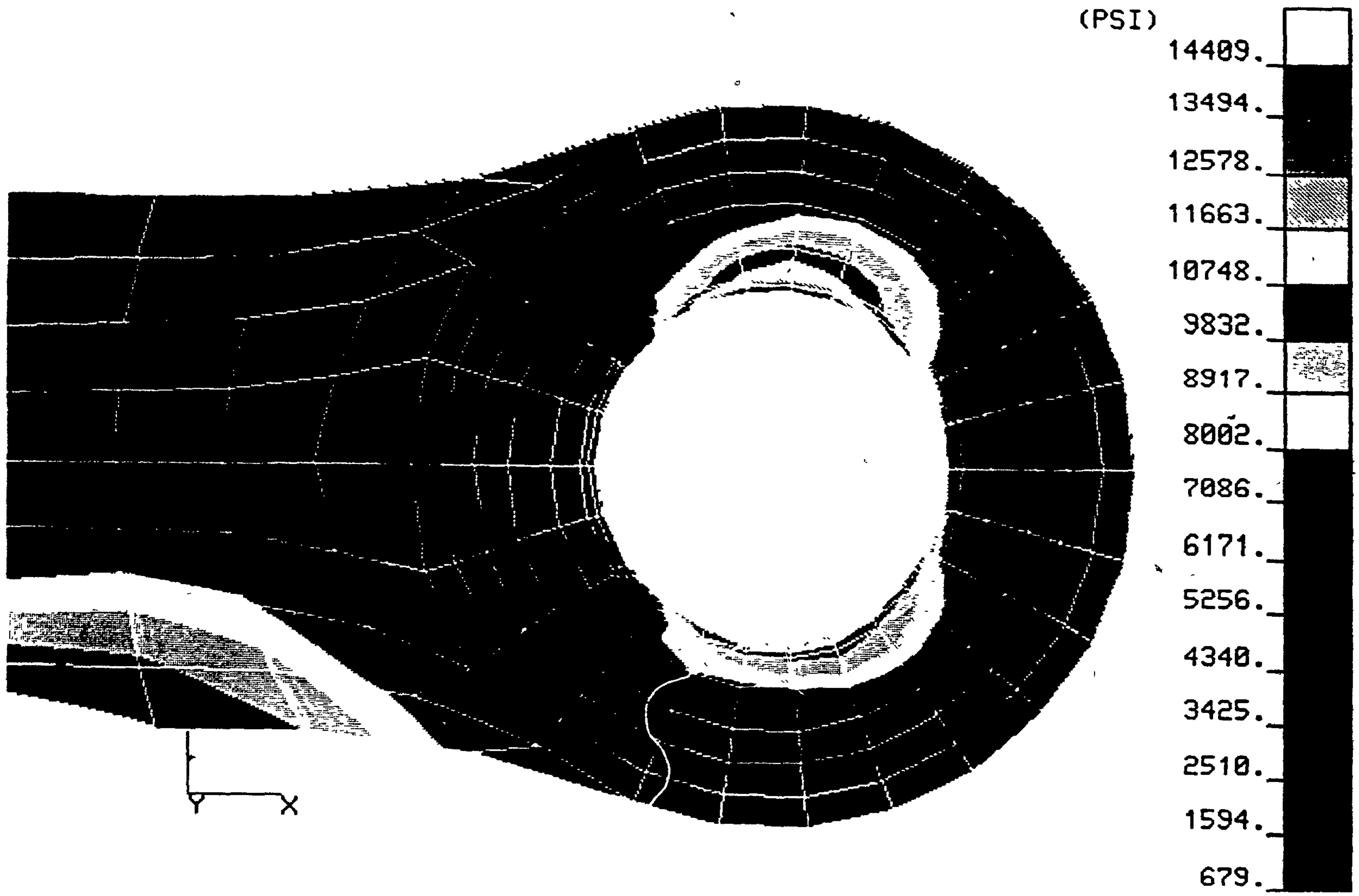


Figure 4.16: Equivalent alternating stress under "high amplitude cycle" conditions, computed using $S_a = \frac{P_1}{2} - \frac{P_3}{3}$

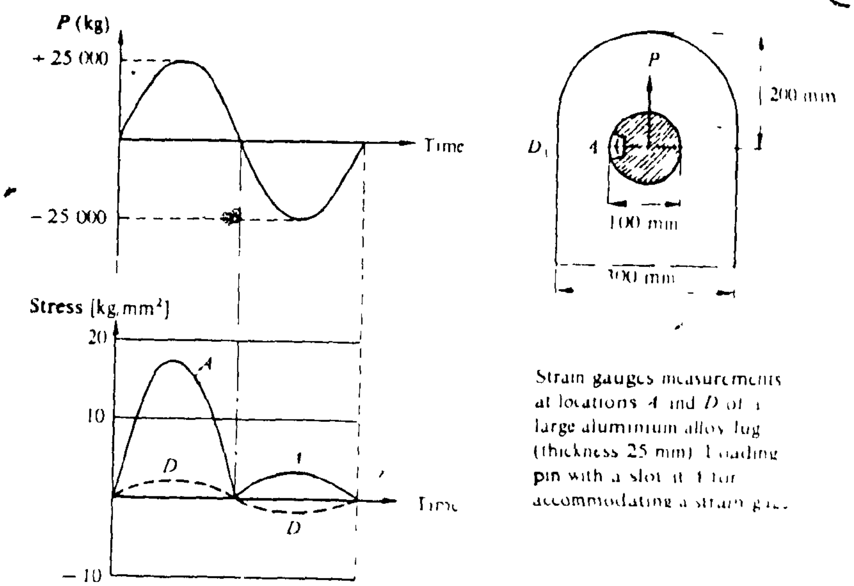


Figure 4.17: Comparison of the stress state at the critical zone to the oscillating loading condition

When the external load does one cycle, the critical area is submitted to 2 cycles, one having a large amplitude, the other having a small amplitude.

In the lug/pin joint under study though, the lugs always stay in tension over a loading cycle and, therefore, the "extreme" stress states correspond to the "extreme" loading conditions.

4.3 Crack Propagation Criterion

To get a fatigue failure, a crack must *initiate* and *propagate* until the component cannot stand the load it is submitted to, then a "static" fracture occurs. In the previous section, it was explained how equivalent alternating and mean stresses could be determined to set a crack *initiation* criterion. In the present section, a crack *propagation* criteron will be developed.

For uniaxial loading, there will be crack propagation if:

$$S_{at} > S_{cat}, \tag{4.34}$$

where:

- S_{at} is the "*nominal* alternating tensile stress" i.e. $S_{at} = \frac{1}{2}(S_{max\ tensile} - S_{min\ tensile})$ and
- S_{cat} is a material constant.

The definition of S_{at} is illustrated in Figure 4.18. S_{cat} is small (see Table 4.7) and, as suggested in [15], $S_{cat} = 0$ will be used in the present investigation. In [14,15,9], it is mentioned to use the *nominal* instead of the actual local value of S_{at} since the cracks are always arrested at some depth from the surface. [9] also mentioned that the absence of K_f on the right side of eqn(4.34) is only "approximately correct". Actually it is more or less un-conservative.

High-strength Aluminum Alloys	$3ksi$
Mild Steels	$4ksi$
Hardened Steels	$10ksi$

Table 4.7: Approximate values of S_{cat} for various metals

For multiaxial loading the crack propagation criterion becomes eqn(4.35) if the principal directions do not move [9].

$$S_{at_i} > S_{cat}, \tag{4.35}$$

where $S_{at_i} = \frac{1}{2}(S_{max\ tensile_i} - S_{min\ tensile_i})$; $i = 1, 2, 3$.

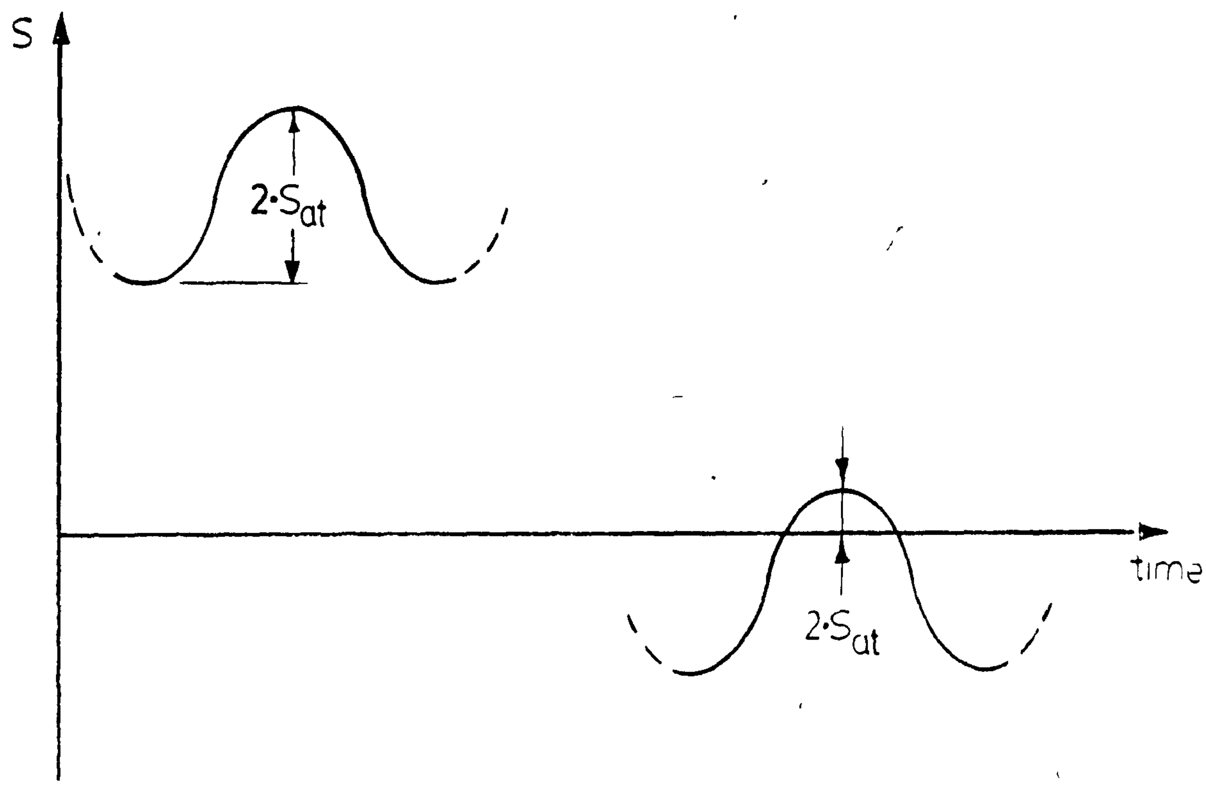


Figure 4.18: S_{at} Definitions

$$
\begin{aligned}
S_{at} &= \frac{1}{2}(S_{max\ tensile} - S_{min\ tensile}) \\
S_{max\ tensile} &= MAX(S_{max}\ ,\ 0.0) \\
S_{min\ tensile} &= MAX(0.0\ ,\ S_{min})
\end{aligned}
$$

Again [14,15,9] propose to use the nominal values of S_{at_i}. If, because of the complexity of geometry under study, the "nominal" values cannot be computed, then the present investigator proposes to use the actual local values. This is conservative.

A survey of the literature on multiaxial fatigue has not permitted to find any suitable fatigue crack propagation criterion if the principal directions are moving. The present investigator proposes to use the following criterion:

$$P_{1_1} > 0. \text{ and } P_{1_2} > 0, \tag{4.36}$$

where:

- P_{1_1} is the maximum principal stress at instant t_1 and
- P_{1_2} is the maximum principal stress at instant t_2.

the instants t_1 and t_2 corresponding to the two "extreme" stress states. Here the *actual local* values of the stress are used to compute P_{1_1} and P_{1_2}.

In the ENDUR program (Appendix L), eqn(4.36) is used for every F.E. node to check whether crack propagation at this point is possible. If it is not, the life at this node is considered infinite. If crack propagation is possible, then the equivalent stresses are computed and the appropriate working $S-N$ curve is used to evaluate the endurance.

4.4 Cumulative Damage

Components are rarely submitted to constant amplitude loading. In high cycle fatigue, the Palmgren-Miner rule is widely used to evaluate the life of components submitted to a variable amplitude loading from the knowledge of their fatigue behaviors under constant amplitude loading. This Palmgren-Miner rule states that the total fatigue damage "D" a component has been subjected to is:

$$D = \sum \frac{n_i}{N_i}, \tag{4.37}$$

and that failure will occur when:

$$D > 1. \tag{4.38}$$

In the above equations:

- N_i is the number of cycles to failure for the stress level "i" determined from the $S - N$ curve and
- n_i is the number of cycles occurring at stress level "i".

This rule should be seen as a reasonable "rule of thumb". Other rules have been devised but they are harder to apply and have not proved to be superior if sequence effect can be neglected. Sequence effect is assumed to be negligible in the present investigation.

From eqns(4.37 & 4.38), the life "N" of a component will be:

$$N = \frac{1}{\sum \frac{x_i}{N_i}}, \tag{4.39}$$

where x_i is the relative frequency of occurrence of a stress cycle at the stress level "i".

The MINER program (Appendix N) uses the Palmgren-Miner rule to evaluate the endurance of any component submitted to a variable amplitude loading. The lug/pin joint under study is submitted to a loading spectrum with 2 types of oscillations (section 2.3) and the MINER program was used to find the life of the lugs of the lug/pin joint being considered in this investigation (see Figure 4.26).æ

4.5 Reliability

There are lots of uncertainty associated with fatigue and a probabilistic approach in its study is very enlightening. In fatigue, the reliability of a component is its probability of survival. There are several ways to look at the reliability of components subjected to fatigue loading situations. For example, one can look for:

1. the probability of failure after a given number of cycles N, the stress amplitude S_a being known; or

2. the probability of failure at a given alternating stress S_a , N being fixed.

Figure 4.19 illustrates these two concepts.

The second method cannot be applied if the applied load S_{a_2} of Figure 4.19 is below the fatigue limit. In this case, it may be important to apply the first method because the applied load could be only slightly below the fatigue limit and the probability of failure unacceptably high (see Figure 4.20).

4.5.1 Computing the Reliability as a Function of the Stress Amplitude

Usually published $S-N$ diagrams represent a 50 % rate of survival. To obtain higher reliability, the applied alternating stress must be lower than the fatigue strength. One way of expressing the reliability of a component is simply to define the reliability factor K_c as:

$$K_c = \frac{\text{applied alternating stress}}{\text{fatigue strength}}. \tag{4.40}$$

Assuming that the fatigue strength at a given life is normally distributed:

$$K_c = 1 - \delta Z_r, \tag{4.41}$$

where:

- δ is the standard deviation of the strength distribution (given as a fraction of the fatigue strength) and

- Z_r is the standardized normal variable, function of the reliability.

At the fatigue limit, δ can be conservatively assumed as 0.08 if no accurate data is available. Table 4.8 shows the reliability factor "K_c" as a function of the reliability.

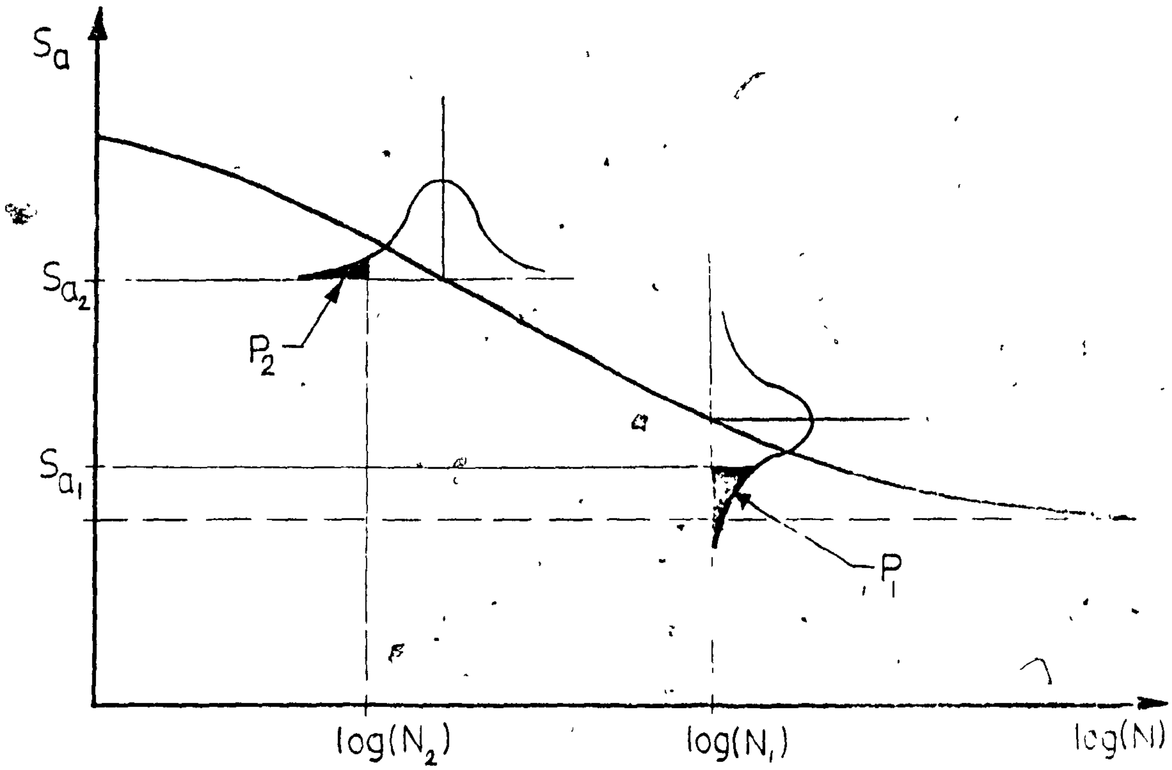


Figure 4.19: Reliability

- P_1 is the probability of failure as a function of the stress amplitude S_{a_1}, the life N_1 being kept constant.
- P_2 is the probability of failure as a function of N_2, the stress amplitude S_{a_2} being kept constant.

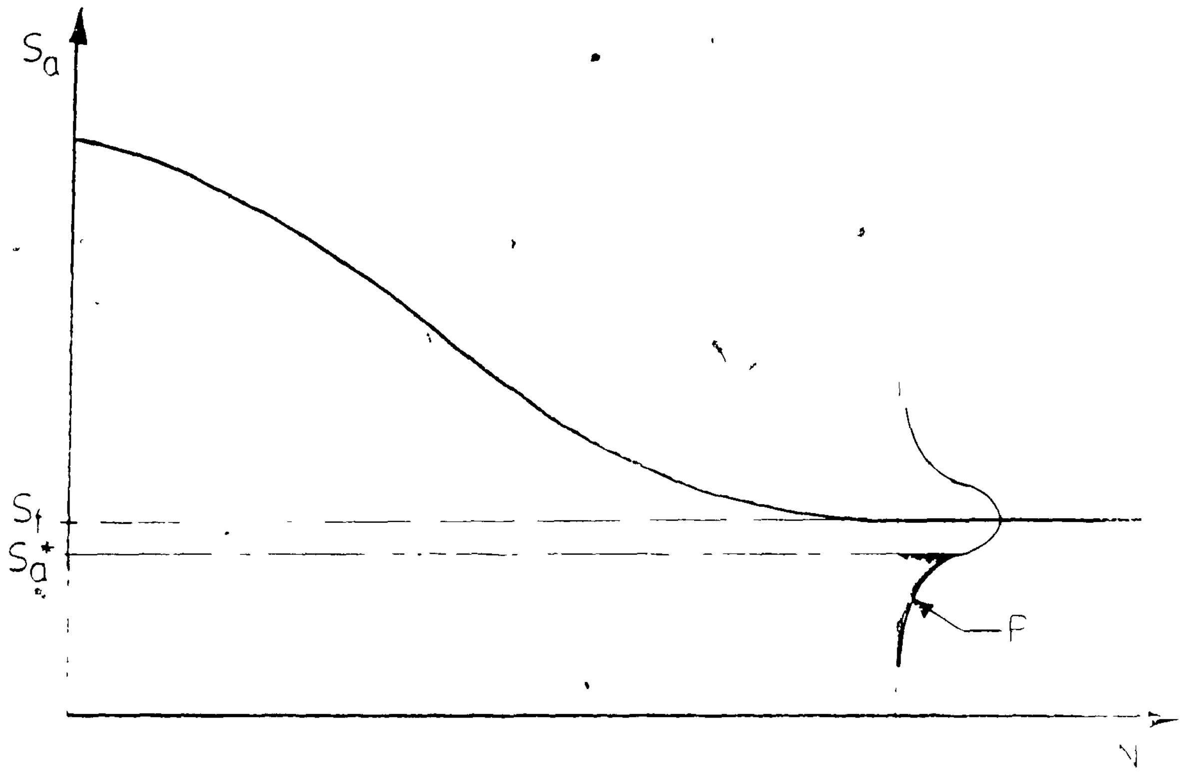


Figure 4.20 Unacceptable Probability of Fatigue failure

Even if the applied alternating stress S_a^* is lower than the fatigue limit S_f and, therefore, a infinite life is computed, the component has an unacceptably high probability of failure.

Reliability	Z_r	Reliability factor K_c
0.50	0.000	1.000
0.90	1.288	0.897
0.95	1.645	0.868
0.99	2.326	0.814
0 999	3.091	0.753
0 999 9	3.719	0.702
0.999 99	4 265	0.659
0.999 999	4 753	0.620
0.999 999 9	5.199	0.584
0 999 999 99	5.612	0.551
0 999 999 999	5.997	0 520

Table 4 8 Reliability Factors Corresponding to an 8% Standard Deviation of the Endurance Limit

From [24]

4.5.2 Computing the Reliability as a Function of the Endurance

The method, proposed in [25,26], is a modified version of the Palmgren-Miner rule and permits for the evaluation of the reliability as a function of the endurance. Instead of stating that failure occurs when the total damage is one (see Section 4.4), it is stated that there is a probability of failure associated with the total damage. The damage at failure can therefore be seen as a random variable that will be denoted by Δ.

Usually, there are not enough data to define the probability distribution at very low levels of probability of failure. 10^{-6} or 10^{-7} are typical desired probabilities of failure. An assumption on the probability law has to be made. Two distributions, the "log-normal" and the "Weibull", are widely used in probabilistic fatigue analysis. In the present investigation, the behavior of materials under fatigue is assumed to be such that the random variable Δ is "log-normally" distributed. Because of the assumption on the probability distribution law, the "probability" computed should be seen more in a comparative than a classical relative frequency sense.

Bearing in mind the above considerations, the reliability can be computed using the following equations [25,26]:

$$\sigma_z^2 = \ln(1 + V_\Delta), \tag{4.42}$$

$$\mu_z = \ln \bar{\Delta} - \frac{1}{2}\ln(1 + V_\Delta), \tag{4.43}$$

$$R = 1 - \Phi\left(\frac{\ln \Delta - \mu_z}{\sigma_z}\right), \tag{4.44}$$

where:

- V_Δ is the coefficient of variation of Δ, namely:

$$V_\Delta = \frac{\sigma_\Delta}{\bar{\Delta}}, \tag{4.45}$$

- $\bar{\Delta}$ is the mean value of Δ,
- $\Phi(z_0)$ is the cumulative probability for a normal distribution for $z < z_0$, and
- R in the Reliability.

If possible, both $\bar{\Delta}$ and V_Δ should be experimentally found for a given design. If not, the following values can be used:

For steels, in the high cycle fatigue range:

$$\bar{\Delta} \simeq 1.37 \ ; \ V_\Delta \simeq 0.638. \tag{4.46}$$

For aluminums, in the high cycle fatigue range:

$$\bar{\Delta} \simeq 1.33 \ ; \ V_\Delta \simeq 0.65. \tag{4.47}$$

If $\bar{\Delta}$ and V_Δ are not known, $\bar{\Delta} = 1.0$ and $V_\Delta = 0.65$ are reasonably conservative assumptions.

Tables 4.9 to 4.11 give the cumulative damage Δ for various values of $\bar{\Delta}$ and V_Δ as a function of the reliability.

Example 1 : For an aluminum specimen, the reliability when the cumulative damage is 0.5 can be computed as follows.

- It is known that $\bar{\Delta} \simeq 1.33 \ ; \ V_\Delta \simeq 0.65$.
- From eqns(4.40 & 4.41) σ_x and μ_x may be computed as $\sigma_x = 0.5936$ and $\mu_x = 0.10897$.
- Since $\Delta = 0.5$ then:

$$R = 1 - \Phi\left(\frac{\ln 0.5 - 0.10897}{0.5936}\right) \ or$$
$$R = 1 - \Phi(-1.35).$$

From a normal distribution table : $\Phi(-1.35) = 0.0885 \Rightarrow$
The Reliability "R" is: 91%.

Example 2 : For aluminum components, the cumulative damage for a 99.9 % reliability can be computed as follows:

$$\Phi\left(\frac{\ln \Delta - 0.10897}{0.5936}\right) = 1 - 0.999 = 0.001.$$

By looking at a normal distribution table (Table 4.12):

$$\frac{\ln \Delta - 0.10897}{0.5936} = -3.09 \ \Rightarrow \ \Delta = 0.178,$$

i.e., when the cumulative damage Δ has reached 0.178, there is a cumulative probability of failure of 0.1 %. In other words, when the component has been in service for 17.8 % of its expected life (according to the Palmgren-Miner rule), there is a cumulative probability of failure of 0.1%.

Reliability %	Δ
99.99999	0.051
99.9999	0.066
99.999	0.088
99.99	0.12
99.9	0.18
99.	0.28
95.	0.42
90.	0.52
50.	1.12

Table 4.9 Reliability for Aluminum Alloy Components with $\bar{\Delta} = 1.33$ and $V_\Delta = 0.65$

Reliability %	Δ
99.99999	0.055
99.9999	0.072
99.999	0.095
99.99	0.13
99.9	0.19
99.	0.30
95.	0.44
90.	0.54
50.	1.15

Table 4.10: Reliability for Steel Components with $\bar{\Delta} = 1.37$ and $V_\Delta = 0.638$

Reliability %	Δ
99.99999	0.038
99.9999	0.050
99.999	0.066
99.99	0.092
99.9	0.13
99.	0.21
95.	0.32
90.	0.39
50.	0.82

Table 4 11. Reliability for $\bar{\Delta} = 1\ 0$ and $V_\Delta = 0.65$

Probability of Failure	Reliability %	Z
1.0E-7	99.99999	-5.20
1.0E-6	99.9999	-4.75
1.0E-5	99.999	-4.27
1.0E-4	99.99	-3.72
0.1 %	99.9	-3.09
1.0 %	99.	-2.326
5.0 %	95.	-1.645
10.0 %	90.	-1.288
50,0 %	50.	0.0

Table 4.12: Normal Distribution Variable Z

4.6 Fatigue Analysis Results

4.6.1 Pin

The stress fields of the pin when the lug/pin joint is submitted to the loading conditions *HL1* and *LL1* have previously been found (Chapter 3).

As explained in Section 2.3, the stress fields corresponding to the loadings *HL2* and *LL2* can be deduced from the knowledge of *HL1* and *LL1* stress fields. The plane of symmetry is the XZ plane. The YMIR program, explained in Appendix M, has been used to find the *HL2* and *LL2* stress fields.

Table 4.13 shows the ENDUR program input file that was used to evaluate the fatigue resistance of the pin when the loading oscillates between *LL1* and *LL2* values. Likewise, Table 4.14 shows the ENDUR program input file that was used to evaluate the fatigue resistance of the pin when the loading oscillates between *HL1* and *HL2* values.

For both cases, the pin has an infinite life. Figure 4.21 shows the reliability at $N = 1.0E7$ if the pin were submitted to "high amplitude cycles" only. (Actually, it is submitted to a "high amplitude cycles" 0.03 % of the time only). The probability of failure would then be $< 1.0E-6$ (see Table 4.8). If a crack does appear, which is very unlikely, it will in the red area.

4.6.2 Bushing

Tables 4.15 and 4.16 give the inputs that were used by the ENDUR program to find the endurance and reliability factor of the bushings and the lug. For both oscillating loading conditions, ("high" and "low" amplitude cycles), the bushing endurance was infinite. Figure 4.22 shows the bushing reliability factor after $N = 1.0E7$ "high amplitude cycles". The probability of failure is $< 1.0E-9$ (see Table 4.8). As expected, the bushing is FAR from being critical.

```
*          -- FILE: [MCGO1.C]FLPIN.DAT --
*          PIN FATIGUE, LOADS:  LL1 & LL2
*          ------------------------------
*                NEUTRAL FILE NAME: [MCGO1.F]PATRAN.OUT
*
*        STRESS STATE #1 FILE NAME: [MCGO1.GEO]S3.NOD
*        STRESS STATE #2 FILE NAME: [MCGO1.GEO]YMIRS3.NOD
*
*   "SA"  COMPUTATION METHOD (1,2): 2
*
*                   NODE ID'S FILE: PINS.DAT
*             SURFACE (1,2,3 OR 0): 1
*         MATERIAL PROPERTIES FILE: S4340FC.DAT
*        SUPERIMPOSED STRESS STATE: 0.  ,0. ,0.
*
*                   NODE ID'S FILE: PINB.DAT
*             SURFACE (1,2,3 OR 0): 1
*         MATERIAL PROPERTIES FILE: S4340FS.DAT
*        SUPERIMPOSED STRESS STATE: 0.  ,0. ,0.
*
*                  NODE. FROM, TO: 1287,    1625
*             SURFACE (1,2,3 OR 0): 0
*         MATERIAL PROPERTIES FILE: S4340.DAT
*        SUPERIMPOSED STRESS STATE: 0., 0. ,0.
```

Table 4.13: ENDUR program input file for Pin Fatigue Evaluation when Loading Alternates between *LL1* and *LL2*

S3.NOD and YMIRS3.NOD are the stress states resulting from loading conditions *LL1* and *LL2*, respectively.

PINS.DAT and PINB.DAT give the pin nodes mating with the sleeve and bushing, while S4340FC.DAT and S4340FS.DAT identify the respective fatigue properties of these nodes. All other nodes (those numbered from 1287 to 1675) have the fatigue properties given by the file S4340.DAT.

```
*     -- FILE: [MCGO1.C]FPIN.DAT --
*     PIN FATIGUE, LOADS: HL1 & HL2
*     -----------------------------
*                NEUTRAL FILE NAME: [MCGO1.F]PATRAN.OUT
*
*       STRESS STATE #1 FILE NAME: [MCGO1.GEO]S4.NOD
*       STRESS STATE #2 FILE NAME: [MCGO1.GEO]YMIRS4.NOD
*
*  "SA"  COMPUTATION METHOD (1,2): 1
*
*                  NODE ID'S FILE: PINS.DAT
*            SURFACE (1,2,3 OR 0): 1
*        MATERIAL PROPERTIES FILE: S4340FC.DAT
*       SUPERIMPOSED STRESS STATE: O.  ,O. ,O.
*
*                  NODE ID'S FILE: PINB.DAT
*            SURFACE (1,2,3 OR 0): 1
*        MATERIAL PROPERTIES FILE: S4340FS.DAT
*       SUPERIMPOSED STRESS STATE: O.  ,O. ,O.
*
*                NODE:   FROM, TO: 1287,     1625
*            SURFACE (1,2,3 OR 0): 0
*        MATERIAL PROPERTIES FILE: S4340.DAT
*       SUPERIMPOSED STRESS STATE: O., O. ,O.
```

Table 4.14: ENDUR program input file for Pin Fatigue Evaluation when Loading Alternates between *HL1* and *HL2*

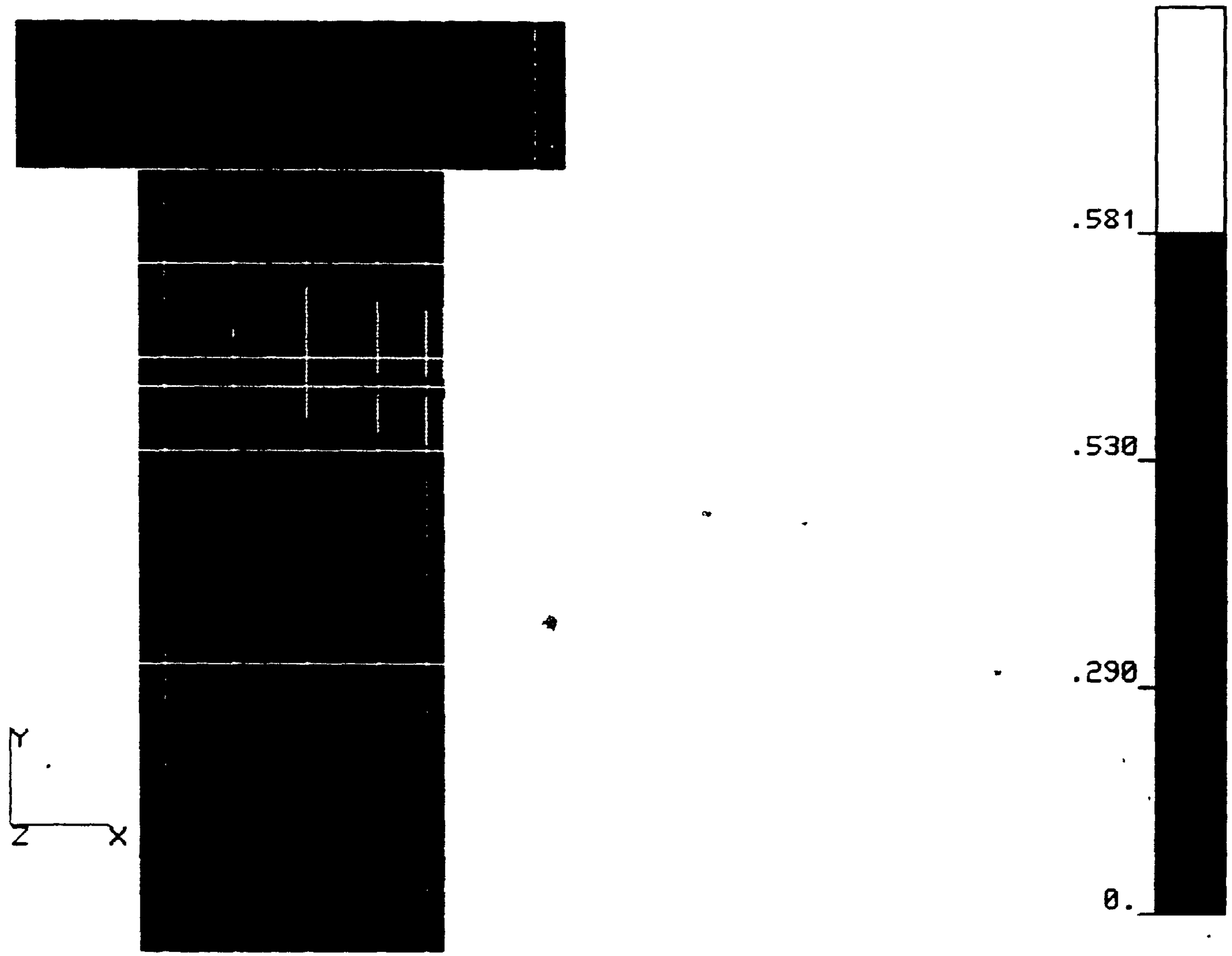


Figure 4.21: Pin Reliability Factor, after 1.E7 "High Amplitude Cycles"

The lower half of the pin is symmetric to the upper half and is not shown.

```
*     -- FILE: [MCGO1.C]FLUG.DAT --
*     LUG FATIGUE, LOADS: LL1 & LL2
*
*                NEUTRAL FILE NAME: [MCGO1.F]PATRAN.OUT
*
*       STRESS STATE #1 FILE NAME: [MCGO1.F]L1.ELS
*       STRESS STATE #2 FILE NAME: [MCGO1.F]L2.ELS
*
*  "SA"  COMPUTATION METHOD (1,2): 2
*
*                  NODE ID'S FILE: HOLE.DAT
*             SURFACE (1,2,3 OR 0): 1
*        MATERIAL PROPERTIES FILE: AL7075F.DAT
*       SUPERIMPOSED STRESS STATE: O.,  -30000.,  -30000.
*
*                  NODE ID'S FILE: FLANGE.DAT
*             SURFACE (1,2,3 OR 0): 3
*        MATERIAL PROPERTIES FILE: AL7075F.DAT
*       SUPERIMPOSED STRESS STATE: O.  ,O. ,O.
*
*                  NODE ID'S FILE: HEAD.DAT
*             SURFACE (1,2,3 OR 0): 3
*        MATERIAL PROPERTIES FILE: AL7075F.DAT
*       SUPERIMPOSED STRESS STATE: O.  ,O. ,O.
*
*                NODE:   FROM, TO: 1, 804
*             SURFACE (1,2,3 OR 0): 0
*        MATERIAL PROPERTIES FILE: AL7075.DAT
*       SUPERIMPOSED STRESS STATE: O., O. ,O.
*
```

Table 4.15: ENDUR program input file for Fatigue Endurance of Lug and Bushing when Loading Alternates between *LL1* and *LL2*

(continued on next page)

```
*                 NODE ID'S FILE: BUSHFZP.DAT
*             SURFACE (1,2,3 OR 0): 3
*       MATERIAL PROPERTIES FILE: SS174PHF.DAT
*      SUPERIMPOSED STRESS STATE: 0.   ,0. ,0.
*
*                 NODE ID'S FILE: BUSHFR.DAT
*             SURFACE (1,2,3 OR 0): 1
*       MATERIAL PROPERTIES FILE: SS174PHF.DAT
*      SUPERIMPOSED STRESS STATE: 0.   ,0. ,0.
*
*              NODES:    FROM, TO: 805, 1932
*             SURFACE (1,2,3 OR 0): 0
*       MATERIAL PROPERTIES FILE: SS174PH.DAT
*      SUPERIMPOSED STRESS STATE: 0., 0  ,0.
```

(continued from previous page)

HOLE.DAT and AL7075F.BAT are the files that give the nodes located on the lug hole wall and their fatigue properties, respectively. The residual stresses induced by shot-peening are $-30ksi$ along the Z' direction and $-30ksi$ along the θ direction.

FLANGE.DAT and HEAD.DAT files contain the lists of nodes where the lug mates with the flange and pin head. Nodes 1 to 804 and *not* included in HOLE.DAT, HEAD.DAT and FLANGE.DAT have the fatigue properties given in AL7075.DAT.

BUSHFZP.DAT contains the bushing flange nodes in contact with washer or lug while BUSHFR.DAT contains the bushing nodes in contact with pin or lug. Their fatigue properties are contained in SS174PHF.DAT.

Finally, nodes numbered from 805 to 1932 are bushing nodes that have the fatigue properties given in SS174PH.DAT, provided they are not described by the above files.

```
*     -- FILE: [MCGO1.C]FLUG.DAT --
*     LUG FATIGUE. LOADS: {\em HL1} & {\em HL2}
*
*                NEUTRAL FILE NAME: [MCGO1.F]PATRAN.OUT
*
*        STRESS STATE #1 FILE NAME: [MCGO1.F]H1.ELS
*        STRESS STATE #2 FILE NAME: [MCGO1.F]H2.ELS
*
*  "SA"  COMPUTATION METHOD (1,2): 2
*
*                  NODE ID'S FILE: HOLE.DAT
*             SURFACE (1,2,3 OR 0): 1
*         MATERIAL PROPERTIES FILE: AL7075F.DAT
*        SUPERIMPOSED STRESS STATE: 0.,  -30000.,  -30000.
*
*                  NODE ID'S FILE: FLANGE.DAT
*             SURFACE (1,2,3 OR 0): 3
*         MATERIAL PROPERTIES FILE: AL7075F.DAT
*        SUPERIMPOSED STRESS STATE: 0.  ,0.  ,0.
*
*                  NODE ID'S FILE: HEAD.DAT
*             SURFACE (1,2,3 OR 0): 3
*         MATERIAL PROPERTIES FILE: AL7075F.DAT
*        SUPERIMPOSED STRESS STATE: 0.  ,0. ,0.
*
*                NODES:   FROM, TO: 1, 804
*             SURFACE (1,2,3 OR 0): 0
*         MATERIAL PROPERTIES FILE: AL7075.DAT
*        SUPERIMPOSED STRESS STATE: 0., 0. ,0.
*
```

Table 4.16: ENDUR program input file for Fatigue Endurance of Lug and Bushing when Loading Alternates between *HL1* and *HL2*

(continued on next page)

```
*                NODE ID'S FILE: BUSHFZP.DAT
*          SURFACE (1,2,3 OR 0): 3
*      MATERIAL PROPERTIES FILE: SS174PHF.DAT
*     SUPERIMPOSED STRESS STATE: 0.  ,0. ,0.
*
*                NODE ID'S FILE: BUSHFR.DAT
*          SURFACE (1,2,3 OR 0): 1
*      MATERIAL PROPERTIES FILE: SS174PHF.DAT
*     SUPERIMPOSED STRESS STATE: 0.  ,0. ,0.
*
*             NODES:   FROM, TO: 805, 1932
*          SURFACE (1,2,3 OR 0): 0
*      MATERIAL PROPERTIES FILE: SS174PH.DAT
*     SUPERIMPOSED STRESS STATE: 0., 0. ,0.
```

(continued from previous page)

Except for the stress states files, it is identical to Table 4.15

4.6.3 Lug

Figure 4.23 shows the lug fatigue endurance when the loading oscillates between *LL1* and *LL2* values ("low amplitude cycles"). Similarly, Figure 4.24 shows the endurance when the loading conditions oscillate between *HL1* and *HL2* values ("high amplitude cycles"). Note that Tables 4.15 and 4.16 again give the inputs that were used in the ENDUR program to generate these figures.

Figure 4.25 shows what would happen if the lug hole walls had not been shot peened. From a comparison of Figures 4.24 and 4.25, it is seen that shot peening increases the fatigue life by a factor of 2.5 .

Figure 4 26 shows the expected endurance when the high amplitude cycles are applied 0.03 % of the time and low amplitude cycles are applied 99.97 % of the time Finally, Figure 4.27 shows the reliability factor at $N = 1\ 0E7$ "high amplitude cycles". Failure before $N = 1.0E7$ is almost certain

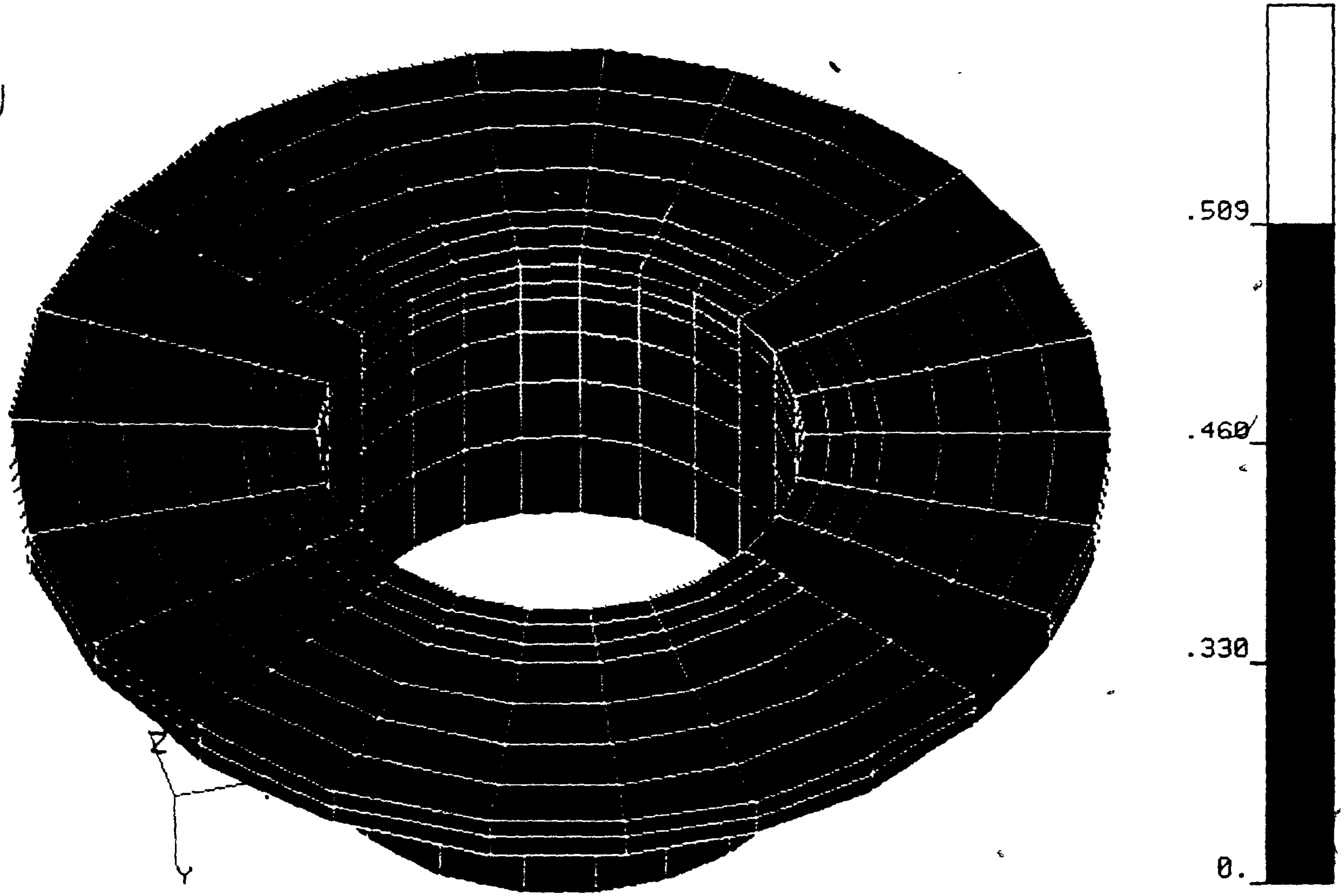


Figure 4.22: Bushing Reliability Factor after 1.$E7$ "High Amplitude Cycles"

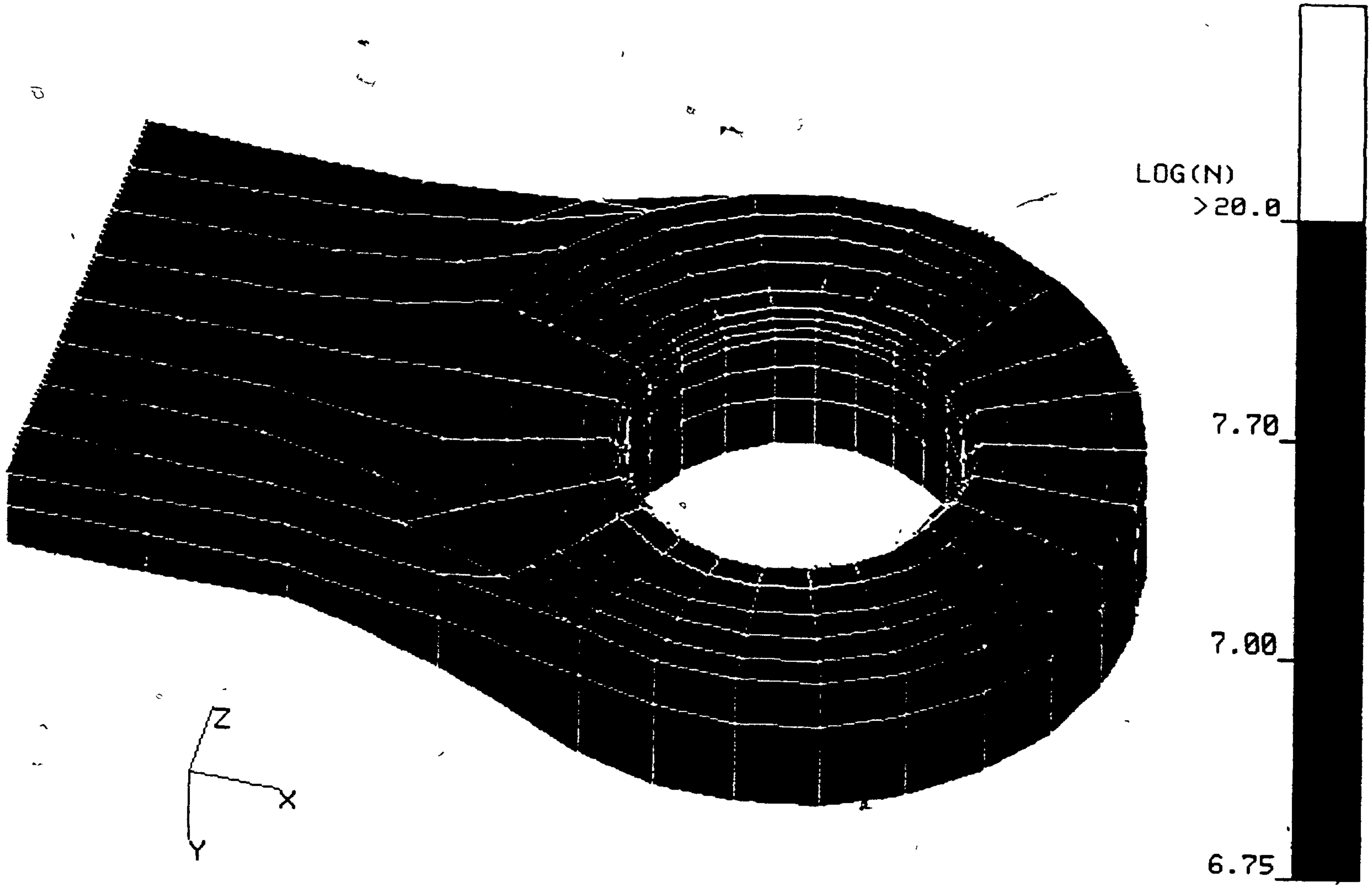


Figure 4.23: "Low Amplitude Cycles" Lug Fatigue Endurance - With Shot-Peening

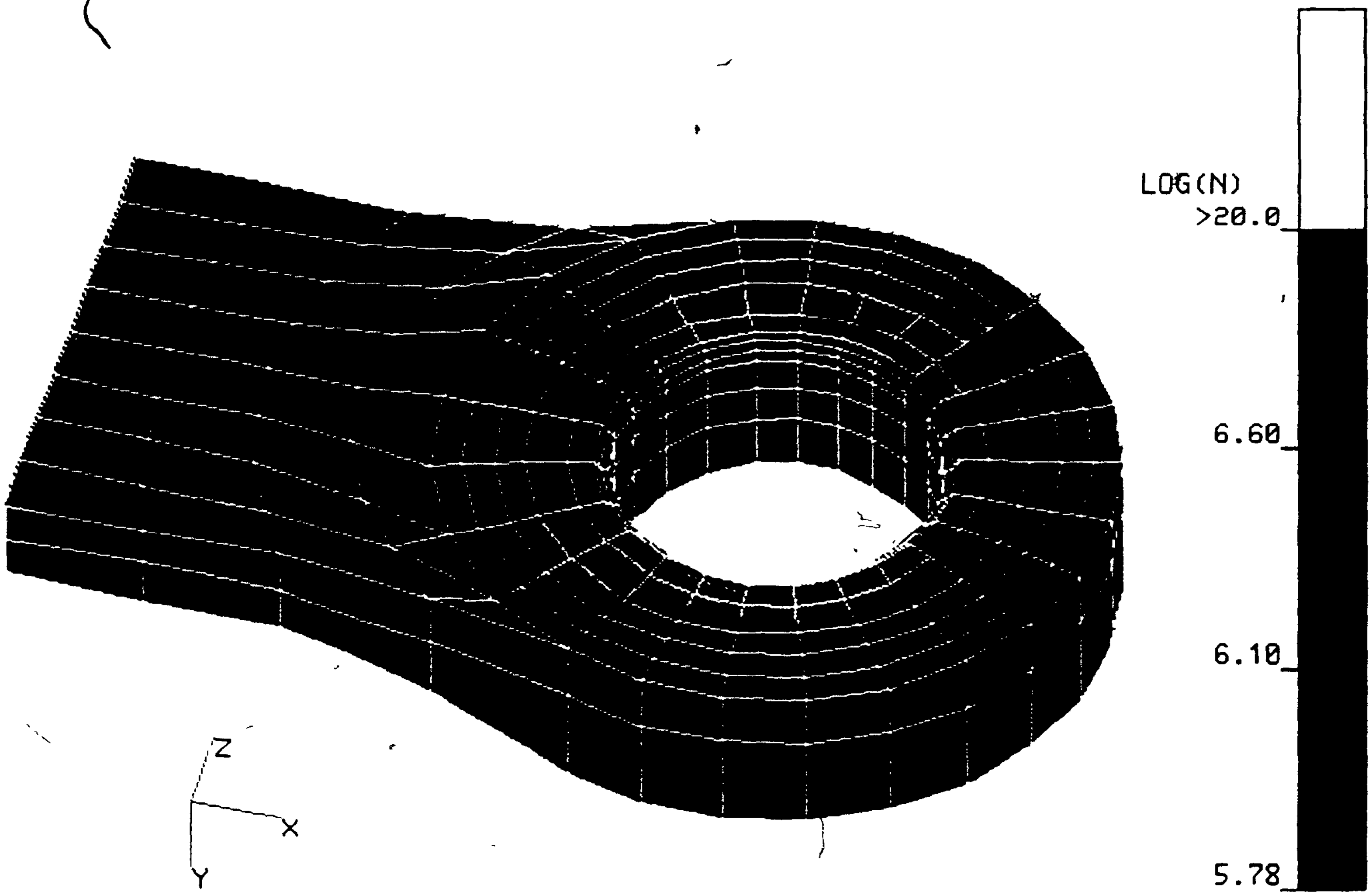


Figure 4.24: "High Amplitude Cycles" Lug Fatigue Endurance - With Shot-Peening

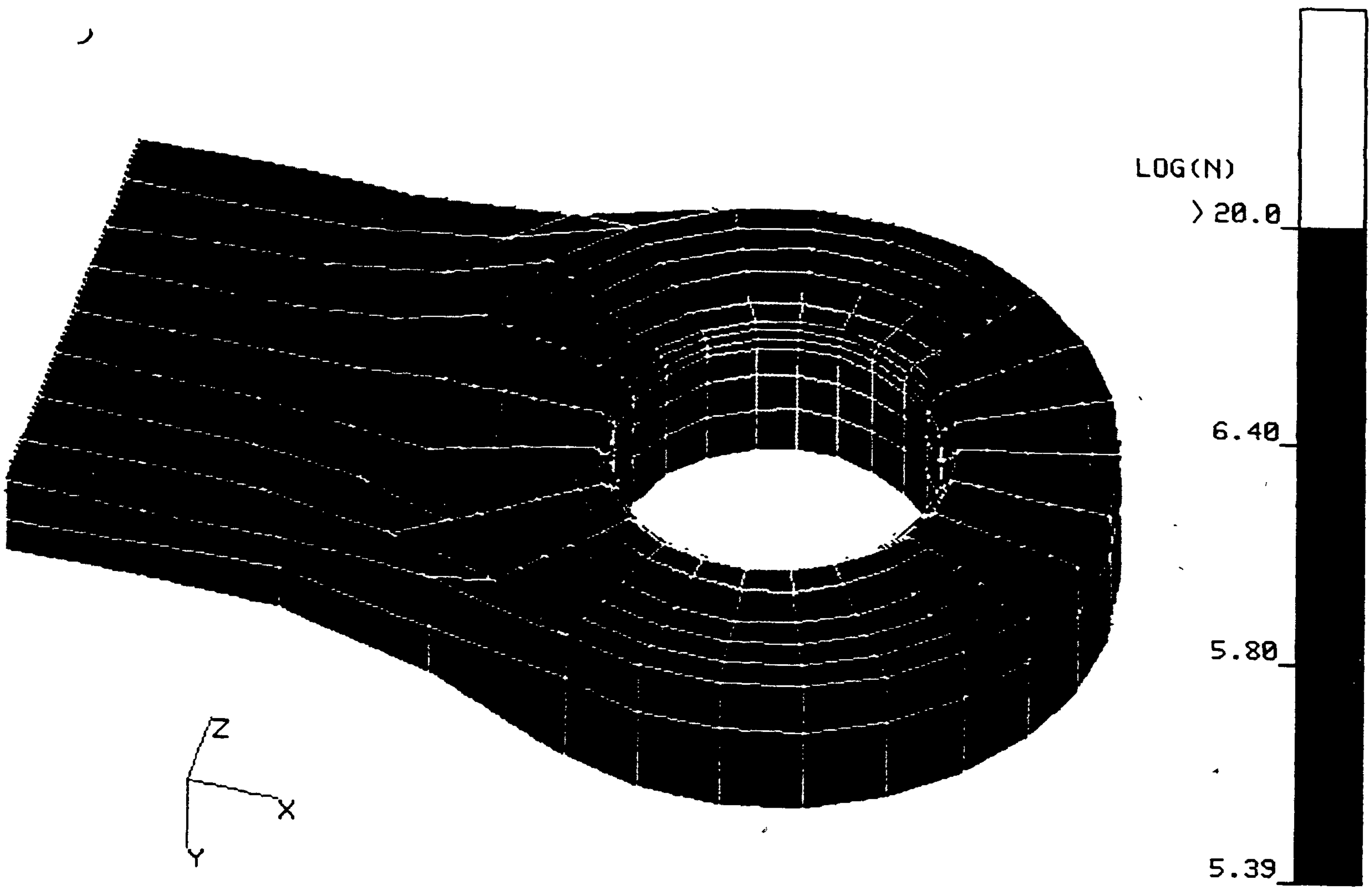


Figure 4.25: "High Amplitude Cycles" Lug Fatigue Endurance - Without Shot-Peening

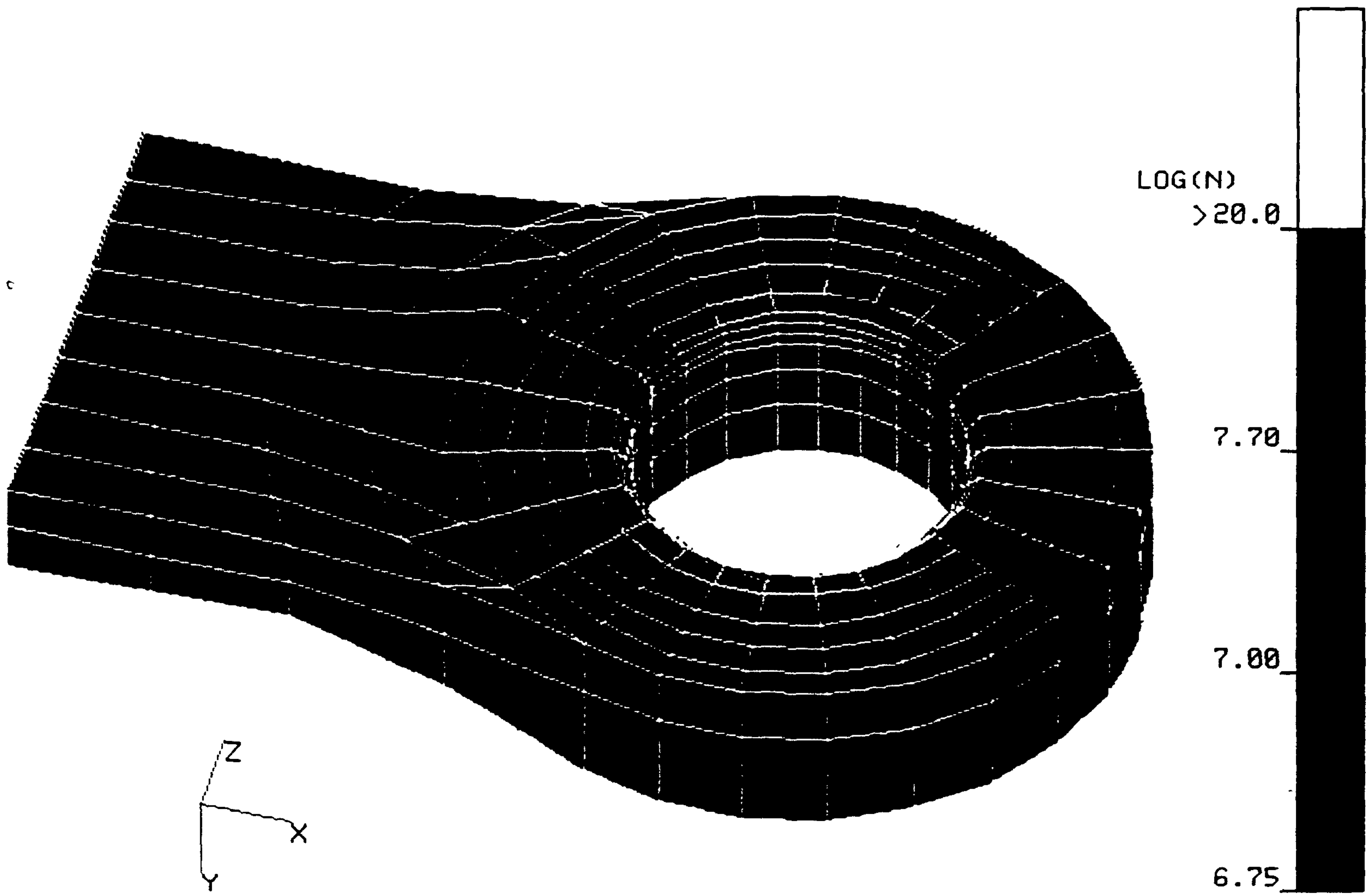


Figure 4.26: Lug Fatigue Endurance - With Shot-Peening

When the frequency of "High Amplitude Cycles" is 0.03%
and the frequency of "Low Amplitude Cycles" is 99.97 %

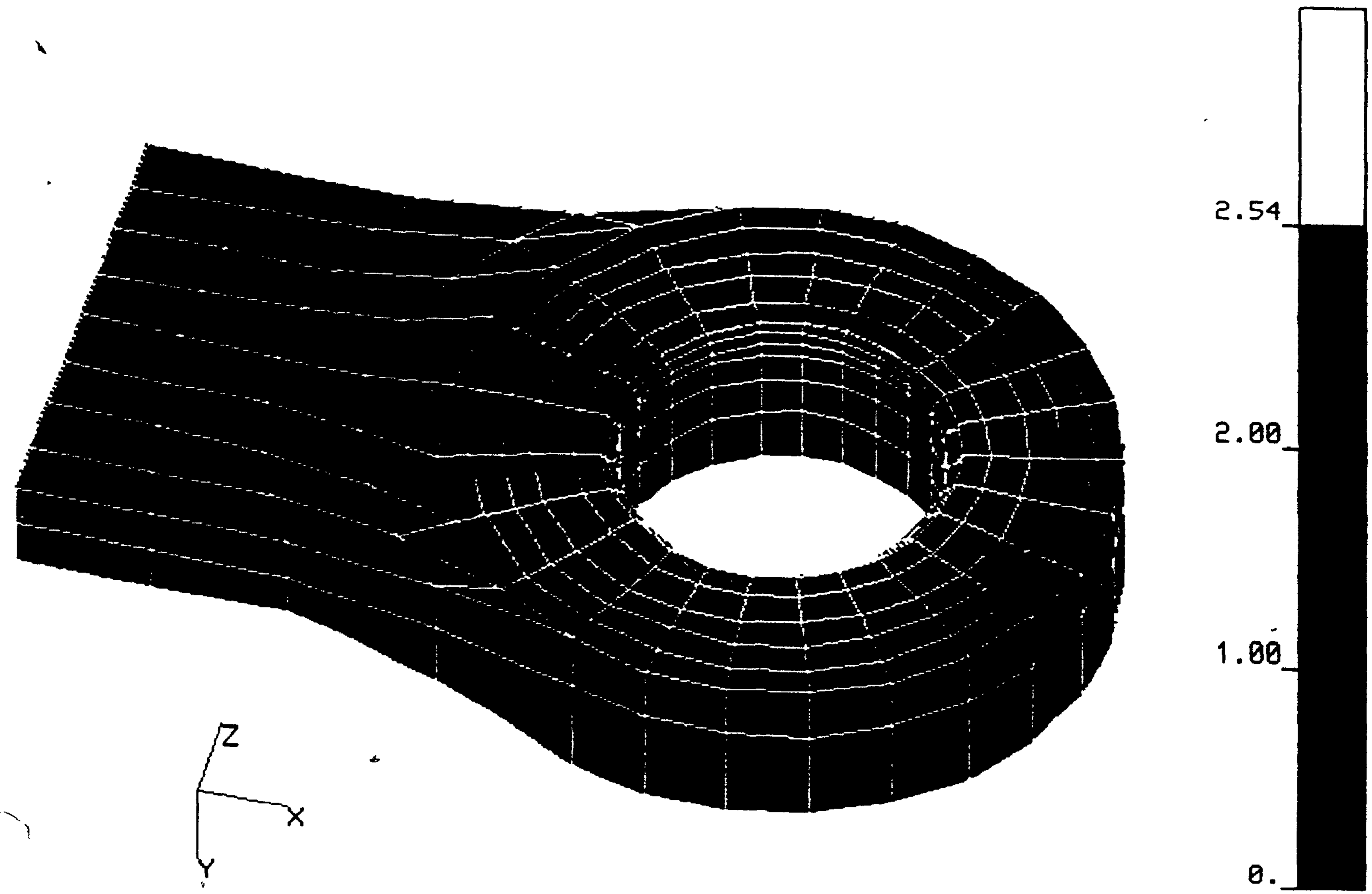


Figure 4.27: "High Amplitude Cycles" Lug Reliability Factor - With Shot-Peening

Chapter 5

Discussion

5.1 Discussion on the stress analysis

1. It has been found that, when the lug/pin joint is submitted to *HL1*, the top lug and bush lose contact over a large area. Results might have been slightly different if friction had been considered or if a finer F.E. mesh had been used (see the two paragraphs below) but it is doubted that the top lug and bush would have kept contact when the lug/pin joint were highly loaded

2. The accuracy of the stress analysis would certainly be improved by using a finer F.E. mesh around the lug hole. It has been noted in Section 3 3 2 that the stresses around the lug hole were underestimated.

3. Another possible improvement to the accuracy would result from considering the friction between the lug/pin joint components. Friction could be taken into account easily by adding a non zero coefficient of friction in the NASTRAN PGAP cards [5]. Rerunning NASTRAN would require a large amount of computer time though. As explained in [27], this would improve the fatigue analysis even if fretting occurs.

   Notice that if the bolt had been tightened, friction at the pin head/lug, bushing flange/lug, bush flange/washer and washer/blade interfaces would have had to be considered since a large proportion of the load would have been transmitted by shear stresses through these interfaces. If it can be reasonably assumed that no slip occurs between two components, then they can be considered as only one component and their F.E. mesh can be made continuous. This would make the computation much faster.

4. The use of the substructuring technique in the F.E. analysis could decrease the computer time required to perform the stress analysis of the lug/pin joint. Each component of the lug/pin joint could be seen as a linear substructure whose internal degrees of freedom can be "condensed". The total structure stiffness would then be formed from the substructure stiffness matrices. Referring to the coarse F.E. model (Figure 3.7), instead of having around:

$$1625\ nodes \times 3\ degrees/node = 4875\ degrees\ of\ freedom, \tag{5.1}$$

a condensed structure would have around:

$$((468\ gaps + 72\ MPC\ cards)\times 2)\times 3\ degrees/node = 3240\ degrees\ of\ freedom. \tag{5.2}$$

5.2 Discussion on the results of the lug/pin joint fatigue analysis

1. The crack growth time is small relatively to the crack initiation time for lugs in high cycle fatigue range. The endurance computed using the methodology proposed in this thesis can be assimilated into either the crack initiation time or the life to rupture (crack initiation period + crack growth period).

2. The reliability of the pin and bushings is so much higher than the reliability of the lugs that no pin or bushing failure is expected.

3. Cracks are likely to appear on the areas colored in red on Figure 4.26. It shows that the crack will appear at the interface between the lug and the bushing flange. Cracks located in this position cannot be detected visually because of the presence of the bushing (see Figure 5.1).

4. The expected life of the component is $10^{6.75} = 5.6E6$ cycles (Figure 4.26). To get a reliability of 0.9999999, the lugs must be retired from service after $0.038 \times 5.6E6 = 2.1E5$ cycles. The factor 0.038 was taken in Table 4.11. Although the lugs are made of aluminum, the Table 4.9 value was not used because of the fretting that may modify the endurance distribution.

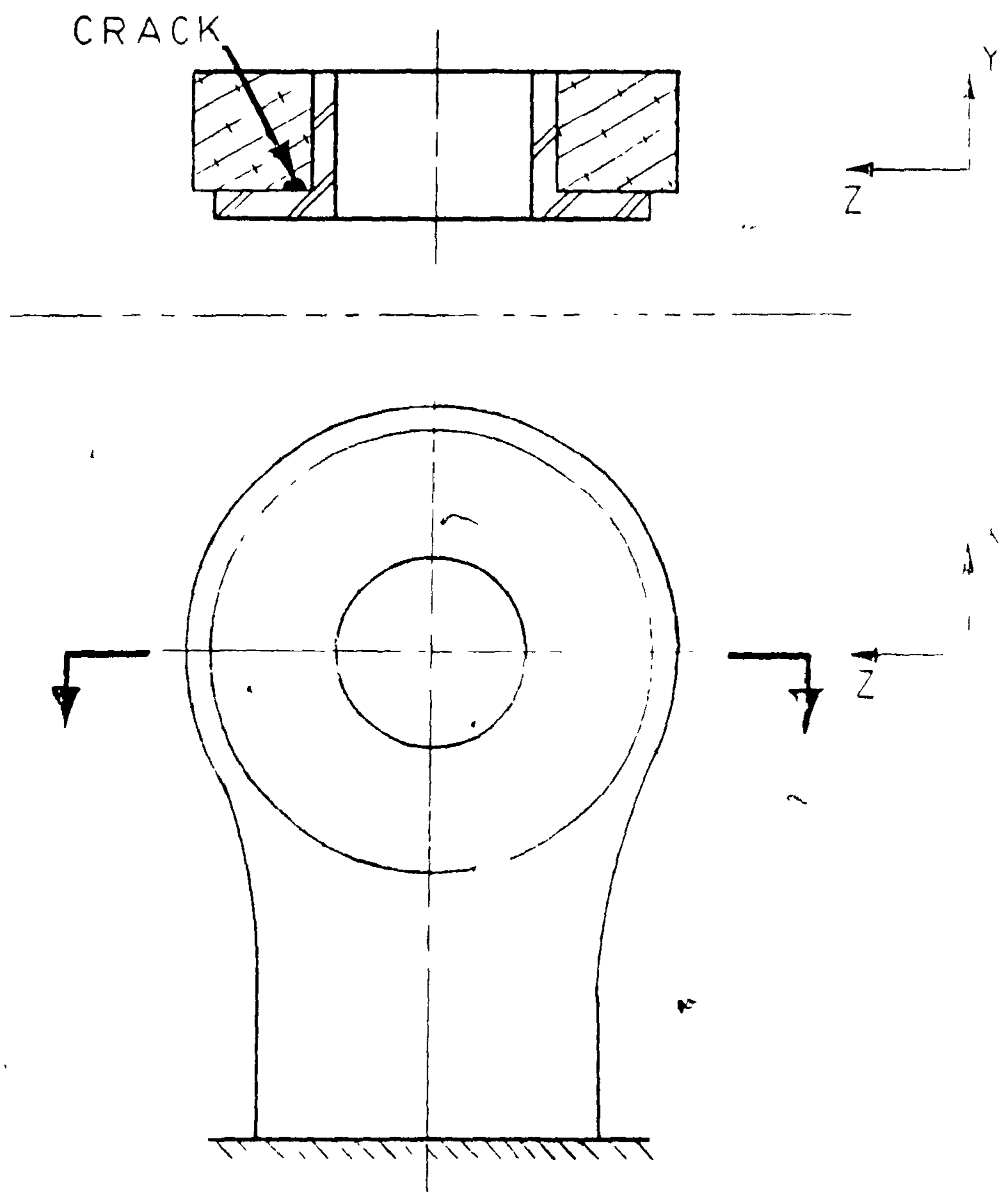


Figure 5.1: Crack Initiation Site

The crack initiation site cannot be detected without removing the bushing.

5.3 Discussion on the fatigue analysis methodology

1. The fatigue analysis methodology developed here can be applied to a wide range of problems, not only to lug/pin joints. There are, however, a few restrictions to the use of the ENDUR program. They may be listed as follows.

   (a) **The stress cycle must reflect the loading cycle and the loading conditions that induced the extreme stress states must be identified.** Usually, this is straightforward but the user is advised to be careful. Figure 4.17 shows an example where a stress cycle at the critical zone is not similar to the loading cycle.

   (b) All stresses must be expressed using the same material coordinate system, e.g., the R, θ, Z' coordinate system was used in this investigation.

   (c) Nodes numbering must start at 1 and no number can be skipped. The maximum number of nodes is 3000. The maximum number of elements is 2000.

   (d) Fretted surface must be normal to one of the unit vectors of the material coordinate system. In the lug/pin joint concentrated upon in the present investigation, all fretted surfaces were normal to either R or Z'.

   It would be possible to make the ENDUR program more sophisticated so it could overcome most of these restrictions.

2. Fretting effects are hard to incorporate into an explicit equation. A good equation would probably have to take into account the normal pressure and the slip between the mating surfaces. If improved equations for the computation of the fretting effect could be found, they could probably be a implemented in the ENDUR program.

3. The ENDUR program is built to be easily modified if different equations are prefered for the $S-N$ curve, mean stress effect, fretting effect, etc..

4. As previously mentioned in the Introduction, the present state of knowledge on fatigue does not permit very accurate evaluation of the fatigue endurance particularly in the high cycle range. The methodology proposed in this thesis is to be used to *compare* alternative designs and select those that are worth testing.

5.4 Possible improvements to the design

The fact that the crack initiation sites are not visible without removing the interference fit bushes is certainly not convenient. As the lug/pin joint is designed now, sophisticated inspection techniques may be required to detect cracks at an early stage of their formation. Flanged bushes tend to induce very high tension at one side of the hole which increases the likelihood of crack initiation at this site. This finding is confirmed by [28]. Though it is hardly possible for the lug/pin joint under study, [28] proposes to use coating or a separate shims to protect lug sides.[3] The present investigator would rather recommend a decrease in the bushes flange diameter and thickness. Moreover since the pin is much more fatigue resistant than the lugs, the present investigator also proposes to decrease the pin diameter to $\approx \frac{7}{8}in$ and increase the bushes and sleeve wall thicknesses accordingly. It would then be important to check the pin static strength to make sure that it can sustain the expected "once in a lifetime" load.

Increasing the lug/bush interference fit from $\approx 0.2\%$ to ≈ 0.3 or 0.4 or even 0.6%, which are more usual values for interference of lug/pin assemblies [1], could prevent the lugs and bushings from losing contact when the lug/pin joint is highly loaded (see Figure 3.43). Introducing higher interference fit bushings in the lugs would be harder though and special care should then be taken to avoid scratching the lug. This would also reduce the slip at the bush/lug interfaces. Fretting would then be less severe.

The geometrical modifications proposed in the above paragraphs would be easy to implement since they would not require any modification of the blade or the lugs.

Pre-stressing the lugs by hole expansion with split-sleeve could be a more effective way to introduce beneficial compressive stresses in the lugs than the shot peening technique. The problem with both techniques is that it is difficult to control how well they have been applied.

An important increase in endurance can be obtained from the application of a clamping pressure, via tightening of the bolt for example, because the load is then transferred by friction rather than pin load (See the third paragraph of Section 5.1). Notice that lubrication of such joints is detrimental [29].

Chapter 6

Conclusions

6.1 Conclusions

A methodology for the stress and fatigue analysis of metallic lug/pin joints has been developed. For the particular lug/pin joint that was under investigation, the methodology has permitted the suggestion of possible improvements to its design. It is clear, however, that the methodology could be more extensively exploited to simplify the comparison of different lug/pin joint designs thereby aiding the selection of those that are worthy of testing.

To be more specific, the methodology permits:

- easy performance of the stress analysis of topologically similar lug/pin joints,
- a determination of the fatigue crack initiation site,
- an evaluation of the fatigue lives of critical components of an assembly, and
- the assessment of their reliability.

The methodology involves the following steps:

- definition of the lug/pin joint geometry,
- a F.E. mesh definition using solid and interface elements,
- computation using the F.E. method of the stress states induced by extreme loading conditions,

- evaluation of the endurance for each stress level, and
- computation of the endurance taking into account the relative frequency of occurrence of each stress level.

Notice that the fatigue analysis program ENDUR (explained in Appendix L) can be applied to different metallic assemblies.

6.2 Proposed Future Investigations

As is usually the case in high cycle fatigue, the crack initiation time is much longer than the crack propagation time. The application of fracture mechanics techniques that permit an evaluation of the crack growth once a "small" crack has formed would be useful since it would permit an evaluation of the time interval between inspections.

Secondly, although the manufacture of a few lug/pin joint specimens and their testing would be very expensive and time consuming, empirical verification of the results predicted by this methodology would certainly be very informative and would permit an improvement in some of the assumptions that had to be made in its development. In particular, it would be of interest to see the crack initiation *position* and how the cadmium plating on the bushes resists fretting.

Finally, it must be recalled that most of the assumptions made during the development of this thesis have been conservative. On the other hand, the stresses around the lug hole have been underestimated (Section 3.3.2) due to the implementation of a relatively coarse F.E. grid. It would therefore be beneficial for the stress analysis to be repeated with a much finer F.E. mesh at this location.

References

[1] R.B. Heywood, *Designing Against Fatigue*, Reinhold, New-York, 1962

[2] S.E. Larsson, *The Development of a Calculation Method for the Fatigue Strength of Lugs and a Study of Test Results for Lugs of Aluminum*, from *Fatigue Design Procedures, Proceedings of the 4th Symposium of the International Committee on Aeronautical Fatigue, Munich, 1965*, edited by E. Gassner & W Schutz, Pergamon Press, 1969.

[3] A Buch, *Fatigue and Fretting of Pin-Lug Joints with and without Interference Fit*, Wear, vol 16, no 9, 1977.

[4] ASM, *Metals Handbook*, 9th edition, Volume 2, *Properties & Selection of Non-ferrous Alloys & Pure Metals*, 1979

[5] *MSC NASTRAN user's manuals*, MacNeal-Schwendler Corporation, Los Angeles, 1986.

[6] *PATRAN user's guide*, Vol 1 & 2, PDA Engineering, Santa-Ana, 1986

[7] N.E. Frost, K.J. Marsh, L.P. Pook, *Metal Fatigue*, Clarendon Press, Oxford, 1974.

[8] W. Weibull, *Fatigue Testing & Analysis of Results*, Pergamon Press, New-York, 1961.

[9] SAE, *Fatigue Design Handbook*, SAE, New-York, 1968.

[10] L. Sors, *Fatigue Design of Machine Components*, Pergamon Press, Oxford, 1971.

[11] R.B. Waterhouse, *Fretting Corrosion*, Pergamon Press, London, 1972.

[12] P G. Forrest, *Fatigue of Metals*, Pergamon Press, London, 1962.

[13] ASME, *Metal Engineering Design Handbook*, McGraw-Hill, New-York, 1953.

[14] H.O Fuchs, *Prestressing and Fatigue*, Metal Engineering Quarterly, American Society for Metals, Aug 1966.

[15] H.O. Fuchs, *The Effect of Self-stresses on High cycle Fatigue*, Journal of Testing and Evaluation, vol 10, no 4, July 1982.

[16] F. Och, *Fatigue Strength* in *Helicopter Fatigue Design Guide*, AGARD-AG-292, AGARD, Neuilly-sur-Seine, 1983.

[17] P.E. Peterson, *Stress Concentration Factors*, John Wiley & sons, New-York, 1974.

[18] Y.S Garud, *Multiaxial Fatigue: A Survey of the State of the Art*, Journal of Testing & Evaluation, vol 9, no 3, May 1983

[19] H.O Fuchs, Stephens R I., *Metal Fatigue in Engineering*, John Wiley & sons, New-York, 1980

[20] A Cardou, *Analyse des Contraintes*, Université Laval, Québec, 1983

[21] B.F. Langer, *Design of Vessel involving Fatigue* in *Pressure Vessel Engineering Technology*, R W Nichols editor, Applied Science Publishers ltd, London, 1971

[22] H.O. Fuchs, *Fatigue Research with Discrimating Specimens*, Fatigue of Engineering Materials & Structures vol 2, no 2, 1979.

[23] J. Schijve, *Fatigue of Lugs* from *Contributions to the Theory of Aircraft Structure*, Delft University Press, 1972.

[24] W.D. Prayoorat, S. Taylor, *Computed-Aided Fatigue Analysis*, Computer-Aided Design, vol 18, no 5, June 1986.

[25] ASCE, *Fatigue Reliability: Variable amplitude Loading*, Journal of Structural Division, vol 108, st-1, January 1982.

[26] ASCE, *Fatigue Reliability: An Introduction*, Journal of Structural Division, vol 108, st-1, January 1982.

[27] J.J. O'Connor, *The Role of Elastic Stress Analysis in the Interpretation of Fretting Fatigue Failures*, from *Fretting Fatigue*, edited by R.B. Waterhouse, Applied Science Publishers, London, 1981.

[28] F.F. Liard, *Fatigue Strength Improvement & Deterioration* in *Helicopter Fatigue Design Guide*, AGARD-AG-292, AGARD, Neuilly-sur-Seine, 1983.

[29] H.P. Van Leeuwe, A. Nederveen, H.H. Ruiter, *Fretting & Fatigue under Complex Loads*, National Aerospace Laboratory, NLR, Netherlands, 1973

[30] *CRC Handbook of Materials Science*, Vol 2, *Metals, Composites & Refractory Materials*, CRC Press, Cleveland, 1974.

Appendix A

Geometry Definition Program

The GEO program creates the PATRAN session files corresponding to the lug/pin joint geometry given in the input file. The format of the input file must be the same as the file GEO.DAT (TABLE A.1). The input file contains only independent variables. They are shown in Figure A.1.

Each of the 6 GEO output files is used to create a different component of the lug/pin joint geometry.

The GEO output files are:

LUGx.SES
BLAx.SES
BUSx.SES
SLEx.SES
WASx.SES
PINx.SES

where "x" is a one digit user chosen ID.

The geometric entities (points, lines, surfaces, solids) have been numbered according to the following scheme:

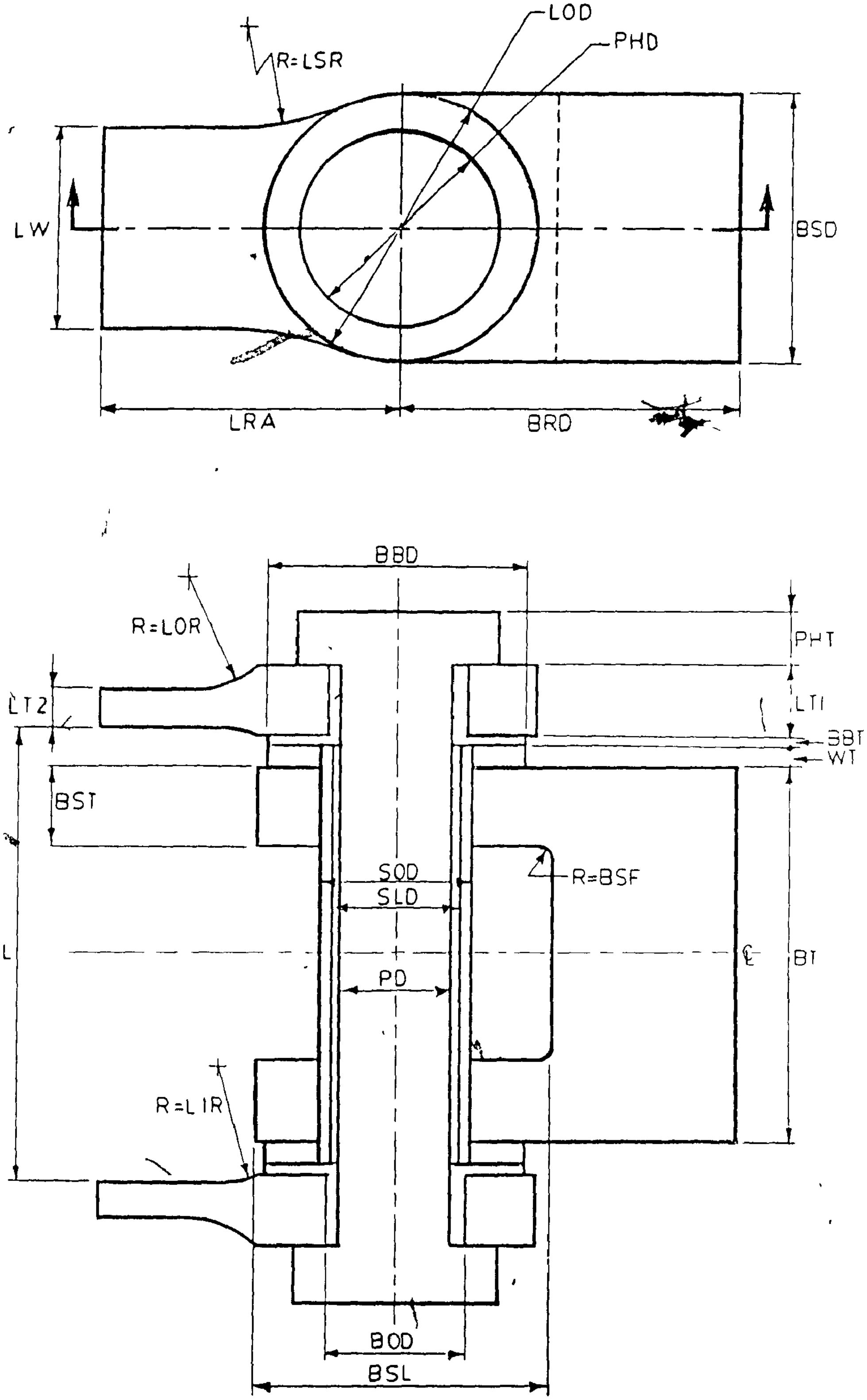


Figure A.1: Lug/pin joint geometric parameters

components	geometric entities numbering
lugs	from 101 to 399
blade	from 401 to 599
bushings	from 601 to 699
sleeve	from 1701 to 1899
	(except the solids: from 701 to 799)
pin	from 901 to 1099
washers	from 801 to 899

The output files must be executed by PATRAN separately. They can be put together using the "neutral" file feature of PATRAN. (see Figure A.2). No geometric entity ID must be changed while putting the neutral files together.

The GEO program cannot handle "topological" modifications. If, for a set of input data, the topology is modified, then PATRAN will give a warning or an error message when executing the GEO output files. In this case, the user may have to enter the modified part of the model within PATRAN.

Notice that the measurements in the input file must not take into account any clearance or interference. These will be taken into account at a subsequent stage. Small fillets and chamfers are not taken into account at this stage either.

To run GEO, type:

```
$RUN GEO
ENTER INPUT FILE NAME:              <--- written by GEO
GEO.DAT                             <--- example, typed by the user
ENTER OUTPUT FILES ID (ONE DIGIT): <--- written by GEO
1                                   <--- example, typed by the user
```

```
*
*
PIN DIAMETER                            [in]    (PD) =1
PIN HEAD DIAMETER                       [in]    (PHD)=1 8
PIN HEAD THICKNESS                      [in]    (PHT)= 5
*
SLEEVE OUT DIAMETER                     [in]    (SOD)=1 315
SLEEVE LINER DIAMETER                   [in]    (SLD)=1 125
*
WASHER THICKNESS                        [in]    (WT) = 21
*
BUSHING OUT DIAMETER                    [in]    (BOD)=1 2
BUSHING BASE OUT DIAMETER               [in]    (BBD)=2 25
BUSHING BASE THICKNESS                  [in]    (BBT)= 097
*
LUG THICKNESS #1                        [in]    (LT1)= 625
LUG THICKNESS #2                        [in]    (LT2)=.359
LUG "IN" LENGTH                         [in]    (LIL)=4 01
LUG OUT DIAMETER                        [in]    (LOD)=2.47
LUG WIDTH                               [in]    (LW) =1 83
LUG REFERENCE TO PIN AXIS DISTANCE      [in]    (LRA)=2.6
LUG "IN" RADIUS                         [in]    (LIR)=1
LUG "OUT" RADIUS                        [in]    (LOR)=1
LUG SUPPORT RADIUS                      [in]    (LSR)=3.
*
BLADE THICKNESS                         [in]    (BT) =3.328
BLADE SUPPORT DIAMETER                  [in]    (BSD)=2.47
BLADE SUPPORT THICKNESS                 [in]    (BST)=.71
BLADE SUPPORT LENGTH                    [in]    (BSL)=2.6
BLADE SUPPORT FILLET RADIUS             [in]    (BSF)=.25
BLADE REFERENCE TO PIN AXIS DISTANCE    [in]    (BRD)=3.
*
```

Table A.1: Typical GEO program input file

Only the values placed right to the equal signs can be changed. No new line must be introduced. No line can be deleted. The variables are illustrated in Figure A.1.

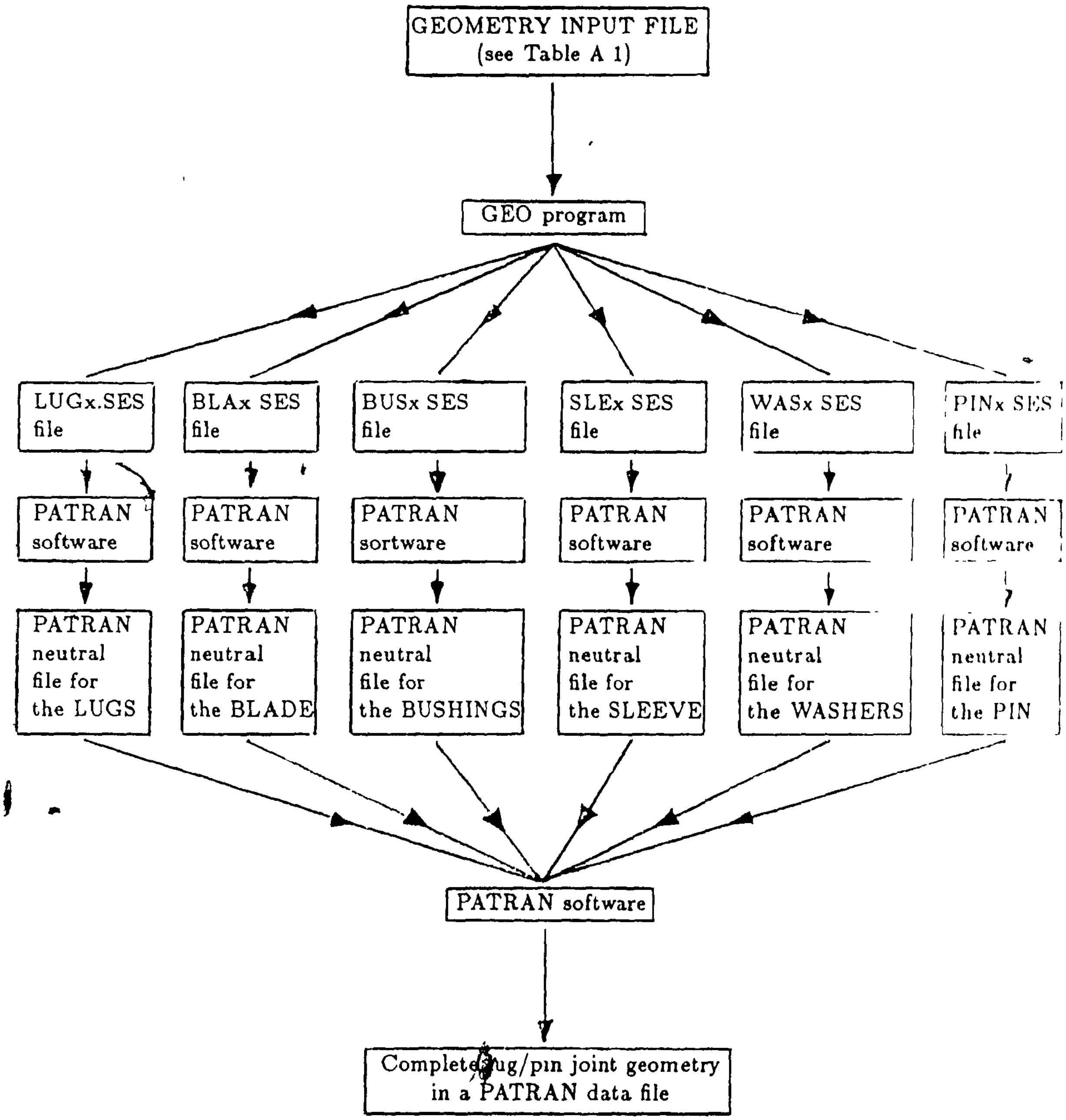


Figure A.2: Lug/pin joint geometry definition flow chart

```
$
$                                    PIN
$
 GR,901,,0,0
GR,902,,0,1.664000
GR,903,TR,0/-0.710000,902
GR,904,TR,0/ 0.210000,902
GR,905,TR,0/ 0.097000,904
GR,906,TR,0/ 0.725000,905
GR,907,TR,0/ 0.500000,906
 LI,901T906,ST,, 901/903/902/904T906,903/902/904T907
LI,6#,TR, 0.500000,901T906
LI,#,TR, 0.165000,912
LI,#,TR,0.900000,906
 PA,901T908,2L,, 901T906/912/913,907T912/913/914
HP,901T916,ARC,G901/G902/110.140862,901T908
 PA,8#,MI,X,901T908
HP,8#,ARC,G901/G902/-69.859138,909T916
 LI,1T#,D
 PA,1T#,D
$
 HP,24#,MI,Z,901T924
 HP,48#,MI,Y,901T948
$
```

Table A.2: Example of a GEO program output file

This file is a PATRAN session file. It creates the pin geometry when executed in PATRAN.

Table A.3: GEO program FORTRAN code

```
C
      PROGRAM GEO
C     -----------
C
      IMPLICIT REAL I,J,K,L,M,N
      CHARACTER*1 A1,ID
      CHARACTER*50 A50, INPUT
C
      WRITE(6,2)
 2     FORMAT(' ENTER INPUT FILE NAME  ')
      READ(5,3) INPUT
C
      WRITE(6,4)
 4     FORMAT(' ENTER OUTPUT FILES ID (ONE DIGIT)  ')
      READ(5,3) ID
C
 3     FORMAT(A)
C
      OPEN ( UNIT=2,
     &       FILE=INPUT,
     &       STATUS='OLD',
     &       ACCESS='SEQUENTIAL',
     &       FORM='FORMATTED',
     &       READONLY )
C
C      READING INPUTS
C      --------------
C
 1      FORMAT(//,3(51X,F9 6,/),/,2(51X,F9 6,/),/,(51X,F9 6,/),/,
     &  3(51X,F9.6,/),/,9(51X,F9 6,/),/,6(51X,F9 6/))
C
       READ(2,1) PD
     &           ,PHD
     &           ,PHT
     &           ,SOD
     &           ,SLD
```

```
     &          ,WT
     &          ,BOD
     &          ,BBD
     &          ,BBT
     &          ,LT1
     &          ,LT2
     &          ,LIL
     &          ,LOD
     &          ,LW
     &          ,LRA
     &          ,LIR
     &          ,LOR
     &          ,LSR
     &          ,BT
     &          ,BSD
     &          ,BST
     &          ,BSL
     &          ,BSF
     &          ,BRD
C
C
 9999      FORMAT(A18)
 9998      FORMAT('$')
 9997      FORMAT('$',A40)
C
C     CREATING "LUGx.SES" FILE
C     ------------------------
C
      OPEN ( UNIT=1,
     &        FILE='LUG'//ID//'.SES',
     &        STATUS='NEW',
     &        ACCESS='SEQUENTIAL',
     &        CARRIAGECONTROL='LIST',
     &        FORM='FORMATTED' )
C
          WRITE(1,*) 'SET.LINES.O'
C
C
          WRITE(1,9998)
```

```
      WRITE(1,9997)'          LUG'
      WRITE(1,9998)
      WRITE(1,*) 'SET,TOL,.010'
      WRITE(1,9997)'     GRID 101, 102 DEFINE AN AXIS'
      WRITE(1,*) 'GR,101,,0'
      WRITE(1,*) 'GR,102,,0,1'
      WRITE(1,9998)
C
      L1=-SQRT((LOD/2.+LSR)**2-(LW/2.+LSR)**2)
      L2=-(LSR+LW/2 )
C
      WRITE(1,1002)L1,L2
 1002  FORMAT('GR,#,,',F9 6,',0,',F9 6)
C
      L7=BT/2.+WT+BBT
      L3=L7+LT1
C
      WRITE(1,1004) L3
 1004  FORMAT('GR,#,TR,/',F9.6,',103')
      WRITE(1,1006) LSR
 1006  FORMAT('GR,#,TR,//',F9.6',104')
      WRITE(1,9998)
      WRITE(1,*) 'LI,101,ARC,G103/G104/40.,105'
      WRITE(1,9997) '40 ABOVE N.I. (Not Important)'
      WRITE(1,9998)

      WRITE(1,*) 'LI,#,EXT,-.3,101'
      WRITE(1,9997) '  .3 ABOVE N.I.'
      WRITE(1,9998)
      WRITE(1,*) 'LI,#,MER,,101/102'
      WRITE(1,*) 'LI,#,TR,/-1,103'
      WRITE(1,9997)'  -1 ABOVE N.I.'
      WRITE(1,9998)
      WRITE(1,*) 'PA,101,2L,,103,104'
C
      L6 = LIL/2.+LT2+LOR
      L4 = SQRT(LOR**2-(L6-L3)**2)
      L5 = -(LOD/2.+L4)
C
      WRITE(1,1007) L5,L6
```

```
 1007     FORMAT('GR,#,,',F9.6,',',F9.6)
         WRITE(1,*) 'GR,#,TR,//1,110'
         WRITE(1,9997) '  1 ABOVE N.I.'
         WRITE(1,9998)
C
         LORM=-LOR
C
         WRITE(1,1008) LORM
 1008     FORMAT('GR,#,TR,/',F9.6,',110')
         WRITE(1,9998)
C
         L8 = LIL/2.-L7
         L9 = LIR-L8
         L10 = -SQRT(LIR**2-L9**2)-LOD/2.
         L11 = LIL/2.-LIR
C
         ANGL2 = -ASIND((LSR+LW/2.)/(LSR+LOD/2.))
         ANGL1 = 180.+ANGL2
         ANGL3 = -90.-ANGL2
         ANGL4 = ASIND(L4/LOR)
         ANGL5=-ANGL2
         ANGL6 = ACOSD(L9/LIR)
         ANGL7 = -ANGL1/2
C
         WRITE(1,1010) ANGL4
 1010     FORMAT( 'LI,#,ARC,G110/G111/',F11.6,',112')
         WRITE(1,1012) ANGL5
 1012     FORMAT( 'PA,102,ARC,G102/G101/',F11.6,',105')
         WRITE(1,9998)
         WRITE(1,9998)
         WRITE(1,*) 'LI,106,INT,,101,102'
         WRITE(1,*) 'N'
         WRITE(1,*) '1'
         WRITE(1,*) 'GR,117,,.1,.1'
         WRITE(1,*) 'GR,117,ER          ! GRID 117 IS DUMMY'
         WRITE(1,9998)
         WRITE(1,*) 'PA,103,2L,P102,105,106'
C
         WRITE(1,1014)L10,L11
```

```
 1014      FORMAT( 'GR,#,,',F9.6,',',F9 6)
          WRITE(1,9998)
          WRITE(1,*) 'GR,119,TR,//1,118'
          WRITE(1,9997) '1 ABOVE N I '
          WRITE(1,9998)
          WRITE(1,1016) LIR
 1016      FORMAT( 'GR,120,TR,/',F9.6,',118')
          WRITE(1,9998)
C
C
          WRITE(1,1018) ANGL6
 1018      FORMAT( 'LI,109,ARC,G119/G118/',F11.6,',120')
C
          LT2M=-LT2
C
          WRITE(1,1020) LT2M
 1020      FORMAT( 'GR,122,TR,/',F9.6,',112')
          WRITE(1,*) 'LI,110,2G,,122,120'
          WRITE(1,*) 'LI,111,MER, 005,110/109'
          WRITE(1,1022) ANGL2
 1022      FORMAT( 'PA,104,ARC,G101/G102/',F11.6,',111')
          WRITE(1,*) 'LI,#,INT,,101,104'
          WRITE(1,*) 'N'
          WRITE(1,*) '1'
          WRITE(1,*) 'GR,126,,-.1,-.1'
          WRITE(1,*) 'GR,126,ER'
          WRITE(1,9997) ' GRID 126 IS DUMMY'
          WRITE(1,9998)
          WRITE(1,*) 'PA,105,2L,104,111,112'
          WRITE(1,9998)
          WRITE(1,*) 'HP,101,2P,,103,105'
C
          LIL2D = LIL/2.
          BSLM  = -BSL
C
          WRITE(1,1024) BSLM,LIL2D
 1024      FORMAT( 'GR,#,,',F9.6,',',F9.6)
          WRITE(1,9998)
C
```

```
      LW2DM=-LW/2
C
      WRITE(1,1026) LW2DM
 1026  FORMAT( 'GR,128,TR,//',F9 6,',127' )
      WRITE(1,*) 'LI,113,2G,P104,122,125'
      WRITE(1,*) 'LI,114,2G,,127,128'
      WRITE(1,9998)
      WRITE(1,*) 'SET,TOL, 015'
      WRITE(1,*) 'PA,106,2L,,113,114'
      WRITE(1,1028) LT2
 1028  FORMAT( 'PA,107,TR,/',F9 6,',106')
      WRITE(1,*) 'HP,102,2P,,106,107'
C
      BBD2DM = -BBD/2
      BOD2DM = -BOD/2.
C
      WRITE(1,1040) BBD2DM,L7
 1040 FORMAT('GR,#,,',F9 6,',',F9 6)
      WRITE(1,1050) BOD2DM, L7
 1050 FORMAT('GR,132,,'F9 6,',',F9 6)
      WRITE(1,9998)
      WRITE(1,9997) '  MODIFIED, JULY 1  ,1987'
      WRITE(1,9998)
C
      PHD2DM = -PHD/2
C
      WRITE(1,1060) PHD2DM,L7
 1060 FORMAT('GR,#,,',F9 6,',',F9 6)
      WRITE(1,*) 'LI,3#,ST,,132/133/131,133/131/121'
      WRITE(1,1065) LT1
 1065 FORMAT('LI,3#,TR,/',F9 6,',115T117')
      WRITE(1,*) 'PA,3#,2L,,115T117,118T120'
      WRITE(1,*) 'PA,3#,MI,X,108T110'
      WRITE(1,1070) ANGL2
 1070 FORMAT( 'HP,3#,ARC,G101/G102/',F11 6,',108T110')
      WRITE(1,1080) ANGL1
 1080 FORMAT('HP,6#,ARC,G101/G102/',F11 6,',111T113')
      WRITE(1,*) 'LI,1T#,DEL'
      WRITE(1,*) 'PA,1T#,DEL'
```

```
WRITE(1,*) 'GR,119/118/103/104/110/105,DEL'
WRITE(1,*) 'SET,TOL, 05'
WRITE(1,*) 'HP,11#,MI,Z,H101T111'
WRITE(1,*) 'HP,22#,MI,Y,H101T122'
WRITE(1,9998)
WRITE(1,9998)
WRITE(1,*) 'SET,TOL, 015'
WRITE(1,9998)
WRITE(1,*) 'HP,101,LAB,,113/114'
WRITE(1,*) 'HP,112,LAB,,113/112'
WRITE(1,*) 'HP,123,LAB,,180/181'
WRITE(1,*) 'HP,134,LAB,,180/179'
WRITE(1,9998)
WRITE(1,*) 'HP,103 ,LAB,,145/147'
WRITE(1,*) 'HP,106 ,LAB,,137/138'
WRITE(1,*) 'HP,107 ,LAB,,151/152'
WRITE(1,*) 'HP,114 ,LAB,,132/133'
WRITE(1,*) 'HP,117 ,LAB,,172/171'
WRITE(1,*) 'HP,118 ,LAB,,166/168'
WRITE(1,9998)
WRITE(1,*) 'HP,125 ,LAB,,191/195'
WRITE(1,*) 'HP,129 ,LAB,,194/198'
WRITE(1,*) 'HP,128 ,LAB,,206/205'
WRITE(1,*) 'HP,139 ,LAB,,203/204'
WRITE(1,*) 'HP,140 ,LAB,,231/232'
WRITE(1,*) 'HP,136 ,LAB,,225/227'
WRITE(1,9998)
WRITE(1,*) 'HP,104 ,LAB,,147/149'
WRITE(1,*) 'HP,108 ,LAB,,138/141'
WRITE(1,*) 'HP,109 ,LAB,,152/155'
WRITE(1,*) 'HP,115 ,LAB,,133/131'
WRITE(1,*) 'HP,120 ,LAB,,168/170'
WRITE(1,*) 'HP,119 ,LAB,,171/175'
WRITE(1,9998)
WRITE(1,*) 'HP,130 ,LAB,,205/212'
WRITE(1,*) 'HP,131 ,LAB,,198/202'
WRITE(1,*) 'HP,126 ,LAB,,195/199'
WRITE(1,*) 'HP,137 ,LAB,,227/229'
WRITE(1,*) 'HP,142 ,LAB,,232/235'
```

```
          WRITE(1,*) 'HP,141 ,LAB,,204/211'
          WRITE(1,9998)
C
      CLOSE(1)

C
C     CREATING "BLAx.SES" FILE
C     ------------------------
C
C
      OPEN ( UNIT=1,
     &        FILE='BLA'//ID//'.SES',
     &        STATUS='NEW',
     &        ACCESS='SEQUENTIAL',
     &        CARRIAGECONTROL='LIST',
     &        FORM='FORMATTED' )
C
          WRITE(1,*) 'SET,LINES,0'
C
          WRITE(1,9998)
          WRITE(1,9997) '    BLADE'
          WRITE(1,9998)
          WRITE(1,9997) '    GR 401,402 DEFINE AN AXIS'
          WRITE(1,9998)
          WRITE(1,*) 'GR,401,,0'
          WRITE(1,*) 'GR,402,,0,1'
          WRITE(1,*) 'GR,401T402,ER'
          WRITE(1,9998)
C
          WRITE(1,9998)
      BT2D = BT/2.
      SOD2DM = -SOD/2.
      BSD2DM = -BSD/2.
          WRITE(1,4010) SOD2DM,BT2D
 4010     FORMAT('GR,#,,'F9.6,',',F9.6)
          WRITE(1,4020) PHD2DM,BT2D
 4020     FORMAT('GR,#,,',F9.6,',',F9.6)
          WRITE(1,4030) BBD2DM,BT2D
 4030     FORMAT('GR,#,,',F9.6,',',F9 6)
          WRITE(1,4040) BSD2DM,BT2D
```

```
4040    FORMAT('GR,#,,'F9.6,',',F9 6)
        WRITE(1,*) 'LI,401T403,ST,,403T405,404T406'
        WRITE(1,4050) ANGL2
4050    FORMAT('PA,401T403,ARC,G401/G402/',F11 6,',401T403')
        WRITE(1,*) 'LI,2#,MI,X,401T402'
        WRITE(1,4060) ANGL1
4060    FORMAT('PA,4#,ARC,G401/G402/',F11 6,',404T405')
        WRITE(1,9998)
        WRITE(1,4070) ANGL3
4070    FORMAT('LI,#,ARC,G401/G402/',F11.6,',410')
        WRITE(1,*) 'LI,#,EX,1,406'
        WRITE(1,9998)
        WRITE(1,*) 'GR,#,TR,//-1,416'
        WRITE(1,*) 'LI,#,ST,,416,419'
        WRITE(1,*) 'GR,#,INT,,407,408'
        WRITE(1,*) 'LI,#,ST,,417,420'
        WRITE(1,*) 'LI,#,MER,,406/409'
        WRITE(1,4080) ANGL7
4080    FORMAT('LI,411,ARC,G401/G402/',F11 6,',409')
        WRITE(1,*) 'PA,408,2L,,410,411'
        WRITE(1,9998)
      D1 = BSL - BSD/2.
        WRITE(1,4090) D1,BT2D
4090    FORMAT('GR,#,,',F9 6,',',F9 6)
        WRITE(1,4100) BSF
4100    FORMAT('GR,#,TR,',F9.6,',421')
        WRITE(1,4110) BRD,BT2D
4110    FORMAT('GR,#,,',F9.6,',',F9 6)
        WRITE(1,4120) BSD2DM
4120    FORMAT('GR,#,TR,//',F9.6,',421')
        WRITE(1,*) 'LI,2#,ST,,420/416,424/424'
        WRITE(1,*) 'LI,#,ST,,421,424'
        WRITE(1,4130)ANGL7
4130    FORMAT( 'LI,#,ARC,G402/G401/',F11.6,',413')
        WRITE(1,*) 'PA,2#,2L,,415/413,414/412'
        WRITE(1,*) 'LI,2#,ST,,421/422,422/423'
        WRITE(1,4140) BSD2DM
4140    FORMAT('LI,2#,TR,//',F9.6,',416/417')
        WRITE(1,*) 'PA,2#,2L,,416/417,418/419'
```

```
        D6 = -BST-BSF
          WRITE(1,4150) D6
 4150     FORMAT('GR,#,TR,/',F9.6,',421')
          WRITE(1,*) 'GR,#,TR,//1,427'
          WRITE(1,9997) ' 1 ABOVE NOT IMPORTANT'
          BSTM = -BST
          WRITE(1,4160) BSTM
 4160     FORMAT('GR,#,TR,/',F9.6,',421')
          WRITE(1,*) 'LI,#,ARC,G428/G427/90,429'
          WRITE(1,4170) BSD2DM
 4170     FORMAT('LI,#,TR,//',F9.6,',420')
          WRITE(1,4180) BSTM
 4180     FORMAT('PA,10#,TR,/',F9.6,',401T410')
          WRITE(1,*) 'PA,#,2L,,420,421'
          WRITE(1,4190) D6
 4190     FORMAT('PA,#,TR,/',F9 6,',412')
          WRITE(1,*) 'HP,401T412,2P,,401T412,413T424'
      BT2DM = -BT/2.
          WRITE(1,4200) BT2DM
 4200     FORMAT('PA,#,TR,/',F9 6,',412')
          WRITE(1,*) 'HP,#,2P,,424,425'
          WRITE(1,9998)
          WRITE(1,*) 'PA,1T#,DEL'
          WRITE(1,*) 'LI,1T#,DEL'
          WRITE(1,*) 'GR,427/428/419,DEL'
          WRITE(1,9998)
          WRITE(1,*) 'HP,13#,MI,Z,401T413'
          WRITE(1,*) 'HP,26#,MI,Y,401T426'
          WRITE(1,9998)
C
      CLOSE(1)
C
C     CREATING "BUSx.SES" FILE
C     ------------------------
C
      OPEN ( UNIT=1,
     &       FILE='BUS'//ID//'.SES',
     &       STATUS='NEW',
     &       ACCESS='SEQUENTIAL',
```

```
     &       CARRIAGECONTROL='LIST',
     &       FORM='FORMATTED' )
C
           WRITE(1,*) 'SET,LINES,0'
C
           WRITE(1,9998)
           WRITE(1,9997) '   BUSHING'
           WRITE(1,9998)
           WRITE(1,9997) '   GR 601,602 DEFINE AN AXIS'
           WRITE(1,9998)
           WRITE(1,*) 'GR,601,,0'
           WRITE(1,*) 'GR,602,,0,1'
           WRITE(1,*) 'GR,601T602,ER'
           WRITE(1,9998)
           WRITE(1,9998)
        PD2D = PD/2.
        BT2DWTP = BT/2.+WT
        SODPDM2D = (SQD-PD)/2.
        BODPDM2D = (BOD-PD)/2
           WRITE(1,9998)
           WRITE(1,6010) PD2D,BT2DWTP
 6010      FORMAT( 'GR,#,,',F9 6,',',F9 6)
           WRITE(1,6020) BBT
 6020      FORMAT('GR,#,TR,0/',F9 6,',603')
           WRITE(1,6030) LT1
 6030      FORMAT('GR,#,TR,0/',F9 6,',604')
           WRITE(1,6040)SODPDM2D
 6040      FORMAT( 'GR,#,TR,',F9 6,',603')
           WRITE(1,6050)BODPDM2D
 6050      FORMAT('GR,2#,TR, ',F9 6,',604T605')
        BBD2D=BBD/2.
           WRITE(1,6060) BBD2D,BT2DWTP
 6060      FORMAT('GR,#,,',F9.6,',',F9 6)
           WRITE(1,6070) BBT
 6070      FORMAT('GR,#,TR,0/ ',F9.6,',609')
           WRITE(1,9998)
           WRITE(1,9998)
           WRITE(1,9997) '  MODIFIED, JULY 1, 1987'
           WRITE(1,9998)
```

```
      PHD2D = PHD/2.
        WRITE(1,6080) PHD2D,BT2DWTP
6080    FORMAT('GR,#,,'F9.6,',',F9.6)
        WRITE(1,6090) BBT
6090    FORMAT('GR,612,TR,/',F9.6,',611')
        WRITE(1,*) 'LI,601T606,ST,,
     &603/606/611/604/607/612,606/611/609/607/612/610'
        WRITE(1,*) 'LI,#,ST,,605,608'
        WRITE(1,*) 'PA,601T604,2L,,601T604,604T607'
        WRITE(1,*) 'PA,4#,MI,X,601T604'
        WRITE(1,6100) ANGL2
6100    FORMAT('HP,601T604,ARC,G601/G602/',F11.6,',605T608')
        WRITE(1,6110) ANGL1
6110    FORMAT('HP,8#,ARC,G601/G602/',F11.6,',601T604')
        WRITE(1,9998)
        WRITE(1,*) 'PA,1T#,DEL'
        WRITE(1,*) 'LI,1T#,DEL'
        WRITE(1,9998)
        WRITE(1,*) 'HP,12#,MI,Z,601T612'
        WRITE(1,*) 'HP,24#,MI,Y,601T624'
        WRITE(1,9998)
        WRITE(1,*) 'HP, 605,LAB ,, 607/604'
        WRITE(1,*) 'HP, 606,LAB ,, 635/633'
        WRITE(1,*) 'HP, 601,LAB ,, 625/626'
        WRITE(1,*) 'HP, 607,LAB ,, 607/606'
        WRITE(1,*) 'HP, 608,LAB ,, 635/636'
        WRITE(1,*) 'HP, 602,LAB ,, 625/623'
        WRITE(1,*) 'HP, 611,LAB ,, 607/608'
        WRITE(1,*) 'HP, 612,LAB ,, 635/642'
        WRITE(1,*) 'HP, 604,LAB ,, 625/631'
        WRITE(1,9998)
        WRITE(1,*) 'HP, 613,LAB ,, 615/616'
        WRITE(1,*) 'HP, 618,LAB ,, 646/645'
        WRITE(1,*) 'HP, 617,LAB ,, 656/654'
        WRITE(1,*) 'HP, 614,LAB ,, 615/614'
        WRITE(1,*) 'HP, 620,LAB ,, 646/644'
        WRITE(1,*) 'HP, 619,LAB ,, 656/655'
        WRITE(1,*) 'HP, 616,LAB ,, 615/621'
        WRITE(1,*) 'HP, 624,LAB ,, 646/652'
```

```
      WRITE(1,*) 'HP, 623,LAB ,, 656/662'
      WRITE(1,9998)
      WRITE(1,*) 'HP, 641,LAB ,, 690/686'
      WRITE(1,*) 'HP, 642,LAB ,, 715/713'
      WRITE(1,*) 'HP, 637,LAB ,, 705/706'
      WRITE(1,*) 'HP, 625,LAB ,, 670/667'
      WRITE(1,*) 'HP, 630,LAB ,, 669/668'
      WRITE(1,*) 'HP, 629,LAB ,, 689/685'
      WRITE(1,9998)
      WRITE(1,*) 'HP, 631,LAB ,, 690/694'
      WRITE(1,*) 'HP, 632,LAB ,, 689/693'
      WRITE(1,*) 'HP, 626,LAB ,, 669/673'
      WRITE(1,*) 'HP, 638,LAB ,, 670/674'
      WRITE(1,*) 'HP, 644,LAB ,, 705/708'
      WRITE(1,*) 'HP, 643,LAB ,, 715/717'
      WRITE(1,9998)
      WRITE(1,*) 'HP, 635,LAB ,, 690/686'
      WRITE(1,*) 'HP, 636,LAB ,, 689/685'
      WRITE(1,*) 'HP, 628,LAB ,, 669/668'
      WRITE(1,*) 'HP, 640,LAB ,, 670/667'
      WRITE(1,*) 'HP, 648,LAB ,, 705/706'
      WRITE(1,*) 'HP, 647,LAB ,, 715/713'
C
      CLOSE(1)
C
C     CREATING "SLEx.SES" FILE
C     ------------------------
C
      OPEN ( UNIT=1,
     &       FILE='SLE'//ID//'.SES',
     &       STATUS='NEW',
     &       ACCESS='SEQUENTIAL',
     &       CARRIAGECONTROL='LIST',
     &       FORM='FORMATTED' )
C
      WRITE(1,*) 'SET,LINES,0'
C
      WRITE(1,9998)
      WRITE(1,9997) '   SLEEVE'
```

```
      WRITE(1,9998)
      WRITE(1,9997) '   GR 1701,1702 DEFINE AN AXIS'
      WRITE(1,9998)
      WRITE(1,*) 'GR,1701,,0'
      WRITE(1,*) 'GR,1702,,0,1'
      WRITE(1,*) 'GR,1701T1702,ER'
      WRITE(1,9998)
   SOD2D = SOD/2.
      WRITE(1,9998)
      WRITE(1,7010)PD2D
7010  FORMAT( 'GR,#,,  ',F9.6)
   SLD2D = SLD/2.
      WRITE(1,7020) SLD2D
7020  FORMAT('GR,#,,',F9.6)
      WRITE(1,7030) SOD2D
7030  FORMAT('GR,#,,  ',F9.6)
      WRITE(1,*) 'LI,1701T1702,ST,,1703/1704,1704/1705'
      WRITE(1,7040) BT2D
7040  FORMAT('LI,2#,TR,0/ ',F9 6,',1701T1702')
      WRITE(1,7050) BSTM
7050  FORMAT('LI,2#,TR,0/',F9 6,',1703T1704')
      WRITE(1,7060) WT
7060  FORMAT('LI,2#,TR,0/ ',F9.6,',1703T1704')
      WRITE(1,9998)
      WRITE(1,*) 'PA,1701T1703,2L,,1701/1702/1705,1705/1706/1703'
      WRITE(1,*) 'PA,3#,2L,,1706/1703/1704,1704/1707/1708'
      WRITE(1,*) 'PA,1707T1712,MI,X,1701T1706'
      WRITE(1,9998)
      WRITE(1,7070) ANGL1
7070  FORMAT('HP,701T706,ARC,
    &G1701/G1702/',F11.6,',1701/1703/1705')
      WRITE(1,7080) ANGL2
7080  FORMAT('HP,3#,ARC,G1701/G1702/',F11.6,',1707/1709/1711')
      WRITE(1,*) 'HP,9#,MI,Z,701T709'
      WRITE(1,*) 'HP,18#,MI,Y,701T718'
      WRITE(1,7090) ANGL1
7090  FORMAT('HP,6#,ARC,G1701/G1702/',F11.6,',1702/1704/1706')
      WRITE(1,7100) ANGL2
7100  FORMAT('HP,3#,ARC,G1701/G1702/',F11.6,',1708/1710/1712')
```

```
      WRITE(1,*) 'LI,1T#,DEL'
      WRITE(1,*) 'PA,1T#,DEL'
      WRITE(1,9998)
      WRITE(1,*) 'HP,9#,MI,Z,737T745'
      WRITE(1,*) 'HP,18#,MI,Y,737T784'
      WRITE(1,9998)
C
      CLOSE(1)
C
C     CREATING "WASx SES" FILE
C     ------------------------
C
      OPEN ( UNIT=1,
     &       FILE='WAS'//ID//'.SES',
     &       STATUS='NEW',
     &       ACCESS='SEQUENTIAL',
     &       CARRIAGECONTROL='LIST',
     &       FORM='FORMATTED' )
C
      WRITE(1,*) 'SET,LINES,0'
C
      WRITE(1,9998)
      WRITE(1,9997) '    WASHER'
      WRITE(1,9998)
      WRITE(1,9997) '    GR 801,802 DEFINE AN AXIS'
      WRITE(1,9998)
      WRITE(1,*) 'GR,801,,0'
      WRITE(1,*) 'GR,802,,0,1'
      WRITE(1,*) 'GR,801T802,ER'
      WRITE(1,9998)
      WRITE(1,8010) SOD2D,BT2D
8010  FORMAT('GR,#,,',F9.6,',',F9.6)
      WRITE(1,8020) PHD2D,BT2D
8020  FORMAT('GR,#,,',F9.6,',',F9 6)
      WRITE(1,8030) BBD2D,BT2D
8030  FORMAT('GR,#,,',F9.6,',',F9.6)
      WRITE(1,9998)
      WRITE(1,*)'LI,801T802,ST,,803/804,804/805'
      WRITE(1,8040) WT
```

```
8040    FORMAT('LI,2#,TR,/',F9.6,',801T802')
        WRITE(1,*)'PA,801T802,2L,,801T802,803T804'
        WRITE(1,*)'PA,2#,MI,X,801T802'
        WRITE(1,9998)
        WRITE(1,8050) ANGL1
8050    FORMAT('HP,801T804,ARC,G801/G802/',F11.6,',801T802')
        WRITE(1,8060) ANGL2
8060    FORMAT('HP,2#,ARC,G801/G802/',F11.6,',803T804')
        WRITE(1,9998)
        WRITE(1,*)'PA,1T#,DEL'
        WRITE(1,*)'LI,1T#,DEL'
        WRITE(1,9998)
        WRITE(1,*)'HP,6#,MI,Z,801T806'
        WRITE(1,*)'HP,12#,MI,Y,801T812'
        WRITE(1,9998)
        WRITE(1,*)'HP, 801 ,LAB ,, 806/803'
        WRITE(1,*)'HP, 802 ,LAB ,, 815/816'
        WRITE(1,*)'HP, 805 ,LAB ,, 819/820'
        WRITE(1,*)'HP, 811 ,LAB ,, 812/809'
        WRITE(1,*)'HP, 808 ,LAB ,, 832/831'
        WRITE(1,*)'HP, 807 ,LAB ,, 828/827'
        WRITE(1,9998)
        WRITE(1,*)'HP, 803 ,LAB ,, 807/804'
        WRITE(1,*)'HP, 804 ,LAB ,, 817/818'
        WRITE(1,*)'HP, 806 ,LAB ,, 821/822'
        WRITE(1,*)'HP, 812 ,LAB ,, 811/810'
        WRITE(1,*)'HP, 810 ,LAB ,, 834/833'
        WRITE(1,*)'HP, 809 ,LAB ,, 830/829'
        WRITE(1,9998)
        WRITE(1,*)'HP, 819 ,LAB ,, 842/839'
        WRITE(1,*)'HP, 820 ,LAB ,, 863/864'
        WRITE(1,*)'HP, 823 ,LAB ,, 867/868'
        WRITE(1,*)'HP, 817 ,LAB ,, 859/857'
        WRITE(1,*)'HP, 814 ,LAB ,, 848/847'
        WRITE(1,*)'HP, 813 ,LAB ,, 841/840'
        WRITE(1,9998)
        WRITE(1,*)'HP, 821 ,LAB ,, 846/843'
        WRITE(1,*)'HP, 822 ,LAB ,, 865/866'
        WRITE(1,*)'HP, 824 ,LAB ,, 869/870'
```

```
          WRITE(1,*)'HP, 818 ,LAB ,, 860/858'
          WRITE(1,*)'HP, 816 ,LAB ,, 850/849'
          WRITE(1,*)'HP, 815 ,LAB ,, 845/844'
C
      CLOSE(1)
C
C     CREATING "PINx.SES" FILE
C     ------------------------
C
      OPEN ( UNIT=1,
     &        FILE='PIN'//ID//'.SES',
     &        STATUS='NEW',
     &        ACCESS='SEQUENTIAL',
     &        CARRIAGECONTROL='LIST',
     &        FORM='FORMATTED' )
C
          WRITE(1,*) 'SET,LINES,0'
C
C
          WRITE(1,9998)
          WRITE(1,9997) '   PIN'
          WRITE(1,9998)
          WRITE(1,*) 'GR,901,,0,0'
          WRITE(1,9001) BT2D
 9001      FORMAT('GR,902,,0,',F9.6)
          WRITE(1,9002) BSTM
 9002      FORMAT('GR,903,TR,0/',F9.6,',902')
          WRITE(1,9902) WT
 9902      FORMAT('GR,904,TR,0/',F9.6,',902')
          WRITE(1,9003) BBT
 9003      FORMAT('GR,905,TR,0/',F9.6,',904')
          WRITE(1,9004) LT1
 9004      FORMAT('GR,906,TR,0/',F9.6,',905')
          WRITE(1,9005) PHT
 9005      FORMAT('GR,907,TR,0/',F9.6,',906')
          WRITE(1,*) 'LI,901T906,ST,,
     &901/903/902/904T906,903/902/904T907'
          WRITE(1,9006) PD2D
 9006      FORMAT('LI,6#,TR,',F9.6,',901T906')
```

```
         WRITE(1,9007) BODPDM2D
 9007     FORMAT('LI,#,TR,',F9.6,',912')
C
         WRITE(1,9008) PHD2D
 9008     FORMAT('LI,#,TR,',F9.6,',906')
         WRITE(1,*) 'PA,901T908,2L,,
    &901T906/912/913,907T912/913/914'
         WRITE(1,9009) ANGL1
 9009     FORMAT('HP,901T916,ARC,G901/G902/',F11.6,',901T908')
         WRITE(1,*) 'PA,8#,MI,X,901T908'
         WRITE(1,9010) ANGL2
 9010     FORMAT('HP,8#,ARC,G901/G902/',F11.6,',909T916')
         WRITE(1,*) 'LI,1T#,D'
         WRITE(1,*) 'PA,1T#,D'
         WRITE(1,9998)
         WRITE(1,*) 'HP,24#,MI,Z,901T924'
         WRITE(1,*) 'HP,48#,MI,Y,901T948'
         WRITE(1,9998)
C
      END
```

Appendix B

Finite Element Definition Program

The FED program helps to define the F.E. of the lug/pin joint geometry. The input file must have the same format as the FED.DAT file (Table B.1). The output files are PATRAN session files containing the node definition statements (GFEG) and the connectivity definition statements (CFEG) used to define the finite elements [6].

They are named:

FELUGx.SES
FEBUSx.SES
FEWASx.SES
FEPINx.SES
FESLEx.SES
FEBLAx.SES

where "x" is a user chosen one digit ID.

Note that before creating the output files, the FED program will compute the numbers of F.E. of every lug/pin joint component and prompt to know whether the user is satisfied. The FED output files can be edited and modified to fit the user's needs. After the FED output files have been executed by PATRAN, the user may have to modify some of the elements whose skewness would bring NASTRAN to compute inaccurate results. Moreover, the washer F.E. mesh is a transition between the blade and the bushings. It may have to be modified within PATRAN so that the nodes at the bushing-washer and blade-washer interfaces match.

The elements created by the FED program are numbered according to the following scheme:

lug elements	: from 1001 to a maximum of 3999
bushing elements	: from 6001 to a maximum of 6999
pin elements	: from 9001 to a maximum of 10999
washer elements	: from 8001 to a maximum of 8999
sleeve elements	: from 7001 to a maximum of 7499
sleeve lining elements	: from 7501 to a maximum of 7999
blade elements	: from 4001 to a maximum of 5999

The material property ID's given to the elements are

lug elements	1
bushing elements	6
pin elements	9
washer elements	8
sleeve elements	: 71
sleeve lining elements	: 72
blade elements	4

To run FED, type.

```
$RUN FED
 ENTER INPUT FILE NAME.                  <--- written by FED
 FED.DAT                                 <--- example, typed by the user
 ENTER OUTPUT FILES ID (ONE DIGIT):      <--- written by FED
 1                                       <--- example, typed by the user
```

```
*
*      LUG & PIN F E  DEFINITTION (refer to Figure B 1)
*
L1 =2
L21=1
L22=1
L3 =1
L4 =2
L5 =1
L6 =2
L7 =1
*
B1 =1
B2 =2
*
P1 =1
P2 =1
P21=1
P3 =1
*
W1 =1
*
D1 =1
D2 =1
D21=1
D22=1
D3 =1
D4 =1
D5 =1
D6 =1
*
S1 =1
S2 =1
*
```

Table B.1: Typical FED program input file

The format is as follows: only the one digit integer right to the equal sign can be changed, the first 3 lines are available for comments, no line can be added or deleted. The variables L1, L2,... S2 define the numbers of elements the user wants to have for every hyperpatch (refer to Figure B.1).

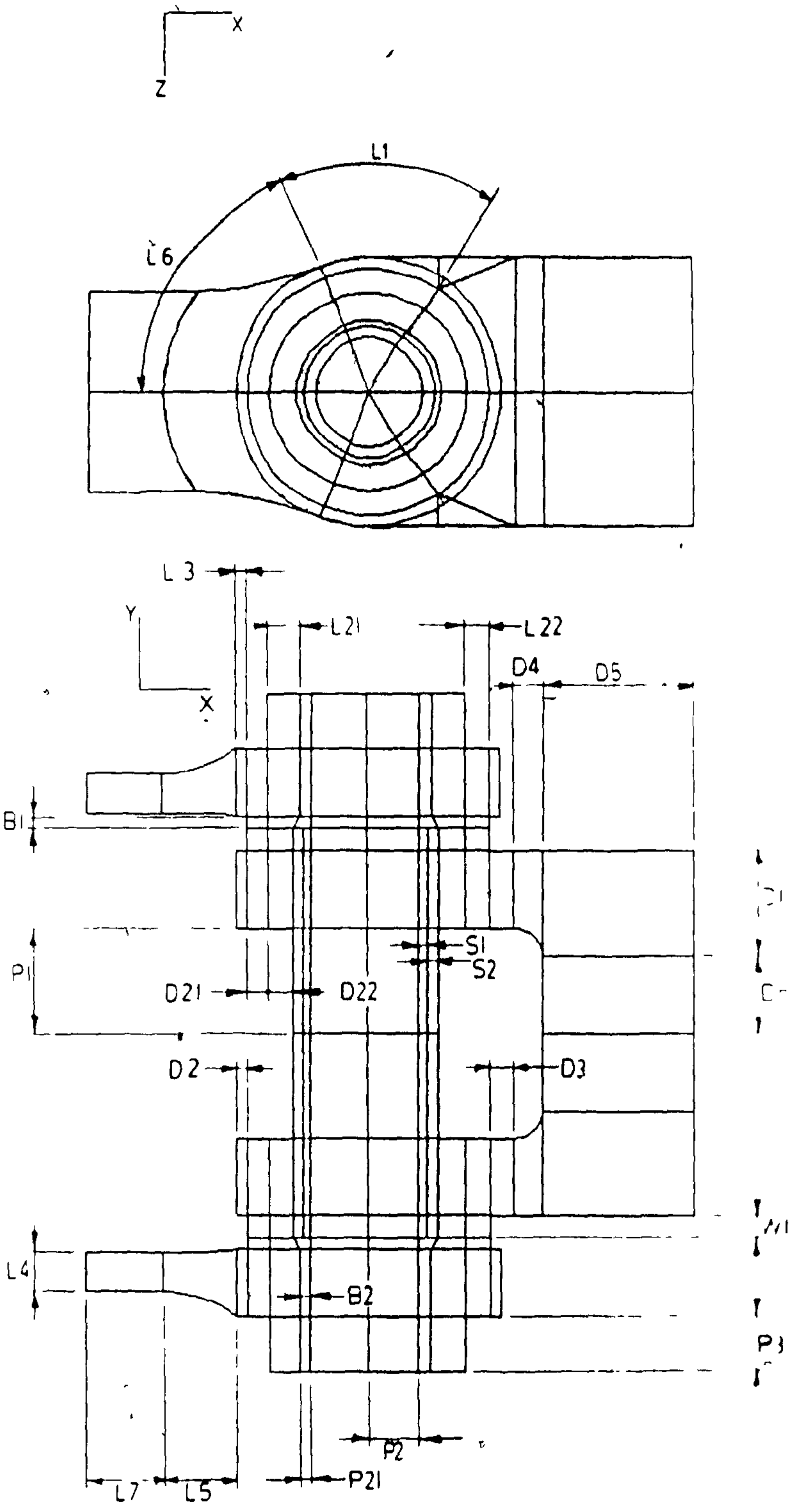


Figure B.1: F.E. mesh definition variables

See Table B.1

```
$
$
$                         PIN F E. DEFINITION
$
GF,H901/902/973/974,,2/3/2,2
GF,H903/904/975/976,,2/3/2,2
GF,H905/906/977/978,,2/3/1,2
GF,H907/908/979/980,,2/3/2,2
GF,H909/910/981/982,,2/3/2,2
GF,H911/912/983/984,,2/3/2,2
GF,H913/914/985/986,,2/3/2,2
GF,H915/916/987/988,,2/3/2,2
GF,H925/926/949/950,,3/2/2,2
GF,H927/928/951/952,,3/2/2,2
GF,H929/930/953/954,,3/2/1,2
GF,H931/932/955/956,,3/2/2,2
GF,H933/934/957/958,,3/2/2,2
GF,H935/936/959/960,,3/2/2,2
GF,H937/938/961/962,,3/2/2,2
GF,H939/940/963/964,,3/2/2,2
GF,H917/989,,2/3/2,2
GF,H918/990,,2/3/2,2
GF,H919/991,,1/3/2,2
GF,H920/992,,2/3/2,2
GF,H921/993,,2/3/2,2
GF,H922/994,,2/3/2,2
GF,H923/995,,2/3/2,2
GF,H924/996,,2/3/2,2
GF,H941/965,,3/2/2,2
GF,H942/966,,3/2/2,2
GF,H943/967,,3/1/2,2
GF,H944/968,,3/2/2,2
GF,H945/969,,3/2/2,2
GF,H946/970,,3/2/2,2
GF,H947/971,,3/2/2,2
GF,H948/972,,3/2/2,2
 CF,H901T996,HEX,,M9,9001T10999
```

Table B.2: Typical FED program output file.

This particular file was used to generate the pin F.E. mesh.

Table B.3: FED program FORTRAN code

```
C
C     LUG & PIN, FINITE ELEMENT DEFINITION PROGRAM
C
      PROGRAM FED
C     -----------
C
C
      INTEGER L1,L3,L4,L5,L6,L7,B1,B2,P1,P2,P3, P21,
     &         W1,D1,D2,D3,D4,D5,D6, L21,L22, D21,D22, S1,S2
      INTEGER M1,M3,M4,M5,M6,M7,C1,C2,Q1,Q2,Q3, Q21,
     &         X1,E1,E2,E3,E4,E5,E6, M21,M22, E21,E22, T1,T2
      CHARACTER*1 A1,OUTPUT
      CHARACTER*50 INPUT
C
      WRITE(6,2)
 2     FORMAT(' ENTER INPUT FILE NAME: ')
      READ(5,3) INPUT
C
      WRITE(6,4)
 4     FORMAT(' ENTER OUTPUT FILES ID (ONE DIGIT)  ')
      READ(5,3) OUTPUT
C
 3     FORMAT(A)
 9998  FORMAT('$',A40)
 9999  FORMAT('$')
C
      OPEN ( UNIT=2,
     &       FILE=INPUT,
     &       STATUS='OLD',
     &       ACCESS='SEQUENTIAL',
     &       FORM='FORMATTED',
     &       READONLY )
C
C
C      READING DATA FROM INPUT FILE
C      ----------------------------
```

```
C
    1     FORMAT(///8(4X,I1/),/,2(4X,I1/),/,
     &          4(4X,I1/),/,4X,I1//,8(4X,I1/),/,2(4X,I1/) )
          READ(2,1)L1,L21,L22,L3,L4,L5,L6,L7,B1,B2,P1,P2,P21,P3
     &          ,W1,D1,D2,D21,D22,D3,D4,D5,D6,S1,S2
C
C
C         COMPUTING THE NUMBER OF ELEMENTS
C         --------------------------------
C
          NLUGS = 4*(L4*L6*(L5+L7) +
     &               (L6+2*L1)*(L4*(L21+L22+L3)+2) )
          NBUSH = 4*( (L6+2*L1)*(B1*(L21+L22+B2)+(B2*L4)+6) )
          NPIN  = 8*L1*(P2*(P1+D1+W1+B1+L4+P3) + (B2+L21)*P3 ) +
     &            4*L6*(P2*(P1+D1+W1+B1+L4+P3) + (B2+L21)*P3 )
          NWASH = 4*((2*L1+L6)*((D21+D22)*W1+4))
          NSLEE = 4*((L6+2*L1)*((S1+S2)*(P1+D1+W1)))
          NBLAD = (L6+2*L1)*(D1*(D2+D21+D22)) +
     &            D1*(L1*D2 + D3*D2 + D3*L1 + L1*D4 + L1*D5) +
     &            D6*D5*L1
C
          NTOT  = NLUGS + NBUSH + NPIN + NWASH + NSLEE + NBLAD
C
   21     FORMAT('                        BOTH LUGS     = ',I5,
     &'   FROM   1001 TO ',I5)
   22     FORMAT('                        BOTH BUSHINGS = ',I5,
     &'   FROM   6001 TO ',I5)
   23     FORMAT('                        PIN           = ',I5,
     &'   FROM   9001 TO ',I5)
   24     FORMAT('                        BOTH WASHERS  = ',I5,
     &'   FROM   8001 TO ',I5)
   25     FORMAT('                        SLEEVE        = ',I5,
     &'   FROM   7001 TO ',I5)
   26     FORMAT('                        BLADE         = ',I5,
     &'   FROM   4001 TO ',I5)
   27     FORMAT('                        TOTAL         = ',I5)
C
          WRITE(6,*)' '
          WRITE(6,*) ' APPROX NUMBER OF ELEMENTS'
```

```
      NLUFIN=1000+NLUGS
      WRITE(6,21) NLUGS,NLUFIN
C
      NBUFIN=6000+NBUSH
      WRITE(6,22) NBUSH,NBUFIN
C
      NPIFIN=9000+NPIN
      WRITE(6,23) NPIN,NPIFIN
C
      NWAFIN=8000+NWASH
      WRITE(6,24) NWASH,NWAFIN
C
      NSLFIN=7000+NSLEE
      WRITE(6,25) NSLEE,NSLFIN
C
      NBLFIN=4000+NBLAD
      WRITE(6,26) NBLAD,NBLFIN
C
      WRITE(6,*) '                                    -----------------------'
      WRITE(6,27) NTOT
      WRITE(6,*)' '
      WRITE(6,*)' '
      WRITE(6,*)' '
C
      WRITE(6,*)' PROCEED ? Y/N'
C
 5    FORMAT(A1)
      READ (5,5) A1
      IF (A1.EQ.'N') STOP
C
C     NUMBER OF NODES
C     ---------------
C
      M1=L1+1
      M21=L21+1
      M22=L22+1
      M3=L3+1
      M4=L4+1
      M5=L5+1
```

```
      M6=L6+1
      M7=L7+1
C
      C1=B1+1
      C2=B2+1
C
      Q1=P1+1
      Q2=P2+1
      Q21=P21+1
      Q3=P3+1
C
      X1=W1+1
C
      T1=S1+1
      T2=S2+1
C
      E1=D1+1
      E2=D2+1
      E21=D21+1
      E22=D22+1
      E3=D3+1
      E4=D4+1
      E5=D5+1
      E6=D6+1
C
C     CREATES THE OUTPUT FILES
C     ------------------------
C

      OPEN ( UNIT=1,
     &       FILE='FELUG'//OUTPUT//'.SES',
     &       STATUS='NEW',
     &       ACCESS='SEQUENTIAL',
     &       CARRIAGECONTROL='LIST',
     &       FORM='FORMATTED' )
C
C
      WRITE (1,9999)
      WRITE (1,9999)
C
```

```
      WRITE (1,9998)'        LUG F.E. DEFINITION'
C
C
 103   FORMAT('GF, H103/114/125/136 ,,',I1,'/',I1,'/',I1,',2,ED9')
 128   FORMAT('GF, H128/129/139/140/106/107/117/118 ,, '
     &        ,I1,'/',I1,'/',I1,',2,ED9')
C
 104   FORMAT('GF, H104/115/126/137 ,, ',I1,'/',I1,'/',I1,',2')
 108   FORMAT('GF, H108/109/119/120/130/131/141/142 ,, '
     &        ,I1,'/',I1,'/',I1,',2')
C
 101   FORMAT('GF, H101/123 ,, ',I1,'/',I1,'/',I1,',2')
 102   FORMAT('GF, H102/135 ,, ',I1,'/',I1,'/',I1,',2')
 105   FORMAT('GF, H105/138 ,, ',I1,'/',I1,'/',I1,',2')
 110   FORMAT('GF, H110/111/143/144 ,, ',I1,'/',I1,'/',I1,',2')
C
 112   FORMAT('GF, H112/134 ,, ',I1,'/',I1,'/',I1,',2')
 113   FORMAT('GF, H113/124 ,, ',I1,'/',I1,'/',I1,',2')
 116   FORMAT('GF, H116/127 ,, ',I1,'/',I1,'/',I1,',2')
 121   FORMAT('GF, H121/122/132/133 ,, ',I1,'/',I1,'/',I1,',2')
C
C
      WRITE(1,103) M4,M21,M6
      WRITE(1,128) M4,M21,M1
C
      WRITE(1,104) M4,M22,M6
      WRITE(1,108) M4,M22,M1
C
      WRITE(1,101) M5,M6,M4
      WRITE(1,102) M6,M7,M4
      WRITE(1,105) M4,M6,M3
      WRITE(1,110) M3,M1,M4
C
      WRITE(1,112) M6,M5,M4
      WRITE(1,113) M7,M6,M4
      WRITE(1,116) M6,M4,M3
      WRITE(1,121) M1,M3,M4
C
      WRITE(1,*)'CF,H101T144,HEX,,M1,1001T3999'
```

```
C
      CLOSE(1)
C
      OPEN ( UNIT=1,
     &       FILE='FEBUS'//OUTPUT//'.SES',
     &       STATUS='NEW',
     &       ACCESS='SEQUENTIAL',
     &       CARRIAGECONTROL='LIST',
     &       FORM='FORMATTED' )
C
C
      WRITE(1,9999)
      WRITE(1,9998)'       BUSHING F.E. DEFINITION'
      WRITE(1,9999)
C
 601  FORMAT('GF,H601/637/613/625 ,,',I1,'/',I1,'/',I1,',2,ED9')

 605  FORMAT('GF,H605/606/641/642/617/618/629/630 ,,',
     &       I1,'/',I1,'/',I1,',2,ED9')
C
 602  FORMAT('GF,H602/614 ,,',I1,'/',I1,'/',I1,',2,ED9')
 604  FORMAT('GF,H604/616 ,,',I1,'/',I1,'/',I1,',2,ED9')
 607  FORMAT('GF,H607/608/619/620 ,,',I1,'/',I1,'/',I1,',2,ED9')
 611  FORMAT('GF,H611/612/623/624 ,,',I1,'/',I1,'/',I1,',2,ED9')
C
 638  FORMAT('GF,H638/626 ,,',I1,'/',I1,'/',I1,',2,ED9')
 640  FORMAT('GF,H640/628 ,,',I1,'/',I1,'/',I1,',2,ED9')
 643  FORMAT('GF,H643/644/631/632 ,,',I1,'/',I1,'/',I1,',2,ED9')
 647  FORMAT('GF,H647/648/635/636 ,,',I1,'/',I1,'/',I1,',2,ED9')
C
 621  FORMAT('GF,H621/622/633/634 ,,',I1,'/',I1,'/',I1,',2')
 609  FORMAT('GF,H609/610/645/646 ,,',I1,'/',I1,'/',I1,',2')
C
 603  FORMAT('GF,H603/639 ,,',I1,'/',I1,'/',I1,',2')
 615  FORMAT('GF,H615/627 ,,',I1,'/',I1,'/',I1,',2')
C
      WRITE(1,601) C1,C2,M6
      WRITE(1,605) C1,C2,M1
C
```

```
      WRITE(1,602) M21,C1,M6
      WRITE(1,604) C2,M4,M6
      WRITE(1,607) M21,C1,M1
      WRITE(1,611) C2,M4,M1
C
      WRITE(1,638) C1,M21,M6
      WRITE(1,640) M4,C2,M6
      WRITE(1,643) C1,M21,M1
      WRITE(1,647) M4,C2,M1
C
      WRITE(1,621) C1,M1,M22
      WRITE(1,609) M1,C1,M22
C
      WRITE(1,603) M6,M22,C1
      WRITE(1,615) M22,M6,C1
C
      WRITE(1,*)'CF,H601T648,HEX,,M6,6001T6999'
C
      CLOSE(1)
C
      OPEN ( UNIT=1,
     &       FILE='FEPIN'//OUTPUT//'.SES',
     &       STATUS='NEW',
     &       ACCESS='SEQUENTIAL',
     &       CARRIAGECONTROL='LIST',
     &       FORM='FORMATTED' )
C
C
      WRITE (1,9999)
      WRITE (1,9998)'      PIN F.E. DEFINITION'
      WRITE (1,9999)
C
C
 901  FORMAT('GF,H901/902/973/974,,',I1,'/',I1,'/',I1,',2')
 903  FORMAT('GF,H903/904/975/976,,',I1,'/',I1,'/',I1,',2')
 905  FORMAT('GF,H905/906/977/978,,',I1,'/',I1,'/',I1,',2')
 907  FORMAT('GF,H907/908/979/980,,',I1,'/',I1,'/',I1,',2')
 909  FORMAT('GF,H909/910/981/982,,',I1,'/',I1,'/',I1,',2')
 911  FORMAT('GF,H911/912/983/984,,',I1,'/',I1,'/',I1,',2')
```

```
 913   FORMAT('GF,H913/914/985/986,,',I1,'/',I1,'/',I1,',2')
 915   FORMAT('GF,H915/916/987/988,,',I1,'/',I1,'/',I1,',2')
C
 925   FORMAT('GF,H925/926/949/950,,',I1,'/',I1,'/',I1,',2')
 927   FORMAT('GF,H927/928/951/952,,',I1,'/',I1,'/',I1,',2')
 929   FORMAT('GF,H929/930/953/954,,',I1,'/',I1,'/',I1,',2')
 931   FORMAT('GF,H931/932/955/956,,',I1,'/',I1,'/',I1,',2')
 933   FORMAT('GF,H933/934/957/958,,',I1,'/',I1,'/',I1,',2')
 935   FORMAT('GF,H935/936/959/960,,',I1,'/',I1,'/',I1,',2')
 937   FORMAT('GF,H937/938/961/962,,',I1,'/',I1,'/',I1,',2')
 939   FORMAT('GF,H939/940/963/964,,',I1,'/',I1,'/',I1,',2')
C
 917   FORMAT('GF,H917/989,,',I1,'/',I1,'/',I1,',2')
 918   FORMAT('GF,H918/990,,',I1,'/',I1,'/',I1,',2')
 919   FORMAT('GF,H919/991,,',I1,'/',I1,'/',I1,',2')
 920   FORMAT('GF,H920/992,,',I1,'/',I1,'/',I1,',2')
 921   FORMAT('GF,H921/993,,',I1,'/',I1,'/',I1,',2')
 922   FORMAT('GF,H922/994,,',I1,'/',I1,'/',I1,',2')
 923   FORMAT('GF,H923/995,,',I1,'/',I1,'/',I1,',2')
 924   FORMAT('GF,H924/996,,',I1,'/',I1,'/',I1,',2')
C
 941   FORMAT('GF,H941/965,,',I1,'/',I1,'/',I1,',2')
 942   FORMAT('GF,H942/966,,',I1,'/',I1,'/',I1,',2')
 943   FORMAT('GF,H943/967,,',I1,'/',I1,'/',I1,',2')
 944   FORMAT('GF,H944/968,,',I1,'/',I1,'/',I1,',2')
 945   FORMAT('GF,H945/969,,',I1,'/',I1,'/',I1,',2')
 946   FORMAT('GF,H946/970,,',I1,'/',I1,'/',I1,',2')
 947   FORMAT('GF,H947/971,,',I1,'/',I1,'/',I1,',2')
 948   FORMAT('GF,H948/972,,',I1,'/',I1,'/',I1,',2')
C
      WRITE(1,901) Q2,M1,Q1
      WRITE(1,903) Q2,M1,E1
      WRITE(1,905) Q2,M1,W1
      WRITE(1,907) Q2,M1,C1
      WRITE(1,909) Q2,M1,M4
      WRITE(1,911) Q2,M1,Q3
      WRITE(1,913) Q21,M1,Q3
      WRITE(1,915) M21,M1,Q3
C
```

```
      WRITE(1,925) M1,Q2,Q1
      WRITE(1,927) M1,Q2,E1
      WRITE(1,929) M1,Q2,W1
      WRITE(1,931) M1,Q2,C1
      WRITE(1,933) M1,Q2,M4
      WRITE(1,935) M1,Q2,Q3
      WRITE(1,937) M1,Q21,Q3
      WRITE(1,939) M1,M21,Q3
C
      WRITE(1,917) Q1,M6,Q2
      WRITE(1,918) E1,M6,Q2
      WRITE(1,919) W1,M6,Q2
      WRITE(1,920) C1,M6,Q2
      WRITE(1,921) M4,M6,Q2
      WRITE(1,922) Q3,M6,Q2
      WRITE(1,923) Q3,M6,Q21
      WRITE(1,924) Q3,M6,M21
C
      WRITE(1,941) M6,Q1,Q2
      WRITE(1,942) M6,E1,Q2
      WRITE(1,943) M6,W1,Q2
      WRITE(1,944) M6,C1,Q2
      WRITE(1,945) M6,M4,Q2
      WRITE(1,946) M6,Q3,Q2
      WRITE(1,947) M6,Q3,Q21
      WRITE(1,948) M6,Q3,M21
      WRITE(1,*)'CF,H901T996,HEX,,M9,9001T10999'
C
      CLOSE(1)
C
      OPEN ( UNIT=1,
     &       FILE='FEWAS'//OUTPUT//'.SES',
     &       STATUS='NEW',
     &       ACCESS='SEQUENTIAL',
     &       CARRIAGECONTROL='LIST',
     &       FORM='FORMATTED' )
C
C
      WRITE (1,9999)
```

```
      WRITE (1,9998)'         WASHER F E  DEFINITION'
      WRITE (1,9999)
C
 801  FORMAT('GF,H801/802/808/807,,',I1,'/',I1,'/',I1,',2,ED12')
 805  FORMAT('GF,H805/811,,',I1,'/',I1,'/',I1,',2,ED12')
 803  FORMAT('GF,H803/804/810/809,,',I1,'/',I1,'/',I1,',2,ED9')
 806  FORMAT('GF,H806/812,,',I1,'/',I1,'/',I1,',2,ED9')
 819  FORMAT('GF,H819/820/814/813,,',I1,'/',I1,'/',I1,',2,ED12')
 823  FORMAT('GF,H823/817,,',I1,'/',I1,'/',I1,',2,ED12')
 821  FORMAT('GF,H821/822/816/815,,',I1,'/',I1,'/',I1,',2,ED9')
 824  FORMAT('GF,H824/818,,',I1,'/',I1,'/',I1,',2,ED9')
C
      WRITE(1,801) E21,X1,M1
      WRITE(1,805) E21,X1,M6
      WRITE(1,803) E22,X1,M1
      WRITE(1,806) E22,X1,M6
      WRITE(1,819) E21,X1,M1
      WRITE(1,823) E21,X1,M6
      WRITE(1,821) E22,X1,M1
      WRITE(1,824) E22,X1,M6
C
      WRITE(1,*)'CF,H801T824,HEX,,M8,8001T8999'
C
      CLOSE(1)
C
      OPEN ( UNIT=1,
     &        FILE='FESLE'//OUTPUT//' SES',
     &        STATUS='NEW',
     &        ACCESS='SEQUENTIAL',
     &        CARRIAGECONTROL='LIST',
     &        FORM='FORMATTED' )
C
C
C
      WRITE (1,9999)
      WRITE (1,9998)'         SLEEVE F E  DEFINITION'
      WRITE (1,9999)
C
 701  FORMAT('GF,H701/702/728/729 ,,  ',I1,'/',I1,'/',I1,',2')
```

```
 703  FORMAT('GF,H703/704/730/731 ,, ',I1,'/',I1,'/',I1,',2')
 705  FORMAT('GF,H705/706/732/733 ,, ',I1,'/',I1,'/',I1,',2')
 707  FORMAT('GF,H707/734 ,, ',I1,'/',I1,'/',I1,',2')
 708  FORMAT('GF,H708/735 ,, ',I1,'/',I1,'/',I1,',2')
 709  FORMAT('GF,H709/736 ,, ',I1,'/',I1,'/',I1,',2')
C
 710  FORMAT('GF,H710/711/719/720 ,, ',I1,'/',I1,'/',I1,',2')
 712  FORMAT('GF,H712/713/721/722 ,, ',I1,'/',I1,'/',I1,',2')
 714  FORMAT('GF,H714/715/723/724 ,, ',I1,'/',I1,'/',I1,',2')
 716  FORMAT('GF,H716/725 ,, ',I1,'/',I1,'/',I1,',2')
 717  FORMAT('GF,H717/726 ,, ',I1,'/',I1,'/',I1,',2')
 718  FORMAT('GF,H718/727 ,, ',I1,'/',I1,'/',I1,',2')
C
 737  FORMAT('GF,H737/738/764/765 ,, ',I1,'/',I1,'/',I1,',2')
 739  FORMAT('GF,H739/740/766/767 ,, ',I1,'/',I1,'/',I1,',2')
 741  FORMAT('GF,H741/742/768/769 ,, ',I1,'/',I1,'/',I1,',2')
 743  FORMAT('GF,H743/770 ,, ',I1,'/',I1,'/',I1,',2')
 744  FORMAT('GF,H744/771 ,, ',I1,'/',I1,'/',I1,',2')
 745  FORMAT('GF,H745/772 ,, ',I1,'/',I1,'/',I1,',2')
C
 746  FORMAT('GF,H746/747/755/756 ,, ',I1,'/',I1,'/',I1,',2')
 748  FORMAT('GF,H748/749/757/758 ,, ',I1,'/',I1,'/',I1,',2')
 750  FORMAT('GF,H750/751/759/760 ,, ',I1,'/',I1,'/',I1,',2')
 752  FORMAT('GF,H752/761 ,, ',I1,'/',I1,'/',I1,',2')
 753  FORMAT('GF,H753/762 ,, ',I1,'/',I1,'/',I1,',2')
 754  FORMAT('GF,H754/763 ,, ',I1,'/',I1,'/',I1,',2')
C
C
      WRITE(1,701) M1,Q1,T1
      WRITE(1,703) M1,E1,T1
      WRITE(1,705) M1,X1,T1
      WRITE(1,707) M6,T1,Q1
      WRITE(1,708) M6,T1,E1
      WRITE(1,709) M6,T1,X1
C
      WRITE(1,710) Q1,M1,T1
      WRITE(1,712) E1,M1,T1
      WRITE(1,714) X1,M1,T1
      WRITE(1,716) T1,M6,Q1
```

```
      WRITE(1,717) T1,M6,E1
      WRITE(1,718) T1,M6,X1
C
      WRITE(1,737) M1,Q1,T2
      WRITE(1,739) M1,E1,T2
      WRITE(1,741) M1,X1,T2
      WRITE(1,743) M6,T2,Q1
      WRITE(1,744) M6,T2,E1
      WRITE(1,745) M6,T2,X1
C
      WRITE(1,746) Q1,M1,T2
      WRITE(1,748) E1,M1,T2
      WRITE(1,750) X1,M1,T2
      WRITE(1,752) T2,M6,Q1
      WRITE(1,753) T2,M6,E1
      WRITE(1,754) T2,M6,X1
C
      WRITE(1,*) 'CF,H701T736,HEX,,M71,7001T7499'
      WRITE(1,*) 'CF,H737T772,HEX,,M72,7501T7999'
C
      CLOSE(1)
C
      OPEN ( UNIT=1,
     &        FILE='FEBLA'//OUTPUT//' SES',
     &        STATUS='NEW',
     &        ACCESS='SEQUENTIAL',
     &        CARRIAGECONTROL='LIST',
     &        FORM='FORMATTED' )
C
C
      WRITE (1,9999)
      WRITE (1,9998)'          BLADE F E. DEFINITION'
      WRITE (1,9999)
C
 401  FORMAT('GF,H401/440 ,, ',I1,'/',I1,'/',I1,',2')
 402  FORMAT('GF,H402/441 ,, ',I1,'/',I1,'/',I1,',2')
 403  FORMAT('GF,H403/442 ,, ',I1,'/',I1,'/',I1,',2')
 404  FORMAT('GF,H404/405/443/444 ,, ',I1,'/',I1,'/',I1,',2')
 406  FORMAT('GF,H406/407/445/446 ,, ',I1,'/',I1,'/',I1,',2')
```

```
 408  FORMAT('GF,H408/447 ,, ',I1,'/',I1,'/',I1,',2')
 409  FORMAT('GF,H409/448 ,, ',I1,'/',I1,'/',I1,',2')
 410  FORMAT('GF,H410/449 ,, ',I1,'/',I1,'/',I1,',2')
 411  FORMAT('GF,H411/450 ,, ',I1,'/',I1,'/',I1,',2')
 412  FORMAT('GF,H412/451 ,, ',I1,'/',I1,'/',I1,',2')
 413  FORMAT('GF,H413/452 ,, ',I1,'/',I1,'/',I1,',2')
C
 414  FORMAT('GF,H414/427 ,, ',I1,'/',I1,'/',I1,',2')
 415  FORMAT('GF,H415/428 ,, ',I1,'/',I1,'/',I1,',2')
 416  FORMAT('GF,H416/429 ,, ',I1,'/',I1,'/',I1,',2')
 417  FORMAT('GF,H417/418/430/431 ,, ',I1,'/',I1,'/',I1,',2')
 419  FORMAT('GF,H419/420/432/433 ,, ',I1,'/',I1,'/',I1,',2')
 421  FORMAT('GF,H421/434 ,, ',I1,'/',I1,'/',I1,',2')
 422  FORMAT('GF,H422/435 ,, ',I1,'/',I1,'/',I1,',2')
 423  FORMAT('GF,H423/436 ,, ',I1,'/',I1,'/',I1,',2')
 424  FORMAT('GF,H424/437 ,, ',I1,'/',I1,'/',I1,',2')
 425  FORMAT('GF,H425/438 ,, ',I1,'/',I1,'/',I1,',2')
 426  FORMAT('GF,H426/439 ,, ',I1,'/',I1,'/',I1,',2')
C
C
      WRITE(1,401) M6,E21,E1
      WRITE(1,402) M6,E22,E1
      WRITE(1,403) M6,E2,E1
      WRITE(1,404) M1,E21,E1
      WRITE(1,406) M1,E22,E1
      WRITE(1,408) M1,E2,E1
      WRITE(1,409) M1,E3,E1
      WRITE(1,410) E3,E2,E1
      WRITE(1,411) E4,M1,E1
      WRITE(1,412) E5,M1,E1
      WRITE(1,413) E5,M1,E6
C
      WRITE(1,414) E21,M6,E1
      WRITE(1,415) E22,M6,E1
      WRITE(1,416) E2,M6,E1
      WRITE(1,417) E21,M1,E1
      WRITE(1,419) E22,M1,E1
      WRITE(1,421) E2,M1,E1
      WRITE(1,422) E3,M1,E1
```

```
      WRITE(1,423) E2,E3,E1
      WRITE(1,424) M1,E4,E1
      WRITE(1,425) M1,E5,E1
      WRITE(1,426) M1,E5,E6
C
      WRITE(1,*)'CF,H401T452,HEX,,M4,4001T5999'
      WRITE (1,9999)
      WRITE (1,9999)
C
      CLOSE(1)
C
      END
```

Appendix C

Node Offsetting Program

This program is used to modify small details of a F.E model that were not specified in the geometrical definition. It uses the OFFSET option of the PATRAN NODE command ([6], PATRAN user's guide, section 13.2.6) to perform that task.

The input file must have the format of the file OFFSET.DAT (Table C.1). It contains:

1. A PATRAN neutral file ("PATRAN.OUT" file).

2. Zones whose nodes have to be offset (See Figure C.1). They are defined in a cylindrical coordinates (R, θ, Z') where:

$$\begin{aligned} \text{R1} &< R < \text{R2}, \\ \text{THETA1} &< \theta < \text{THETA2}, \\ \text{Z'1} &< Z' < \text{Z'2}. \end{aligned}$$

The R, θ, Z' coordinates are related to the basic coordinate system X, Y, Z by:

$$R = \sqrt{X^2 + Z^2} \tag{C.1}$$

$$\theta = \arctan \frac{-Z}{X} \tag{C.2}$$

$$Z' = Y \tag{C.3}$$

Zones in the input file are defined using free format.

3. The offset values are given by DR, DZ'. There is no offset value allowed in θ-direction.

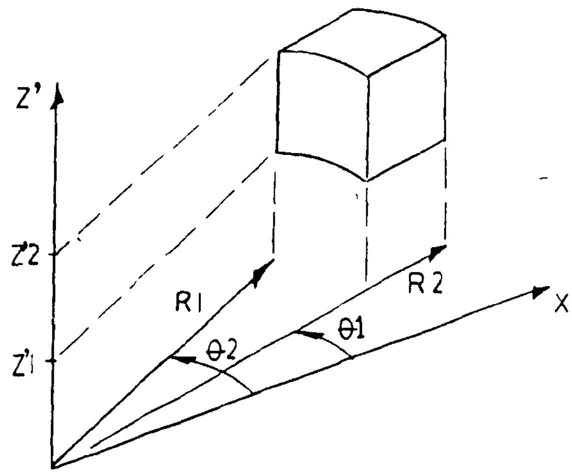


Figure C.1: Definition of a Zone whose Nodes need to be Offset

Recall that the NODE OFFSET instruction does modify the node's actual position without modifying its parametric "(ξ_1, ξ_2)" coordinates.

To run OFFSET, type:

```
$RUN OFFSET
 ENTER INPUT FILE NAME:      <--- written by OFFSET
 OFFSET.DAT                  <--- example, typed by the user
 ENTER OUTPUT FILE NAME:     <--- written by OFFSET
 OFFSET.SES                  <--- example, typed by the user
```

```
*
*    BUSH FLANGE ROOT MODEL
*
PATRAN.OUT FILE :[MCGO1.F]PATRAN.OUT
*
*     R1      R2       THETA1     THETA2       Z'1    Z'2      DR   DZ'
*
  .599     .601      -360.        360.     1.970 1.972 .014645 .014645
 .582 .601    -360.  360.  2.016 2.019   0. .00358
 .674 .676    -360.  360.  1.970 1.972 -.025 0.
 .764 .767    -360.  360.  1.921 1.924 -.014375 0.
 .674 .676    -360.  360.  2.016 2.019 -.05 -.02142
 .684 .687    -360  360.  1.945 1.947 -.025 0.
```

Table C.1: Typical OFFSET program input file

- The first 3 lines are available for comments.
- On the 4th line, the neutral file name is entered right to ":".
- The 5th and 7th lines are for comments.
- The 6th line indicates the order that must be used to enter the rest of the data.
- Each of the other lines described a zone whose nodes have to be "offset" and by how much they have to be. Each zone is described by 6 limits R1,R2, THETA1,THETA2, Z'1,Z'2 entered in that order. The offset values "DR" and "DZ'" are entered right to the Z'2 value. Free format is used.

```
NOD,     826,OFF,-0.005585, 0.014645,-0.013538
NOD,     894,OFF,-0.009313, 0.014645,-0.011302
NOD,     895,OFF,-0.002104, 0.014645,-0.014493
NOD,     988,OFF,-0.012171, 0.014645,-0.008145
NOD,     989,OFF, 0.001451, 0.014645,-0.014573
NOD,    1082,OFF,-0.014000, 0.014645,-0.004299
NOD,    1085,OFF, 0.004922, 0.014645,-0.013793
NOD,    1173,OFF,-0.014645, 0.014645, 0.000000
NOD,    1184,OFF, 0.008145, 0.014645,-0.012171
NOD,    1249,OFF,-0.014000, 0.014645, 0.004299
NOD,    1278,OFF, 0.010874, 0.014645,-0.009810
NOD,    1343,OFF,-0.012171, 0.014645, 0.008145
NOD,    1380,OFF, 0.012918, 0.014645,-0.006899
NOD,    1470,OFF,-0.009313, 0.014645, 0.011302
NOD,    1514,OFF, 0.014200, 0.014645,-0.003581
                        .
                        .
                        .
NOD,    1769,OFF,-0.008402, 0.000000,-0.023546
NOD,    1773,OFF,-0.018563, 0.000000,-0.016746
NOD,    1777,OFF,-0.013904, 0.000000,-0.020777
```

Table C.2: Typical OFFSET program output file

Table C.3: OFFSET program FORTRAN code

```
C
      PROGRAM OFFSET
C     --------------
C
      INTEGER NO(15000)
      REAL R(15000), TETA(15000), ZP(15000)
      CHARACTER*50 INPUT,OUTPUT,PATOUT
      CHARACTER*1 ALPHA
C
C     INPUTS
C     ------
C
      WRITE(6,2)
 2     FORMAT(' ENTER INPUT FILE NAME: ')
      READ(5,5) INPUT
C
      WRITE(6,4)
 4     FORMAT(' ENTER OUTPUT FILE NAME: ')
      READ(5,5) OUTPUT
C
 5    FORMAT(A)
C
      OPEN ( UNIT=2,
     &        FILE=INPUT,
     &        STATUS='OLD',
     &        ACCESS='SEQUENTIAL',
     &        FORM='FORMATTED',
     &        READONLY )
C
C
      READ(2,10) PATOUT
 10   FORMAT(///17X,A,///)
C
C
C     UNIT=3 IS THE OUTPUT
C     --------------------
```

```
C
        OPEN ( UNIT=3,
     &        FILE=OUTPUT,
     &        STATUS='NEW',
     &        ACCESS='SEQUENTIAL',
     &        CARRIAGECONTROL='LIST',
     &        FORM='FORMATTED' )
C
C
C
C   CREATES A LIST OF NODES NO(I) WITH THEIR COORD X(I),Y(I),Z(I)
C   --------------------------------------------------------------
C
C
        OPEN ( UNIT=1,
     &        FILE=PATOUT,
     &        STATUS='OLD',
     &        ACCESS='SEQUENTIAL',
     &        FORM='FORMATTED',
     &        READONLY )
C
C          READS THE NUMBER OF NODES
C          -------------------------
C
        READ(1,60) NN
 60     FORMAT(//,26X,I8,/)
C
C          READS & TRANSFORMS THE NODES COORDINATES
C          ----------------------------------------
C
      DO I=1,NN
        READ(1,65) NO(I),X,Y,Z
        R(I) = SQRT(X**2+Z**2)
        IF (X.NE.O.) THEN
           TETA(I) = ATAND(-Z/X)
           IF (X.LT.O.)   TETA(I)=TETA(I)+180.
        ELSEIF (Z.GE.O.) THEN
           TETA(I) = -90.
        ELSE
```

```
          TETA(I) = 90.
        ENDIF
        ZP(I) = Y
      ENDDO
C
 65   FORMAT(2X,I8,/,3E16.9,/)
C
C
C     READS THE INPUT FILE
C     --------------------
C
      DO N=1,999
C
        READ(2,*,END=999) RI1,RI2, TI1,TI2, ZPI1,ZPI2, DR,DZP
C
C
C     FINDS WHETHER THE NODE BELONG TO THE
C     ZONE WHOSE NODES NEED TO BE OFFSET
C     ------------------------------------
C
      DO I=1,NN
        IF (((RI1.LE.R(I))   .AND.(R(I).LE.RI2))     .AND.
     &      ((TI1.LE.TETA(I)).AND.(YTETA.LE.TI2))    .AND.
     &      ((ZPI1.LE.ZP(I)) .AND.(ZP(I).LE.ZPI2)) )      THEN
C
C
C     COMPUTE THE OFFSET IN BASIC COORDINATES
C     ---------------------------------------
C
          DX = DR*COSD(TETA(I))
          DY = DZP
          DZ = -DR*SIND(TETA(I))
C
C
C         WRITES OFFSET COMMAND FOR NODES SUCH THAT
C         RI1<R<RI2, TI1<TETA<YI2, ZPI1<ZP<ZPI2
C         ----------------------------------------
C
         WRITE(3,70)NO(I),DX,DY,DZ
```

```
C
 70        FORMAT('NOD,',I8,',OFF',3(',',F9.6))
C
          ENDIF
        ENDDO
C
      ENDDO
 999    STOP
        END
```

Appendix D

Gap Element Definition Program

The IGAP program creates GAP elements between pairs of coincident nodes that belong to 2 hyperpatches. GAP elements are used to describe the interaction between two components that may or may not be in contact according to the loading conditions they are submitted to.

The IGAP input file must have the same format as the file IGAP.DAT (Table D.1). A PATRAN neutral file (i.e., PATRAN.OUT) having the GFEG table must be available as an input to IGAP. For each pair of hyperpatches, the nodes having the same location are identified and a list of CGAP elements are written in the output file. The output file must be included in the NASTRAN data deck created by PATRAN. A file named "IGAP.ERR" containing the list of hyperpatches pairs of the input file that have no common nodes will be written.

A gap opening is not defined by the distance between the nodes it connects but rather by the GAP element property card (PGAP) which is not created by IGAP. Actually nodes connected by GAP elements should be made exactly coincident in the NASTRAN data deck (See Appendix G).

To run IGAP, type:

```
$RUN IGAP
 ENTER INPUT FILE NAME:        <--- written by IGAP
 IGAP.DAT                      <--- example, typed by the user
 ENTER OUTPUT FILE NAME:       <--- written by IGAP
 IGAP.SES                      <--- example, typed by the user
```

Program limits: 15000 nodes in the model
500 hyperpatches in the model

550 “surface” nodes per hyperpatche
(i.e., nodes placed on the faces of the hyperpatch)
3000 gap elements in the output file

```
*
*     SLEEVE & PIN GAP ELEMENTS, ( R DIRECTION )
*
PATRAN OUT FILE               [MCGO1 GEO]PATRAN OUT
DEFAULT CGAP ELEMENT CID  2
DEFAULT CGAP ELEMENT PID  190
FIRST ELEMENT NUMBER      19001
*
HP PAIRS (PIN, SLEEVE)
*
901 701
902 702
903 703
904 704
905 705
906 706
917 707
918 708
919 709



989 734
990 735
991 736
```

Table D 1 Typical IGAP program input file

- The first 3 lines and the 8th to the 10th lines are available for comments.
- For the 4th to the 7th lines, the user must write the required information right to ":" only.
- From the 11th line to the end, the user inputs the hyperpatches pairs in contact. Always put the hyperpatch (hp) pair in the same order: hp belonging to first component followed by the hp belonging to second component.

CGAP	19001	190	1462	453	2
CGAP	19002	190	1475	448	2
CGAP	19003	190	1461	554	2
CGAP	19004	190	1474	547	2
CGAP	19005	190	1458	631	2
CGAP	19006	190	1471	575	2
CGAP	19007	190	1456	652	2
CGAP	19008	190	1469	583	2
CGAP	19082	190	1437	706	2
CGAP	19083	190	1424	733	2
CGAP	19084	190	1411	736	2

Table D.2: Typical IGAP program output file

Table D.3: IGAP program FORTRAN code

```
C
      PROGRAM IGAP
C     ------------
C
      LOGICAL WARN
      INTEGER CCID, CPID, P1, P2, H(500), N(500,550),
     &        NO(15000), NKEEP1(3000), NKEEP2(3000), NS(500),
     &        NP1(550), NP2(550)
      REAL X(15000), Y(15000), Z(15000),
     &     X1(15000), X2(15000),
     &     Y1(15000), Y2(15000),
     &     Z1(15000), Z2(15000)
      CHARACTER*50 INPUT,OUTPUT,PATOUT
      CHARACTER*2 A2,A44
C
C     INPUTS
C     ------
C
      WRITE(6,2)
 2     FORMAT(' ENTER INPUT FILE NAME  ')
      READ(5,5) INPUT
C
      WRITE(6,4)
 4     FORMAT(' ENTER OUTPUT FILE NAME  ')
      READ(5,5) OUTPUT
C
 5    FORMAT(A)
C
      OPEN ( UNIT=2,
     &        FILE=INPUT,
     &        STATUS='OLD',
     &        ACCESS='SEQUENTIAL',
     &        FORM='FORMATTED',
     &        READONLY )
C
C
```

```
      READ(2,10) PATOUT, CCID, CPID, NGRID
 10   FORMAT(///26X,A,/,26X,I8,/,26X,I8,/,26X,I8,///)
C
C     INITIALIZATION
C     --------------
C
      NK=1
      NKEEP1(1)=0
      NKEEP2(1)=0
C
C     UNIT=11 FILE CONTAINS THE LIST HYPERPATCHES HAVING NO COMMON NODES
C     ------------------------------------------------------------------
C
      OPEN ( UNIT=11,
     &       FILE='IGAP.ERR',
     &       STATUS='NEW',
     &       ACCESS='SEQUENTIAL',
     &       CARRIAGECONTROL='LIST',
     &       FORM='FORMATTED' )
C
C     UNIT=3 IS THE OUTPUT FILE
C     -------------------------
C
      OPEN ( UNIT=3,
     &       FILE=OUTPUT,
     &       STATUS='NEW',
     &       ACCESS='SEQUENTIAL',
     &       CARRIAGECONTROL='LIST',
     &       FORM='FORMATTED' )
C
C
C     CREATES A LIST OF NODES NO(I) WITH THEIR COORD X(I), Y(I), Z(I)
C     ---------------------------------------------------------------
C
C
      OPEN ( UNIT=1,
     &       FILE=PATOUT,
     &       STATUS='OLD',
     &       ACCESS='SEQUENTIAL',
```

```
     &        FORM='FORMATTED',
     &        READONLY )
C
C           READS THE NUMBER OF NODES
C           -------------------------
C
        READ(1,60) NN
 60     FORMAT(//,26X,I8,/)
C
C           READS THE NODES COORDINATES
C           ---------------------------
C
      DO I=1,NN
        READ(1,65) NO(I),X(I),Y(I),Z(I)
      ENDDO
C
 65   FORMAT(2X,I8,/,3E16 9,/)
C
C
C     CREATES LIST OF HYPERPATCHES WITH THEIR NODES N(I,J)
C     -----------------------------------------------------
C
C           DETECTS BEGINNING OF HYPERPATCHES DEFINITION
C           --------------------------------------------
C
      DO I=1,40000
        READ(1,67) A2
        IF (A2.EQ '44') GOTO 70
      ENDDO
 67   FORMAT(A)
C
 70     BACKSPACE(1)
C
C          READS HYPERPATCH ID & NUMBER OF NODES IN THE HYPERPATCH
C          -------------------------------------------------------
C
      DO I=1,500
        K=0
        READ(1,75)A44,H(I),NNODE     !  ( HP ID & NUMBER OF NODES )
```

```
 75     FORMAT(A,2I8,/////)
C
        IF (A44.NE.'44') GOTO 85
C
C        READS NODES
C        -----------
C
        DO J=1,NNODE
C
          READ(1,80)XI1,XI2,XI3,NODE
 80       FORMAT(3E16.9,8X,I8)
C
          IF (((XI1.EQ.0.).OR.(XI1.EQ.1.))  .OR.
     &         ((XI2 EQ.0.).OR (XI2.EQ 1.))  .OR.
     &         ((XI3.EQ.0.).OR.(XI3 EQ.1 ))     ) THEN
             K=K+1
             N(I,K)=ABS(NODE)
          ENDIF
C
        NS(I)=K     !  NUMBER OF "SURFACE" NODES FOR HYPERPATCH "I"
C                   !  (I.E. THAT MAY BE IN CONTACT WITH OTHER NODES)
C
        ENDDO
C
      ENDDO
C
 85   NH=I-1     !   NH IS THE NUMBER OF HYPERPATCHES
C
C      FINDS COMMON NODES
C      -----------------
C
      DO M=1,500
C
C      SET WARNING FLAG TO TRUE
C      ( THE FLAG STAYS .T. IF THE HYPERPATCHES
C        BEING PROCESSED HAVE NO COMMON NODE )
C      -------------------------------------
C
      WARN = .TRUE.
```

```
C
C     READS A PAIR (P1,P2) OF HYPERPATCHES
C     ------------------------------------
C
      READ(2,*,END=999) P1,P2
      WRITE(6,95)P1,P2
 95   FORMAT(' PROCESSING HYPERPATCHES',2I8)
C
C     FINDS THE NODES (NP1(J), NP2(J)) BELONGING TO P1, P2
C     ----------------------------------------------------
C
      DO I=1,NH                  ! NH IS THE NUMBER OF HYPERPATCHES
        IF (P1.EQ.H(I)) THEN
          DO J=1,NS(I)
            NP1(J) = N(I,J)
          ENDDO
          NNP1=NS(I)
        ELSEIF (P2 EQ H(I)) THEN
          DO J=1,NS(I)
            NP2(J) = N(I,J)
          ENDDO
          NNP2=NS(I)
        ENDIF
      ENDDO
C
C
C     FINDS THE COORDINATES OF NP1(J), NP2(J)
C     ---------------------------------------
C
      DO K=1,NN                  ! NN IS THE NUMBER OF NODES
        DO J=1,NNP1
          IF (NP1(J).EQ.NO(K)) THEN
                X1(J)=X(K)
                Y1(J)=Y(K)
                Z1(J)=Z(K)
          ENDIF
        ENDDO
C
        DO J=1,NNP2
```

```
            IF (NP2(J).EQ.NO(K)) THEN
                 X2(J)=X(K)
                 Y2(J)=Y(K)
                 Z2(J)=Z(K)
            ENDIF
          ENDDO
        ENDDO
C
C
C      FINDS THE NODES OF P1, P2 HAVING THE SAME POSITION
C      ------------------------------------------------
C
        DO J=1,NNP1
          DO K=1,NNP2
            IF(( ABS(X1(J)-X2(K)).LE 2 E-5  )  AND
     &          ( ABS(Y1(J)-Y2(K)).LE.2.E-5  )  AND
     &          ( ABS(Z1(J)-Z2(K)).LE.2.E-5  )      )  THEN
C
C      SET WARNING FLAG TO FALSE
C      -------------------------
C
                WARN =  .FALSE.
C      VERIFY IF THE PAIR OF NODES HAS BEEN WRITTEN
C      and WRITE THEM IF NOT WRITTEN YET
C      ------------------------------------------
C
                DO L=1,NK
                  IF ((NP1(J).EQ.NKEEP1(L)) AND
     &                (NP2(K).EQ NKEEP2(L))      ) GOTO 110
                ENDDO
C
                WRITE(3,100) NGRID, CPID, NP1(J), NP2(K), CCID
 100            FORMAT('CGAP    ',4I8,24X,I8)
C
                NK=NK+1
                NKEEP1(NK)=NP1(J)
                NKEEP2(NK)=NP2(K)
                NGRID=NGRID+1
C
```

```
 110              CONTINUE
           ENDIF
         ENDDO
       ENDDO
C
C      WARNS IF NO COMMON NODE FOUND
C      -----------------------------
C
       IF(WARN) THEN
         WRITE(6,96)
 96      FORMAT('   WARNING !
     &   THE ABOVE HYPERPATCHES HAVE NO COMMON NODE ')
         WRITE(11,*)P1,P2
       ENDIF
C
       ENDDO
 999   STOP
       END
```

Appendix E

Bar Element Definition Program

The IBAR program creates BAR elements between pairs of coincident nodes that belong to 2 hyperpatches. BAR elements are used in linear analyses to simulate the interaction of 2 components known to stay in contact when loaded.

Bars created by IBAR program can transmit axial loads only. They can be pre-strained to simulate interference fit effect. MPC cards are recommended if there is no pre-stress (see Appendix F). Bars rigidity should be set to a value much higher than the component material rigidities. The BAR elements property card (PBAR) is not created by the IBAR program.

The input file must have the same format as the file IBAR.DAT (Table E.1). A PATRAN neutral file (i.e. PATRAN.OUT) having the GFEG table must be available as an input to IBAR.

For each pair of hyperpatches, the nodes having the same location will be identified and a list of CBAR elements will be written in the output file. The output file must be included in the NASTRAN data deck created by PATRAN. A file named "IBAR.ERR" containing the list of hyperpatches pairs of the input file having no common nodes will be written. A list of DEFORM cards will also be written in a file named "DEFORM.OUT" (Table E.2).

The bars length is defined in the IBAR input file, it is not defined by the distance between the nodes bars connect. Actually, nodes connected by BARs must be made exactly coincident in the NASTRAN data deck (see Appendix G).

To run IBAR, type:

```
$RUN IBAR
 ENTER INPUT FILE NAME:        <--- written by IBAR
 IBAR.DAT                      <--- example, typed by the user
```

```
ENTER OUTPUT FILE NAME:        <--- written by IBAR
IBAR.SES                       <--- example, typed by the user
```

Program limits: 15000 nodes in the model
500 hyperpatches in the model
550 “surface” nodes per hyperpatch
(i.e., nodes placed on the faces of an hyperpatch)
3000 bar elements in the output file

```
*
*      LUG & BUSHING CBAR ELEMENT DEFINITION (R DIRECTION, CID 2)
*
PATRAN.OUT FILE                                  :[MCGO1.F]PATRAN.OUT
DEFAULT CBAR ELEMENT PID                         :120
FIRST ELEMENT NUMBER                             :12001
BAR ORIENTATION RELATIVE TO END 1ST CID          :1
BAR LENGTH                                       :.002
INITIAL DEFORMATION (>0 IF COMPRESS)             :.001
DEFORM SET ID                                    :104
*
HYPERPATCHES PAIRS : (BUSHING LUG)
*
604 103
612 107
611 106
616 114
624 118
623 117
```

Table E.1: Typical IBAR program input file

- The first 3 lines and the 11th to the 13th lines are available for comments.
- From the 4th to the 10th lines, the user must write at the right of ":" only.
- From the 14th line to the end, the user puts the hyperpatches pairs in contact. Always put the hyperpatch (hp) pair in the same order: hp belonging to first component followed by the hp belonging to the second component.

```
CBAR       12001    120     827     196    0.     1        0.       E 12001
+E 12001      23     23-0.00100                  0 00100
CBAR       12002    120     832     197    0.     1        0.       E 12002
+E 12002      23     23-0.00100                  0 00100
CBAR       12003    120     833     198    0.     1.       0        E 12003
+E 12003      23     23-0.00100                  0 00100
CBAR       12004    120     841     202    0      1.       0.       E 12004
+E 12004      23     23-0.00100                  0 00100
CBAR       12005    120     848     204    0      1.       0        E 12005
+E 12005      23     23-0 00100                  0 00100
CBAR       12006    120     896     126    0      1        0        E 12006
+E 12006      23     23-0.00100                  0.00100
CBAR       12007    120     906     128    0.     1.       0        E 12007
+E 12007      23     23-0 00100                  0 00100
CBAR       12008    120     907     129    0      1        0.       E 12008
+E 12008      23     23-0.00100                  0 00100
CBAR       12009    120     924     132    0      1        0.       E 12009
+E 12009      23     23-0 00100                  0 00100

CBAR       12118    120    1744     744    0.     1        0.       E 12118
+E 12118      23     23-0.00100                  0 00100
CBAR       12119    120    1748     745    0      1.       0        E 12119
+E 12119      23     23-0.00100                  0.00100
CBAR       12120    120    1752     746    0.     1.       0        E 12120
+E 12120      23     23-0.00100                  0 00100
```

Table E.2: Typical IBAR program output file

It corresponds to the Table E.1

DEFORM	104	12001	0 001
DEFORM	104	12002	0 001
DEFORM	104	12003	0 001
DEFORM	104	12004	0 001
DEFORM	104	12119	0 001
DEFORM	104	12120	0 001

Table E 3 Typical DEFORM OUT file created by IBAR program

Table E 4 IBAR program FORTRAN code

```
C
      PROGRAM IBAR
C     ------------
C
C
      LOGICAL WARN
      INTEGER CPID, P1, P2, H(500), N(500,550),
     &        NO(15000), NKEEP1(3000), NKEEP2(3000),
     &        NP1(550), NP2(550), NS(500),
     &        PIN, SID
      REAL X(15000), Y(15000), Z(15000),
     &     X1(15000), X2(15000),
     &     Y1(15000), Y2(15000),
     &     Z1(15000), Z2(15000)
      CHARACTER*50 INPUT,OUTPUT,PATOUT
      CHARACTER*2 A2,A44
C
C     INPUTS
C     ------
C
      WRITE(6,2)
 2     FORMAT(' ENTER INPUT FILE NAME  ')
      READ(5,5) INPUT
C
      WRITE(6,4)
 4     FORMAT(' ENTER OUTPUT FILE NAME  ')
      READ(5,5) OUTPUT
C
 5    FORMAT(A)
C
      OPEN ( UNIT=2,
     &       FILE=INPUT,
     &       STATUS='OLD',
     &       ACCESS='SEQUENTIAL',
     &       FORM='FORMATTED',
     &       READONLY )
```

```
C
C
      READ(2,10) PATOUT, CPID, NGRID, PIN, BARL, DEFORM,,SID
 10   FORMAT(///41X,A,3(/,41X,I8),2(/,41X,F10 7),/,41X,I8,///)
C
C
C     INITIALIZATION
C     --------------
C
      NK=1
      NKEEP1(1)=0
      NKEEP2(1)=0
C
C     END A & B SHIFT
C     ---------------
C
      BARL2 = BARL/2
      BARL1 = -BARL2
C
C     UNIT=12 FILE CONTAINS THE DEFORM CARDS
C     --------------------------------------
C
      OPEN ( UNIT=12,
     &       FILE='DEFORM OUT',
     &       STATUS='NEW',
     &       ACCESS='SEQUENTIAL',
     &       CARRIAGECONTROL='LIST',
     &       FORM='FORMATTED' )
C
C     UNIT=11 FILE CONTAINS THE LIST OF HYPERPATCHES WITHOUT COMMON NODES
C     -------------------------------------------------------------------
C
      OPEN ( UNIT=11,
     &       FILE='IBAR ERR',
     &       STATUS='NEW',
     &       ACCESS='SEQUENTIAL',
     &       CARRIAGECONTROL='LIST',
     &       FORM='FORMATTED' )
C
```

```
C     UNIT=3 IS THE OUTPUT
C     --------------------
C
      OPEN ( UNIT=3,
     &       FILE=OUTPUT,
     &       STATUS='NEW',
     &       ACCESS='SEQUENTIAL',
     &       CARRIAGECONTROL='LIST',
     &       FORM='FORMATTED' )
C
C
C     CREATES A LIST OF NODES NO(I) WITH THEIR COORD X(I), Y(I), Z(I)
C     ---------------------------------------------------------------
C
C
      OPEN ( UNIT=1,
     &       FILE=PATOUT,
     &       STATUS='OLD',
     &       ACCESS='SEQUENTIAL',
     &       FORM='FORMATTED',
     &       READONLY )
C
C         READS THE NUMBER OF NODES
C         -------------------------
C
      READ(1,60) NN
60    FORMAT(//,26X,I8,/)
C
C         READS THE NODES COORDINATES
C         ---------------------------
C
      DO I=1,NN
        READ(1,65) NO(I),X(I),Y(I),Z(I)
      ENDDO
C
65    FORMAT(2X,I8,/,3E16 9,/)
C
C
C     CREATES LIST OF HYPERPATCHES WITH THEIR NODES N(I,J)
```

```
C     -------------------------------------------
C
C          DETECTS BEGINNING OF HYPERPATCHES DEFINITION
C          ------------------------------------------
C
      DO I=1,40000
        READ(1,67) A2
        IF (A2.EQ '44') GOTO 70
      ENDDO
 67   FORMAT(A)
C
 70     BACKSPACE(1)
C
C          READS HYPERPATCH ID & NUMBER OF NODES IN THE HYPERPATCH
C          -------------------------------------------------------
C
      DO I=1,500
        K=0
        READ(1,75)A44,H(I),NNODE     !  ( HP ID & NUMBER OF NODES )
 75     FORMAT(A,2I8,/////)
C
        IF (A44 NE '44') GOTO 85
C
C          READS NODES
C          -----------
C
        DO J=1,NNODE
C
          READ(1,80)XI1,XI2,XI3,NODE
 80       FORMAT(3E16.9,8X,I8)
C
          IF (((XI1.EQ.O.).OR.(XI1 EQ 1 ))  OR.
     &        ((XI2.EQ.O.).OR.(XI2 EQ.1.))  OR
     &        ((XI3.EQ.O.).OR.(XI3.EQ.1.))      ) THEN
            K=K+1
            N(I,K)=ABS(NODE)
          ENDIF
C
        NS(I)=K    !  NUMBER OF "SURFACE" NODES FOR HYPERPATCH "I"
```

```
C                 !  (I E  THAT MAY BE IN CONTACT WITH OTHER NODES)
C
          ENDDO
C
        ENDDO
C
 85     NH=I-1     !   NH IS THE NUMBER OF HYPERPATCHES
C
C
C        FINDS COMMON NODE
C        -----------------
C
         DO M=1,500
C
C        SETS WARNING FLAG TO TRUE
C        ( THE FLAG STAYS .T  IF THE HYPERPATCHES
C          BEING PROCESSED HAVE NO COMMON NODE )
C        ---------------------------------------
C
         WARN =  TRUE
C
C        READS A PAIR (P1,P2) OF HYPERPATCHES
C        ------------------------------------
C
         READ(2,*,END=999) P1,P2
         WRITE(6,95)P1,P2
 95      FORMAT(' PROCESSING HYPERPATCHES',2I8)
C
C        FINDS THE NODES (NP1(J), NP2(J)) BELONGING TO P1, P2
C        ----------------------------------------------------
C
         DO I=1,NH                    ! NH IS THE NUMBER OF HYPERPATCHES
           IF (P1.EQ.H(I)) THEN
             DO J=1,NS(I)
               NP1(J) = N(I,J)
             ENDDO
             NNP1=NS(I)
           ELSEIF (P2.EQ.H(I)) THEN
             DO J=1,NS(I)
```

```
            NP2(J) = N(I,J)
          ENDDO
          NNP2=NS(I)
        ENDIF
      ENDDO
C
C
C     FINDS THE COORDINATES OF NP1(J), NP2(J)
C     -------------------------------------
C
      DO K=1,NN                    ! NN IS THE NUMBER OF NODES
        DO J=1,NNP1
          IF (NP1(J) EQ NO(K)) THEN
                X1(J)=X(K)
                Y1(J)=Y(K)
                Z1(J)=Z(K)
          ENDIF
        ENDDO
C
        DO J=1,NNP2
          IF (NP2(J).EQ.NO(K)) THEN
                X2(J)=X(K)
                Y2(J)=Y(K)
                Z2(J)=Z(K)
          ENDIF
        ENDDO
      ENDDO
C
C
C     FINDS THE NODES OF P1, P2 HAVING THE SAME POSITION
C     ------------------------------------------------
C
      DO J=1,NNP1
        DO K=1,NNP2
          IF(( ABS(X1(J)-X2(K)).LE.2.E-5  ) AND
     &       ( ABS(Y1(J)-Y2(K)).LE.2.E-5  ) .AND
     &       ( ABS(Z1(J)-Z2(K)).LE.2.E-5  )      ) THEN
C
C     SETS WARNING FLAG TO FALSE
```

```
C     ------------------------
C
            WARN = .FALSE
C
C     VERIFIES IF THE PAIR OF NODES HAS BEEN WRITTEN
C     and WRITES THE APPROPRATE CBAR ELEMENT
C     IF NOT WRITTEN YET
C     -------------------------------------------
C                        &
C     WRITES THE APPROPRIATE DEFORM CARD
C     --------------------------------
C
            DO L=1,NK
              IF ((NP1(J) EQ NKEEP1(L)) AND
     &           (NP2(K) EQ NKEEP2(L))      ) GOTO 110
            ENDDO
C
            WRITE(12,98) SID,NGRID,DEFORM
 98         FORMAT('DEFORM  ',2I8,F8 5)
C
C
            WRITE(3,100) NGRID, CPID, NP1(J), NP2(K), NGRID
 100        FORMAT('CBAR    ',4I8,'       0       1       0
     &      E',I6)
            IF (PIN EQ 1) THEN
              WRITE(3,105) NGRID, BARL1, BARL2
 105          FORMAT('+E',I6,'      23      23',F8 5,16X,F8 5)
            ELSEIF (PIN.EQ.2) THEN
              WRITE(3,106) NGRID, BARL1, BARL2
 106          FORMAT('+E',I6,'      23      23',8X,F8 5,16X,F8 5)
            ELSE
              WRITE(3,107) NGRID, BARL1, BARL2
 107          FORMAT('+E',I6,'      23      23',16X,F8 5,16X,F8 5)
            ENDIF
C
            NK=NK+1
            NKEEP1(NK)=NP1(J)
            NKEEP2(NK)=NP2(K)
            NGRID=NGRID+1
```

```
C
 110                CONTINUE
              ENDIF
            ENDDO
          ENDDO
C
C         WARNS IF NO COMMON NODE
C         ----------------------
C
          IF(WARN) THEN
            WRITE(6,96)
 96         FORMAT('    WARNING !
     &      THE ABOVE HYPERPATCHES HAVE NO COMMON NODES ')
            WRITE(11,*)P1,P2
C
          ENDIF
          ENDDO
 999      STOP
          END
```

Appendix F

Multi-Point Constraint (MPC) Definition Program

The IMPC program writes MPC cards corresponding to coincident pairs of nodes that belong to 2 hyperpatches. MPC cards written by the IMPC program are used when two components are known (or assumed) to stay in contact when loaded

The input file must have the same format as the file IMPC DAT (Table F 1) A PATRAN neutral file (i.e. PATRAN OUT) having the GFEG table must be available as an input to IMPC.

For each pair of hyperpatches, the nodes having the same location will be identified and a list of MPC cards will be written in the output file. The output file must be included in the NASTRAN data deck created by PATRAN. A file named "IMPC.ERR" containing the list of hyperpatches pairs of the input file that have no common nodes will be written.

To run IMPC, type:

```
$RUN IMPC
 ENTER INPUT FILE NAME:        <--- written by IMPC
 IMPC.DAT                      <--- example, typed by the user
 ENTER OUTPUT FILE NAME:       <--- written by IMPC
 IMPC.SES                      <--- example, typed by the user
```

Program limits:15000 nodes in the model
500 hyperpatches in the model
550 "surface" nodes per hyperpatch
(i.e., nodes placed on the faces of an hyperpatch)

3000 bar elements in the output file

```
*
*     WASHER & BLADE ( Y DIRECTION )
*
PATRAN.OUT FILE               :[MCGO1.GEO]PATRAN.OUT
MPC SID NUMBER                :501
DEPENDANT DEGREE OF FREEDOM :3
*
HP PAIRS (BLADE, WASHER)
*
401 805
402 806
404 801
405 802
406 803
407 804
414 811
415 812
417 807
    .
   ..
    .
444 820
445 821
446 822
```

Table F.1: Typical IMPC input file

- The first 3 lines and the 7th to the 9th lines are available for comments.
- For the 4th to the 6th lines, the user must write the required information right to ":" only.
- From the 10th line to the end, the user puts the hyperpatches pairs in contact. Always put the hyperpatch (hp) pair in the same order: hp belonging to the first component followed by the hp belonging to the second component.

MPC	501	517	3	1.	1147	3	-1.
MPC	501	520	3	1.	1145	3	-1.
MPC	501	516	3	1.	1159	3	-1.
MPC	501	521	3	1.	1157	3	-1.
MPC	501	515	3	1.	1171	3	-1
MPC	501	518	3	1.	1169	3	-1.
MPC	501	429	3	1.	1143	3	-1.
	⋮						
MPC	501	535	3	1.	1251	3	-1.
MPC	501	619	3	1.	1239	3	-1.
MPC	501	651	3	1.	1227	3	-1.

Table F.2: Typical IMPC output file

Table F.3: IMPC program FORTRAN code

```
C
      PROGRAM IMPC
C     ------------
C
C
      LOGICAL WARN
      INTEGER DFREE, CPID, P1, P2, H(500), N(500,550),
     &        NO(15000), NKEEP1(3000), NKEEP2(3000),
     &        NP1(550), NP2(550), NS(500)
      REAL X(15000), Y(15000), Z(15000),
     &     X1(15000), X2(15000),
     &     Y1(15000), Y2(15000),
     &     Z1(15000), Z2(15000)
      CHARACTER*50 INPUT,OUTPUT,PATOUT
      CHARACTER*2 A2,A44
C
C     INPUTS
C     ------
C
      WRITE(6,2)
 2     FORMAT(' ENTER INPUT FILE NAME. ')
      READ(5,5) INPUT
C
      WRITE(6,4)
 4     FORMAT(' ENTER OUTPUT FILE NAME: ')
      READ(5,5) OUTPUT
C
 5    FORMAT(A)
C
      OPEN ( UNIT=2,
     &        FILE=INPUT,
     &        STATUS='OLD',
     &        ACCESS='SEQUENTIAL',
     &        FORM='FORMATTED',
     &        READONLY )
C
```

```
C
      READ(2,10) PATOUT, MPCID, DFREE
 10   FORMAT(///29X,A,/,29X,I8,/,29X,I8,///)
C
C     INITIALIZATION
C     --------------
C
      NK=1
      NKEEP1(1)=0
      NKEEP2(1)=0
C
C     UNIT=11 FILE CONTAINS THE PAIRS OF HYPERPATCHES WHITOUT COMMON NODES
C     --------------------------------------------------------------------
C
      OPEN ( UNIT=11,
     &        FILE='IMPC ERR',
     &        STATUS='NEW',
     &        ACCESS='SEQUENTIAL',
     &        CARRIAGECONTROL='LIST',
     &        FORM='FORMATTED' )
C
C     UNIT=3 IS THE OUTPUT FILE
C     -------------------------
C
      OPEN ( UNIT=3,
     &        FILE=OUTPUT,
     &        STATUS='NEW',
     &        ACCESS='SEQUENTIAL',
     &        CARRIAGECONTROL='LIST',
     &        FORM='FORMATTED' )
C
C
C     CREATES A LIST OF NODES NO(I) WITH THEIR COORD X(I), Y(I), Z(I)
C     ----------------------------------------------------------------
C
C
      OPEN ( UNIT=1,
     &        FILE=PATOUT,
     &        STATUS='OLD',
```

```
     &       ACCESS='SEQUENTIAL',
     &       FORM='FORMATTED',
     &       READONLY )
C
C          READS THE NUMBER OF NODES
C          ------------------------
C
          READ(1,60) NN
 60       FORMAT(//,26X,I8,/)
C
C          READS THE NODES COORDINATES
C          --------------------------
C
        DO I=1,NN
          READ(1,65) NO(I),X(I),Y(I),Z(I)
        ENDDO
C
 65     FORMAT(2X,I8,/,3E16 9,/)
C
C
C       CREATES LIST OF HYPERPATCHES WITH THEIR NODES N(I,J)
C       ---------------------------------------------------
C
C          DETECTS BEGINNING OF HYPERPATCHES DEFINITION
C          --------------------------------------------
C
        DO I=1,40000
          READ(1,67) A2
          IF (A2 EQ '44') GOTO 70
        ENDDO
 67     FORMAT(A)
C
 70       BACKSPACE(1)
C
C         READS HYPERPATCH ID & NUMBER OF NODES IN THE HYPERPATCH
C         -------------------------------------------------------
C
        DO I=1,500
          K=0
```

```
      READ(1,75)A44,H(I),NNODE     !  ( HP ID & NUMBER OF NODES )
 75   FORMAT(A,2I8,/////)
C
      IF (A44 NE '44') GOTO 85
C
C      READS NODES
C      -----------
C
      DO J=1,NNODE
C
        READ(1,80)XI1,XI2,XI3,NODE
 80     FORMAT(3E16 9,8X,I8)
C
        IF (((XI1.EQ 0 ) OR (XI1 EQ 1 ))  OR
     &      ((XI2 EQ 0 ) OR (XI2 EQ 1 ))  OR
     &      ((XI3 EQ 0 ) OR (XI3 EQ 1 ))     ) THEN
          K=K+1
          N(I,K)=ABS(NODE)
        ENDIF
C
      NS(I)=K     !  NUMBER OF "SURFACE" NODES FOR HYPERPATCH "I"
C                 !  (I E  THAT MAY BE COINCIDENT WITH OTHER NODES)
C
      ENDDO
C
      ENDDO
C
 85   NH=I-1    !   NH IS THE NUMBER OF HYPERPATCHES
C
C     FINDS COMMON NODES
C     ------------------
C
      DO M=1,500
C
C     SETS WARNING FLAG TO TRUE
C     ( THE FLAG STAYS .T. IF THE HYPERPATCHES
C       BEING PROCESSED HAVE NO COMMON NODE )
C     ---------------------------------------
C
```

```
      WARN =  TRUE.
C
C     READS A PAIR (P1,P2) OF HYPERPATCHES
C     --------------------------------
C
      READ(2,*,END=999) P1,P2
      WRITE(6,95)P1,P2
 95   FORMAT(' PROCESSING HYPERPATCHES',2I8)
C
C     FINDS THE NODES (NP1(J), NP2(J)) BELONGING TO P1, P2
C     ---------------------------------------------------
C
      DO I=1,NH                     ! NH IS THE NUMBER OF HYPERPATCHES
        IF (P1 EQ H(I)) THEN
          DO J=1,NS(I)
            NP1(J) = N(I,J)
          ENDDO
          NNP1=NS(I)
        ELSEIF (P2 EQ H(I)) THEN
          DO J=1,NS(I)
            NP2(J) = N(I,J)
          ENDDO
          NNP2=NS(I)
        ENDIF
      ENDDO
C
C
C     FINDS THE COORDINATES OF NP1(J), NP2(J)
C     --------------------------------------
C
      DO K=1,NN                     ! NN IS THE NUMBER OF NODES
        DO J=1,NNP1
          IF (NP1(J) EQ.NO(K)) THEN
                X1(J)=X(K)
                Y1(J)=Y(K)
                Z1(J)=Z(K)
          ENDIF
        ENDDO
C
```

```
        DO J=1,NNP2
          IF (NP2(J).EQ.NO(K)) THEN
                 X2(J)=X(K)
                 Y2(J)=Y(K)
                 Z2(J)=Z(K)
          ENDIF
        ENDDO
      ENDDO
C
C
C     FINDS THE NODES OF P1, P2 HAVING THE SAME POSITION
C     ---------------------------------------------------
C
      DO J=1,NNP1
        DO K=1,NNP2
          IF(( ABS(X1(J)-X2(K)) LE.2 E-5  )  AND
     &        ( ABS(Y1(J)-Y2(K)) LE 2 E-5  )  AND
     &        ( ABS(Z1(J)-Z2(K)) LE 2 E-5  )       )  THEN
C
C     SETS WARNING FLAG TO FALSE
C     --------------------------
C
            WARN =  FALSE
C
C     VERIFIES IF THE PAIR OF NODES HAS BEEN WRITTEN
C     and WRITES THE APPROPRIATE MPC CARD IF NOT WRITTEN YET
C     -------------------------------------------------------
C
            DO L=1,NK
              IF ((NP1(J).EQ.NKEEP1(L)).AND.
     &              (NP2(K).EQ.NKEEP2(L))      ) GOTO 110
            ENDDO
C
            WRITE(3,100) MPCID, NP1(J), DFREE, NP2(K), DFREE
 100        FORMAT('MPC     ',I8,2I8,'      1 ',2I8,'     -1.')
C
            NK=NK+1
            NKEEP1(NK)=NP1(J)
            NKEEP2(NK)=NP2(K)
```

```
 110               CONTINUE
            ENDIF
          ENDDO
        ENDDO
C
C       WARNS IF NO COMMON NODE
C       -----------------------
C
        IF(WARN) THEN
          WRITE(6,96)
 96       FORMAT('    WARNING !
     &    THE ABOVE HYPERPATCHES HAVE NO COMMON NODES ')
          WRITE(11,*)P1,P2
        ENDIF
C
        ENDDO
 999    STOP
        END
```

Appendix G

Node Coordinates Rounding off Program

The TRONC program rewrites a NASTRAN data file with the node positions (given on the NASTRAN GRID cards) rounded off. The last digit of the R, θ, Z' coordinates are removed. It is used to make sure that nodes that should be coincident do coincide exactly in the NASTRAN data deck.

For example, suppose that a NASTRAN data deck have the following GRID cards.

```
                    .
                    .
                    .
GRID           21 0.49999 112 1103 2.01901
                    .
                    .
                    .
GRID          298 0.50003 112.1100 2.01900
                    .
                    .
                    .
```

Upon running, NASTRAN will give an error message if a GAP element connects these 2 nodes since these are not exactly coincident.

The program TRONC rewrites the NASTRAN data deck with rounded off GRID coordinates.

The TRONC output file corresponding to the input file shown above will be:

```
                    .
                    .
                    .
GRID          21 0.5000  112 110  2.0190
                    .
                    .
                    .
GRID         298 0.5000  112 110  2.0190
                    .
                    .
```

...where nodes 21 and 298 are exactly coincident

The TRONC program will modify the "GRID" cards only All the other lines will be rewritten in the output file unchanged.

To run TRONC, type:

```
$RUN TRONC
 ENTER INPUT FILE NAME:        <--- written by TRONC
 NASTRAN.BDF                   <--- example, typed by the user
 ENTER OUTPUT FILE NAME:       <--- written by TRONC
 NASTRA2.BDF                   <--- example, typed by the user
```

Table G.1: TRONC program FORTRAN code

```
C
      PROGRAM  TRONC
C     --------------
C
C
      REAL*8 X,Y,Z
      CHARACTER*50 NASTRAN,OUTPUT
      CHARACTER*4  ALPHA
      CHARACTER*80 ALINE
      CHARACTER*24 A24
      INTEGER OCID, CCID, G,CID,CD
C
C
C     INPUTS
C     ------
C
      WRITE(6,2)
 2     FORMAT(' ENTER INPUT FILE NAME   ')
      READ(5,5) NASTRAN
C
      WRITE(6,4)
 4     FORMAT(' ENTER OUTPUT FILE NAME   ')
      READ(5,5) OUTPUT
C
 5    FORMAT(A)
C
C
C     OPENS NASTRAN.BDF INPUT FILE AND NASTRAN.BDF OUTPUT FILE
C     --------------------------------------------------------
C
      OPEN ( UNIT=2,
     &        FILE=NASTRAN,
     &        STATUS='OLD',
     &        ACCESS='SEQUENTIAL',
     &        FORM='FORMATTED',
     &        READONLY )
```

```
C
      OPEN ( UNIT=3,
     &       FILE=OUTPUT,
     &       STATUS='NEW',
     &       ACCESS='SEQUENTIAL',
     &       CARRIAGECONTROL='LIST',
     &       FORM='FORMATTED' )
C
C
C     DETECTS WHERE GRID DEFINITION BEGINS & ENDS
C     -----------------------------------------
C
      DO I=1,13000
        READ(2,30) ALPHA
 30     FORMAT(A)
        IF (ALPHA EQ 'GRID') GOTO 40
      ENDDO
C
C
 40   IBEGIN = I-1
      DO I=1,13000
        READ(2,30) ALPHA
        IF (ALPHA NE.'GRID') GOTO 50
      ENDDO
C
 50   NG=I
      CLOSE(2)
C
C     WRITES THE OUTPUT NASTRAN FILE UP TO THE 1ST GRID
C     ------------------------------------------------
C
C
      OPEN ( UNIT=2,
     &       FILE=NASTRAN,
     &       STATUS='OLD',
     &       ACCESS='SEQUENTIAL',
     &       FORM='FORMATTED',
     &       READONLY )
C
```

```
      DO I=1,IBEGIN
        READ(2,60)ALINE
        WRITE(3,60)ALINE
 60     FORMAT(A)
      ENDDO
C
C
C          READS GRIDS AND THEIR COORDINATES
C        & WRITES THEM AFTER ROUNDING OFF
C          -------------------------------
C
      DO I=1,NG
C
        READ(2,63) G,CID,X,Y,Z,CD,A24
 63     FORMAT(8X,2I8,F8 5,F8 4,F8 5,I8,A24)
C
        WRITE(3,70)G,CID,X,Y,Z,CD,A24
 70     FORMAT('GRID    ',2I8,F8 4,F8 3,F8 4,I8,A24)
C
      ENDDO
C
C
C     WRITES THE REST OF THE OUTPUT NASTRAN FILE
C     ----------------------------------------
C
      DO I=1,10000
        READ(2,60,END=999)ALINE
        WRITE(3,60)ALINE
      ENDDO
C
 999  STOP
      END
```

Appendix H

Node Location Program

Two node location programs have been written. They permit to identify the nodes that belong to a volume entered as input and find those that are coincident. The volumes can be entered either in the basic cartesian coordinate system (IVOL program) or in cylindrical coordinate system (IVOLC program).

H.1 IVOL Program: Node Location Program (basic cartesian coordinates)

The IVOL program finds the nodes that belong to volumes entered as input. Coincident nodes are put in pairs.

The input file must have the same format as the file IVOL.DAT (Table H.1). A PATRAN neutral file (i.e. PATRAN.OUT) must be available as an input to IVOL.

To run IVOL, type:

```
$RUN IVOL
ENTER INPUT FILE NAME:          <--- written by IVOL
IVOL.DAT                        <--- example, typed by the user
ENTER OUTPUT FILE NAME:         <--- written by IVOL
IVOL.OUT                        <--- example, typed by the user
```

Program limit: 15000 nodes in the model

```
*
*     TITLE
*
PATRAN.OUT FILE :[MCG01.GEO]PATRAN.OUT
*
*  ZONE ID,     X1 TO X2,    Y1 TO Y2,  Z1 TO Z3
*
1   -1.126 -1.124      -10.  10.      -.001 .001
2   -.901  -.899       1.8   10.      -.001 .001
3   -.901  -.899       -10.  -1.8     -.001 .001
```

Table H.1: Typical IVOL program input file

- The first 3 lines are available for comments.
- The 4th line gives the PATRAN neutral file.
- The 5th line is available for comments.
- The 6th line indicates how to input the volumes whose nodes need to be identified. Each volume is identified by a user chosen ID and its upper and lower limits given in the basic cartesian coordinate system. Free format is used.
- The 7th line is available for comments.
- Each of the subsequent line gives the parameters defining a volume whose nodes need to be identified.

```
FOR ZONE         1,
                      X FROM   -1 12600 TO   -1 12400
                      Y FROM  -10 00000 TO   10 00000
                      Z FROM   -0 00100 TO    0 00100
   NODE ID'S          X          Y          Z
    15     817 -1 12500   1 97100   0 00000
    16    NONE -1 12500   2 59600   0 00000
   167    1009 -1 12500  -1 97100   0 00000
   168    NONE -1 12500  -2 59600   0 00000
   429    1143 -1 12500   1 66400   0 00000
   431    NONE -1 12500   0 95400   0 00000
   679    1233 -1 12500  -1 66400   0 00000
   681    NONE -1 12500  -0 95400   0 00000
   817    NONE -1 12500   1 97100   0 00000
   818    1144 -1 12500   1 87400   0 00000
  1009    NONE -1 12500  -1 97100   0 00000
  1010    1234 -1 12500  -1 87400   0 00000
  1143    NONE -1 12500   1 66400   0 00000
  1144    NONE -1 12500   1 87400   0 00000
  1233    NONE -1 12500  -1 66400   0 00000
  1234    NONE -1 12500  -1 87400   0 00000
FOR ZONE         2,
                      X FROM   -0 90100 TO   -0 89900
                      Y FROM    1 80000 TO   10 00000
                      Z FROM   -0 00100 TO    0 00100
   NODE ID'S          X          Y          Z
    17    1507 -0 90000   2 59600   0 00000
    20     860 -0 90000   1 97100   0 00000
   859    1146 -0 90000   1 87400   0 00000
   860    NONE -0 90000   1 97100   0 00000
  1146    NONE -0 90000   1 87400   0 00000
  1506    NONE -0.90000   3 09600   0 00000
  1507    NONE -0 90000   2 59600   0 00000
FOR ZONE         3,
                      X FROM   -0 90100 TO   -0 89900
                      Y FROM  -10 00000 TO   -1 80000
                      Z FROM   -0 00100 TO    0 00100
   NODE ID'S          X          Y          Z
   169    1290 -0 90000  -2 59600   0 00000
   172    1051 -0 90000  -1 97100   0.00000
  1051    NONE -0 90000  -1 97100   0 00000
  1052    1236 -0 90000  -1 87400   0 00000
  1236    NONE -0 90000  -1 87400   0 00000
  1289    NONE -0 90000  -3 09600   0 00000
  1290    NONE -0 90000  -2 59600   0 00000
```

Table H.2: Typical IVOL program output file

The corresponding input file is given in Table H.1.

Table H.3: IVOL program FORTRAN code

```
C
      PROGRAM IVOL
C     ------------
C
C
      INTEGER NO(15000), NOK(1000)
      REAL X(15000), Y(15000), Z(15000)
      REAL XK(1000), YK(1000), ZK(1000)
      CHARACTER*50 INPUT,OUTPUT,PATOUT
      CHARACTER*1 ALPHA
      LOGICAL FLAG
C
C     INPUTS
C     ------
C
      WRITE(6,2)
 2     FORMAT(' ENTER INPUT FILE NAME   ')
      READ(5,5) INPUT
C
      WRITE(6,4)
 4     FORMAT(' ENTER OUTPUT FILE NAME   ')
      READ(5,5) OUTPUT
C
 5    FORMAT(A)
C
      OPEN ( UNIT=2,
     &        FILE=INPUT,
     &        STATUS='OLD',
     &        ACCESS='SEQUENTIAL',
     &        FORM='FORMATTED',
     &        READONLY )
C
C
      READ(2,10) PATOUT
 10   FORMAT(///17X,A,///)
C
```

```
C
C     UNIT=3 IS THE OUTPUT
C     -------------------
C
      OPEN ( UNIT=3,
     &       FILE=OUTPUT,
     &       STATUS='NEW',
     &       ACCESS='SEQUENTIAL',
     &       CARRIAGECONTROL='LIST',
     &       FORM='FORMATTED' )
C
C
C
C     CREATES A LIST OF NODES NO(I) WITH THEIR COORD X(I), Y(I), Z(I)
C     - ------------------------------------------------------------
C
C
      OPEN ( UNIT=1,
     &       FILE=PATOUT,
     &       STATUS='OLD',
     &       ACCESS='SEQUENTIAL',
     &       FORM='FORMATTED',
     &       READONLY )
C
C         READS THE NUMBER OF NODES
C         -------------------------
C
        READ(1,60) NN
 60     FORMAT(//,26X,I8,/)
C
C         READS THE NODES COORDINATES
C         ---------------------------
C
      DO I=1,NN
        READ(1,65) NO(I),X(I),Y(I),Z(I)
      ENDDO
C
 65   FORMAT(2X,I8,/,3E16 9,/)
C
```

```
C
C     READS THE INPUT FILE
C     --------------------
C
      DO N=1,999
C
        READ(2,*,END=999) IDZONE, XI1, XI2, YI1, YI2, ZI1, ZI2
        WRITE(3,69) IDZONE, XI1, XI2, YI1, YI2, ZI1, ZI2
 69     FORMAT(//,'  FOR ZONE  ',I8,',',
     &         /,20X,'  X FROM ',F9.5,' TO ',F9.5,
     &         /,20X,'  Y FROM ',F9.5,' TO ',F9.5,
     &         /,20X,'  Z FROM ',F9.5,' TO ',F9.5,/,
     &         /,6X,'NODE ID''S          X          Y          Z' )
C
C          KEEPS THE NODES SUCH THAT XI1<X<XI2, YI1<Y<YI2, ZI1<Z<ZI2
C          ------------------------------------------------------
C
      NK=0
C
      DO I=1,NN
        IF (((XI1.LE X(I)).AND (X(I).LE XI2))  AND.
     &      ((YI1.LE Y(I)) AND (Y(I) LE YI2))  AND
     &      ((ZI1 LE Z(I)) AND.(Z(I) LE ZI2)) )          THEN
          NK=NK+1
          NOK(NK)=NO(I)
          XK(NK)=X(I)
          YK(NK)=Y(I)
          ZK(NK)=Z(I)
        ENDIF
      ENDDO
C
C     PUTS THE NODES KEPT IN PAIRS
C     ----------------------------
C
      DO I=1,NK
        IP1=I+1
        FLAG=.FALSE.
        DO J=IP1,NK
          IF(( ABS(XK(J)-XK(I)).LE.2.E-5  ) .AND.
```

```
     &         ( ABS(YK(J)-YK(I)) LE 2 E-5  ) .AND
     &         ( ABS(ZK(J)-ZK(I)).LE.2.E-5  )       )  THEN
C
              WRITE(3,70)NOK(I),NOK(J), XK(J), YK(J), ZK(J)
 70           FORMAT(2I8,3F9.5)
              FLAG = .TRUE.     !  SAYS THAT A PAIR HAS BEEN IDENTIFIED
            ENDIF
C
          ENDDO
C
        IF ( NOT.FLAG) WRITE (3,80)NOK(I), XK(I), YK(I), ZK(I)
 80       FORMAT(I8,'    NONE  ',3F9 5)
C
      ENDDO
C
      ENDDO
 999  STOP
      END
```

H.2 IVOLC Program: Node Location program (cylindrical coordinates)

The IVOLC program finds the nodes that belong to volumes entered as input. Coincident nodes are put in pairs.

The input file must have the same format as the file IVOLC.DAT (Table H.4). A PATRAN neutral file (i.e. PATRAN.OUT) must be available as an input to IVOLC.

The cylindrical coordinate (R, θ, Z') system used is related to the basic cartesian ccordinate system (X, Y, Z) by:

$$\begin{aligned} R &= \sqrt{X^2 + Z^2} \\ \theta &= \arctan \frac{-Z}{X} \\ Z' &= Y \end{aligned} \tag{H.1}$$

To run IVOLC, type:

```
$RUN IVOLC
ENTER INPUT FILE NAME:       <--- written by IVOLC
IVOLC.DAT                    <--- example, typed by the user
ENTER OUTPUT FILE NAME:      <--- written by IVOLC
IVOLC.OUT                    <--- example, typed by the user
```

Program limits: 15000 nodes in the model

```
*
*     TO FIND PIN NODES SUBMITTED TO FRETTING
*
PATRAN.OUT FILE :[MCGO1.GEO]PATRAN.OUT
*
*  ZONE ID,    R1 TO R2,   TETA1 TO TETA2,  ZP1 TO ZP2
*
     1          .499 .501  -180. -90.  -1.873 1.873
     1          .499 .501   90. 180.   -1.873 1.873
     2          .499 .501   -90. 90.   1.873 2.597
     3          .499 .501   -90. 90    -2.597 -1.873
```

Table H.4: Typical IVOLC program input file

- The first 3 lines are available for comments.
- The 4th line gives the PATRAN neutral file.
- The 5th line is available for comments.
- The 6th line indicates how to input the volumes whose nodes need to be identified. Each volume is identified by a user chosen ID and its upper and lower limits given in the cylindrical coordinate system
- The 7th line is available for comments.
- Each of the subsequent line gives the parameters defining a volume whose nodes need to be identified.

```
 FOR ZONE        1,
                       R FROM     0 49900 TO     0 50100
                    TETA FROM -180 00000 TO   -90 00000
                      ZP FROM   -1 87300 TO     1 87300
    NODE ID'S           R        TETA           ZP
    499     1477   0 50000-112 41609    1 66400
    501     1476   0 50000-146 20804    1 66400
    586     1464   0 50000-112 41610    0 95400

      T

  1477   NONE     0 50000-112 41609    1 66400
 FOR ZONE        1,
                       R FROM     0 49900 TO     0 50100
                    TETA FROM   90 00000 TO   180 00000
                      ZP FROM   -1 87300 TO     1 87300
    NODE ID'S           R        TETA           ZP
    500     1481   0 50000 112 41611    1 66400
    502     1480   0 50000 146 20802    1 66400
    503     1479   0 50000 180 00000    1 66400


  1481   NONE     0 50000 112 41611    1 66400
 FOR ZONE        2,
                       R FROM     0 49900 TO     0 50100
                    TETA FROM  -90 00000 TO    90 00000
                      ZP FROM    1 87300 TO     2 50000
    NODE ID'S           R        TETA           ZP
    373      829   0 50000    0 00000   1 87400
    373     1501   0 50000    0 00000   1 87400
    378      781   0 50000  -28 10402   1 87400


  1551   NONE     0 50000    0 00000    2 28350
 FOR ZONE        3,
                       R FROM     0 49900 TO     0 50100
                    TETA FROM  -90 00000 TO    90 00000
                      ZP FROM   -2 50000 TO    -1 87300
    NODE ID'S           R        TETA           ZP
    464     1021   0 50000    0 00000  -1 87400
    464     1422   0 50000    0 00000  -1 87400
    542      973   0 50000  -28 10403  -1 87400


  1423   NONE     0 50000   28 10403  -1 87400
```

Table H.5: Typical IVOLC program output file

Table H.6: IVOLC program FORTRAN code

```
C
      PROGRAM IVOLC
C     -------------
C
C
      INTEGER NO(15000), NOK(1000)
      REAL R(15000), TETA(15000), ZP(15000)
      REAL XK(1000), YK(1000), ZK(1000)
      CHARACTER*50 INPUT,OUTPUT,PATOUT
      CHARACTER*1 ALPHA
      LOGICAL FLAG
C
C     INPUTS
C     ------
C
      WRITE(6,2)
 2     FORMAT(' ENTER INPUT FILE NAME  ')
      READ(5,5) INPUT
C
      WRITE(6,4)
 4     FORMAT(' ENTER OUTPUT FILE NAME  ')
      READ(5,5) OUTPUT
C
 5    FORMAT(A)
C
      OPEN ( UNIT=2,
     &        FILE=INPUT,
     &        STATUS='OLD',
     &        ACCESS='SEQUENTIAL',
     &        FORM='FORMATTED',
     &        READONLY )
C
C
      READ(2,10) PATOUT
 10   FORMAT(///17X,A,///)
C
```

```
C
C     UNIT=3 IS THE OUTPUT
C     --------------------
C
      OPEN ( UNIT=3,
     &       FILE=OUTPUT,
     &       STATUS='NEW',
     &       ACCESS='SEQUENTIAL',
     &       CARRIAGECONTROL='LIST',
     &       FORM='FORMATTED' )
C
C
C
C     CREATES A LIST OF NODES NO(I) WITH THEIR COORD R(I), TETA(I), ZP(I)
C     -------------------------------------------------------------------
C
C
      OPEN ( UNIT=1,
     &       FILE=PATOUT,
     &       STATUS='OLD',
     &       ACCESS='SEQUENTIAL',
     &       FORM='FORMATTED',
     &       READONLY )
C
C         READS THE NUMBER OF NODES
C         -------------------------
C
      READ(1,60) NN
 60   FORMAT(//,26X,I8,/)
C
C         READS & TRANSFORMS THE NODES COORDINATES
C         ----------------------------------------
C
      DO I=1,NN
        READ(1,65) NO(I),X,Y,Z
        R(I) = SQRT(X**2+Z**2)
        IF (R(I).EQ.O.) THEN
          TETA(I)=0
        ELSE
```

```
          TETA(I)=ATAN2D(-Z,X)    ! -180 TO 180 DEG
        ENDIF
        ZP(I)=Y
      ENDDO
C
 65   FORMAT(2X,I8,/,3E16 9,/)
C
C
C     READS THE INPUT FILE
C     --------------------
C
      DO N=1,999
C
        READ(2,*,END=999) IDZONE, XI1, XI2, YI1, YI2, ZI1, ZI2
        WRITE(3,69) IDZONE, XI1, XI2, YI1, YI2, ZI1, ZI2
 69     FORMAT(//,'  FOR ZONE  ',I8,',',
     &         /,20X,'    R FROM ',F10 5,' TO ',F10 5,
     &         /,20X,' TETA FROM ',F10 5,' TO ',F10 5,
     &         /,20X,'   ZP FROM ',F10 5,' TO ',F10 5,/,
     &         /,6X,'NODE ID''S          R       TETA          ZP' )
C
C          KEEPS THE NODES SUCH THAT XI1<R<XI2, YI1<TETA<YI2, ZI1<ZP<ZI2
C          -------------------------------------------------------------
C
      NK=0
C
      DO I=1,NN
        IF (((XI1 LE.R(I)).AND.(R(I).LE.XI2))  AND.
     &      ((YI1.LE.TETA(I)) AND (TETA(I) LE YI2))  AND
     &      ((ZI1.LE.ZP(I)) AND (ZP(I) LE.ZI2)) )       THEN
            NK=NK+1
            NOK(NK)=NO(I)
            XK(NK)=R(I)
            YK(NK)=TETA(I)
            ZK(NK)=ZP(I)
        ENDIF
      ENDDO
C
C     PUTS THE NODES KEPT IN PAIRS
```

```
C     --------------------------
C
      DO I=1,NK
         IP1=I+1
         FLAG=.FALSE.
         DO J=IP1,NK
           IF(( ABS(XK(J)-XK(I)) LE 2 E-5  )  AND
     &          ( ABS(YK(J)-YK(I)) LE 2 E-5  )  AND
     &          ( ABS(ZK(J)-ZK(I)) LE 2 E-5  )        )  THEN
C
            WRITE(3,70)NOK(I),NOK(J), XK(J), YK(J), ZK(J)
 70         FORMAT(2I8,3F10 5)
            FLAG =  TRUE      !  SAYS THAT A PAIR HAS BEEN IDENTIFIED
          ENDIF
C
        ENDDO
C
      IF (.NOT FLAG) WRITE (3,80)NOK(I), XK(I), YK(I), ZK(I)
 80      FORMAT(I8,'   NONE ',3F10 5)
C
      ENDDO
C
      ENDDO
 999  STOP
      END
```

Appendix I

Displacement Definition Program

The programs DF61, DF62, DF9 are used to define the boundary displacement fields on the lug & bushing F.E. mesh (see Section 3.5).

I.1 DF61 Program

This program computes the 16 algebraic coefficients (see eqn(3.3)) required to define a displacement field from the knowledge of the displacements at 6 points located on the parametric coordinate system at (See Figure I.1(a)):

$$(\xi_1, \xi_2) = \begin{array}{l} 0,0 \\ 0,1 \\ \frac{1}{2},1 \\ 1,1 \\ 1,0 \\ \frac{1}{2},0 \end{array}$$

Refer to PATRAN user guide [6], Section 7.13.1 . The input file must have the same format as the DF61.DAT file (Table I.1). The DF61 output file (Table I.3) must be run as a PATRAN session file.

To run DF61, type:

```
$RUN DF61
ENTER INPUT FILE NAME:        <--- written by DF61
```

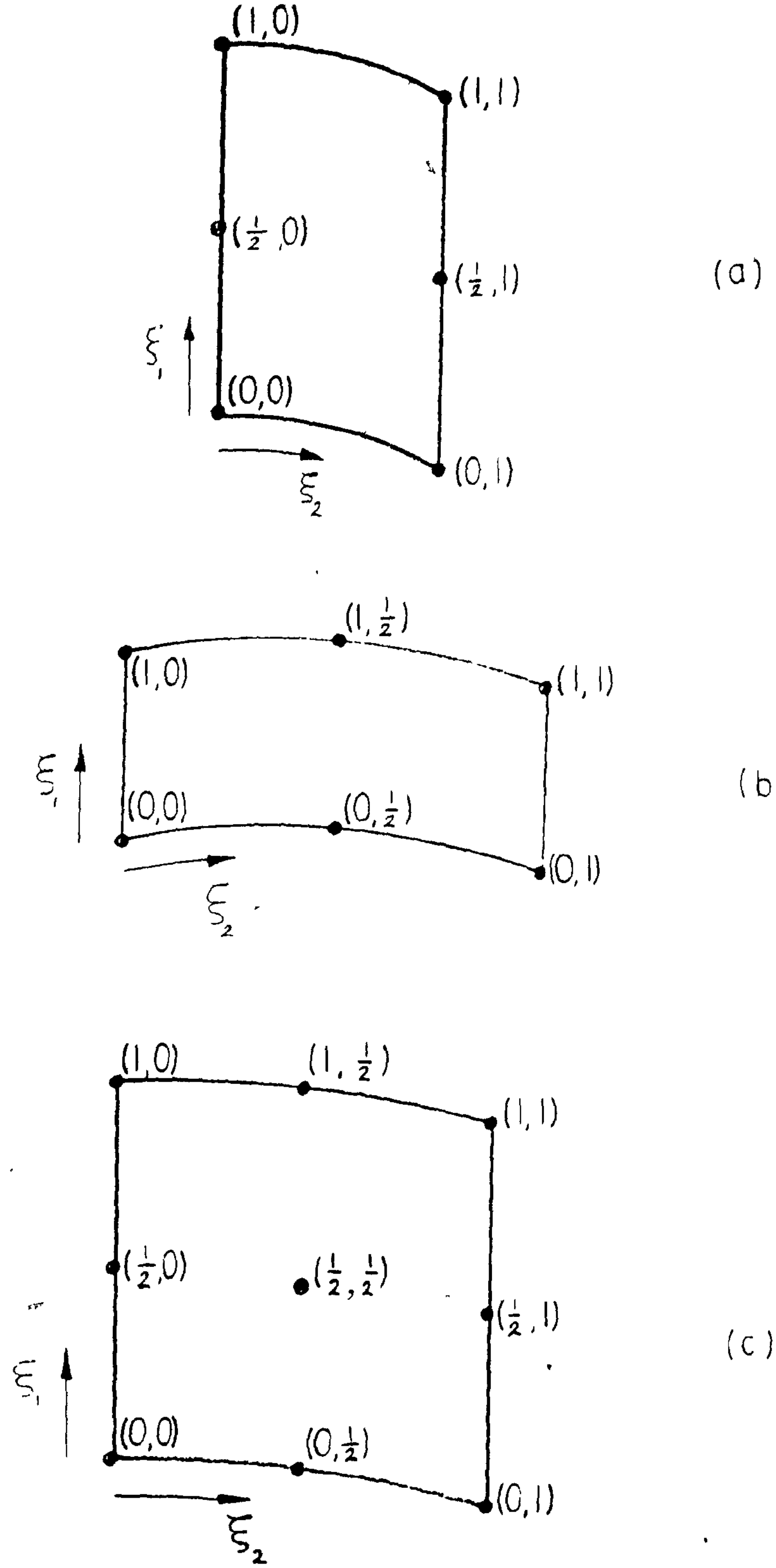


Figure I.1: Point positions (ξ_1, ξ_2) where displacements are known.

```
DF61.DAT                      <--- example, typed by the user
ENTER OUTPUT FILE NAME:       <--- written by DF61
DF61.OUT                      <--- example, typed by the user
```

```
*
*
*
SCALE             001
FIRST  DPATCH  ID 101
CID                2
*
*  DFEG ID      HP   FACE   DIR   DISPLACEMENTS (6 VALUES)
   11           605  4      1     6 02 6 44 5 21 2 56 2 27 4 81
   11           606  4      1     2 27 2 56 - 62 -2 98 -2 99 - 749
   11           617  4      1     3 57 3 81 5 91 6 44 6 02 5 55
   11           618  4      1     -1 66 -1 73  771 3 81 3 57  728
   12           605  4      1     7 93 8 39 6 44 2 79 2 42 5 98
   12           606  4      1     2 42 2 79 -1 47 -4 54 -4 63 -1 64
   12           617  4      1     5 47 5 71 8 11 8 39 7 93 7 72
   12           618  4      1     -1 41 -1 54 1 86 5 71 5 47 1 89
   13           605  4      1     3 19 3 58 3 40 2 24 2 04 3 07
   13           606  4      1     2 04 2 24  055 - 878 - 793  514
   13           617  4      1      769 1 03 2 71 3 58 3 19 2 34
   13           618  4      1     -2 05 -2 06 - 767 1 03  769 - 895
   14           605  4      1     2 42 2 86 3 17 2 55 2 37 2 83
   14           606  4      1     2 37 2 56 1 23 - 027  126 1 27
   14           617  4      1     3 23  02 1 67 2 85 2 42 1 25
   31           608  4      3     - 903 - 533 -1 25 -1 99 -1 97 -1 39
   31           602  4      3     -1 97 -1 99 -2 92 -3 51 -3 09 -2 67
   31           614  4      3     -3 09 -3 51 -3 41 -2 77 -2 54 -3 02
   31           620  4      3     -2 54 -2 77 -2 07 -1 22 -1 41 -1 99
   32           608  4      3     -1 55 -1 1 -1 67 -2 21 -2 39 -1 93
   32           602  4      3     -2 39 -2 21 -3 12 -3 98 -3 73 -3 11
   32           614  4      3     -3 73 -3 98 -4 34 -4 14 -3 78 -3 96
   32           620  4      3     -3 78 -4 14 -3 64 -2 72 -2 74 -3 38
   32           603  5      3     -3 98 -4 21 -3 1 -2 05 -2 21 -3 12
   32           622  1      3     -2 72 -2 81 -3 93 -4 48 -4 14 -3 64
   33           620  4      3      704  602 1 73 2 86 2 35 1 55
   33           619  4      3     2 35 2 86 3 67 3 84 2 96 2 86
   33           607  4      3     2 96 3 84 3 28 2 16 1 84 2 57
   33           608  4      3     1 84 2 16  903 - 107  186  964
   33           622  1      3     2 86 3 3 1 85  469  602 1 73
   33           621  1      3     3 84 4 64 4 39 3 3 2 86 3 67
   34           620  4      3     1 83 1 78 3 26 4 61 3 87 2 92
   34           619  4      3     3 87 4 61 5 4 5 34 4 31 4 38
   34           607  4      3     4 31 5 34 4 42 2 88 2 61 3 68
   34           608  4      3     2 61 2 88 1 21  087  565 1 43
   34           622  1      3     4 61 5 26 3 51 1 69 1 78 3 26
   34           621  1      3     5 34 6 25 6 29 5 26 4 61 5 4
```

Table I.1: Typical DF61 program input file

The file format is given in Table I.2.

- The first 3 lines are available for comments.
- The 4th, 5th, and 6th lines are the scaling factor, the first data patch ID that will be used to define the imposed displacement, and the reference coordinate system ID (only the values right to the ":" can be changed). Make sure that all the data patch ID's that will be created by DF61 are unique.
- The 7th and 8th lines are available for comments.
- Each of the other lines indicates an imposed displacement It is defined by:
  - a DFEG ID,
  - the hyperpatch (HP) the DFEG is applied to,
  - the HP face (1 to 6) [6],
  - the direction (1, 2, or 3),
  - and the 6 displacements applied on the surface indicated by the HP and the FACE. They must be entered in the (ξ_1, ξ_2) order shown on page 255

Table I.2: DF61 program input file format

See Table I.1.

```
DATA,101,.........,-0.1800E+00, 0.5000E-01, 0.4200E+00,,
-0.2660E+01,-0.1090E+01, 0.6020E+01
DPAT,101,ALG,0.0010,D101
DF,H605,DISP,P101,    11,F4,2
DATA,102,.........,  0.8400E-01,-0.3640E+00, 0.2900E+00,,
 0.1556E+01,-0.6816E+01, 0.2270E+01
DPAT,102,ALG,0.0010,D102
DF,H606,DISP,P102,    11,F4,2
DATA,103,.........,-0.1200E+00, 0.3000E+00, 0.2400E+00,,
-0.3020E+01, 0.5470E+01, 0.3570E+01
DPAT,103,ALG,0.0010,D103
DF,H617,DISP,P103,    11,F4,2
DATA,104,.........,  0.1680E+00, 0.1420E+00,-0.7000E-01,,
 0.9080E+00, 0.4322E+01,-0.1660E+01
DPAT,104,ALG,0.0010,D104
DF,H618,DISP,P104,    11,F4,2

     .
     .
     .

DATA,136,.........,  0.8000E-01,-0.8200E+00, 0.6500E+00,,
-0.2200E+00,-0.2610E+01, 0.4610E+01
DPAT,136,ALG,0.0010,D136
DF,H622,DISP,//P136,    34,F1,2
DATA,137,.........,-0.4400E+00, 0.1800E+00, 0.9100E+00,,
-0.1700E+01, 0.9700E+00, 0.5340E+01
DPAT,137,ALG,0.0010,D137
DF,H621,DISP,//P137,    34,F1,2
```

Table I.3: Typical DF61 program output file

It corresponds to the input file given by Table I.1.

Table I.4: DF61 program FORTRAN code

```
C
      PROGRAM DF61
C     ------------
C
C
      CHARACTER*50 INPUT,OUTPUT
      INTEGER HPID,DFID,FACE,DIR,DPID,CID
C
C     INPUTS
C     ------
C
      WRITE(6,2)
 2     FORMAT(' ENTER INPUT FILE NAME: ')
      READ(5,5) INPUT
C
      WRITE(6,4)
 4     FORMAT(' ENTER OUTPUT FILE NAME: ')
      READ(5,5) OUTPUT
C
 5    FORMAT(A)
C
      OPEN ( UNIT=1,
     &        FILE=INPUT,
     &        STATUS='OLD',
     &        ACCESS='SEQUENTIAL',
     &        FORM='FORMATTED',
     &        READONLY )
C
C
      OPEN ( UNIT=2,
     &        FILE=OUTPUT,
     &        STATUS='NEW',
     &        ACCESS='SEQUENTIAL',
     &        CARRIAGECONTROL='LIST',
     &        FORM='FORMATTED' )
C
```

```
C
C     READS GENERAL INPUTS
C     --------------------
C
      READ(1,10)SCALE,DPID,CID
 10   FORMAT(///,18X,F6 4,/,2(18X,I8/),/)
C
C
      DO I = 1,999
C
C       READS EACH CASE
C       ---------------
C
        READ(1,*,ERR=999)DFID,HPID,FACE,DIR,D1,D2,D4,D6,D5,D3
C
C       COMPUTES THE ALGEBRAIC COEFFICIENTS
C       (SEE SECTION 7.13.1 IN PATRAN USER GUIDE)
C       -----------------------------------------
C
        SOO = D1
        SO1 = D2-D1
        S10 = 4.*(D3-D1) -    (D5-D1)
        S20 = 2.*(D5-D1) - 4.*(D3-D1)
C
         E4 = D4 -SOO -.5*S10 - 25*S20 -    SO1
         E6 = D6 -SOO -    S10 -     S20 -    SO1
C
        S11 = 4.*E4 - E6
        S21 = 2.*(E6-2.*E4)
C
C       DATA PATCH DEFINITION
C       ---------------------
C
        WRITE(2,60)DPID,S21,S11,SO1
        WRITE(2,62)S20,S10,SOO
C
 60     FORMAT('DATA,',I3,',,,,,,,,,,',3(E11.4,','),',')
 62     FORMAT(2(E11.4,','),E11.4)
C
```

```
      WRITE(2,65)DPID,SCALE,DPID
 65   FORMAT('DPAT,',I3,',ALG,',F6 4,',D',I3)
C
C     DFEG DEFINITION
C     ---------------
C
      IF (DIR.EQ 1) THEN
         WRITE(2,70)HPID,DPID,DFID,FACE,CID
      ELSEIF (DIR.EQ.2) THEN
         WRITE(2,80)HPID,DPID,DFID,FACE,CID
      ELSE
         WRITE(2,90)HPID,DPID,DFID,FACE,CID
      ENDIF
C
 70   FORMAT('DF,H',I3,',DISP,P',I3,',',I5,',F',I1,',',I1)
 80   FORMAT('DF,H',I3,',DISP,/P',I3,',',I5,',F',I1,',',I1)
 90   FORMAT('DF,H',I3,',DISP,//P',I3,',',I5,',F',I1,',',I1)
      DPID = DPID + 1
C
    ENDDO
C
C
 999  STOP
C
      END
```

I.2 DF62 Program

This program computes the 16 algebraic coefficients (see eqn(3.3)) required to define a displacement field from the knowledge of the displacements at 6 points located on the parametric coordinate system at (see Figure I.1(b)):

$$
(\xi_1, \xi_2) = \begin{array}{l} 0,0 \\ 0,\frac{1}{2} \\ 0,1 \\ 1,1 \\ 1,\frac{1}{2} \\ 1,0 \end{array}
$$

Refer to PATRAN user guide [6] Section 7.13.1 . The input file must have the same format as the DF62.DAT file (Table I.5). The DF62 output (Table I.6) file must be run as a PATRAN session file.

To run DF62, type:

```
$RUN DF62
ENTER INPUT FILE NAME:          <--- written by DF62
DF62.DAT                        <--- example, typed by the user
ENTER OUTPUT FILE NAME:         <--- written by DF62
DF62.OUT                        <--- example, typed by the user
```

```
*
*
*
SCALE              :.001
FIRST  DPATCH  ID:201
CID                :2
*
*  DFEG ID      HP    FACE    DIR    DISPLACEMENTS
   32           610   3       3      -1.1 -1.67 -2.21 -2.05 -1.48 -.791
   32           615   5       3      -3.98 -4.34 -4.14 -4.48 -4.67 -4.21
   33           609   3       3      3.84 3.28 2.16 2.42 3.9 4.64
   33           610   3       3      2.16 .903 -.107 -.422 .82 2.42
   34           609   3       3      5.34 4.42 2.88 3.11 5 09 6.25
   34           610   3       3      2.88 1.21 .087 -.416  956 3.11
```

Table I.5: Typical DF62 input file

- The first 3 lines are available for comments.
- The 4th, 5th, and 6th lines are the scaling factor, the first data patch ID that will be used to define the imposed displacement, and the reference coordinate system ID (only the values right to the ":" can be changed). Make sure that all the data patch ID's that will be created by DF62 are unique.
- The 7th and 8th lines are available for comments.
- Each of the other lines indicates an imposed displacement. It is defined by:
  - a DFEG ID,
  - the hyperpatch (HP) the DFEG is applied to,
  - the HP face (1 to 6) [6],
  - the direction (1, 2, or 3),
  - and the 6 displacements applied on the surface indicated by the HP and the FACE. They must be entered in the (xi_1, xi_2) order shown on page 264.

```
DATA,201,......, 0.1780E+00, 0.6000E-01,..
-0.3270E+00,-0.1170E+01,..
 0.3090E+00,-0.1100E+01
DPAT,201,ALG,0.0010,D201
DF,H610,DISP,//P201,    32,F3,2
DATA,202,......, 0.1800E+00, 0.1120E+01,..
-0.2900E+00,-0.1280E+01,..
-0.2300E+00,-0.3980E+01
DPAT,202,ALG,0.0010,D202
DF,H615,DISP,//P202,    32,F5,2
DATA,203,......,-0.3600E+00,-0.1120E+01,..
-0.1800E+00,-0.5600E+00,..
 0.8000E+00, 0.3840E+01
DPAT,203,ALG,0.0010,D203
DF,H609,DISP,//P203,    33,F3,2
DATA,204,......, 0.2220E+00, 0.4940E+00,..
-0.7970E+00,-0.2761E+01,..
 0.2600E+00, 0.2160E+01
DPAT,204,ALG,0.0010,D204
DF,H610,DISP,//P204,    33,F3,2
DATA,205,......,-0.4000E+00,-0.1240E+01,..
-0.2800E+00,-0.1220E+01,..
 0.9100E+00, 0.5340E+01
DPAT,205,ALG,0.0010,D205
DF,H609,DISP,//P205,    34,F3,2
DATA,206,......, 0.4700E+00, 0.1094E+01,..
-0.1203E+01,-0.3887E+01,..
 0.2300E+00, 0.2880E+01
DPAT,206,ALG,0.0010,D206
DF,H610,DISP,//P206,    34,F3,2
```

Table I.6: Typical DF62 output file

It corresponds to the input file given by Table I.5.

Table I.7: DF62 program FORTRAN code

```
C
      PROGRAM DF62
C     ------------
C
      CHARACTER*50  INPUT,OUTPUT
      INTEGER HPID,DFID,FACE,DIR,DPID,CID
C
C     INPUTS
C     ------
C
      WRITE(6,2)
 2     FORMAT(' ENTER INPUT FILE NAME  ')
      READ(5,5) INPUT
C
      WRITE(6,4)
 4     FORMAT(' ENTER OUTPUT FILE NAME  ')
      READ(5,5) OUTPUT
C
 5    FORMAT(A)
C
      OPEN ( UNIT=1,
     &        FILE=INPUT,
     &        STATUS='OLD',
     &        ACCESS='SEQUENTIAL',
     &        FORM='FORMATTED',
     &        READONLY )
C
C
      OPEN ( UNIT=2,
     &        FILE=OUTPUT,
     &        STATUS='NEW',
     &        ACCESS='SEQUENTIAL',
     &        CARRIAGECONTROL='LIST',
     &        FORM='FORMATTED' )
C
C
```

```
C      READS GENERAL INPUTS
C      --------------------
C
       READ(1,10)SCALE,DPID,CID
  10   FORMAT(///,18X,F6 4,/,2(18X,I8/),/)
C
C
       DO I = 1,999
C
C        READS EACH CASE
C        ---------------
C
         READ(1,*,ERR=999)DFID,HPID,FACE,DIR,D1,D3,D5,D6,D4,D2
C
C        COMPUTES THE ALGEBRAIC COEFFICIENTS
C        (SEE SECTION 7.13.1 IN PATRAN USER GUIDE)
C        -----------------------------------------
C
         S00 = D1
         S10 = D2-D1
         S01 = 4.*(D3-D1) -    (D5-D1)
         S02 = 2.*(D5-D1) - 4.*(D3-D1)
C
         E4 = D4 -S00 -.5*S01 -.25*S02 -   S10
         E6 = D6 -S00 -    S01 -    S02 -   S10
C
         S11 = 4.*E4 - E6
         S12 = 2.*(E6-2.*E4)
C
C        DATA PATCH DEFINITION
C        ---------------------
C
         WRITE(2,60)DPID,S12,S02
         WRITE(2,61)S11,S01
         WRITE(2,62)S10,S00
C
 60      FORMAT('DATA,',I3,',,,,,,,',2(E11.4,','),',,')
 61      FORMAT(2(E11.4,','),',,')
 62      FORMAT(E11.4,',',E11.4)
```

```
C
      WRITE(2,65)DPID,SCALE,DPID
 65   FORMAT('DPAT,',I3,',ALG,',F6 4,',D',I3)
C
C     DFEG DEFINITION
C     ---------------
C
      IF (DIR.EQ.1) THEN
         WRITE(2,70)HPID,DPID,DFID,FACE,CID
      ELSEIF (DIR.EQ.2) THEN
         WRITE(2,80)HPID,DPID,DFID,FACE,CID
      ELSE
         WRITE(2,90)HPID,DPID,DFID,FACE,CID
      ENDIF
C
 70   FORMAT('DF,H',I3,',DISP,P',I3,',',I5,',F',I1,',',I1)
 80   FORMAT('DF,H',I3,',DISP,/P',I3,',',I5,',F',I1,',',I1)
 90   FORMAT('DF,H',I3,',DISP,//P',I3,',',I5,',F',I1,',',I1)
      DPID = DPID + 1
C
    ENDDO
C
C
 999 STOP
C
    END
```

I.3 DF9 Program

This program computes the 16 algebraic coefficients (see eqn(3.3)) required to define a displacement field from the knowledge of the displacements at 9 points located on the parametric coordinate system at (see Figure I.1(c)):

$$(\xi_1, \xi_2) =$$

$$0,0$$
$$0,\tfrac{1}{2}$$
$$0,1$$
$$\tfrac{1}{2},1$$
$$1,1$$
$$1,\tfrac{1}{2}$$
$$1,0$$
$$\tfrac{1}{2},0$$
$$\tfrac{1}{2},\tfrac{1}{2}$$

Refer to PATRAN user guide [6] Section 7.13.1 . The input file must have the same format as the DF9.DAT file (Table I.8). The DF9 output (table I.10) file must be run as a PATRAN session file.

To run DF9, type:

```
$RUN DF9
ENTER INPUT FILE NAME:        <--- written by DF9
DF9.DAT                       <--- example, typed by the user
ENTER OUTPUT FILE NAME:       <--- written by DF9
DF9.OUT                       <--- example, typed by the user
```

```
*
*
*
SCALE            001
FIRST  DPATCH  ID 1
CID              2
*
*  DFEG ID     HP   FACE   DIR   D1
   11          611   2      1  6 02 4.81 2.27 1 5  936 2 93 4 01 4 93 3 78
   11          612   2      1  2 27 - 749 -2 99 -3.01 -3.12 -1 42  936 1 5 -1 18
   11          623   2      1  3 57 5 55 6 02 4 93 4.01 3 91 2 66  3 02 4 64
   11          624   2      1  -1 66 .728 3.57 3.02 2 66  622 -1 36 -1 46  598
   12          611   2      1  7 93 5 98 2 42 1 45  732 3 8 5 75 6 72 4 77
   12          612   2      1  2 42 -1 64 -4 63 -4 81 -4 90 -2 63  732 1 45 -2 24
   12          623   2      1  5 47 7 72 7 93 6 72 5.75 6 08 4 75  4 99 6 79
   12          624   2      1  -1 41 1 89 5 47 4 99 4 75 2 14 - 716 -1 01 1 95
   13          611   2      1  3 19 3 07 2 04 1.49 1.07 1 35 1 1 2.08 2 15
   13          612   2      1  2 04  514 -.793 - 59 - 39  368 1 07 1 49  369
   13          623   2      1   769 2 34 3 19 2.08 1.1  383  - 622  016 1 3
   13          624   2      1  -2 05 - 895  769  016 - 622 -1 61 -2 06 -2 05 -1 3
   14          611   2      1  2 42 2 83 2.37 1.91 1 53  997 .028  1 17 1 86
   14          612   2      1  2 37 1 27  126  553  991 1 52 1 53  1 91 1 35
   14          623   2      1  - 323 1 25 2 42 1.17  028 -1 15 -2 22 -1 34 - 009
   14          624   2     1 -2 49 -1.7 - 323 -1.34 -2 22 -2 73 -2 63 -2 56 -2 26
   21      103    2       3  -2 39 -2 82 -3 04 -2 91 -2 87  -2 63 -2 24 -2 31 -2 70
   21      114    2       3  -3 04 -3 07 -2 84 -2 86 -2.90  -2 93 -2 87 -2 91 -2 99
   22      103    2       3  -3.19 -3 7 -4 09 -3 87 -3 78 -3 3 -2 8 -2 96 -3 47
   22      114    2       3  -4 09 -4 36 -4.29 -4 34 -4 41  -4.12 -3 78 -3 87 -4 22
   23      107    2       3  1 55 .695 - 069 - 171 - 307  702 1 95 1 75 .705
   23      103    2       3  - 069 - 618 - 715 - 858 - 962  - 888 - 307 - 171 - 761
   23      114    2       3  - 715 - 348 .404  399  377 - 589 - 962 - 858 - 448
   23      118    2       3   404 1 22 2  2 32 2 64 1 49  377  399 1 37
   24      107    2       3  2 33 1.27  389  165 - 105 1.06 2 64 2 49 1 18
   24      103    2       3   389 - 175 -.121 -.32 - 479 - 623 - 105  165 - 415
   24      114    2       3  - 121  514 1 58 1 59 1 55  224 -.479 - 32  369
   24      118    2       3  1.58 2 59 3 45 3 89 4 34 3 02 1.55 1 59 2 81
   24      117    2       3  3 45 3 93 3 86 4 41 4 96 5 05 4 34 3 89 4 48
```

Table I.8: Typical DF9 program input file

The format is given in Table I.9

- The first 3 lines are available for comments.
- The 4th, 5th, and 6th lines are the scaling factor, the first data patch ID that will be used to define the imposed displacement, and the reference coordinate system ID (only the values right to the ":" can be changed). Make sure that all the data patch ID's that will be created by DF9 are unique.
- The 7th and 8th lines are available for comments.
- Each of the other lines indicates an imposed displacement. It is defined by.
  - a DFEG ID,
  - the hyperpatch (HP) the DFEG is applied to,
  - the HP face (1 to 6) [6],
  - the direction (1, 2, or 3),
  - and the 9 displacements applied on the surface indicated by the HP and the FACE. They must be entered in the (xi_1, xi_2) order shown on page 270.

Table I.9: DF9 program input file format

See Table I.8

```
DATA, 1,....., 0.6400E-01, 0.7680E+00,-0.2660E+01,,
 0.7998E-02,-0.1640E+00,-0.1090E+01,,
 0.3400E+00,-0.2350E+01, 0.6020E+01
DPAT, 1,ALG,0.0010,D  1 .
DF,H611,DISP,P  1,   11,F2,2
DATA, 2,.....,-0.1064E+01, 0.8200E+00, 0.1556E+01,,
 0.4720E+00, 0.9760E+00,-0.6816E+01,,
 0.4120E+00,-0.1746E+01, 0.2270E+01
DPAT, 2,ALG,0.0010,D  2
DF,H612,DISP,P  2,   11,F2,2
DATA, 3,.....,-0.2861E-05, 0.7200E+00,-0.3020E+01,,
-0.4000E-01,-0.1780E+01, 0.5470E+01,,
 0.3800E+00,-0.1290E+01, 0.3570E+01
DPAT, 3,ALG,0.0010,D  3
DF,H623,DISP,P  3,   11,F2,2
DATA, 4,.....,-0.8720E+00; 0.7600E-01, 0.9080E+00,,
 0.1452E+01,-0.1866E+01, 0.4322E+01,,
-0.2000E+00, 0.5000E+00,-0.1660E+01
DPAT, 4,ALG,0.0010,D  4
DF,H624,DISP,P  4,   11,F2,2
      .
      .
      .
DATA, 28,.....,-0.8000E-01, 0.8000E-01,-0.3000E+00,,
 0.2000E+00, 0.7200E+00, 0.2170E+01,,
-0.1000E+00, 0.7000E-01, 0.1580E+01
DPAT, 28,ALG,0.0010,D 28
DF,H118,DISP,//P 28,   24,F2,2
DATA, 29,.....,-0.1200E+00,-0.3800E+00,-0.1100E+01,,
 0.1000E+00, 0.6100E+00, 0.1510E+01,,
 0.2000E-01, 0.8700E+00, 0.3450E+01
DPAT, 29,ALG,0.0010,D 29
DF,H117,DISP,//P 29,   24,F2,2
```

Table I.10: Typical DF9 program output file

It corresponds to the input file given by Table I.8.

Table I.11: DF9 program FORTRAN code

```
C
      PROGRAM DF9
C     -----------
C
      CHARACTER*50 INPUT,OUTPUT
      INTEGER HPID,DFID,FACE,DIR,DPID,CID
C
C     INPUTS
C     ------
C
      WRITE(6,2)
 2     FORMAT(' ENTER INPUT FILE NAME  ')
      READ(5,5) INPUT
C
      WRITE(6,4)
 4     FORMAT(' ENTER OUTPUT FILE NAME. ')
      READ(5,5) OUTPUT
C
 5    FORMAT(A)
C
      OPEN ( UNIT=1,
     &       FILE=INPUT,
     &       STATUS='OLD',
     &       ACCESS='SEQUENTIAL',
     &       FORM='FORMATTED',
     &       READONLY )
C
C
      OPEN ( UNIT=2,
     &       FILE=OUTPUT,
     &       STATUS='NEW',
     &       ACCESS='SEQUENTIAL',
     &       CARRIAGECONTROL='LIST',
     &       FORM='FORMATTED' )
C
C
```

```
C     READS GENERAL INPUTS
C     --------------------
C
      READ(1,10)SCALE,DPID,CID
 10   FORMAT(///,18X,F6.4,/,2(18X,I8/),/)
C
C
      DO I = 1,999
C
C        READS EACH CASE
C        ---------------
C
         READ(1,*,ERR=999)DFID,HPID,FACE,DIR,D1,D2,D3,D6,D9,D8,D7,D4,D5
C
C        COMPUTES THE ALGEBRAIC COEFFICIENTS
C        (SEE SECTION 7.13.1 IN PATRAN USER GUIDE).
C        ------------------------------------------
C
         S00 = D1
         S01 = 4.*(D2-D1) -    (D3-D1)
         S02 = 2.*(D3-D1) - 4.*(D2-D1)
         S10 = 4.*(D4-D1) -    (D7-D1)
         S20 = 2.*(D7-D1) - 4.*(D4-D1)
C
         E5 = D5 -S00 -.5*S10 -.25*S20 -.5*S01 -.25*S02
         E6 = D6 -S00 -.5*S10 -.25*S20 -   S01 -    S02
         E8 = D8 -S00 -   S10 -    S20 -.5*S01 -.25*S02
         E9 = D9 -S00 -   S10 -    S20 -   S01 -    S02
C
         F5 = 16.*E5
         F6 = F5 -8.*E6
         F8 = F5 -8.*E8
         F9 = 4.*E9 -F5
C
        S22 = F6+F8+F9
        S12 = -.5*(F6+S22)
        S21 = -.5*(F8+S22)
        S11 = .25*(F5 -S22 -2.*(S21+S12))
C
```

```
C
C       DATA PATCH DEFINITION
C       --------------------
C
        WRITE(2,60)DPID,S22,S12,S02
        WRITE(2,61)S21,S11,S01
        WRITE(2,62)S20,S10,S00
C
 60     FORMAT('DATA,',I3,',,,,,,',3(E11.4,','),',')
 61     FORMAT(3(E11.4,','),',')
 62     FORMAT(2(E11.4,','),E11.4)
C
        WRITE(2,65)DPID,SCALE,DPID
 65     FORMAT('DPAT,',I3,',ALG,',F6.4,',D',I3)
C
C       DFEG DEFINITION
C       ---------------
C
        IF (DIR.EQ.1) THEN
           WRITE(2,70)HPID,DPID,DFID,FACE,CID
        ELSEIF (DIR.EQ.2) THEN
           WRITE(2,80)HPID,DPID,DFID,FACE,CID
        ELSE
           WRITE(2,90)HPID,DPID,DFID,FACE,CID
        ENDIF
C
 70     FORMAT('DF,H',I3,',DISP,P',I3,',',I5,',F',I1,',',I1)
 80     FORMAT('DF,H',I3,',DISP,/P',I3,',',I5,',F',I1,',',I1)
 90     FORMAT('DF,H',I3,',DISP,//P',I3,',',I5,',F',I1,',',I1)
        DPID = DPID + 1
C
      ENDDO
C
C
 999  STOP
C
      END
```

Appendix J

Commented Versions of the Data Decks used for the Finite Element Analysis

Table J.1: Commented version of the NASTRAN data deck used for the analysis of the interference fit induced stresses

```
ID GC, MCGO1
$
$---------------------
$ Solution "24": linear
$
SOL 24
$---------------------
TIME 25
$
$----------------------------------------------------------------------
$ THE LINES BELOW ARE NECESSARY TO CREATE AN OUTPUT FILE THAT CAN BE POST
$ PROCESSED
$
ALTER      187  $
MATMOD     CSTM,SIL,BGPDT,,,/TRANSGB,/5//-1  $
MPYAD      TRANSGB,UGV,/UGVBASIC  $
SDR2       CASECC,,,EQEXIN,,,,,,UGVBASIC,,/
     ,,OUGV1PAT,,,/STATICS/S,N,NOSORT2/V,N,NOCOMP $
OUTPUT2    OUGV1PAT,OES1X//-1/11/V,N,Z  $
```

```
$
$-----------------------------------------------------
$ THE NEXT LINE IS TO MINIMIZE THE STIFFNESS MATRIX BAND WIDTH
..BM 2 RFA RF24D74
$
$-----------------------------------------------------
CEND
TITLE     =     LUG & BUSHING
SUBTITLE = INTERFERENCE FIT INDUCED STRESS
ECHO=UNSORT
$
SUBCASE   = 1
LABEL     =  ( 0.001 INCH ON THE  RADIUS )
SPC=100
MPC=200
DEFORM=104
LOAD=100
DISPL=ALL
STRESS(VONM)=ALL
STRAIN=NONE
FORCE =  ALL
$
BEGIN BULK
$
$-----------------------------------------------------------------
$ The coordinate system 2 is the cylindrical coordinate system R,THETA,Z'
$ as defined in Chapter 2.
$ The coordinate system 3 is not used here.
$
CORD2C        2       0.      0.      0.      0.      1.00000 0.      CF      1
+CF     1 1.00000 1.00000 0.
CORD2R        3       0.      0.      0.      1.00000 0.      0.      CF      2
+CF     2 1.00000 1.00000 0.
$
$-----------------------------------------------
$   (All FORCE cards define very small DUMMY loadings.
$   They were necessary to avoid any error message.)
$
FORCE         100     1642    0       1.      1.
FORCE         100     688     0       1.      1.
$
$-----------------------------------------------------------------
$ Node 1427 belonging to the bushing has the same THETA and Z' motions as the
$ coincident node 611 that belongs to the lug.
```

```
$
$    TO AVOID BUSHING THETA RIGID BODY MOTION
MPC         200    1427     2      1.      661       2      -1.
$    TO AVOID BUSHING Z' RIGID BODY MOTION
MPC         200    1427     3      1.      661       3      -1.
$
$-------------------------------------------------------------------------
$ The DEFORM cards below define the INTERFERENCE FIT (0.001 INCH ON RADIUS).
$ They were created by the IBAR program.
$
DEFORM      104    12001 0.00100
DEFORM      104    12002 0.00100
DEFORM      104    12003 0.00100
DEFORM      104    12004 0.00100
DEFORM      104    12005 0.00100
DEFORM      104    12006 0,00100
DEFORM      104    12007 0.00100
DEFORM      104    12008 0.00100
DEFORM      104    12009 0.00100
             .
             .
             .
DEFORM      104    12118 0.00100
DEFORM      104    12119 0.00100
DEFORM      104    12120 0.00100
$
$---------------------------------------------
$ The lug end is constrained in R, THETA, Z'.
$ The SPC1 cards below were created by PATRAN.
$
SPC1        100     123      1       2       3       5       8      11
SPC1        100     123     17      19      25      66      68      72
SPC1        100     123    139     141     145     212     214     218
SPC1        100     123    285     287     291     362     364     368
SPC1        100     123    139     141     145
SPC1        100     123    438     440     444
$
$-------------------------------------------------------------------------
$ NO NODE ROTATION ALLOWED. The SPC1 card below has been manually entered.
$
SPC1        100     456      1     THRU    1932
$
$-------------------------------------------------------------------------
$ NODE DEFINITION. The GRID cards below were created by PATRAN and modified
```

```
$ by TRONC.
$ The last digit of each coordinate was removed by TRONC to make sure that
$ coincident nodes do coincide exactly.
$
GRID           1       2  2.7563 160.612  2.0050        2
GRID           2       2  2.7563 160.612  2.1845        2
GRID           3       2  2.7563 160.612  2.3640        2
GRID           4       2  2.2867 156.485  2.0043        2
GRID           5       2  2.6890 165.214  2.0050        2
GRID           6       2  2.2623 162.121  2.0045        2
GRID           7       2  2.2867 156.485  2.1838        2
GRID           8       2  2.6890 165.214  2.1845        2
GRID           9       2  2.2623 162.121  2.1840        2
               .
               .
               .
GRID        1930       2  0.5333 -56.208  2.0706        2
GRID        1931       2  0.6000 -56.208  2.0706        2
GRID        1932       2  0.5667 -56.208  2.0706        2
$
$-----------------------------------------------
$ SOLID element definition as created by PATRAN.
$
CHEXA       1001       1     153     150      77      82     152     149 E      1
+E      1     76      80
CHEXA       1002       1     150     147      74      77     149     144 E      2
+E      2     71      76
CHEXA       1003       1      82      77      30      33      80      76 E      3
+E      3     29      32
CHEXA       1004       1      77      74      27      30      76      71 E      4
+E      4     22      29
CHEXA       1006       1      30      27      24      65      29      22 E      5
+E      5     16      63
CHEXA       1008       1      65      24      23      64      63      16 E      6
+E      6     15      61
CHEXA       1009       1     152     149      76      80     151     148 E      7
+E      7     75      78
CHEXA       1010       1     149     144      71      76     148     143 E      8
+E      8     70      75
               .
               .
               .
CPENTA      2005       1     373     296     299     372     295     298
CPENTA      2006       1     299     376     373     298     375     372
```

```
CPENTA    2007    1    449    373    376    448    372    375

CHEXA     6766    6    1878   1877   1799   1800   1890   1889 E  1294
+E  1294  1811    1812
CHEXA     6767    6    1877   1876   1798   1799   1889   1888 E  1295
+E  1295  1810    1811
CHEXA     6768    6    1876   1875   1797   1798   1888   1887 E  1296
+E  1296  1809    1810
$
$-------------------------------------------------------------------
$  BAR element definition cards.  They were created by the IBAR program.
$
CBAR      12001   120     827    196    0.     1.     0.     E 12001
+E 12001     23     23-0.00100              0.00100
CBAR      12002   120     832    197    0.     1.     0.     E 12002
+E 12002     23     23-0.00100              0.00100
CBAR      12003   120     833    198    0.     1.     0.     E 12003
+E 12003     23     23-0.00100              0.00100
CBAR      12004   120     841    202    0.     1.     0.     E 12004
+E 12004     23     23-0.00100              0.00100
CBAR      12005   120     848    204    0.     1.     0.     E 12005
+E 12005     23     23-0.00100              0.00100
CBAR      12006   120     896    126    0.     1.     0.     E 12006
+E 12006     23     23-0.00100              0.00100
               .
               .
               .
CBAR      12120   120     1752   746    0.     1.     0.     E 12120
+E 12120     23     23-0.00100              0.00100
$
$-------------------------------------------------------------------
$ The PSOLID 1 & 6, MAT1 1 & 6 cards define the lug and bushing materials.
$ The PBAR 120 & MAT1 99 cards define the BAR elements properties.
$
PSOLID       1       1       2       3
PSOLID       6       6       2       3
PBAR       120      99      1.      .1      .1
MAT1         1 1.000+7 3.759+6 0.33000 1.00000 0.      0.      0.
MAT1         6 2.850+7 1.120+7 0.27200 1.00000 0.      0.      0.
MAT1        99   20.+4              .3 1.00000 0.      0.      0.
ENDDATA
```

Table J.2: Commented version of the NASTRAN data deck used for the global analysis of the lug/pin joint

```
ID GC, MCG01
APP DISP
$
$-----------------------------------------------
$      Solution 66: "Non-linear Material Analysis"
$
SOL 66
TIME 180
$
$-------------------------------------------------------------------
$ THE LINES BELOW ARE NECESSARY TO CREATE AN OUTPUT FILE THAT CAN BE POST
$ PROCESSED
$
ALTER    1  $
OUTPUT2          ,,,,//C,N,-1/C,N,11/V,N,Z  $
ALTER    962  $
OUTPUT2          OUGV1,OES1//0/C,N,11/V,N,Z  $
$
$---------------------------------------------
$
CEND
TITLE    =       P I N    J O I N T
SUBTITLE = NON LINEAR ANALYSIS,         1625 NODES
$
ECHO=UNSORT
SUPER=ALL
SEALL=ALL
SPC=101
MPC=501
$
$
$    The HIGH CYCLE LOAD (i.e. LL1) is input in 3 steps.
$    5%, 25%, 100%
$
SUBCASE=1
LABEL =         5% HIGH CYCLE LOAD
LOAD = 1
NLPARM = 1
$
SUBCASE=2
```

```
LABEL =         25% HIGH CYCLE LOAD
LOAD = 2
NLPARM = 2
$
SUBCASE=3
LABEL =         HIGH CYCLE LOAD
LOAD = 3
NLPARM = 3
STRAIN=NONE
DISP = ALL
STRESS(VONM) = ALL
FORCE =         ALL
$
$  The "LOW CYCLE LOAD" is load HL1.
$
SUBCASE=4
LABEL =         LOW CYCLE LOAD
LOAD = 4
NLPARM = 4
STRAIN=NONE
DISP = ALL
STRESS(VONM) = ALL
FORCE =         ALL
$
$
$
$
BEGIN BULK
$
$-------------------------------------------------------------
$  The cylindrical coordinate system defined below
$  is the R, Theta, Z' coordinate system shown in Chapter 2.
$
CORD2C         2       0.      0.      0.      0.      1.00000 0.      CF      1
+CF     1 1.00000 1.00000 0.
$
$-----------------------------------------------------------------
$  The R unit vector of the cylindrical coordinate
$  system defined below corresponds to the Y axis of the basic
$  coordinate system defined in Chapter 2.
$  It is used to define some of the CGAP cards.
$
CORD2C         3       0.      0.      0.      0.      0.      -1.00000 CF     2
+CF     2 0.      1.00000 -1.00000
```

```
$
$-----------------------
$     5% HIGH CYCLE LOAD
$
LOAD          1      .05    16000.      201   16000.      202   12400.      412 LO    11
+LO    11   12400.   413    12400.      425   12400.      426   12400.      438 LO    12
+LO    12   12400.   439    12400.      451   12400.      452    1260.      601
NLPARM        1        1        0.     ITER        1       -20        P       NO NL     1
+NL     1           .01
$
$    25% HIGH CYCLE LOAD
$
LOAD          2      .25    16000.      201   16000.      202   12400.      412 LO    21
+LO    21   12400.   413    12400.      425   12400.      426   12400.      438 LO    22
+LO    22   12400.   439    12400.      451   12400.      452    1260.      601
NLPARM        2        1        0.     ITER        1       -20        P       NO NL     2
+NL     2          .005
$
$    100% HIGH CYCLE LOAD
$
LOAD          3     1.00    16000.      201   16000.      202   12400.      412 LO    31
+LO    31   12400.   413    12400.      425   12400.      426   12400.      438 LO    32
+LO    32   12400.   439    12400.      451   12400.      452    1260.      601
NLPARM        3        1        0.     ITER        1       -20        P       NO NL     3
+NL     3          .001
$
$
$    100% LOW CYCLE LOAD
$
LOAD          4     1.00    17400.      201   17400.      202   23900.      412 LO    41
+LO    41   23900.   413    23900.      425   23900.      426   23900.      438 LO    42
+LO    42   23900.   439    23900.      451   23900.      452    2870.      601
NLPARM        4        1        0.     ITER        1       -20        P       NO NL     4
+NL     4          .001
$
$-----------------------------------------------------------------
$ The "PLOAD4" cards below have been created by PATRAN.
$
$        TENSION "P" (See Chapter 2) are defined by PLOAD4 cards numbered
$                        201 & 202
$
PLOAD4       201     4022-0.12165-0.12165-0.12165-0.12165      320      308
PLOAD4       201     4023-0.12165-0.12165-0.12165-0.12165      322      309
PLOAD4       201     4024-0.12165-0.12165-0.12165-0.12165      361      322
```

```
PLOAD4      201    4025-0.12165-0.12165-0 12165-0.12165     365     324
PLOAD4      202    4047-0.12165-0.12165-0.12165-0.12165     318     307
PLOAD4      202    4048-0.12165-0.12165-0.12165-0.12165     316     306
PLOAD4      202    4049-0.12165-0.12165-0.12165-0.12165     358     320
PLOAD4      202    4050-0.12165-0.12165-0.12165-0.12165     354     318
PLOAD4      202    4072-0.12165-0.12165-0.12165-0.12165     369     364
PLOAD4      202    4073-0.12165-0.12165-0 12165-0 12165     371     370
PLOAD4      202    4074-0.12165-0 12165-0.12165-0 12165     365     363
PLOAD4      202    4075-0.12165-0.12165-0 12165-0.12165     367     369
PLOAD4      201    4097-0.12165-0 12165-0 12165-0.12165     363     360
PLOAD4      201    4098-0.12165-0.12165-0.12165-0.12165     359     356
PLOAD4      201    4099-0.12165-0.12165-0.12165-0 12165     361     359
PLOAD4      201    4100-0.12165-0 12165-0 12165-0.12165     358     355
$
$         BENDING "M" (See Chapter 2) are defined by PLOAD4 cards numbered
$                         412, 413, 425, 426, 438, 439, 451 & 452
$
PLOAD4      412    4022-0 09279-0.09279-0 21933-0.21933     320     308
PLOAD4      412    4023-0 09279-0 09279-0 21933-0.21933     322     309
PLOAD4      413    4024 0       0       -0 09279-0.09279     361     322
PLOAD4      413    4025 0.      0       -0 09279-0.09279     365     324
PLOAD4      425    4047-0 09279-0 09279-0 21933-0.21933     318     307
PLOAD4      425    4048-0.09279-0 09279-0.21933-0.21933     316     306
PLOAD4      426    4049 0.      0       -0.09279-0 09279     358     320
PLOAD4      426    4050 0.      0.      -0.09279-0.09279     354     318
PLOAD4      438    4072 0.09279 0.09279 0.21933 0 21933     369     364
PLOAD4      438    4073 0.09279 0 09279 0 21933 0.21933     371     370
PLOAD4      439    4074 0.      0.      0 09279 0.09279     365     363
PLOAD4      439    4075 0.      0.      0.09279 0 09279     367     369
PLOAD4      451    4097 0.09279 0.09279 0.21933 0.21933     363     360
PLOAD4      451    4098 0.09279 0.09279 0.21933 0.21933     359     356
PLOAD4      452    4099 0.      0.      0 09279 0 09279     361     359
PLOAD4      452    4100 0.      0       0.09279 0.09279     358     355
$
$
$          TORSION "T" (See Chapter 2) are defined by FORCE cards "601"
$                         They have been manually entered.
$
FORCE       601     367  .098124     0.       1.      0.
FORCE       601     354 -.098124     0.       1.      0.
FORCE       601     371  .087821     0.       1.      0.
FORCE       601     324  .087821     0.       1       0.
FORCE       601     316 -.087821     0.       1       0
FORCE       601     355 -.087821     0.       1.      0.
```

```
FORCE       601     364 -.087331      0       0.      1.
FORCE       601     307  .087331      0       0.      1
FORCE       601     363 -.023550      0.      0       1
FORCE       601     320  023550       0.      0.      1
$
$--------------------------------------------------------------------
$   The 4 cards below state that the lugs are constrained at their ends.
$   They have been created by PATRAN
$
SPC1        101     123       1       2       3       4       5       6
SPC1        101     123       7       8       9      10     153     154
SPC1        101     123     155     156     157     158     159     160
SPC1        101     123     161     162
$
$----------------------------------------------------
$   The card below states that NO node ROTATION is allowed
$   It has been manually entered.
$
SPC1        101     456       1    THRU    1625
$
$-------------------------------------------------------------------
$   The 2 cards below state that the grids along X axis are constrained
$   in theta (The node ID's were determined using the IVOL program)
$
SPC1        101       2     361     362     453    1462     454     455
SPC1        101       2     661    1453     668     671    1459
$
$------------------------------------------------------------------
$   The 6 cards below were necessary TO AVOID WASHER RIGID BODY MOTION
$   (The node ID's were determined using the IVOL program)
$
MPC         501    1143       1      1.     429       1      -1
MPC         501    1233       1      1.     679       1      -1.
MPC         501    1143       2      1      429       2      -1.
MPC         501    1233       2      1.     679       2      -1.
MPC         501    1147       2      1.     517       2      -1
MPC         501    1237       2      1.     752       2      -1.
$
$-------------------------------------------------------------------
$   The 2 cards below were necessary TO AVOID BUSHING RIGID BODY MOTION
$   (The node ID's were determined using the IVOL program)
$
MPC         501     807       2      1.     116       2      -1.
MPC         501     999       2      1.     264       2      -1.
```

```
$
$------------------------------------------------------------------------
$    The "MPC 501" cards define the WASHER & BLADE INTERFACE
$    They were created by the IMPC program.
$

MPC          501     517      3       1.     1147      3      -1.
MPC          501     520      3       1.     1145      3      -1
MPC          501     516      3       1.     1159      3      -1.
MPC          501     521      3       1.     1157      3      -1.
MPC          501     515      3       1.     1171      3      -1.
MPC          501     518      3       1      1169      3      -1.
MPC          501     429      3       1      1143      3      -1
MPC          501     430      3       1      1155      3      -1
MPC          501     427      3       1.     1167      3      -1.
MPC          501     439      3       1      1213      3      -1
MPC          501     441      3       1.     1207      3      -1
MPC          501     476      3       1      1211      3      -1

MPC          501     535      3       1.     1251      3      -1
MPC          501     619      3       1.     1239      3      -1.
MPC          501     651      3       1.     1227      3      -1.
$
$-------------------------------------------------------------------------------
$ The GRID cards below define the node positions. They were created by PATRAN
$ and modified by the TRONC program.
$
GRID           1       2  2.6399-170.020  2.3640       2
GRID           2       2  2.6399-170.020  2.0050       2
GRID           3       2  2.7563-160.612  2.3640       2
GRID           4       2  2.7563-160.612  2.0050       2
GRID           5       2  2.6000 180.000  2.3640       2
GRID           6       2  2.6000 180.000  2.0050       2
GRID           7       2  2.6399 170.020  2.3640       2
GRID           8       2  2.6399 170.020  2.0050       2
GRID           9       2  2.7563 160.612  2.3640       2
GRID          10       2  2.7563 160.612  2.0050       2
GRID          11       2  1.8755 180.000  2.0050       2
GRID          12       2  1.8755 180.000  2.3640       2

GRID        1624       2  0.9000  28.104  2.5960       2
```

```
GRID        1625       2  0.9000   0 000  2.5960        2
$
$----------------------------------------------------
$ The CHEXA, CPENTA cards below define the solid elements.
$ They were created by PATRAN.
$
CHEXA       1001       1      13      12      40      43      14      11 E       1
+E     1      39      44
CHEXA       1003       1      11      39       8       6      12      40 E       2
+E     2       7       5
CHEXA       1004       1      39      41      10       8      40      42 E       3
+E     3       9       7
CHEXA       1005       1      71      70      73      74      49      48 E       4
+E     4      51      52
CHEXA       1006       1      74      73      72      75      52      51 E       5
+E     5      50      53
CHEXA       1007       1      70      69      72      73      48      47 E       6
+E     6      50      51
CHEXA       1008       1      49      48      51      52      19      18 E       7
+E     7      21      22
CHEXA       1009       1      52      51      50      53      22      21 E       8


CPENTA      9230       9    1385    1390    1394    1372    1377    1381
CPENTA      9231       9    1399    1398    1407    1386    1385    1394
CPENTA      9232       9    1386    1385    1394    1373    1372    1381
CPENTA      9233       9    1372    1377    1381    1349    1354    1362
CPENTA      9234       9    1373    1372    1381    1350    1349    1362
CHEXA       9235       9    1377    1354    1349    1372    1342    1341 E     859
+E   859    1340    1339
CHEXA       9236       9    1372    1349    1350    1373    1339    1340 E     860
+E   860    1335    1336
CHEXA       9237       9    1342    1341    1340    1339    1316    1315 E     861
+E   861    1314    1313
CHEXA       9238       9    1339    1340    1335    1336    1313    1314 E     862
+E   862    1312    1311
CHEXA       9239       9    1316    1315    1314    1313    1290    1289 E     863
+E   863    1288    1287
CHEXA       9240       9    1313    1314    1312    1311    1287    1288 E     864
+E   864    1291    1292
$
$------------------------------------------------------
$     The CGAP cards below were created by the IGAP program.
```

```
$      They define various interfaces between the components.
$
$
$      PIN, LUG Z' INTERFACE
CGAP        10001      100       71      1565                    3
CGAP        10002      100       70      1567                    3
                                  .
                                  .
                                  .
CGAP        10070      100      278      1360                    3
CGAP        10071      100      277      1328                    3
CGAP        10072      100      271      1304                    3
$     BUSHING, LUG R INTERFACE
CGAP        12001      120      910        75                    2
CGAP        12002      120      921        74                    2
                                  .
                                  .
                                  .
CGAP        12070      120      974       302                    2
CGAP        12071      120      977       299                    2
CGAP        12072      120      978       278                    2
$     PIN, BUSHING R INTERFACE
CGAP        13001      130     1516       914                    2
CGAP        13002      130     1514       928                    2
                                  .
                                  .
                                  .
CGAP        13094      130     1413       989                    2
CGAP        13095      130     1412      1035                    2
CGAP        13096      130     1411      1064                    2
$     PIN, SLEEVE R INTERFACE
CGAP        19001      190     1462       453                    2
CGAP        19002      190     1475       448                    2
                                  .
                                  .
                                  .
CGAP        19082      190     1437       706                    2
CGAP        19083      190     1424       733                    2
CGAP        19084      190     1411       736                    2
$      WASHER, BUSHING Z' INTERFACE
CGAP        15001      150     1172       877                    3
CGAP        15002      150     1170       865                    3
                                  .
                                  .
```

```
CGAP      15070     150     1252      968                           3
CGAP      15071     150     1240      952                           3
CGAP      15072     150     1228      984                           3
$          BUSHING, LUG Z' INTERFACE
CGAP      14001     140      910       75                           3
CGAP      14002     140      866       72                           3
                              .
                              .
                              .
CGAP      14070     140      967      262                           3
CGAP      14071     140      951      260                           3
CGAP      14072     140      983      242                           3
$
$--------------------------------------------------------------
$ The PSOLID and MAT1 cards below were manually entered.
$ They define the components material properties.
$ The PGAP cards define the gap element properties.
$
PSOLID        1       1       2       3
PSOLID        4       4       2       3
PSOLID        6       6       2       3
PSOLID       71      71       2       3
PSOLID       72      72       2       3
PSOLID        8       8       2       3
PSOLID        9       9       2       3
$
PGAP        100      .0      .0    1.E7    100.
PGAP        120   -.001      .0    1.E7    100.
PGAP        130   .0008     .08    1.E7    100.
PGAP        140      .0      .0    1.E7    100.
PGAP        150      .0      .0    1.E7    100.
PGAP        190   .0008     .08    1.E7    100.
$
$--------------------------------
$   Component material properties.
$
MAT1          1 1.000+7         0.33000 1.00000 0.      0.      0.
MAT1          4 7.300+6         0.30000 1.00000 0.      0.      0.
MAT1          6 2.860+7         0.27200 1.00000 0.      0.      0.
MAT1         71 1.700+7         0.34    1.      0.      0.      0.
MAT1         72 2.850+7         0.27000 1.00000 0.      0.      0.
MAT1          8 2.850+7         0.27000 1.00000 0.      0.      0.
MAT1          9 2.900+7         0.32000 1.00000 0.      0.      0.
```

ENDDATA

Table J.3: Commented version the NASTRAN data deck used for the detailed analysis of the lug & bushing

```
ID GC, MCG01
APP DISP
$
$----------------------------------
$ SOLUTION 66: "Non-linear Material"
$
SOL 66
TIME 220
$-------------------------------------------------------------------------
$ THE LINES BELOW ARE NECESSARY TO CREATE AN OUTPUT FILE THAT CAN BE POST
$ PROCESSED
$
ALTER   1  $
OUTPUT2         ....//C,N,-1/C,N,11/V,N,Z  $
ALTER   962  $
OUTPUT2         OUGV1,OES1//0/C,N,11/V,N,Z  $
$
$------------------------------------------
$
CEND
MAXLINES=500000
TITLE   =     B U S H  &   L U G
SUBTITLE = NON LINEAR ANALYSIS,       1932 NODES
$
ECHO=UNSORT
SUPER=ALL
SEALL=ALL
MPC=500
$
$--------------------------------
$           HL2 loading conditions
$
SUBCASE       = 1
LABEL =       HIGH LOAD #2
LOAD = 1
NLPARM = 1
SPC=100
STRAIN=NONE
DISP = ALL
STRESS(VONM) = ALL
```

```
FORCE =        ALL
$
$--------------------------------
$            LL2 loading conditions
$
SUBCASE        = 2
LABEL =        LOW LOAD #2
LOAD = 2
NLPARM = 2
SPC=200
STRAIN=NONE
DISP = ALL
STRESS(VONM) = ALL
FORCE =        ALL
$
$--------------------------------
$            LL1 loading conditions
$
SUBCASE        = 3
LABEL =        LOW LOAD #1
LOAD = 3
NLPARM = 3
SPC=300
STRAIN=NONE
DISP = ALL
STRESS(VONM) = ALL
FORCE =        ALL
$
$--------------------------------
$            HL1 loading conditions
$
SUBCASE        = 4
LABEL =        HIGH LOAD #1
LOAD = 4
NLPARM = 4
SPC=400
STRAIN=NONE
DISP = ALL
STRESS(VONM) = ALL
FORCE =        ALL
$
$--------------------------------
$
BEGIN BULK
```

```
$
$-----------------------------------------------
$  The cylindrical coordinate system defined below
$  is the R, Theta, Z' coordinate system shown in Chapter 2.
$
CORD2C         2         0.      0.      0.      0.      1.00000 0.      CF    1
+CF    1 1.00000 1.00000 0.
$
$-----------------------------------------------------------
$  The X unit vector of the coordinate system defined below
$  corresponds to the Y axis shown in Chapter 2. It is used to
$  define some of the CGAP elements.
$
CORD2R         3         0.      0       0       1.00000 0.      0       CF    2
+CF    2 1.00000 1.00000 0
$
$
$------------------------------------------------------------
$      The MPC card below permits to AVOID BUSHING Z' ROTATION
MPC            500     1427      2       1.      661      2      -1.
$
$
$-------------------------------------------------
$   All FORCE cards define very small DUMMY loadings.
$   They were necessary to avoid any error message.
$
$   HIGH LOAD #2
FORCE            1      660              1.      1.
FORCE            1     1645              1.      1.
SPCADD         100      999     34       24      14       1
NLPARM           1        1     0.     ITER       1     -30       P      NO           NL     1
+NL    1              .0001
$
$   LOW LOAD #2
FORCE            2      660              1.      1.
FORCE            2     1645              1.      1.
SPCADD         200      999     33       23      13       1
NLPARM           2        1     0.     ITER       1     -20       P      NO           NL     2
+NL    2              .0001
$
$   LOW LOAD #1
FORCE            3      660              1.      1.
FORCE            3     1645              1.      1.
SPCADD         300      999     31       21      11       1
```

```
NLPARM      3     1    0.   ITER    1   -20    P    NO     NL  3
+NL   3         .0001
$
$  HIGH LOAD #1
FORCE       4    660          1.    1.
FORCE       4   1645          1.    1.
SPCADD    400    999   32    22    12    1
NLPARM      4      1   0.   ITER    1   -20    P    NO     NL  4
+NL   4         .0001
$
$----------------------------------------
$ The SPC cards below were created by PATRAN.
$
$ ID's 21, 22, 23, and 24 define the displacements imposed by the washer
$ on the bushing for the LL1, HL1, LL2, HL2 loading conditions.
$
$ ID's 11, 12, 13, and 14 define the radial displacements imposed by the pin
$ on the bushing for the LL1, HL1, LL2, HL2 loading conditions.
$
$ ID's 31, 32, 33, and 34 define the displacements imposed by the pin head
$ on the lug for the LL1, HL1, LL2, HL2 loading conditions.
$
SPC        21     45    3-2.630-3
SPC        22     45    3-3.300-3
SPC        23     45    3-8.880-4
SPC        24     45    3-6.230-4
SPC        21     51    3-2.700-3
SPC        22     51    3-3.470-3
SPC        23     51    3-7.610-4
SPC        24     51    3-4.150-4
            .
            .
            .
SPC        31    814    3-1.970-3
SPC        32    814    3-2.390-3
SPC        33    814    3 1.860-4
SPC        34    814    3 5.650-4
            .
            .
            .
SPC        11   1898    1 3.147-3
SPC        12   1898    1 5.093-3
SPC        13   1898    1 2.036-4
SPC        14   1898    1-1.085-3
```

```
SPC            11     1899          1 3.374-3
SPC            12     1899          1 5.291-3
SPC            13     1899          1 5.166-4
SPC            14     1899          1-6.619-4
$
$----------------------------------------------------
$ Lug ends are constrained.
$ The SPC1 cards below have been created by PATRAN.
$
SPC1            1      123        1       2       3       5       8      11
SPC1            1      123       17      19      25      66      68      72
SPC1            1      123      139     141     145     212     214     218
SPC1            1      123      285     287     291     362     364     368
SPC1            1      123      438     440     444
$
$-----------------------------------------------------------
$ No node rotation allowed for the types of elements used.
$ The SPC card below has been manually entered.
$
SPC1          999      456        1    THRU    1932
$
$---------------------------------------------------------------------
$ The GRID cards below define the nodes positions.
$ They have been created by PATRAN and modified by the TRONC program.
$
GRID            1        2  2.7563 160.612  2.0050       2
GRID            2        2  2.7563 160.612  2.1845       2
GRID            3        2  2.7563 160.612  2.3640       2
GRID            4        2  2.2867 156.485  2.0043       2
GRID            5        2  2.6890 165.214  2.0050       2
GRID            6        2  2.2623 162.121  2.0045       2
GRID            7        2  2.2867 156.485  2.1838       2
GRID            8        2  2.6890 165.214  2.1845       2
GRID            9        2  2.2623 162.121  2.1840       2
GRID           10        2  2.2867 156.485  2.3633       2
GRID           11        2  2.6890 165.214  2.3640       2
GRID           12        2  2.2623 162.121  2.3635       2
GRID           13        2  1.8349 150.279  2.0037       2
GRID           14        2  1.8451 157.611  2.0039       2
GRID           15        2  1.8349 150.279  2.1832       2
                .
                .
                .
GRID         1930        2  0.5333 -56.208  2.0706       2
```

```
GRID        1931       2  0.6000 -56.208  2.0706       2
GRID        1932       2  0.5667 -56.208  2.0706       2
$
$------------------------------------------------------------
$ The CHEXA, CPENTA cards below define the solid elements.
$ They were created by PATRAN.
$
CHEXA       1001       1     153     150      77      82     152     149 E       1
+E       1    76      80
CHEXA       1002       1     150     147      74      77     149     144 E       2
+E       2    71      76
CHEXA       1003       1      82      77      30      33      80      76 E       3
+E       3    29      32
CHEXA       1004       1      77      74      27      30      76      71 E       4
+E       4    22      29
CHEXA       1006       1      30      27      24      65      29      22 E       5
+E       5    16      63
CHEXA       1008       1      65      24      23      64      63      16 E       6
+E       6    15      61
CHEXA       1009       1     152     149      76      80     151     148 E       7
+E       7    75      78
CHEXA       1010       1     149     144      71      76     148     143 E       8
+E       8    70      75
               .
               .
               .
CPENTA      2125       1      63      81      32      62      79      31
CPENTA      2126       1      32      29      63      31      28      62
CPENTA      2129       1      61     137      81      60     136      79
CPENTA      2130       1      81      63      61      79      62      60
               .
               .
               .
CHEXA       6766       6    1878    1877    1799    1800    1890    1889 E    1294
+E    1294  1811    1812
CHEXA       6767       6    1877    1876    1798    1799    1889    1888 E    1295
+E    1295  1810    1811
CHEXA       6768       6    1876    1875    1797    1798    1888    1887 E    1296
+E    1296  1809    1810
$
$----------------------------------------------
$ The CGAP elements were created by the IGAP program.
$
$ BUSHING-LUG RADIAL INTERFACE
```

```
CGAP      12001     120     827     196                                   2
CGAP      12002     120     832     197                                   2
CGAP      12003     120     833     198                                   2
CGAP      12004     120     841     202                                   2
CGAP      12005     120     848     204                                   2
CGAP      12006     120     896     126                                   2
CGAP      12007     120     906     128                                   2
CGAP      12008     120     907     129                                   2
CGAP      12009     120     924     132                                   2
            .
            .
            .
CGAP      12118     120    1744     744                                   2
CGAP      12119     120    1748     745                                   2
CGAP      12120     120    1752     746                                   2
$
$
$ BUSHING-LUG AXIAL (Z') INTERFACE
CGAP      14001     140     828     192                                   3
CGAP      14002     140     836     180                                   3
CGAP      14003     140     837     166                                   3
CGAP      14004     140     898     122                                   3
CGAP      14005     140     914     110                                   3
CGAP      14006     140     915      96                                   3
CGAP      14007     140     991      53                                   3
CGAP      14008     140    1008      47                                   3
CGAP      14009     140    1009      40                                   3
            .
            .
            .
CGAP      14120     140    1436     574                                   3
CGAP      14121     140    1545     521                                   3
CGAP      14122     140    1430     516                                   3
$
$------------------------------------------------------------------------
$   The PSOLID and MAT1 cards define the component material properties.
$   The PGAP cards define the GAP properties.
$   These cards have been manually entered.
$
PSOLID         1       1       2       3
PSOLID         6       6       2       3
PGAP         120   -.001      0.    5.E7    100.
PGAP         140      .0      .0    5.E7    100.
MAT1           1 1.000+7 3.759+6 0.33000 1.00000 0.       0.       0.
```

```
MAT1            6 2.850+7 1 120+7 0 27200 1 00000 0        0        0
ENDDATA
```

Appendix K

Material Fatigue Properties

This section explains how the parameters α, β, a_1, a_2 that describe the basic material fatigue properties have been computed.

K.1 Pin Material Fatigue Properties

The pin is made of AISI4340 steel treated such that its tensile strength is $S_u = 180ksi$. No fatigue data were available for such steel. Only $S_u = 150ksi$ 4340 steel & $S_u = 208ksi$ 4340 steel data were available (Figures K.1, K.2) The average fatigue strength at $N = 1.0E5$ and $N = 4.0E4$ were taken to estimate α and β parameters for the pin steel, i e .

$$\begin{aligned} S_{a0}(N = 1.0E5) &= 0.5 \times (78 + 79) = 79ksi, \\ S_{a0}(N = 4.0E4) &= 0.5 \times (85 + 102) = 94ksi. \end{aligned} \tag{K.1}$$

$S_u = 150ksi$ 4340 steel and $S_u = 208ksi$ 4340 steels have the fatigue limits of 69 and 70ksi, the fatigue limit of $S_u = 180ksi$ 4340 steel is assumed to be 70ksi. Using these data, α and β can be computed as:

$$\alpha = 1.6120E - 4 \ ; \ \beta = 5.9962. \tag{K.2}$$

The mean stress effect parameters a_1 and a_2 of eqn(4.10) have been computed assuming that they are the same as those describing the mean stress effect of $S_u =$

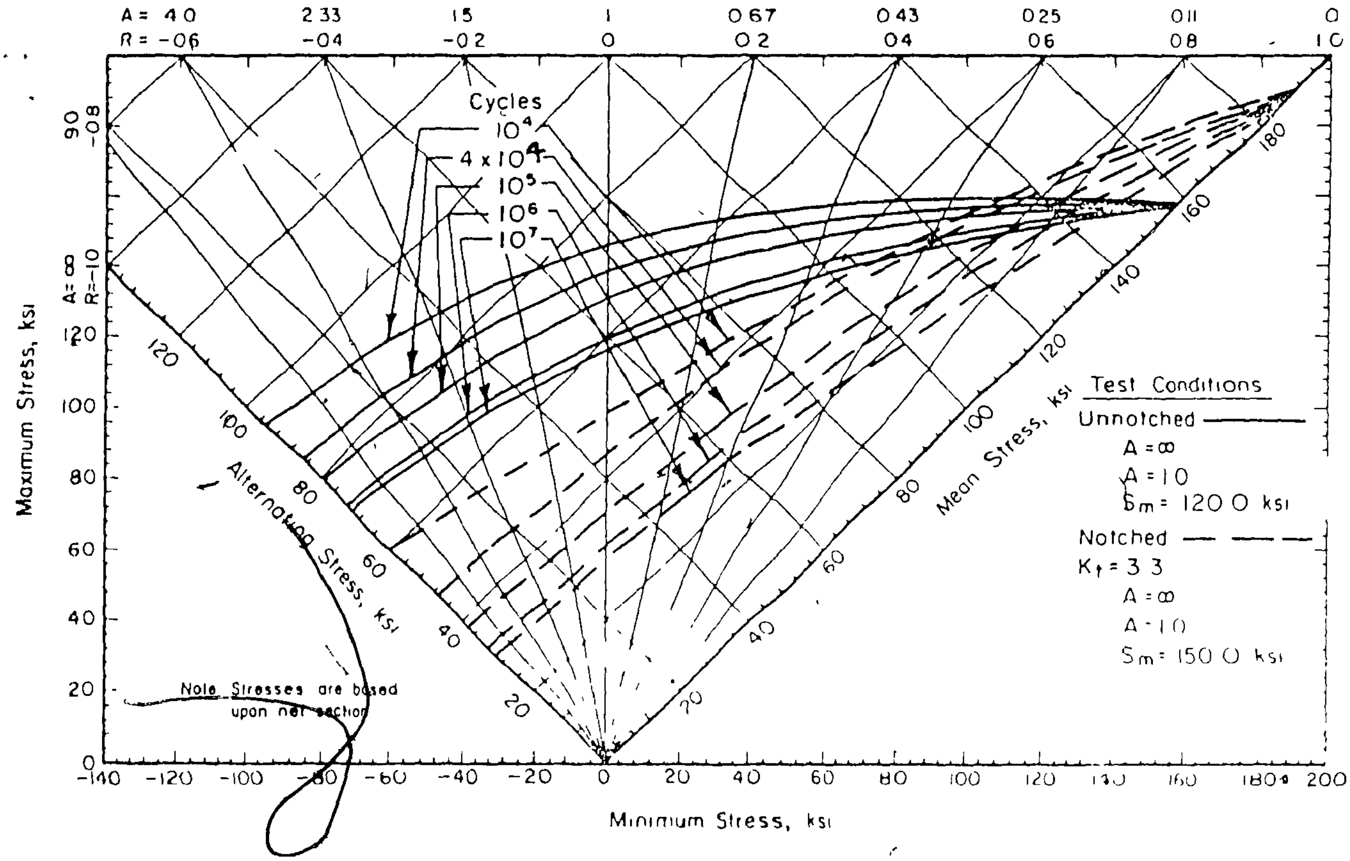


Figure K.1: Constant-Life Fatigue Diagram for heat treated AISI4340 Alloy Steel bar, $S_u = 150ksi$. (Ref MIL-HDBK-5D)

Specimen diameter: 0.4in
Assumed specimen length: 2.0in

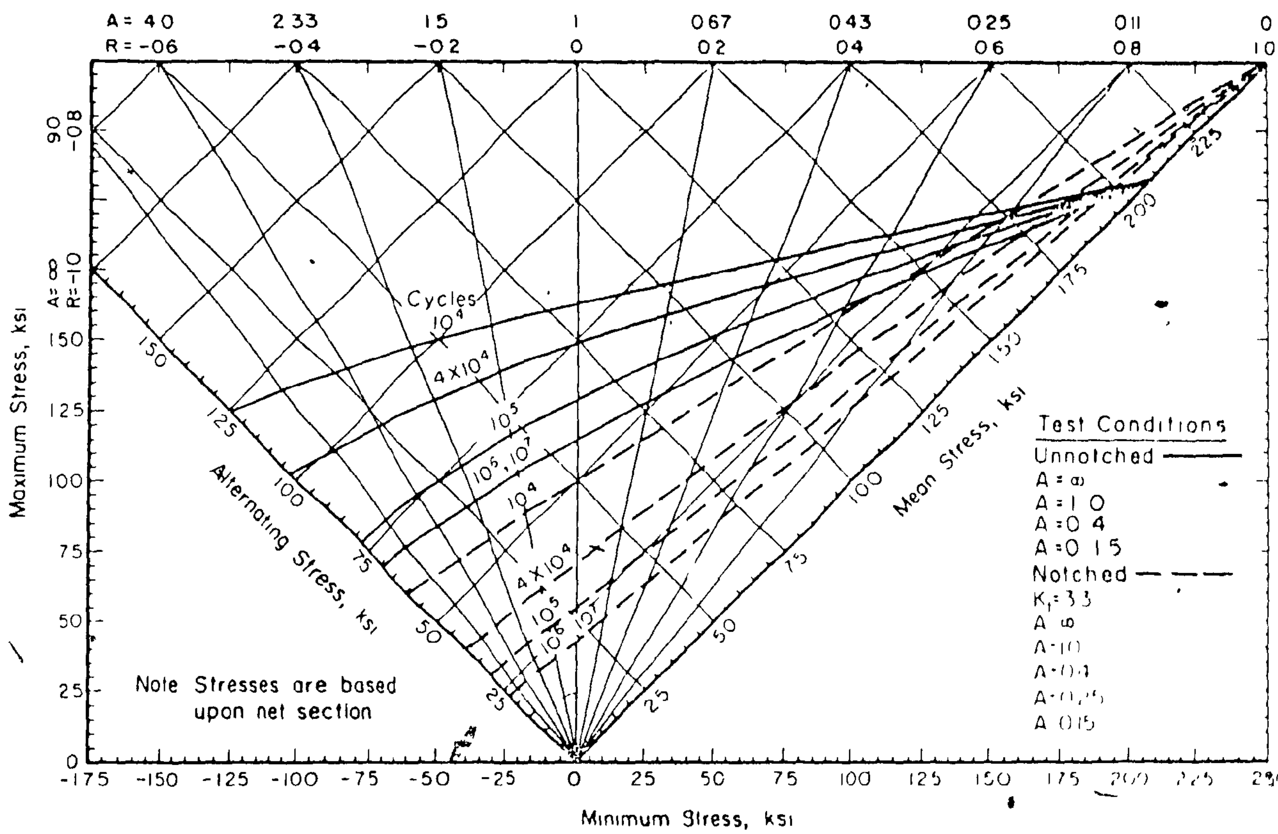


Figure K.2: Constant-Life Fatigue Diagram for heat treated AISI4340 Alloy Steel bar, $S_u = 200ksi$ (Ref MIL-HDBK-5D)

Specimen diameter: $0.4in$
Assumed specimen length: $2.0in$

208ksi 4340 steel at $N = 1.0E7$ cycles (see Figure K.3). They are:

$$a_2 = -0.5456 \;\; ; \;\; a_1 = -0.4544. \tag{K.3}$$

K.2 Lug Material Fatigue Properties

Static and fatigue properties of lug material ($7075 - T73$ aluminum) are shown in Figures K.4 and K.5. Aluminum alloys are known to show no actual fatigue limit. Since $S_f = 0$, eqn(4.2) becomes:

$$S_{a0} = S_u e^{-\alpha(\log N)^\beta}, \tag{K.5}$$

or:

$$\ln\ln\left(\frac{S_u}{S_{a0}}\right) = \ln\alpha + \beta\ln(\log N). \tag{K.6}$$

$\ln\ln\left(\frac{S_u}{S_{a0}}\right)$ is linearly dependent on $\ln(\log N)$. Using the following 3 points (taken from Figure K.4):

$$S_{a0}(N = 2.0E7) = 24ksi \;\; ; \;\; S_{a0}(N = 1.2E6) = 30ksi \;\; ; \;\; S_{a0}(N = 9.0E4) = 40ksi, \tag{K.7}$$

the least square line parameters were computed and the α and β determined as:

$$\alpha = 4.498872E-2 \;\; ; \;\; \beta = 1.619441. \tag{K.8}$$

From Figure K.5, the mean stress effect is taken into account using the Goodman line. The parameters a_2 and a_1, of eqn(4.10) are set equal to 0., −1.

K.3 Bushing Material Fatigue Properties

The bushings are made of 17-4PH stainless steel. The known data are:

$$S_u = 170ksi \;\; ; \;\; S_y = 155ksi \text{ from Table 2.1} \;\; ; \;\; S_f = 90ksi \; [30]. \tag{K.9}$$

Assuming that the fatigue strength at 1000 cycles is 90 % of S_u, i.e., $S_{a0}(N = 1.0E3) = 153ksi$, that $\log S_{a0}$ varies linearly with $\log(N)$, and that $S_{a0}(N = 1.0E6) \simeq 90ksi$, the α and β can be evaluated to give:

$$\alpha = 4.2884E-3 \;\; ; \;\; \beta = 3.6592. \tag{K.10}$$

The a_2, a_1 parameters used in eqn(4.10) are assumed to be both equal to −0.5.

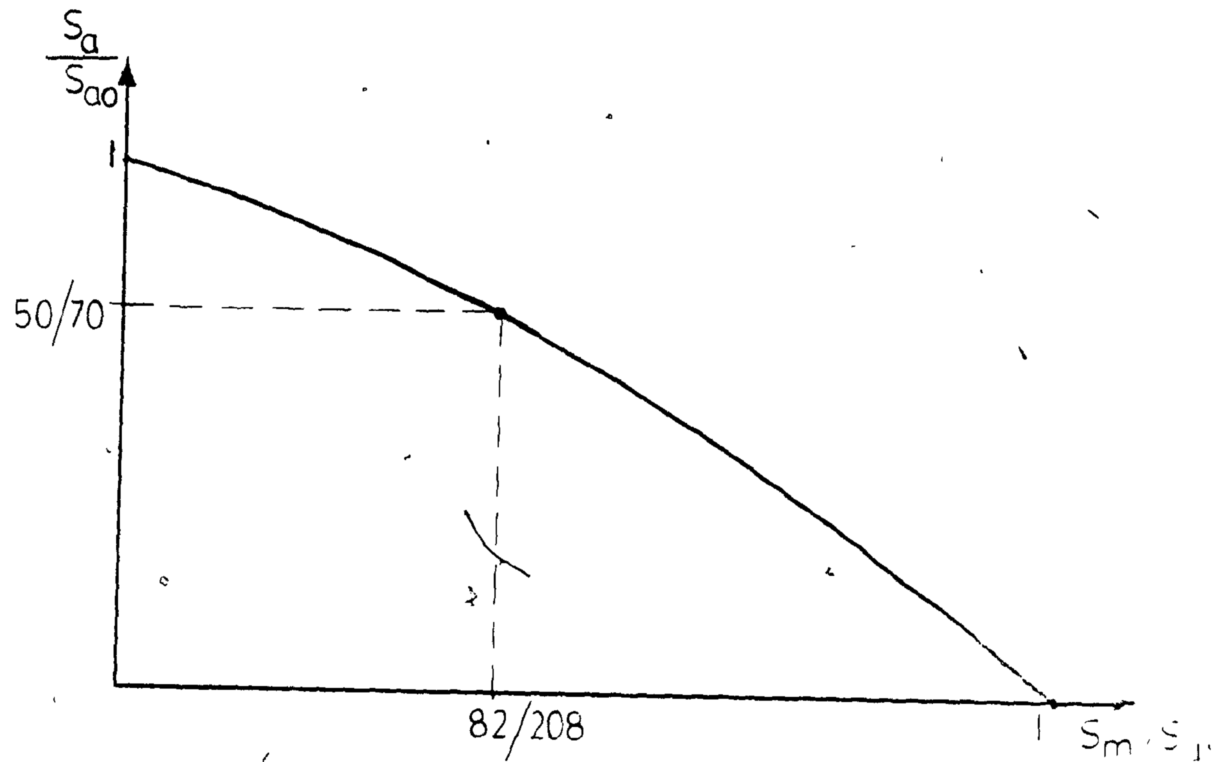


Figure K.3: $S_u = 208ksi$ 4340 Steel - Mean Stress Effect

If the curve is approximated by:

$$(\frac{S_a}{S_{a0}}) = a_2(\frac{S_m}{S_u})^2 + a_1(\frac{S_m}{S_u}) + 1 \qquad \text{(K.4)}$$

then $a_2 = -0.5456$, $a_1 = -0.4544$.

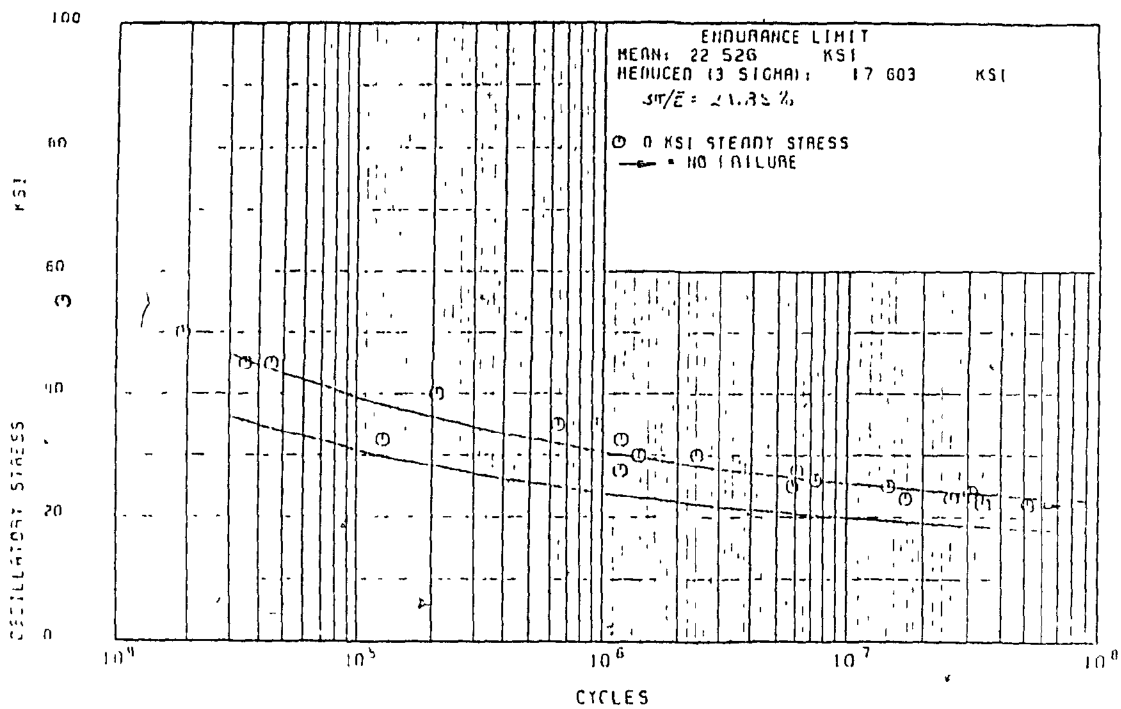


Figure K.4: $S - N$ Diagram of $7075 - T73$ Aluminum Alloy (supplied by Bell Helicopter Textron Canada)

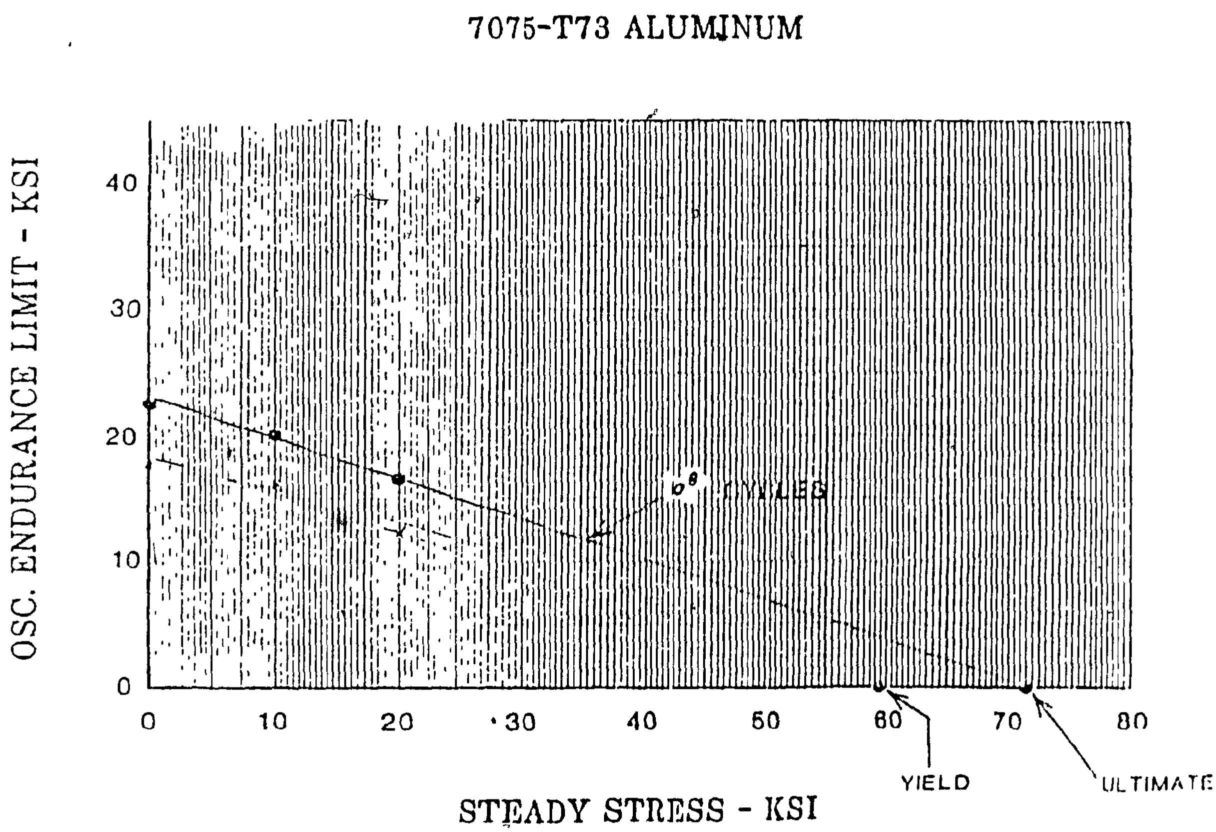


Figure K.5: Goodman diagram of 7075 − $T73$ Aluminum Alloy (supplied by Bell Helicopter Textron Canada)

K.4 Material Fatigue Properties Data Files

Tables K.1 to K.7 show the material properties data files. These files are used as input to the ENDUR program for the computation of endurance and reliability of lug/pin joint components.

The following format must be employed.

1. The first three lines are for user comments and are not read by the ENDUR program.
2. Lines 4 to 7 input the α, β, a_2, and a_1 parameters of eqns(4.2 and 4.10).
3. Line 8 gives for the Fretting Factor at $N = 1.0E7$, $S_m = 0.0$; see Section 4.1.4.
4. Line 9 gives the Size Effect Factor; see Section 4.1.4.
5. Line 10 gives the Surface roughness Factor; see Section 4.1.4.
6. Lines 11 through 13 are for the Fatigue Limit, Yield Strength and Tensile Strength, respectively.
7. Note that the first 43 characters of each line are not read.

```
*
*         MATERIAL:  LUG, 7075 ALUMINUM, FRETTING
*
*                        ALPHA PARAMETER: 4.498872E-2
*                         BETA PARAMETER: 1.61941
*                           A2 PARAMETER: O.
*                           A1 PARAMETER: -1.
*              FRETTING FACTOR AT N=1.E7: .25
*           SIZE EFFECT FACTOR AT N=1.E7: 1.
*  SURFACE FINISH EFFECT FACTOR AT N=1.E7: 1.
*                          FATIGUE LIMIT: O.
*                         YIELD STRENGTH: 59000.
*                       TENSILE STRENGTH: 72000.
*
```

Table K.1: AL7075F.DAT File

```
*
*         MATERIAL:  LUG, 7075 ALUMINUM
*
*                          ALPHA PARAMETER: 4.49887E-2
*                           BETA PARAMETER: 1.61941
*                             A2 PARAMETER: 0.
*                             A1 PARAMETER: -1.
*             FRETTING FACTOR AT N=1.E7: 1.
*          SIZE EFFECT FACTOR AT N=1.E7: 1.
*  SURFACE FINISH EFFECT FACTOR AT N=1.E7: .88
*                            FATIGUE LIMIT: 0.
*                           YIELD STRENGTH: 59000.
*                         TENSILE STRENGTH: 72000.
*
```

Table K.2: AL7075.DAT File

```
*
*         MATERIAL:  BUSHING, 17-4PH STAINLESS STEEL
*
*                          ALPHA PARAMETER: 4.2884E-3
*                           BETA PARAMETER: 3.6592
*                             A2 PARAMETER: -.5
*                             A1 PARAMETER: -.5
*             FRETTING FACTOR AT N=1.E7: 1.
*          SIZE EFFECT FACTOR AT N=1.E7: 1.
*  SURFACE FINISH EFFECT FACTOR AT N=1.E7: .88
*                            FATIGUE LIMIT: 90000.
*                           YIELD STRENGTH: 155000.
*                         TENSILE STRENGTH: 170000.
*
```

Table K.3: SS174PH.DAT File

```
*
*         MATERIAL:  BUSHING, 17-4PH STAINLESS STEEL,FRETTING
*
*                               ALPHA PARAMETER: 4.2884E-3
*                                BETA PARAMETER: 3.6592
*                                  A2 PARAMETER: -.5
*                                  A1 PARAMETER: -.5
*                       FRETTING FACTOR AT N=1.E7: .4
*                    SIZE EFFECT FACTOR AT N=1.E7: 1.
*    SURFACE FINISH EFFECT FACTOR AT N=1.E7: 1.
*                                 FATIGUE LIMIT: 90000.
*                                YIELD STRENGTH: 155000.
*                              TENSILE STRENGTH: 170000.
*
```

Table K.4: SS174PHF.DAT File

```
*
*         MATERIAL:  PIN, 4340 STEEL
*
*                               ALPHA PARAMETER: 1.6120E-4
*                                BETA PARAMETER: 5.9962
*                                  A2 PARAMETER: -.5456
*                                  A1 PARAMETER: -.4544
*                       FRETTING FACTOR AT N=1.E7: 1.
*                    SIZE EFFECT FACTOR AT N=1.E7: 1.
*           SURFACE FINISH FACTOR AT N=1.E7: .88
*                                 FATIGUE LIMIT: 70000.
*                                YIELD STRENGTH: 163000.
*                              TENSILE STRENGTH: 180000.
*
```

Table K.5: S4340.DAT File

```
*
*      MATERIAL:  PIN, 4340 STEEL, FRETTING WITH STAINLESS STEEL
*
*                     ALPHA PARAMETER: 1.6120E-4
*                      BETA PARAMETER: 5.9962
*                        A2 PARAMETER: -.5456
*                        A1 PARAMETER: -.4544
*           FRETTING FACTOR AT N=1.E7: .4
*        SIZE EFFECT FACTOR AT N=1.E7: 1.
*    SURFACE FINISH FACTOR AT N=1.E7: 1.
*                       FATIGUE LIMIT: 70000.
*                      YIELD STRENGTH: 163000.
*                    TENSILE STRENGTH: 180000.
*
```

Table K.6: S4340FS.DAT File

```
*
*      MATERIAL:  PIN, 4340 STEEL, FRETTING WITH COPPER
*
*                     ALPHA PARAMETER: 1.6120E-4
*                      BETA PARAMETER: 5.9962
*                        A2 PARAMETER: -.5456
*                        A1 PARAMETER: -.4544
*           FRETTING FACTOR AT N=1.E7: .67
*        SIZE EFFECT FACTOR AT N=1.E7: 1.
*    SURFACE FINISH FACTOR AT N=1.E7: 1.
*                       FATIGUE LIMIT: 70000.
*                      YIELD STRENGTH: 163000.
*                    TENSILE STRENGTH: 180000.
*
```

Table K.7: S4340FC.DAT File

Appendix L

ENDUR Program

The ENDUR program evaluates the high cycle fatigue life (ENDURance) and the reliability factor at $N = 1.0E7$ cycles.

A typical input file to the ENDUR program is shown in Table L.1. The input file to this program contains:

1. The PATRAN neutral file name.
2. The two extreme stress states representing the oscillating loading condition (They are either two xxxx.NOD files or two xxxx.ELS files produced by NASPAT). The program assumes that nodes are numbered from 1 to N and no number is skipped.
3. The "S_a" computation method (see the section entitled "Computation of the Equivalent Alternating & Mean Stresses" below).
4. **Sets of nodes.** They may be listed in a file ("NODE ID'S FILE:") or identified by a "range" ("NODES: FROM, TO:"). Notice that, once processed, the nodes are flagged to avoid processing them subsequently. (As an example, check the files ENDUR.DAT and HE.DAT (Tables L.1, L.3). The nodes 1,2,3,4,9,10,11,12 are processed first. When the program is asked to process nodes 1 to 32, ("NODES: FROM, TO:1,32"), the nodes 1,2,3,4,9,10,11,12 are skipped.). The first three lines of a node ID's file (such as Table L.3) are not read and may be used for comments. The program can process as many as 99 sets of nodes. The maximum number of nodes in the input files is 3000. The maximum number of elements is 2000.

   For each set of nodes:

(a) The material data file which contains the properties that permit to evaluate the alternating stress S_a as a function of the mean stress S_m and the life N ($S_a = f(N, S_m)$). The fretting effect, surface finish effect and size effect factors at $N = 1.0E7$ and $S_m = 0$ are included there. These factors are never > 1.

(b) A superimposed steady stress ($S11, S22, S33$). It may be used to take into account the shot peening effect. It must be input in the displacement coordinate system.

Follow the example (files: ENDUR.DAT, HE.DAT, AL7075F.DAT) to know how to input the data (Tables L.1, L.3, K.1).

For each node (input in a node ID's file or within a range), the program will compute the endurance and the reliability factor at $N = 1.0E7$. The output file can be used in PATRAN to get a drawing of the endurance or reliability of the component(s). The user has simply to type:

```
RUN,CON,COL,i
```

where:

- i=1 gives the equivalent alternating stress,
- i=2 gives the equivalent mean stress,
- i=3 gives the log of the expected life,
- i=4 gives the reliability factor at $1.0E7$ cycles,

and enter the ENDUR program output file name.

To run the ENDUR program, type:

```
$RUN ENDUR
 ENTER INPUT FILE NAME:          <--- written by ENDUR
 ENDUR.DAT                       <--- entered by the user, (example)
 ENTER OUTPUT FILE NAME:         <--- written by ENDUR
 ENDUR.OUT                       <--- entered by the user, (example)
```

The paragraphs below explain the main concepts used in the program.

L.1 Computation of the Equivalent Alternating & Mean Stresses

The user inputs the 2 "extreme" stress states file names. The stress states at each node are given by the following 2 extreme stress tensor components:

$$
\begin{array}{cccccc}
S_{xx1} & S_{yy1} & S_{zz1} & S_{xy1} & S_{yz1} & S_{zx1} \\
S_{xx2} & S_{yy2} & S_{zz2} & S_{xy2} & S_{yz2} & S_{zx2}
\end{array}
$$

The difference stress tensor D_{ij} is:

$$D_{xx} = S_{xx1} - S_{xx2} \quad \text{(L 1)}$$

$$D_{yy} = S_{yy1} - S_{yy2} \quad \text{(L 2)}$$

$$D_{zz} = S_{zz1} - S_{zz2} \quad \text{(L 3)}$$

$$D_{xy} = S_{xy1} - S_{xy2} \quad \text{(L 4)}$$

$$D_{yz} = S_{yz1} - S_{yz2} \quad \text{(L 5)}$$

$$D_{zx} = S_{zx1} - S_{zx2} \quad \text{(L.6)}$$

The equivalent alternating stress S_a is given by either of the following equations

1. "von Mises like equation"

$$
\begin{aligned}
S_a = {} & \frac{1}{2\sqrt{2}}[(D_{xx} - D_{yy})^2 + (D_{yy} - D_{zz})^2 + (D_{zz} - D_{xx})^2 \\
& +6(D_{xy}^2 + D_{yz}^2 + D_{zx}^2)]^{\frac{1}{2}},
\end{aligned} \quad \text{(L 7)}
$$

or

2. "Fuch's equation"

$$S_a = \frac{P_1}{2} - \frac{P_3}{3}, \quad \text{(L 8)}$$

where: P_1 and P_3 are the maximum and minimum principal values of the D_{ij} tensor.

Because there is no "universally" accepted method to compute the equivalent alternating stress, the user is given the opportunity to choose either eqn(L.7) or eqn(L.8).

The equivalent mean stress S_m is the average of the first stress tensor invariant (which measures a kind of "hydrostatic" pressure).

$$S_m = \frac{1}{2}(S_{xx1} + S_{yy1} + S_{zz1} + S_{xx2} + S_{yy2} + S_{zz2}). \quad \text{(L.9)}$$

L.2 Crack Propagation Criterion

It is known that there will be no fatigue failure at a point which always stays in compression over a cycle. Over a complete cycle, the maximum principal stresses at such a point are always $\lesssim$ zero.

L.3 Computation of the Acceptable Alternating Stress

L.3.1 Basic S-N curve

The basic S-N curve for zero mean stress is given by

$$S_{a0} = S_f + (S_u - S_f) \times \exp(-\alpha(\log N)^{\beta}) \tag{L.10}$$

where:

- S_{a0} is the acceptable alternating stress,
- S_f is the fatigue limit (input by the user in the material properties file),
- S_u is the tensile strength (input by the user in the material properties file),
- α and β are curve fitting parameters (input by the user in the material properties file),
- N is the endurance, i.e. the number of cycles at failure

In principle, this equation permits the prediction of the life for $N = 1$ to ∞.

L.3.2 Surface finish and size effects

- For $N < 1000$, surface finish and size have no effect.
- For $N > 1.0E7$, surface finish and size effects are constant. The user supplied factor (never > 1) is used.
- For $1000 < N < 1.0E7$, the factors vary linearly with $\log N$.

The factors reduce both the acceptable tensile stength to S'_u and acceptable alternating stress to S'_{a0}

L.3.3 Mean stress effect

The mean stress effect is taken into account the following way:

$$S_{am} = S'_{a0} \left(a_2 \left(\frac{S_m}{S'_u}\right)^2 + a_1 \left(\frac{S_m}{S'_u}\right) + 1. \right) \qquad \text{(L 11)}$$

where:

- S_m is the mean stress,
- a_2 , a_1 are user supplied (within material properties file) curve fitting parameters (if they cannot be evaluated, use $a_2 = -1.$, $a_1 = 0.$ for the Gerber's parabola, or $a_2 = 0.$, $a_1 = -1.$ for the Goodman's line),
- S'_u is the acceptable tensile strength, and
- S'_{a0} is the acceptable alternating stress.

L.3.4 Fretting effect

Computation of the zero mean stress fretting effect "F_0"

- For $N < 1000.$: Fretting has no effect.
- For $N > 1.0E7$: Fretting as a constant effect. The user supplied factor (never > 1) is used.
- For $1000 < N < 1.0E7$: Fretting effect varie linearly with $\log N$.

Mean stress effect on fretting. If the tangential mean stress S_{mt} is $< 0.$, it is known that the fretting is not as damaging. It is assumed that fretting has no effect if $S_{mt} < -S_y$, and the fretting strength reduction coefficient "F" varies linearly for $-S_y < S_{mt} < 0$,(i.e.,

$$\begin{aligned} if S_{mt} \geq 0. \quad & \text{then} \quad F = F_0 \\ & \text{else} \quad F = \text{MIN}(1., F_0(\frac{S_{mt}}{S_y} + 1) - \frac{S_{mt}}{S_y}). \end{aligned} \qquad \text{(L.12)}$$

L.3.5 Acceptable alternating stress S_{aa}

$$S_{aa} = F \times S_{am} \qquad \text{(L.13)}$$

L.3.6 Reliability Factor

The reliability factor is simply the ratio of the applied alternating stress over the fatigue strength, for a given mean stress and at a given endurance (here chosen as $1.0E7$). The reliability factor can be related to a reliability level (or probability of survival) if the fatigue strength distribution is known.

```
*
*                    EXAMPLE
*
*               NEUTRAL FILE NAME: PATRAN.OUT
*
*       STRESS STATE #1 FILE NAME: E1.NOD
*       STRESS STATE #2 FILE NAME: E2.NOD
*
*  SA COMPUTATION METHOD (1 OR 2): 1
*  -----* "A" GROUP BELOW *------
*                  NODE ID'S FILE: HE.DAT
*            SURFACE (1,2,3 OR 0): 1
*        MATERIAL PROPERTIES FILE: AL7075F.DAT
*       SUPERIMPOSED STRESS STATE: 0.   ,-30000. ,-30000.
*  -----* "B" GROUP BELOW *------
*                NODES: FROM, TO: 1, 32
*            SURFACE (1,2,3 OR 0): 0
*        MATERIAL PROPERTIES FILE: AL7075.DAT
*       SUPERIMPOSED STRESS STATE: 0., 0. ,0.
```

Table L.1: ENDUR.DAT file. Typical input file to ENDUR program

See Table L.2 for the file format

1. The first 3 lines are available for comments and are not read by ENDUR.
2. The 4th line gives the neutral file name.
3. The 5th and 8th lines are not read by ENDUR.
4. The 6th and 7th lines give the 2 extreme stress state file names.
5. The 9th line gives the equivalent alternating stress S_a computation method, (enter 1 or 2 which correspond respectively to eqn(L.7), eqn(L.8)).
6. The rest of the file is made of up to 99 groups of lines similar to either the "A" or "B" (Tables 4.13 and 4.15 give other examples).
   (a) The first line of the group is available for comments.
   (b) The 2nd line gives either a node ID's file name (as in the "A" group) or a list of nodes ("FROM...TO..." as in the "B" group). Once processed, each node is flagged and will not be reprocessed subsequently.
   (c) The 3rd line gives the surface on which the nodes are located. The surface is indicated after its normal unit vector in the node coordinate system, (1,2 or 3). This is used only if there is fretting. If there is no fretting, input 0.
   (d) The 4th line gives the material properties file name associated with the nodes given on the second line.
   (e) The 5th line gives a superimposed stress state given in the node coordinate system. (Here it is used to take into account the residual stress induced by shot peening). Each value MUST be separated by a comma.

Table L.2: Table L.1 Format

```
*
*   NODES 1,2,3,4,9,10,11,12: FRETTING
*
1
2
3
4
9
10
11
12
```

Table L.3: HE.DAT file

The first 3 lines are available for comments. The first number on each of the subsequent line must be an integer. It indicates a node ID. It can be followed by any comment that ENDUR program will not read.

```
     B U S H  . +   L U G
   32   32  0.000000E+00      0    31
 EXAMPLE
 HIGH LOAD =1 SUBCASE    4
10.0000000E+000.0000000E+000.0000000E+000.0000000E+000.0000000E+00
0.0000000E+000.0000000E+000.0000000E+000.0000000E+000.0000000E+00
0.9731590E+040.0000000E+000.0000000E+000.0000000E+000.0000000E+00
0.0000000E+000.0000000E+000.0000000E+000.0000000E+000.0000000E+00
-.5793461E+040.1175065E+050.1316092E+040.4313637E+040.2776492E+05
0.3791462E+040.4312430E+040.7677600E+020.7645117E+030.2257690E+04
0.9731590E+04
20.0000000E+000.0000000E+000.0000000E+000.0000000E+000.0000000E+00
0.0000000E+000.0000000E+000.0000000E+000.0000000E+000.0000000E+00
0.1152708E+050.0000000E+000.0000000E+000.0000000E+000.0000000E+00
0.0000000E+000.0000000E+000.0000000E+000.0000000E+000.0000000E+00
-.9768359E+040.1717181E+050.4859469E+040.7273805E+040.4510268E+05
0.2931219E+050.7271191E+040.8303320E+030.9290801E+030.2739287E+04
0.1152708E+05
        .
        .
        .

320.0000000E+000.0000000E+000.0000000E+000.0000000E+000.0000000E+00
0.0000000E+000.0000000E+000.0000000E+000.0000000E+000.0000000E+00
0.9873090E+040.0000000E+000.0000000E+000.0000000E+000.0000000E+00
0.0000000E+000.0000000E+000.0000000E+000.0000000E+000.0000000E+00
-.3426025E+040.9870250E+04-.7823145E+030.1190158E+040.4726641E+04
0.5249973E+040.3014644E+030.4433434E+040.1008877E+030.6162075E+03
0.9873090E+04
```

Table L.4: E1.NOD file

This file has been produced by the "NASPAT" NASTRAN to PATRAN translator and represents a stress state.

```
FATIGUE
   32   32   0.000000E+00      0      4
EXTREME STRESS STATE #1 :  E1.NOD
EXTREME STRESS STATE #2 :  E2.NOD
        10.1401622E+05-.5032420E+050.1094078E+020.3278132E+00
        20.1716331E+05-.2154772E+050.8585935E+010.6623231E+00
        30.5336852E+04-.3543446E+050.1294026E+020.1666287E+00
        40.5412157E+04-.5753621E+050.1385409E+020.1195328E+00
        50.2112029E+050.3680421E+050.4094871E+010.2275350E+01
        60.4648361E+040.1347467E+040.1221861E+020.2144510E+00
        70.6088972E+040.3108959E+040.1132813E+020.2891259E+00
        80.3978417E+040.1362010E+050.1202869E+020.2288300E+00
        90.5275813E+04-.5621006E+050.1389507E+020.1177267E+00
       100.6869380E+04-.5621375E+050.1317358E+020.1532819E+00
                .
                .
                .
       300.3900500E+04-.4157479E+040.1296315E+020.1652767E+00
       310.3163163E+040.1591883E+050.1255673E+020.1907541E+00
       320.3081373E+040.6648188E+040.1313459E+020.1554456E+00
\---1--/\-----2-----/\-----3-----/\-----4-----/\-----5-----/
```

Table L.5: ENDUR program output file when the input file is ENDUR.DAT file given in Table L.1

1. The first column gives the node ID's.
2. The 2nd and 3rd columns give the equivalent alternating & mean stresses.
3. The 4th column gives $\log N$. Infinite life can be considered when $\log N = 20$.
4. The 5th column gives the reliability factor at $N = 1.0E7$ cycles.

Table L.6: ENDUR program FORTRAN code

```
C
      PROGRAM ENDUR
C     -------------
C
C
      COMMON SU,SF,SY,ALPHA,BETA,SIZE,FINISH,FRETTING,
     &       SA,SM,A2,A1,SSX,SSY,SSZ,NODE,SMTAN,FRETSURF,METHOD
C
      CHARACTER*1 A
      CHARACTER*33 CHAR
      CHARACTER*50 INPUT,OUTPUT,MATERIAL,STATE1,STATE2,LIST,NEUTRAL
      INTEGER IDELN(2000), NTIME(3000), NODEE(3000,8), FRETSURF
      LOGICAL FLAG(3000)
      REAL ALT(3000),MEAN(3000),LOGLIFE(3000),RELY(3000)
      REAL SXX1(3000),SYY1(3000),SZZ1(3000)
      REAL SXY1(3000),SYZ1(3000),SZX1(3000)
      REAL SXX2(3000),SYY2(3000),SZZ2(3000)
      REAL SXY2(3000),SYZ2(3000),SZX2(3000)
      REAL S1(6),S2(6),SNODE1(0:3000,1:6),SNODE2(0:3000,1:6)
C
      EQUIVALENCE (SNODE1(1,1),SXX1(1))
      EQUIVALENCE (SNODE1(1,2),SYY1(1))
      EQUIVALENCE (SNODE1(1,3),SZZ1(1))
      EQUIVALENCE (SNODE1(1,4),SXY1(1))
      EQUIVALENCE (SNODE1(1,5),SYZ1(1))
      EQUIVALENCE (SNODE1(1,6),SZX1(1))
C
      EQUIVALENCE (SNODE2(1,1),SXX2(1))
      EQUIVALENCE (SNODE2(1,2),SYY2(1))
      EQUIVALENCE (SNODE2(1,3),SZZ2(1))
      EQUIVALENCE (SNODE2(1,4),SXY2(1))
      EQUIVALENCE (SNODE2(1,5),SYZ2(1))
      EQUIVALENCE (SNODE2(1,6),SZX2(1))
C
C
C     USER INPUTS
```

```
C     -----------
C
      WRITE(6,2)
 2     FORMAT(' ENTER INPUT FILE NAME  ')
      READ(5,5) INPUT
C
      WRITE(6,4)
 4     FORMAT(' ENTER OUTPUT FILE NAME: ')
      READ(5,5) OUTPUT
C
 5    FORMAT(A)
C
      OPEN ( UNIT=1,
     &       FILE=INPUT,
     &       STATUS='OLD',
     &       ACCESS='SEQUENTIAL',
     &       FORM='FORMATTED',
     &       READONLY )
C
C     READS THE NEUTRAL FILE NAME
C     --------------------------
C
      READ(1,7)NEUTRAL
  7   FORMAT(///,33X,A)
C
C     READS STRESS FILE NAMES
C     -----------------------
C
      READ(1,8) STATE1,STATE2
  8   FORMAT(/,33X,A,/,33X,A)
C
C     CHECKS IF 'ELS' OR 'NOD' FILES ARE USED
C     ---------------------------------------
C
      IF (INDEX(STATE1,'NOD').EQ.0) THEN
C
C       READING THE STRESS FROM THE STRESS AT ELEMENT CENTROIDS
C       FROM 'ELS' FILES
C       --------------------------------------------------------
```

```
C
        OPEN ( UNIT=3,
     &         FILE=STATE1,
     &         STATUS='OLD',
     &         ACCESS='SEQUENTIAL',
     &         FORM='FORMATTED',
     &         READONLY )
C
        OPEN ( UNIT=7,
     &         FILE=STATE2,
     &         STATUS='OLD',
     &         ACCESS='SEQUENTIAL',
     &         FORM='FORMATTED',
     &         READONLY )
C
        OPEN ( UNIT=9,
     &         FILE=NEUTRAL,
     &         STATUS='OLD',
     &         ACCESS='SEQUENTIAL',
     &         FORM='FORMATTED',
     &         READONLY )
C
C       SKIPS THE NODE SECTION IN THE NEUTRAL FILE
C       ------------------------------------------
C
        READ(9,150)NNODE,NELEMENT
 150    FORMAT(//,26X,2I8,/)
C
        DO I=1,NNODE
          READ(9,200)A
 200      FORMAT(//A)
        ENDDO
C
C       READS ELEMENTS & THEIR NODES FORM NEUTRAL FILE
C       ----------------------------------------------
C
        DO I=1,NELEMENT
          READ(9,300,ERR=777)IDELN(I),ITYPE
 300      FORMAT(2X,2I8,/)
```

```
         IF(ITYPE.EQ.8)THEN
           READ(9,400) (NODEE(I,J),J=1,8)
 400       FORMAT(8I8)
         ELSE
           READ(9,500) (NODEE(I,J),J=1,6)
 500       FORMAT(6I8)
         ENDIF
       ENDDO
C
 777   CONTINUE
C
C      SKIPS 4 LINES IN THE xxx.ELS FILES
C      ----------------------------------
C
       READ(3,600)A
       READ(7,600)A
 600   FORMAT(///A)
C
C
       DO I=1,NELEMENT
C
C        READS ELEMENTS AND THEIR STRESS STATES FROM xxx.ELS FILES
C        ---------------------------------------------------------
C
         READ(3,700,ERR=666)IDELE,ITYPE,(S1(J),J=1,6)
         READ(7,710,ERR=666) (S2(J),J=1,6)
 700     FORMAT(2I8,/////,6E13.7,/)
 710     FORMAT(/////,6E13.7,/)
C
           DO K=1,NELEMENT
C
C
C            FINDS NODES CORRESPONDING TO THE ELEMENT READ &
C            ADDS THE STRESS TENSOR COMPONENTS
C            ----------------------------------------------
C
             IF (IDELE.EQ.IDELN(K)) THEN
                 DO L=1,8
                   DO J=1,6
```

```
                  SNODE1(NODEE(K,L),J)=SNODE1(NODEE(K,L),J)+S1(J)
                  SNODE2(NODEE(K,L),J)=SNODE2(NODEE(K,L),J)+S2(J)
                ENDDO
                NTIME(NODEE(K,L))=NTIME(NODEE(K,L))+1
              ENDDO
            GOTO 888
          ENDIF
        ENDDO
C
 888    CONTINUE
C
      ENDDO
C
 666  CONTINUE
C
      DO N=1,NNODE
C
C       AVERAGING THE STRESSES AT THE NODES
C       -----------------------------------
        DO J=1,6
          SNODE1(N,J)=SNODE1(N,J)/NTIME(N)
          SNODE2(N,J)=SNODE2(N,J)/NTIME(N)
        ENDDO
C
      ENDDO
C
      CLOSE(3)
      CLOSE(7)
      CLOSE(9)
C
    ELSE
C
C     READS THE STRESS STATE IN "NOD" FILES
C     -------------------------------------
C
C     OPENS THE STRESS STATE #1 FILE
C     ------------------------------
C
      OPEN ( UNIT=3,
```

```
     &          FILE=STATE1,
     &          STATUS='OLD',
     &          ACCESS='SEQUENTIAL',
     &          FORM='FORMATTED',
     &          READONLY )
C
C
      READ(3,20)NNODE
 20   FORMAT(/I5//)
C
C
      DO NODE=1,NNODE
        READ(3,30)SXX1(NODE),SYY1(NODE),SZZ1(NODE),
     &            SXY1(NODE),SYZ1(NODE),SZX1(NODE)
C
 30     FORMAT(////,52X,E13.7,/,5E13.7,/)
C
        FLAG(NODE)=.FALSE.    ! INITIALIZATION
C
      ENDDO
C
      CLOSE(3)
C
C     OPENS THE STRESS STATE #2 FILE
C     ------------------------------
C
      OPEN ( UNIT=3,
     &          FILE=STATE2,
     &          STATUS='OLD',
     &          ACCESS='SEQUENTIAL',
     &          FORM='FORMATTED',
     &          READONLY )
C
C
      READ(3,50)     ! SKIPS 3 LINES
 50   FORMAT(///A)
C
      DO NODE=1,NNODE
        READ(3,30)SXX2(NODE),SYY2(NODE),SZZ2(NODE),
```

```
&                       SXY2(NODE),SYZ2(NODE),SZX2(NODE)
C
          ENDDO
C
          CLOSE(3)
C
        ENDIF
C
C       READS THE "SA" COMPUTATION METHOD
C       --------------------------------
C
        READ(1,72)METHOD
 72     FORMAT(/,33X,I3)
C
C
C       FOR EACH NODE LIST
C       ------------------
C
        DO IC=1,99
C
C         CHECKS IF NODE ID'S FILE NAME OR    "FROM    TO   "
C         -----------------------------------------------------
C
          READ(1,78,ERR=999)CHAR
 78       FORMAT(/,A)
C
C         READS THE SURFACE ON WHICH THE NODES ARE LOCATED
C         (USED FOR FRETTING COMPUTATION ONLY)
C         -----------------------------------------------
C
          READ(1,80)FRETSURF
 80       FORMAT(33X,I6)
C
C
C         READS THE MATERIAL PROPERTIES FILE NAME
C         ---------------------------------------
C
          READ(1,90)MATERIAL
 90       FORMAT(33X,A)
```

```
C
          OPEN ( UNIT=5,
     &           FILE=MATERIAL,
     &           STATUS='OLD',
     &           ACCESS='SEQUENTIAL',
     &           FORM='FORMATTED',
     &           READONLY )
C
C
C         READS THE MATERIAL PROPERTIES
C         ----------------------------
C
          READ(5,100)ALPHA,BETA,A2,A1,FRETTING,SIZE,FINISH,SF,SY,SU
 100      FORMAT(///,9(43X,F12 4,/), 43X,F12 4)
C
          CLOSE(5)
C
C         READS THE SX,SY,SZ STRESS SUPERPOSITION
C         -------------------------------------
C
          READ(1,110)SSX,SSY,SSZ
 110      FORMAT(33X,3E10 3)
C
C
C         BACKSPACING TO READ NODE ID'S FILE NAME OR FIRST & LAST NODES
C         ---------------------------------------------------------------
C
          BACKSPACE(1)
          BACKSPACE(1)
          BACKSPACE(1)
          BACKSPACE(1)
C
          IF (INDEX(STATE1,'FROM') EQ.0) THEN
C
C           READS NODE-ID'S FILE NAME
C           -----------------------
C
            READ(1,112)LIST
 112        FORMAT(33X,A,///)
```

```
C
              OPEN ( UNIT=4,
     &               FILE=LIST,
     &               STATUS='OLD',
     &               ACCESS='SEQUENTIAL',
     &               FORM='FORMATTED',
     &               READONLY )
C
C             FOR EACH NODE IN THE LIST.
C             ----------------------------
C
              READ(4,115)   ! SKIPS THE FIRST 3 LINES (WHICH ARE COMMENTS)
 115          FORMAT(//A)
C
              DO J=1,3000
C
                READ(4,*,ERR=997)NODE
C
C               COMPUTES ALTERNATING, MEAN STRESSES, LOG(N),
C               AND RELIABILITY FACTOR AT N=1 OE7
C               -------------------------------------------
C
                CALL LIFE(SXX1(NODE),SXX2(NODE),SYY1(NODE),
     &                    SYY2(NODE),SZZ1(NODE),SZZ2(NODE),
     &                    SXY1(NODE),SXY2(NODE),SYZ1(NODE),
     &                    SYZ2(NODE),SZX1(NODE),SZX2(NODE),
     &                    LOGLIFE(NODE),ALT(NODE),MEAN(NODE),RELY(NODE))
C
C
C               FLAGS THE NODE,  FLAG=.T. MEANS "PROCESSED"
C               -------------------------------------------
C
                FLAG(NODE)= TRUE.
              ENDDO
C
 997          CLOSE(4)
C
            ELSE
C
```

```
C
C         READS FIRST & LAST NODES OF THE LIST
C         ----------------------------------
C
          READ(1,91)NODEF,NODEL
 91       FORMAT(33X,2I8,///)
C
C
          DO NODE=NODEF,NODEL
            IF ( NOT.FLAG(NODE)) THEN    ! IF UNPROCESSED
C
C             COMPUTES THE ALT & MEAN STRESSES AND LOG(N)
C             ----------------------------------------
C
              CALL LIFE(SXX1(NODE),SXX2(NODE),SYY1(NODE),
     &                  SYY2(NODE),SZZ1(NODE),SZZ2(NODE),
     &                  SXY1(NODE),SXY2(NODE),SYZ1(NODE),
     &                  SYZ2(NODE),SZX1(NODE),SZX2(NODE),
     &                  LOGLIFE(NODE),ALT(NODE),MEAN(NODE),RELY(NODE))
C
C
C           FLAGS THE NODE,  FLAG=.T  MEANS "PROCESSED"
C           -----------------------------------------
C
              FLAG(NODE)=.TRUE.
C
            ENDIF
C
          ENDDO
C
        ENDIF
      ENDDO
C
 999  CONTINUE
C
C     WRITES RESULT FILE
C     -----------------
C
      OPEN ( UNIT=2,
```

```
     &        FILE=OUTPUT,
     &        STATUS='NEW',
     &        ACCESS='SEQUENTIAL',
     &        CARRIAGECONTROL='LIST',
     &        FORM='FORMATTED' )
C
      WRITE(2,195)NNODE,NNODE
 195  FORMAT(' FATIGUE',/,2I5,'   0 000000E+00     0     4')
      WRITE(2,196)STATE1
      WRITE(2,197)STATE2
 196  FORMAT(' EXTREME STRESS STATE #1   ',A)
 197  FORMAT(' EXTREME STRESS STATE #2   ',A)
C
      DO NODE=1,NNODE
        WRITE(2,199)NODE,ALT(NODE),MEAN(NODE),LOGLIFE(NODE),RELY(NODE)
 199    FORMAT(I8,4E13 7)
      ENDDO
C
      STOP
      END
C
C ====================================================================
C
C       LIFE SUBROUTINE
C       ---------------
C
C       This subroutine computes the expected life of a point
C       using the extreme stress states and the material properties
C
       SUBROUTINE LIFE(SXX1,SXX2,SYY1,SYY2,SZZ1,SZZ2,
     &                 SXY1,SXY2,SYZ1,SYZ2,SZX1,SZX2,
     &                 LOGLIFE,ALT,MEAN,RELY)
C
       COMMON SU,SF,SY,ALPHA,BETA,SIZE,FINISH,FRETTING,
     &        SA,SM,A2,A1,SSX,SSY,SSZ,NODE,SMTAN,FRETSURF,METHOD
C
       INTEGER FRETSURF
       REAL LOGLIFE,MEAN
C
```

```
C
C     COMPUTES EQUIV ALT & MEAN STRESSES
C     --------------------------------
C
      DXX = SXX1 -SXX2
      DYY = SYY1 -SYY2
      DZZ = SZZ1 -SZZ2
      DXY = SXY1 -SXY2
      DYZ = SYZ1 -SYZ2
      DZX = SZX1 -SZX2
C
C
      IF (METHOD.EQ 2) THEN
        CALL PRINCIPAL(DXX,DYY,DZZ,DXY,DYZ,DZX,P1,P2,P3)
        SA =AMAX1(P1,P2,P3)/2. - AMIN1(P1,P2,P3)/3
      ELSE
        SA =  .5 * (SQRT( ((DXX-DYY)**2+(DYY-DZZ)**2+(DZZ-DXX)**2 +
     &                        6 *(DXY**2+DYZ**2+DZX**2)) /2 ))
      ENDIF
C
C
      SM =  AMAX1(-SY, (
     &             .5 * ( SXX1 +SYY1 +SZZ1 +
     &                    SXX2 +SYY2 +SZZ2  ) +
     &                    SSX+SSY+SSZ                          )  )
C
C
C     ... ASSUMING YIELDING IS THE MINIMUM EQUIVALENT STRESS
C         (BECAUSE OF PLASTIC DEFORMATION)
C         ACTUALLY, THE MEAN STRESS EFFECT EQUATION IS NOT
C         VALID FOR ANY SM. THE LOWER LIMIT SM=-SY WAS ASSUMED.
C
C     COMPUTES TANGENTIAL MEAN STRESS (FOR FRETTING EFFECT ONLY)
C     ----------------------------------------------------------
C
      IF (FRETSURF.EQ.1) THEN
         SMTAN = .5*(SYY1 +SZZ1 + SYY2 +SZZ2) + SSY+SSZ
      ELSEIF (FRETSURF.EQ.2) THEN
         SMTAN = .5*(SXX1 +SZZ1 + SXX2 +SZZ2) + SSX+SSZ
```

```
      ELSEIF (FRETSURF.EQ.3) THEN
         SMTAN = .5*(SXX1 +SYY1 + SXX2 +SYY2) + SSX+SSY
      ELSE
         SMTAN=SM ! ACTUALLY THE NODE BEING PROCESSED IS NOT AT THE
C                    SURFACE
      ENDIF
C
C
C      STATE #1 ACTUAL NORMAL STRESSES
C      ------------------------------
C
       SRX = SXX1 +SSX
       SRY = SYY1 +SSY
       SRZ = SZZ1 +SSZ
C
C      FINDS STATE #1 PRINCIPAL STRESSES
C      --------------------------------
C
       CALL PRINCIPAL(SRX,SRY,SRZ,SXY1,SYZ1,SZX1,P11,P21,P31)
C
C      STATE #2 ACTUAL NORMAL STRESSES
C      ------------------------------
C
       SRX = SXX2 +SSX
       SRY = SYY2 +SSY
       SRZ = SZZ2 +SSZ
C
C      FINDS STATE #2 PRINCIPAL STRESSES
C      --------------------------------
C
       CALL PRINCIPAL(SRX,SRY,SRZ,SXY2,SYZ2,SZX2,P12,P22,P32)
C
C      CHECKS IF CRACK PROPAGATION IS POSSIBLE
C      -------------------------------------
C
       IF ((P11.LE.0.) .AND.
     &     (P21.LE.0.) .AND.
     &     (P31.LE.0.) .AND.
     &     (P12.LE.0.) .AND.
```

```
     &      (P22.LE.O.) .AND.
     &      (P32.LE.O.)        ) THEN
C
         XLIFE =20.    ! NO CRACK PROPAGATION, "INFINITE" LIFE
C                        (LOG(N)=20.)
C
      ELSE
C
C
C          USING SECANT ALGO, COMPUTES THE LOG(LIFE) "XLIFE"
C          FROM MATERIAL PROPERTIES, SA, SM
C          ----------------------------------------------
C
C          initialisation
C          --------------
C
         XLIFEO = 7
         XLIFE1 = 7.05
         SERRORO = SAA(XLIFEO)-SA
C
C          iterations
C          ----------
C
         DO L=1,100
           SERROR1 = SAA(XLIFE1)-SA
C
           IF ( ((XLIFEO.GT.20.).AND.(XLIFE1 GT XLIFEO))
     &       .OR. (SERROR1.EQ.SERRORO)                   ) THEN
             XLIFE1=20.
             XLIFE =20.
C            ... TO AVOID USELESS COMPUTATION IF LIFE -> INFINITE
           ELSE
             XLIFE = AMAX1(O., XLIFE1-SERROR1*(XLIFE1-XLIFEO)
     &                               /(SERROR1-SERRORO)   )
C                  ... MINIMUM LIFE = 1. CYCLE !
           ENDIF
C
C          check for convergence
C          ---------------------
```

```
C
            CONVERG = ABS(XLIFE-XLIFE1)
            IF (CONVERG.LT.0.01) GOTO 1
            XLIFE0 = XLIFE1
            XLIFE1 = XLIFE
            SERROR0 = SERROR1
C
         ENDDO
C
            WRITE(6,363)NODE
 363        FORMAT(' LIFE COMPUTATION FOR NODE ',I6,
     &             ' DID NOT CONVERGE, LIFE SET TO 1 E20')
C
         XLIFE=20.
C
         ENDIF
C
 1       CONTINUE
C
         IF (XLIFE.GE.20 ) THEN
            ALT =SA
            MEAN =SM
            LOGLIFE =20.
         ELSE
            ALT =SA
            MEAN =SM
            LOGLIFE =XLIFE
         ENDIF
C
C        COMPUTES RELIABILITY FACTOR AT N=1 E7
C        ------------------------------------
C
         RELY = SA/SAA(7 )
C
         RETURN
         END
C
C ======================================================================
C
```

```
      FUNCTION SAA(ALOGN)
C     ------------------
C
C     This function gives the acceptable alternate stress Saa
C     computed from the knowledge of log(N) & Sm
C
      COMMON SU,SF,SY,ALPHA,BETA,SIZE,FINISH,FRETTING,
     &       SA,SM,A2,A1,SSX,SSY,SSZ,NODE,SMTAN,FRETSURF,METHOD
C
      INTEGER FRETSURF
C
C       BASIC S-N CURVE FOR ZERO MEAN STRESS
C       ------------------------------------
C
        SAO = SF + (SU-SF)*EXP(-ALPHA*ALOGN**BETA)
C
C       SIZE EFFECT "SSE"
C       -----------------
C
        IF (ALOGN.LE.3.) THEN
           SSE = 1.
        ELSEIF (ALOGN.GE.7.) THEN
           SSE = SIZE
        ELSE
           SSE = .25*(SIZE-1.)*(ALOGN-3.) + 1.
        ENDIF
C
C       SURFACE FINISH EFFECT "SFE"
C       ---------------------------
C
        IF (ALOGN.LE.3.) THEN
           SFE = 1.
        ELSEIF (ALOGN.GE.7.) THEN
           SFE = FINISH
        ELSE
           SFE = .25*(FINISH-1.)*(ALOGN-3.) + 1
        ENDIF
C
C       SIZE & FINISH EFFECTS ON ACCEPTABLE TENSILE STRENGTH "SUSF"
```

```
C                                       AND ALTERNATE STRESS "SAOSF"
C       ------------------------------------------------------------
C
        SUSF  = SU  * SSE * SFE
        SAOSF = SAO * SSE * SFE
C
C       MEAN STRESS EFFECT
C       ------------------
C
        SA1 = SAOSF * ( A2*(SM/SUSF)**2 + A1*(SM/SUSF) + 1   )
C
C       FRETTING EFFECT
C       ---------------
C
        IF (ALOGN.LE.3 ) THEN
           FRETO = 1.
        ELSEIF (ALOGN.GE.7.) THEN
           FRETO = FRETTING
        ELSE
           FRETO =  25*(FRETTING-1 )*(ALOGN-3 ) + 1.
        ENDIF
C
C
C       TANGENTIAL MEAN STRESS EFFECT ON FRETTING COEF.
C       -----------------------------------------------
C
        IF (SMTAN.GE.O.) THEN
           FRET = FRETO
        ELSEIF (SMTAN.GT.-SY) THEN
           FRET = FRETO*(SMTAN/SY+1.) - SMTAN/SY
        ELSE
           FRET = 1.
        ENDIF
C
C       ACCEPTABLE ALTERNATE STRESS "SAA"
C       ---------------------------------
C
        SAA = FRET*SA1
C
```

```
      RETURN
      END
C
C ===========================================================
C
      SUBROUTINE PRINCIPAL(SRX,SRY,SRZ,SXY,SYZ,SZX,P0,P1,P2)
C     --------------------
C
C     This subroutine finds the principal stresses of a stress state
C
      COMMON /SA/SA2,SA1,SA0
C
C     COMPUTES THE STRESS INVARIANTS
C     ------------------------------
C
       SA2 = - SRX - SRY - SRZ
       SA1 = SRX*SRY + SRY*SRZ + SRZ*SRX - SXY**2 - SYZ**2 - SZX**2
       SA0 = -SRX*SRY*SRZ -2.*SXY*SYZ*SZX +
     &        SRX*SYZ**2 + SRY*SZX**2 + SRZ*SXY**2
C
C      USING SECANT ALGO, COMPUTES ONE ROOT
C      OF THE POLY X**3 +SA2*X**2 +SA1*X +SA0 =0
C      ----------------------------------------
C
C        initialisation
C        --------------
C
        X0 = -10.
        X1 =  10.
        Y0 = POLY3(X0)
C
C        iterations
C        ----------
C
        DO L=1,100
          Y1 = POLY3(X1)
          IF (Y1.EQ.Y0) THEN
            X2 = .5*(X1+X0)
          ELSE
```

```
             X2 = X1 - Y1*(X1-X0)/(Y1-Y0)
           ENDIF
C
C          check for convergence
C          --------------------
C
           CONVERG = ABS(X2-X1)
           IF (CONVERG.LT.0.01) GOTO 1
C
           X0 = X1
           X1 = X2
           Y0 = Y1
         ENDDO
C
 1    CONTINUE
C
C     FIRST ROOT
C     ----------
C
      P0 = X2
C
C     FINDS B1 AND B0 OF   (X**2 + B1*X + B0)*(X-P0) = 0.
C     --------------------------------------------------
C
      B1 = SA2 + P0
      B0 = SA1 + B1*P0
C
      C = B1**2-4.*B0
      IF ( C .LT. 0. ) THEN
        WRITE(9,101) SRX,SRY,SRZ,SXY,SYZ,SZX,SA2,SA1,SA0,P0,B1,B0,C
 101    FORMAT(' PRINC, C<0 ',6E12.4,/3E12.4,/,4E12.4)
        C=0.
      ENDIF
      SC=SQRT(C)
C
C     FINDS THE OTHER ROOTS OF
C     THE POLY X**3 +SA2*X**2 +SA1*X +SA0 =0
C     ----------------------------------------
C
```

```
      P1 =  5*(-B1-SC)
      P2 =  5*(-B1+SC)
C
      RETURN
      END
C
C ==========================================================================
C
      FUNCTION POLY3(X)
C
      COMMON /SA/SA2,SA1,SA0
C
      POLY3 = ((X+SA2)*X+SA1)*X+SA0
C
      RETURN
      END
```

Appendix M

YMIR Program

The YMIR program finds the symmetric stress state of a known stress state. The plane of symmetry is the X-Z plane.

The YMIR program input file has the format of YMIR.DAT (Table M.1). YMIR assumes that nodes numbering begins from 1 and that no node ID is skipped.

It outputs the symmetric stress state of that given in the input file. i.e. The stress state that was applied at a node located at X,Y,Z in the input file is applied at the node X,-Y,Z in the output file. (Table M.3 shows the YMIR output stress state corresponding to stress state given in M.2).

To run YMIR, type:

```
$RUN YMIR
ENTER INPUT FILE NAME:                    <--- written by YMIR
YMIR.DAT                                  <--- example, typed by the user
ENTER OUTPUT STRESS STATE FILE NAME       <--- written by YMIR
[MCGO1.GEO]YMIRS4.NOD                     <--- example, typed by the user
```

```
*
*     PIN "HL2" STRESS STATE FROM "HL1" STRESS STATE
*
*        NEUTRAL FILE NAME: [MCGO1.GEO]PATRAN.OUT
* INPUT STRESS STATE FILE: [MCGO1.GEO]S4.NOD
*
*            NODES: FROM, TO. 1287, 1625
```

Table M.1: Typical YMIR program input file

- The first 3 lines are available for comments.
- The 4th and 5th lines give the neutral file name and the input stress state that must be a xxxx.NOD file output by NASPAT (example: Table M.2).
- The 6th line is available for comment.
- All the other lines give the node ranges on which the YMIR program will be applied.

This file was used to find the HL2 stress state which is symmetric to the HL1 stress state. Nodes 1287 to 1625 belong to the pin.

Table M.2 shows the S4.NOD File.

```
     P I N   J O I N T
 1625 1625   0 000000E+00     0       31
 NON LINEAR ANALYSIS, 1625 NODES
 LOW CYCLE LOAD     SUBCASE      4
       10 0000000E+000 0000000E+000 0000000E+000 0000000E+000 0000000E+00
0 0000000E+000 0000000E+000 0000000E+000 0000000E+000 0000000E+00
0 1794370E+050 0000000E+000 0000000E+000 0000000E+000 0000000E+00
0 0000000E+000 0000000E+000 0000000E+000 0000000E+000 0000000E+00
- 1777239E+050.2962689E+050 1045863E+050 1323166E+050 2567062E+05
0 1441745E+050 1322910E+05- 3773638E+04- 1355041E+040 6441363E+04
0 1794370E+05


    12870 0000000E+000 0000000E+000 0000000E+000 0000000E+000 0000000E+00
0 0000000E+000 0000000E+000 0000000E+000 0000000E+000 0000000E+00
0 1261567E+050 0000000E+000 0000000E+000 0000000E+000 0000000E+00
0 0000000E+000 0000000E+000 0000000E+000 0000000E+000 0000000E+00
0 2505621E+030 6892414E+04- 7630242E+04- 1385809E+020 2318168E+04
0 1439700E+04- 4509555E+04- 1935040E+040 2512495E+04- 5224105E+04
0 1261567E+05


    15320 0000000E+000 0000000E+000 0000000E+000 0000000E+000 0000000E+00
0 0000000E+000 0000000E+000 0000000E+000 0000000E+000 0000000E+00
0 1109716E+050 0000000E+000 0000000E+000 0000000E+000 0000000E+00
0 0000000E+000 0000000E+000 0000000E+000 0000000E+000 0000000E+00
0 7179309E+030 5593609E+04- 7190207E+04- 5571985E+030 2128878E+04
0 3128220E+03- 4595492E+04- 1450005E+04- 1340035E+040 4871043E+04
0 1109716E+05


    16250 0000000E+000 0000000E+000 0000000E+000 0000000E+000 0000000E+00
0 0000000E+000.0000000E+000 0000000E+000 0000000E+000 0000000E+00
0 2680197E+040.0000000E+000 0000000E+000 0000000E+000 0000000E+00
0 0000000E+000.0000000E+000 0000000E+000 0000000E+000 0000000E+00
0 7489937E+030.3666296E+03- 2514852E+04- 9875800E+020 2168166E+03
- 1978722E+04- 4850757E+03- 4488635E+02- 7513348E+02- 2999270E+03
0 2680197E+04
```

Table M.2: S4.NOD file

This file has been created by the NASTRAN to PATRAN translator("NASPAT") . Nodes 1287 and 1532 are symmetrically placed.

```
      P I N  J'O I N T
 1625 1625   0.000000E+00      0      31
 NON LINEAR ANALYSIS, 1625 NODES
 Y MIRROR OF  LOW CYCLE LOAD SUBCASE  4
      10.0000000E+000.0000000E+000.0000000E+000 0000000E+000 0000000E+00
0 0000000E+000.0000000E+000 0000000E+000 0000000E+000 0000000E+00
0 0000000E+000.0000000E+000.0000000E+000.0000000E+000 0000000E+00
0 0000000E+000.0000000E+000.0000000E+000.0000000E+000 0000000E+00
0 0000000E+000 0000000E+000 0000000E+000 0000000E+000 0000000E+00
0 0000000E+000.0000000E+000 0000000E+000 0000000E+000 0000000E+00
0 0000000E+00


    12870 0000000E+000 0000000E+000 0000000E+000 0000000E+000 0000000E+00
0 0000000E+000 0000000E+000 0000000E+000 0000000E+000 0000000E+00
0 1109716E+050.0000000E+000 0000000E+000 0000000E+000 0000000E+00
0 0000000E+000.0000000E+000 0000000E+000 0000000E+000 0000000E+00
0 7179309E+030 5593609E+04- 7190207E+04- 5571985E+030 2128878E+04
0 3128220E+03- 4595492E+04- 1450005E+04- 1340035E+04- 4871043E+04
0 1109716E+05


    15320 0000000E+000 0000000E+000 0000000E+000 0000000E+000 0000000E+00
0 0000000E+000 0000000E+000 0000000E+000 0000000E+000 0000000E+00
0 1261567E+050 0000000E+000 0000000E+000 0000000E+000 0000000E+00
0 0000000E+000 0000000E+000 0000000E+000 0000000E+000 0000000E+00
0 2505621E+030.6892414E+04-.7630242E+04- 1385809E+020 2318168E+04
0 1439700E+04- 4609555E+04- 1935040E+040 2512495E+040 5224105E+04
0 1261567E+05


    16250.0000000E+000 0000000E+000 0000000E+000 0000000E+000 0000000E+00
0 0000000E+000.0000000E+000 0000000E+000 0000000E+000 0000000E+00
0 1346635E+040.0000000E+000 0000000E+000 0000000E+000 0000000E+00
0 0000000E+000.0000000E+000 0000000E+000.0000000E+000 0000000E+00
0 3771882E+030 2079775E+03- 1258356E+04- 8118622E+020 1208574E+03
- 9765344E+03- 2758879E+030 1588784E+02-.1840277E+02- 2006893E+03
0 1346635E+04
```

Table M.3: Typical YMIR program output file

It corresponds to input files shown in Table M.1 and Table M.2.

As asked in Table M.1, only nodes numbered from 1287 to 1625 have been processed. Since nodes 1532 and 1287 are symmetrically placed, the stress state of node 1532 in Table M.2 corresponds to that of node 1287 in Table M.3.

Table M.4: YMIR program FORTRAN code

```
C
      PROGRAM YMIR
C     ------------
C
C
      CHARACTER*80 LINE1,LINE2,LINE3,LINE4
      CHARACTER*50 NEUTRAL,STATE1,STATE2,INPUT
      INTEGER NODE(3000)
      REAL X(3000),Y(3000),Z(3000)
      REAL S(3000,31)
C
      EPSILON=5 E-5     ! SMALL VALUE
C
C     INPUTS
C     ------
C
      WRITE(6,1)
 1    FORMAT(' ENTER INPUT FILE   NAME   ')
      READ(5,5)    INPUT
C
      OPEN ( UNIT=4,
     &         FILE=INPUT,
     &         STATUS='OLD',
     &         ACCESS='SEQUENTIAL',
     &         FORM='FORMATTED',
     &         READONLY )
C
      READ(4,2)NEUTRAL,STATE1
 2    FORMAT(///,24X,A,/,24X,A,/)
C
C     OUTPUT
C     ------
C
      WRITE(6,3)
 3    FORMAT(' ENTER OUTPUT STRESS STATE FILE NAME  ')
      READ(5,5)    STATE2
```

```
C
  6    FORMAT(A)
C
       OPEN ( UNIT=1,
     &        FILE=NEUTRAL,
     &        STATUS='OLD',
     &        ACCESS='SEQUENTIAL',
     &        FORM='FORMATTED',
     &        READONLY )
C
C     READS NODE POSITIONS
C     --------------------
C
      READ(1,10)NNODE
 10   FORMAT(//26X,I8,/)
C
      DO I=1,NNODE
        READ(1,20)X(I),Y(I),Z(I)
 20     FORMAT(/,3E16 9,/)
      ENDDO
C
      CLOSE(1)
C
C     READS NODE LIST
C     ---------------
C
      DO N=1,99
C
        READ(4,25,ERR=888)NFROM,NTO
 25     FORMAT(24X,2I8)
C
C     FINDS THE MIRROR NODE OF EACH NODE
C     ----------------------------------
C
   DO I=NFROM,NTO
     IF (ABS(Y(I)).LE.EPSILON) THEN
       NODE(I)=I   ! IF A NODE IS ITS OWN "MIRROR"
     ELSE
       DO J=I+1,NTO
```

```
          IF ( (ABS(X(I)-X(J)) LE.EPSILON) .AND.
     &           (ABS(Y(I)+Y(J)) LE.EPSILON)  AND
     &           (ABS(Z(I)-Z(J)).LE.EPSILON)        ) THEN
          NODE(I)=J
          NODE(J)=I
          WRITE(9,*)I,J
          ENDIF
        ENDDO
      ENDIF
    ENDDO
C
      ENDDO
C
 888  CONTINUE
C
C     WRITES THE STRESS STATES OF THE MIRROR IMAGE
C     ------------------------------------------
C
      OPEN ( UNIT=2,
     &       FILE=STATE1,
     &       STATUS='OLD',
     &       ACCESS='SEQUENTIAL',
     &       FORM='FORMATTED',
     &       READONLY )
C
      OPEN ( UNIT=3,
     &       FILE=STATE2,
     &       STATUS='NEW',
     &       ACCESS='SEQUENTIAL',
     &       CARRIAGECONTROL='LIST',
     &       FORM='FORMATTED' )
C
C
C     READS & WRITES THE STRESS STATES IDENTIFICATION
C     ---------------------------------------------
C
      READ(2,30)LINE1,LINE2,LINE3,LINE4
 30   FORMAT(3(A80/),A80)
C
```

```
      WRITE(3,40)LINE1,LINE2,LINE3
 40   FORMAT(2(A80/),A80)
C
      WRITE(3,50)LINE4
 50   FORMAT(' Y MIRROR OF ',A80)
C
C     READS THE INPUT STRESS STATE
C     ---------------------------
C
      DO I=1,NNODE
C
        READ(2,55) (S(I,J),J=1,31)
 55     FORMAT(8X,5E13 7,/,
     &          5E13 7,/,5E13.7,/,5E13.7,/,5E13 7,/,5E13 7,/,E13 7)
C
      ENDDO
C
C     WRITES THE MIRROR STRESS STATE
C     -----------------------------
C
      DO I=1,NNODE
C
C       NOTE: RZ' SHEAR BECOMES ITS INVERSE
C
        S(NODE(I),30) =     -S(NODE(I),30)
        WRITE(3,60) (I,(S(NODE(I),J),J=1,31))
 60     FORMAT(I8,5E13.7,/,
     &          5E13.7,/,5E13.7,/,5E13.7,/,5E13.7,/,5E13.7,/,E13.7)
C
      ENDDO
C
 999  STOP
C
      END
```

æ

Appendix N

MINER program

The MINER program evaluates the high cycle fatigue life using the Palmgren-Miner rule. The Palmgren-Miner rule states that the life "N" of a component submitted to a variable amplitude loading is given by:

$$N = \frac{1.}{\sum \frac{X_i}{N_i}} \tag{N.1}$$

where:

- N_i is the life at loading level i,
- X_i is the frequency of occurrence of a cycle at the loading level i.

The inputs to this program are:

- the files representing the endurance at each node and at each load level (the format of the endurance files must be the same as the ENDUR program output file format, see Table L.5). Up to 99 endurance files are allowed.
- the probability or relative frequency of occurrence of each load level.

The MINER program input file format is as follows:

- the first 2 lines are for comments,
- each load level are described by 3 lines:
  - one comment line,

- one line that gives the ENDURance file name,
- one line that gives the frequency of occurrence (in %).

Follow the example (file: MINER.DAT, Table N.1).

The MINER program outputs can be illustrated using PATRAN. The user has simply to type "RUN,CON,COL,1" within PATRAN. He will then be prompted to enter a file name. The file name to be entered is the MINER output file name.

```
*
*   LUG & BUSH ENDURANCE, USING PALMGREN-MINER RULE
***** HIGH AMPLITUDE CYCLES *****
  ENDURANCE FILE NAME:[MCGO1.F]FLUG.OUT
% OCCURRENCE            :.03
***** LOW  AMPLITUDE CYCLES *****
  ENDURANCE FILE NAME:[MCGO1.F]FLLUG.OUT
% OCCURRENCE            :99.97
```

Table N.1: Typical MINER program input file

The files FLUG.OUT, FLLUG.OUT were created by the ENDUR program.

```
ENDURANCE
1932 1932   0.000000E+00     0     1
INPUT FILE: MINER.DAT
-----------
      10.1108417E+02
      20.1067445E+02
      30.1022085E+02
      40.1130773E+02
      50.1140515E+02
       .
       .
       .
    1920.6994275E+01
    1930.1269851E+02
       .
       .
       .
   19250.2000000E+02
   19260.2000000E+02
   19270.2000000E+02
   19280.2000000E+02
   19290.2000000E+02
   19300.2000000E+02
   19310.2000000E+02
   19320.2000000E+02
```

Table N.2: Typical MINER program output file

It corresponds to input file shown in Table N.1.

Table N.3: MINER program FORTRAN code

```
C
      PROGRAM MINER
C     -------------
C
      REAL LOGLIFE
      REAL TLIFE(3000)
      CHARACTER*50 INPUT,OUTPUT,NAME
C
C     INPUTS
C     ------
C
      WRITE(6,2)
 2     FORMAT(' ENTER INPUT FILE NAME: ')
      READ(5,5) INPUT
C
      WRITE(6,4)
 4     FORMAT(' ENTER OUTPUT FILE NAME: ')
      READ(5,5) OUTPUT
C
 5    FORMAT(A)
C
      OPEN ( UNIT=1,
     &        FILE=INPUT,
     &        STATUS='OLD',
     &        ACCESS='SEQUENTIAL',
     &        FORM='FORMATTED',
     &        READONLY )
C
      READ(1,8)NAME
 8    FORMAT(///,20X,A)
C
C     READS THE NUMBER OF NODES
C     ------------------------
C
      OPEN ( UNIT=3,
     &        FILE=NAME,
```

```
     &        STATUS='OLD',
     &        ACCESS='SEQUENTIAL',
     &        FORM='FORMATTED',
     &        READONLY )
C
      READ(3,30)NNODE
 30   FORMAT(/,I5)
C
      CLOSE(3)
      BACKSPACE(1)
      BACKSPACE(1)
C
C
C     INITIALIZATION
C     --------------
C
      DO K=1,NNODE
        TLIFE(K)=0.
      ENDDO
C
C     FOR EACH FILE...
C     ----------------
C
      DO J=1,99
C
        READ(1,35,ERR=888)NAME,APPL100
 35     FORMAT(/,20X,A,/,20X,F12.5)
C
        APPL = .01 * APPL100    ! % -> PROPORTION
C
        OPEN ( UNIT=3,
     &        FILE=NAME,
     &        STATUS='OLD',
     &        ACCESS='SEQUENTIAL',
     &        FORM='FORMATTED',
     &        READONLY )
C
C       SKIPS 4 LINES
C       -------------
```

```
C
        READ(3,45)
 45     FORMAT(///)
C
C       READS LOG(N) & COMPUTES CUMULATIVE LIFE FOR EACH NODE
C       -----------------------------------------------------
C
        DO K=1,NNODE
          READ(3,50)LOGLIFE
 50       FORMAT(34X,E13.7)
          XN=APPL/10.**LOGLIFE
          TLIFE(K)=TLIFE(K)+XN
        ENDDO
C
      ENDDO
C
 888  CONTINUE
C
C     WRITES THE OUTPUT FILE
C     ----------------------
C
      OPEN ( UNIT=2,
     &       FILE=OUTPUT,
     &       STATUS='NEW',
     &       ACCESS='SEQUENTIAL',
     &       CARRIAGECONTROL='LIST',
     &       FORM='FORMATTED' )
C
C     WRITES OUTPUT FILE HEADER
C     -------------------------
C
      WRITE(2,55)NNODE,NNODE,INPUT
 55   FORMAT(' ENDURANCE',/,2I5,'    0 0000000E+00      0      1',/,
     &       ' INPUT FILE· 'A,/' -----------')
C
C     WRITES THE LIFE OF EACH NODE
C     ----------------------------
C
      DO K=1,NNODE
```

```
      TOTLIFE=1./TLIFE(K)     ! FRACTURE AT DAMAGE = 1
      LOGLIFE=ALOG10(TOTLIFE)
      WRITE(2,60)K,LOGLIFE
 60   FORMAT(I8,E13.7)
     ENDDO
C
 999 STOP
C
     END
```