

https://dspace.org/

The design and implementation of an ultra high cycle fatigue testing machine

McMillan, Tristan John

2016-03-16
http://hdl.handle.net/10539/20102

Downloaded from a DSpace repository. https://dspace.org/

The Design and Implementation of an Ultra High Cycle Fatigue Testing Machine

Tristan John McMillan

This research dissertation is submitted to the Faculty of Engineering and the Built Environment, University of the Witwatersrand, in fulfilment of the requirements for the degree of Masters in Engineering.

26 October 2015, Johannesburg

Declaration

I Tristan John McMillan hereby declare that this dissertation is my own unaided work. It is being submitted for the Degree of Master of Science in Engineering at the University of the Witwatersrand, Johannesburg. This document has not been submitted before for any degree or examination to any other University.

_____

Tristan J. McMillan

26th day of October 2015

Abstract

This work serves to detail the design, implementation and validation of the Ultra High Cycle Fatigue Test rig.

The rig has been designed using a modular design methodology. The design philosophy implemented was to keep the overall design as simple as possible so as to reduce manufacturing costs while allowing for modular changes so that the rig can be modified for future conceivable test methods in the ultra-high frequency region.

Three versions of the design were created namely the Mark 1, 2 and 3. Each of the versions was an iterative improvement on the previous. The initial Mark 1 was a simple system with no rig and a simple strain measurement system. The Mark 2 consisted of the full test rig setup with both strain gauges and laser acquisition used for the measurement system. Finally the Mark 3 system was created with only the laser used for data acquisition. The final design only implemented the non-contact laser vibrometer measurement system. This was due to the fact that the operational costs were lower with the laser vibrometer, as unlike the strain gauges, it did not need to be replaced after each test. The data was also more repeatable and reliable as the system was stable during testing with no abrupt failures.

A 2D finite element method (FEM) code was developed and validated against MSc Nastran for the development of the horn and specimen geometries. The FEM code was used in the design process to accurately predict the geometry such that an axial displacement natural mode is excited at the input frequency of the system. The horn and specimen were designed to resonate at the power generation output frequency of 20 kHz.

The final design was validated using comparative data for Aluminium AL2024 with current test data available from literature. This data comparison indicated that the designed rig produces accurate and repeatable data with a good correspondence to similar test data.

Furthermore the rig was used to obtain fatigue data for aluminium material AL7075 in the region of 10^6 to 10^9 cycles. The data for AL7075 was presented against comparative data as found in the literature survey which indicated that data showed a good correspondence to standard fatigue test data for smooth specimens.

Comparisons on the effect of the surface finish on the data obtained using the ultra-high cycle fatigue testing system were conducted and it was found that, as per the literature study, this had a negligible effect on the final result.

Acknowledgements

The author gratefully acknowledges the following:

1. Prof. R.G Reid for his help and support as project supervisor.
2. S. Dinur, P. Loveday and S. Smit for their guidance and assistance during the project and use of laboratory equipment.
3. Denel Aerostructures (DAe) and the Aerospace Industry Support Initiative (AISI) for financial support.

Dedication

I dedicate this dissertation to my loving and supportive girlfriend Corina Rieger for all her support in completing this dissertation together with my beloved family, the McMillan clan.

Contents

Declaration i
Abstract ii
Acknowledgements iii
Dedication iv
Contents v
List of Figures x
List of Tables xiii
Nomenclature xiv
1 Introduction 1
1.1 Background and Motivation 1
1.2 Overall Aims and Objectives 2
2 Literature Survey 3
2.1 Typical Ultra High Cycle Testing Machines 3
2.1.1 Generator 4
2.1.2 Converter 4
2.1.3 Booster 4
2.1.4 Horn 4
2.1.5 Specimen 7
2.1.6 Control and Monitoring System 7
2.2 Applications of UHCF Testing 11
2.3 Typical Fatigue Failures 19
2.4 Material Data Obtained 20
2.4.1 Aluminium 2024-T351 Fatigue Data 20
2.4.2 Aluminium 7075-T7351 Fatigue Data 28
3 Product Requirement Specification 32
3.1 Identification of a need 33
3.2 Requirements 33
3.3 Constraints 33
3.4 Criteria 33
4 Design 34
4.1 Mark 1 - Proof of Concept 34
4.1.1 Concept 34

4.1.2 Design 34
4.1.3 Results 44
4.1.4 Discussion 44
4.2 Mark 2 – Initial Rig Design 46
4.2.1 Concept 46
4.2.2 Design 46
4.2.3 Results 62
4.2.4 Discussion 63
4.3 Mark 3 – Final Design 64
4.3.1 Concept 64
4.3.2 Design 64
4.3.3 Results 65
4.3.4 Discussion 66
5 Results 68
5.1 Test Numbering 68
5.2 Material Delivery Test Reports 68
5.3 Cost Analysis 68
5.4 Defining Possible Errors 69
5.4.1 Input Displacement and Voltage Variation 69
5.4.2 Geometric Manufacturing Errors 70
5.4.3 Young's Modulus Errors 70
5.4.4 Frequency Variation Errors 71
5.4.5 Total Error 72
5.5 AL2024 UHCF Data 73
5.6 AL7075 UHCF Data 75
6 Conclusions and Recommendations 77
7 References 78
Appendix A - Analytical Formulation 80
A.1 Section 1 – General Differential (0 <= x < L1) 81
A.2 Section 2 – General Differential (L_1 <= x < L_1 +L_2) 83
A.3 Section 3 – General Differential (L1 +L2 <= x < L1 +L2+L3) 84
A.4 Section 4 – General Differential (L_1+L_2 +L_3 <= x<L_1 +L_2+L_3+L_4) 84
A.5 Section 5 – General Differential (L_1+L_2+L_3+L_4<=x<L_1+L_2+L_3+L_4+L_5) 85
A.6 General boundary conditions 86

A.6.1 Interface sections 1 and 2 86
A.6.2 Interface sections 2 and 3 86
A.6.3 Interface sections 3 and 4 87
A.6.4 Interface sections 4 and 5 88
A.6.5 Booster Interface 89
Appendix B - Measurement Devices 94
B.1 Laser Vibrometer 94
B.2 Strain Gauges 96
B.2.1 Background 96
B.2.2 Shunt Calibration 98
Appendix C - Finite Element Formulation 100
C.1 QUAD4 100
C.1.1 Consistent Mass Matrix Q4 100
C.1.2 Lumped Mass Matrix Q4 104
C.1.3 General Implementation of Q4FI 105
C.1.4 Implementation of Q4SS 118
C.1.5 Implementation of Q4RI 120
C.1.6 Implementation of 5 Beta Hybrid element 121
C.1.7 Stress Derivation 122
C.2 Material matrices for plane strain or plane stress 127
C.2.1 2D Elements 127
C.2.2 2D Axisymmetric Elements 127
C.3 QUAD8 127
C.3.1 Changes to code 127
C.3.2 Gauss Quadrature Approximate Solution 130
C.3.3 Full integration 131
C.3.4 Further common Changes from Quad4 code 132
C.4 TRIA3 136
C.4.1 CST (Using analytical integration instead of numerical) 136
C.4.2 Linear triangle as per Kwon 142
C.5 Axis-symmetric QUAD4 element 145
C.6 Axis-symmetric TRIA3 element 152
C.6.1 CST (using analytical integration instead of numerical) 152
C.6.2 Linear Triangle using Isoparametric Method 158

C.7 FEM Verification 163
C.7.1 Four Element Model 163
C.7.2 Eight Element Model 165
Appendix D - Initial Power Supply 167
Appendix E Design Drawings 170
E.1 Full Mark 3 170
E.2 Support Assembly 172
E.3 Base Support 173
E.4 Base of Test Rig 174
E.5 Pillar 175
E.6 Top Support 176
Appendix F - Statistical Methods Implemented 180
F.1 Least squares solution 180
Appendix G - Matlab Code 183
G.1 Main code 183
G.1.1 Tapered_ODE_Solver function 189
G.1.2 getCircleConstants function 190
G.1.3 GetVoltageFromStrain function 190
G.2 AL7075 Horn and AL2024 Specimen Analytical Code 191
G.3 Calibration Code 199
G.3.1 AL2024 - Calc_TaperedHorn_Spec_GenerateCalibrationCurve_AL2024.m 199
G.3.2 AL7075 - Calc_TaperedHorn_Spec_GenerateCalibrationCurve_AL7075.m 202
G.4 Error Tolerance Code 205
G.4.1 Calculations_TaperedHorn_Specimen_loopTillEndDispMatches.m 205
G.5 FEM Formulation Functions 213
G.5.1 InsertElements.m 213
G.5.2 InsertGrids.m 214
G.5.3 InsertParameters.m 215
G.5.4 InsetPropAndMats.m 217
G.5.5 WriteBDF.m 218
G.5.6 RunNastran.m 219
G.5.7 ReadInpType.m 219
G.5.8 ReadBDFFileType.m 222
G.5.9 GetNumberOnly.m 223

G.5.10 ReadBDFLineCommaFormat.m 224
G.5.11 ReadBDFLine8CharFormat.m 225
G.5.12 ReadBDFLine16CharFormat.m 226
G.5.13 GetMeshArbShape.m (version 2.0) 227
G.5.14 GenerateMesh.m (version 2.0) 229
G.5.15 SplitStringIntoArray.m 236
G.5.16 SplitString.m 237
G.5.17 ReadF06_MandK_FileType.m 238
G.5.18 ReadBDFandRunModalAnalysis.m 241
G.5.19 GetModalAnalysis.m 243
Appendix H Residual Load Data 244
H.1 AL2024 244
H.1.1 S2-5 and S2-6 244
H.1.2 S2-7 to S2-10 245
H.1.3 S2-11 to S2-15 246
H.1.4 S2-16 to S2-25 247
H.2 AL7075 248
H.2.1 S1-1, S1-8 to S1-11 248
H.2.2 S1-5 249
H.2.3 S1-6 250
H.2.4 S1-12 to S1-15 251
H.2.5 S1-16 to S1-25 252
Appendix I Material Delivery Test Reports 253
Appendix J Cost Analysis 255

List of Figures

Figure 1-1: Fatigue lives of modern machinery as per American Standard Test Methods (ASTM) [1] .. 1
Figure 1-2: Possible material fatigue properties in giga-cycle region [1] 2
Figure 2-1: Typical ultrasonic fatigue testing system [2] 3
Figure 2-2: Variation of stress and displacement in a typical stack [4] 8
Figure 2-3: Constant cyclic loading [5] 8
Figure 2-4: Three point Rainflow counting [5] 9
Figure 2-5: Image of propeller size on A400M and blade number 11
Figure 2-6: Typical pressure plot of one harmonic of the blade pass frequency 12
Figure 2-7: Typical SN curves for aluminium [7] 13
Figure 2-8: Haibach curve compared to Stanzl's UHCF data 14
Figure 2-9: Crack initiation in surface inclusion and crack initiation in internal inclusion [9] 15
Figure 2-10: S-N curve of aluminium alloy 2024/T3; R=0.1 [1] 16
Figure 2-11: Endurance data T6 (a), T66 (b) and T64(c) at 20 kHz (dots) and 100Hz (circles) [16] 17
Figure 2-12: Endurance data of (a) commercially pure Niobium annealed and (b) commercially pure Niobium cold worked, (c) commercially pure Tantalum annealed and commercially pure Tantalum cold worked [15] 18
Figure 2-13: Endurance data for commercial pure anneal Titanium and Titanium [15] 19
Figure 2-14: Typical fatigue failure mechanism [18] 19
Figure 2-15: Typical failure method achieved during standard fatigue testing [18] 20
Figure 2-16: Fatigue data for aluminium 2024-T351, Stanzl [12] 21
Figure 2-17: Typical aluminium 2024 specimen, Chen [14] 21
Figure 2-18: Fatigue data for aluminium 2024, Chen [14] 22
Figure 2-19: Data for AL2024 [19] 22
Figure 2-20: AL2024-T4 as per MMPDS [20] 23
Figure 2-21: AL2024-T3 as per MMPDS [20] 23
Figure 2-22: Specimen size for AL2024 data, Mayer [21] 24
Figure 2-23 AL2024 data for load ratio of -1, Mayer [21] 24
Figure 2-24: AL2024 combine data 25
Figure 2-25: Dumbbell formula geometry definition [14] 26
Figure 2-26: AL2024 cleavage fracture [14] 26
Figure 2-27: Geometry definition for specimen as defined by Bathias [4] 27
Figure 2-28: S/N curves for un-notched 7075-T6 aluminium alloy [20] 28
Figure 2-29: AL7075-T73 fatigue data [22] 29
Figure 2-30: Aluminium 7075-UA typical specimen [23] 29
Figure 2-31: Data for AL7075 [24] 30
Figure 2-32: Papkriacou data for AL7075 (AlZnMgCu1.5-T6) [16] 30
Figure 2-33: AL7075 combined data 31
Figure 3-1: PRS for Design 32
Figure 4-1: Mark 1 proof of concept horn with two gauges indicated 35
Figure 4-2: Branson A-200A horn analyser 35
Figure 4-3: Mark 1 specimen sizing 36
Figure 4-4: Axisymmetric Tria3 element model 36

Figure 4-5: Tet10 element 3D model 37
Figure 4-6: Mode shape for axisymmetric model 37
Figure 4-7: Mode shape and element size for 2mm square axisymmetric element 38
Figure 4-8: Mark 1 horn and specimen combination axial mode 40
Figure 4-9: Typical horn natural frequency before gauges are applied 42
Figure 4-10: Typical horn natural frequency after gauges are applied 42
Figure 4-11: Analytical solution for proof of concept test 43
Figure 4-12: Proof of concept comparison to known data 44
Figure 4-13: Test rig construction 46
Figure 4-14: Test rig combined with the actuating system and power source 47
Figure 4-15: Axial resonance mode of the AL7075 horn design 48
Figure 4-16: 3D FEM Tet10 horn model 49
Figure 4-17: Nearest non-axial modes 49
Figure 4-18: Amplification affects for one degree of freedom system 50
Figure 4-19: Analytical displacement and stress variation of horn length 51
Figure 4-20: Comparison to power generator noise output 51
Figure 4-21: Comparison of strain gauges to laser 52
Figure 4-22: Laser vibrometer data for Al7075 horn and Al2024 test specimen 4 53
Figure 4-23: 2D Specimen model and applicable mode in 10 kHz to 30 KHz range 54
Figure 4-24: 3D Nastran FEM model 54
Figure 4-25: Typical displacement, strain and stress output 56
Figure 4-26: Laser vibrometer transfer function of AL2024 56
Figure 4-27: Typical displacement, strain and stress output 57
Figure 4-28: Strain gauge location on horn 57
Figure 4-29: Strain gauge transfer function of AL2024 58
Figure 4-30: Position where vibration of Rig was investigated 59
Figure 4-31: Booster coupling displacement 60
Figure 4-32: Top support displacement 60
Figure 4-33: Right pillar displacement 61
Figure 4-34: first bending mode of top support 61
Figure 4-35: Mark 2 AL2024-T351 62
Figure 4-36: Mark 2 AL7075-T7351 62
Figure 4-37: AL2024 specimen micrographic image of surface crack 63
Figure 4-38: Mark 3 final setup 64
Figure 4-39: Mark 3 AL2024-T351 65
Figure 4-40: Mark 3 AL7075-T7351 65
Figure 4-41: AL2024 comparison of surface finish 66
Figure 4-42: AL7075 variation due to surface finish 67
Figure 5-1: Comparison of AL2024-T351 Data 73
Figure 5-2: Micrographic image of AL2024 Specimen H1-S2-13 74
Figure 5-3: Comparison of AL7075-T7351 Data 75
Figure 5-4: Typical failure of AL7075 76
Figure 5-5: Failed specimen showing tension failure (H1-S1-6) 76

Figure A-1: General sonic fatigue specimen 80
Figure A-2: Profile plot of specimen using Matlab 81
Figure A-3: Plot for 5 section specimen model fixed geometry 92
Figure A-4: Plot for 5 section specimen model variable geometry 93
Figure B-1 : Laser vibrometer internal setup [25] 94
Figure B-2: Typical metallic bonded strain gauge [26] [27] 96
Figure B-3: Full bridge configuration [27] 96
Figure B-4: Tension configuration [5] 97
Figure B-5: Use of dummy gauge to eliminate temperature affects [26] 97
Figure B-6: Shunt resistor across R3 [27] 98
Figure B-7: HBM shunt calibration unit 99
Figure C-1: Four node quadrilateral (Q4) [28] 100
Figure C-2: Four node Isoperimetric quadrilateral (Q4) [28] 100
Figure C-3: Bilinear quadrilateral Q4 [28] 105
Figure C-4: four node element with boundary tractions 123
Figure C-5: Boundary displacement at a location on side 12 124
Figure C-6: Cyclic indices for [G] matrix 125
Figure C-7: Third order Gauss Quadrature node locations [28] 129
Figure C-8: General node number of quad8 element 132
Figure C-9: Linear triangle 136
Figure C-10: Area of parallelogram is the magnitude of the cross product 138
Figure C-11: Three node triangular element 142
Figure C-12: Axis definition for an element 147
Figure C-13: linear triangle element nodal co-ordinates and displacement [30] 153
Figure C-14: Numerical integration factors for triangle elements 162
Figure C-15: Four element verification model 163
Figure C-16: Eight element verification model 165
Figure D-1: ATX power supply for strain gauge measurement system 167
Figure D-2: Strain gauge signal conditioning system 168
Figure D-3: Two input signals 168
Figure D-4: Combined input signal 168
Figure D-5: Test of Circuitry 169
Figure E-1: Schematic exploded view of full Mark 3 system 171
Figure I-1: Certified Test report from Kaiser Aluminium for AL7075-T735 253
Figure I-2: Certified Test report from Kaiser Aluminium for AL2024-T351 254

List of Tables

Table 2-1: Strain gauge and laser vibrometer comparison 10
Table 2-2: Summary of Al2024 specimen Geometry 27
Table 4-1: Tet10 frequencies (10 kHz to 35 kHz) 38
Table 4-2: Lumped mass frequency comparison for one TRIA3 element model 39
Table 4-3: Coupled mass frequency comparison for one TRIA3 element model 40
Table 4-4: Frequency Comparison 41
Table 5-1: Stress variation due to input amplitude errors 69
Table 5-2: Stress variation due to geometric tolerance errors 70
Table 5-3: Young's modulus variation from residual plots 70
Table 5-4: Stress variation due to Young's modulus variation 71
Table 5-5: Stress variation due to frequency variation 71
Table 5-6: Chemical composition comparison of AL2024 74
Table 5-7: Chemical composition comparison of Al7075 75

Table C-1: Comparison of four element model using lumped mass matrix 164
Table C-2: Comparison of four element model using coupled mass matrix 164
Table C-3: Comparison of eight element model using lumped mass matrix 166
Table C-4: Comparison of eight element model using coupled mass matrix 166
Table J-1: Equipment cost estimate 255

Nomenclature

UHCF	- Ultra High Cycle Fatigue (Giga-cycle fatigue from 10^6 to 10^{12} cycles)
UHC	- Ultra High Cycle
Quad4	- Quadrilateral with 4 nodes
Quad8	- Quadrilateral with 8 nodes
Tria3	- Triangular element with 3 nodes
Tria6	- Triangular element with 6 nodes
HCF	- High Cycle Fatigue (megacycle fatigue from 10^3 to 10^6 cycles)
LCF	- Low Cycle Fatigue (Generally from 1 to 10^3 cycles)
SN	- Stress versus number of cycles to failure
εN	- Strain versus number of cycles to failure
WFF	- Wing to Fuselage Fairing
DAQ	- Data Acquisition
GF	- Gauge Factor
AC	- Alternating Current
DC	- Direct Current
RF	- Reserve factor
DUFTS	- Denel Ultrasonic Fatigue Testing System
CSIR	- Council for Scientific and Industrial Research
DAe	- Denel Aerostructures
DAv	- Denel Aviation
DAS	- Data Acquisition System
ALCOA	- Aluminium Company of America
MMPDS	- Metallic Material Properties Development and Standardisation (previously known as Military Handbook 5)
PRS	- Product Requirement Specification
CSIR	- Council for Scientific and Industrial Research

1 INTRODUCTION

In material science, fatigue is the weakening of a material from repeatedly applied loads (stresses) on an object. This is the case for any material subjected to a loading which varies continuously in time. Generally the loads applied during fatigue testing are much lower than the materials allowable yield strength. During fatigue testing a crack usually occurs. When the crack has grown sufficiently through the test piece the applied load will cause the material to fail statically due to the reduced material section through which the load must be transferred. Monitoring the rate at which the crack grows falls under the concept of damage tolerance whereas fatigue is the time it takes for the crack to initiate. This work will focus on fatigue of material test coupons.

1.1 Background and Motivation

In traditional fatigue the concept of the endurance limit is well documented, for cycles exceeding 10^6. The endurance limit is a stress limit line whereby if the component is designed with a maximum stress level below this line then the component will have infinite fatigue life. This is a valid assumption only for some materials where the life will exceed 10^6 cycles. However, in this age where the cycles of new machinery are likely to exceed 10^6 cycles, this region of the SN (stress vs. cycles to failure) curve needs to be better represented and understood [1]. Figure 1-1 illustrates this concept of standard fatigue limits for components exceeding 10^6 cycles as found in cars (10^8 cycles), gas turbines (10^{10} cycles), and trains (10^9 cycles).

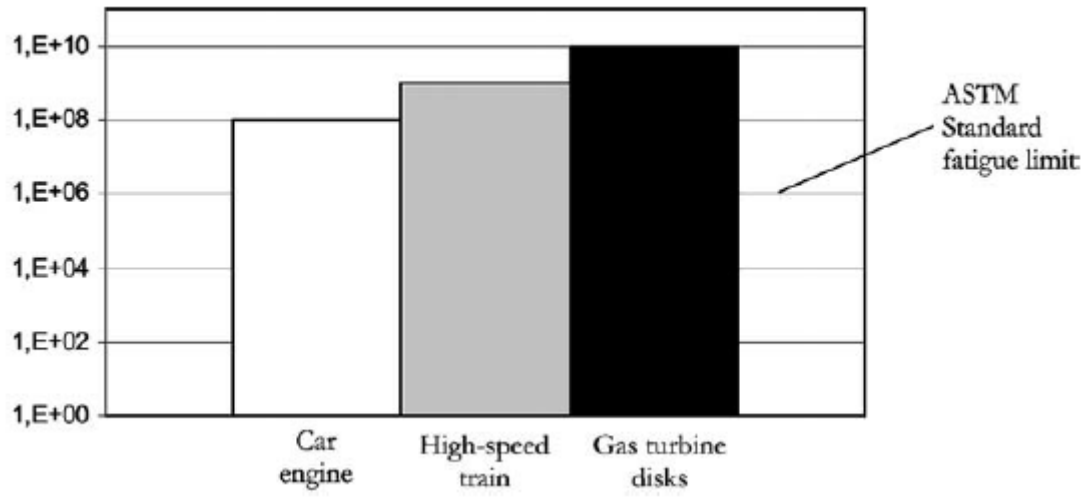


Figure 1-1: Fatigue lives of modern machinery as per American Standard Test Methods (ASTM) [1]

Some materials may have infinite lives where fatigue life exceeds 10^6 cycles. This is not feasible for all materials, therefore it is necessary that the gigacycle (cycles exceeding 10^9) region for each material is analysed to determine what fatigue relationship is applicable. Figure 1-2 indicates just some of the possibilities a material SN curve may have where the steeper gradients are indicative of lower fatigue tolerant materials, while the ideal fatigue material solution would be an infinite fatigue life.

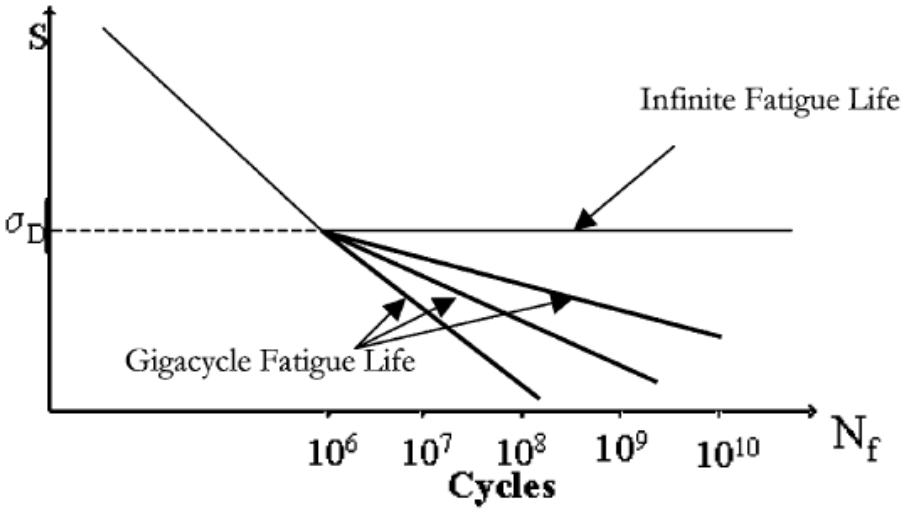


Figure 1-2: Possible material fatigue properties in giga-cycle region [1]

Tests on conventional powered mechanically or hydraulically fatigue testing machinery, require long run time periods to generate results in the high cycle, HCF, (cycles exceeding 10^3) to ultra-high cycle (cycles exceeding 10^7) fatigue regime. Typical servo-hydraulic systems operating at 100 Hz will take approximately 28 hours to reach 10^7 cycles or approximately 115 days to reach 10^9 cycles [2] for one data point. This is concerning considering that typically the standard ASTM specification requirement for a research fatigue curve requires a minimum of 12 data points [3]. Consequently the total time for uninterrupted testing would approach four years for a full data curve valid to 10^9 cycles without extrapolation which is consequently impractical.

Hence it is necessary to have another testing method to produce results in a more suitable time frame. The use of an ultrasonic fatigue testing shows advanced benefits as the time taken for each specimen test to complete is vastly reduced as compared to conventional fatigue testing. Ultra-sonic fatigue testing machines operate at a frequency range of 20 kHz to 30 kHz and can produce results at 10^7 cycles in less than ten minutes and results at 10^9 cycles in approximately 14 hours per data point. This will reduce testing time and lower operating costs, which is more economical. To develop an UHCF (Ultra High Cycle Fatigue) testing machine is beneficial for future fatigue testing and material property investigation in South Africa.

1.2 Overall Aims and Objectives

The main aim is to create a fatigue testing machine capable of producing accurate repeatable SN fatigue data, in the region where the cycles exceed 10^6 in an appropriate time period.

2 LITERATURE SURVEY

2.1 Typical Ultra High Cycle Testing Machines

The first ultrasonic fatigue testing machine was constructed by Mason in 1950. Since then, several laboratories have been setup to specialise in ultrasonic fatigue testing; these include those of Willertz in the US, Stanzl in Austria, Bathias in France, Ishii in Japan and Puskar in Slovakia. However no UHCF testing machine has currently been setup in the southern hemisphere, let alone in Africa.

The basic system that will need to be constructed is indicated in Figure 2-1.

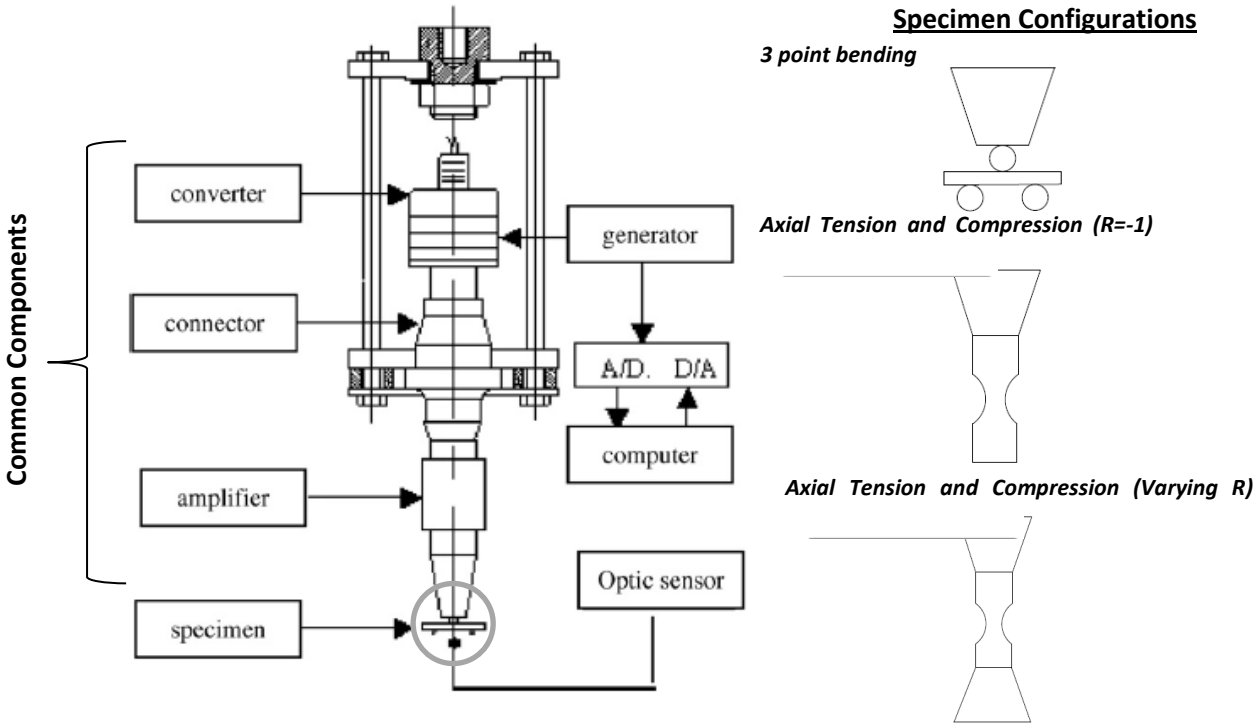


Figure 2-1: Typical ultrasonic fatigue testing system [2]

The system indicated in Figure 2-1 has a typical 3-point bending configuration (shown encircled); however the underlying system is common for most UHCF test machines. The main change between the different testing systems is the type of specimen and it's method of contact or connection with the rest of the system. Some typical contact and connection test methods that are possible are indicated in the right of Figure 2-1 such as the 3-point bending as shown, axial tension with a R ratio of -1 and the axial tension/compression assembly for varying R ratios, these can be implemented simply by changing the encircled region.

The main components required for an ultrasonic fatigue testing machine are:

2.1.1. Generator
2.1.2. Converter
2.1.3. Amplifier (horn)
2.1.4. Connector (booster)
2.1.5. Specimen
2.1.6. Control/Monitoring System

The function of each of these components is described briefly in the following sections to give background as to what is required of the Denel Ultrasonic Fatigue Testing System, DUFTS.

2.1.1 Generator

The **generator** is required to convert the voltage from the main power supply which in South Africa is typically 240 Volts at a frequency of 50 Hz, into a sinusoidal 20 kHz signal that can be interpreted by the piezoelectric converter [2].

2.1.2 Converter

The **converter** is a piezoelectric transducer that is excited via the sinusoidal signal received from the generator and consequently produces longitudinal displacements corresponding to the input signal [2]. These longitudinal vibration displacements then induce longitudinal ultrasonic waves in the horn and specimen. The displacements are generated by a cylindrical stack of ceramic piezoelectric transducers bonded together. Each cylindrical stack produces a finite displacement and when they are coupled together the overall displacement will be the summation of all the displacements.

2.1.3 Booster

The **booster** or **connector** as indicated in Figure 2-1 amplifies the displacements received from the converter by means of cross sectional area reductions which also act to increase the stress. The booster is designed to allow for one nodal line (line across the circumference with zero displacement) along its length. Since this line experiences no displacement it can be connected or clamped to the supporting rig.

2.1.4 Horn

The **horn** or **amplifier** amplifies the displacements received from the booster by means of cross sectional area reductions as in the case of the booster. These area changes act as displacement and consequently stress modifiers. These area changes act to accelerate and decelerate the longitudinal displacement wave through the stack. The wave movement through the horn and specimen can be defined by the analytical wave equation.

Analytical Wave Equation

The wave can be modelled using the second order differential normal wave equation given by Bathias [4].

$$\rho \mathrm{S(x)} \frac{\partial^2 u}{\partial t^2} = \frac{\partial f}{\partial x} \qquad \textbf{1}$$

Where,

S(x) - Is the area of the cross-section at location x,
f - Is the force acting on the section,
u - Is the longitudinal displacement,
ρ - Is the density of the material,
x - Is the longitudinal position
t - Is time.

This equation can be solved using either an ODE (ordinary differential equation) solver in Matlab or it can be analytically solved using the area function together with some mathematical manipulation as follows:

Initially the force in the longitudinal direction is assumed to be based on the axial strain/stress, thus the following is standard representation of axial stress:

$$f = \sigma.S(x) = ES(x)\varepsilon$$

$$f = ES(x)\frac{\partial u}{\partial x} \qquad \textbf{2}$$

Where,

E - Is the materials Young's modulus,
f - Is the force acting on the section,
σ - Is the stress at location x,
ε - Is the strain at the location x.

If equation 2 is substituted into equation *1* the following can be obtained:

$$\rho \mathrm{S(x)} \frac{\partial^2 u}{\partial t^2} = \frac{\partial}{\partial x}\left(ES(x)\frac{\partial u}{\partial x} \right)$$

Assuming the material is homogenous in nature then E is constant throughout the length of the beam, thus the equation simplifies to:

$$\frac{\partial^2 u}{\partial t^2} = \frac{E}{\rho S(x)}\frac{\partial}{\partial x}\left(S(x)\frac{\partial u}{\partial x} \right)$$

Applying the differentiation chain rule the following is derived:

$$\frac{\partial^2 u}{\partial t^2} = \frac{E}{\rho S(x)}\left[\frac{\partial S(x)}{\partial x}\frac{\partial u}{\partial x} + S(x)\frac{\partial^2 u}{\partial x^2} \right]$$

Simplifying,

$$\frac{\partial^2 u}{\partial t^2} = \frac{E}{\rho S(x)} \frac{\partial S(x)}{\partial x} \frac{\partial u}{\partial x} + \frac{E}{\rho} \frac{\partial^2 u}{\partial x^2} \quad \textbf{3}$$

But if it is noted that the speed of a wave on a one dimensional material is:

$$c = \sqrt{\frac{E}{\rho}} \quad \textbf{4}$$

Then the above equation 3 becomes:

$$\frac{\partial^2 u}{\partial t^2} = \frac{c^2}{S(x)} \frac{\partial S(x)}{\partial x} \frac{\partial u}{\partial x} + c^2 \frac{\partial^2 u}{\partial x^2} \quad \textbf{5}$$

And if a function P(x) is defined for simplification of the above expression as:

$$\mathrm{P(x)} = \frac{1}{\mathrm{S(x)}} \frac{\partial \mathrm{S(x)}}{\partial \mathrm{x}}$$

Then equation 5 becomes as per reference [4]:

$$\frac{\partial^2 u}{\partial t^2} = c^2 \left[P(x) \frac{\partial u}{\partial x} + \frac{\partial^2 u}{\partial x^2} \right]$$

The general solution of equation 5 (under boundary conditions of ultra-sonic fatigue specimens) is [4]:

$$u(x,t) = U(x)\sin(wt) \quad \textbf{6}$$

Substitution of equation 6 into equation 5 then the following equation can be obtained by eliminating the trigonometric functions.

$$\frac{\partial^2 U(x)}{\partial x^2} + P(x) \frac{\partial U(x)}{\partial x} + k^2 U(x) = 0$$
$$w = 2\pi f \quad \textbf{7}$$
$$k = \frac{w}{c}$$

Where

f - is the frequency

To find a solution to equation 7 it is necessary to have a defined outer profile for each section of horn and specimen. An example of the analytical solution for a given horn and specimen profile is indicated in Appendix A.

The horn is designed to give the required strain in the centre of the specimen and the horn itself should have a resonant frequency with an axial displacement mode at the chosen input frequency of the converter.

2.1.5 Specimen

The **specimen** is attached or in-contact with the horn and is designed to resonate at the input frequency, which in this case will be approximately 20 kHz. The design consequently gives maximum stress at the smallest cross section of the specimen, which corresponds to the minimum displacement. This is done to ensure that fatigue failure occurs in the specimen and not in another part of the testing equipment. In this dissertation the design will be focussed on the typical dumbbell specimen for the loading ratio (R) of -1, as indicated in Figure 2-2.

2.1.6 Control and Monitoring System

General System

The **control and monitoring system** is a computer based system which monitors the expected load at the centre of the specimen (location of smallest cross sectional area) together with the input frequency. This data is necessary to determine the number of cycles to induce failure at a constant stress and hence generate a point on the SN diagram for the given material specimen. The load is maintained to ensure that even when a crack is formed in the specimen, thereby reducing the stiffness, that the load at the centre is not altered drastically. The stiffness of the specimen controls the rate at which the axial excitation propagates through the stack (booster, horn and specimen combination). Accordingly any material modulus or cross sectional area changes in the stack will cause the displacement and hence the stress to increase or decrease as per the analytical wave equation described in the previous section 2.1.4. A typical stress and displacement variation across the stack is indicated in Figure 2-2.

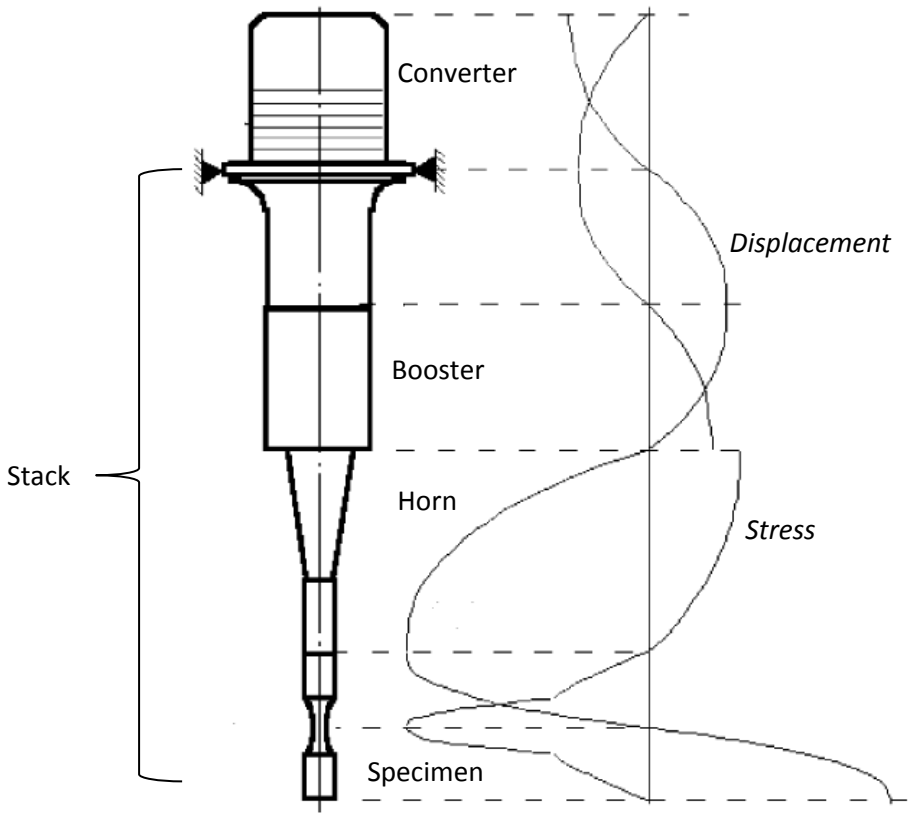


Figure 2-2: Variation of stress and displacement in a typical stack [4]

Cycle Counting

The life of the specimen can be obtained via the recorded frequency (cycles per second), which can be multiplied by the time of each test to obtain the total number of cycles. A secondary method of cycle counting will be used on the data accumulated from the measurement system.

Cycle counting is required in any fatigue life test which includes stress-life (SN) and strain-line (εN) curves. The cycles to failure are calculated based on the cyclic loads experienced by the specimen. In a perfect coupon test the cyclic loading will remain at an ideal level of stress or strain for the duration of the test. Thus a typical cyclic input could be represented as indicated in Figure 2-3:

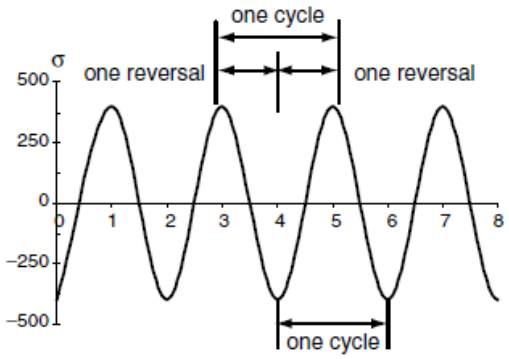


Figure 2-3: Constant cyclic loading [5]

In testing coupons the number of cycles needs to be calculated and recorded per specimen. Figure 2-3 indicates a constant amplitude signal and hence a simple one-parameter cycle counting method such as level crossing, peak-valley and range counting can be implemented to evaluate the total number of cycles. These methods are however not so efficient if the amplitude is varying in time.

One of the most popular methods to use at the moment for varying load cycles is the Rainflow counting method. This method is based on the analogy of raindrops falling down off a pagoda roof. Rain flow counting is a two parameter cycle counting method of which there are two defined options, namely the three point (ASTM E1049,1985) or the four point method [5]. The methods indicated in this section will that of the three point method.

The three point method uses three consecutive points (S1, S2, and S3), shown in Figure 2-4, to determine if a load cycle is formed [5]. Thus two consecutive ranges are defined by [5]:

$$\Delta S_1 = |S_1 - S_2|$$

$$\Delta S_2 = |S_2 - S_3|$$

If $\Delta S_1 \leq \Delta S_2$ then one cycle exists from S_1 to S_2. If $\Delta S_1 > \Delta S_2$ then no cycle is counted. The three point cycle counting method requires that the signal be rearrange from highest or lowest valley, whichever is greatest in magnitude and that the signal only consists of peaks and valleys [5]. This method is summarised in the Figure 2-4.

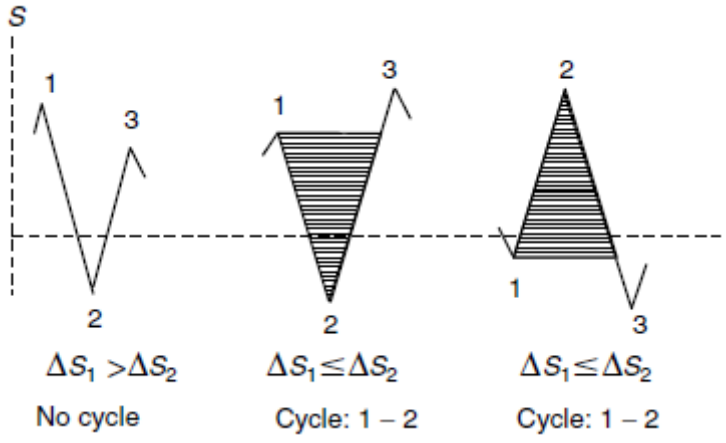


Figure 2-4: Three point Rainflow counting [5]

A table can then be setup comprising the number of cycles, the starting points, the end points, the range and the mean. This method is initially chosen to verify how stable the peak magnitude is maintained during testing. If it is maintained to a high accuracy a quicker method such as peak-picking will be implemented where the peaks are identified by changes in the curve gradient.

Use of Strain Gauges and Laser Vibrometers

One of the conundrums within UHCF is the method to measure the cycles and strain experienced by the specimen. The literature surveyed indicated that most UHCF testing machines, had a measurement system that consisted of either laser (optical) measurement devices or strain gauges

as the main measurement source. Both options have their pros and cons these are summarised in Table 2-1.

Table 2-1: Strain gauge and laser vibrometer comparison

	Strain Gauges	Laser/ Optical measurement
Cost	Constant consumable cost	Large initial outlay in cost but no real consumable costs except general maintenance
Detection method	Measured strain directly off gauge location. This may result in direct fatigue of the strain gauge which may affect data. It will be necessary to regularly check the strain gauges (re-calibrate) and possibly add redundancy for failure during testing.	Measures velocity directly using Doppler affect which is a non-contact measurement method thus vibrations will not affect the data
Accuracy	Strain gauges can provide good accuracy, depending on type of gauge used and signal conditioning	Very high accuracy, but as the accuracy and resolution increases so will the cost of the vibrometer

For further information on both the strain gauges and the laser vibrometer refer to Appendix B.

2.2 Applications of UHCF Testing

In the new age of high speed components in advanced equipment it has become apparent that these new devices will be required to operate in fatigue conditions in the giga-cycle region of the SN curve. In this section a typical example of where UHCF test data can be used will be discussed to indicate the applications for UHCF testing. The example indicated here is just one example that was implemented at Denel Aerostructures, however there are substantially more possibilities.

One of the problems encountered during work on the Airbus A400M WFF (Wing to Fuselage Fairing) project was where sonic fatigue was considered. This was required for certification due to Federal Aviation Requirements and Joint Aviation regulations (CS 25.571 subchapter d) where the structure needed to be considered for sonic fatigue failure due to sonic excitation. The sonic excitation is created via the impinging air pressure on the structure generated by the giant propellers used on the A400M. This can be perceived from the size of the propellers in the Figure 2-5 of the Airbus A400M, Atlas.



Figure 2-5: Image of propeller size on A400M and blade number

Consequently the pressure distribution due to the rotational velocity of the propellers is imparted onto the structure including all subsequent harmonics. These different pressures can be estimated for the entire structure and hence various pressure distributions are created for each harmonic. A typical pressure plot for one harmonic is indicated in Figure 2-6.

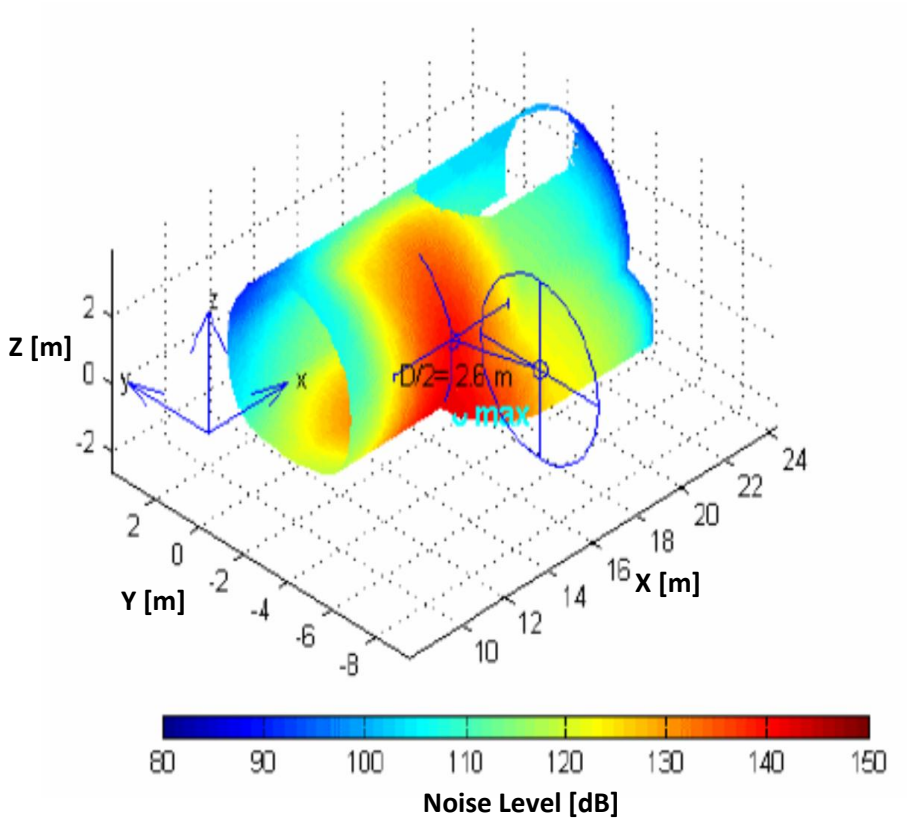


Figure 2-6: Typical pressure plot of one harmonic of the blade pass frequency

Typically when dealing with sonic fatigue the number of cycles for the first blade pass frequency (BPF) can be calculated based on a rotating propeller speed of 10.9Hz (655 rpm). Consequently the typical cycles for the first blade frequency pressure over a 3 hour flight for 10,000 flight cycles is given by:

N_1=10.9[Hz] x 8[blades] x 3600 [sec/hour] x 3[hour/FC] x 10,000[FC] x 5[SF] = 4.7e10

Note: 10,000 flight cycles is the typical regulated fatigue cycle (FC) life estimate for an aircraft. The typical safety factor (SF) implied for fatigue is taken as 5 and 3 hours is the typical flight time.

Accordingly the 5^{th} harmonic will be five times the first harmonic value where N_5=2.6e11. As a result, even though the stresses in the structure are low for the impinging pressure, the effect of applying the load over the life of the aircraft will destroy the structure long before 10,000 flight cycles is reached if extrapolation of the typical SN (stress vs. cycles) curve is utilised. This methodology is indicated by the solid blue line in **Figure** 2-7.

Two options exist to obtain a representative fatigue allowable at the required number of cycles. Either direct extrapolation from the known data up 10^6 can be implemented (generally far too conservative). The other option is some form of gradient change of the SN curve to match known test data. One of the best known modifications to the SN curve has been suggest by Haibach [34]. In this Haibach suggests that the typical Basquin SN relationship given by,

$$S^k.N = Constant$$

Can be modified by replacing the power k with 2k-1 for cycles exceeding 10^6 as defined by Schijve [6]. This change equivalently reduces the gradient of the SN curve past 10^6 cycles. However to use this form of the SN curve it is required that the material conforms to this theoretical model. Thus it is necessary to acquire data points in the ultra-high cycle region of the SN curve to validate this approximation for the specific material under consideration. An application of Haibach is indicated for aluminium in Figure 2-7. This aluminium curve is based on data obtained from ESDU (Engineering and Science Data Unit) for aluminium 7050. The data experimental data of ESDU only went to a maximum cycle range of 10^7 cycles thus initially extrapolation was implemented.

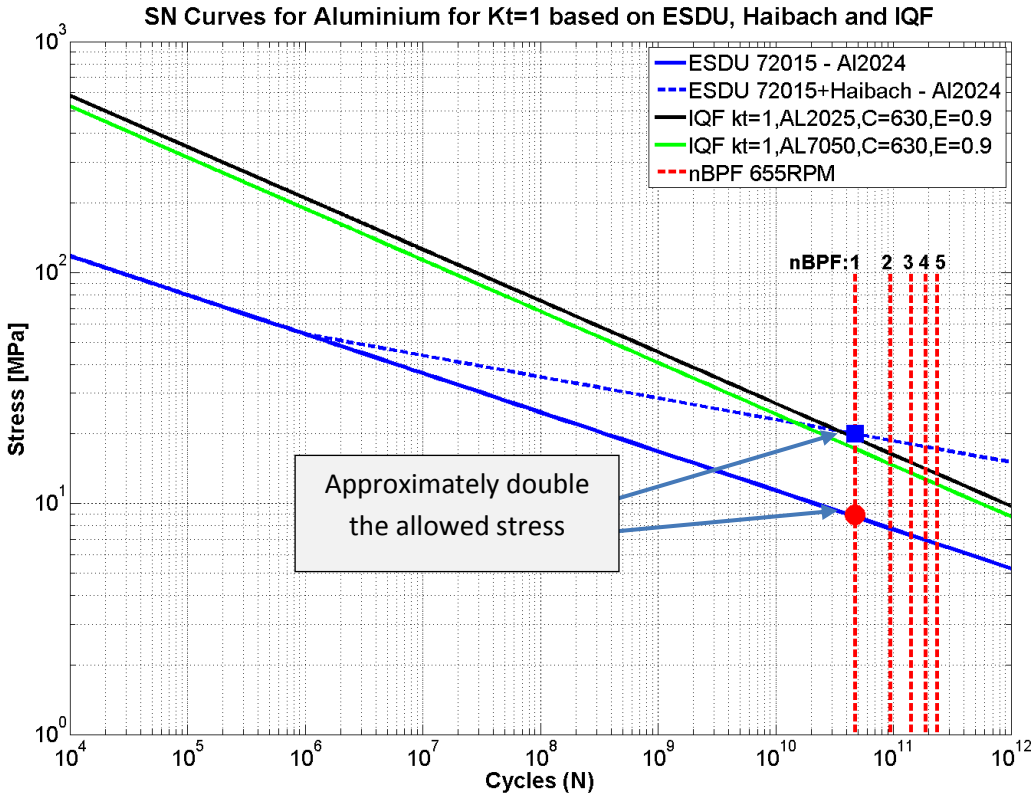


Figure 2-7: Typical SN curves for aluminium [7]

The extrapolated solid blue curve indicates that the stress allowed for the first harmonic (blue square) more than doubles if the Haibach modification is used instead of the standard extrapolated (red dot) SN curve. As an example, if the approximate application stress from the impinging air pressure was 10MPa in the structure then using extrapolation of the HCF (High Cycle Fatigue) SN data from ESDU the structure would have an RF (reserve factor) of less than one (indicated by lower red dot). However, if Haibach is implemented then the RF will easily surpass one, even for the fifth harmonic. The other two curves are extrapolated data for AL2024 which were added to indicate that even if a separate material is chosen with better fatigue properties the component will still not be acceptable due to the over conservative nature of extrapolation.

The above Haibach affect has been verified using UHCF testing data to better estimate the affect in the ultrahigh cycle region of the SN curve and has been implemented in the certification of the Airbus A400M. This methodology was implemented in references [7] and [8] and a comparison of

the data of Stanzl to that of the applied Haibach method is indicated below for AL2024 where the p value of 5.9 is the reduced gradient from 10.8 for the original curve as per ESDU.

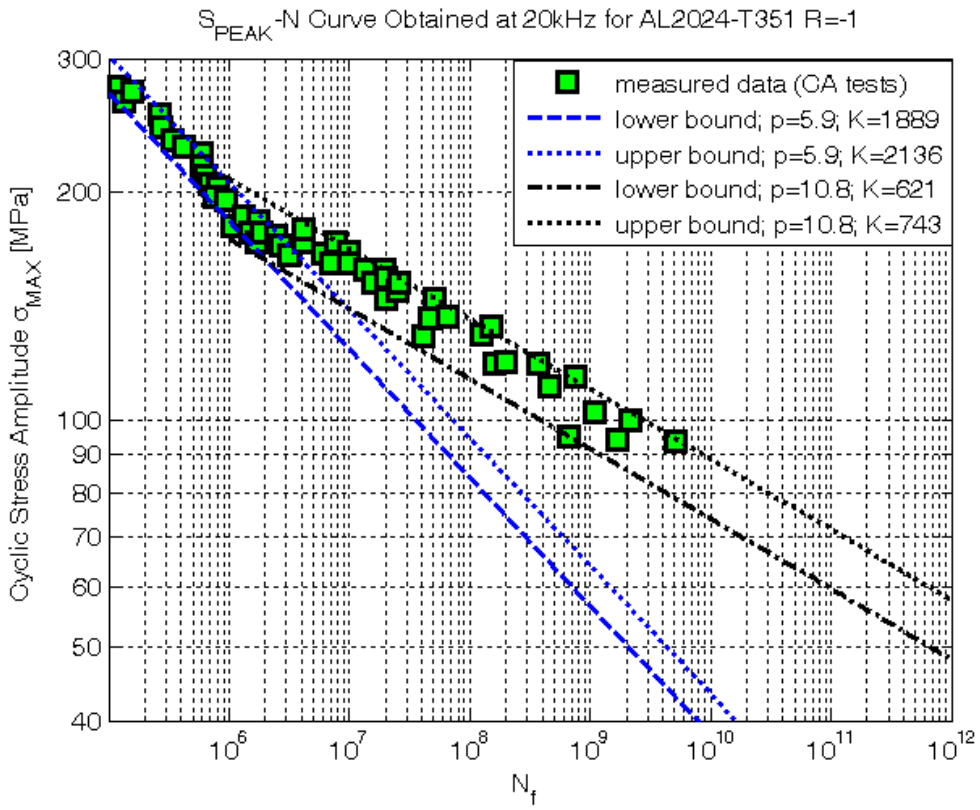


Figure 2-8: Haibach curve compared to Stanzl's UHCF data

A similar method to Haibach is that of the stepwise SN curve suggest by I Marines *et al.* [1]. The stepwise SN curve is the proposition of two logarithmic curves of varying gradient which are dependent on the number of cycles for the same material.

In this model, two different failure modes were noted which were based on a specific location of the SN curve. The different modes of fatigue initiation and propagation that are observed in the mega-cycle and giga-cycle domains are illustrated in Figure 2-9 showing micrographic cuts of coupons (material 42CRMo4) after fatigue tests at N_f=2.59x10^5 and N_f=5.75x10^8, respectively [9]. The different initiation modes are evident. The upper image of Figure 2-9 shows the typical surface inclusion and growth while the lower image indicates a so call "fish-eye" effect. Consequently due to the two different failure methods, two different slopes are proposed for a typical Basquin SN curve.

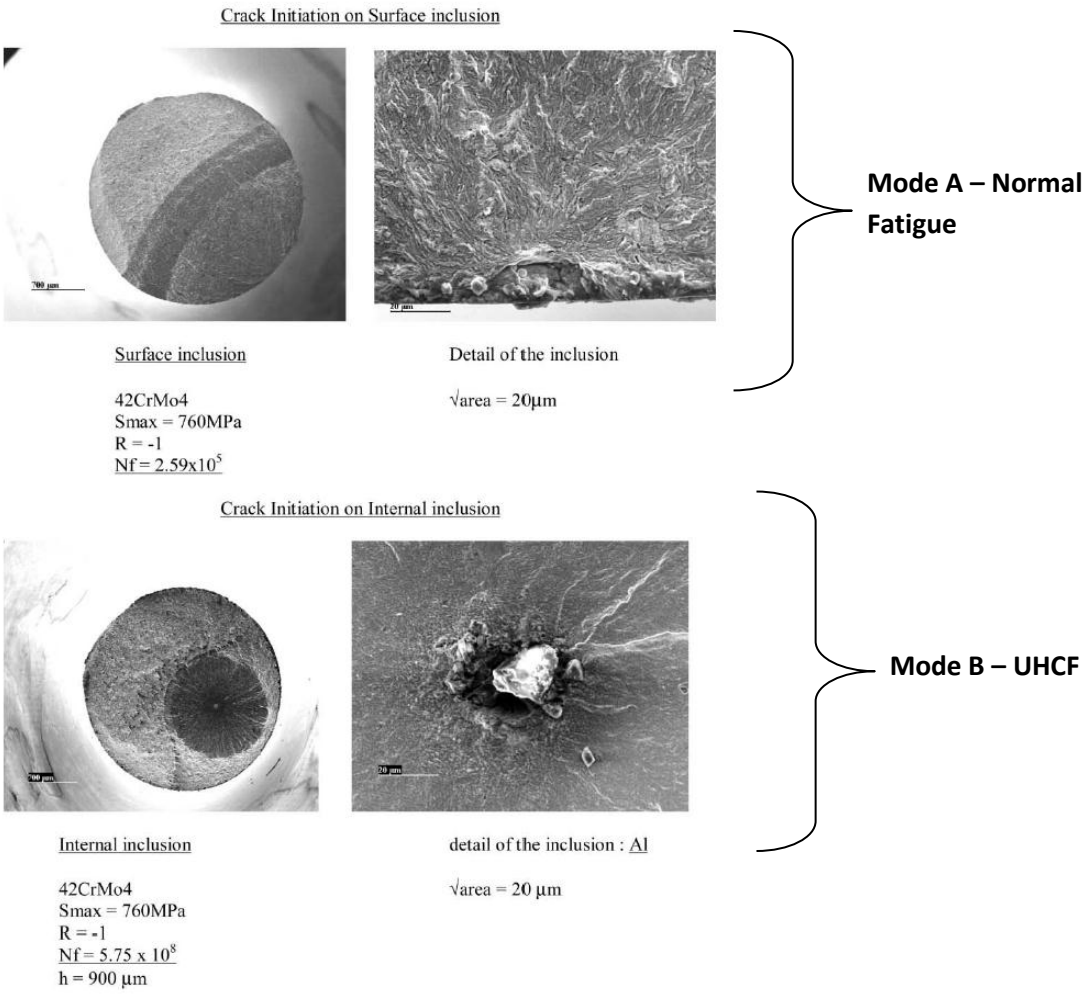


Figure 2-9: Crack initiation in surface inclusion and crack initiation in internal inclusion [9]

Figure 2-10 illustrates data for Aluminium 2024/T3 tested on conventional fatigue test machine (INSTRON 8501) [1]. One hundred specimens were tested at 13 constant stress levels and the dispersion in life time is very large at 'transition' stress levels (240 to 320 MPa). Micrographic cuts using SEM (scanning electron microscopy) revealed the existence of the two initiation modes as shown in figure 2-10. Mode A was typical initiation via slip bands before 10^6 while after 10^6 cycles the initiation was mainly due to inclusions [1]. This reinforces the notion of stepwise S-N curve introduced by Murakami [10] and Nisijima [11].

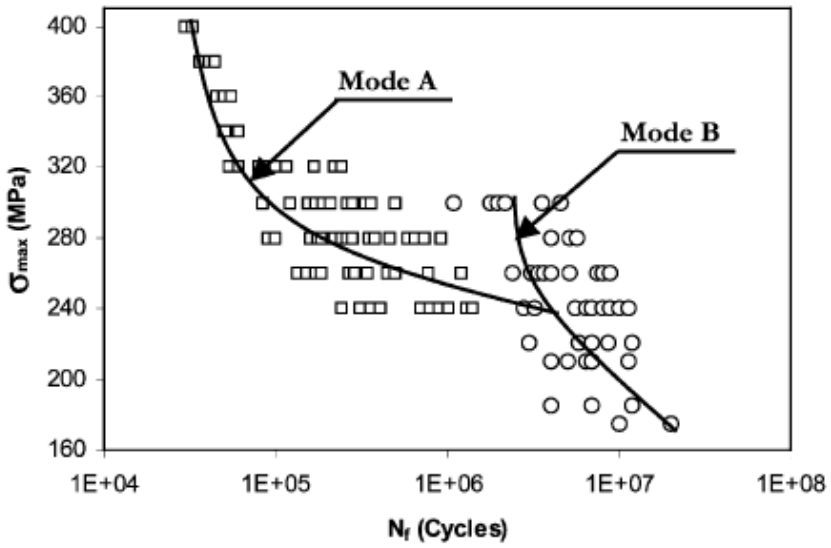


Figure 2-10: S-N curve of aluminium alloy 2024/T3; R=0.1 [1]

The literature showed that the fish-eyeing affect was mainly evident in steels as indicated by Nisijima [11]. Marines *et al*. [1] indicated that inclusions are evident at initiation in Al2024-T3; however, the fish eyeing affect was not directly evident. Conversely Stanzl [12], Mayer [13] and Chen [14] had no mention of any fish eyeing affects with the testing of AL2024.

One of the concerns that can be readily identified is what the effect of testing at such high frequency rates has on the obtained data. In combination with this is the concern of whether the data at measured high cyclic rates is still applicable to equivalent cycles for lower rates over a prolonged period of time. Thus the second mode B of Figure 2-10 could be due to the method of testing and not the material itself.

The effects of the frequency of the load application on the fatigue properties of materials have been considered by Papakyriacou [15, 16]. Papakyriacou [16] indicated that no statistically significant influence of cyclic frequency on the lifetimes was evident when compared to standard testing methods. Laird and Charsley [17] noted that in FCC (face centred cubic) materials it was found that the cyclic frequency had a small effect on the fatigue properties due to the small strain rate sensitivity of the material on shear stress. Typical FCC materials are aluminium, copper, nickel, gold and platinum. This can be verified by considering the data of ALZnMgCu1.5 which is an aluminium alloy typically referred to as AL7075 in the Aerospace industry. The following data comparisons were obtained for ALZnMgCu1.5 [16]:

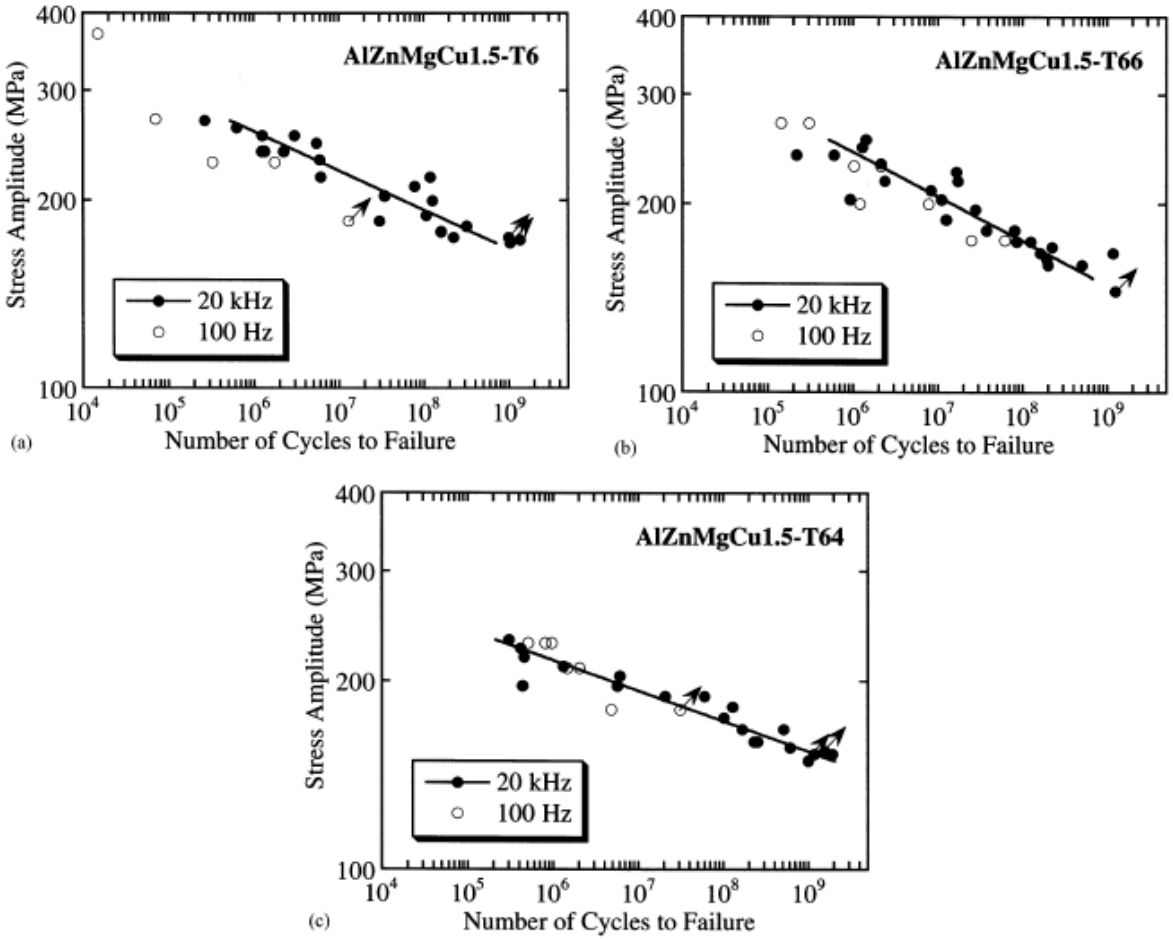


Figure 2-11: Endurance data T6 (a), T66 (b) and T64(c) at 20 kHz (dots) and 100Hz (circles) [16]

All the annealed materials indicated that for the region of 10^5 to 10^9 that there was no evidence of an endurance limit. The data indicated that for a 95% confidence level the data measurement overlapped for all three test materials for both testing methods which indicated that the frequency had little statistical effect on the results.

Papakyriacou [15] also considered the effect of loading frequency on the BCC (body centred cubic) and HCP (hexagonal close pack) material types. Typically BCC materials are Niobium and tantalum while typical HCP materials are titanium and zinc. Papakyriacou noted that work done by Roth and Willertz indicated that FCC materials are not affected by frequency affects if adequately cooled for tests carried out between 60 Hz and 20 kHz. However, Papakyriacou found that for BCC Tantalum that there was significant loading frequency affects due to the cold working on Tantalum when the material was cycled at 20 kHz [15]. This effect was not evident in the Niobium which is also BCC. This is indicated in the extracted data curves of Figure 2-12.

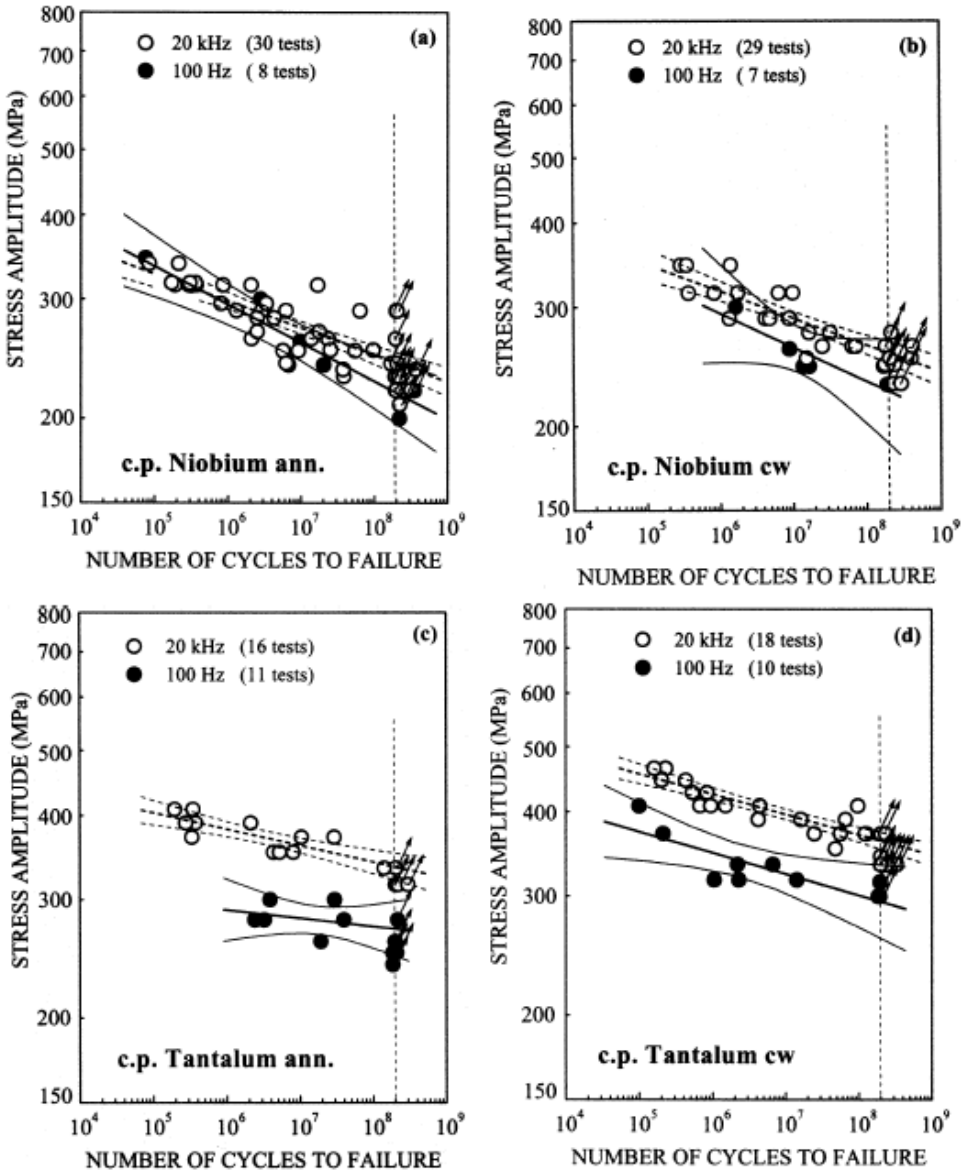


Figure 2-12: Endurance data of (a) commercially pure Niobium annealed and (b) commercially pure Niobium cold worked, (c) commercially pure Tantalum annealed and commercially pure Tantalum cold worked [15]

The effect of loading frequency on the HCP Titanium material indicated that there was no significant affect, but there was an affect for commercially pure annealed Titanium [15] as indicated in Figure 2-13.

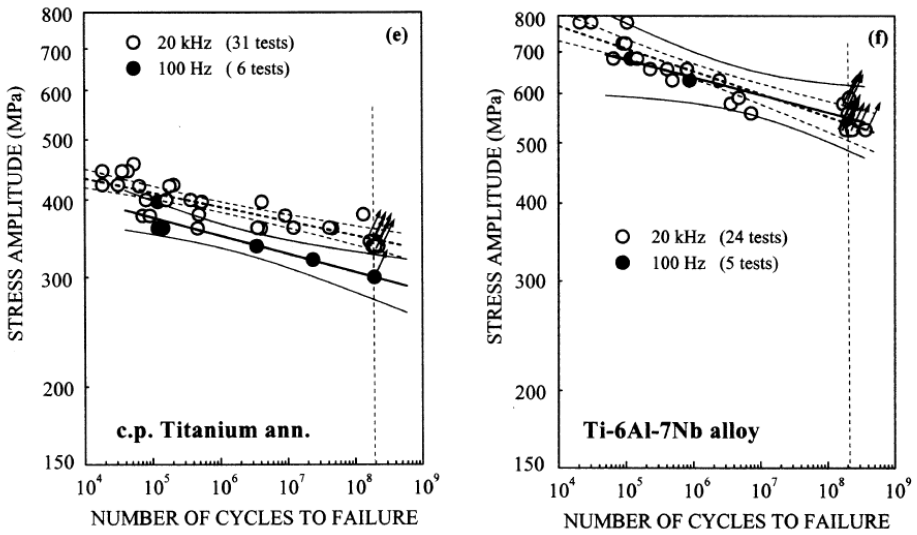


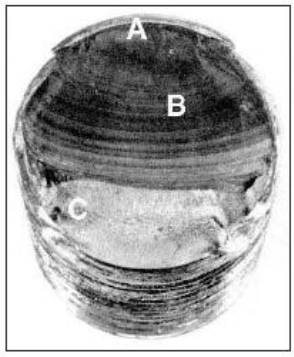
Figure 2-13: Endurance data for commercial pure anneal Titanium and Titanium [15]

The identifying feature from both of Papakyriacou's papers is that depending on the material type, strain rate affects may be present but cannot be isolated based on the crystalline structure of the material. The effects seem to be more dependent on the specific material properties of each material. Overall the effect of the frequency of testing should have little effect on the fatigue results for the materials being tested in this dissertation since both specimens types will be aluminium with a FCC crystalline structure which has been shown to not be affected by the frequency of testing; however this should be validated for each material.

Both Papakyriacou and Mayer did find that FCC aluminium was not affected by the loading frequency when comparing 100Hz fatigue testing to UHCF testing at 20 kHz. This bodes well for the validation tests of this dissertation which will be based on AL2024 and AL7075.

2.3 Typical Fatigue Failures

In the testing of the specimens typical fatigue failures are expected. Typical fatigue failure is defined by the method of load application. In general for each load cycle a striation is generated and these build incrementally until the remaining cross sectional is no longer sufficient to support the alternating load. The typical fatigue failure of a cyclic tension specimen is indicated in Figure 2-14.



Typical fatigue failures are indicated by:

A. Initial growth from a crack or stress concentration.
B. Striation propagation (beach marks)
C. Final failure

Figure 2-14: Typical fatigue failure mechanism [18]

Since the testing that will be accomplished in this dissertation will be under tension and compression with an r-ratio of -1 the following are typical failure patterns for HCF.



Tension - Compression Reversed Bending

Figure 2-15: Typical failure method achieved during standard fatigue testing [18]

In the Figure 2-15, the typical failure method for reversed bending is also indicated. This type of failure method should not be present in any of the specimens as this would indicate that the main mode excited at resonance is not a pure axial mode but rather a combination of other modes and possible bending modes.

The literature indicates that fish-eyeing affects generally occur in steels and initiate by impurities or porosities in the material. When this fish eyeing affect happens it is evident that no striation growth was seen. However, all the literature on aluminium indicated no such fish-eyeing affect thus this effect is not expected to be found during testing.

2.4 Material Data Obtained

In this dissertation two materials will be considered namely Aluminium AL2024-T351 and AL7075-T7351. These materials will be used for the validation of the UHCF test system.

2.4.1 Aluminium 2024-T351 Fatigue Data

AL2024 is a 2000 series aluminium with copper as the major alloying element. The chemical composition is typically: 0.50%-Si, 0.50%-Fe, 4.9%-Cu, 0.90%-Mn, 1.8%-Mg, 0.10%-Cr, 0.25%-Zn, 0.15%-Ti and 0.15%-Others. The rest of the weight percentage is aluminium.

The T351 designation implies that the material is thermally solution heat-treated, cold worked and naturally aged. The Tx51 indicates that the material is cold worked after the heat treatment into bar form. According to specification AS9100B:2004 the aluminium AL2024-T351 meets the requirements for T4. This is useful as MMPDS (Metallic Material Properties Development and Standardisation) has fatigue data for T4 but not T351.

Testing of AL2024-T351 has been accomplished by Stanzl [12] in Austria where data was compiled for both random amplitude (combined using Miners rule) and constant amplitude loading in air as indicated in Figure 2-16 on an UHCF testing machine.

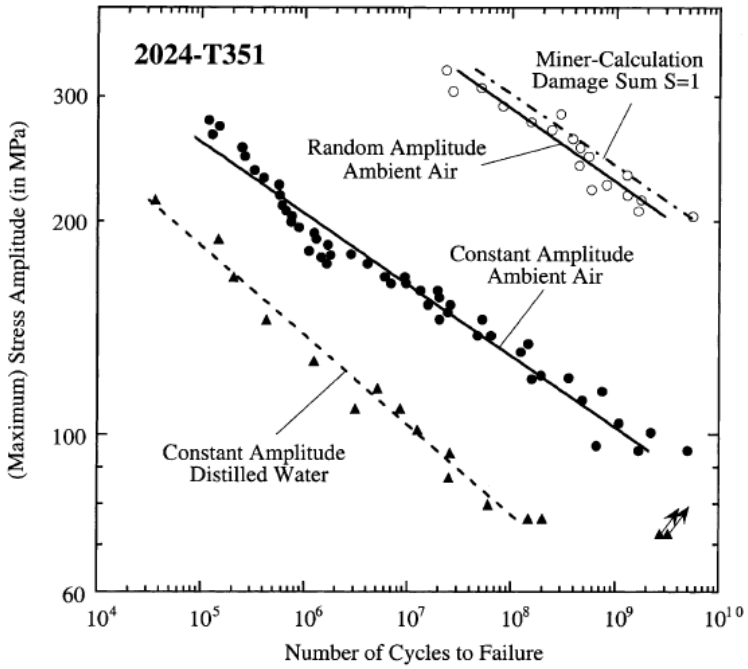
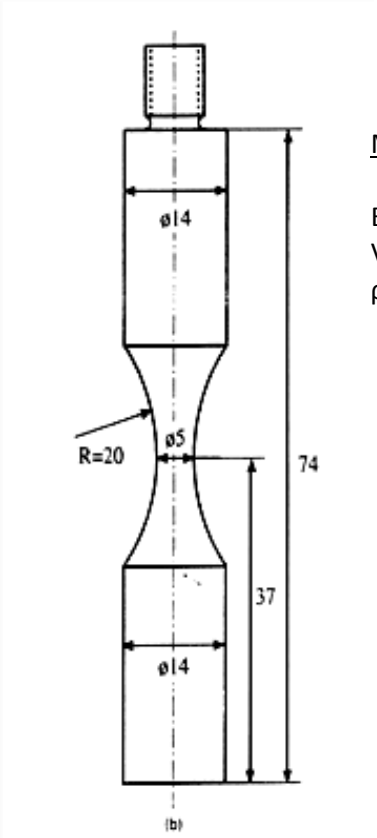


Figure 2-16: Fatigue data for aluminium 2024-T351, Stanzl [12]

A typical UHCF specimen geometry can be considered to have the following dimensions as taken from the work of Chen [14]:



Material Properties:

E=73.1GPa
V=0.33
ρ=2780kg/m^3

Figure 2-17: Typical aluminium 2024 specimen, Chen [14]

The data obtained using the geometry of Figure 2-17 for AL2024 as defined by Chen *et al*. [14] is indicated in Figure 2-18.

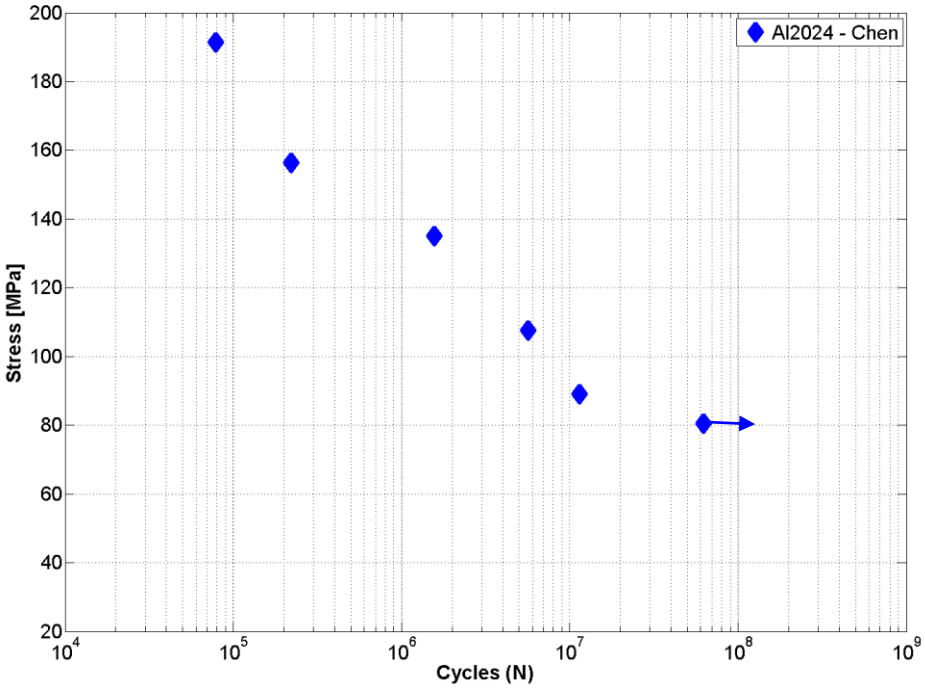


Figure 2-18: Fatigue data for aluminium 2024, Chen [14]

Design data is also available in many material handbooks; such data is indicated below and is typically obtained using the standard fatigue testing methodology on machines implementing cycling speeds up to 100Hz. Data obtained from ALCOA (Aluminium Company of America) is indicated in Figure 2-19.

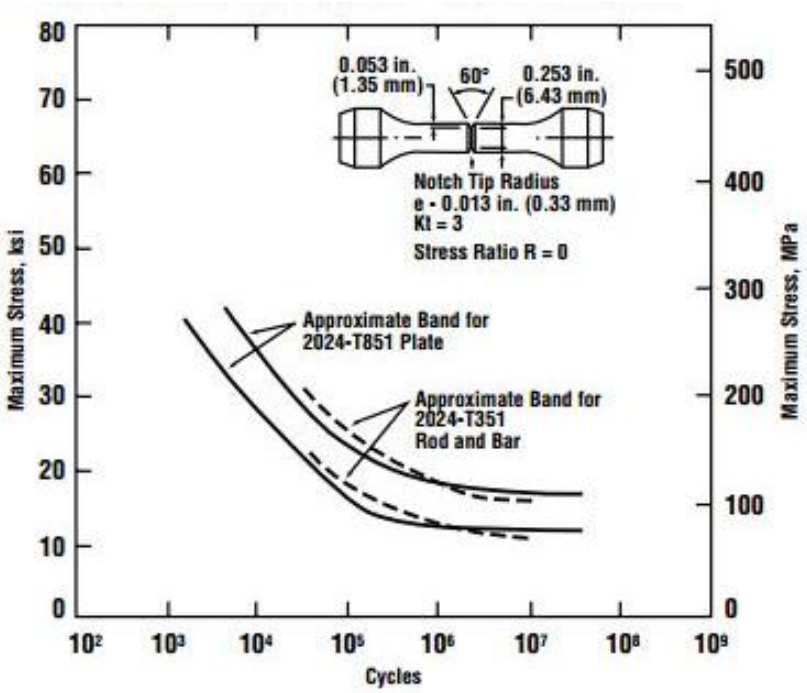


Figure 2-19: Data for AL2024 [19]

Data of AL2024-T4 and AL2024-T3 is available in MMPDS [20] and is indicated in Figure 2-20 which will be used for comparison since this should be the same as the T351 2024 aluminium.

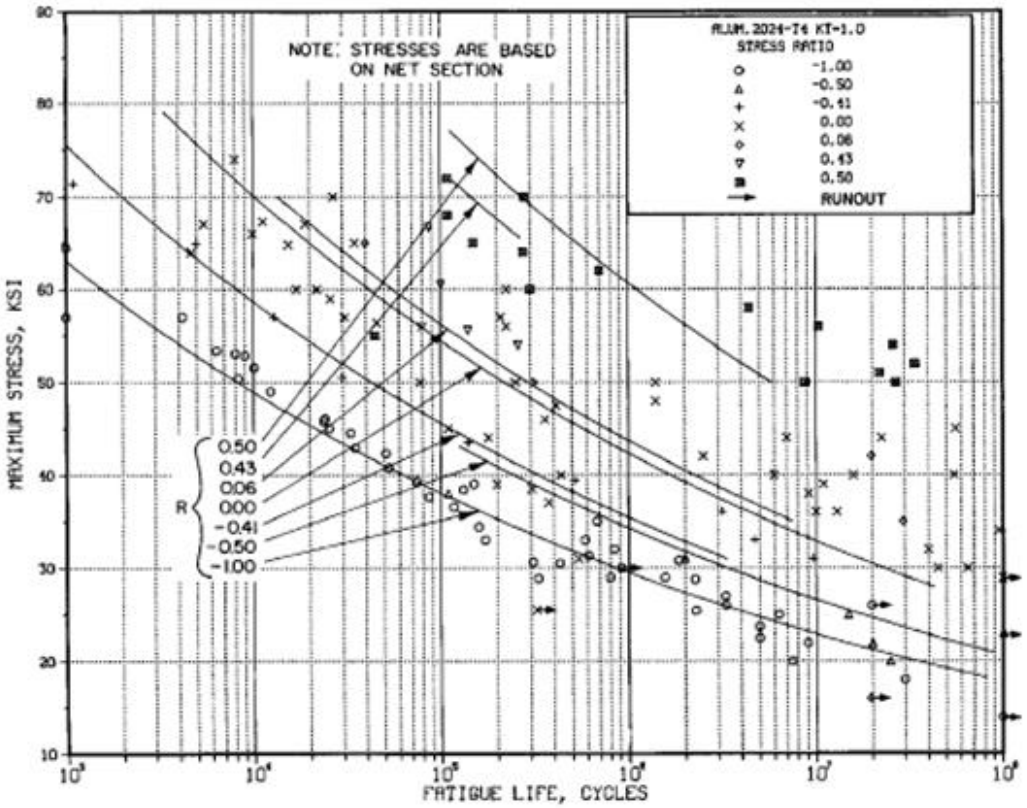


Figure 2-20: AL2024-T4 as per MMPDS [20]

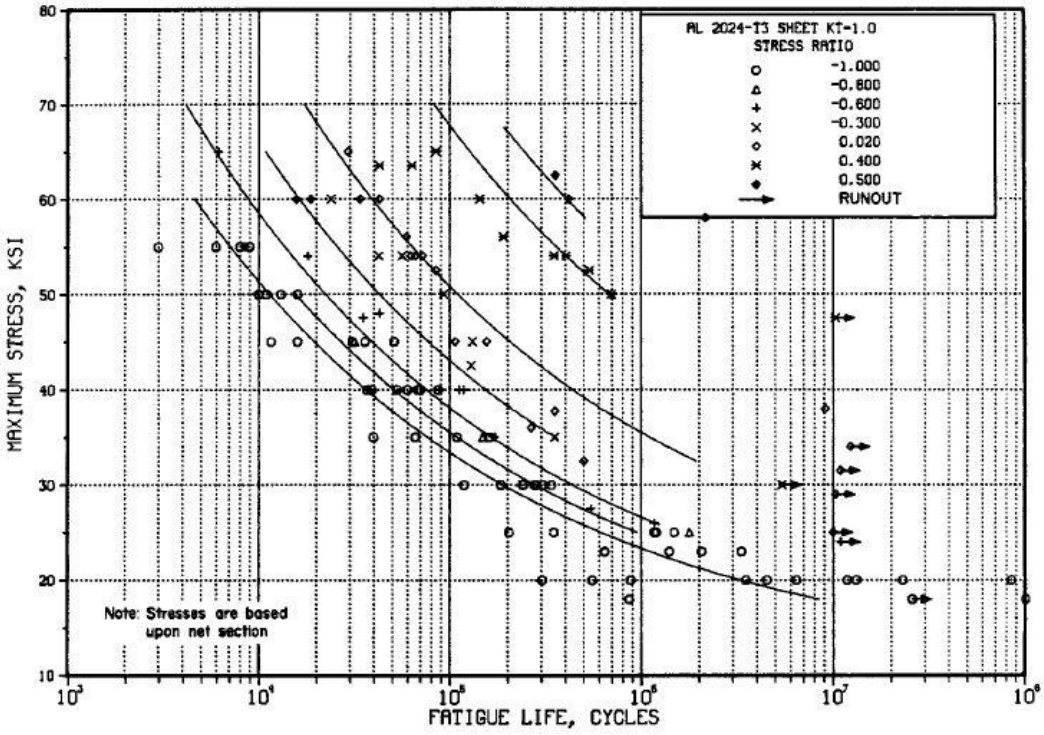


Figure 2-21: AL2024-T3 as per MMPDS [20]

During further investigation, additional data from that of Mayer [21] was obtained using the following specimen design for 20 kHz testing:

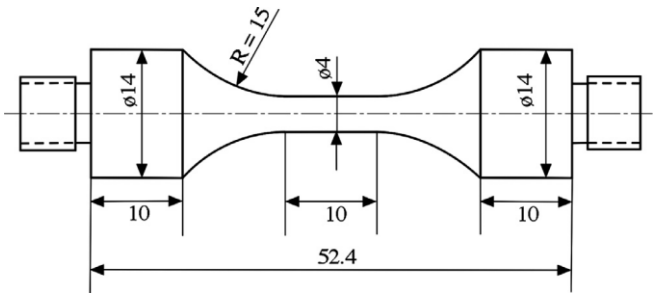


Figure 2-22: Specimen size for AL2024 data, Mayer [21]

The specimen geometry of Figure 2-22 was used for the examination of varying load ratios of -1, 0.5 and 0.1. The following is the data extracted for the load ratio for -1 which is applicable to this investigation.

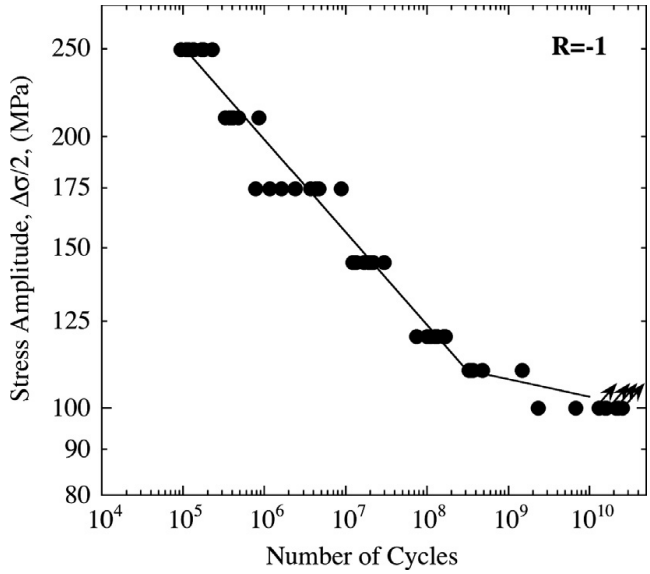


Figure 2-23 AL2024 data for load ratio of -1, Mayer [21]

The typical properties for AL2024-T351 bar are:

E=10.5x10^{3}ksi=72395MPa [20]
Ftu=62 ksi = 427 MPa [20]
Fty=45 ksi= 310 MPa [20]
ρ=0.101lb/in^3=2796kg/m^3 [20]
υ=0.33[20]

The yield (Fty) and ultimate (Ftu) strengths are given as a guide to see how close to yield the test stresses are. All the above curve data was combined onto a singular curve to obtain Figure 2-24.

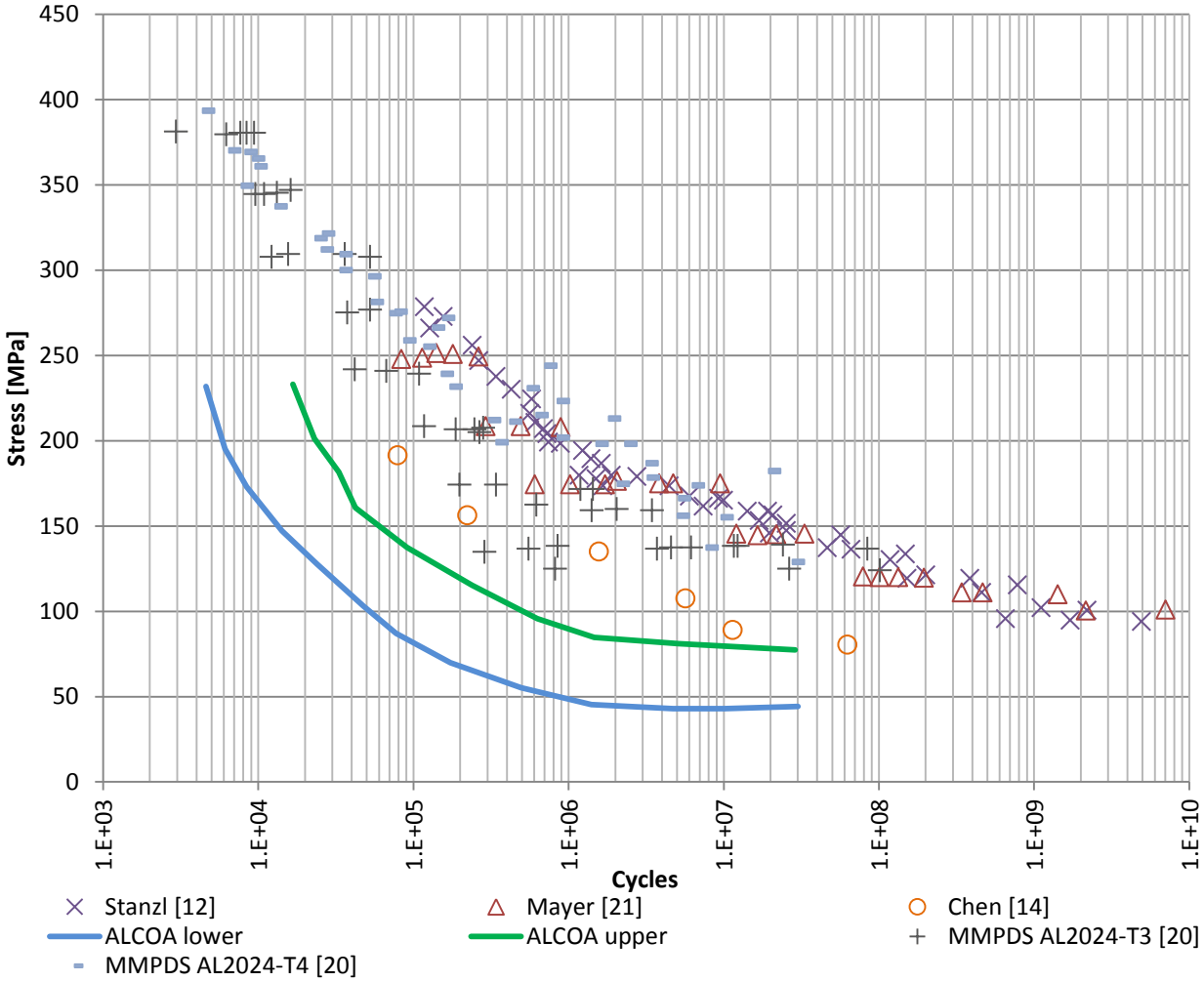


Figure 2-24: AL2024 combine data

The combined data indicates that the data of Mayer and Stanzl have a good correlation and indicate a Haibach affect from 10^6 cycles. The data of Chen however indicates a much more conservative trend. Chen's paper had no direct comparison to other literature data however it was noted that in these test the 2024 alloy indicated ductile facture as indicated in Figure 2-26. Chen doesn't explicitly define the type of temper of AL2024 that was used. The data of Chen corresponds to the data of MMPDS AL2024-T3 for cycles less than $5x10^6$. However for higher cycles Chen predicts much lower stresses for the same cycle range of 10^7 to 10^8 cycles. The reasons for this disparity in the data of Chen are unclear as the investigation was not carried out in the paper. But it is believed that the variation in the data may be due to the application of the dumbbell formula of Roth used by Chen which states:

$$\varepsilon_{max} = kA_o \left(\frac{d_1}{d_2}\right)^2 \frac{\sin(kl_1)}{\cos(kl_2)}$$

Where,

$k = \frac{2\pi}{\lambda}, \lambda = \frac{1}{f}\sqrt{\frac{E}{\rho}}$

A_o – is the displacement at the end of the specimen,

And the geometry is defined by:

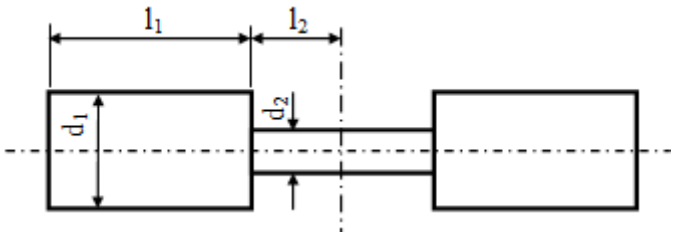


Figure 2-25: Dumbbell formula geometry definition [14]

However the specimen type implemented by Chen is that of the continuous curved specimen indicated in Figure 2-17. Thus this formula is very penalising on the stress/strain at the centre of the specimen. A simple comparison will be used to indicate how conservative Chen's formula is. This comparison will be done by comparing the equation 2.30a of Bathias [4] for a hyperbolic cosine curve specimen. The maximum strain at the centre of the hyperbolic cosine curve specimen profile as defined by Bathias equation 2.30a [4] is given by:

$$\varepsilon_{max} = \beta A_o \frac{\cos(kL_2)\cosh(\alpha L_2)}{\sinh(\beta L_2)}$$

Where,

$k = \frac{w}{c} = \frac{2\pi f}{c} = 2\pi f\sqrt{\frac{\rho}{E}}$

f - is the frequency (20000Hz)

ρ - is the density (2768kg/m^3)

E - is the Young's modulus (72.395GPa)

$\alpha = \frac{1}{L_2} arccosh\frac{R_2}{R_1}$

$\beta = \sqrt{\alpha^2 - k^2}$

L_2, R_1 and R_2, are the geometry of the specimen see Figure 2-27.

Thus implementing the geometry of Chen's specimen Figure 2-17, the following maximum strain comparison can between Bathias and Chen:

Chen – $\varepsilon_{max} = 116.50A_0$

Bathias – $\varepsilon_{max} = 139.65A_0$

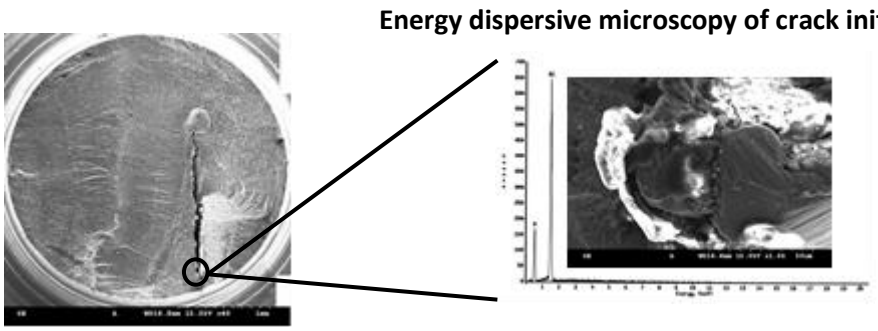


Figure 2-26: AL2024 cleavage fracture [14]

Consequently it is evident that if this is the formula employed in the calculations of Chen, that his strain and hence his stress will be slightly lower than that of Bathias by a ratio of 19.87%.

The data of ALCOA was significantly lower than all of the indicated curves in Figure 2-24 which is believed to be due to statistical knock down values to give a higher degree of confidence and to form a B-basis for design.

The geometry of the various 20 kHz UHCF test specimens is summarised in Table 2-2 based on the geometric definitions indicated in Figure 2-27:

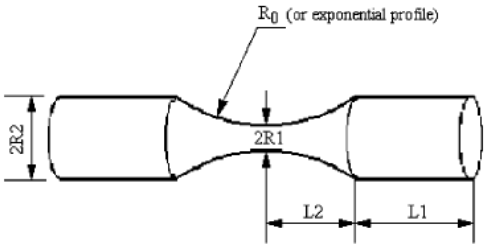


Figure 2-27: Geometry definition for specimen as defined by Bathias [4]

Table 2-2: Summary of Al2024 specimen Geometry

Geometric Dimension [mm]	Configuration		
	Stanzl	**Mayer**	**Chen**
R0	No direct geometry available	15 (10mm flat zone)	20
R1		4	5
R2		14	14
L1		10	25[1]
L2		16.2	12.51

It is evident from Table 2-2 that the specimen is sized depending on the horn configuration being used as all the specimens where implemented on 20 kHz UHCF testing machines. Unfortunately most paper found in the literature survey did not give any information regarding the horn geometry and thus no comparisons could be made.

[1] Estimated from scaled drawing due to missing dimension in reference [14]

2.4.2 Aluminium 7075-T7351 Fatigue Data

The AL7075 to be tested in this dissertation is AL7075-T7351. The AL7075 is a 7000 series alloy indicating that its major alloying component is zinc. The chemical composition is typically: 0.40%-Si, 0.50%-Fe, 2.0%-Cu, 0.30%-Mn, 2.90%-Mg, 0.28%-Cr, 6.10%-Zn, 0.20%-Ti and 0.15%-Other. The rest of the weight percentage is aluminium. The T7351 indicates that the material is solution heat treated and overage or stabilised. This material is specifically resistant to stress corrosion cracking.

In the MMPDS [20] material data for AL7075, only the T6 (solution treated and artificially aged) temper fatigue information was available as indicated in Figure 2-28.

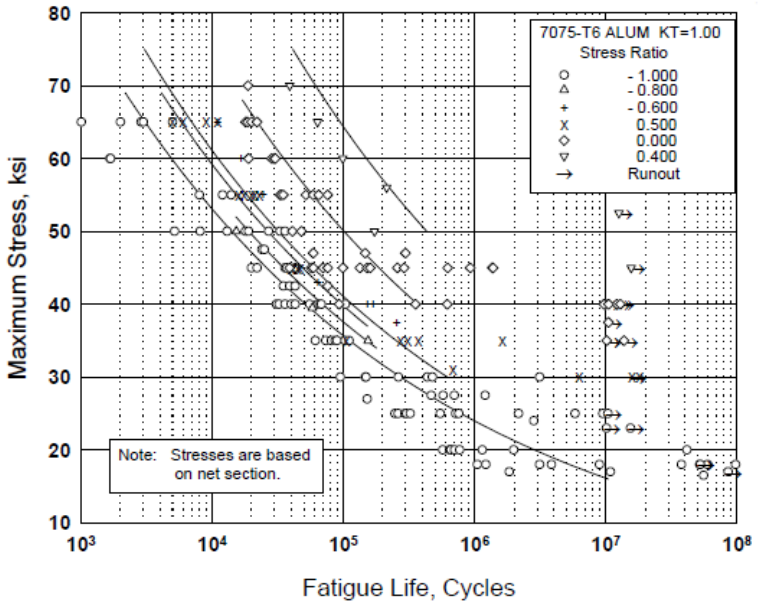


Figure 2-28: S/N curves for un-notched 7075-T6 aluminium alloy [20]

Note: The conversion of MPa to ksi is given by: 1 MPa = 0.1450377 ksi.

Figure 2-28 indicates that the number of cycles normally does not exceed 10^7 cycles for normal data acquired using hydraulic servo actuated fatigue testing machines. However when ultra-sonic testing machines are used the data can easily exceed 10^9 cycles.

The American Society for Metals (ASM) published fatigue data on light structural alloys [22], for a significant range of cycles up to and including the UHCF region. The data provided was obtained for AL7075-T73, see Figure 2-29, for both un-notched and notched specimens using standard fatigue testing methods. The data indicates that the material was tested well past 10^8 cycles with no significant indication of an endurance limit. Figure 2-29 also indicates the typical AL7075-T6 band which indicates that the difference between AL7075-T73 and AL7075-T6 for smooth specimens is almost negligible.

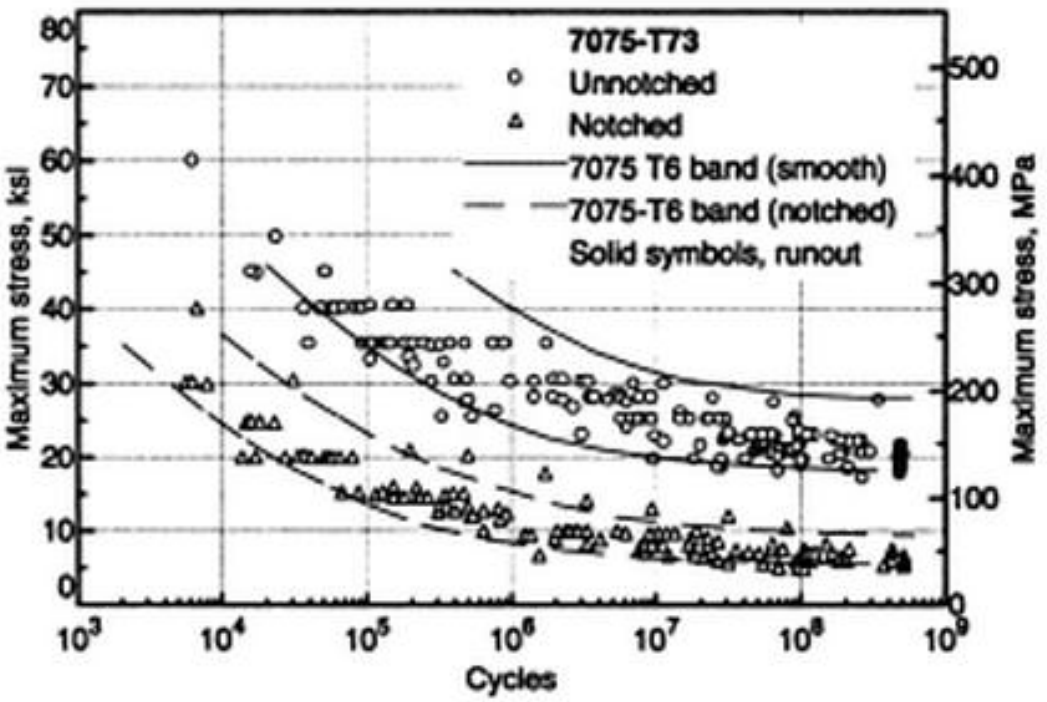


Figure 2-29: AL7075-T73 fatigue data [22]

A typical UHCF specimen sizing is defined in Figure 2-30 as extracted from the work of Hopler *et al*. [23] which considered the crack growth affect within aluminium alloys.

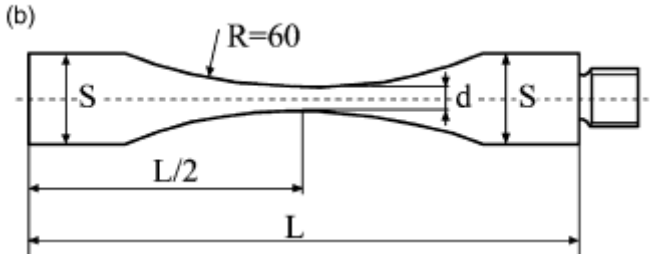


Figure 2-30: Aluminium 7075-UA typical specimen [23]

Where

d=3mm	E=10.3x10^{3}ksi=71016MPa [20]
S=15	ρ=0.101lb/in^3=2796kg/m^3[20]
L=82mm	υ=0.33[20]

Unfortunately Hopler did not provide any fatigue data which could be used for comparison in his publication.

Data obtained from ALCOA is presented below:

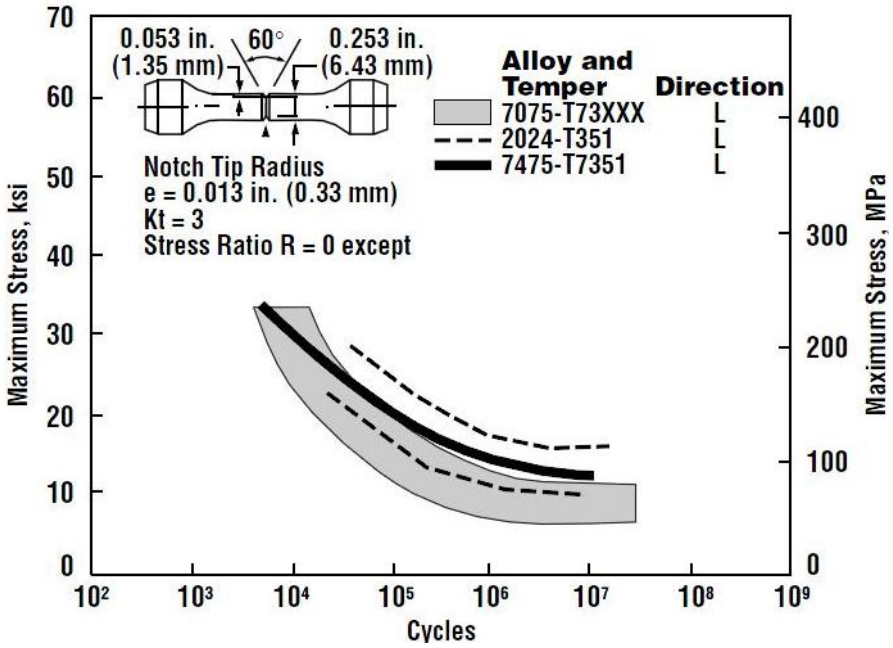


Figure 2-31: Data for AL7075 [24]

The typical ultimate tensile strength of the AL7075-T7351 is given as 68 ksi which is equivalent to 469MPa. The yield stress of Al7075 is given as 56ksi (386MPa) while the compressive yield is 54 ksi (372MPa). The Poisson's ratio is 0.33 [20] (based on T7351 or AMS-QQ-A-225).

The other name typically given to Aluminium 7075 is AlZnMgCu1.5. This material indicated the following data in Figure 2-32 as completed by Papakyriacou *et al*. [16] for two different test setups one at 100Hz and the other at 20 kHz.

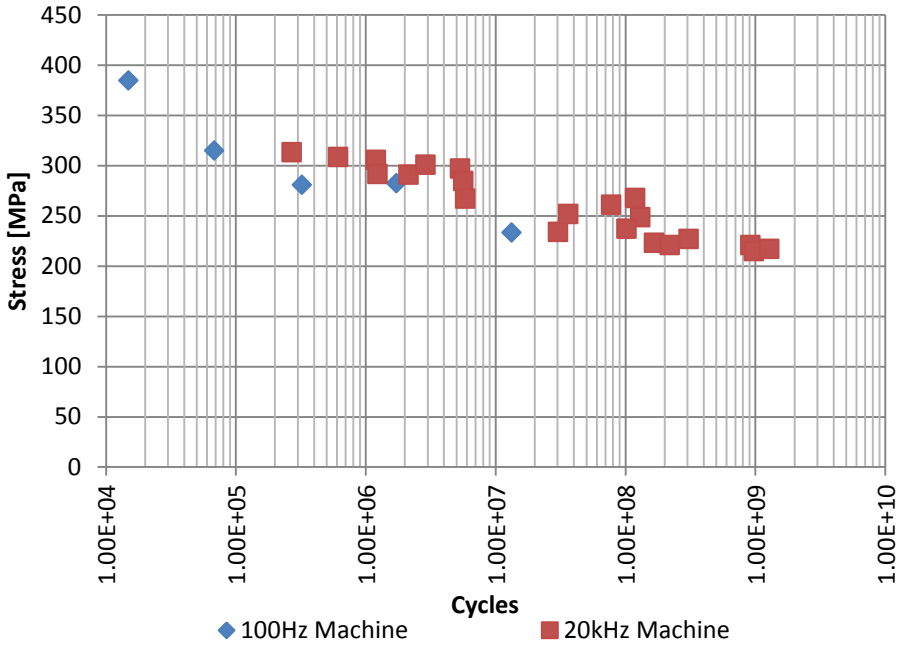


Figure 2-32: Papkriacou data for AL7075 (AlZnMgCu1.5-T6) [16]

The data in Figure 2-32 indicates two sets of data: One testing on a servo hydraulic fatigue system at 100Hz while the other data was provided by an UHCF testing machine at 20 kHz. Overall it can be seen that the effect of the different cyclic loading rates has a negligible effect on the final result as discussed previously.

All the data for AL7075 from the presented literature has been combined into a single Figure 2-33.

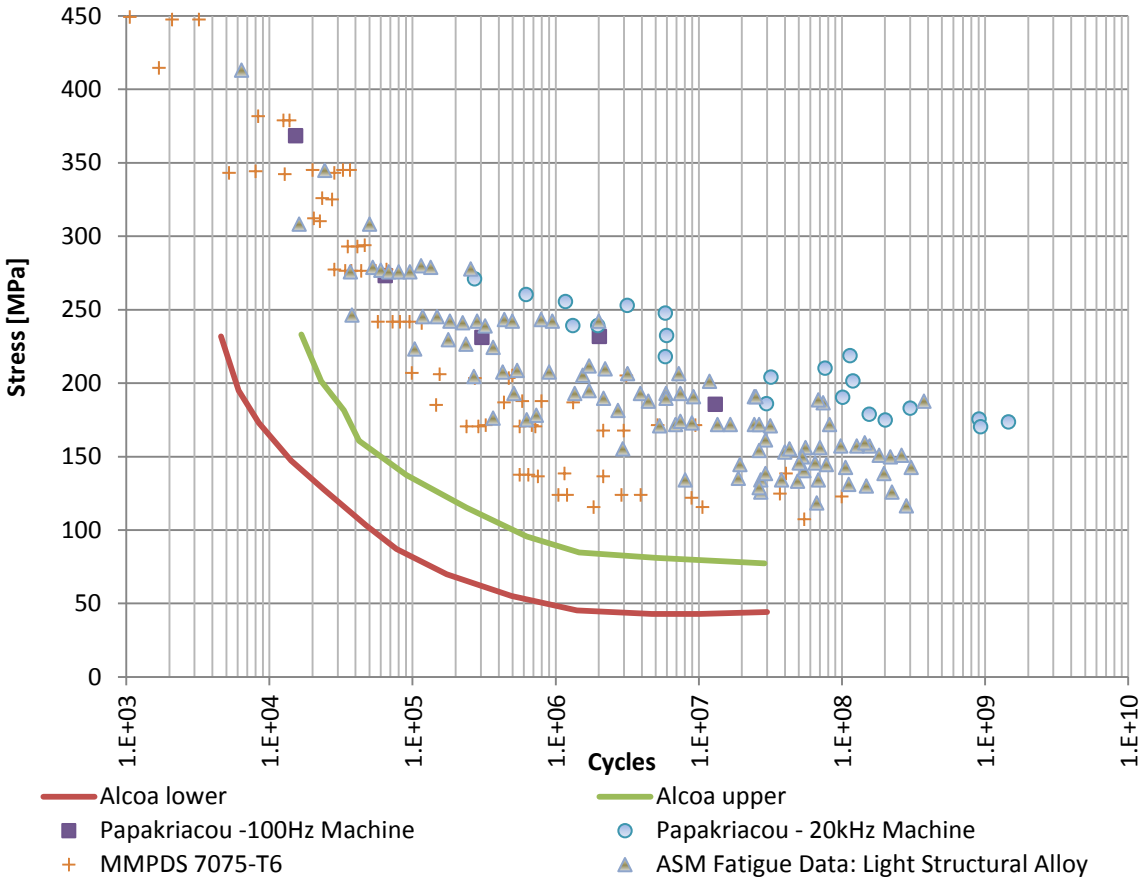


Figure 2-33: AL7075 combined data

Looking at the final combination of all the AL7075 data it can be seen that holistically there is a very high spread of the data. Thus the use of the estimated equations (trend lines) as provided in the documentation such as MMPDS and ALCOA can be misleading since the fatigue data has a significant scatter in general.

The scatter of fatigue data is generally owing to the following possible sources:

- Corrosion and aggressive operating environments affect the specimen's life, especially for tests performed over long periods of time. For example humidity will play a role in a fatigue test which is subject to long test periods if the laboratory does not have a controlled environment which is not always the case.
- Surface quality of the specimens has possible variation
- Machining of specimens can deteriorate the fatigue life, if the parts experience any high temperatures or sudden forces during manufacture.
- Typically if the tests are spread over large periods of time, then the specimens will be from different material batches of varying properties.

3 PRODUCT REQUIREMENT SPECIFICATION

The product requirement specification (PRS) for this design project can be summarised as by the following flow chart:

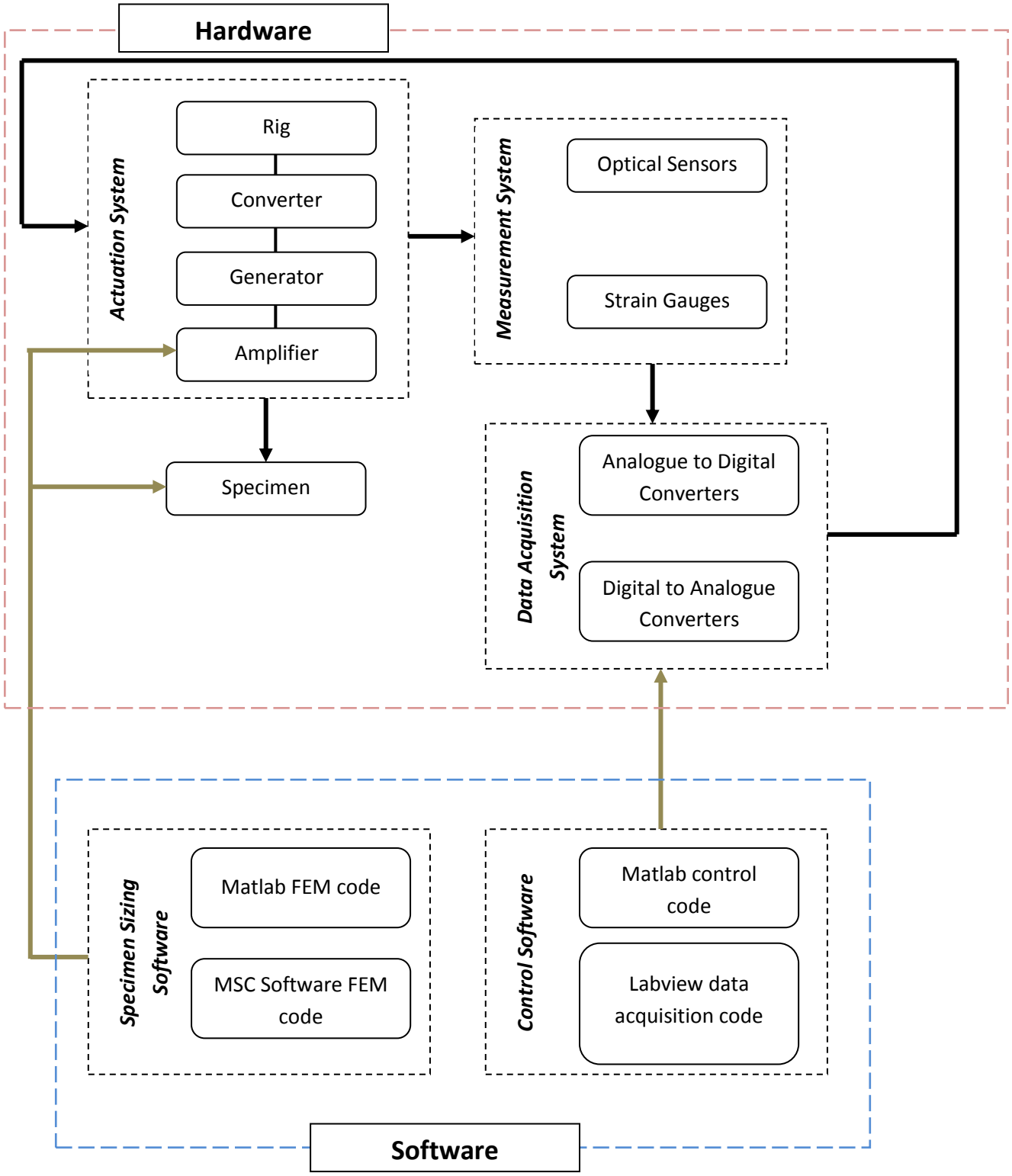


Figure 3-1: PRS for Design

3.1 Identification of a need

To date, there is no facility in South Africa capable of obtaining fatigue design SN data curves exceeding 10^6 cycles in a short period of time. Consequently the need for an UHCF testing machine is a viable design innovation for advance manufacturing in South Africa.

3.2 Requirements

The requirements of the UHCF test rig are defined as:

- The rig will be modular in design to allow for future development and tests.
- The setup will be able to produce repeatable and reliable ultra-high cycle fatigue test results for all ferrous and non-ferrous metals.
- The system will be a standalone system whereby all features are capable of being created without the need for engineering specific licenced software packages.
- The test system will be able to run continuously with minimum operator interaction.

3.3 Constraints

The constraints of the system can be defined as:

- Operating frequency of the system will be 20,000 Hz.
- The test rig's natural frequency should be sufficiently separated from the operating frequency of 20 kHz, to prevent resonance effects of the rig.
- The rig must be able to be operated using 240 Volt, 50Hz mains power supply.
- The system must be able to be placed in a standard room of maximum size 6 metre x 6 metres. This is the maximum allowed laboratory space at Denel Aerostructures (DAe).

3.4 Criteria

The criteria that will help to decide between the possible designs will include:

- The test rig should also be able to perform varying R fatigue ratios without the need for drastic changes to the setup.
- The test rig should allow for the possibility of testing composites in the future.
- The rig should operate in the smallest possible space to minimise the need for a large testing facilities.
- The manufacturing costs should be minimised to keep additional Research and Design funding available for testing.
- The costs of the full system including all code and hardware are to be kept to a minimum.
- Operational costs should be kept to a minimum.
- The system should be aesthetically pleasing.

4 DESIGN

During the design three iterations of the system were created which will be referred to as:

3.1. Mark 1 – Initial proof of concept test system
3.2. Mark 2 – Rig manufactured with strain gauge and laser vibrometer measurement systems implemented.
3.3. Mark 3 – Final design implementing only the laser vibrometer as the measuring device.

4.1 Mark 1 - Proof of Concept

4.1.1 Concept

The Mark 1 concept design consisted of an oversize horn; see Figure 4-1, manufactured from Al2024. The horn was oversized on one of the cylindrical sections, as indicated in Figure 4-1, so that it could be machined down on an iterative basis to obtain a correct resonance frequency of 20 kHz. The purpose of this design was:

- To allow for comparisons to be made with the 2 dimensional (2D) and 3 dimensional (3D) finite element models for verification and validation;
- To ensure that the off the shelf purchased Branson power supply and converter were sufficient to setup and maintain an axial wave in the specimen for a suitable time period to obtain failure or initial cracking of the specimen;
- To test if readable data could be obtained using typical foil gauges in a full bridge tension configuration;
- To validate the effect of the strain gauges on the frequency of the system and thus to determine if the horn design needed to account for the addition of strain gauges extra mass and stiffness.
- To estimate if the stress at failure could be calibrated to the analytical solution defined by the wave equation using the input of the strain gauge.

4.1.2 Design

The Mark 1 was designed to be a proof of concept design. In that it was used to verify that the Branson power supply could be used to setup a standing wave in the specimen, thereby generating fatigue failure. If this could be done it would give confidence that an UHCF machine could be manufactured and be used to obtain UHCF data. At this point no rig was manufactured and the converter was simply run in a horizontal position on a wooden table.

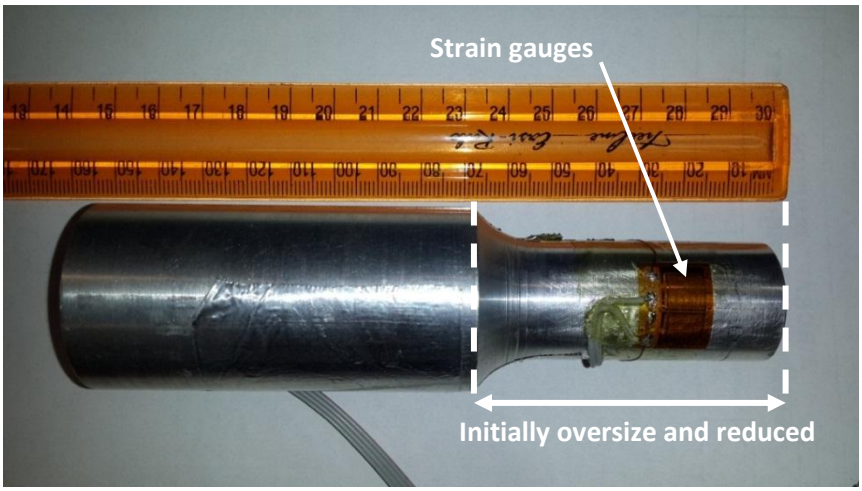


Figure 4-1: Mark 1 proof of concept horn with two gauges indicated

The calibration process was accomplished using a Branson horn analyser A-200A through continuous modification and retesting of the horn and specimen combination. The modification of the horn was accomplished by removing material from the oversize end as indicated in Figure 4-1. The resonant frequency of the horn and specimen combination was obtained by varying the frequency input until the applied current reduced to a minimum. The typical setup is indicated in Figure 4-2, where the horn and specimen are being tested in unison.

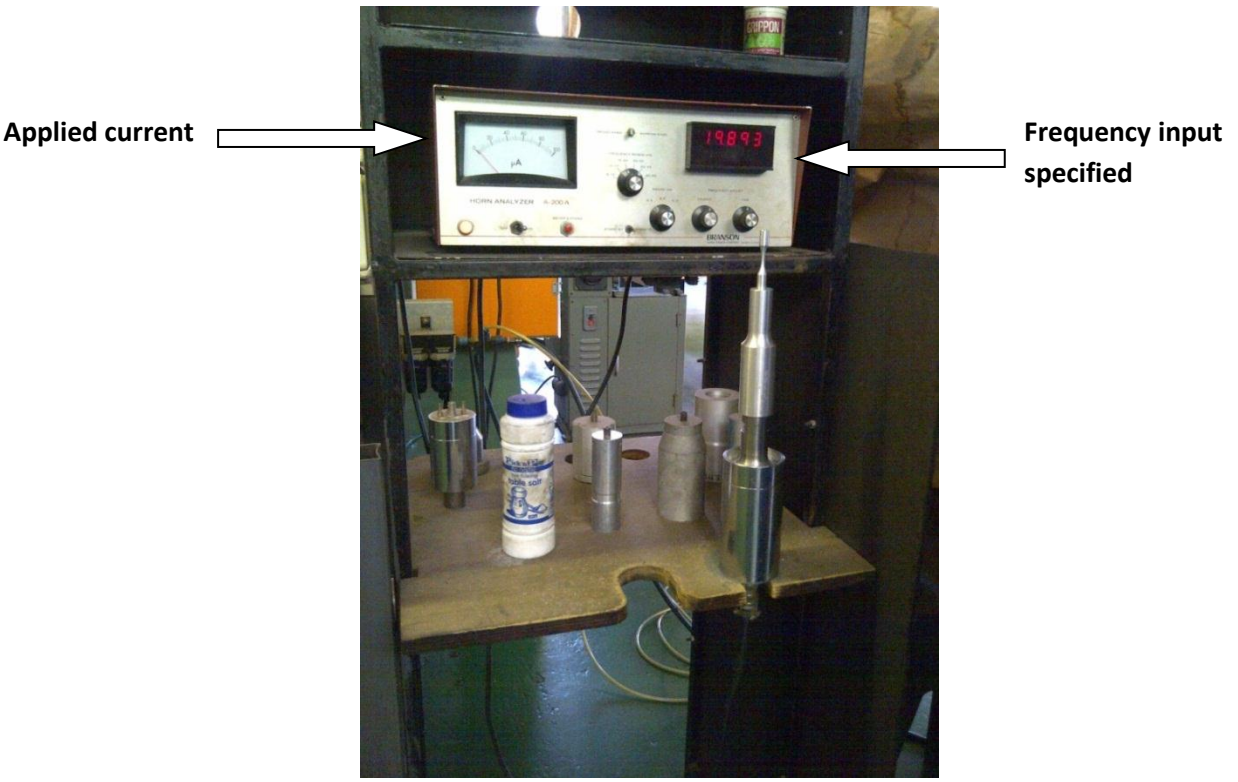


Figure 4-2: Branson A-200A horn analyser

The specimens were sized such that the maximum diameter at the throat of the dumbbell was 3mm.

The specimen for the Mark 1 test specimen had the following dimensions:

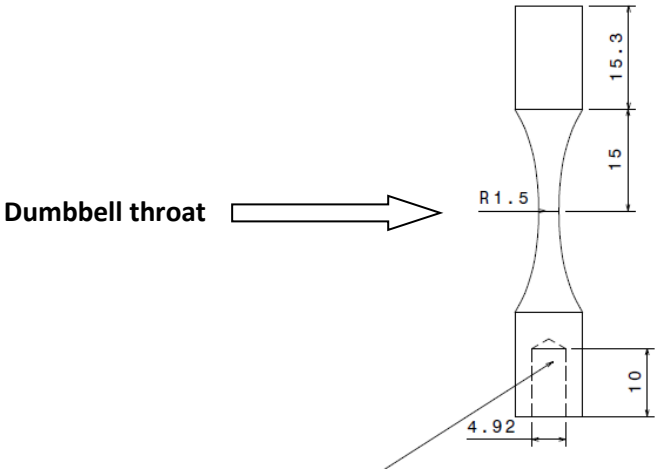


Figure 4-3: Mark 1 specimen sizing

FEM Models Comparison

To better understand the system FEM (Finite Element Method) models were constructed. Initially full 3D FEM models were implemented; however, these models were difficult to create and took a considerable time to execute. This is problematic for iterative optimisation of the horn and specimen. A much more efficient option was found to be that of the 2D axisymmetric FEM model. One of the concerns was the correlation between the 2D axisymmetric and that of the 3D FEM model. Subsequently a study of a simple cylinder was undertaken whereby the Matlab FEM code was compared with the 2D axisymmetric models of MSC Nastran and the 3D modelling techniques.

The comparison was compiled for a cylinder of radius 15mm and length of 100mm. The two models were defined as follows:

- The first model was that of a 2D axisymmetric model, consisting of linear triangular (Tria3 – see appendix C.4 for definition) elements of height and width equivalent to 0.5mm, see Figure 4-4. The total number of elements in this model was 12,000.

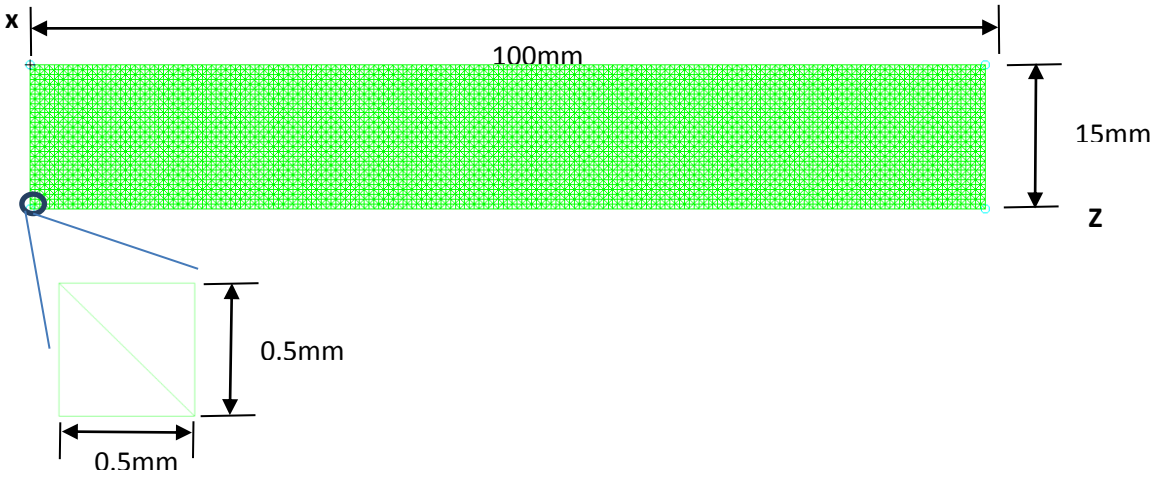


Figure 4-4: Axisymmetric Tria3 element model

- The second model was a Tet10 full three dimensional (3D) model, which was based on an element side length of 2mm. Thus the total number of elements in the model is 92,825.

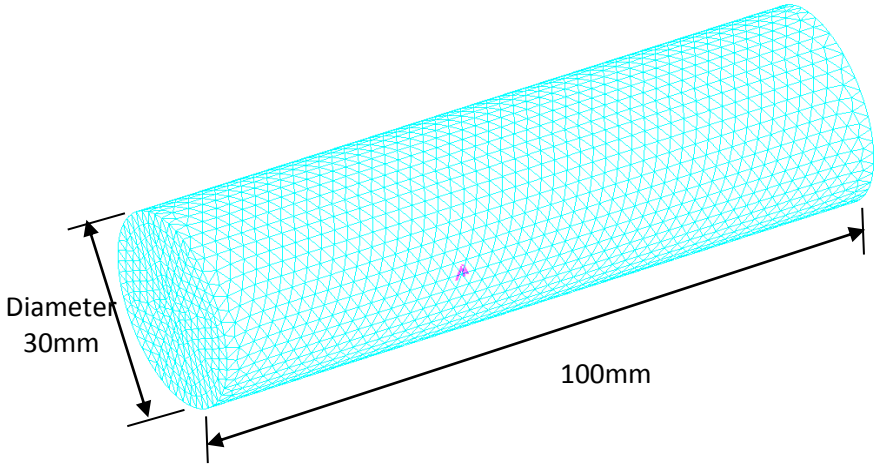


Figure 4-5: Tet10 element 3D model

The following modes shapes and respective frequencies were obtained for each model between 10,000 Hz and 35,000 Hz and are summarised in Table 4-1.

In the axisymmetric FEM model, only one mode was obtained in this range, at 25,202Hz. This mode was an axial excitation mode, as indicated in Figure 4-6. The natural frequency was obtained using the Matlab FEM code using full integration and compared to the 2D axisymmetric FEM model in Nastran. The Nastran model had a frequency of 25,204Hz. The Matlab code compared well with the Nastran axisymmetric FEM model, to be within 2 Hz of each other. The main reason for the creation of the Matlab FEM code is to allow for the specimens and horns to be designed without licensing requirements. This enables the UHCF system to be a self-sufficient unit, once the Matlab code is compiled to an executable.

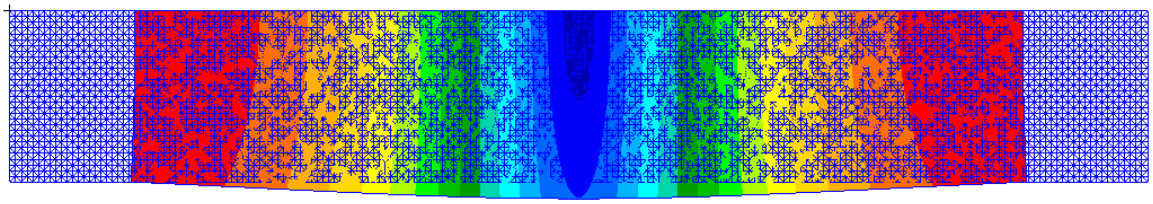
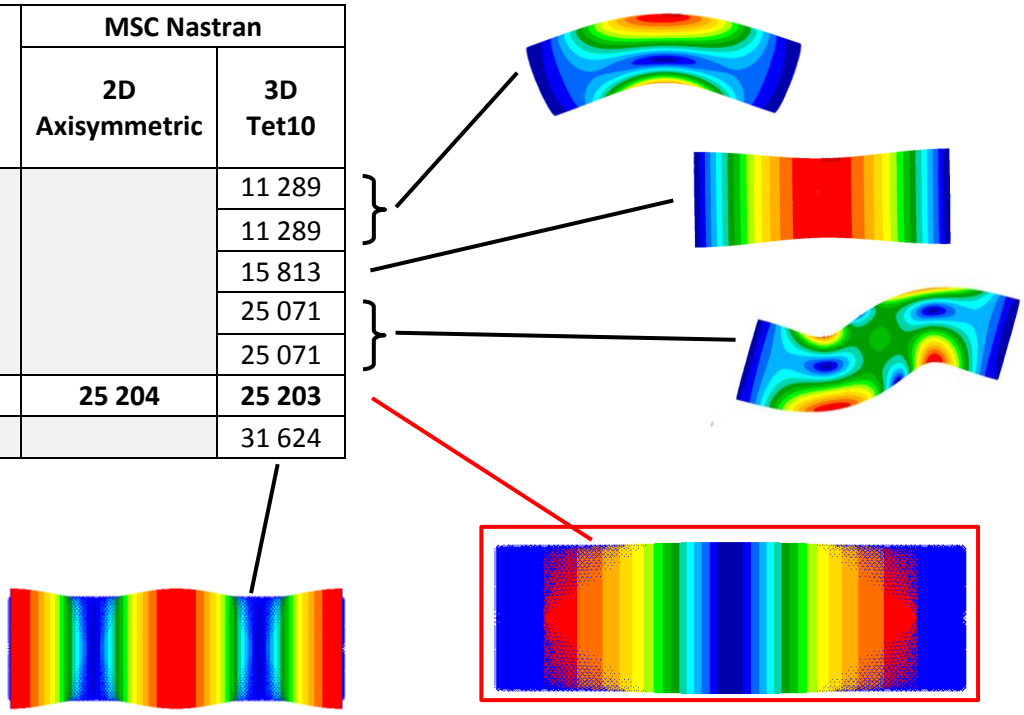


Figure 4-6: Mode shape for axisymmetric model

The 3D model indicated different frequencies within the selected range; however for the axial excitation the mode shape and frequency matched the axisymmetric FEM models.

Table 4-1: Tet10 frequencies (10 kHz to 35 kHz)

Mode Number	Matlab 2D Axisymmetric	MSC Nastran	
		2D Axisymmetric	3D Tet10
1			11 289
2			11 289
3			15 813
4			25 071
5			25 071
6	**25 202**	**25 204**	**25 203**
7			31 624



The seven mode shapes for the Tet10 3D model are indicated in Table 4-1, with the corresponding axial displacement mode highlighted in red to show the direct comparison to the 2D axisymmetric modelling technique.

It can be seen, that for the same mode shape, the frequency only differs by 1 Hz between the 2D and 3D methods. Of course this is for a very fine axisymmetric mesh size of 0.5mm by 0.5mm. Consequently a study was undertaken to explore the effect of increasing the axisymmetric mesh size to 2mm by 2mm. The obtained frequency for the larger element size was 25,199Hz (Nastran) and 25,199Hz (Matlab) in the range of 10,000 to 35,000 Hz.

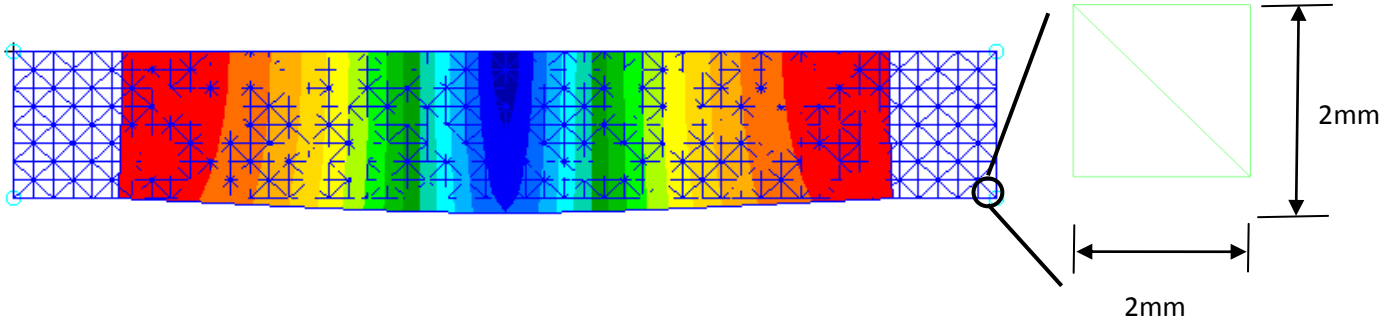


Figure 4-7: Mode shape and element size for 2mm square axisymmetric element

It was discovered that as the element size was increased, a noticeable decrease in the frequency of 4.4Hz occurred. This is a 0.02% error on the Tet10 model's frequency. However the time saved in

computation and complexity makes the slight error acceptable. Therefore it was decided that the Matlab FEM code created would only consider axisymmetric elements in the calculation of the frequencies of the horn and specimens. The main reason for this is that the Tet10 model (3D modelling element) would take vastly higher computation times to complete as compared to the axisymmetric model.

The Tet10 model is useful in defining which other non-axial modes are relatively near to the operating frequency region of 20 kHz. This is helpful in eliminating these unwanted modes which are near the excitation frequency of 20 kHz through design modifications. Therefore the specimen and horn should be initially sized using the axisymmetric 2D FEM model. After which a final full 3D FEM model should be created (if available) to validate that all other non-axial modes are sufficiently far away so as to not be excited at the 20 kHz frequency input.

The axisymmetric element is 2D in nature, thus the Matlab code started out by first considering the modal analysis solution of typical Quad4, Quad8 and Tria3 elements implementing 2D plane stress and strain, as defined analytically in Appendix C. Using this as a base, the solution of the 2D axisymmetric element is a simple manipulation of the tria3 and quad4 codes with subtle integration changes. The axisymmetric code generation and methodology is indicated in Appendix C.5 and C.6.

Three integration methods were implemented namely the full, reduced and area integration. These were validated against MSC Nastran for only the TRIA3 (TRIA3X as defined by Nastran) element type for one, four and eight elements in combination. Unfortunately, the QUAD4 element type is not supported in Nastran for axisymmetric modelling. Thus this element has not been validated. As a result the Quad4 element is not used for the sizing of the horns and specimens. The modal frequency comparisons showed a high degree of correlation for the full integration method with that of MSC Nastran. This is indicated in Table 4-2 and Table 4-3 for one triangle element.

Table 4-2: Lumped mass frequency comparison for one TRIA3 element model

Solver	**Nastran**	**Matlab**		
Integration method	**Standard**	**Full**	**Reduced**	**Area**
Mode # 1	0	0	0	0
Mode # 2	34187	34162	0	0
Mode # 3	46439	44554	34531	55210
Mode # 4	71474	71017	66382	189700
Mode # 5	110022	107540	78473	254550
Mode # 6	140776	134884	131351	979720

Similarly the results one element using a coupled mass matrix is indicated in Table 4-3.

Table 4-3: Coupled mass frequency comparison for one TRIA3 element model

Solver	Nastran	Matlab		
Integration method	Standard	Full	Reduced	Area
Mode # 1	0	0	0	0
Mode # 2	64813	64323	-	0
Mode # 3	91378	87750	-	9050
Mode # 4	120162	117099	118010	29410
Mode # 5	152849	154899	-	50470
Mode # 6	310705	306293	-	176820

The comparison was also completed for 4 and 8 TRIA3 element combinations, see Appendix C.7. Overall the FEM models indicated that the Full integration method provides the best data as compared to that of Nastran. Hence this integration method will be used for all indicated results in this project and be used for the horn and specimen sizing. The reduced integration method had some complications with the inversion of the matrices. Thus some of the modes had infinite frequencies. These frequencies are indicated with a dash in the tables. Due to this effect, the reduced method was not considered for sizing of the specimen and horn. The area method had good correlation for certain modes however other had significant deviations. Therefore this method is used only for reference. Throughout the design phase's further validation was completed for each horn and specimen design to improve confidence in the Matlab FEM code.

The horn and specimen for the Mark 1 were analysed using the 2D axisymmetric FEM code and a frequency of 19,935Hz was obtained. This mode was verified using MSC Nastran to have the exact same frequency. The axial mode shape in the range of 10,000 Hz to 30,000 Hz is indicated in Figure 4-8. The frequency obtained from the FEM models correlated well with the manually resized horn and specimen combination as indicated in Figure 4-2 which estimated a frequency of 19,893Hz.

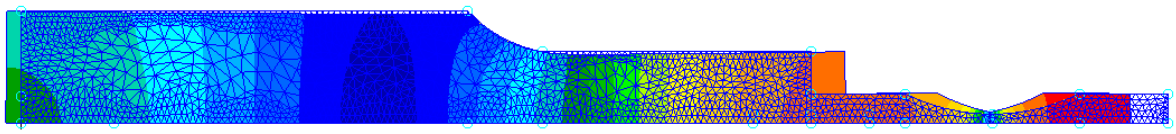


Figure 4-8: Mark 1 horn and specimen combination axial mode

Validation of Branson System

The Mark 1 system indicated that the Branson power supply and converter would be sufficient to generate an ultra-sonic fatigue failure in a correctly designed specimen and horn stack. If the diameter of the specimen centre is small enough then the crack is capable of growing through the entire test section of the specimen causing an abrupt failure. This is due to the fact that the crack growth has no significant time to affect the stiffness of the stack. Consequently the resonant frequency remains in the operating range of 20,000±50Hz of the Branson power source for the duration of the test. If the frequency of the stack exceeds the operational range then the power source is required to give more current to the piezoelectric converter. This may cause damage and

hence the Branson system automatically shuts down to save the converter. Conversely if the specimen diameter is large enough which would allow the crack to grow slowly through the section. Then the crack can affect the stiffness and hence the resonant frequency by more than 50Hz. If this is the case then the test will be forced to stop due to the converter and power supply limitations. If this happens it would then require a residual strength test or tension test to break the remaining area of the specimen not affected by the crack growth, thereby revealing the failure surface.

Validation of Strain Gauge Data

The strain gauge system of the Mark 1 design consisted of homemade power supply, see Appendix D, and the use of a signal conditioning system received from Denel Dynamics. This signal conditioning system was initially design to be used to measure low cycling stresses. Thus it implemented a low pass filter. The low pass filter was removed and validated using Matlab tools, Appendix D, to produce the correct un-filtered data. The validation process involved passing a merged low and high frequency signal through the modelled circuitry in Matlab using the electronics toolbox. This indicated that the full signal was passed through the op-amp without any filtering of the signal if one capacitor was removed. This data acquisition system provided a lot of background noise which masked the strain gauge data and hence the system was not effective for the monitoring of the system. Accordingly for the Mark 2 design a second strain gauge condition system was implemented. In future studies a simple battery may prove to be the most cost effective option for the system instead of the personal computer power supply.

Effects of Strain Gauges on Resonant Frequency

During the initial setup a concern was what effect the strain gauges would have on the resonant frequency of the system. As a result the horns were scanned before and after the strain gauges were applied. The following natural frequencies were obtained using the Branson Horn scan software, indicated by Table 4-4.

Table 4-4: Frequency Comparison

Test Number	Frequency of Horn	
	***With* Strain Gauges [Hz]**	***Without* Strain Gauges [Hz]**
1	20 071	20 089
2	20 076	20 086
3	20 073	20 089
4	20 070	20 091
5	20 075	20 085
6	20 068	20 087
mean	**20 072**	**20 088**

During the scanning of the horns, for the resonant frequency, it was noted that there was a scatter in the modal frequencies obtained. The presented data of Table 4-4 are six consecutive tests to obtain a statistical understanding of the error in the modal frequency measured. A typical Branson horn scan output is indicated in Figure 4-9 for one test scan without and with strain gauges.

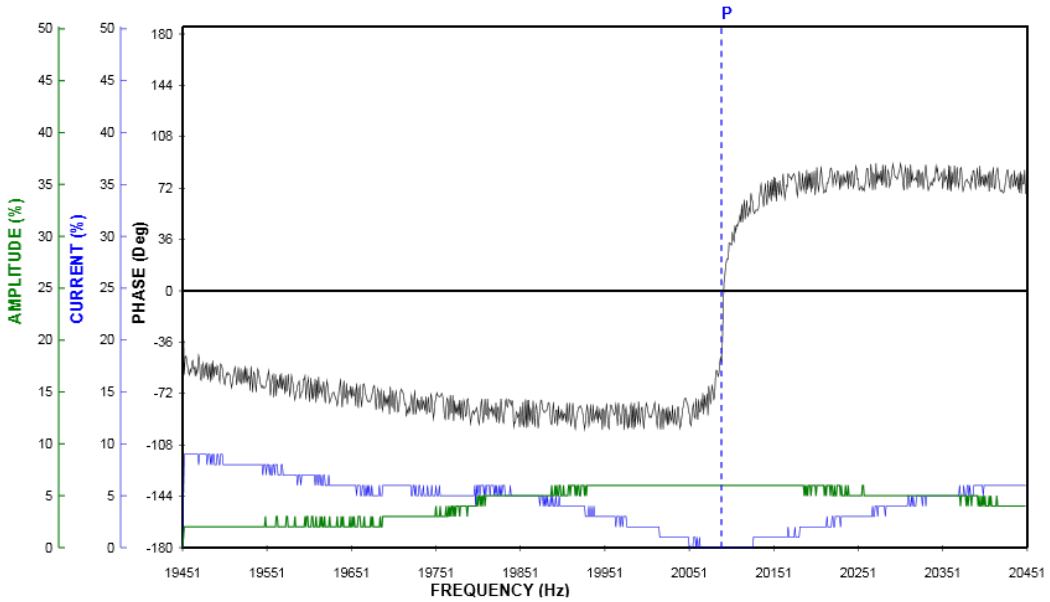


Figure 4-9: Typical horn natural frequency before gauges are applied

The results of these tests indicated that the resonance frequency of the horn was 20,088 Hz prior to the application of the strain gauges. After the application of the strain gauges the frequency was only 16 Hz different at 20,072Hz. Thus the effect of the gauges had a negligible effect on the system frequency. It was noted that the horn scans had significant variability, as a result, the mean of six recorded values for each configuration were considered to give more clarity. Another noticeable fact was that the frequencies increased without the strain gauges. This indicates that the mass of the strain gauge had a greater effect on the frequency than the additional stiffness added by the bonding of the strain gauge to the surface.

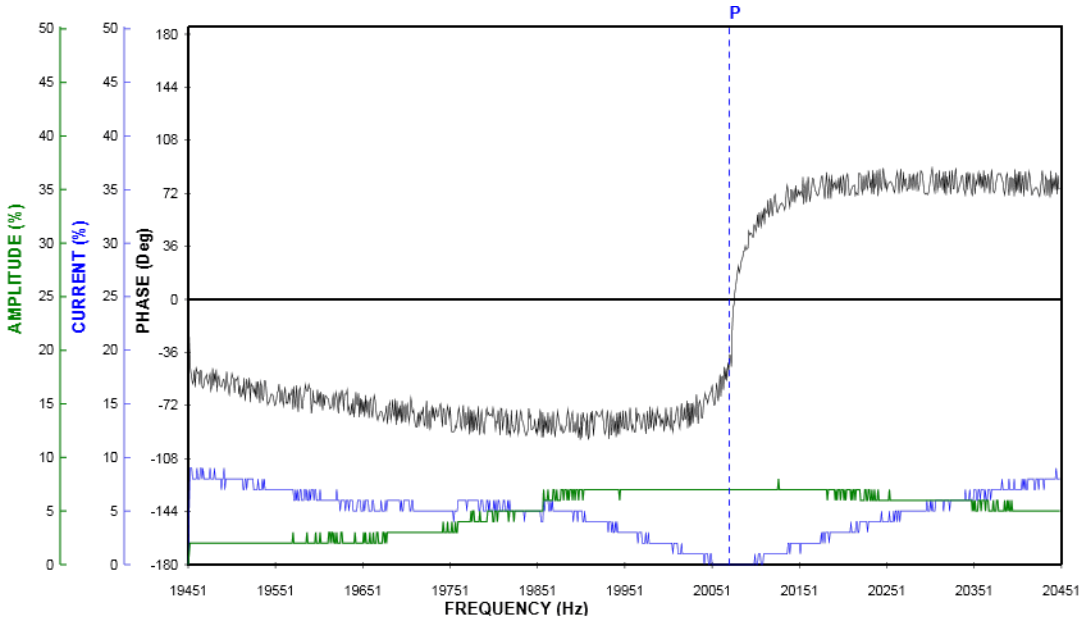


Figure 4-10: Typical horn natural frequency after gauges are applied

Estimate of Cycles to Failure and Corresponding Stress

Initially it was hoped that there would be six specimens to allow for a good validation via correlation with existing data. However, due to manufacturing difficulty only one point was available. The single

proof of concept specimen was recorded using a stopwatch to fail at approximately 363.5 seconds at 100% power output. It was estimated that the error in reaction time of using the stop watch was conservatively taken as 0.2 seconds. The time can be used directly with the known frequency of 20 kHz to obtain a total number of cycles of test to failure. Thus a total number of cycles of $7.27 \times 10^6 \pm 4{,}000$ cycles was obtained. The variation of the cycles cannot be meaningfully viewed in the semi-log plot of the SN curve for cycles exceeding 10^6 and hence the error bars are not plotted.

Too determine the stress on the SN curve there are two possibilities to identify where this test point should lie for the known cycles:

- Use the known cycles with an applicable SN curve from literature to obtain the stress.
- Use the known maximum output displacement of the converter to obtain the calculated stress using the Matlab code and the analytical wave equation.

The second method provides a better approximation of whether the Mark 1 is providing valid data. Thus the stress was calculated assuming a maximum input of 18µm as defined in the Branson specification for 100% performance operation. The analytical stress at the specimen centre was estimated to be 103MPa. The analytical solution is indicated in Figure 4-11, which depicts the horn and specimen geometry in the first subplot and the displacement together with the stress in the subsequent subplots respectively.

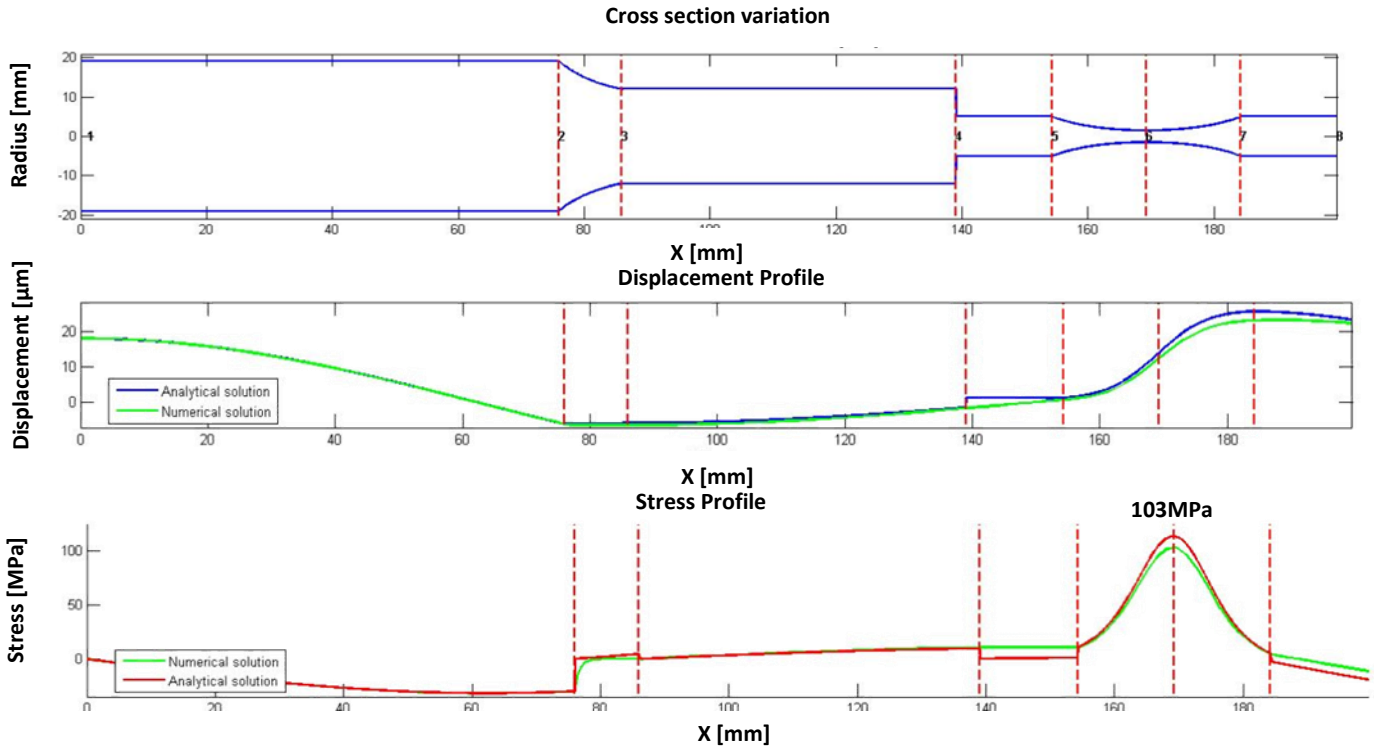


Figure 4-11: Analytical solution for proof of concept test

4.1.3 Results

The proof of concept test indicated that the life of the specimen would be approximately $7.0x10^6$ cycles for a stress of 103 MPa. This was plotted against the known data of section 2.4.1 in Figure 4-12.

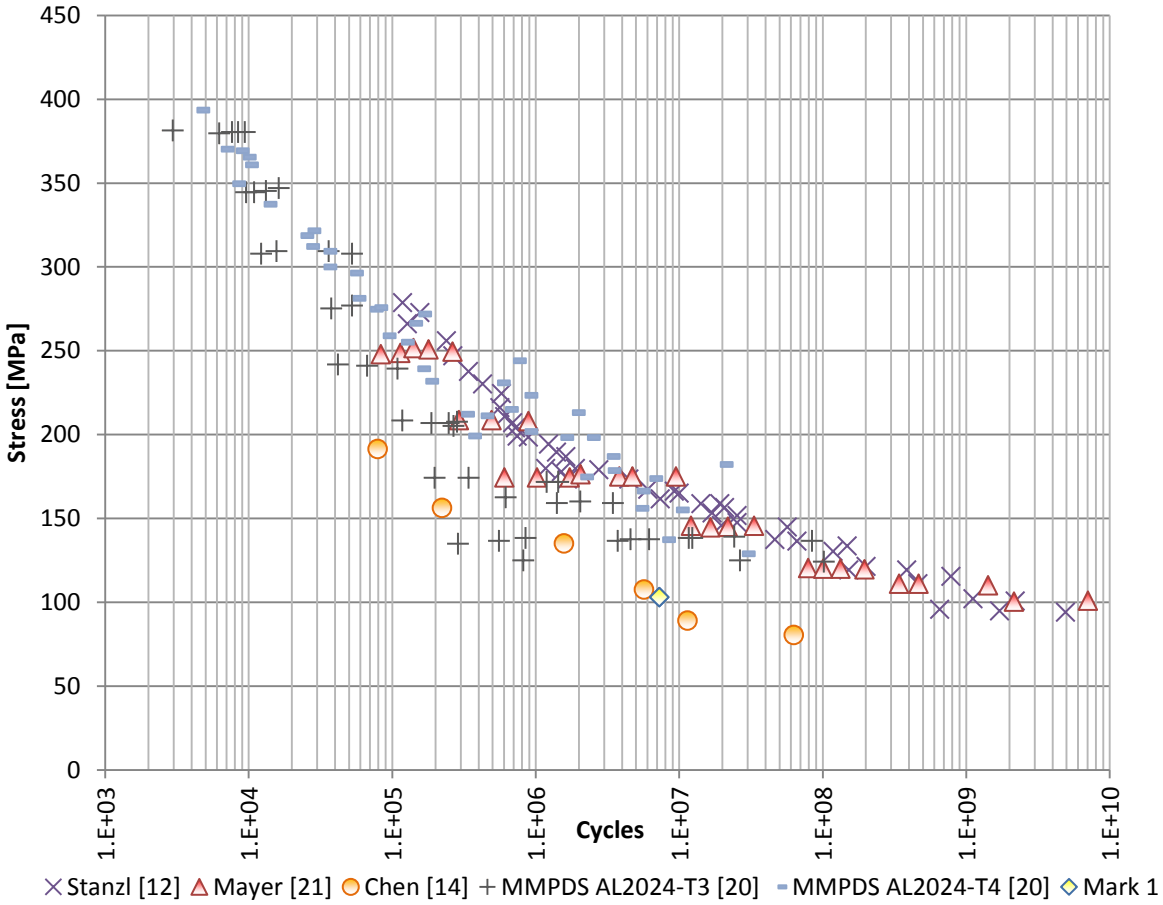


Figure 4-12: Proof of concept comparison to known data

4.1.4 Discussion

This Mark 1 design identified a few fundamental problems that were corrected for the Mark 2 design. These problems were:

- The throat diameter of 3mm, as indicated in Figure 4-3, was problematic for manufacture as during turning there was a high probability that the section would fail. Out of the 6 specimens specified for production only one specimen made it to testing. Even for the specimens that did not fail there was a concern that the specimens could be pre-fatigued during manufacture. This could occur during turning as tool loading may induce cracks or damage within the specimen. Consequently the diameter at the centre of the specimen was increased for the Mark 2 design.
- The step from the horn to the specimen was problematic for the computational analysis of the wave equation as described in section 2.1.4. This was due to the fact that the ODE solver within Matlab is only sufficient for the solution of smooth continuous functions, whereas the step creates a discontinuity. A solution was created that implemented two ODE solvers, one

for the horn and one for the specimen. The boundary conditions at the interface were matched assuming the displacement and the strain at the interface were equivalent. This modification of the code is indicated in Appendix G.3; however for the Mark 2 the step will be removed.

- The testing of the specimen in the horizontal position was not ideal due to the fact that as the specimen began developing the crack, in the centre, it also experienced some bending under the weight of the dumbbell. This is not appropriate for a pure tension-compression fatigue tests. Thus all future tests for the Mark 2 will be accomplished in the vertical orientation.
- The use of steel grub screws to connect the horn to the specimen significantly affected the frequency of the stack. Due to the density and stiffness differences of the grub screw relative to that of the specimen. Thus it was decided that the connection method for the specimen should be created from the same material as that of the specimen itself for future designs. The grub screw also added an additional element of complexity and inconsistency to the system. As the screw could easily loosen during testing from either side and create a false reading or stop the test.

The measured data point conformed favourably with the data of Chen *et al.* [14] thus validating that this hardware could provide the necessary data for a UHCF test system. Further tests were however necessary to ensure the statistical viability of the test.

The test also showed that a suitable method of counting the cycles more accurately than a stopwatch was needed to improve the accuracy of the system together with a reliable displacement/strain measurement system. Consequently the Mark 2 was designed to improve on the current starting point of the Mark 1 design.

4.2 Mark 2 – Initial Rig Design

4.2.1 Concept

The development of the Mark 2 design was an evolution on the Mark 1 discussed previously. The improvements implemented followed directly from the problems identified in the Mark 1 testing together with further application improvements to meet the PRS.

4.2.2 Design

Rig

This design was the first design to be incorporated in the modular test rig. The design of the rig was completed with the PRS in mind and the design drawings are indicated in Appendix E. The complete rig setup is indicated in **Figure** 4-13 below.

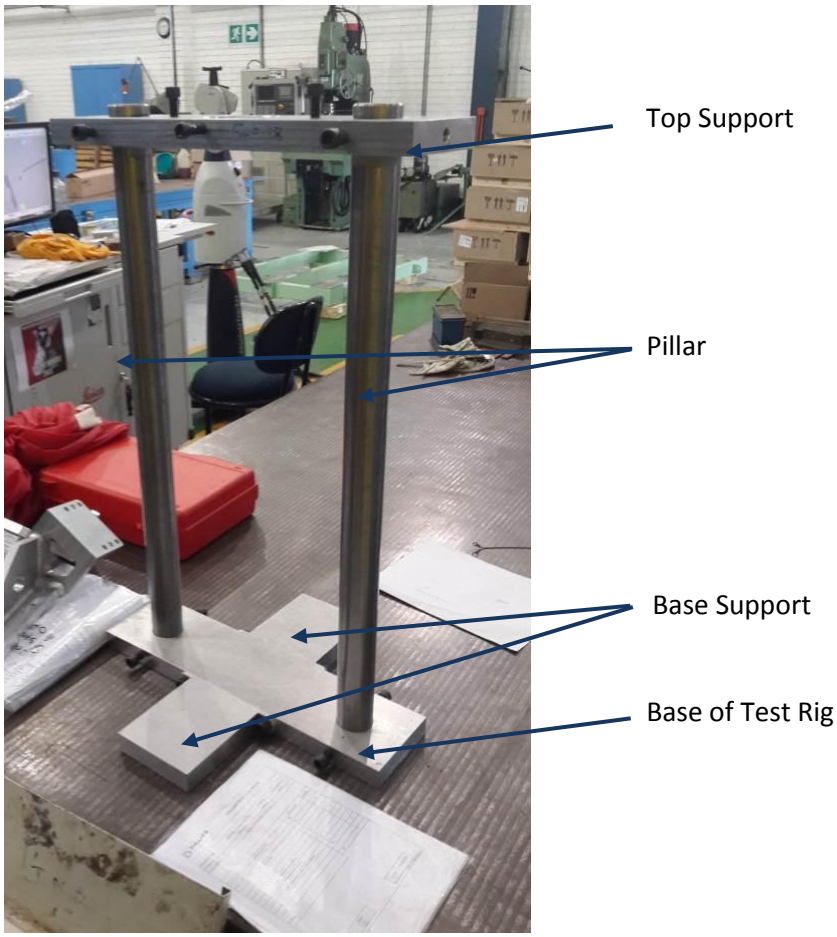


Figure 4-13: Test rig construction

The rig was created from mild steel to keep costs to a minimum and allow for more testing from the remaining budget. The top support is manufactured from Aluminium to make it light to allow for easy sliding on the pillars. The rig was not sprayed so as to minimise cost and thus the bare material colour is maintained with corrosion resistant oil applied. No chamfers or radii were specified on the part to minimise manufacture, complexity, time and cost. The lack of chamfers does give the rig a

slightly less aesthetically pleasing look but this is offset by the savings in manufacturing. The rig's foot print was design to be less that 500mm x 500mm. This allows for the rig to be placed on most commercially available tables.

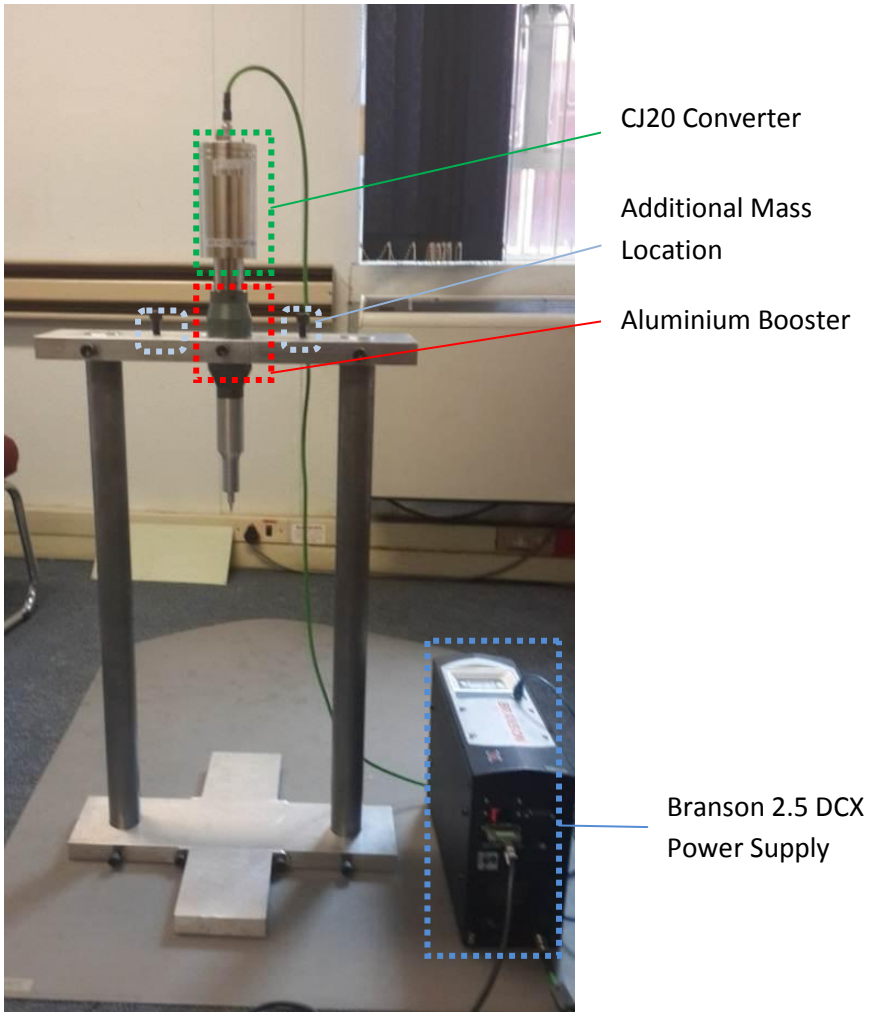


Figure 4-14: Test rig combined with the actuating system and power source

The design philosophy used in the creation of the rig was to make the rig as secure and as stable, as possible, while allowing for possible alterations in the future for different tests. This may include:

- Changing the mean stress loading to obtain R ratios other than -1. This will be achieved by loading the top support while the horn will need to be mirrored on the other side of the specimen and be bolted directly to the base. The specimen can then be loaded with a constant tension or compression and then the 20 kHz UHCF load can be applied. A loading mechanism will need to be designed for this set of testing.
- Allowing for the combination of normal fatigue loads and UHCF. This can be achieved by incorporating the rig into a conventional fatigue testing machine to allow for the testing of the combination of UHCF and HCF. As for the various R ratios setup, the horn will need to be mirrored about the specimen and fixed to the base. A grip will need to be created to allow the top support to be connected to one of

the Instron Fatigue machines at DAe. The vertical pillars will need to be oiled to allow for a smooth sliding action to occur as per the sliding fit specified for the top support.

Threaded holes were created in the upper support to allow for the addition of masses in the event that the natural frequency of the rig is excited by the actuator. In this situation masses can be added to the test rig to modify the rig's natural frequencies. These holes are indicated in Figure 4-14 above. However, since the clamping of the rig is applied to the boosters nodal line (point of zero displacement) this should never happen, unless damage to booster has occurred. These holes may prove advantageous for future modifications of the rig or may even be used to compress the specimen for future compressive R ratio tests. During testing the Rig was checked using the laser vibrometer and found to experience minimal displacement.

Horn

The horn was designed by optimising a 2D axisymmetric model of element size 1mm by 1mm. The Matlab code developed was used and validated against the Nastran solution. The horn model was found, using the Matlab code, to have an operating resonant frequency of 19,980 Hz (19,980 Hz Nastran) when the dimensions of the final horn were implemented. The harmonic axial mode of the model is indicated in Figure 4-15 and the material properties used are as per section 2.4.2 for AL7075.

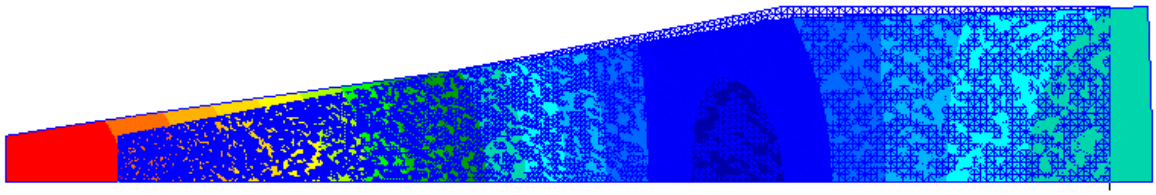


Figure 4-15: Axial resonance mode of the AL7075 horn design

Only one horn was manufactured for testing as the power supply will allow for variation of the input displacement which in turn will vary the final stress experienced by the specimen. There are 100 increments in power available on the power supply. This effectively allows for one hundred different stress variations per horn design.

The 2D FEM analysis was validated against a full 3D Tet10 element model consisting of 18,035 elements in MSC Nastran. The two methods showed good correlation with the frequency of the 3D FEM model being 19,978 Hz (a difference of a mere 2Hz). The 3D model is indicated in Figure 4-16.

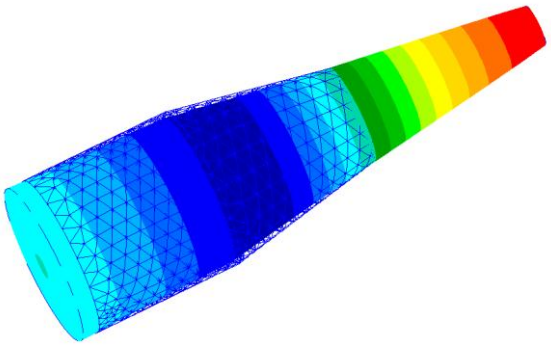
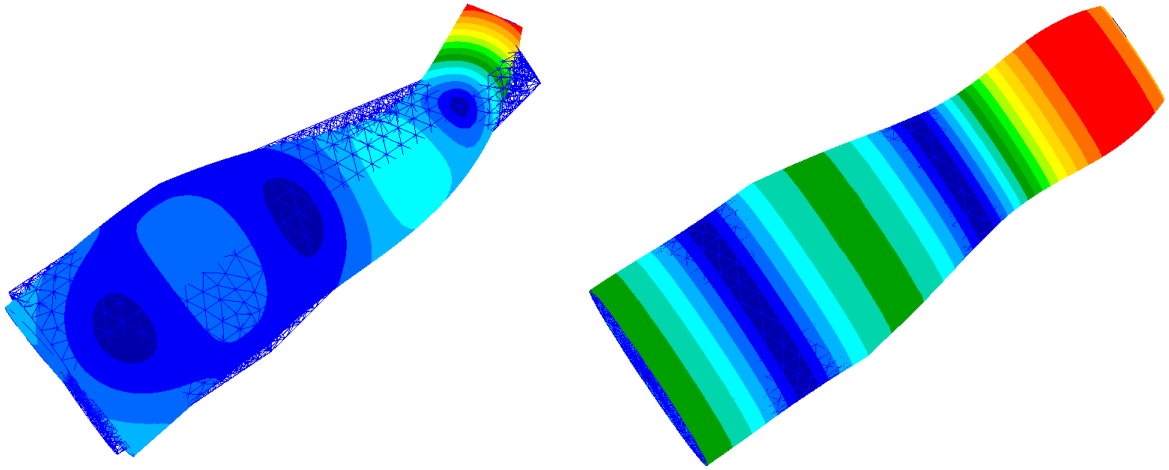


Figure 4-16: 3D FEM Tet10 horn model

The nearest modes to the axial mode were identified using the 3D model in MSC Nastran. The first mode below the axial mode was obtained at 16,424Hz which was a bending mode. The next highest mode is a circumference expansion mode at 23,646Hz. These modes are indicated in Figure 4-17.



Bending Mode – 16,424Hz **Expansion Mode – 23,646Hz**

Figure 4-17: Nearest non-axial modes

These modes are at least 3,500Hz away from the excitation frequency of 20,000Hz. This provides a safe operating frequency range for the current horn design. The reason for determining that the frequency separation is safe, is based on the consideration of the one degree of freedom amplification equation, given by:

$$A = \frac{1}{\sqrt{\left(1 - \left[\frac{f}{f_0}\right]^2\right)^2 - 4\alpha^2\left[\frac{f}{f_0}\right]^2}}$$

Where

f - is the external excitation frequency

f_0 - is the resonant frequency

α - is the damping assumed to be 2%, 4% and 6%.

Using this equation it can be shown that for the various damping ratios the amplification is significantly reduced at 3,500Hz from the 20,000 Hz peak. This variation is indicated in Figure 4-18.

Figure 4-18 indicates that for 3,500Hz either side (red vertical lines) of the resonant peak of 20,000Hz. The amplification is significantly less. Using the highest damping (6%) the knock down factor between the peak amplification and the amplification at 3500Hz off peak is 2.78 (8.33/2.99) as compared to 2% damping with a knock down of 8.03 (25.00/3.11). Typical aluminium damping normally lies around 2-4%. Thus these modes should be sufficiently separated from the operating frequency to not be excited during testing.

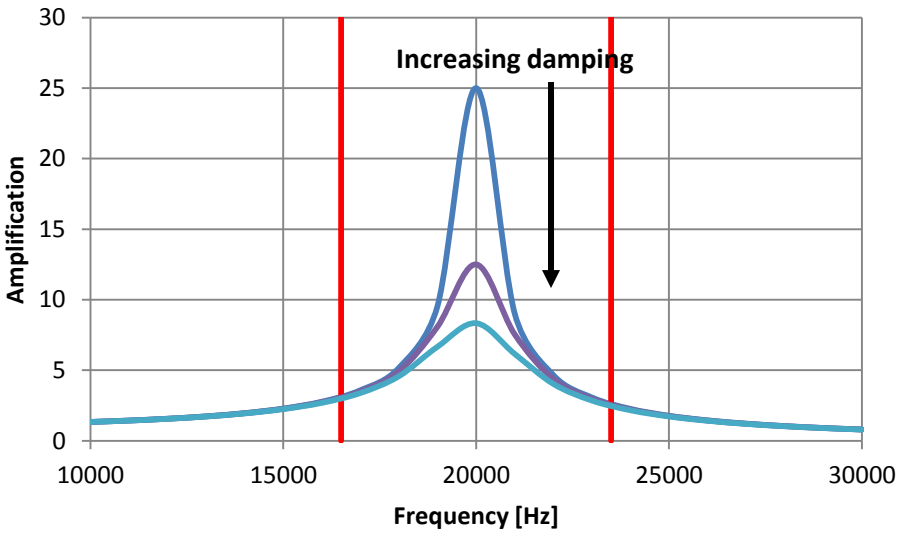


Figure 4-18: Amplification affects for one degree of freedom system

The displacement amplification factor of the horn alone is calculated using the Matlab code to be 3.01 and the displacement over its length and consequent strain for a 10 micrometre input displacement is given by Figure 4-19.

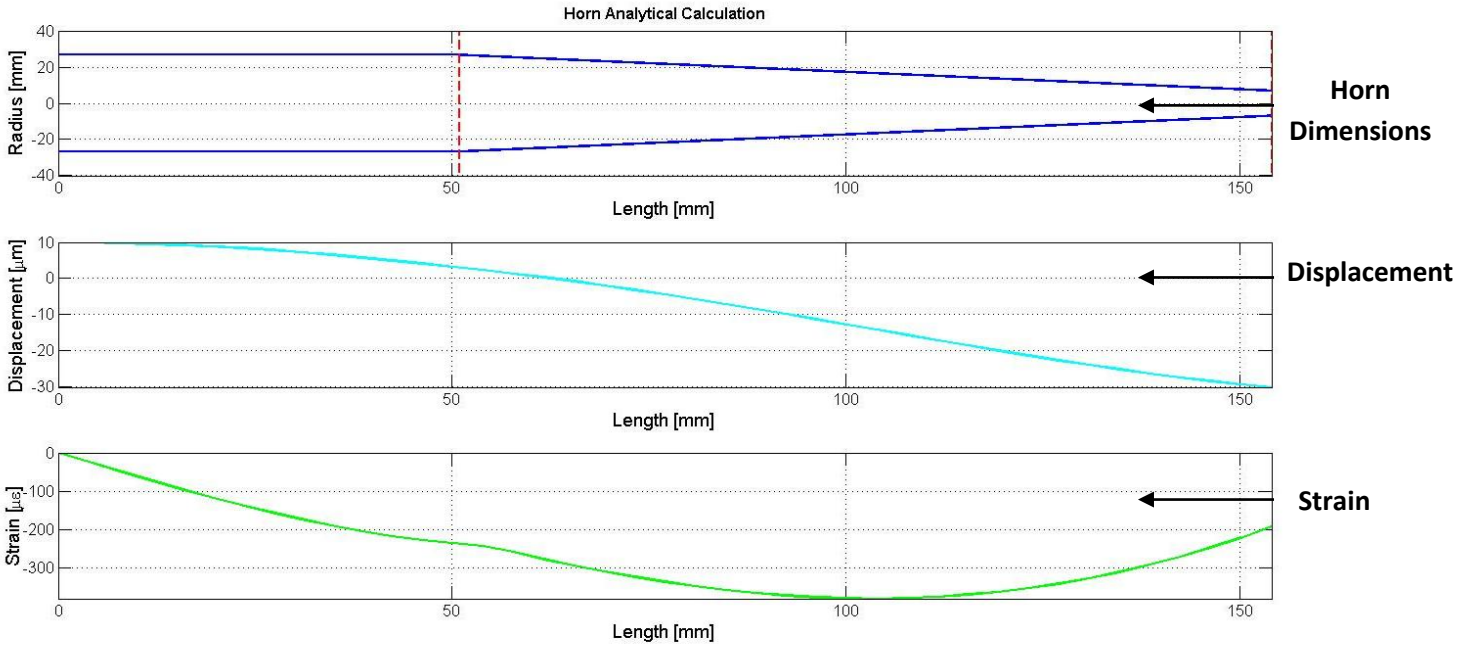
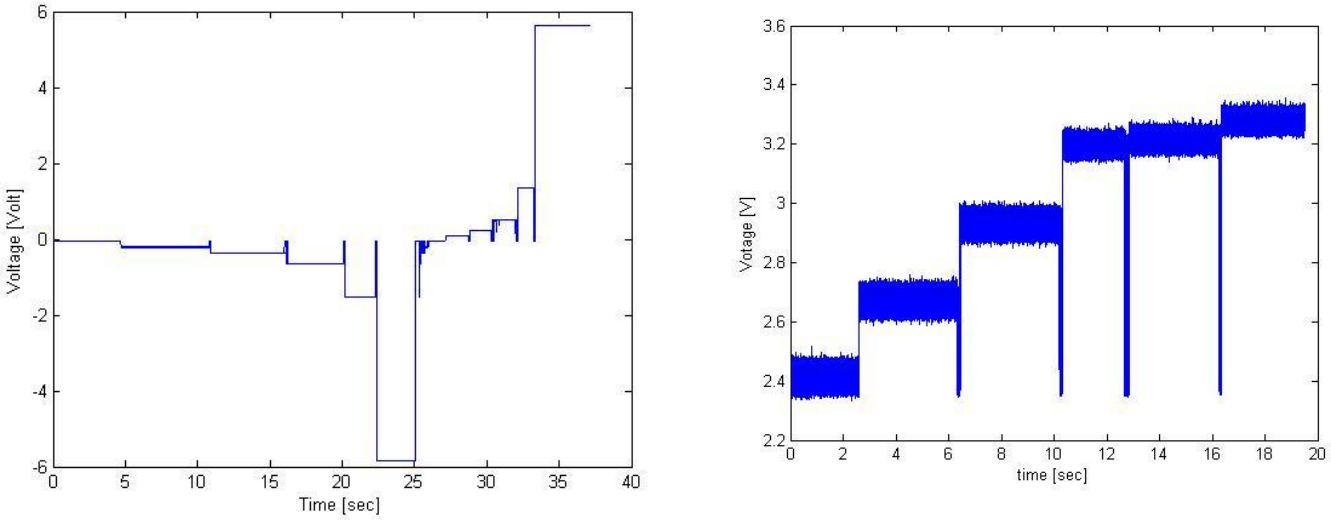


Figure 4-19: Analytical displacement and stress variation of horn length

Measurement System

In the Mark 2 design implemented a combination of strain gauges and a Polytec laser vibrometer for the measurement system. The laser was implemented to validate the strain gauges and to prove that the strain gauges can be used effectively on their own to monitor the system.

The strain gauge setup of the Mark 1 was replaced by a more sophisticated Topward TPS-4000 power supply which provided less noise in the signal. This is visually apparent from the comparison to the two power supplies respective output signals curves in Figure 4-20.



Mark 2 Power Supply **Mark 1 Power Supply**

Figure 4-20: Comparison to power generator noise output

The improvement in signal to noise ratio is clearly visible in Figure 4-20 which shows the output of the two different power supplies. Figure 4-20 indicates two different scales; this was due to the implementation of two different signal conditioning boards with different gains. The data was created using a shunt calibration system from HBM as discussed in B.2.2. The data indicates that for the same resistance change on the shunt system the voltage change recorded is different for the two signal conditioning systems with the Mark 2 data producing a more expected trend.

The Topward power supply for the Mark 2 design was used in conjunction with a signal condition board that allowed gains of approximately, 600. This is the final power generator and signal conditioning system used in the testing system as it provided the most stable data. This strain gauge system was implemented for the Mark 2 and validated against a laser vibrometer from the CSIR (Council for Scientific and Industrial Research).

A comparison of the stresses obtained during the Mark 2 testing, using the laser vibrometer and the strain gauge were correlated to indicate if there was a direct comparison, see Figure 4-21.

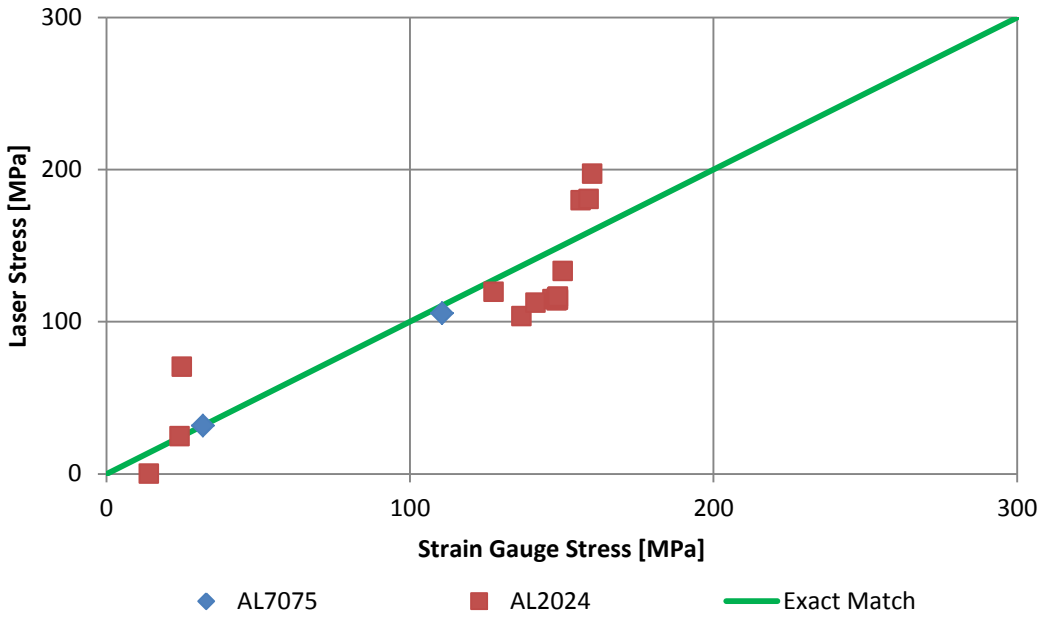


Figure 4-21: Comparison of strain gauges to laser

Overall there is a reasonable correlation between the stress measurements obtained using strain gauges and the laser vibrometer. The data obtained for the two AL7075 tests indicated an almost exact correlation. However the AL2024 data does not conform to any norms. The strain gauge measurement system had one fundamental problem, in that due to the high fatigue loads imposed on the gauges, they kept failing and needed repair on a constant basis, almost once per specimen. The failures varied from direct gauge damage to connecting wiring failing at the solder. This proved very time consuming and in some cases both gauges would fail during the same test. If not for the laser vibrometer the test would have needed to be stopped and the strain gauges replaced or fixed before continuation of the test could go ahead.

The laser vibrometer data also indicated some rather peculiar data spikes, see Figure 4-22. Similar data spikes were noted in the strain gauge data. There are two possibilities as to what these spikes represent. Either these spikes are simple data logging errors or they are real displacement spikes. If these spikes are indeed real, then this would mean that the strain gauges would see some high over stresses which could cause them to fail prematurely, however failure was not evident in testing. These spikes would also cause crack retardation which may improve fatigue life data. However, since these fatigue tests are only run till crack initiation this will not pose a problem in the final data. Figure 4-22 represents the direct voltage measurement as received from the Laser vibrometer with the scatter spikes indicated. These data spikes are believe to be ghosting in the data acquisition as some of the results indicated stress values sufficiently high enough to cause ultimate failure of the specimen. However this never occurred indicating that those spikes are not valid. Hence the cycles for these data spikes are counted and since there are only a few per test, their average has a negligible effect on the overall average maximum stress of the test.

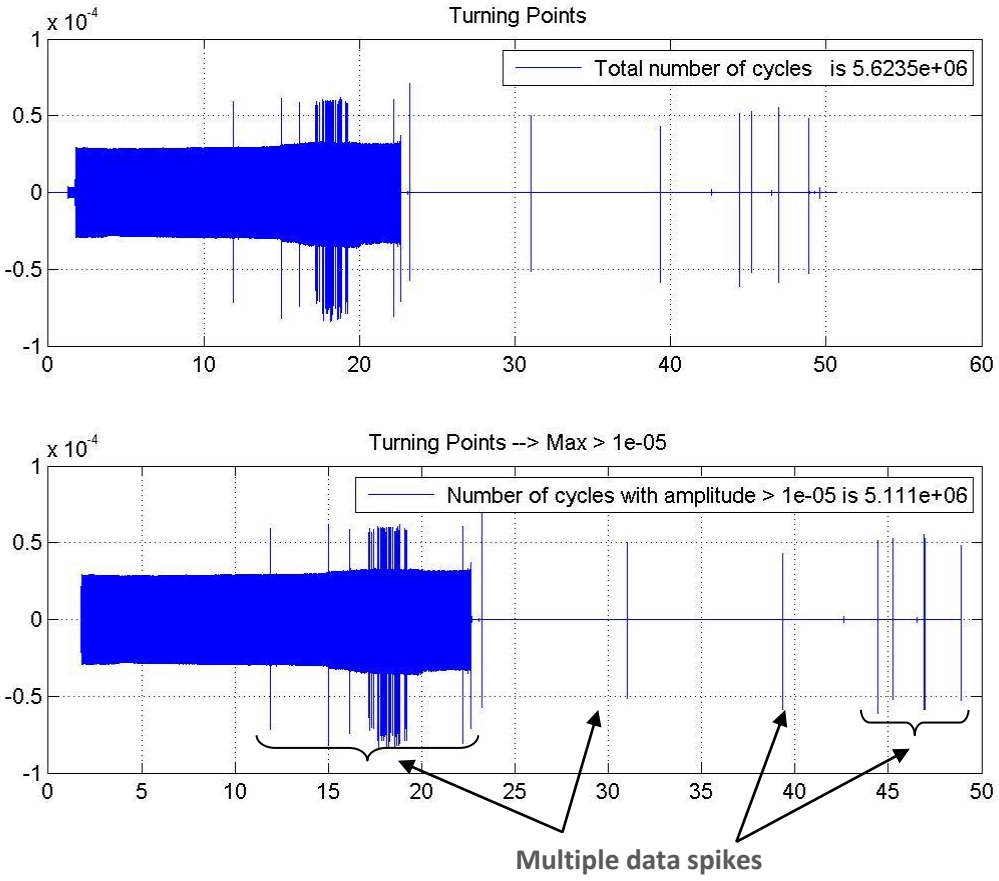
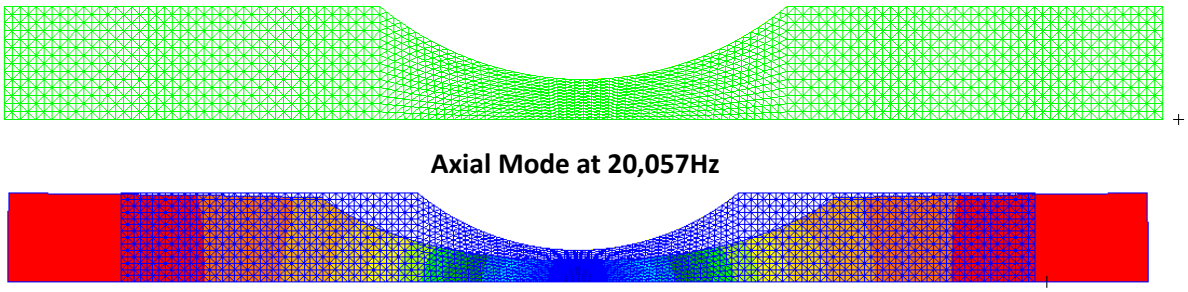


Figure 4-22: Laser vibrometer data for Al7075 horn and Al2024 test specimen 4

Specimen Design

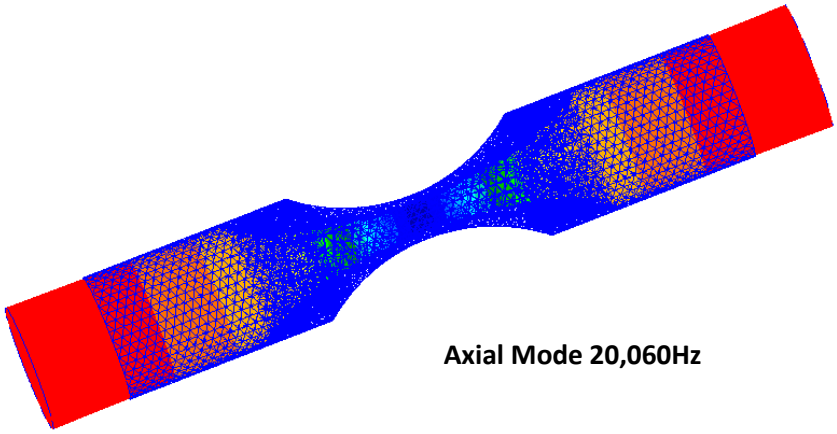
The design of the specimen followed the same procedure as that of the horn. Initially the specimen was sized using the Matlab axisymmetric FEM methods. The specimen was then verified against the MSC Nastran solution of both 2D and 3D models. The 2D mesh of the specimen was define to have a total of 4464 TRIA3X elements. The model and the first axial mode within the range of 10,000Hz to 30,000Hz was at a frequency of 20,057Hz (20,057Hz Nastran) and is indicated in Figure 4-23.



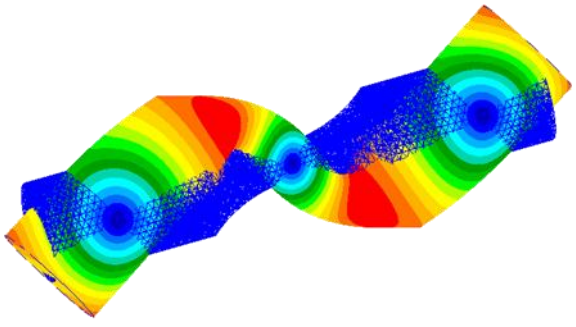
Axial Mode at 20,057Hz

Figure 4-23: 2D Specimen model and applicable mode in 10 kHz to 30 KHz range

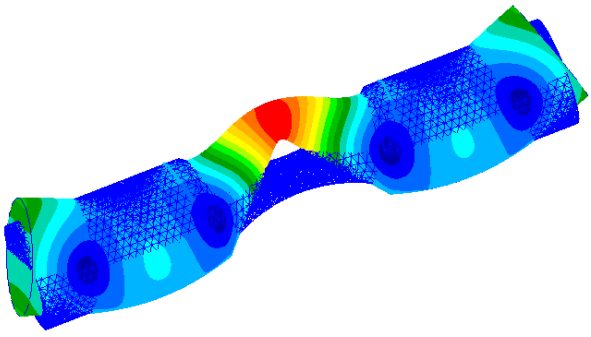
This model was verified against a 3D FEM model using Nastran and the axial mode is indicated in Figure 4-24. The axial mode frequency in the range of 10,000Hz to 30,000Hz was obtained at 20,060Hz for the 3D model. This axial frequency mode compared well with the 2D axisymmetric modelling technique further added to the confidence of the Matlab FEM code. The next nearest frequencies are bending modes, at 16,869Hz and 49,359Hz. As for the horn the separation of the nearest frequencies exceeds 3,000Hz. Thus the amplification of these modes at 20,000 Hz is low and even if a crack generates then the frequencies vary by ±50Hz the other modes will not be excited due to the significant frequency separation.



Axial Mode 20,060Hz



Axial Mode 16,869Hz



Axial Mode 49,359Hz

Figure 4-24: 3D Nastran FEM model

Specimen Stress Estimate

To estimate the stress experienced by the specimen a transfer function was created for both the laser and the strain gauge measurement systems. This transfer function is used to go directly from the voltage measured by the strain gauges or laser vibrometer to the stress at the centre of the specimen.

Laser Vibrometer

The laser vibrometer acquired data at 1 m/s/V. Thus the voltage received is directly converted to a velocity by simple multiplication:

$$v = x\,[V]\ \times 1\ \left[\frac{m}{s}/V\right] = x\,[\frac{m}{s}]$$

Note: units are in the square brackets with [m] being metres, [s] seconds and [V] volts.

Where

x- Is the input voltage read from the laser vibrometer,
v- Is the velocity at the end of the specimen which is sinusoidal in nature.

The maximum displacement is then obtained using the amplitude of the velocity $[v]$ as indicated by equation 35 in Appendix B.1:

$$d = \frac{v}{w} = \frac{v}{2\pi f} = \frac{v}{40000\pi}$$

This gives the displacement at the end of the specimen which can now be related to the stress at the centre of the specimen by iteratively solving the wave equation *1* until the displacement at the end matches what the laser is measuring as indicated in Figure 4-25. The solution of the second order differential wave equation is obtained using the ODE45 solver in Matlab. The ODE solver uses the displacement amplitude input at the start of the horn as a parameter which varies the stress calculated in the specimen. Therefore this input is varied using an iterative bracketing convergence method until the displacement at the end matches the lasers displacement. The code is indicated in Appendix section G.2. The typical displacement variation over the length of the stack is indicated in Figure 4-25.

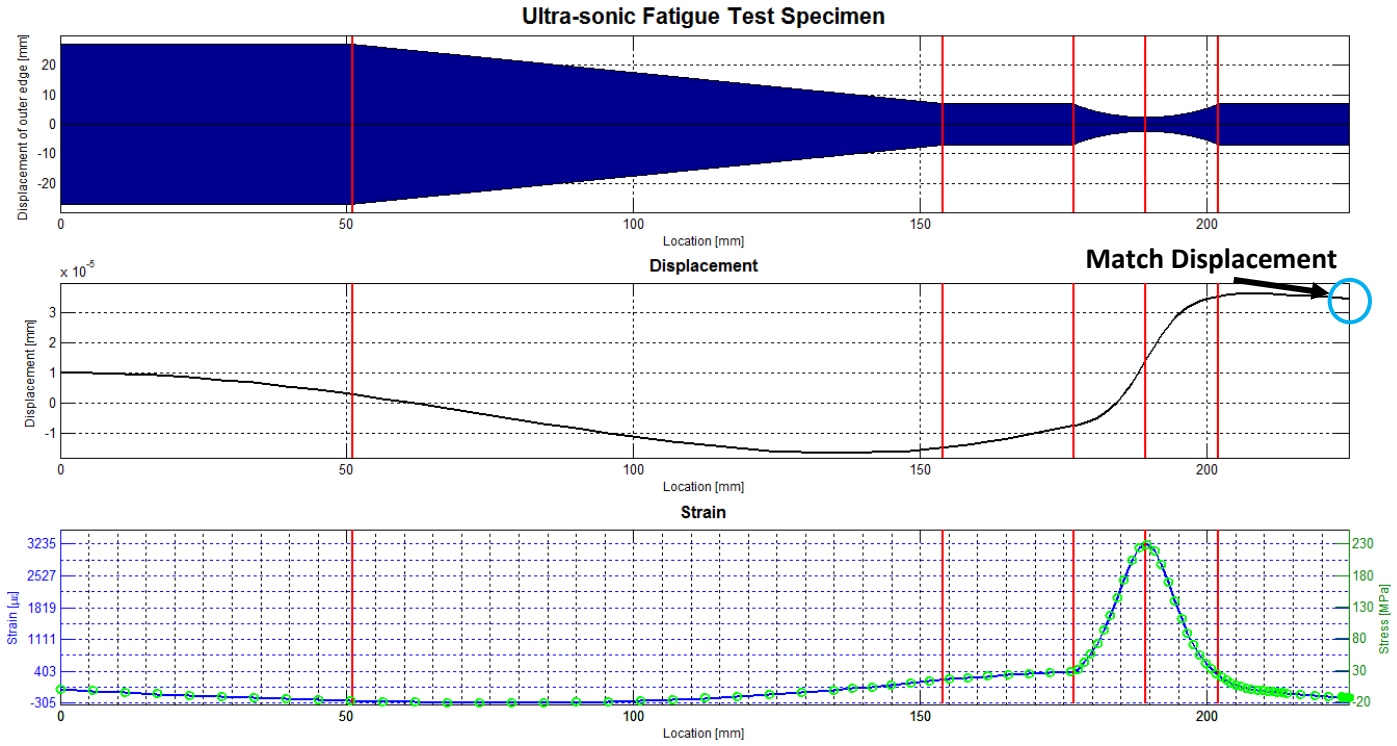


Figure 4-25: Typical displacement, strain and stress output

The Matlab code of Appendix G.3.1 and G.3.2 were used to create a calibration curve (transfer function) that directly takes the displacement at the end and outputs the resultant stress at the specimen centre. This transfer function is indicated in Figure 4-26. This was similarly done for AL7075, where the equation varied slightly as compared to AL2024 and was define as $y = 6.8239x$. It is noted that even though the distribution of the stress along the specimen is non-linear, a comparison of end displacement to throat stress has a direct linear relationship.

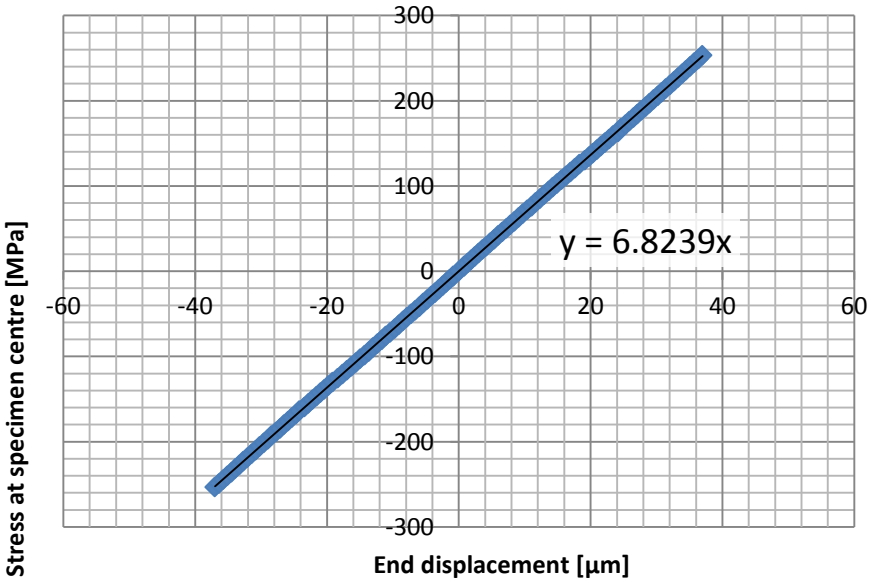


Figure 4-26: Laser vibrometer transfer function of AL2024

Strain Gauges

The strain gauges measure strain directly from a location 25mm from the start of the horn. Thus the same procedure is implemented as above however the convergence is done on the strain and not the displacement. Thus the point at which the strain is matched is indicated in Figure 4-27.

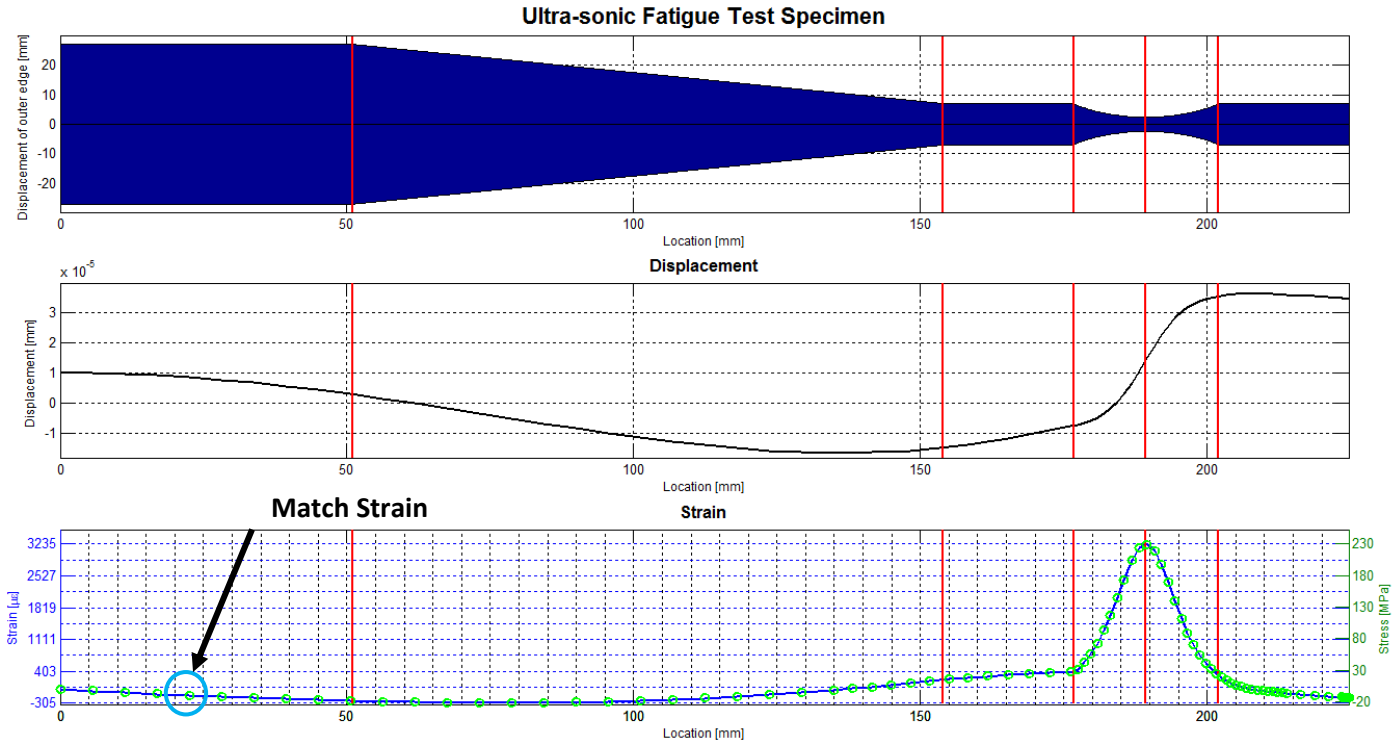


Figure 4-27: Typical displacement, strain and stress output

The strain gauges are located at 25mm from the edge of the horn. This measurement is indicated in Figure 4-28.

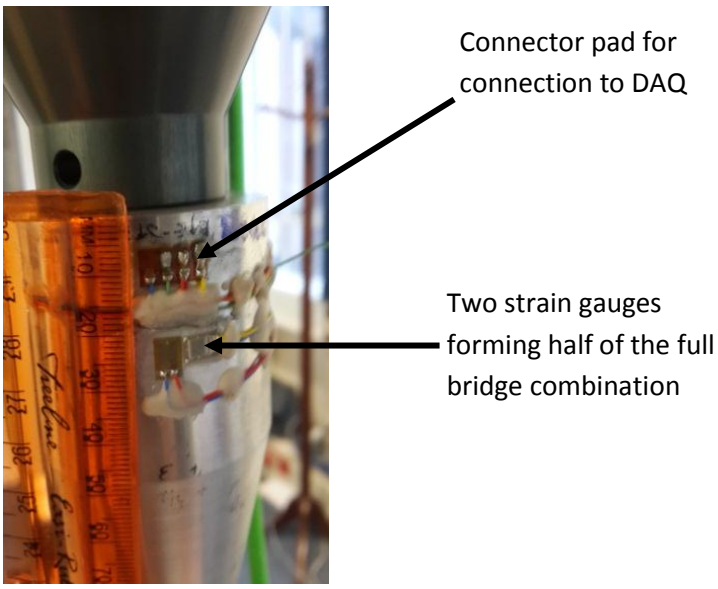


Figure 4-28: Strain gauge location on horn

Again the Matlab code of Appendix G.3.1 and G.3.2 was used to create a transfer function that takes the strain at 25mm from the start of the horn and outputs the resultant stress at the specimen centre. This transfer function is indicated in Figure 4-29. This was similarly done for AL7075 where the transfer function was defined as $y = -1.4388x$. Again it is noted that even though the distribution of the stress along the specimen is non-linear the point to point comparison has a direct linear relationship.

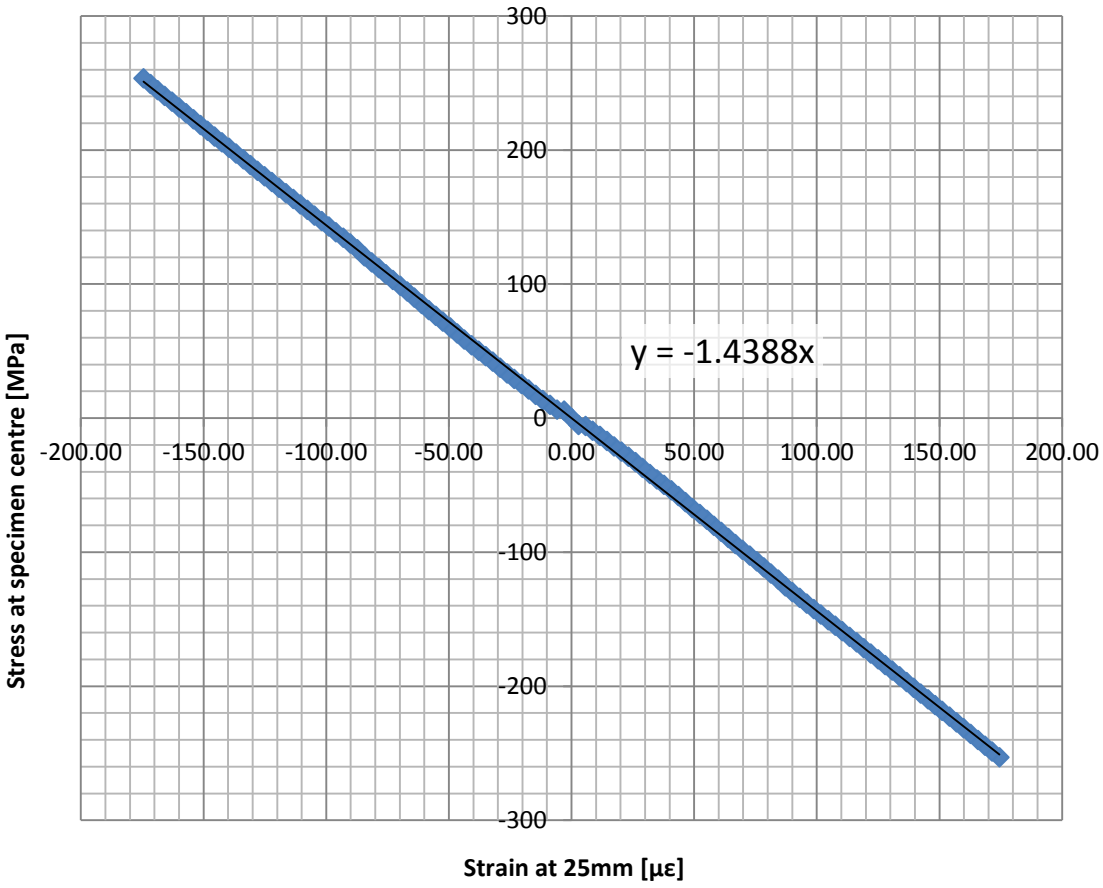


Figure 4-29: Strain gauge transfer function of AL2024

The negative sign is due to the setup of the strain gauge wiring. However since this application is based on an R ratio of -1 the effect of the maximum and minimum stress swop has no effect on the final result. This would need to be monitored more closely for other R ratios where tension and compression maximum and minimums need to be known specifically.

Verification of Rig Resonance

To verify that the rig resonance and vibration was sufficiently low to ensure that it did not affect the measurement from the laser vibrometer, three tests were carried out to measure the vibration during operation:

A. The first location that was assessed was the coupling ring on the booster
B. The second location that was assessed was the upper frame assembly
C. The third location that was assessed was the top of the right pillar

These three locations are indicated below with the displacement amplitudes as measured from the laser vibrometer described afterward. The arrows indicated the direction in which the displacement and velocity was measured. The power setting during the tests was set to 70% on the Branson power generator.

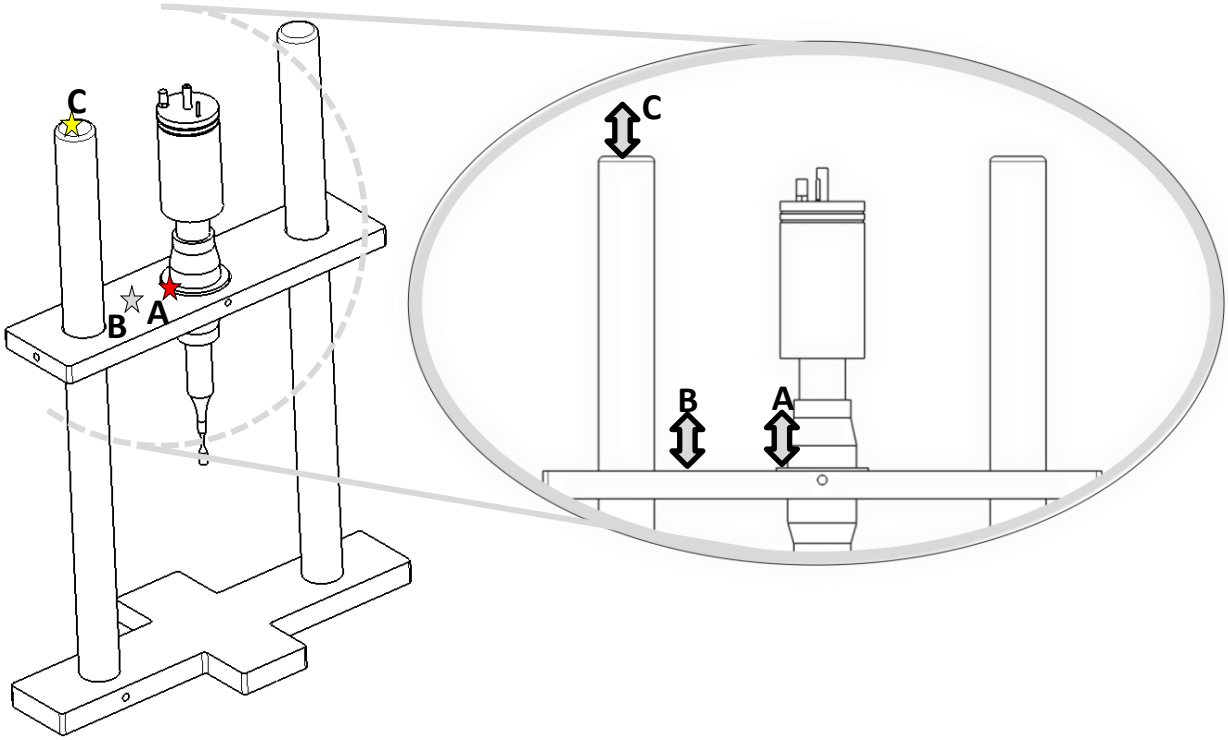


Figure 4-30: Position where vibration of Rig was investigated

The measurements at location A on the booster coupling (bush), resulted in the velocity measurement shown in Figure 4-31 which when integrated indicated maximum displacement amplitude of 1.20µm. If equation 35 in Appendix B.1 is implemented then the displacement measured is calculated to be 2.05µm. This displacement is expected since the nodal line is a single line about the circumference of zero displacement. However the coupling is a mechanical seal which is spread over an approximate longitudal length of 10mm which means it is not on the nodal line alone. This displacement has no affect on the measurement as it is part of the stack displacement as shown in Figure 2-2 and is expected to have a low displacement. The plot of velocity and displacement for the coupling is indicated in Figure 4-31. The green velocity data is filtered data using a 10^{th} order Butterworth highpass filter with the passband starting at 16,000Hz. This data is then integrated using the cumtrapz Matlab function to obtain the displacement with the amplitude being compared to that of equation 35 in Appendix B.1.

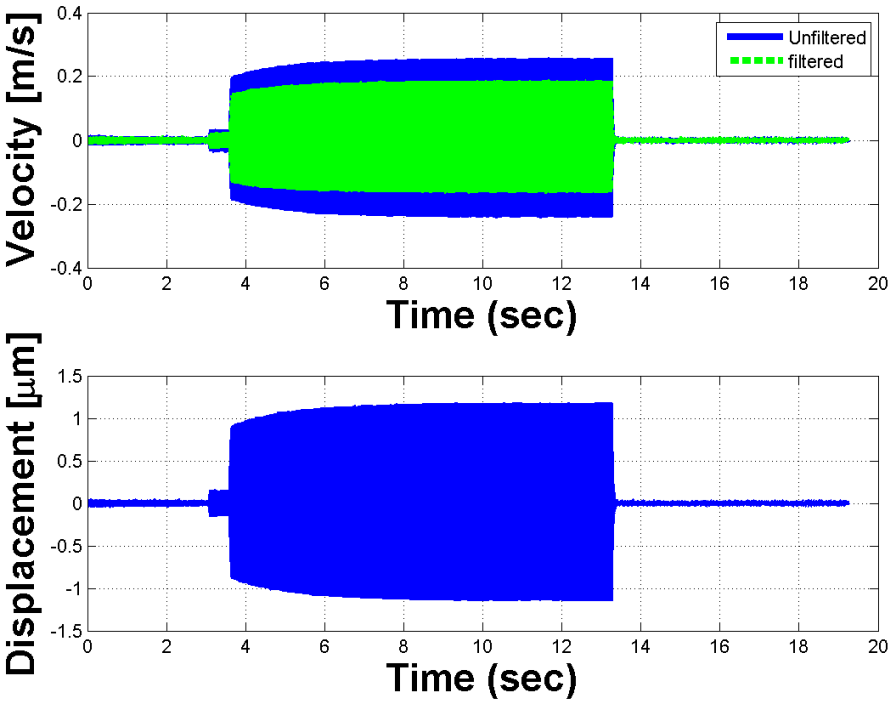


Figure 4-31: Booster coupling displacement

A velocity plot for location B on the top support is shown in Figure 4-32. This indicates that even though the coupling which is in direct contact with the rig experienced displacement, the displacement is not transmitted and amplified through the rig as the displacement measure on the top support had a maximum value of only 0.05µm.

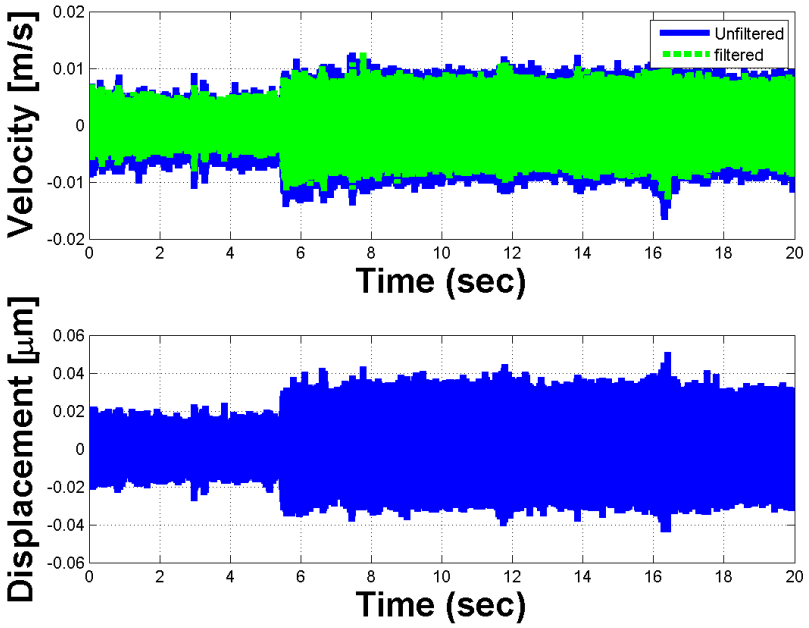


Figure 4-32: Top support displacement

The pillar measurement indicated a similarly low displacement with the maximum displacement measured, shown in Figure 4-33 being only 0.08µm.

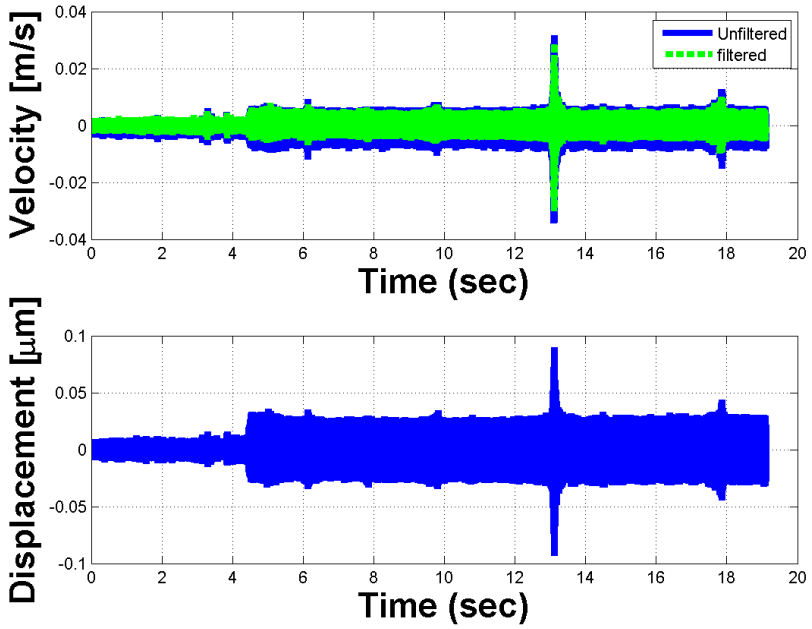


Figure 4-33: Right pillar displacement

Overall it is evident that since the general displacement measured on the specimen is in the range of 24µm for 70% input power then the displacement of the support stand is only about 0.08µm. This is a percentage error of less than 0.33%. Consequently the rig vibration should have negligible effect of the final measured displacement. This correlates well with the modal analysis of the rig which indicates the first bending mode of the top support is as approximately 8 Hz which is well below the operating frequency of 20 kHz.

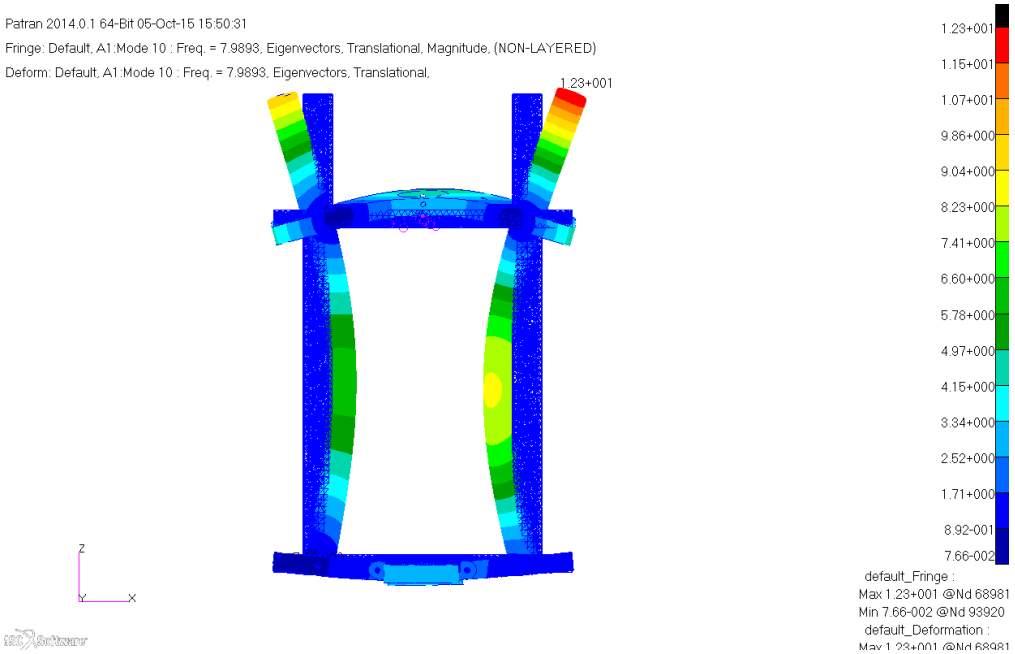


Figure 4-34: First bending mode of top support

4.2.3 Results

The results of the specimens tested for both AL2024 and Al7075 are indicated in Figure 4-35 and Figure 4-36 respectively.

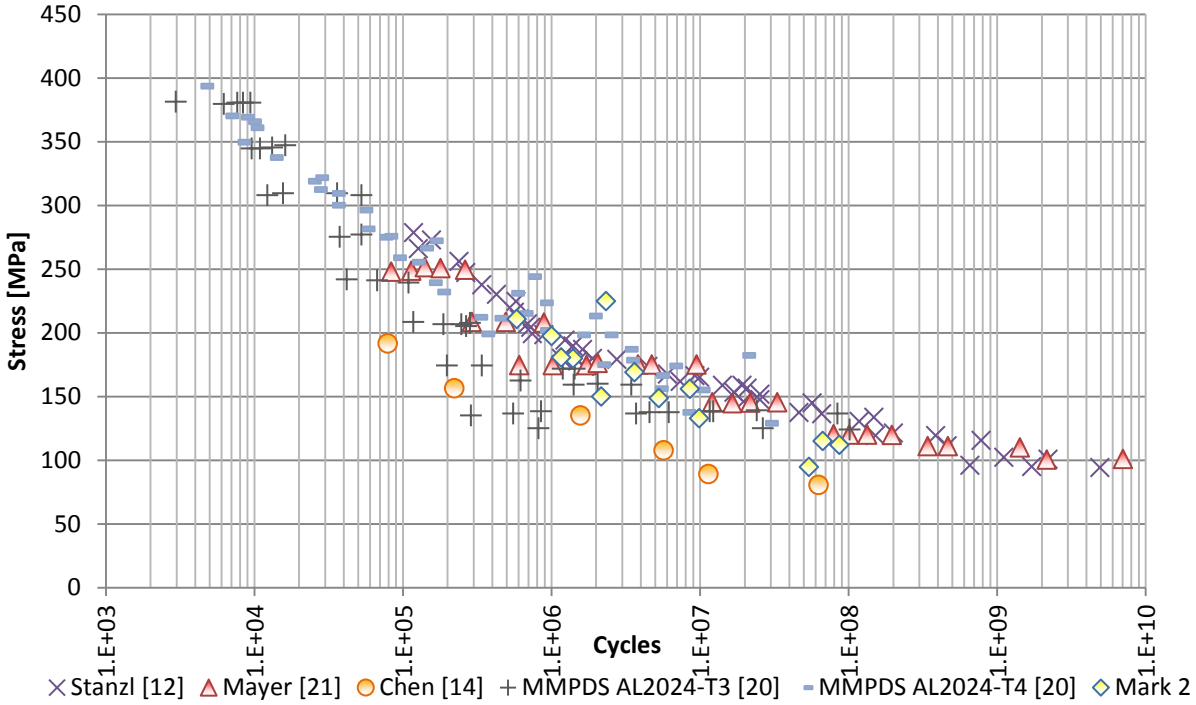


Figure 4-35: Mark 2 AL2024-T351

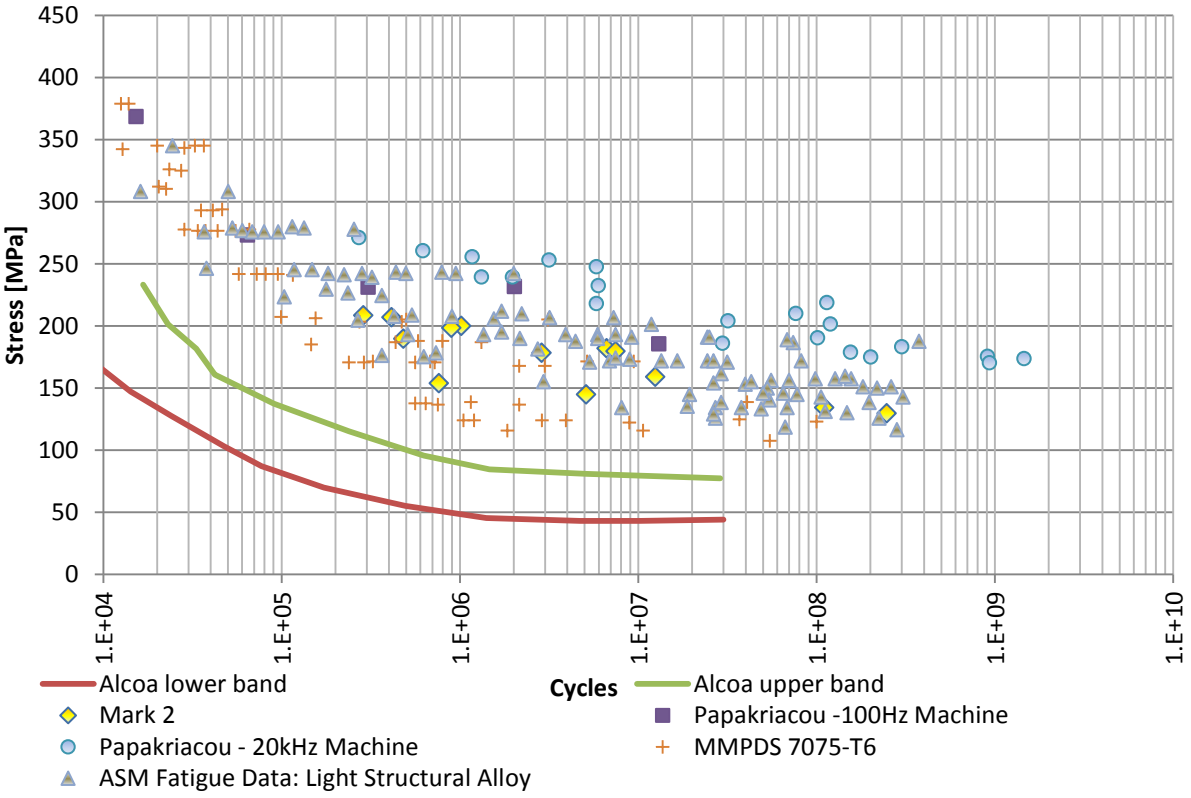


Figure 4-36: Mark 2 AL7075-T7351

4.2.4 Discussion

The Mark 2 was a vast improvement on the Mark 1 and the data proved it. However, the Mark 2 did have a significant flaw in that the strain gauge measurement system data was not as consistent when compared to the laser vibrometer. One of the main reasons for this was that where the strain gauges are implemented, they are measuring a small strain on the horn as indicated in Figure 4-27. This is not the best location. The better location would be at the higher strain location on the centre of the specimen. This has two inherent problems: firstly the diameter is so small the implementation of the gauge on the section is not feasible; the second is that due to the high strains the life expectancy of the gauge will be significantly reduced in the high strain area to the brink that the gauge may need to be replaced multiple times during each single test. Even in the current gauge location the connecting wires and even the gauges themselves had continual failure.

The data obtained from the gauges did however correlate well with the laser vibrometer data when the stresses were within a manageable range, exceeding 10^6 cycles. Thus the gauges could last for a suitable period of time if the test where at a low enough stress level. Overall the gauges work. However they are unreliable and cannot be expected to work for the duration of a single test and even the redundancy measures of implemented more than one set of strain gauges does not rectify this.

Consequently the main improvement to be made for the Mark 3 will be the use of only laser vibrometer data for the measurements.

Another aspect noted was that the cracks that initiated on the surface of the Al2024 specimens seemed to following the surface finish marks created during turning. This is indicated in Figure 4-37. Consequently an investigation into the effect of the surface finish will provide valuable information and was completed during the Mark 3 testing.

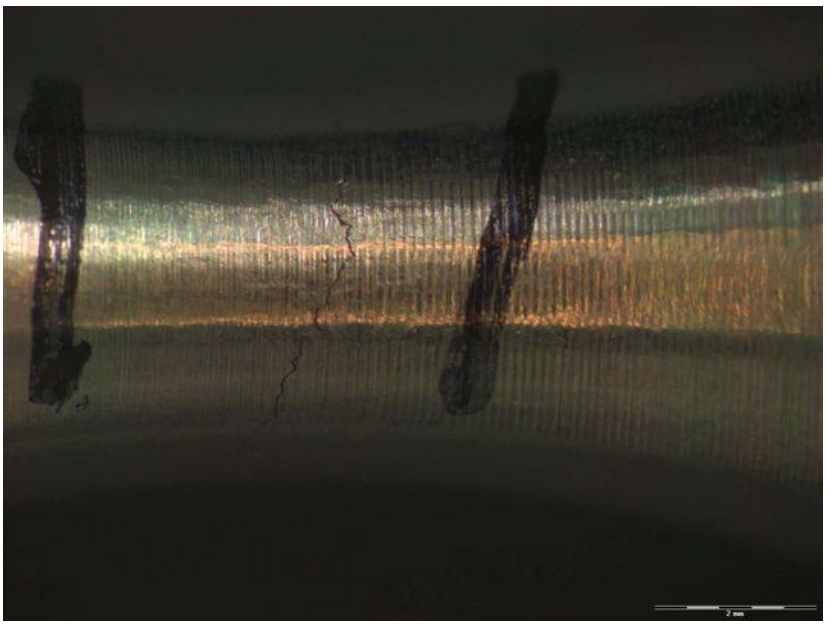


Figure 4-37: AL2024 specimen micrographic image of surface crack

4.3 Mark 3 – Final Design

4.3.1 Concept

The concept of the Mark 3 UHCF testing platform is to obtain data using the final stable configuration. This implies that only the laser vibrometer will be used for the acquisition of the data while the strain gauges will no longer be implemented.

4.3.2 Design

In the final design the strain gauge measurement system was removed and only data from the laser vibrometer was implemented. The specimens were also polished to improve the surface finish from 3.2µm to 1.6 µm to investigate the effects of the surface finish on the overall results.

Thus the following test setup was implemented:

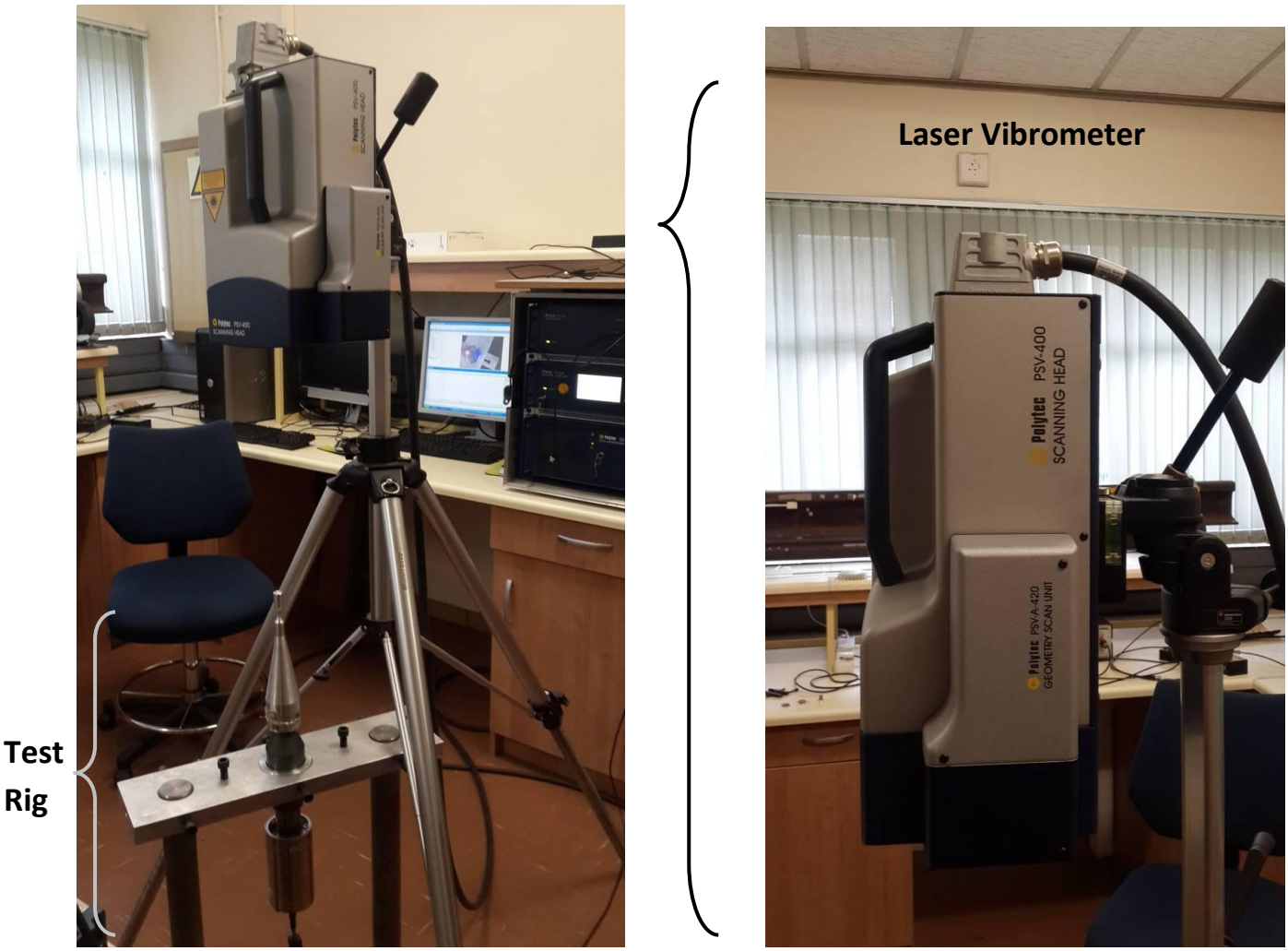


Figure 4-38: Mark 3 final setup

The initial rig setup is run upside down as this allows for the tripod of the laser vibrometer to be used without requiring additional mountings or mirroring. A detailed list of components excluding the laser is indicated in Appendix E.1 together with a schematic of the full component configuration for the final Mark 3 system.

4.3.3 Results

The results of the specimens tested for both AL2024 and Al7075 are indicated in Figure 4-39 and Figure 4-40 respectively.

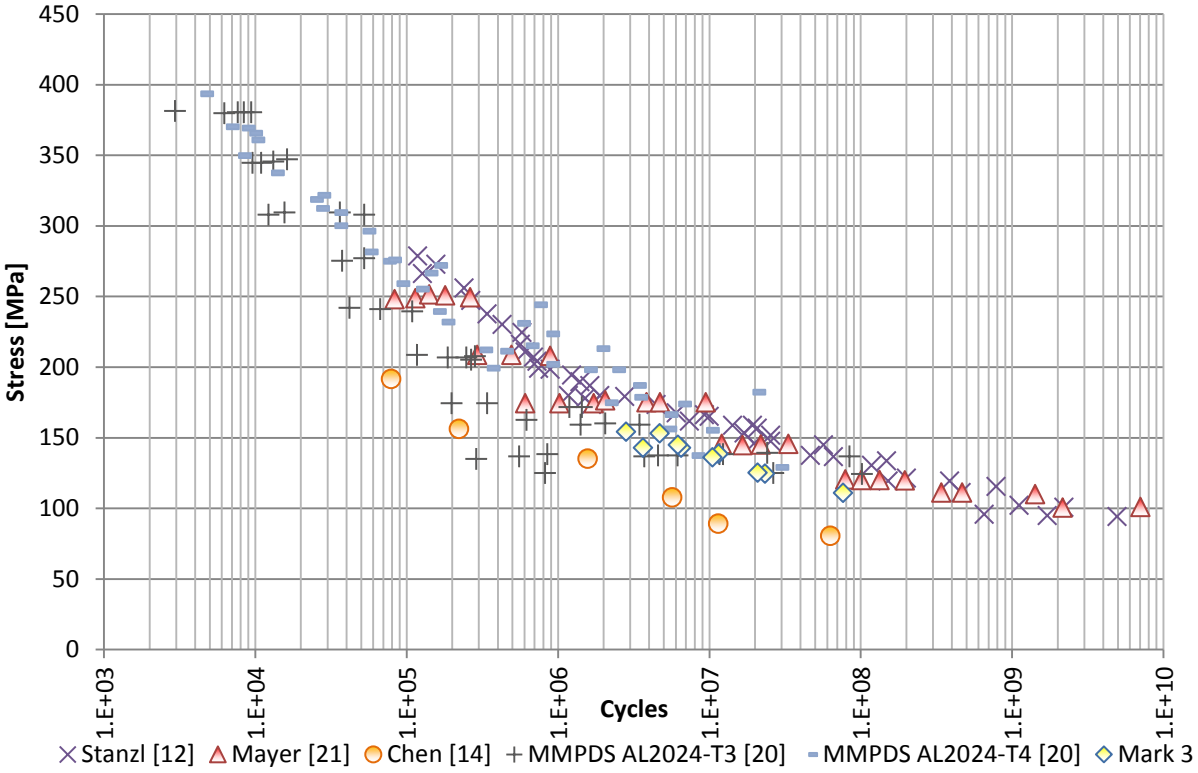


Figure 4-39: Mark 3 AL2024-T351

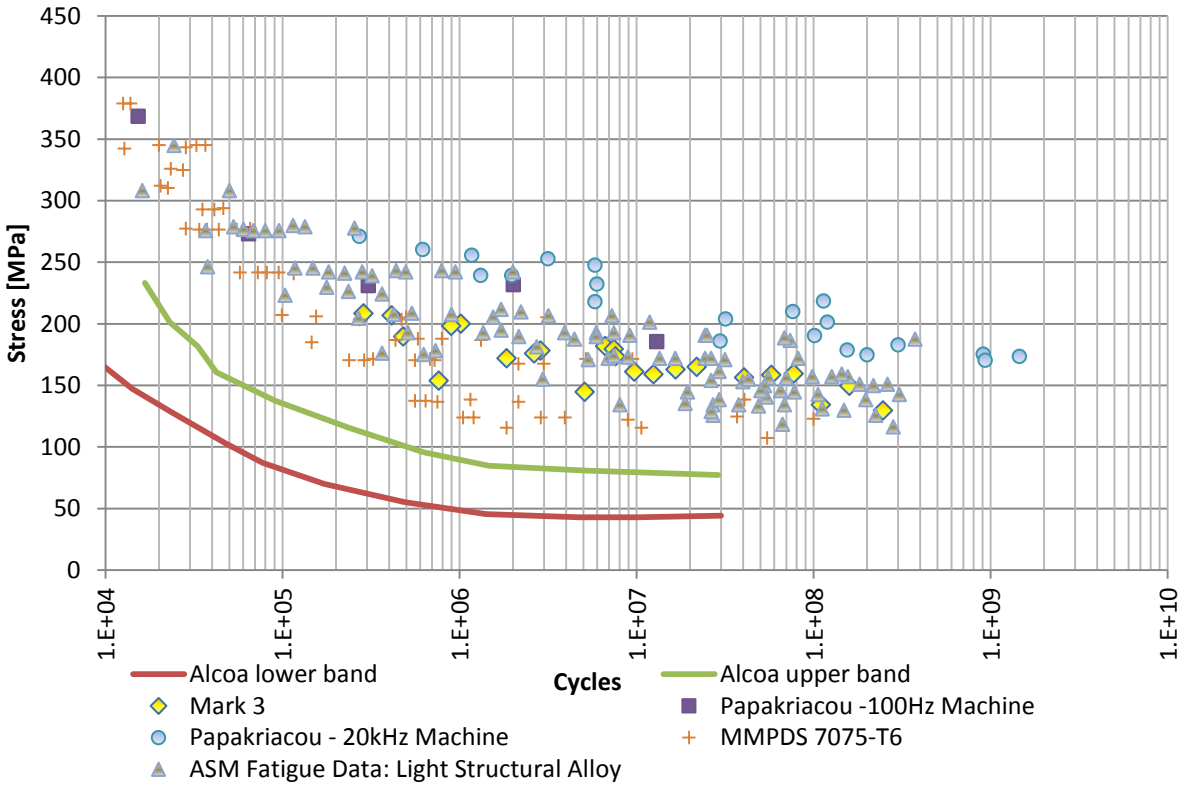


Figure 4-40: Mark 3 AL7075-T7351

4.3.4 Discussion

One of the reasons that may cause this slight shift in the data obtained may be the surface finish as all the specimens tested had a surface finish of 3.2µm as obtained from machining. The effect of the surface finish appeared to be supported with the fact that all the specimens for AL2024 indicated crack initiation from the surface which is visually apparent as indicated in Figure 4-37.

The effect of the surface finish was verified by the additional testing of specimens with an improved surface finish. The obtained data indicated no significant deviation between the two different surface finishes. Thus the data indicated in Figure 4-39 and Figure 4-40 includes all the data for both surface roughness's. The comparison of the polished (surface roughness of 1.6 µm) and as received from turning (surface roughness of 3.2 µm) is given in Figure 4-41 .

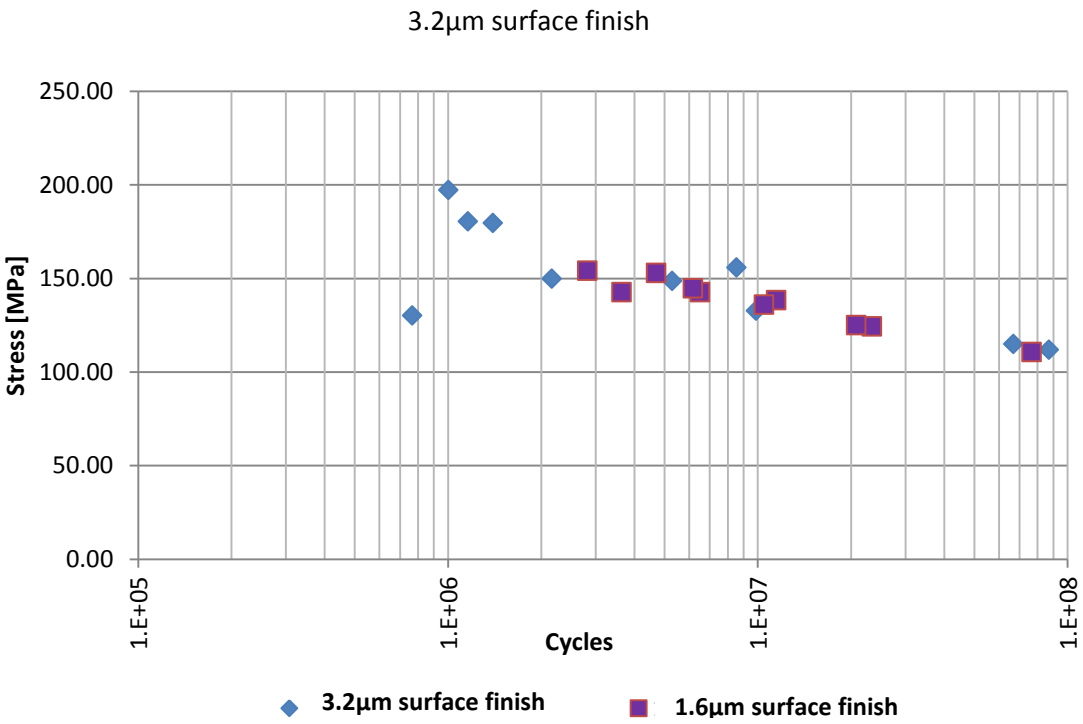


Figure 4-41: AL2024 comparison of surface finish

As for AL2024 the surface finish of the specimens for the AL7075 specimens was varied from 3.2µm initially to 1.6µm. The results of these tests are indicated in Figure 4-42. The effect of the surface finish had a negligible effect on the overall stress for the corresponding number of cycles.

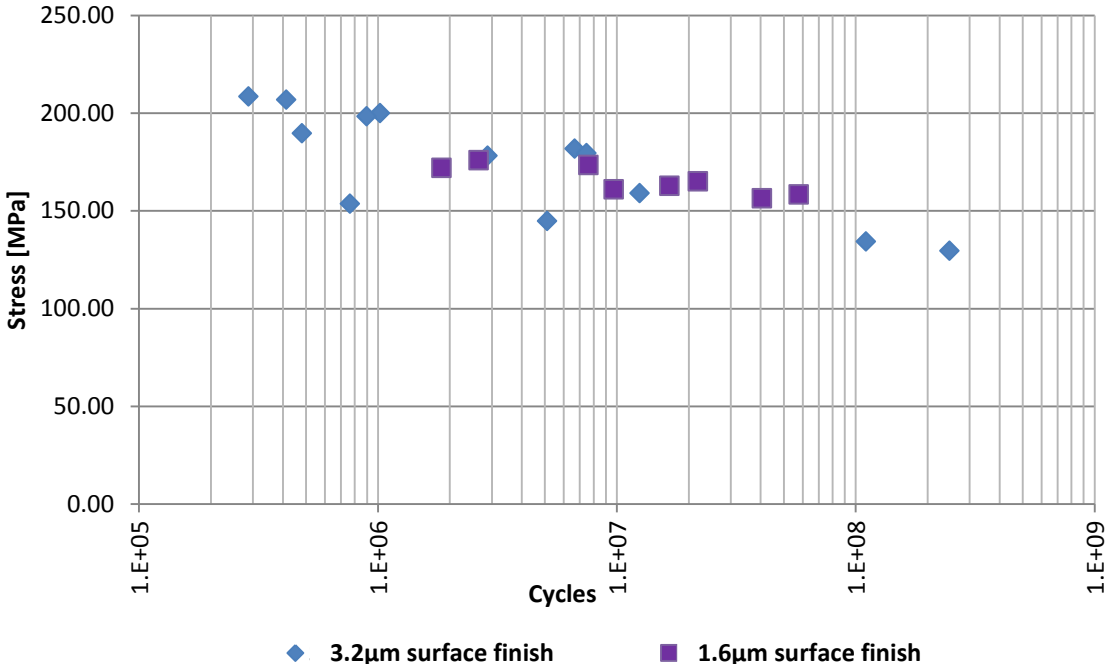


Figure 4-42: AL7075 variation due to surface finish

Overall it is evident that as per the Bathias [4] the surface finish has a negligible effect on the final results for testing where the cycles exceed 10^6. However, it is notable that the consistency of the current tests improved with a better surface finish.

5 RESULTS

In this section the UHCF data is presented for the AL2024 material followed by the AL7075 material. All the data from both the Mark 2 and Mark 3 tests are presented and error bars are defined based on possible error introductions. The data for both materials is compared to literature for validation purposes.

The accumulation of data was done using peak picking on the strain gauge and the laser data for the Mark 2 and Mark 3 designs (Mark 3 data only consisted of laser vibrometer data). This was directly compared to the frequency maintained and recorded by the generator using the Branson built in tool kits.

5.1 Test Numbering

In testing its good practice to keep a detailed test log and maintain a defined numbering convention. Thus the number of the specimens will be as follows:

H1_SY_Z

The definition of each symbol is as follows:

H1 - Is the symbol indicating the horn material type which is always AL7075 for the Mark 2 and 3 data.
S- Is the symbol indicating the specimen material type.
Y- Is the specimen material number either 1 for Aluminium 7075 or 2 for Aluminium 2024.
Z- Is the specimen group number starting at one and incrementing per test.

5.2 Material Delivery Test Reports

Material delivery test reports were obtained for each of the materials received for manufacture of the horns and the specimens. These test reports are indicated in Appendix I.

5.3 Cost Analysis

During procurement of the hardware and manufacture of the rig a cost analysis was undertaken whereby suppliers were sourced. In the end suppliers were chosen based on:

- Capability to supply the correct required components
- Lowest cost
- Support and future availability

Consequently the components were sourced with the final costing and supplier list indicated in Appendix J.

5.4 Defining Possible Errors

To determine the possible errors that could affect the final measurements from the system and to determine the uncertainty in the measurements, a total of four possible sources of error were considered:

- Errors being introduced from the voltage measurements and inherently the end displacement.
- Errors from deviations from the manufacturing drawings. This could arise as the defined allowable tolerance was ±0.2mm on all lengths, including all the radii. The radii include the radius at the neck of the dog-bone together with the dog-bone and all other profile radii.
- Errors due to deviations in the Young's modulus.
- Frequency variation.

All uncertainties were calculated for the respective specimen materials, however, only AL7075 will be indicated below. The uncertainties are typically based on the end displacement measured at the laser vibrometer location since both the Mark 2 and 3 implemented the laser data acquisition system.

5.4.1 Input Displacement and Voltage Variation

The errors introduced by the voltage measurements were assumed to be due to the displacement being estimated from a constant frequency of 20 kHz from the velocity measured by the laser vibrometer, see equation 35 of Appendix B.1. Generally the frequency can vary at 20,000±50Hz. Thus for a velocity of 1m/s, unit input, and a possible frequency fluctuation of ±50Hz. The error can equate to a displacement amplitude range of 49.875 µm (1/20,050) to 50.125 µm (1/19,950) with a mean at 20 kHz with a 50Hz fluctuation. This equates to displacement error tolerance of ±0.125µm. This tolerance variation was then run though the analytical wave equation implemented in Matlab, Appendix G.4, and validated for what affect the input variation would have on the stress. Three end displacements were chosen namely: 10, 20 and 30 µε. This gave the following variation for the lower and upper tolerances:

Table 5-1: Stress variation due to input amplitude errors

End Displacement	**10µm**		**20µm**		**30µm**	
Range	Lower 9.875µm	Upper 10.125µm	Lower 19.875µm	Upper 20.125µm	Lower 29.875µm	Upper 30.125µm
Original Stress [MPa]	67.65		133.41		197.67	
Stress at Specimen [MPa]	65.39	68.95	132.59	134.23	196.38	198.38
Stress Range [MPa]	3.56		1.64		2.00	
Percentage Range to original	5.26%		1.23%		1.01%	

Thus a suitable error range to implement would be conservatively ±4.0MPa ($e_1 = 4$).

5.4.2 Geometric Manufacturing Errors

The errors introduced by the maximum and minimum geometric manufacturing tolerance were considered for three end displacements as indicated in Table 5-2. All geometric dimensions were increase by 0.2mm and conversely were decreased by 0.2mm.

Table 5-2: Stress variation due to geometric tolerance errors

End Displacement		10µm	20µm	30µm
Stress [MPa]	Original	67.65	133.41	197.67
	Dimensions increased by 0.2mm	62.36	128.06	190.45
	Dimensions decreased by 0.2mm	70.19	140.34	203.61
Absolute stress error [MPa]	Dimensions increased by 0.2mm	5.29	5.35	7.22
	Dimensions decreased by 0.2mm	2.54	6.93	5.94
Percentage error on	Dimensions increased by 0.2mm	7.82%	4.01%	3.65%
	Dimensions decreased by 0.2mm	3.75%	5.19%	3.01%

Thus an overall error range of ±7.5MPa ($e_2 = 7.5$) is sufficient to account for possible variations in the stress level.

5.4.3 Young's Modulus Errors

The Young's modulus of the material affects the analytically calculated stress at the specimen centre through the solution of the second order wave equation. The errors induced by the Young's modulus were examined using the residual plots; an example is given in Appendix H. Using the residual plot, the Young's modulus was calculated based on an estimated experimental area remaining after the crack was generated and the specimen separated by tensile test. The uncertainty in the Young's modulus was calculated to be:

Table 5-3: Young's modulus variation from residual plots

Material	Young's Modulus	
	Calculated	As per MMPDS [20]
AL2024	2.0% variation	72.395
AL7075		71.016

Using the MMPDS [20] specifications the compressive and tensile moduli differed by 200 ksi (1.4 GPa). This error also equates conservatively to an approximate error of 2% on the original tensile moduli for both AL2024 and Al7075.

If this variation of 1.4GPa was input into the analytical solution for the second order differential equation, Appendix G.4, then the errors obtained for the end displacements are given in Table 5-4.

Table 5-4: Stress variation due to Young's modulus variation

End displacement	10µm	20µm	30µm
Mean Stress [MPa]	67.65	133.41	197.67
E (+1.4GPa) [MPa]	69.98	141.37	204.83
E (-1.4GPa) [MPa]	63.04	126.46	188.62
Absolute E (+1.4GPa) error	2.33	7.96	7.16
Absolute E (-1.4GPa) error	4.61	6.95	9.05
Range Error [%]	10.25%	11.18%	8.20%

The total error possible is conservatively taken as ±10MPa $(e_3 = 10.0)$.

5.4.4 Frequency Variation Errors

The frequency is used in the laser vibrometer data to calculate the displacement from the velocity amplitude as discussed in section 4.2.2. However, it is also used in the second order analytical solution of the wave equation. Thus its effect on the data was examined to identify the possible error it could introduce into the final results.

The allowable tolerance on the Branson power supply is a frequency of ±50Hz on the 20 kHz input frequency. Consequently Table 5-5 was generated to analyse the variability in the error from 10µm to 30µm end displacements at the end of the specimen.

Table 5-5: Stress variation due to frequency variation

End displacement	10µm	20µm	30µm
Frequency [Hz]	**Stress [MPa]**		
20,000	67.65	133.41	197.67
20,050	65.93	132.21	195.9
19,950	66.63	134.77	204.76
Absolute error for 20,050Hz	1.72	1.2	1.77
Absolute error for 19,950Hz	1.02	1.36	7.09
Range Error [%]	1.03%	1.92%	4.48%

The possible error for frequency variation is taken as ±7.2MPa($e_4 = 7.2$). It is noted that the error for 30µm does not seem to follow the trend of a 2MPa difference. This is believed to be due to the non-linearity variation of the stress along the specimen. Conservatively this effect is included in the error bar range.

5.4.5 Total Error

The total error that will be implemented as the error bar is taken as the combination of the all the above errors:

- Input amplitude variation – ±4.0MPa
- Geometric tolerance variation -±7.5MPa
- Young's modulus variation – ±10.0MPa
- Frequency variation – ±7.2MPa
- Total Error – ±15.0MPa

The total error is obtained using the sum of squares, square rooted, given by the following equation taken from Rabinovich [33]:

$$e_{tot} = \sqrt{\sum_{i=1}^{4} e_i}$$

This assumes that the errors are independent of one another. If the errors were dependant then a normal summation would be more applicable.

The error in the number of cycles can be considered to be exact as even for a couple of missed seconds equating to 40,000 cycles. The plotting of this error bar will not be evident on the log curve in the region exceeding 10^6 cycles. Thus the horizontal uncertainty is omitted as the error in the time measurements will never exceed 2 seconds.

Similarly the following errors were obtained for the AL2024 specimen with the AL7075 horn:

- Input amplitude variation – ±2.0MPa
- Geometric tolerance variation -±7.5MPa
- Young's modulus variation – ±12.0MPa
- Frequency variation – ±2.0MPa
- Total Error – ±14.4MPa

5.5 AL2024 UHCF Data

The data obtained during validation testing of the UHCF test rig for AL2024-T351 is indicated in Figure 5-1 together with the data as obtained from section 2.4.1.

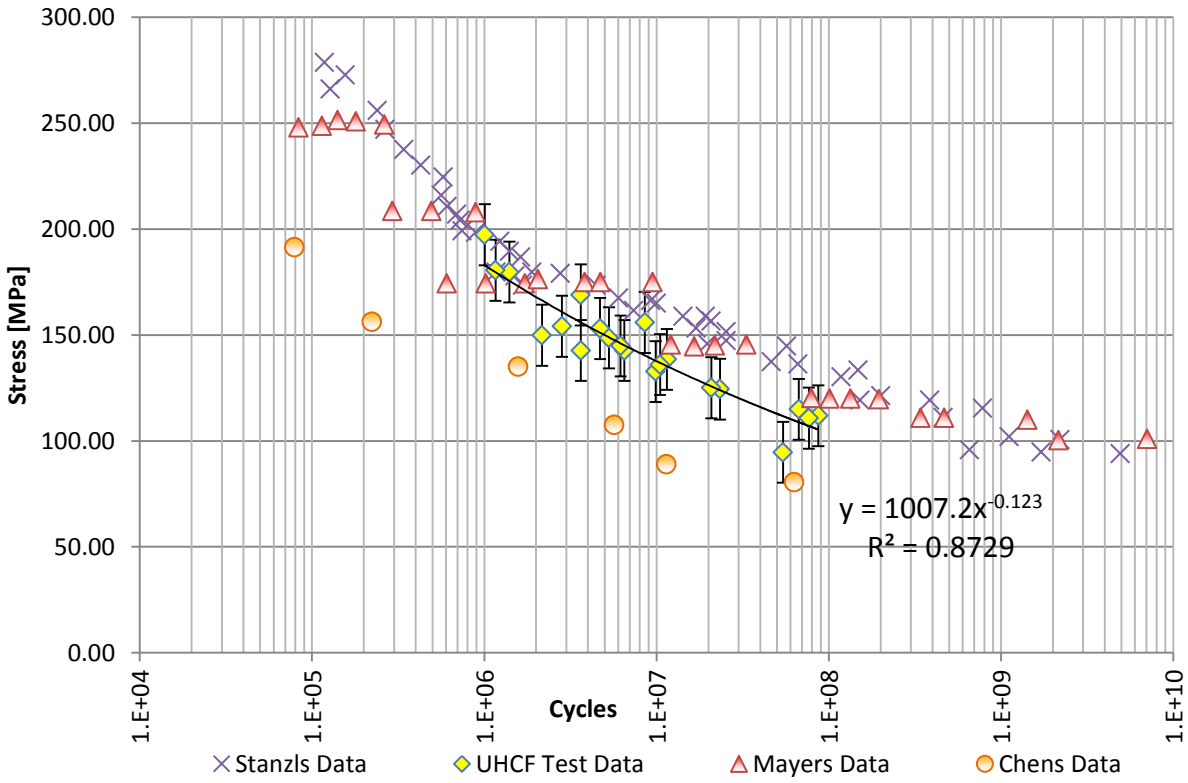


Figure 5-1: Comparison of AL2024-T351 Data

The data fit is obtained using least squares method implemented in Matlab as per Appendix F.1. The trend line data fit of the data gives the following equation:

$$y = 1007.2x^{-0.123}$$

The data fits shows an acceptable R^2 error of 0.87 which indicates the curve fit is good fit.

Overall it can be seen that the Data obtain is slightly lower than both the data of Stanzl and Mayer for cycles exceeding $2x10^6$. The upper band of the error bars show very good correlation which suggests further investigation into the errors is required for future tests. In the region lower than $2x10^6$ cycles the data shows a great correlation.

The data obtain by Mayer [21] had the following Basquin equation for a load ratio of R=-1:

$$y = 823.63x^{-0.103} \quad for\ \sigma_{amp} > 110MPa$$

$$y = 160.00x^{-0.019} \quad for\ \sigma_{amp} < 110MPa$$

The Basquin coefficient at 1 cycle for the data of Meyer indicates a stress of 823.63MPa. This differs slightly, 21.37% difference, for the current test estimation of 999.63MPa.

The ALCOA data is obtained from the technical specification [19] which indicates data all the way up till $2x10^7$ cycles. This data is not indicated in Figure 5-1 as both of these ALCOA curves, which form a band, are much lower than the data obtain here and that of Stanzl [12] and even Chen [14]. It is believed this is due to the fact that the data has been statistically reduced to have a defined confidence level. Consequently the ALCOA data is not raw data and is thus not indicated.

The typical micrographic image of the failed specimen is indicated in Figure 5-2.

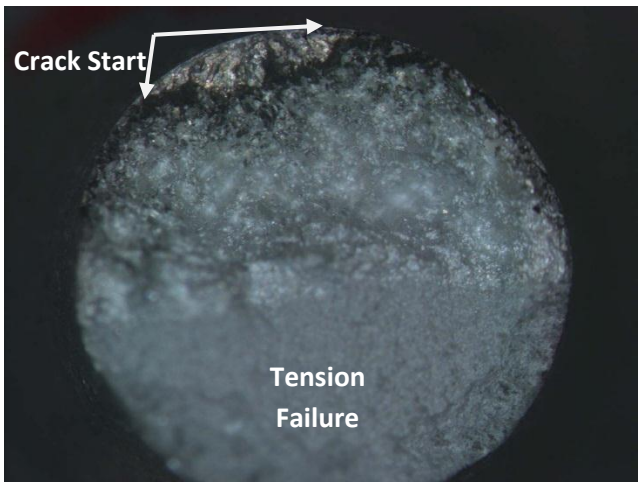


Figure 5-2: Micrographic image of AL2024 Specimen H1-S2-13

The region of the crack is very uneven and thus the tension failure region it not in focus with the crack start location, being the region focussed on.

The data of Chen *et al*. [14] indicated in Figure 5-1 indicates a trend of lower stress for the same cycles as compared to the UHCF data. Chen *et al.* [14] did not state to what surface finish the specimens had been implemented, however the test performed by this report corroborated with Bathias [4] that the surface finish has no effect on the final result for cycles exceeding 10^6. A comparison of the material composition of AL2024 from Chen *et al*. [14] was compared to that delivered from Kaiser for this report, see Appendix I. This comparison is indicated in Table 5-6.

Table 5-6: Chemical composition comparison of AL2024

Material	Zn	Mn	Cu	Fe	Si	Cr	Ti	Mg	Others
AL2024-T351 (Kaiser)	0.23	0.77	4.5	0.11	0.07	0.02	0.01	1.3	-
AL2024 Chen [14]	0.25	0.5	4.0	0.5	0.5	0.1	0.15	1.5	-

It is evident that the two materials differ slightly in chemical composition. Whether this affects the results significantly is unknown at present and will only be verified by further testing of the same composition.

Overall the data obtained shows a good correlation with known data from literature. Typically in fatigue there is significant scatter and the AL2024 data is no different. The data obtained falls safely in the region of known data which proves the data is acceptable and the rig is working effectively and with future improvements this will only improve.

5.6 AL7075 UHCF Data

The data obtained during the validation of the UHCF test rig for AL7075-T7351 is indicated in Figure 5-3 together with the obtained data as per section 2.4.2.

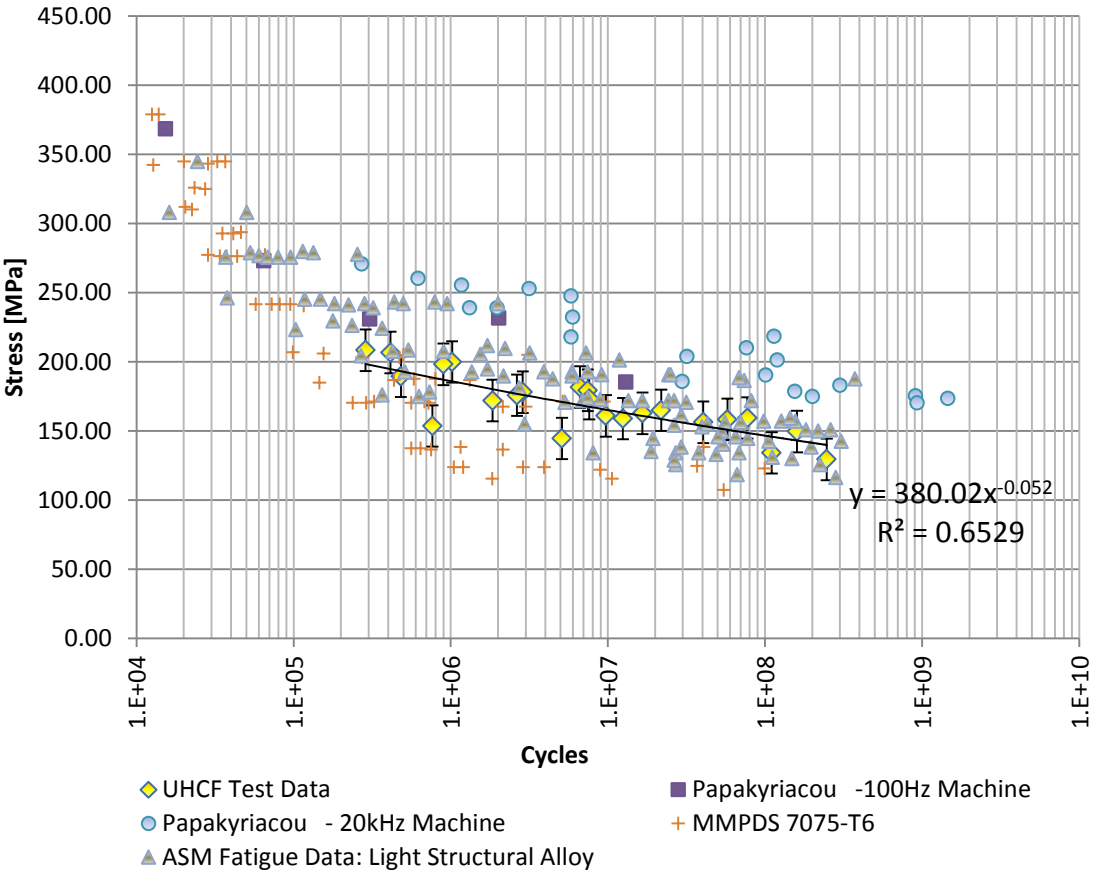


Figure 5-3: Comparison of AL7075-T7351 Data

The other test data is that of Papakyriacou [16] which was for ALZnMgCu1.5. This material differs somewhat from the material received from the supplier, Kaiser (see Appendix I) as indicated by the comparison of the chemical composition Table 5-7:

Table 5-7: Chemical composition comparison of Al7075

Material	Si	Fe	Cu	Mn	Mg	Cr	Zn	Ti	Zr	Others
AL7075-T73 (Kaiser)	0.10	0.25	1.40	0.04	2.70	0.20	5.80	0.03	-	0.05
AA7075 ALZnMgCu1.5	0.14	0.27	1.73	0.07	2.72	0.02	7.26	0.06	0.13	-

Using Matlab and the least squares method as described in Appendix F.1, the following is obtained:

$$y = 380.02x^{-0.052}$$

The R^2 ratio indicates that the fit is not entirely accurate to the data. However fatigue data generally has a high scatter and thus the ratio will typically be low for fatigue curve fits. The gradient of the

curve fit follows the ASM fatigue data for cycles exceeding $2x10^5$ and corresponds well to the data of Papakyriacou using the 20 kHz machine.

The ALCOA data is obtained from the technical specification [24] and indicates the data all the way up and till $2x10^7$ cycles. Both of these ALCOA curves (which form a band) are much lower than the data obtain by this report. It is believed this is due to the fact that the data is either A, B or S-basis. Thus the ALCOA data has been statistically reduced to have a statistical confidence as required by each basis for design. Consequently this data is not indicated as it is not raw data.

The specimens of AL7075 all indicate that the crack initiated at the surface and grew until the stiffness reduction significantly modified the resonance frequency such that operation was no longer viable and testing was stopped. The specimens were subsequently loaded in tension to failure in a universal testing machine to allow for the examination of the failure surface. A typical failure image is indicated for AL7075 specimen 3 (H1_S1_3):

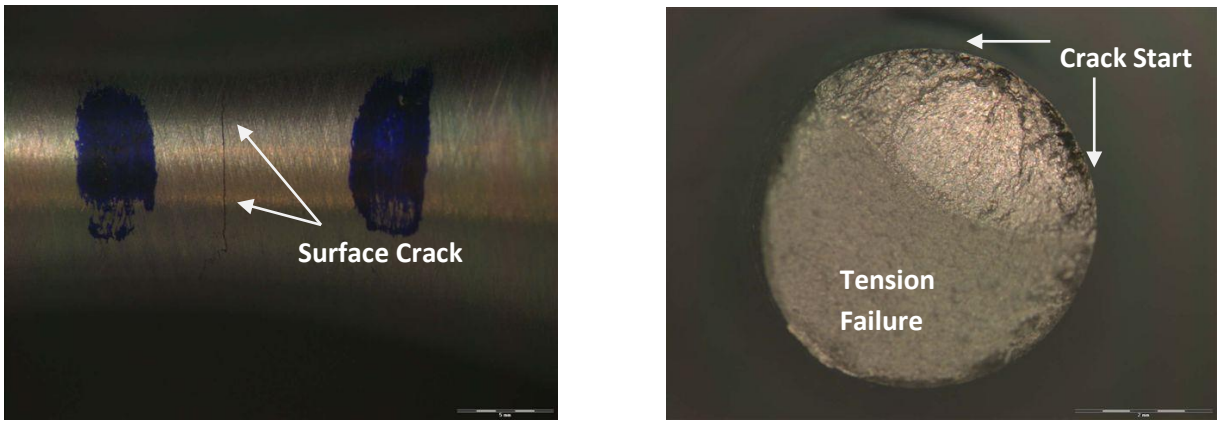


Figure 5-4: Typical failure of AL7075

The tension failure is indicated by the marked 45° shear line, see Figure 5-5, which was induced from the tension loading used to break the specimen for visualisation.



Figure 5-5: Failed specimen showing tension failure (H1-S1-6)

Overall the AL7075 had a very good correlation to the known data, further supporting the notion that the UHCF testing rig is operating within acceptable limits to obtain valid UHCF data.

6 CONCLUSIONS AND RECOMMENDATIONS

It is concluded that the design and manufacture of an UHCF testing machine has been accomplished and that the validation against known fatigue data has been proven successful.

The testing indicated that the strain gauges provided accurate cycle counting for all phases of testing. Testing did reveal that the strain gauges at lower ends of the UHCF ($<10^6$) spectrum were ineffective due to regular failure during testing under the higher loading. The strain gauge system did continually fail during testing of the Mark 2 system thus incurring higher costs, longer testing times and continuous interruptions. As a result in the final Mark 3 design the use of the more accurate and reliable laser vibrometer was implemented solely.

The obtained SN curve data indicated that a set trend can be obtained with repeatable data. It was initially assumed that deviations of the AL2024 data from the literature may indicate that the surface finish plays an important role in the final fatigue life as in normal HCF. However further conducted tests indicated that this was not the case. The Data of AL7075 indicated very good correlation with data obtained on normal fatigue testing equipment in the region of 10^6 cycles which gives confidence in the current design configuration.

It is recommended that future tests be carried out on steel and titanium to advance the knowledge base and create a detail database of information. These tests will also allow for the measurement system to be improved further.

If possible future composite material testing methods should be devised and tested to obtain composite allowables for future designs. It is understood that fatigue in composites is completely different to that in metals and to actually get a resonant coupon in composites will be the most difficult task. Also, during fatigue, cracks will be initiated and retarded within the composite matrix which will change the resonance of the tested specimen. This effect needs to be tracked to maintain a 20 kHz operating range or the operating range will need to be widened.

7 REFERENCES

[1] I Marines *et. Al*, An Understanding of Very High Cycle Fatigue of Metals, International Journal of Fatigue, 25, (2003), pg1101-1107

[2] C Bathias, Piezoelectric fatigue testing machines and devices, International Journal of Fatigue 28, (2006), 1438-1445.

[3] ASTM, E739-91, Standard Practice for Statistical Analysis of Linear or Linearized stress-Life (S-N) and Strain-Life (ε-N) Fatigue Data, Conshohocken, PA.

[4] Gigacycle fatigue in mechanical practice, C Bathias.

[5] Y. Lee, J Pan, R Hathaway and M Barkley, Fatigue Testing And Analysis, Theory and Practice Elsevier 2005.

[6] J. Schijve, Fatigue of structures and materials, Springer 2009.

[7] S Dinur and T McMillan, Sonic Fatigue Consideration for the A400M, International Aerospace Symposium of South Africa (IASA) 2009.

[8] S Dinur, Implementation of Ultra High Cycle Fatigue (Giga-cycles) in Propeller-induced Sonic Fatigue Analysis, International Aerospace Symposium of South Africa (IASA) 2011.

[9] C. Bathias *et. Al*, How and why the Fatigue S-N curve does not Approach a Horizontal Asymptote , International Journal of Fatigue, 23, (2001), pg143-151.

[10] Y. Murakami *et. Al*, Factors Influencing Mechanisms of Super long Fatigue Failures, , Fatigue Fracture Engineering Material Structure, vol 22, (1999), pg. 581-590.

[11] Stepwise S-N curve and fish-eye failure in Gigacycle Fatigue, S. Nisijima et. all, Fatigue Fracture Engineering Material Structure, 22, (1999), 601-607

[12] Stanzl-Tschegg *et. Al*, Fatigue and Fatigue Growth of Aluminium at Very High Numbers of Cycles, S.C., International Journal of Fatigue, vol 23, (2001), S231-S237

[13] H. Mayer, Ultrasonic torsion and tension-compression fatigue testing: Measuring principles and investigation on 2024-T351 aluminium alloy, international Journal of Fatigue 28 (2006) pg. 1446-1455.

[14] Y Chen, B Wei and W Hsu, Ultrasonic Fatigue Testing of Aluminium and Magnesium Alloys, Journal of Chinese Society of Mechanical Engineers, vol 29 No4 page 299-308, 2008.

[15] M. Papakyriacou, H. Mayer, S. Stanzl-Tschegg, C. Pypen and H. Plenk Jr, Influence of loading frequency on high cycle fatigue properties of b.c.c and h.c.p metals, 23 November 2000, Material Science and Engineering Journal A308.

[16] M. Papakyriacou, H. Mayer, S. Stanzl-Tschegg, and R. Pippan, Influence of loading frequency on high cycle fatigue properties on AlZnMgCu1.5 aluminium alloy, 2001, Material Science and Engineering Journal A314.

[17] C. Laird, P. Charsley, Ultrasonic Fatigue, The Metall. Society of AIME Philadelphia, 1982, pp. 187-205.

[18] Budynas-Nisbett, Shigley's Mechanical Engineering Design, McGraw-Hill Primis, 8th edition.

[19] ALLOY 2024 Sheet and Plate ALCOA Mill Product technical sheet, SPD-10-036, http://www.alcoa.com/mill_products/catalog/pdf/alloy2024techsheet.pdf, 11/10/2014.

[20] Metallic Material Properties Development and Standardization (MMPDS), U.S Department of Transportation, January 2003.

[21] H. Mayer, R. Schuller and M. Fitzka, Fatigue of 2024-T351 aluminium alloy at different load ratios up to 10^{10} cycles, 19 January 2012, International Journal of Fatigue.

[22] ASM International, Fatigue Data Book: Light Structural Alloys, November 1995.

[23] H Mayer, B Hopler, A Vasudevan and S Stanzl-Tschegg, Near threshold fatigue crack growth in aluminium allows at low and ultrasonic frequency: Influences of specimen thickness, strain rate, slip behaviour and air humidity, International Journal of Fatigue, 30 October 2002

[24] ALLOY 7075 Sheet and Plate ALCOA Mill Product technical sheet, SPD-10-037, https://www.alcoa.com/mill_products/catalog/pdf/alloy7075techsheet.pdf, 11/10/2014.

[25] http://www.polytec.com/us/solutions/vibration-measurement/basic-principles-of-vibrometry/, 07/11/2014.

[26] Www.ni.com/white-paper/3642/en/, 01/04/2014.

[27] http://www.omega.com/techref/pdf/StrainGage_Measurement.pdf, 02/04/2014.

[28] R.D. Cook, Concepts and applications of Finite element analysis, 4th edition.

[29] Colorado university – Advanced Finite Element method Notes - http://www.colorado.edu/engineering/CAS/courses.d/AFEM.d/Home.html (20/02/2014)

[30] Andras Szekrenyes, Modelling of Axisymmetric state by FEM software Systems, Modelling Analysis of Problem Evaluation, BME, www.tankonyvtar.hu.

[31] http://web2.murraystate.edu/andy.batts/ps/POWERSUPPLY.HTM, ATX power supply data, 25/04/2014.

[32] Department of Defence handbook, Metallic materials and elements for aerospace vehicle structures, MIL-HDBK-5J, 31 January 2003.

[33] S.G Rabinovich, Measurement Errors and Uncertainties: Theory and Practice, Springer & Business Media December 2006.

[34] E. Haibach, Modified linear damage accumulation hypothesis accounting for decreasing fatigue strength during increasing fatigue damage (in German). Lab. für Betriebsfestigkeit, LBF, Darmstadt, TM Nr.50, 1970.

APPENDIX A - ANALYTICAL FORMULATION

This appendix details the analytical formulation of the displacement, strain and stress using the second order differential wave equation. The following is a typical specimen horn and actuator arrangement with geometry indicated.

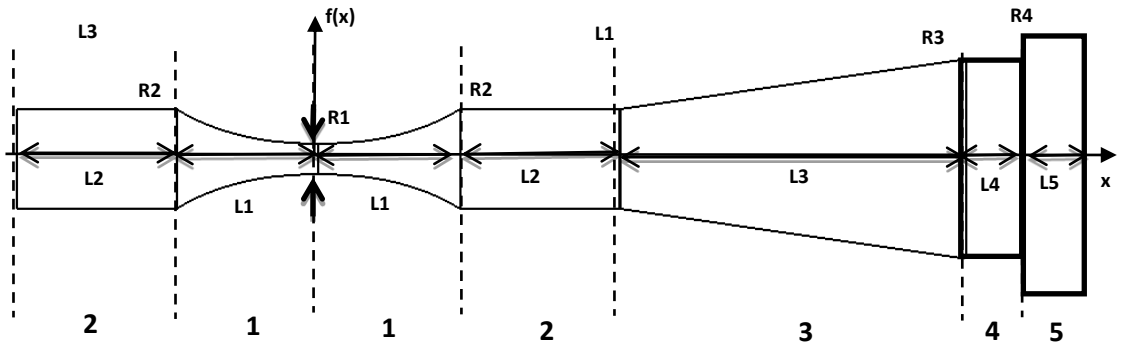


Figure A-1: General sonic fatigue specimen

Using the above geometry the radius variation in the x direction can be defined as:

$$R(x)=\begin{cases} R_1\cosh(\alpha_1 x) & 0 <= x < L_1 \\ R_2 & L_2 <= x < L_1+L_2 \\ R_2\cosh(\alpha_2 x_2) & L_1+L_2 <= x < L_1+L_2+L_3 \\ R_3 & L_1+L_2+L_3 <= x < L_1+L_2+L_3+L_4 \\ R_4 & L_1+L_2+L_3+L_4 <= x < L_1+L_2+L_3+L_4+L_5 \end{cases} \quad \textbf{8}$$

Where

$$x_2 = x - L_1 - L_2$$

$$\alpha_1 = \frac{1}{L_1}\operatorname{arccosh}\left(\frac{R_2}{R_1}\right) \quad \alpha_2 = \frac{1}{L_3}\operatorname{arccosh}\left(\frac{R_3}{R_2}\right)$$

The profile on the negative x-axis is a mirror image of that of the right and hence the displacements will be a mirror image. Thus if plotting the above equations for:

L1=15*10^-3; %meters
R1=3/2*10^-3; %meters
R2=10/2*10^-3; %meters
R3=15.5*10^-3; %meters
L4=20*10^-3; %meters
L3=40*10^-3; %meters

L2 is found using the following equation [4] which is obtained by comparing equations 22 and 23.

$$L_2 = \frac{1}{k}\arctan\left\{\frac{1}{k}\left[\beta_1 \coth(\beta_1 L_1) - \alpha_1 \tanh(\alpha_1 L_1)\right]\right\} \quad \textbf{9}$$

Then the following profile plot can be generated for a mirrored geometry:

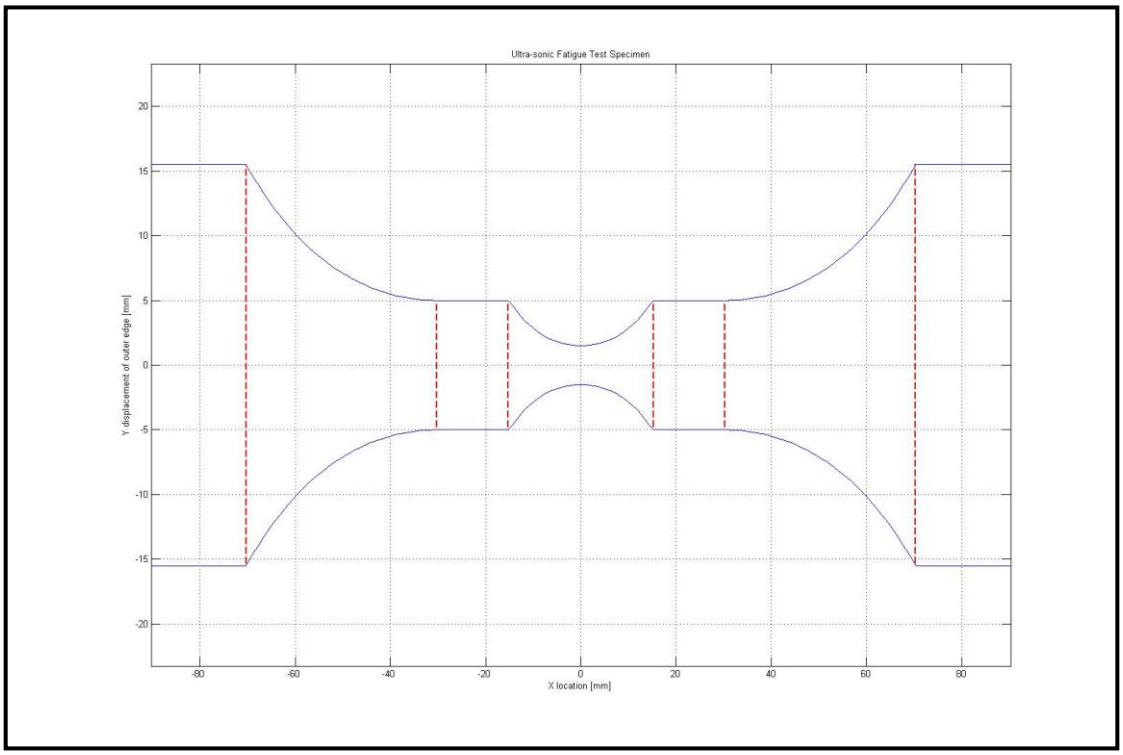


Figure A-2: Profile plot of specimen using Matlab

Hence the cross sectional area variation is given by:

$$S(x) = \begin{cases} \pi R_1^2 \cosh^2(\alpha_1 x) & 0 <= x < L_1 \\ \pi R_2^2 & L_2 <= x < L_1 + L_2 \\ \pi R_2^2 \cosh^2(\alpha_2 x_2) & L_1 + L_2 <= x < L_1 + L_2 + L_3 \\ \pi R_3^2 & L_1 + L_2 + L_3 <= x < L_1 + L_2 + L_3 + L_4 \\ \pi R_4^2 & L_1 + L_2 + L_3 + L_4 <= x < L_1 + L_2 + L_3 + L_4 + L_5 \end{cases} \quad \textbf{10}$$

The solution for displacement, stress and strain at each section can then be done systematically for each of the sections indicated in Bathias [4] .

A.1 Section 1 – General Differential (0 <= x < L1)

Starting from the general equation 7:

$$\frac{\partial^2 U(x)}{\partial x^2} + P(x)\frac{\partial U(x)}{\partial x} + k^2 U(x) = 0$$

The variation of the area within this region is given by:

$$S(x) = \pi R_1^2 \cosh^2(\alpha_1 x)$$

Thus the term P(x) is given by:

$$P(x) = \frac{1}{S(x)} \frac{\partial S(x)}{\partial x} = \frac{1}{\pi R_1^2 \cosh^2(\alpha_1 x)} \frac{\partial}{\partial x} \left(\pi R_1^2 \cosh^2(\alpha_1 x) \right)$$

Simplifying and using the chain rule the following is obtained:

$$P(x) = \frac{2\pi R_1^2 \cosh(\alpha_1 x).\alpha_1.\sinh(\alpha_1 x)}{\pi R_1^2 \cosh^2(\alpha_1 x)} = 2\alpha_1 \tanh(\alpha_1 x)$$

Substituting the above into equation 7:

$$\frac{\partial^2 U(x)}{\partial x^2} + 2\alpha_1 \tanh(\alpha_1 x) \frac{\partial U(x)}{\partial x} + k^2 U(x) = 0 \qquad \textbf{11}$$

To solve this, following function is introduced [4]:

$$w_1(x) = \cosh(\alpha_1 x).U(x)$$

Then the first derivative of this function is:

$$w_1'(x) = \frac{dw_1(x)}{dx} = \left[\alpha_1 \sinh(\alpha_1 x) U(x) + \cosh(\alpha_1 x) \frac{dU(x)}{dx} \right]$$

Then the second derivative of this function is:

$$w_1''(x) = \frac{d^2 w_1(x)}{dx^2} = \left[\alpha_1^2 \cosh(\alpha_1 x) U(x) + \alpha_1 \sinh(\alpha_1 x) \frac{dU(x)}{dx} + \alpha_1 \sin(\alpha_1 x) \frac{dU(x)}{dx} + \cosh(\alpha_1 x) \frac{d^2 U(x)}{dx^2} \right]$$

$$\therefore w_1''(x) = \frac{d^2 w_1(x)}{dx^2} = \left[\alpha_1^2 \cosh(\alpha_1 x) U(x) + 2\alpha_1 \sin(\alpha_1 x) \frac{dU(x)}{dx} + \cosh(\alpha_1 x) \frac{d^2 U(x)}{dx^2} \right]$$

$$\therefore w_1''(x) = \frac{d^2 w_1(x)}{dx^2} = \left[\alpha_1^2 U(x) + 2\alpha \tanh(\alpha_1 x) \frac{dU(x)}{dx} + \frac{d^2 U(x)}{dx^2} \right] \cosh(\alpha_1 x) \qquad \textbf{12}$$

Subtracting equation **12** from equation **11** times cosh ($\alpha_1 x$):

$$w_1''(x) = \frac{d^2 w_1(x)}{dx^2} = \left[\alpha_1^2 U(x) + 2\alpha \tanh(\alpha_1 x) \frac{dU(x)}{dx} + \frac{d^2 U(x)}{dx^2} - \left(\frac{\partial^2 U(x)}{\partial x^2} + 2\alpha_1 \tanh(\alpha_1 x) \frac{\partial U(x)}{\partial x} + k^2 U(x) \right) \right] \cosh(\alpha_1 x)$$

$$\therefore w_1''(x) = \frac{d^2 w_1(x)}{dx^2} = \left[\alpha_1^2 U(x) - \left(k^2 U(x) \right) \right] \cosh(\alpha_1 x)$$

$$\therefore w_1''(x) = \frac{d^2 w_1(x)}{dx^2} = \left[\alpha_1^2 - k^2 \right] U(x) \cosh(\alpha_1 x)$$

But

$$w_1(x) = \cosh(\alpha_1 x).U(x)$$

Thus the following is obtained:

$$\therefore w_1''(x) = \frac{d^2 w_1(x)}{dx^2} = \left[\alpha_1^2 - k^2\right] w_1(x)$$

Thus this new differential equation has the general solution:

$$w_1(x) = C_1 e^{(\beta_1 x)} + C_2 e^{(-\beta_1 x)} \qquad \textbf{13}$$

Where

$$\beta_1 = \sqrt{\alpha_1^2 - k^2}$$

Substituting in the fact that:

$$U(x) = \frac{w_1(x)}{\cosh(\alpha_1 x)}$$

Equation **13** becomes the general solution for section 1:

$$U_1(x) = \frac{C_1 e^{(\beta_1 x)} + C_2 e^{(-\beta_1 x)}}{\cosh(\alpha_1 x)} \qquad \textbf{14}$$

This solution is as per reference [4].

A.2 Section 2 – General Differential ($L_1 <= x < L_1 + L_2$)

Starting from the general equation 7:

$$\frac{\partial^2 U(x)}{\partial x^2} + P(x)\frac{\partial U(x)}{\partial x} + k^2 U(x) = 0$$

The variation of the area within this region is given by:

$$S(x) = \pi R_2^2$$

However since this section is a constant area with x then:

$$P(x) = \frac{1}{S(x)}\frac{\partial S(x)}{\partial x} = 0$$

Since,

$$\frac{\partial S(x)}{\partial x} = 0$$

Therefore equation 7 becomes for this section:

$$\frac{\partial^2 U(x)}{\partial x^2} + k^2 U(x) = 0 \qquad \textbf{15}$$

The general solution for section 2 differential equation is:

$$U_2(x) = C_3 \cos(kx) + C_4 \sin(kx) \qquad \textbf{16}$$

A.3 Section 3 – General Differential (L1 +L2 <= x < L1 +L2+L3)

This region is similar to that of section 1. Thus if a variable x_1 is created which is merely a shift of the x location such that:

$x_1 = x - L_1 - L_2$

Then the same solution as in section 3 is obtained in terms of x_1.

$$U_3(x_1) = \frac{C_5 e^{(\beta_2 x_1)} + C_6 e^{(-\beta_2 x_1)}}{\cosh(\alpha_2 x_1)} \qquad \textbf{17}$$

Where

$$\beta_2 = \sqrt{\alpha_2^2 - k^2}$$

And

$$\alpha_2 = \frac{1}{L_3} \operatorname{arccosh}\left(\frac{R_3}{R_2}\right)$$

A.4 Section 4 – General Differential ($L_1+L_2+L_3$ <= x<$L_1+L_2+L_3+L_4$)

Starting from the general equation 7:

$$\frac{\partial^2 U(x)}{\partial x^2} + P(x)\frac{\partial U(x)}{\partial x} + k^2 U(x) = 0$$

The variation of the area within this region is given by:

$$S(x) = \pi R_3^2$$

However since this section is a constant area with x then:

$$P(x) = \frac{1}{S(x)}\frac{\partial S(x)}{\partial x} = 0$$

Since,

$$\frac{\partial S(x)}{\partial x} = 0$$

Therefore equation 7 becomes for this section:

$$\frac{\partial^2 U(x)}{\partial x^2} + k^2 U(x) = 0 \qquad \textbf{18}$$

The general solution, for section 4, differential equation is:

$$U_4(x) = C_7 \cos(kx) + C_8 \sin(kx) \longrightarrow \qquad \textbf{19}$$

A.5 Section 5 – General Differential ($L_1+L_2+L_3+L_4<=x<L_1+L_2+L_3+L_4+L_5$)

Starting from the general equation 7:

$$\frac{\partial^2 U(x)}{\partial x^2} + P(x)\frac{\partial U(x)}{\partial x} + k^2 U(x) = 0$$

The variation of the area within this region is given by:

$$S(x) = \pi R_4^2$$

However since this section is a constant area with x then the following is valid:

$$P(x) = \frac{1}{S(x)}\frac{\partial S(x)}{\partial x} = 0$$

Since,

$$\frac{\partial S(x)}{\partial x} = 0$$

Therefore equation 7 becomes for this section:

$$\frac{\partial^2 U(x)}{\partial x^2} + k^2 U(x) = 0 \qquad \textbf{20}$$

The general solution, for section 5, differential equation is:

$$U_5(x) = C_9 \cos(kx) + C_{10} \sin(kx) \longrightarrow \qquad \textbf{21}$$

A.6 General boundary conditions

The above equations are linked to each other via their respective boundary conditions at their respective section interfaces. Thus the following relationships are valid for each section interface.

A.6.1 Interface sections 1 and 2

Displacement

$$U_1(L_1) = U_2(L_1)$$

Thus,

$$U_1(x) = \frac{C_1 e^{(\beta_1 L_1)} + C_2 e^{(-\beta_1 L_1)}}{\cosh(\alpha_1 L_1)} = U_2(L_1) = C_3 \cos(kL_1) + C_4 \sin(kL_1)$$

$$\therefore \frac{C_1 e^{(\beta_1 L_1)} + C_2 e^{(-\beta_1 L_1)}}{\cosh(\alpha_1 L_1)} - C_3 \cos(kL_1) - C_4 \sin(kL_1) = 0 \qquad \textbf{22}$$

Strain

$$\frac{dU_1(L_1)}{dx} = \frac{dU_2(L_1)}{dx}$$

$$\therefore \frac{dU_1(L_1)}{dx} = \frac{\left(\beta_1 C_1 e^{(\beta_1 L_1)} - \beta_1 C_2 e^{(-\beta_1 L_1)}\right)\cosh(\alpha_1 L_1) - \left(C_1 e^{(\beta_1 L_1)} + C_2 e^{(-\beta_1 L_1)}\right)\alpha_1 \sinh(\alpha_1 L_1)}{\cosh^2(\alpha_1 L_1)} = \frac{dU_2(L_1)}{dx} = -C_3 k \sin(kL_1) + C_4 k \cos(kL_1)$$

$$\therefore \frac{C_1 e^{(\beta_1 L_1)}(\beta_1 \cosh(\alpha_1 L_1) - \alpha_1 \sinh(\alpha_1 L_1)) - C_2 e^{(-\beta_1 L_1)}(\beta_1 \cosh(\alpha_1 L_1) + \alpha_1 \sinh(\alpha_1 L_1))}{\cosh^2(\alpha_1 L_1)} + C_3 k \sin(kL_1) - C_4 k \cos(kL_1) = 0 \qquad \textbf{23}$$

A.6.2 Interface sections 2 and 3

Displacement

$$U_2(L_1 + L_2) = U_3(x = L_1 + L_2)$$

But for section 3 the data has been shifted using: $x_1 = x - L_1 - L_2$ thus when $x = L_1 + L_2$ thus $x_1 = 0$. Therefore since

$$U_2(x) = C_3 \cos(kx) + C_4 \sin(kx)$$

And

$$U_3(x_1 = 0) = \frac{C_5 e^{(\beta_2 0)} + C_6 e^{(-\beta_2 0)}}{\cosh(\alpha_2 0)}$$

Thus

$$U_2(L_1+L_2)=C_3\cos(k[L_1+L_2])+C_4\sin(k[L_1+L_2])=U_3(L_1+L_2)=\frac{C_5e^{(0)}+C_6e^{(0)}}{\cosh(0)}$$

$$\therefore C_3\cos(k[L_1+L_2])+C_4\sin(k[L_1+L_2])-C_5-C_6=0 \quad \textbf{24}$$

Strain

$$\frac{dU_2(L_1+L_2)}{dx}=\frac{dU_3(x=L_1+L_2)}{dx}=\frac{dU_3(x_1=0)}{dx_1}$$

Therefore since

$$\frac{dU_2(L_1+L_2)}{dx}=-kC_3\sin(k(L_1+L_2))+kC_4\cos(k(L_1+L_2))$$

And

$$\frac{dU_3(0)}{dx_1}=\frac{\left(C_5e^{(\beta_2.0)}-C_6e^{(-\beta_2.0)}\right)\beta_2\cosh(\alpha_2.0)-\left(C_5e^{(\beta_2.0)}+C_6e^{(-\beta_2.0)}\right)\alpha_2\sinh(\alpha_2.0)}{\cosh^2(\alpha_2.0)}$$

$$\therefore \frac{dU_3(0)}{dx_1}=\beta_2C_5-\beta_2C_6$$

Consequently:

$$\therefore -kC_3\sin(k(L_1+L_2))+kC_4\cos(k(L_1+L_2))-\beta_2C_5+\beta_2C_6=0 \quad \textbf{25}$$

A.6.3 Interface sections 3 and 4

Displacement

$$U_3(x=L_1+L_2+L_3)=U_3(x_1=L_3)=U_4(L_1+L_2+L_3)$$

Consequently,

$$U_3(x_1=L_3)=\frac{C_5e^{(\beta_2L_3)}+C_6e^{(\beta_2L_3)}}{\cosh(\alpha_2L_3)}$$

And

$$U_4(L_1+L_2+L_3)=C_7\cos(k[L_1+L_2+L_3])+C_8\sin(k[L_1+L_2+L_3])$$

Therefore combining the two above equation the following is obtained:

$$U_3(x_1=L_3)=\frac{C_5e^{(\beta_2L_3)}+C_6e^{(-\beta_2L_3)}}{\cosh(\alpha_2L_3)}=U_4(L_1+L_2+L_3)=C_7\cos(k[L_1+L_2+L_3])+C_8\sin(k[L_1+L_2+L_3])$$

$$\therefore \frac{C_5 e^{(\beta_2 L_3)} + C_6 e^{(-\beta_2 L_3)}}{\cosh(\alpha_2 L_3)} - C_7 \cos(k[L_1 + L_2 + L_3]) - C_8 \sin(k[L_1 + L_2 + L_3]) = 0 \quad \textbf{26}$$

Strain

$$\frac{dU_3(x = L_1 + L_2 + L_3)}{dx} = \frac{dU_3(x_1 = L_3)}{dx} = \frac{dU_4(L_1 + L_2 + L_3)}{dx}$$

Thus on substitution:

$$\frac{dU_3(L_3)}{dx_1} = \frac{\left(C_5 e^{(\beta_2 L_3)} - C_6 e^{(-\beta_2 L_3)}\right)\beta_2 \cosh(\alpha_2 L_3) - \left(C_5 e^{(\beta_2 L_3)} + C_6 e^{-(\beta_2 L_3)}\right)\alpha_2 \sinh(\alpha_2 L_3)}{\cosh^2(\alpha_2 L_3)}$$

And

$$\frac{dU_4(L_1 + L_2 + L_3)}{dx} = -kC_7 \sin(k[L_1 + L_2 + L_3]) + kC_8 \cos(k[L_1 + L_2 + L_3])$$

Therefore combining the two above equation the following equation is formed:

$$\frac{dU_3(L_3)}{dx_1} = \frac{\left(C_5 e^{(\beta_2 L_3)} - C_6 e^{(-\beta_2 L_3)}\right)\beta_2 \cosh(\alpha_2 L_3) - \left(C_5 e^{(\beta_2 L_3)} + C_6 e^{(-\beta_2 L_3)}\right)\alpha_2 \sinh(\alpha_2 L_3)}{\cosh^2(\alpha_2 L_3)} = \frac{dU_4(L_1 + L_2 + L_3)}{dx} = -kC_7 \sin(k[L_1 + L_2 + L_3]) + kC_8 \cos(k[L_1 + L_2 + L_3])$$

$$\therefore \frac{\left(C_5 e^{(\beta_2 L_3)} - C_6 e^{(-\beta_2 L_3)}\right)\beta_2 \cosh(\alpha_2 L_3) - \left(C_5 e^{(\beta_2 L_3)} + C_6 e^{(-\beta_2 L_3)}\right)\alpha_2 \sinh(\alpha_2 L_3)}{\cosh^2(\alpha_2 L_3)} + kC_7 \sin(k[L_1 + L_2 + L_3]) - kC_8 \cos(k[L_1 + L_2 + L_3]) = 0$$

$$\therefore \frac{C_5 e^{(\beta_2 L_3)}(\beta_2 \cosh(\alpha_2 L_3) - \alpha_2 \sinh(\alpha_2 L_3)) - C_6 e^{(-\beta_2 L_3)}(\beta_2 \cosh(\alpha_2 L_3) + \alpha_2 \sinh(\alpha_2 L_3))}{\cosh^2(\alpha_2 L_3)} + kC_7 \sin(k[L_1 + L_2 + L_3]) - kC_8 \cos(k[L_1 + L_2 + L_3]) = 0 \quad \textbf{27}$$

A.6.4 Interface sections 4 and 5

Displacement

$$U_5(L_1 + L_2 + L_3 + L_4) = U_5(L_1 + L_2 + L_3 + L_4)$$

Consequently:

$$U_4(L_1 + L_2 + L_3 + L_4) = C_7 \cos(k[L_1 + L_2 + L_3 + L_4]) + C_8 \sin(k[L_1 + L_2 + L_3 + L_4])$$

And

$$U_5(L_1 + L_2 + L_3 + L_4) = C_9 \cos(k[L_1 + L_2 + L_3 + L_4]) + C_{10} \sin(k[L_1 + L_2 + L_3 + L_4])$$

Therefore combining the two above equation the following is obtained:

$$U_4(L_1 + L_2 + L_3 + L_4) = C_7 \cos(k[L_1 + L_2 + L_3 + L_4]) + C_8 \sin(k[L_1 + L_2 + L_3 + L_4]) = U_5(L_1 + L_2 + L_3 + L_4) = C_9 \cos(k[L_1 + L_2 + L_3 + L_4]) + C_{10} \sin(k[L_1 + L_2 + L_3 + L_4])$$

$$C_7 \cos(k[L_1 + L_2 + L_3 + L_4]) + C_8 \sin(k[L_1 + L_2 + L_3 + L_4]) - C_9 \cos(k[L_1 + L_2 + L_3 + L_4]) - C_{10} \sin(k[L_1 + L_2 + L_3 + L_4]) = 0 \quad \textbf{28}$$

Strain

$$\frac{dU_4(L_1+L_2+L_3+L_4)}{dx} = \frac{dU_5(L_1+L_2+L_3+L_4)}{dx}$$

Therefore:

$$U_4(L_1+L_2+L_3+L_4) = -C_7 k \sin(k[L_1+L_2+L_3+L_4]) + C_8 k \cos(k[L_1+L_2+L_3+L_4])$$

And

$$U_5(L_1+L_2+L_3+L_4) = -C_9 k \sin(k[L_1+L_2+L_3+L_4]) + C_{10} k \cos(k[L_1+L_2+L_3+L_4])$$

Therefore combining the two above equation the following is obtained:

$$U_4(L_1+L_2+L_3+L_4) = -C_7 k \sin(k[L_1+L_2+L_3+L_4]) + C_8 k \cos(k[L_1+L_2+L_3+L_4]) = U_5(L_1+L_2+L_3+L_4) = -C_9 k \sin(k[L_1+L_2+L_3+L_4]) + C_{10} k \cos(k[L_1+L_2+L_3+L_4])$$

$$\therefore -C_7 k \sin(k[L_1+L_2+L_3+L_4]) + C_8 k \cos(k[L_1+L_2+L_3+L_4]) + C_9 k \sin(k[L_1+L_2+L_3+L_4]) - C_{10} k \cos(k[L_1+L_2+L_3+L_4]) = 0 \quad \textbf{29}$$

A.6.5 Booster Interface

If the total length of the stack is defined to be, L, where:

$$L = L_1 + L_2 + L_3 + L_4 + L_5$$

Then the input displacement is the maximum displacement allowed by the Piezo-actuator which is assumed to be 10µm at 20 kHz. Thus the following boundary condition is present at the connection of the horn and the booster:

Displacement

$$U_5(L) = A_{input} = 10\mu m$$

As a result:

$$U_5(L) = C_9 \cos(k[L]) + C_{10} \sin(k[L]) = A_{input} \quad \textbf{30}$$

It can also be seen that since this point of connection is a point of zero location on the normal cylinder which represents the booster. This is the magnitude A_{input}, since the general solution for a constant cross sectional area is $U(x) = A_{input}\cos(kx)$.

Strain

$$\frac{dU_5(L)}{dx} = 10\mu\varepsilon$$

$$\frac{dU_5(L)}{dx} = -C_9 k \sin(k[L]) + C_{10} k \cos(k[L]) = A_{input} \quad \textbf{31}$$

Thus using these boundary conditions and equations: 14, 16, 17, 19 and 21 the constants can be found by setting up a set of liner equations of the form [A]{c}={B}, where:

$$\{c\} = \begin{Bmatrix} C_1 \\ C_2 \\ C_3 \\ C_4 \\ C_5 \\ C_6 \\ C_7 \\ C_8 \\ C_9 \\ C_{10} \end{Bmatrix} \qquad \{B\} = \begin{Bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ A_{input} \\ A_{input} \end{Bmatrix}$$

$$A=$$

$$\begin{vmatrix}
\frac{e^{(\beta_1 L_1)}}{\cosh(\alpha_1 L_1)} & \frac{e^{(-\beta_1 L_1)}}{\cosh(\alpha_1 L_1)} & -\cos(kL_1) & -\sin(kL_1) & 0 & 0 & 0 & 0 & 0 & 0 \\
\frac{e^{(\beta_1 L_1)}(\beta_1 \cosh(\alpha_1 L_1) - \alpha_1 \sinh(\alpha_1 L_1))}{\cosh^2(\alpha_1 L_1)} & -\frac{e^{(-\beta_1 L_1)}(\beta_1 \cosh(\alpha_1 L_1) + \alpha_1 \sinh(\alpha_1 L_1))}{\cosh^2(\alpha_1 L_1)} & k\sin(kL_1) & -k\cos(kL_1) & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & \cos(k(L_1+L_2)) & \sin(k(L_1+L_2)) & -1 & -1 & 0 & 0 & 0 & 0 \\
0 & 0 & -k\sin(k(L_1+L_2)) & k\cos(k(L_1+L_2)) & -\beta_2 & \beta_2 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & \frac{e^{(\beta_2 L_3)}}{\cosh(\alpha_2 L_3)} & \frac{e^{(-\beta_2 L_3)}}{\cosh(\alpha_2 L_3)} & -\cos(k(L_1+L_2+L_3)) & -\sin(k(L_1+L_2+L_3)) & 0 & 0 \\
0 & 0 & 0 & 0 & \frac{e^{(\beta_2 L_3)}(\beta_2 \cosh(\alpha_2 L_3) - \alpha_2 \sinh(\alpha_2 L_3))}{\cosh^2(\alpha_2 L_3)} & \frac{-e^{(-\beta_2 L_3)}(\beta_2 \cosh(\alpha_2 L_3) + \alpha_2 \sinh(\alpha_2 L_3))}{\cosh^2(\alpha_2 L_3)} & k\sin(k(L_1+L_2+L_3)) & -k\cos(k(L_1+L_2+L_3)) & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & \cos(k(L_1+L_2+L_3+L_4)) & \sin(k(L_1+L_2+L_3+L_4)) & -\cos(k(L_1+L_2+L_3+L_4)) & -\sin(k(L_1+L_2+L_3+L_4)) \\
0 & 0 & 0 & 0 & 0 & 0 & -k\sin(k(L_1+L_2+L_3+L_4)) & k\cos(k(L_1+L_2+L_3+L_4)) & k\sin(k(L_1+L_2+L_3+L_4)) & -k\cos(k(L_1+L_2+L_3+L_4)) \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \cos(k(L)) & \sin(k(L)) \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -k\sin(k(L)) & k\cos(k(L))
\end{vmatrix}$$

Thus using these equations with a fixed geometry and implementing in Matlab, the following plots are obtained:

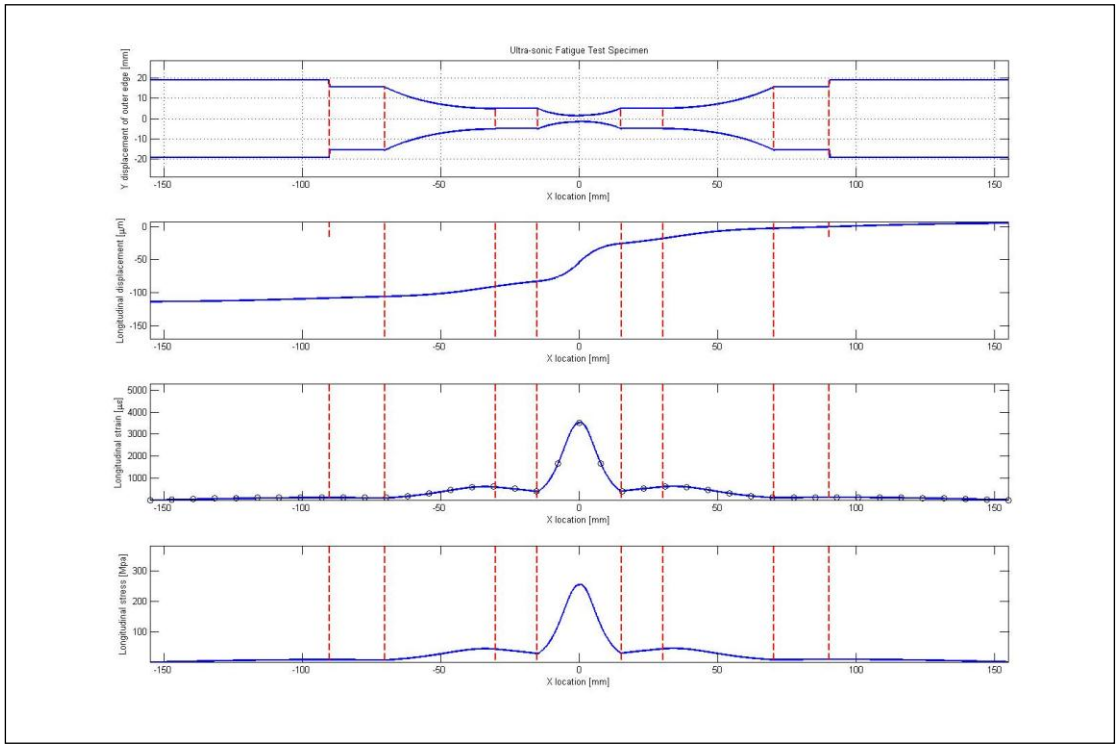


Figure A-3: Plot for 5 section specimen model fixed geometry

The above plot was obtained for the following geometric parameters:

L1=15mm
R1=1.5mm
R2=5.0mm
R3=15.5mm
R4=19.0mm
L5=64.8mm
L3=40mm
L4=20mm
L2=15.3mm

If the geometrical length of L2 is allowed to vary to get the displacement at the smallest cross sectional area to be zero then the following is obtained:

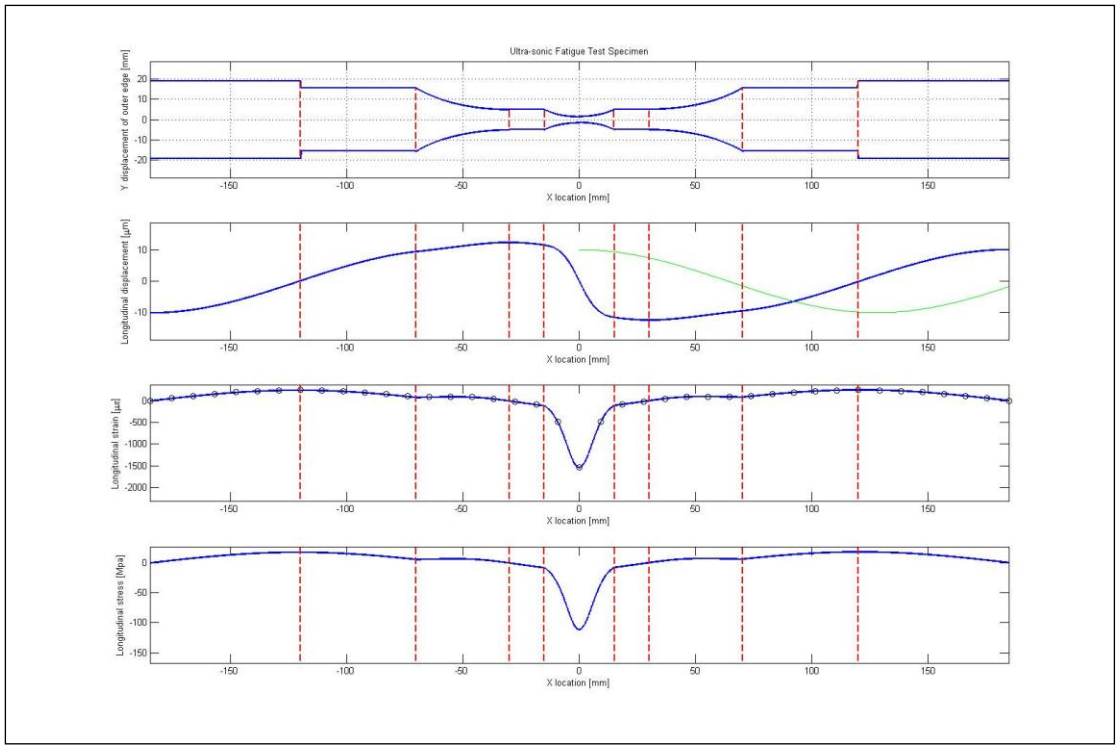


Figure A-4: Plot for 5 section specimen model variable geometry

The above plot was obtained for the following geometric parameters:

L1=15mm
R1=1.5mm
R2=5.0mm
R3=15.5mm
R4=19.0mm
L5=64.8mm
L3=40mm
L4=49.6mm
L2=15.2mm

APPENDIX B - MEASUREMENT DEVICES

B.1 Laser Vibrometer

Laser vibrometers have become fundamental tools in the measurement of vibration within the engineering community. Laser vibrometers implement the Doppler-effect to sense the vibration of the moving component being monitored. The Doppler-effect determines that if a moving object is measured using a laser, then the phase shift is given by [25]:

$$f_D = 2\frac{v}{\lambda} \quad \textbf{32}$$

Where

v - is the velocity of the object
λ - is the wave length of the emitted laser wave

The Laser-Doppler vibrometer works on the basis of optical interference, requiring two coherent light beams with respective light intensities defined by I_1 and I_2 to overlap. The resulting intensity is the sum or the single intensities plus an interference factor as define by [25]:

$$I_{tot} = I_1 + I_2 + 2\sqrt[2]{I_1 I_2 cos(2\pi[r_1 - r_2])} \quad \textbf{33}$$

Equation 33 is used in combination with the following setup to obtain the velocity of the body.

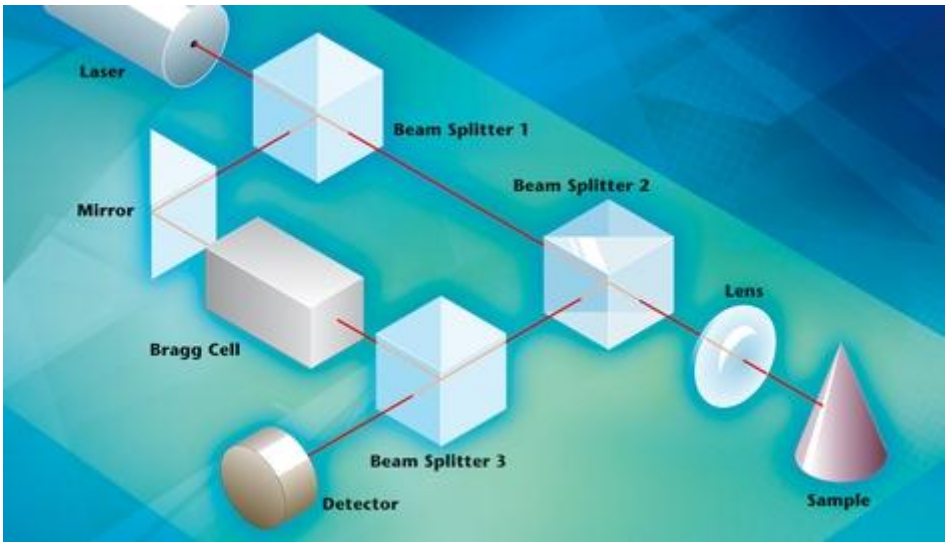


Figure B-1 : Laser vibrometer internal setup [25]

The beam from the laser, generally a helium neon laser is split via two beam splitters as indicated above by beam splitter 1 and 2. The beam split by the first splitter becomes the reference beam and the measurement beam is projected onto the second splitter [25].

The second beam is then yet again split and one part of this beam is focused onto the object and the other is reflected back to the system. The object needs to be cable of reflecting the laser beam or a

special reflective paper may be used to improve the body's reflectivity if it is poor. This reflected beam is then returned and merged with the reference beam by beam splitter 3 [25].

As the path length of the reference beam is constant over time (with the exception of negligible thermal effects on the interferometer) this implies that r_2 is constant of equation 33. Hence a movement of the object under investigation will define that r_1 is a function of time. This will then generate a dark and light fringe pattern typical of interferometry on the detector by varying the output intensity of the combined laser beam. One complete dark and bright cycle on the detector corresponds to an object displacement of exactly half of the wavelength of the light source used. In the case of the typical helium neon laser this would corresponds to a displacement of 316 nm [25].

Changing the optical path length per unit of time manifests itself as the Doppler frequency shift of the measurement beam. This means that the modulation frequency of the interferometer pattern determined, is directly proportional to the velocity of the object. An object movement away from the interferometer generates the same interference pattern (and frequency shift) as object movement towards the interferometer. Thus this setup cannot determine the direction the object is moving in. For this purpose, an acousto-optic modulator (Bragg cell) is placed in the reference beam, which shifts the light frequency by 40 MHz (by comparison, the typical frequency of the laser light is 4.74 x 10^{14} Hz). This generates a modulation frequency of the fringe pattern of 40 MHz when the object is at rest. If the object then moves towards the interferometer, this modulation frequency is reduced and if it moves away from the vibrometer, the detector receives a frequency higher than 40 MHz. This means that it is now possible not only to detect the amplitude of movement but also to clearly define the direction of movement [25].

In principle the laser Doppler Vibrometer can directly measure the displacement as well as the velocity of the object under investigation. In the case of displacement the system measures the fringes of dark and light spots on the detector and interpolates to obtain a displacement [25].

The displacement can also be obtained directly from the velocity of the signal. Thus for a perfect sinusoidal velocity signal the velocity may be defined by:

$$v = Acos(wt) \qquad \textbf{34}$$

Where

$$w = 2\pi f$$

Thus the displacement amplitude can be found using simple integration

$$d = \frac{A}{w} sin(wt) = \frac{A}{w} cos\left(\frac{\pi}{2} - wt\right) \qquad \textbf{35}$$

The displacement amplitude is a factor of the velocity amplitude by the operating circular frequency, ω, and the signal is 90° out of phase with the velocity signal.

B.2 Strain Gauges

B.2.1 Background

The strain gauge is the most common device used to measure strain. Two typical strain gauges are available, the wire gage and the etched foil gage [5]. In this dissertation the strain gauge used was that of the foil variety. The typical strain gauge is defined by:

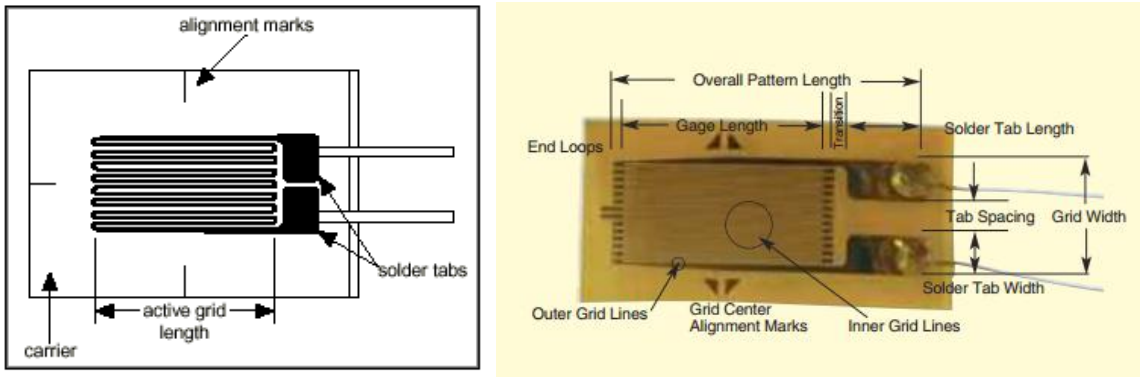


Figure B-2: Typical metallic bonded strain gauge [26] [27]

The typical gauge configuration used in the measurement of the vibration will be a full bridge configuration as indicated in Figure B-3.

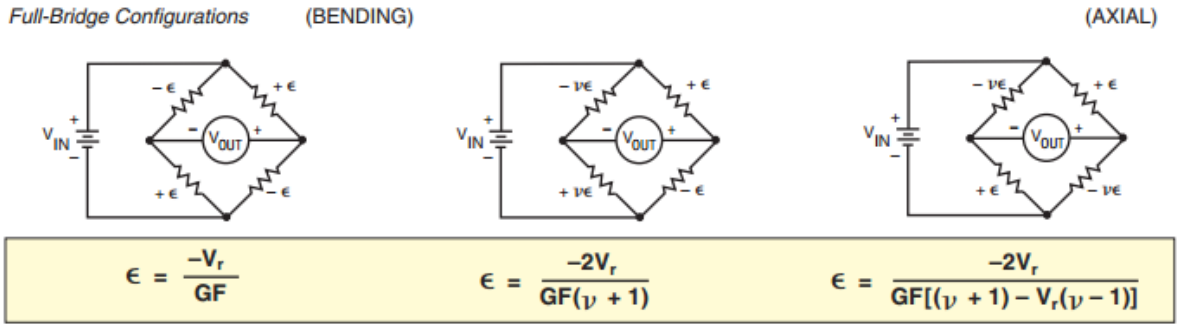


Figure B-3: Full bridge configuration [27]

Where V_r is defined as:

$$V_r = \left[\left(\frac{V_{OUT}}{V_{in}}\right)_{strained} - \left(\frac{V_{OUT}}{V_{in}}\right)_{unstrained}\right] \qquad \textbf{36}$$

Thus if $\left(\frac{V_{OUT}}{V_{in}}\right)_{unstrained} = 0$ i.e the bridge was balanced at the start, then this implies that the equations for the three full bridge strain combinations above become:

$$\frac{V_o}{V_{ex}} = \epsilon . GF \qquad \frac{V_o}{V_{ex}} = \frac{\epsilon . GF}{2} \qquad \frac{V_o}{V_{ex}} = \frac{\epsilon . GF(v+1)}{\epsilon . GF(v-1) - 2}$$

In the case of the UHCF test specimen the load will be only tension and compression. Thus the following full bridge configuration is valid:

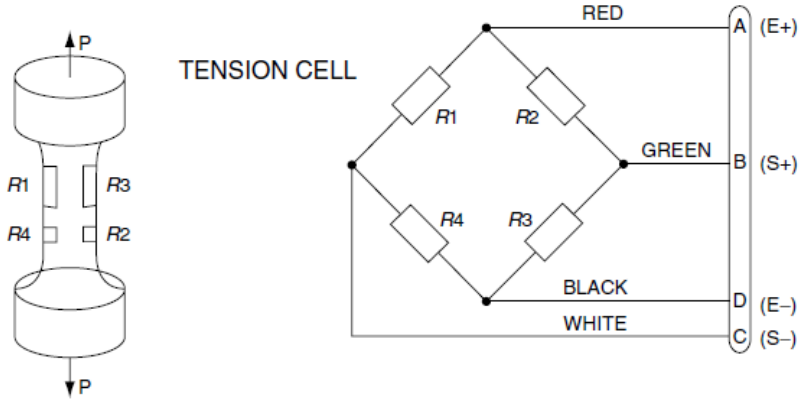


Figure B-4: Tension configuration [5]

The resulting bridge output is:

$$\frac{\Delta V_o}{V_{ex}} = \frac{\epsilon . GF(1+v)}{2}$$

To minimise the effect of temperature on the strain gauges a transverse gauge can be used to maintain the ratio of resistances during expansion or contraction of the specimen due to temperature fluctuations. A typical arrangement is defined by:

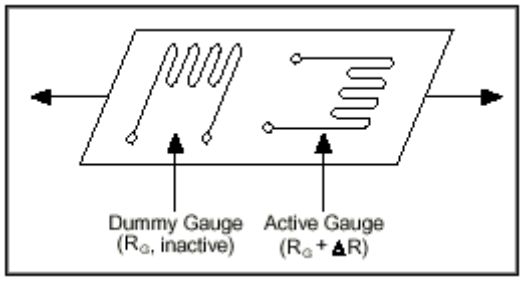


Figure B-5: Use of dummy gauge to eliminate temperature affects [26].

The effect of the dummy gauge can be explain by the fact that any change in resistance ΔR due to expansion of the base material under load is only generated in the active gauge with the dummy gauge experiencing a negligible effect. If the temperature rises or falls the material is assumed to expand or contract uniformly. Thus both gauges will be affected equally. Hence the ratio equations $\frac{R_1}{R_2} = \frac{R_4}{R_3}$, see Figure B-4, will remain the same if either bridge or both bridges are in a dummy and active gauge combination.

Using the same configuration as for temperature compensation also compensates for any Poisson's affect that may occur, thus in this dissertation this gauge setup was implemented to minimise both the Poisson's affect and the temperature affects.

B.2.2 Shunt Calibration

To verify the strain being measured, shunt calibration may be implemented. This is when a known resistance is placed across any arm in the bridge. Thus the expected strain for the change in resistance can be verified. A typical shunt circuit is given by:

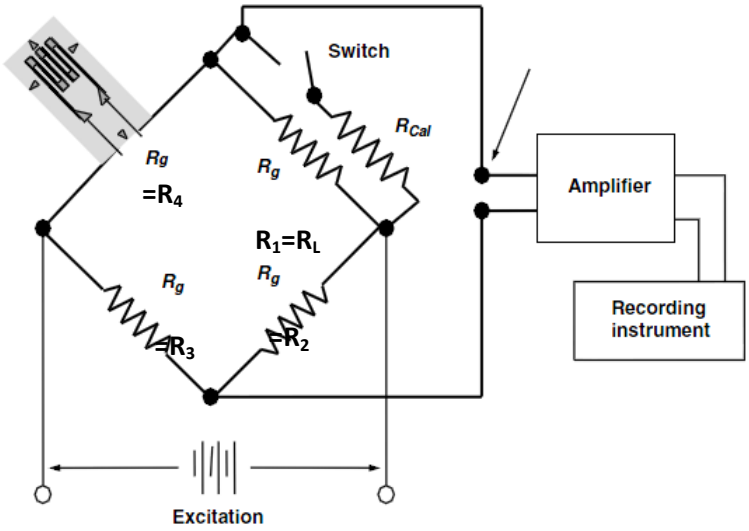


Figure B-6: Shunt resistor across R3 [27]

The calibration formula is defined by [27]:

$$\frac{\Delta E}{Vex} = \frac{R_4}{R_1 + R_4} - \frac{R_3}{R_2 + R_3}$$

ΔE is the change from balance (zero voltage) to the new value.

But if R_2=R_L, then from the sum of parallel resistors:

$$R_{parrellel} = \frac{\mathrm{R}_A \mathrm{R}_B}{\mathrm{R}_A + \mathrm{R}_B}$$

$$\therefore R_L = \frac{\mathrm{R}_g \mathrm{R}_C}{\mathrm{R}_g + \mathrm{R}_C}$$

Simplifying:

$$\frac{\Delta E}{Vex} = \frac{R_\mathrm{g}}{\frac{\mathrm{R}_g \mathrm{R}_C}{\mathrm{R}_g + \mathrm{R}_C} + R_g} - \frac{R_\mathrm{g}}{2R_g}$$

$$\frac{\Delta E}{Vex} = \frac{R_\mathrm{g} + \mathrm{R}_C}{2\mathrm{R}_C + R_g} - \frac{1}{2}$$

$$\frac{\Delta E}{Vex} = \frac{R_{\mathrm{g}}}{4\mathrm{R}_C + 2R_g}$$

In this project a HBM shunt calibrator unit was used as indicated Figure B-4. The shunt calibration unit was re-calibrated prior to calibration on the 02/09/2014



Figure B-7: HBM shunt calibration unit

APPENDIX C - FINITE ELEMENT FORMULATION

This appendix deals with the derivation of the finite elements used in the Matlab code. It begins with the standard 2D element formulation which aids in the understanding of the final 2D axisymmetric element formulation. Using this implementation of 2D elements the 2D axisymmetric element solution is defined.

C.1 QUAD4

C.1.1 Consistent Mass Matrix Q4

To derive the consistent mass matrix for the 4 node quadrilateral (Q4) indicated in Figure C-1, it is required that the shape functions be defined. It is assumed that the parametric shape functions derived in reference can be implemented [28].

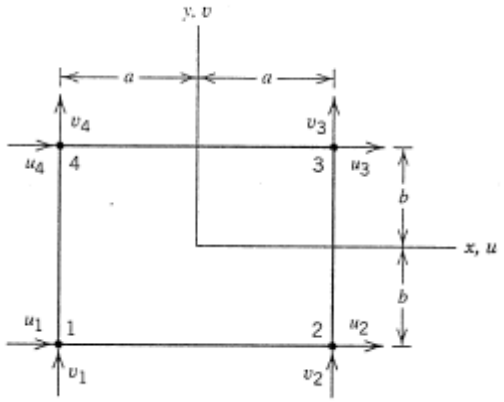


Figure C-1: Four node quadrilateral (Q4) [28]

If only the Isoperimetric element is considered as indicated in the figure below:

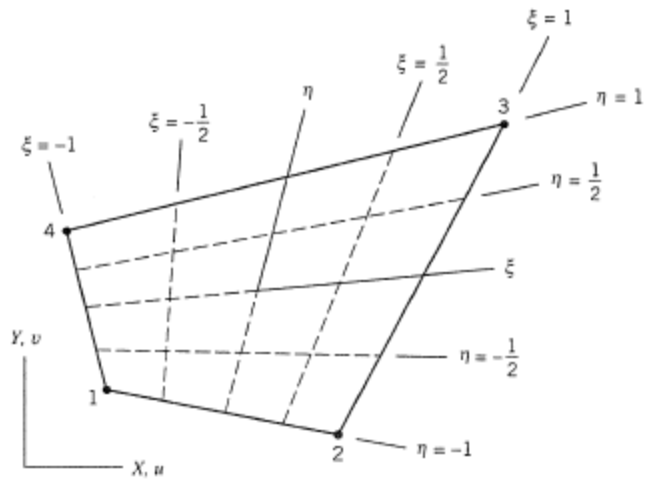


Figure C-2: Four node Isoperimetric quadrilateral (Q4) [28]

Then the four shape functions are defined by [28]:

$$N_1 = \frac{1}{4}(1-\xi)(1-\eta)$$
$$N_2 = \frac{1}{4}(1+\xi)(1-\eta)$$
$$N_3 = \frac{1}{4}(1+\xi)(1+\eta)$$
$$N_4 = \frac{1}{4}(1-\xi)(1+\eta) \quad \textbf{37}$$

The general formula for a consistent mass matrix in 3D is:

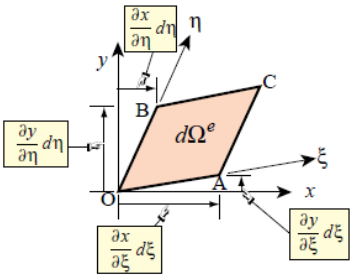
$$[m] = \int_{Ve} [N]^T [N] dV \quad \textbf{38}$$

Thus the consistent mass matrix for a 2D elements assuming constant thickness, t, density, ρ and area, is found using the formula below:

$$[m] = \int_{-y_1}^{y_2} \int_{-x_1}^{x_2} [N]^T [N] \rho t dy dx \quad \textbf{39}$$

To justify the transformation of dA the following is taken from IFEM.CH17 [29]

$$dA = \vec{OB} \times \vec{OA} = \frac{\partial x}{\partial \xi} d\xi \, \frac{\partial y}{\partial \eta} d\eta - \frac{\partial x}{\partial \eta} d\eta \, \frac{\partial y}{\partial \xi} d\xi$$
$$= \begin{vmatrix} \frac{\partial x}{\partial \xi} & \frac{\partial x}{\partial \eta} \\ \frac{\partial y}{\partial \xi} & \frac{\partial y}{\partial \eta} \end{vmatrix} d\xi \, d\eta = |\mathbf{J}| \, d\xi \, d\eta = \det \mathbf{J} \, d\xi \, d\eta.$$



If it is required to transform the above equation into Isoparametric then the determinant is required, thus the following is obtained:

$$[m] = \int_{-1}^{1} \int_{-1}^{1} [N]^T [N] \rho t \det(J) d\xi d\eta \quad \textbf{40}$$

Where

det(J)=A/4 (assume rectangle or parallelograms [28])

The mass matrix must satisfy the following Newtonian equation:

$$[m]\begin{Bmatrix}\ddot{u}_1\\ \ddot{v}_1\\ \ddot{u}_2\\ \ddot{v}_2\\ \ddot{u}_3\\ \ddot{v}_3\\ \ddot{u}_4\\ \ddot{v}_4\end{Bmatrix}=\{F\} \qquad \textbf{41}$$

Consequently it can be seen that for a four node quadrilateral (Q4) with two degrees of freedom per node, the required dimensions of the consistent mass matrix need to be [8x8].

The general shape function matrix is given by:

$$[N]=\begin{bmatrix}N_1 & 0 & N_2 & 0 & N_3 & 0 & N_4 & 0\\ 0 & N_1 & 0 & N_2 & 0 & N_3 & 0 & N_4\end{bmatrix} \qquad \textbf{42}$$

The inner terms of the above expression are given by equation 37. Thus the following is obtained:

$$[m]=\int_{-1}^{1}\int_{-1}^{1}\begin{bmatrix}N_1 & 0\\ 0 & N_1\\ N_2 & 0\\ 0 & N_2\\ N_3 & 0\\ 0 & N_3\\ N_4 & 0\\ 0 & N_4\end{bmatrix}\begin{bmatrix}N_1 & 0 & N_2 & 0 & N_3 & 0 & N_4 & 0\\ 0 & N_1 & 0 & N_2 & 0 & N_3 & 0 & N_4\end{bmatrix}\rho\frac{A}{4}td\xi d\eta \qquad \textbf{43}$$

Simplifying:

$$[m]=\int_{-1}^{1}\int_{-1}^{1}\begin{bmatrix}N_1N_1 & 0 & N_1N_2 & 0 & N_1N_3 & 0 & N_1N_4 & 0\\ 0 & N_1N_1 & 0 & N_1N_2 & 0 & N_1N_3 & 0 & N_1N_4\\ N_2N_1 & 0 & N_2N_2 & 0 & N_2N_3 & 0 & N_2N_4 & 0\\ 0 & N_2N_1 & 0 & N_2N_2 & 0 & N_2N_3 & 0 & N_2N_4\\ N_3N_1 & 0 & N_3N_2 & 0 & N_3N_3 & 0 & N_3N_4 & 0\\ 0 & N_3N_1 & 0 & N_3N_2 & 0 & N_3N_3 & 0 & N_3N_4\\ N_4N_1 & 0 & N_4N_2 & 0 & N_4N_3 & 0 & N_4N_4 & 0\\ 0 & N_4N_1 & 0 & N_4N_2 & 0 & N_4N_3 & 0 & N_4N_4\end{bmatrix}\rho\frac{A}{4}td\xi d\eta \qquad \textbf{44}$$

If it is assumed that the density, elemental area and thickness remain constant across the element, then the integral simplifies too:

$$[m]=\rho\frac{A}{4}t\int_{-1}^{1}\int_{-1}^{1}\begin{bmatrix} N_1N_1 & 0 & N_1N_2 & 0 & N_1N_3 & 0 & N_1N_4 & 0 \\ 0 & N_1N_1 & 0 & N_1N_2 & 0 & N_1N_3 & 0 & N_1N_4 \\ N_2N_1 & 0 & N_2N_2 & 0 & N_2N_3 & 0 & N_2N_4 & 0 \\ 0 & N_2N_1 & 0 & N_2N_2 & 0 & N_2N_3 & 0 & N_2N_4 \\ N_3N_1 & 0 & N_3N_2 & 0 & N_3N_3 & 0 & N_3N_4 & 0 \\ 0 & N_3N_1 & 0 & N_3N_2 & 0 & N_3N_3 & 0 & N_3N_4 \\ N_4N_1 & 0 & N_4N_2 & 0 & N_4N_3 & 0 & N_4N_4 & 0 \\ 0 & N_4N_1 & 0 & N_4N_2 & 0 & N_4N_3 & 0 & N_4N_4 \end{bmatrix}d\xi d\eta \qquad \textbf{45}$$

Substituting in the shape functions the following is obtained:

$$[m]=\frac{\rho At}{64}\int_{-1}^{1}\int_{-1}^{1}\begin{bmatrix} (1-\xi)^2(1-\eta)^2 & 0 & (1-\xi^2)(1-\eta)^2 & 0 & (1-\xi^2)(1-\eta^2) & 0 & (1-\xi)^2(1-\eta^2) & 0 \\ 0 & (1-\xi)^2(1-\eta)^2 & 0 & (1-\xi^2)(1-\eta)^2 & 0 & (1-\xi^2)(1-\eta^2) & 0 & (1-\xi)^2(1-\eta^2) \\ (1-\xi^2)(1-\eta)^2 & 0 & (1+\xi)^2(1-\eta)^2 & 0 & (1+\xi)^2(1-\eta^2) & 0 & (1-\xi^2)(1-\eta^2) & 0 \\ 0 & (1-\xi^2)(1-\eta)^2 & 0 & (1+\xi)^2(1-\eta)^2 & 0 & (1+\xi)^2(1-\eta^2) & 0 & (1-\xi^2)(1-\eta^2) \\ (1-\xi^2)(1-\eta^2) & 0 & (1+\xi)^2(1-\eta^2) & 0 & (1+\xi)^2(1+\eta)^2 & 0 & (1-\xi^2)(1+\eta)^2 & 0 \\ 0 & (1-\xi^2)(1-\eta^2) & 0 & (1+\xi)^2(1-\eta^2) & 0 & (1+\xi)^2(1+\eta)^2 & 0 & (1-\xi^2)(1+\eta)^2 \\ (1-\xi)^2(1-\eta^2) & 0 & (1-\xi^2)(1-\eta^2) & 0 & (1-\xi^2)(1+\eta)^2 & 0 & (1-\xi)^2(1+\eta)^2 & 0 \\ 0 & (1-\xi)^2(1-\eta^2) & 0 & (1-\xi^2)(1-\eta^2) & 0 & (1-\xi^2)(1+\eta)^2 & 0 & (1-\xi)^2(1+\eta)^2 \end{bmatrix}d\xi d\eta \qquad \textbf{46}$$

Integrating with respect to ξ and simplify:

$$[m]=\frac{A\rho t}{64}\int_{-1}^{1}\begin{bmatrix} \frac{8}{3}(1-\eta)^2 & 0 & \frac{4}{3}(1-\eta)^2 & 0 & \frac{4}{3}(1-\eta^2) & 0 & \frac{8}{3}(1-\eta^2) & 0 \\ 0 & \frac{8}{3}(1-\eta)^2 & 0 & \frac{4}{3}(1-\eta)^2 & 0 & \frac{4}{3}(1-\eta^2) & 0 & \frac{8}{3}(1-\eta^2) \\ \frac{4}{3}(1-\eta)^2 & 0 & \frac{8}{3}(1-\eta)^2 & 0 & \frac{8}{3}(1-\eta^2) & 0 & \frac{4}{3}(1-\eta^2) & 0 \\ 0 & \frac{4}{3}(1-\eta)^2 & 0 & \frac{8}{3}(1-\eta)^2 & 0 & \frac{8}{3}(1-\eta^2) & 0 & \frac{4}{3}(1-\eta^2) \\ \frac{4}{3}(1-\eta^2) & 0 & \frac{8}{3}(1-\eta^2) & 0 & \frac{8}{3}(1+\eta)^2 & 0 & \frac{4}{3}(1+\eta)^2 & 0 \\ 0 & \frac{4}{3}(1-\eta^2) & 0 & \frac{8}{3}(1-\eta^2) & 0 & \frac{8}{3}(1+\eta)^2 & 0 & \frac{4}{3}(1+\eta)^2 \\ \frac{8}{3}(1-\eta^2) & 0 & \frac{4}{3}(1-\eta^2) & 0 & \frac{4}{3}(1+\eta)^2 & 0 & \frac{8}{3}(1+\eta)^2 & 0 \\ 0 & \frac{8}{3}(1-\eta^2) & 0 & \frac{4}{3}(1-\eta^2) & 0 & \frac{4}{3}(1+\eta)^2 & 0 & \frac{8}{3}(1+\eta)^2 \end{bmatrix}d\eta \qquad \textbf{47}$$

After the final integration the following is obtained:

$$[m]=\frac{A\rho t}{64}\begin{bmatrix}\frac{8}{3}\left(\frac{8}{3}\right) & 0 & \frac{4}{3}\left(\frac{8}{3}\right) & 0 & \frac{4}{3}\left(\frac{4}{3}\right) & 0 & \frac{8}{3}\left(\frac{4}{3}\right) & 0\\ 0 & \frac{8}{3}\left(\frac{8}{3}\right) & 0 & \frac{4}{3}\left(\frac{8}{3}\right) & 0 & \frac{4}{3}\left(\frac{4}{3}\right) & 0 & \frac{8}{3}\left(\frac{4}{3}\right)\\ \frac{4}{3}\left(\frac{8}{3}\right) & 0 & \frac{8}{3}\left(\frac{8}{3}\right) & 0 & \frac{8}{3}\left(\frac{4}{3}\right) & 0 & \frac{4}{3}\left(\frac{4}{3}\right) & 0\\ 0 & \frac{4}{3}\left(\frac{8}{3}\right) & 0 & \frac{8}{3}\left(\frac{8}{3}\right) & 0 & \frac{8}{3}\left(\frac{4}{3}\right) & 0 & \frac{4}{3}\left(\frac{4}{3}\right)\\ \frac{4}{3}\left(\frac{4}{3}\right) & 0 & \frac{8}{3}\left(\frac{4}{3}\right) & 0 & \frac{8}{3}\left(\frac{8}{3}\right) & 0 & \frac{4}{3}\left(\frac{8}{3}\right) & 0\\ 0 & \frac{4}{3}\left(\frac{4}{3}\right) & 0 & \frac{8}{3}\left(\frac{4}{3}\right) & 0 & \frac{8}{3}\left(\frac{8}{3}\right) & 0 & \frac{4}{3}\left(\frac{8}{3}\right)\\ \frac{8}{3}\left(\frac{4}{3}\right) & 0 & \frac{4}{3}\left(\frac{4}{3}\right) & 0 & \frac{4}{3}\left(\frac{8}{3}\right) & 0 & \frac{8}{3}\left(\frac{8}{3}\right) & 0\\ 0 & \frac{8}{3}\left(\frac{4}{3}\right) & 0 & \frac{4}{3}\left(\frac{4}{3}\right) & 0 & \frac{4}{3}\left(\frac{8}{3}\right) & 0 & \frac{8}{3}\left(\frac{8}{3}\right)\end{bmatrix} \quad \textbf{48}$$

Simplifying:

$$[m]=\frac{A\rho t}{36}\begin{bmatrix}4&0&2&0&1&0&2&0\\0&4&0&2&0&1&0&2\\2&0&4&0&2&0&1&0\\0&2&0&4&0&2&0&1\\1&0&2&0&4&0&2&0\\0&1&0&2&0&4&0&2\\2&0&1&0&2&0&4&0\\0&2&0&1&0&2&0&4\end{bmatrix} \quad \textbf{49}$$

But for a square or rectangle At=V, where V is the volume. Thus the above equation simplifies too:

$$[m]=\frac{m}{36}\begin{bmatrix}4&0&2&0&1&0&2&0\\0&4&0&2&0&1&0&2\\2&0&4&0&2&0&1&0\\0&2&0&4&0&2&0&1\\1&0&2&0&4&0&2&0\\0&1&0&2&0&4&0&2\\2&0&1&0&2&0&4&0\\0&2&0&1&0&2&0&4\end{bmatrix} \quad \textbf{50}$$

C.1.2 Lumped Mass Matrix Q4

To obtain the lumped mass matrix row summation is implemented. The method is to sum each of the consistent mass matrices rows and place the row value on the diagonal. Therefore considering the consistent mass matrix:

$$[m]=\frac{m}{36}\begin{bmatrix}4&0&2&0&1&0&2&0\\0&4&0&2&0&1&0&2\\2&0&4&0&2&0&1&0\\0&2&0&4&0&2&0&1\\1&0&2&0&4&0&2&0\\0&1&0&2&0&4&0&2\\2&0&1&0&2&0&4&0\\0&2&0&1&0&2&0&4\end{bmatrix} \quad \textbf{51}$$

Now summing each row and placing it on the diagonal the following is obtained:

$$[m]=\frac{m}{36}\begin{bmatrix}9&0&0&0&0&0&0&0\\0&9&0&0&0&0&0&0\\0&0&9&0&0&0&0&0\\0&0&0&9&0&0&0&0\\0&0&0&0&9&0&0&0\\0&0&0&0&0&9&0&0\\0&0&0&0&0&0&9&0\\0&0&0&0&0&0&0&9\end{bmatrix} \quad \textbf{52}$$

This simplifies too:

$$[m]=\frac{m}{4}\begin{bmatrix}1&0&0&0&0&0&0&0\\0&1&0&0&0&0&0&0\\0&0&1&0&0&0&0&0\\0&0&0&1&0&0&0&0\\0&0&0&0&1&0&0&0\\0&0&0&0&0&1&0&0\\0&0&0&0&0&0&1&0\\0&0&0&0&0&0&0&1\end{bmatrix} \quad \textbf{53}$$

This makes sense since the Q4 quadrilateral has 4 nodes, thus if the mass is distributed evenly over the four nodes then the mass at each node would be a quarter of the total mass. This also assumes that all the degrees of freedom are displacements and not rotations else the mass matrix should have rotational inertias in it.

C.1.3 General Implementation of Q4FI

The general shape function equations of a bilinear quadrilateral are given in equation 3.6-4 on pg. 97 of reference [28] for the following generic quad 4 element indicated in the figure below:

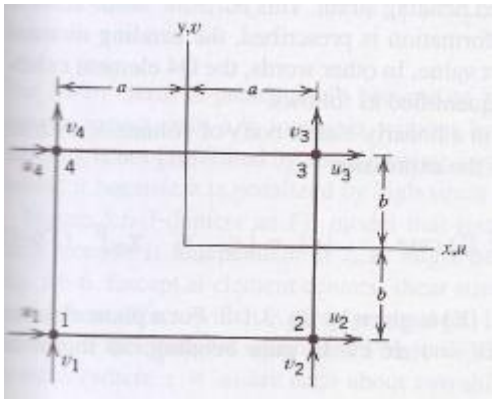


Figure C-3: Bilinear quadrilateral Q4 [28]

These general shape functions are defined:

$$N_1 = \frac{(a-x)(b-y)}{4ab}$$

$$N_2 = \frac{(a+x)(b-y)}{4ab} \qquad \textbf{54}$$

$$N_3 = \frac{(a+x)(b+y)}{4ab}$$

$$N_4 = \frac{(a-x)(b+y)}{4ab}$$

For Isoperimetric elements the following is applicable:

$a = 1$

$b = 1$

$x = \xi$

$y = \eta$

Then the general shape functions for an Isoperimetric bilateral quadrilateral are:

$$N_1 = \frac{1}{4}(1-\xi)(1-\eta)$$

$$N_2 = \frac{1}{4}(1+\xi)(1-\eta) \qquad \textbf{55}$$

$$N_3 = \frac{1}{4}(1+\xi)(1+\eta)$$

$$N_4 = \frac{1}{4}(1-\xi)(1+\eta)$$

However it is known that for an Isoperimetric element the same shape functions are used to interpolate for both the x, y system as the ξ, η system. Thus using equation 6.2-5 of reference [28] the Jacobian matrix is obtained:

$$[J] = \begin{bmatrix} x,_\xi & y,_\xi \\ x,_\eta & y,_\eta \end{bmatrix} = \begin{bmatrix} \sum N_{i,\xi} x_i & \sum N_{i,\xi} y_i \\ \sum N_{i,\eta} x_i & \sum N_{i,\eta} y_i \end{bmatrix} \qquad \textbf{56}$$

The first term of the matrix above is defined by:

J11

$$\sum N_{i,\xi} x_i = \frac{\partial}{\partial \xi}\left[\frac{1}{4}(1-\xi)(1-\eta)\right]x_1 + \frac{\partial}{\partial \xi}\left[\frac{1}{4}(1+\xi)(1-\eta)\right]x_2 + \frac{\partial}{\partial \xi}\left[\frac{1}{4}(1+\xi)(1+\eta)\right]x_3 + \frac{\partial}{\partial \xi}\left[\frac{1}{4}(1-\xi)(1+\eta)\right]x_4$$

$$\therefore \sum N_{i,\xi} x_i = \left[\frac{-1}{4}(1-\eta)\right]x_1 + \left[\frac{1}{4}(1-\eta)\right]x_2 + \left[\frac{1}{4}(1+\eta)\right]x_3 + \left[\frac{-1}{4}(1+\eta)\right]x_4 \quad \textbf{57}$$

Similarly, **J12**

$$\therefore \sum N_{i,\xi} y_i = \left[\frac{-1}{4}(1-\eta)\right]y_1 + \left[\frac{1}{4}(1-\eta)\right]y_2 + \left[\frac{1}{4}(1+\eta)\right]y_3 + \left[\frac{-1}{4}(1+\eta)\right]y_4 \quad \textbf{58}$$

The second row of derivative is done with respect to η, consequently the following is obtained:

J21

$$\sum N_{i,\eta} x_i = \frac{\partial}{\partial \eta}\left[\frac{1}{4}(1-\xi)(1-\eta)\right]x_1 + \frac{\partial}{\partial \eta}\left[\frac{1}{4}(1+\xi)(1-\eta)\right]x_2 + \frac{\partial}{\partial \eta}\left[\frac{1}{4}(1+\xi)(1+\eta)\right]x_3 + \frac{\partial}{\partial \eta}\left[\frac{1}{4}(1-\xi)(1+\eta)\right]x_4$$

$$\therefore \sum N_{i,\eta} x_i = \left[\frac{-1}{4}(1-\xi)\right]x_1 + \left[\frac{-1}{4}(1+\xi)\right]x_2 + \left[\frac{1}{4}(1+\xi)\right]x_3 + \left[\frac{1}{4}(1-\xi)\right]x_4 \quad \textbf{59}$$

Similarly,

J22

$$\therefore \sum N_{i,\eta} y_i = \left[\frac{-1}{4}(1-\xi)\right]y_1 + \left[\frac{-1}{4}(1+\xi)\right]y_2 + \left[\frac{1}{4}(1+\xi)\right]y_3 + \left[\frac{1}{4}(1-\xi)\right]y_4 \quad \textbf{60}$$

This can then be written in matrix form as:

$$\therefore [J] = \frac{1}{4}\begin{bmatrix} -1+\eta & 1-\eta & 1+\eta & -1-\eta \\ -1+\xi & -1-\xi & 1+\xi & 1-\xi \end{bmatrix}\begin{bmatrix} x_1 & y_1 \\ x_2 & y_2 \\ x_3 & y_3 \\ x_4 & y_4 \end{bmatrix} = \begin{bmatrix} J_{11} & J_{12} \\ J_{21} & J_{22} \end{bmatrix} \quad \textbf{61}$$

The above Jacobian is implemented in the Quad4 code via the following code; this code also calculates the determinant of the Jacobian as well as the Jacobian inverse:

```
%set location of Gauss points.
%Loop for each quadrant
xi  = (-1)^iGauss*Gauss_pos;
eta = (-1)^jGauss*Gauss_pos;
D = 0.25*[ -1.0+eta 0  1.0-eta 0 1.0+eta 0 -1.0-eta 0
          -1.0+xi  0 -1.0-xi  0 1.0+xi  0  1.0-xi  0
           0 -1.0+eta 0  1.0-eta 0 1.0+eta 0 -1.0-eta
           0 -1.0+xi  0 -1.0-xi  0 1.0+xi  0  1.0-xi ];
J = D(1:2,1:2:7)*[x1 y1
                  x2 y2
                  x3 y3
                  x4 y4];
detJ = J(1,1)*J(2,2) - J(2,1)*J(1,2);
invJ = 1.0/detJ * [J(2,2)  -J(1,2)
                  -J(2,1)   J(1,1)];
```

The total potential energy of a system consists of the strain energy in all the components less the work done to the component. This is indicated in the following equation:

$$\Pi = U - W \qquad \textbf{62}$$

Where

U - Strain Energy
W - Work done
Π - Total Potential Energy

If the body is assumed to be linearly elastic then the following is valid:

$$\Pi = \frac{1}{2}\int_{v} \langle \varepsilon \rangle [D] \langle \varepsilon \rangle dV - \int_{v} \langle u \rangle \{F\} dV - \int_{S_T} \langle u \rangle \{T\} dS - \langle u_{node} \rangle \{P\} \qquad \textbf{63}$$

Where

P - Is the concentrated forces

T - Is external tractions

F - Is the body forces

u_{node} - Is the displacement or rotations at the node points

$$\{u\} = \begin{Bmatrix} u(x,y,z) \\ v(x,y,z) \\ w(x,y,z) \end{Bmatrix}$$

However for plane stress and strain the out of plane displacements, w(x,y,z) are negligible consequently:

$$\{u\} = \begin{Bmatrix} u(x,y,z) \\ v(x,y,z) \end{Bmatrix}$$

For a 3D strain field the following is applicable:

$$\{\varepsilon\}=\begin{Bmatrix}\varepsilon_{xx}\\ \varepsilon_{yy}\\ \varepsilon_{zz}\\ 2\varepsilon_{xy}\\ 2\varepsilon_{yz}\\ 2\varepsilon_{xz}\end{Bmatrix}=\begin{Bmatrix}\frac{\partial u}{\partial x}\\ \frac{\partial v}{\partial y}\\ \frac{\partial w}{\partial z}\\ \frac{\partial u}{\partial y}+\frac{\partial v}{\partial x}\\ \frac{\partial v}{\partial z}+\frac{\partial w}{\partial y}\\ \frac{\partial u}{\partial z}+\frac{\partial w}{\partial x}\end{Bmatrix}=\begin{bmatrix}\frac{\partial}{\partial x} & 0 & 0\\ 0 & \frac{\partial}{\partial y} & 0\\ 0 & 0 & \frac{\partial}{\partial z}\\ \frac{\partial}{\partial y} & \frac{\partial}{\partial x} & 0\\ 0 & \frac{\partial}{\partial z} & \frac{\partial}{\partial y}\\ \frac{\partial}{\partial z} & 0 & \frac{\partial}{\partial x}\end{bmatrix}\begin{Bmatrix}u\\ v\\ w\end{Bmatrix}=[\partial]\{u\} \qquad \textbf{64}$$

But this can be simplified to a 2D strain field where the extra out of plane strains from plane stress state and out of plane stresses for plain strain state can be determined at the end of the solution once the in-plane values are known. As a result the strain is defined as:

$$\{\varepsilon\}=\begin{Bmatrix}\varepsilon_{xx}\\ \varepsilon_{yy}\\ 2\varepsilon_{xy}\end{Bmatrix}=\begin{Bmatrix}\frac{\partial u}{\partial x}\\ \frac{\partial v}{\partial y}\\ \frac{\partial u}{\partial y}+\frac{\partial v}{\partial x}\end{Bmatrix}=\begin{bmatrix}\frac{\partial}{\partial x} & 0\\ 0 & \frac{\partial}{\partial y}\\ \frac{\partial}{\partial y} & \frac{\partial}{\partial x}\end{bmatrix}\begin{Bmatrix}u\\ v\end{Bmatrix}=[\partial]\{u\}$$

[3x1] [3x1] [3x2] [2x1]

The global 3D stress field is given by:

$$\{\sigma\}=\begin{Bmatrix}\sigma_{xx}\\ \sigma_{yy}\\ \sigma_{zz}\\ 2\sigma_{xy}\\ 2\sigma_{yz}\\ 2\sigma_{xz}\end{Bmatrix}=\frac{E}{(1+\nu)(1-2\nu)}\begin{bmatrix}1-\nu & \nu & \nu & 0 & 0 & 0\\ \nu & 1-\nu & \nu & 0 & 0 & 0\\ \nu & \nu & 1-\nu & 0 & 0 & 0\\ 0 & 0 & 0 & \frac{1-2\nu}{2} & 0 & 0\\ 0 & 0 & 0 & 0 & \frac{1-2\nu}{2} & 0\\ 0 & 0 & 0 & 0 & 0 & \frac{1-2\nu}{2}\end{bmatrix}\begin{Bmatrix}\varepsilon_{xx}\\ \varepsilon_{yy}\\ \varepsilon_{zz}\\ 2\varepsilon_{xy}\\ 2\varepsilon_{yz}\\ 2\varepsilon_{xz}\end{Bmatrix}=[D]\{\varepsilon\}$$

However these can be simplified for the plane stress and plane strain state too:

For plane stress:

$$[D]=\frac{E}{1-\nu^2}\begin{bmatrix}1 & \nu & 0\\ \nu & 1 & 0\\ 0 & 0 & \frac{1-\nu}{2}\end{bmatrix} \qquad 65$$

$$\varepsilon_{zz}=-\frac{\nu}{1-\nu}\left(\varepsilon_{xx}+\varepsilon_{yy}\right)$$

$$\sigma_{zz}=0$$

For plane strain:

$$[D]=\frac{E}{(1+\nu)(1-2\nu)}\begin{bmatrix}1-\nu & \nu & 0\\ \nu & 1-\nu & 0\\ 0 & 0 & \frac{1-2\nu}{2}\end{bmatrix} \qquad 66$$

$$\varepsilon_{zz}=0$$

$$\sigma_{zz}=\nu\left(\sigma_{xx}+\sigma_{yy}\right)$$

The above [D] matrices are implemented via the following code for the plane strain case:

```
% Construct D matrix
% If plane strain
if plane==0
    e = elas/(1-pois^2);
    nu = pois/(1-pois);
else
    e = elas;
    nu = pois;
end

c = e/(1-nu^2);
E = zeros(3,3);
E(1,1) = c;
E(2,2) = c;
E(1,2) = c*nu;
E(2,1) = E(1,2);
E(3,3) = 0.5*c*(1-nu);
```

Now for an m-node element, the shape functions describe the displacement values in-between the actual nodes, thus the displacement is defined as:

$$\{u\}=\begin{Bmatrix}u(x,y)\\ v(x,y)\end{Bmatrix}=\begin{Bmatrix}\sum_{i=1}^{m}u_i N_i(x,y)\\ \sum_{i=1}^{m}v_i N_i(x,y)\end{Bmatrix}=[N(x,y)]\{d\} \qquad 67$$

There is no z co-ordinate as the problem is assumed to be 2D. In the case of a quad element 'm' is equivalent to 4. However if an isoperimetric element is assumed, then the shape functions remain the same in both domains. Thus:

$$\{u\} = \begin{Bmatrix} u(\xi,\eta) \\ v(\xi,\eta) \end{Bmatrix} = \begin{Bmatrix} \sum_{i=1}^{m} u_i N_i(\xi,\eta) \\ \sum_{i=1}^{m} v_i N_i(\xi,\eta) \end{Bmatrix} = [N(\xi,\eta)]\{d\} \qquad \textbf{68}$$

[2x1] [2x1] [2x2m] [2mx1]

Where the nodal displacement vector is:

$$\{d\} = \begin{Bmatrix} u_1 \\ v_1 \\ . \\ . \\ u_m \\ v_m \end{Bmatrix}$$

The geometric locations are thus given by:

$$\{x\} = \begin{Bmatrix} x(\xi,\eta) \\ y(\xi,\eta) \end{Bmatrix} = \begin{Bmatrix} \sum_{i=1}^{m} x_i N_i(\xi,\eta) \\ \sum_{i=1}^{m} y_i N_i(\xi,\eta) \end{Bmatrix} = [N(\xi,\eta)]\{c\} \qquad \textbf{69}$$

Where the nodal displacement vector is:

$$\{c\} = \begin{Bmatrix} x_1 \\ y_1 \\ . \\ . \\ x_m \\ y_m \end{Bmatrix}$$

But from equation 64 the following holds:

$$\{\varepsilon\} = \begin{Bmatrix} \varepsilon_{xx} \\ \varepsilon_{yy} \\ \varepsilon_{zz} \\ 2\varepsilon_{xy} \\ 2\varepsilon_{yz} \\ 2\varepsilon_{xz} \end{Bmatrix} = \begin{bmatrix} \frac{\partial}{\partial x} & 0 & 0 \\ 0 & \frac{\partial}{\partial y} & 0 \\ 0 & 0 & \frac{\partial}{\partial z} \\ \frac{\partial}{\partial y} & \frac{\partial}{\partial x} & 0 \\ 0 & \frac{\partial}{\partial z} & \frac{\partial}{\partial y} \\ \frac{\partial}{\partial z} & 0 & \frac{\partial}{\partial x} \end{bmatrix} \begin{Bmatrix} u \\ v \\ w \end{Bmatrix} = [\partial]\{u\} \qquad \textbf{70}$$

Thus if equation 67 is substituted with the equation above then the following is obtained:

$$\{\varepsilon\} = \begin{Bmatrix} \varepsilon_{xx} \\ \varepsilon_{yy} \\ \varepsilon_{zz} \\ 2\varepsilon_{xy} \\ 2\varepsilon_{yz} \\ 2\varepsilon_{xz} \end{Bmatrix} = \begin{bmatrix} \frac{\partial}{\partial x} & 0 & 0 \\ 0 & \frac{\partial}{\partial y} & 0 \\ 0 & 0 & \frac{\partial}{\partial z} \\ \frac{\partial}{\partial y} & \frac{\partial}{\partial x} & 0 \\ 0 & \frac{\partial}{\partial z} & \frac{\partial}{\partial y} \\ \frac{\partial}{\partial z} & 0 & \frac{\partial}{\partial x} \end{bmatrix} \begin{Bmatrix} \sum_{i=1}^{m} u_i N_i(x,y,z) \\ \sum_{i=1}^{m} v_i N_i(x,y,z) \\ \sum_{i=1}^{m} w_i N_i(x,y,z) \end{Bmatrix} = [\partial][N(x,y,z)]\{d\} \qquad \textbf{71}$$

[6x1] [6x3] [3x1]

Thus for the 2d case:

$$\{\varepsilon\} = \begin{Bmatrix} \varepsilon_{xx} \\ \varepsilon_{yy} \\ 2\varepsilon_{xy} \end{Bmatrix} = \begin{bmatrix} \frac{\partial}{\partial x} & 0 \\ 0 & \frac{\partial}{\partial y} \\ \frac{\partial}{\partial y} & \frac{\partial}{\partial x} \end{bmatrix} \begin{Bmatrix} \sum_{i=1}^{m} u_i N_i(x,y) \\ \sum_{i=1}^{m} v_i N_i(x,y) \end{Bmatrix} = [\partial][N(x,y)]\{d\} \qquad \textbf{72}$$

[3x1] [3x2] [2x1]

This is equivalent to:

$$\{\varepsilon\}=\begin{Bmatrix}\varepsilon_{xx}\\ \varepsilon_{yy}\\ 2\varepsilon_{xy}\end{Bmatrix}=\begin{bmatrix}1&0&0&0\\0&0&0&1\\0&1&1&0\end{bmatrix}\begin{Bmatrix}\frac{\partial u}{\partial x}\\ \frac{\partial u}{\partial y}\\ \frac{\partial v}{\partial x}\\ \frac{\partial v}{\partial y}\end{Bmatrix}=\begin{bmatrix}1&0&0&0\\0&0&0&1\\0&1&1&0\end{bmatrix}\begin{bmatrix}\frac{\partial}{\partial x}&0\\ 0&\frac{\partial}{\partial y}\\ \frac{\partial}{\partial x}&0\\ 0&\frac{\partial}{\partial y}\end{bmatrix}\begin{Bmatrix}\sum_{i=1}^{m}u_i N_i(x,y)\\ \sum_{i=1}^{m}v_i N_i(x,y)\end{Bmatrix}=[\partial][N(x,y)]\{d\}$$

[3x4] [4x1] = [3x4] [4x2] [2x1]

In order to get the above 2D formulation into the local co-ordinates η and ζ, it is required that the chain rule be applied and consequently the Jacobian will be required. The general equation for the Jacobian in terms of any parameter ϕ which could be u or v is given by:

$$\begin{Bmatrix}\frac{\partial\varphi}{\partial\xi}\\ \frac{\partial\varphi}{\partial\eta}\end{Bmatrix}=[J]\begin{Bmatrix}\frac{\partial\varphi}{\partial x}\\ \frac{\partial\varphi}{\partial y}\end{Bmatrix}=\begin{bmatrix}\frac{\partial x}{\partial\xi}&\frac{\partial y}{\partial\xi}\\ \frac{\partial x}{\partial\eta}&\frac{\partial y}{\partial\eta}\end{bmatrix}\begin{Bmatrix}\frac{\partial\varphi}{\partial x}\\ \frac{\partial\varphi}{\partial y}\end{Bmatrix} \qquad \textbf{73}$$

[2x1] [2x2] [2x1]

But from equation 69 the Jacobian [J] is defined as:

$$[J]=\begin{bmatrix}\frac{\partial x}{\partial\xi}&\frac{\partial y}{\partial\xi}\\ \frac{\partial x}{\partial\eta}&\frac{\partial y}{\partial\eta}\end{bmatrix}=\begin{bmatrix}\sum_{i=1}^{m}x_i\frac{\partial}{\partial\xi}N_i(\xi,\eta)&\sum_{i=1}^{m}y_i\frac{\partial}{\partial\xi}N_i(\xi,\eta)\\ \sum_{i=1}^{m}x_i\frac{\partial}{\partial\eta}N_i(\xi,\eta)&\sum_{i=1}^{m}y_i\frac{\partial}{\partial\eta}N_i(\xi,\eta)\end{bmatrix} \qquad \textbf{74}$$

This can be visualised as:

$$[J]=\begin{bmatrix}\frac{\partial x}{\partial\xi}&\frac{\partial y}{\partial\xi}\\ \frac{\partial x}{\partial\eta}&\frac{\partial y}{\partial\eta}\end{bmatrix}=\begin{bmatrix}\frac{\partial}{\partial\xi}N_1(\xi,\eta)&\frac{\partial}{\partial\xi}N_2(\xi,\eta)&\cdot&\frac{\partial}{\partial\xi}N_m(\xi,\eta)\\ \frac{\partial}{\partial\eta}N_1(\xi,\eta)&\frac{\partial}{\partial\eta}N_2(\xi,\eta)&\cdot&\frac{\partial}{\partial\eta}N_m(\xi,\eta)\end{bmatrix}\begin{bmatrix}x_1&y_1\\ x_2&y_2\\ \cdot&\cdot\\ x_m&y_m\end{bmatrix} \qquad \textbf{75}$$

[2x2] [2xm] [mx2]

The inverse of the Jacobian is given by:

$$[J]^{-1}=\frac{1}{\det([J])}\begin{bmatrix}\frac{\partial y}{\partial\eta}&-\frac{\partial y}{\partial\xi}\\ -\frac{\partial x}{\partial\eta}&\frac{\partial x}{\partial\xi}\end{bmatrix} \qquad \textbf{76}$$

This inverse will be sufficient for:

$$\begin{Bmatrix} \frac{\partial u}{\partial x} \\ \frac{\partial u}{\partial y} \end{Bmatrix} = [J]^{-1} \begin{Bmatrix} \frac{\partial u}{\partial \xi} \\ \frac{\partial u}{\partial \eta} \end{Bmatrix}$$

However in this case both u and v are required thus the following matrix [J]*is defined:

$$\{du\} = \begin{Bmatrix} \frac{\partial u}{\partial x} \\ \frac{\partial u}{\partial y} \\ \frac{\partial v}{\partial x} \\ \frac{\partial v}{\partial y} \end{Bmatrix} = \frac{1}{\det([J])} \begin{bmatrix} \frac{\partial y}{\partial \eta} & -\frac{\partial y}{\partial \xi} & 0 & \\ -\frac{\partial x}{\partial \eta} & \frac{\partial x}{\partial \xi} & & \\ & & \frac{\partial y}{\partial \eta} & -\frac{\partial y}{\partial \xi} \\ 0 & & -\frac{\partial x}{\partial \eta} & \frac{\partial x}{\partial \xi} \end{bmatrix} \begin{Bmatrix} \frac{\partial u}{\partial \xi} \\ \frac{\partial u}{\partial \eta} \\ \frac{\partial v}{\partial \xi} \\ \frac{\partial v}{\partial \eta} \end{Bmatrix} = [J]^* [\partial N] \{d\}$$

[4x1] [4x4] [4x1]= [4x4] [4x2m] [2mx1]

Where

$$[J]^* = \frac{1}{\det([J])} \begin{bmatrix} \frac{\partial y}{\partial \eta} & -\frac{\partial y}{\partial \xi} & 0 & \\ -\frac{\partial x}{\partial \eta} & \frac{\partial x}{\partial \xi} & & \\ & & \frac{\partial y}{\partial \eta} & -\frac{\partial y}{\partial \xi} \\ 0 & & -\frac{\partial x}{\partial \eta} & \frac{\partial x}{\partial \xi} \end{bmatrix}$$

77

$$[\partial N] = \begin{bmatrix} \frac{\partial}{\partial \xi} N_1(\xi,\eta) & 0 & \frac{\partial}{\partial \xi} N_2(\xi,\eta) & 0 & . & . & \frac{\partial}{\partial \xi} N_m(\xi,\eta) & 0 \\ \frac{\partial}{\partial \eta} N_1(\xi,\eta) & 0 & \frac{\partial}{\partial \eta} N_2(\xi,\eta) & 0 & . & . & \frac{\partial}{\partial \eta} N_m(\xi,\eta) & 0 \\ 0 & \frac{\partial}{\partial \xi} N_1(\xi,\eta) & 0 & \frac{\partial}{\partial \xi} N_2(\xi,\eta) & . & . & 0 & \frac{\partial}{\partial \xi} N_m(\xi,\eta) \\ 0 & \frac{\partial}{\partial \eta} N_1(\xi,\eta) & 0 & \frac{\partial}{\partial \eta} N_2(\xi,\eta) & . & . & 0 & \frac{\partial}{\partial \eta} N_m(\xi,\eta) \end{bmatrix}$$

$$\{d\} = \begin{Bmatrix} u_1 \\ v_1 \\ . \\ . \\ u_m \\ v_m \end{Bmatrix}$$

But the strain for an element is given by (this is the 2D equation of equation 70):

$$\{\varepsilon\}=\begin{Bmatrix}\varepsilon_{xx}\\ \varepsilon_{yy}\\ 2\varepsilon_{xy}\end{Bmatrix}=\begin{bmatrix}1&0&0&0\\0&0&0&1\\0&1&1&0\end{bmatrix}\begin{Bmatrix}\frac{\partial u}{\partial x}\\ \frac{\partial u}{\partial y}\\ \frac{\partial v}{\partial x}\\ \frac{\partial v}{\partial y}\end{Bmatrix} \qquad \textbf{78}$$

[3x1] [3x4] [4x1]

Thus if equations 78 and 77 are combined:

$$\{\varepsilon\}=\begin{Bmatrix}\varepsilon_{xx}\\ \varepsilon_{yy}\\ 2\varepsilon_{xy}\end{Bmatrix}=\begin{bmatrix}1&0&0&0\\0&0&0&1\\0&1&1&0\end{bmatrix}\begin{Bmatrix}\frac{\partial u}{\partial x}\\ \frac{\partial u}{\partial y}\\ \frac{\partial v}{\partial x}\\ \frac{\partial v}{\partial y}\end{Bmatrix}=\begin{bmatrix}1&0&0&0\\0&0&0&1\\0&1&1&0\end{bmatrix}[J]^{*}\begin{Bmatrix}\frac{\partial u}{\partial \xi}\\ \frac{\partial u}{\partial \eta}\\ \frac{\partial v}{\partial \xi}\\ \frac{\partial v}{\partial \eta}\end{Bmatrix} \qquad \textbf{79}$$

[3x1] [3x4] [4x1] [3x4] [4x4] [4x1]

The differential of the displacement with respect to the Isoparametric parameters are as per equation 77 given by:

$$\{du\}=\begin{Bmatrix}\frac{\partial u}{\partial \xi}\\ \frac{\partial u}{\partial \eta}\\ \frac{\partial v}{\partial \xi}\\ \frac{\partial v}{\partial \eta}\end{Bmatrix}=\begin{Bmatrix}\sum_{i=1}^{m}u_i N_i(x,y)\\ \sum_{i=1}^{m}v_i N_i(x,y)\end{Bmatrix}=\begin{bmatrix}N_{1,\xi}&0&.&N_{m,\xi}&0\\N_{1,\eta}&0&.&N_{m,\eta}&0\\0&N_{1,\xi}&.&0&N_{m,\xi}\\0&N_{1,\eta}&.&0&N_{m,\eta}\end{bmatrix}\begin{Bmatrix}u_1\\ v_1\\ .\\ u_m\\ v_m\end{Bmatrix}=[\partial N]\{d\} \qquad \textbf{80}$$

[4x1] [4x2m] [2mx1]

Thus to obtain the ***Plane stress*** strains equations 65 is used, hence.

$$\begin{Bmatrix}\sigma_{xx}\\ \sigma_{yy}\\ \varsigma_{xy}\end{Bmatrix}=[D]\{\varepsilon\}=\frac{E}{1-\nu^2}\begin{bmatrix}1 & \nu & 0\\ \nu & 1 & 0\\ 0 & 0 & \frac{1-\nu}{2}\end{bmatrix}\begin{bmatrix}1 & 0 & 0 & 0\\ 0 & 0 & 0 & 1\\ 0 & 1 & 1 & 0\end{bmatrix}[J]^*\begin{Bmatrix}\frac{\partial u}{\partial \xi}\\ \frac{\partial u}{\partial \eta}\\ \frac{\partial v}{\partial \xi}\\ \frac{\partial v}{\partial \eta}\end{Bmatrix}=[D][I][J]^*\{du\} \quad \textbf{81}$$

[3x1] [3x3] [3x4] [4x4] [4x1]

Then the other components of strain and stress are found using:

$$\varepsilon_{zz}=-\frac{\nu}{1-\nu}\left(\varepsilon_{xx}+\varepsilon_{yy}\right)$$
$$\sigma_{zz}=0$$

Thus to obtain the ***Plane strains*** equation 66 is applied.

$$\begin{Bmatrix}\sigma_{xx}\\ \sigma_{yy}\\ \varsigma_{xy}\end{Bmatrix}=[D]\{\varepsilon\}=\frac{E}{(1+\nu)(1-2\nu)}\begin{bmatrix}1-\nu & \nu & 0\\ \nu & 1-\nu & 0\\ 0 & 0 & \frac{1-2\nu}{2}\end{bmatrix}\begin{bmatrix}1 & 0 & 0 & 0\\ 0 & 0 & 0 & 1\\ 0 & 1 & 1 & 0\end{bmatrix}[J]^*\begin{Bmatrix}\frac{\partial u}{\partial \xi}\\ \frac{\partial u}{\partial \eta}\\ \frac{\partial v}{\partial \xi}\\ \frac{\partial v}{\partial \eta}\end{Bmatrix}=[D][I][J]^*\{du\} \quad \textbf{82}$$

[3x1] [3x3] [3x4] [4x4] [4x1]

Then the other components of strain and stress are found using:

$$\varepsilon_{zz}=0$$
$$\sigma_{zz}=\nu\left(\sigma_{xx}+\sigma_{yy}\right)$$

Consequently the above equations 81 and 82 in matrix form are:

$$\begin{Bmatrix}\sigma_{xx}\\ \sigma_{yy}\\ \varsigma_{xy}\end{Bmatrix}=[D]\{\varepsilon\}=[D][I][J]^*\{du\} \quad \textbf{83}$$

Thus from equation 63:

$$U^{elm}=\frac{1}{2}\int_v\langle\varepsilon\rangle[D]\{\varepsilon\}dV \quad \textbf{84}$$

Substituting in equation 83

$$U^{elm} = \frac{1}{2}\int_{v_{local}^{elm}} [[I][J]*\{du\}]^T [D][I][J^*]\{du\}dV \quad \textbf{85}$$

Simplifying

$$U^{elm} = \frac{1}{2}\int_{v_{local}^{elm}} \langle du\rangle [J^*]^T [I]^T [D][I][J^*]\{du\}dV \quad \textbf{86}$$

But if a constant thickness is assumed, then the volume can be replace since dV=tdA, hence:

$$U^{elm} = \frac{1}{2}\int_{v_{local}^{elm}}\int_{v_{local}^{elm}} \langle du\rangle [J^*]^T [I]^T [D][I][J^*]\{du\}tdA \quad \textbf{87}$$

But {du} can be expressed as equation 77 together with 80. If 't' is constant in the area of the element and since the 'd' vector doesn't varying in the area thus the formula can be simplified too:

$$U^{elm} = \frac{1}{2}\langle d\rangle \int_{v_{local}^{elm}} [\partial N(\xi,\eta)]^T [J^*]^T [I]^T [D][I][J^*][\partial N(\xi,\eta)]dA\{d\}t \quad \textbf{88}$$

But $dA = d\xi d\eta$ hence:

$$U^{elm} = \frac{1}{2}\langle d\rangle \int_{\xi}\int_{\eta} [\partial N(\xi,\eta)]^T [J^*]^T [I]^T [D][I][J^*][\partial N(\xi,\eta)]d\xi d\eta\{d\}t \quad \textbf{89}$$

Thus the stiffness matrix is given by:

$$K_{elm} = [\partial N(\xi,\eta)]^T [J^*]^T [I]^T [D][I][J^*][\partial N(\xi,\eta)]t \quad \textbf{90}$$

$$K_{elm} = [B]^T [D][B]t \quad \textbf{91}$$

Thus [B] is defined as:

$$[B] = [I][J^*][\partial N(\xi,\eta)] \quad \textbf{92}$$

The [B] is generated in the code using:

```
%[3x4]
I_trans=[1 0 0 0;0 0 0 1;0 1 1 0];
%[4x4]
J_star = [invJ(1,1) invJ(1,2) 0 0;invJ(2,1) invJ(2,2) 0 0;
    0 0 invJ(1,1) invJ(1,2);0 0 invJ(2,1) invJ(2,2)];
%[3x8]=[3x4]*[4x4]*4x8]
B=I_trans*J_star*D;
```

However in the code the integral is calculated using 4 Gauss points. Thus the sum of each K_{elm} matrices is required at each Gauss point. This is implemented in the code by:

```
%Now to calculate the element stiffness using four nodes we sum
%the calculation at each point
k_elem = k_elem + t*B'*E*B*detJ;
```

This is accomplished in the code using the loop jGauss and inner loop iGauss. Thus solutions are found at the following Gauss points:

1. $\left(\frac{1}{\sqrt{3}},\frac{1}{\sqrt{3}}\right)$
2. $\left(\frac{1}{\sqrt{3}},-\frac{1}{\sqrt{3}}\right)$
3. $\left(-\frac{1}{\sqrt{3}},-\frac{1}{\sqrt{3}}\right)$
4. $\left(-\frac{1}{\sqrt{3}},\frac{1}{\sqrt{3}}\right)$

C.1.4 Implementation of Q4SS

To implement the Selective substitution Integration Quad 4 element (Q4SS) all that is required is a modification of the Gauss integration. Thus the Full integration loop is modified from:

```
Gausspt = 0;
    for jGauss = 1:2
        for iGauss = 1:2
            %set location of Gauss points.
            %Loop for each quadrant
            xi  = (-1)^iGauss*Gauss_pos;
            eta = (-1)^jGauss*Gauss_pos;
            %Thus is dN as per equation 25 [4x8]
            D = 0.25*[ -1.0+eta 0  1.0-eta 0 1.0+eta 0 -1.0-eta 0
                      -1.0+xi  0 -1.0-xi  0 1.0+xi  0  1.0-xi  0
                       0 -1.0+eta 0  1.0-eta 0 1.0+eta 0 -1.0-eta
                       0 -1.0+xi  0 -1.0-xi  0 1.0+xi  0  1.0-xi ];
            %This is as per equation 23 [J]=[2x2]
            J = D(1:2,1:2:7)*[x1 y1
                              x2 y2
                              x3 y3
                              x4 y4];
            %standard determinant equation of [2x2] matrix
            detJ = J(1,1)*J(2,2) - J(2,1)*J(1,2);
            %equation 24 [2x2]
            invJ = 1.0/detJ * [J(2,2)  -J(1,2)
                              -J(2,1)   J(1,1)];
            %equation 25
            %[3x4]
            I_trans=[1 0 0 0;0 0 0 1;0 1 1 0];
            %[4x4]
            J_star = [invJ(1,1) invJ(1,2) 0 0;invJ(2,1) invJ(2,2) 0 0;
                0 0 invJ(1,1) invJ(1,2);0 0 invJ(2,1) invJ(2,2)];
            %[3x8]=[3x4]*[4x4]*4x8]
            B=I_trans*J_star*D;

            %Now to calculate the element stiffness using four nodes we sum
            %the calculation at each point
            k_elem = k_elem + t*B'*E*B*detJ;

            %create a global B and E*B for all elements (needed to
            %calculate strain at the end
            B_ALL((i-1)*12+Gausspt*3+1:(i-1)*12+Gausspt*3+3,:)=B;
            EB_ALL((i-1)*12+Gausspt*3+1:(i-1)*12+Gausspt*3+3,:)=E*B;
            Gausspt = Gausspt + 1;
        end
    end
```

The new Selective substitution integration code is indicated below. The main changes are:

1. Accomplish four point Gauss integration as per the Full Integration

2. Complete integration at the Gauss point (0.0,0.0) and replace the shear terms with these values.

    a. This is done via adding 4 times the solution of the 1 point Gauss Quadrature to the K_elm

```
%Replace correction shear components
%Now place the shear strain at (0.0,0.0) in correct plact
%of element stiffness matrix [e]=[3x3] therefore [k]=[8x8]
%4 is for the Gauss quadration for one point
k_elem = k_elem + 4*t*B'*E_with_shear*B*detJ;
```

    b. The building of the B_All and EB_All arrays remains the same with the addition of the shear to each of the calculated sections.

```
%create a global B and E*B for all elements (needed to
%calculate strain at the end (Have to add for each calculated
%point)
Gausspt = 0;
for jGauss = 1:2
    for iGauss = 1:2
        B_ALL((i-1)*12+Gausspt*3+1:(i-1)*12+Gausspt*3+3,:)=B_ALL((i-1)*12+Gausspt*3+1:(i-1)*12+Gausspt*3+3,:)+B;
        EB_ALL((i-1)*12+Gausspt*3+1:(i-1)*12+Gausspt*3+3,:)=EB_ALL((i-1)*12+Gausspt*3+1:(i-1)*12+Gausspt*3+3,:)+E_with_shear*B;
        Gausspt = Gausspt + 1;
    end
end
```

C.1.5 Implementation of Q4RI

To implement the Reduced Integration Quad 4 element (Q4RI) all that changes is the Gauss integration. Thus the Q4 Full integration loop is modified too only integrate at one Gauss point.

```
%positon of 1 point guass quadrature at centre
Gauss_pos = 1.0/sqrt(3.0);
%-------------------------------------------------------------------------
%-------------------Reduced Integration 1 Point Guass Quadrature-----------

%set location of Gauss points.
%Loop for each quadrant
xi  = (-1)^iGauss*Gauss_pos;
eta = (-1)^jGauss*Gauss_pos;
%Thus is dN as per equation 25 [4x8]
D = 0.25*[ -1.0+eta 0  1.0-eta 0 1.0+eta 0 -1.0-eta 0
    -1.0+xi  0 -1.0-xi  0 1.0+xi  0  1.0-xi  0
    0 -1.0+eta 0  1.0-eta 0 1.0+eta 0 -1.0-eta
    0 -1.0+xi  0 -1.0-xi  0 1.0+xi  0  1.0-xi ];
%This is as per equation 23 [J]=[2x2]
J = D(1:2,1:2:7)*[x1 y1
    x2 y2
    x3 y3
    x4 y4];
%standard determinant equation of [2x2] matrix
detJ = J(1,1)*J(2,2) - J(2,1)*J(1,2);
%equation 24 [2x2]
invJ = 1.0/detJ * [J(2,2)  -J(1,2)
    -J(2,1)   J(1,1)];
%Basically equation 25
%Modified by TJ McMillan to confirm to derived equations
%             %[3x4]
%             B2_old = [invJ(1,1) invJ(1,2) 0 0
%                   0 0 invJ(2,1) invJ(2,2)
%                   invJ(2,1) invJ(2,2) invJ(1,1) invJ(1,2)];
%             %[3x8]=[3x4]*[4x8]
%             B_old = B2_old*D;
%[3x4]
I_trans=[1 0 0 0;0 0 0 1;0 1 1 0];
%[4x4]
J_star = [invJ(1,1) invJ(1,2) 0 0;invJ(2,1) invJ(2,2) 0 0;
    0 0 invJ(1,1) invJ(1,2);0 0 invJ(2,1) invJ(2,2)];
%[3x8]=[3x4]*[4x4]*4x8]
B=I_trans*J_star*D;

%Now to calculate the element stiffness using one Guass point
%thus 4 times the area
k_elem =4*t*B'*E*B*detJ;

%create a global B and E*B for all elements (needed to
%calculate strain at the end
B_ALL((i-1)*12+Gausspt*3+1:(i-1)*12+Gausspt*3+3,:)=B;
EB_ALL((i-1)*12+Gausspt*3+1:(i-1)*12+Gausspt*3+3,:)=E*B;
```

The new reduced integration code is indicated above. The main changes are:

1. Integration is no longer accomplished over the 4 Gaussian points.

2. Only one point Gauss integration (2x) as per weighting factor pg. 210 cook

C.1.6 Implementation of 5 Beta Hybrid element

The five beta element is the quad4 element used for analysis in Nastran. The creation of the 5 β element is based on the following mathematical formulae:

$$\{\sigma\} = [P][\beta]$$

Where for the Five-Beta elements a matrix [P] is defined:

$$[P] = \begin{bmatrix} 1 & y & 0 & 0 & 0 \\ 0 & 0 & 1 & x & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

X and y are defined in the eta and xi planes by the sum of the shape functions:

$$\begin{Bmatrix} x \\ y \end{Bmatrix} = \begin{Bmatrix} x(\xi,\eta) \\ y(\xi,\eta) \end{Bmatrix} = \begin{Bmatrix} \sum_{i=1}^{4} x_i N_i(\xi,\eta) \\ \sum_{i=1}^{4} y_i N_i(\xi,\eta) \end{Bmatrix}$$

Thus the [P] matrix becomes:

$$[P] = \begin{bmatrix} 1 & \sum_{i=1}^{4} y_i N_i(\xi,\eta) & 0 & 0 & 0 \\ 0 & 0 & 1 & \sum_{i=1}^{4} x_i N_i(\xi,\eta) & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

And

$$[\beta] = \begin{bmatrix} \beta_1 \\ \beta_2 \\ \beta_3 \\ \beta_4 \\ \beta_5 \end{bmatrix}$$

For the plane strain and stress states the following is applied for the [E] matrix, see section C.1.3 for derivation.

For plane stress:

$$[D]=\frac{E}{1-\nu^2}\begin{bmatrix}1 & \nu & 0\\ \nu & 1 & 0\\ 0 & 0 & \frac{1-\nu}{2}\end{bmatrix} \qquad \textbf{93}$$

$$\varepsilon_{zz}=-\frac{\nu}{1-\nu}\left(\varepsilon_{xx}+\varepsilon_{yy}\right)$$

$$\sigma_{zz}=0$$

For plane strain:

$$[D]=\frac{E}{(1+\nu)(1-2\nu)}\begin{bmatrix}1-\nu & \nu & 0\\ \nu & 1-\nu & 0\\ 0 & 0 & \frac{1-2\nu}{2}\end{bmatrix} \qquad \textbf{94}$$

$$\varepsilon_{zz}=0$$

$$\sigma_{zz}=\nu\left(\sigma_{xx}+\sigma_{yy}\right)$$

C.1.7 Stress Derivation

To obtain the stress from the displacements as new matrix [H] is defined:

$$[H]=\int[P]^T[D]^{-1}[P]dV=\frac{E}{1-\nu^2}\int[P]^T\begin{bmatrix}1 & \nu & 0\\ \nu & 1 & 0\\ 0 & 0 & \frac{1-\nu}{2}\end{bmatrix}^{-1}[P]dV$$

If a constant thickness, t, is assumed then the volume integral can be modified to:

$$[H]=\frac{E}{1-\nu^2}\int[P]^T\begin{bmatrix}1 & \nu & 0\\ \nu & 1 & 0\\ 0 & 0 & \frac{1-\nu}{2}\end{bmatrix}^{-1}[P]dV=\frac{E}{1-\nu^2}\int_{-1}^{1}\int_{-1}^{1}[P]^T\begin{bmatrix}1 & \nu & 0\\ \nu & 1 & 0\\ 0 & 0 & \frac{1-\nu}{2}\end{bmatrix}^{-1}[P]tdxdy \qquad \textbf{95}$$

However working with Isoparametric Quad elements it is required to convert this from the xy domain to the ξη domain, thus the determinant of the Jacobian is required. The Jacobian is given by (see section C.1.3 for derivation):

$$[J]=\begin{bmatrix}\frac{\partial x}{\partial \xi} & \frac{\partial y}{\partial \xi}\\ \frac{\partial x}{\partial \eta} & \frac{\partial y}{\partial \eta}\end{bmatrix}=\begin{bmatrix}\frac{\partial}{\partial \xi}N_1(\xi,\eta) & \frac{\partial}{\partial \xi}N_2(\xi,\eta) & . & \frac{\partial}{\partial \xi}N_m(\xi,\eta)\\ \frac{\partial}{\partial \eta}N_1(\xi,\eta) & \frac{\partial}{\partial \eta}N_2(\xi,\eta) & . & \frac{\partial}{\partial \eta}N_m(\xi,\eta)\end{bmatrix}\begin{bmatrix}x_1 & y_1\\ x_2 & y_2\\ . & .\\ x_m & y_m\end{bmatrix}$$

Thus the inverse is:

$$[J]^{-1}=\frac{1}{\det([J])}\begin{bmatrix}\frac{\partial y}{\partial \eta} & -\frac{\partial y}{\partial \xi}\\ -\frac{\partial x}{\partial \eta} & \frac{\partial x}{\partial \xi}\end{bmatrix}$$

The det([J]) is given by:

$$\det[J]=\frac{\partial x}{\partial \xi}\frac{\partial y}{\partial \eta}-\frac{\partial x}{\partial \eta}\frac{\partial y}{\partial \xi}$$

Consequently by converting equation 95 the following Isoparametric representation is obtain:

$$[H]=\frac{E}{1-\nu^2}\int[P]^T\begin{bmatrix}1 & \nu & 0\\ \nu & 1 & 0\\ 0 & 0 & \frac{1-\nu}{2}\end{bmatrix}^{-1}[P]dV=\frac{E}{1-\nu^2}\int_{-1}^{1}\int_{-1}^{1}[P]^T\begin{bmatrix}1 & \nu & 0\\ \nu & 1 & 0\\ 0 & 0 & \frac{1-\nu}{2}\end{bmatrix}^{-1}[P]\det([J])td\xi d\eta \quad \textbf{96}$$

To obtain [G] the following must be found:

$$[G]=\int_s [R]^T[L]dS$$

For constant thickness along the side of an element, the above equation can be re-written in terms of the side length, s:

$$[G]=\int_s [R]^T[L]tds$$

Where for a side of an element between nodes i and j:

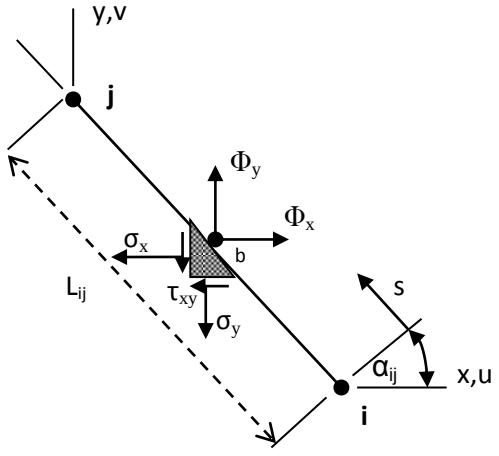


Figure C-4: four node element with boundary tractions

Consequently since,

$$\{\varphi\} = [R]\{\beta\}$$

And the tractions in 2D are given by:

$$\varphi_{x_{ij}} = l_{ij}\sigma_x + m\tau_{xy}$$
$$\varphi_{y_{ij}} = l_{ij}\tau_{xy} + m\sigma_y$$

Where

$l_{ij} = cos\alpha_{ij} = \frac{y_j - y_i}{L_{ij}}$

$m_{ij} = sin\alpha_{ij} = \frac{x_i - x_j}{L_{ij}}$

Thus the [R] matrix can be defined in terms of the Beta elements since the stresses are given by:

$$\{\sigma\} = \begin{Bmatrix}\sigma_x \\ \sigma_y \\ \tau_{xy}\end{Bmatrix} = [P]\{\beta\} = \begin{bmatrix}1 & y & 0 & 0 & 0 \\ 0 & 0 & 1 & x & 0 \\ 0 & 0 & 0 & 0 & 1\end{bmatrix}\begin{Bmatrix}\beta_1 \\ \beta_2 \\ \beta_3 \\ \beta_4 \\ \beta_5\end{Bmatrix}$$

Thus substituting and solving, the [R] matrix is evaluated to be:

$$[R] = \begin{bmatrix}l_{12} & l_{12}y_{12} & 0 & 0 & m_{12} \\ 0 & 0 & m_{12} & m_{12}x_{12} & l_{12} \\ l_{23} & l_{23}y_{23} & 0 & 0 & m_{23} \\ 0 & 0 & m_{23} & m_{23}x_{23} & l_{23} \\ \vdots & \vdots & \vdots & \vdots & \vdots\end{bmatrix}$$

Where

$x_{ij} = x_i - s.sin\alpha_{ij}$

$y_{ij} = y_i - s.cos\alpha_{ij}$

The [L] (displacement anywhere on specific side) matrix is defined by:

$$\{u_b\} = [L]\{d\}$$
$$\begin{Bmatrix}u_{12} \\ v_{12} \\ u_{23} \\ \vdots\end{Bmatrix} = [L]\begin{Bmatrix}u_1 \\ v_1 \\ u_2 \\ \vdots\end{Bmatrix}$$

The displacement at any point/node on a side is defined by the combination of the two nodes making up the side via the use of the shape functions. Consider the side 1 to 2 with vertical displacements in v only:

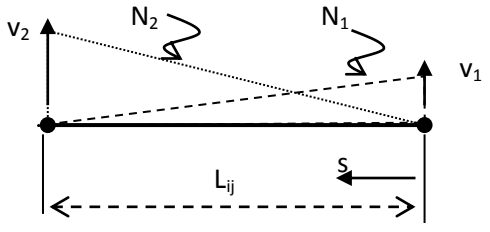


Figure C-5: Boundary displacement at a location on side 12

The linear shape functions are given by:

$$N_1 = \frac{L_{12} - s}{L_{12}}$$
$$N_2 = \frac{s}{L_{12}}$$

Thus the contribution from node 1 is $v_1 N_1$ and the contribution from node 2 is $v_2 N_{12}$. Consequently the displacement in the y direction alongside 12 at any location s from node 1 is:

$$u_{12} = v_1 N_1 + v_2 N_2$$
$$u_{12} = v_1 \frac{L_{12} - s}{L_{12}} + v_2 \frac{s}{L_{12}}$$

Thus the [L] matrix can be found to be:

$$[L] = \begin{bmatrix} 1 - \frac{s}{L_{12}} & 0 & \frac{s}{L_{12}} & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 - \frac{s}{L_{12}} & 0 & \frac{s}{L_{12}} & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 - \frac{s}{L_{23}} & 0 & \frac{s}{L_{23}} & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \end{bmatrix}$$

The [G] matrix can be calculated via the integral above. Thus [G] is defined as {as derived in reference [28], page 169} for the 5 Beta hybrid elements to be:

$$[G] = \begin{bmatrix} Y1 & 0 & Y2 & 0 & Y3 & 0 & Y4 & 0 \\ Z1 & 0 & Z2 & 0 & Z3 & 0 & Z4 & 0 \\ 0 & X1 & 0 & X2 & 0 & X3 & 0 & X4 \\ 0 & W1 & 0 & W2 & 0 & W3 & 0 & W4 \\ X1 & Y1 & X2 & Y2 & X3 & Y3 & X4 & Y4 \end{bmatrix} \quad \textbf{97}$$

[5x8]

Where,

$$Yj = 3(y_k - y_i)$$
$$Xj = 3(x_i - x_k)$$
$$Zj = y_k(y_j + y_k) - y_i(y_i + y_j)$$
$$Wj = x_i(x_i + x_j) - x_k(x_j + x_k)$$

The indices are as per the following element:

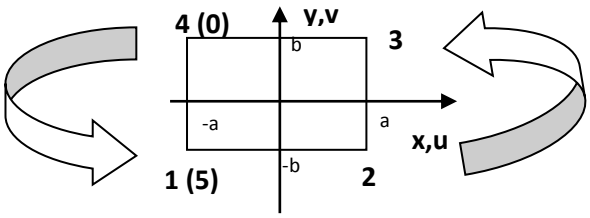


Figure C-6: Cyclic indices for [G] matrix

Thus if =j-1 and k=j+1, and if i, j, k = 0 then this implies the previous node, namely node 4 or if i, j, k=5 this implies the next node namely node 1. Consequently the stiffness matrix for an element is given by:

$$[K]_{elm} = [G]^T [H]^{-1} [G] \qquad \textbf{98}$$

Thus the nodal displacements are solved for using:

$$[K]\{d\} = \{R\}$$

Where

{R} - is as per the standard element.

Once the nodal displacements have been obtained then the stresses are found using:

$$\{\sigma\} = [P][\beta] = [P][H]^{-1}[G]\{d\}$$

The term $[P][H]^{-1}[G]$ can be thought of as the EB_ALL variable in the Matlab code. To calculate the strains there are two methods:

1. Calculate using: $[E]^{-1}[P]\{\beta\}$. This implemented using the following code:

```
Gauss_pos = 1.0/sqrt(3.0);
Gausspt = 0;
for jGauss = 1:2
    for iGauss = 1:2
        %set location of Gauss points.
        %Loop for each quadrant
        xi  = (-1)^iGauss*Gauss_pos;
        eta = (-1)^jGauss*Gauss_pos;

        %define the shape functions
        N1=1/4*(1-xi)*(1-eta);
        N2=1/4*(1+xi)*(1-eta);
        N3=1/4*(1+xi)*(1+eta);
        N4=1/4*(1-xi)*(1+eta);

        %define the nodal location in x-y plane (cartesian)
        X_nodal=N1*x1+N2*x2+N3*x3+N4*x4;
        Y_nodal=N1*y1+N2*y2+N3*y3+N4*y4;

        %Thus is [P] = [3x5]
        P = [ 1 Y_nodal 0 0 0;0 0 1 X_nodal 0; 0 0 0 0 1];

        %get the strains from the stress (at each guass point)
        %[3x1]=[3x3]*[3x5]*[5x1]
        strain(i,(3*Gausspt+1):(3*Gausspt+3))=(inv(E)*P*Beta(i,:)')';
        Gausspt=Gausspt+1;
    end
end
```

2. Calculate using the normal shape functions as for the Q4 standard element formulation. This is as before. B_ALL is as generated for the Q4 full integration system.

```
strain(i,:) = (B_ALL((i-1)*12+1:12*i,:)*U(assembly(i,:)))';
```

C.2 Material matrices for plane strain or plane stress

C.2.1 2D Elements

The material matrices for the 2D elements are defined for plane stress and plane strain as:

For plane stress:

$$[\mathbf{D}] = \frac{\mathbf{E}}{\mathbf{1}-\nu^2}\begin{bmatrix} \mathbf{1} & \nu & \mathbf{0} \\ \nu & \mathbf{1} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \frac{\mathbf{1}-\nu}{\mathbf{2}} \end{bmatrix} \qquad \mathbf{99}$$

$$\varepsilon_{\mathbf{zz}} = -\frac{\nu}{\mathbf{1}-\nu}\left(\varepsilon_{\mathbf{xx}} + \varepsilon_{\mathbf{yy}}\right)$$

$$\sigma_{\mathbf{zz}} = \mathbf{0}$$

For plane strain:

$$[\mathbf{D}] = \frac{\mathbf{E}}{(\mathbf{1}+\nu)(\mathbf{1}-\mathbf{2}\nu)}\begin{bmatrix} \mathbf{1}-\nu & \nu & \mathbf{0} \\ \nu & \mathbf{1}-\nu & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \frac{\mathbf{1}-\mathbf{2}\nu}{\mathbf{2}} \end{bmatrix} \qquad \mathbf{100}$$

$$\varepsilon_{\mathbf{zz}} = \mathbf{0}$$

$$\sigma_{\mathbf{zz}} = \nu\left(\sigma_{\mathbf{xx}} + \sigma_{\mathbf{yy}}\right)$$

C.2.2 2D Axisymmetric Elements

In the Axisymmetric case the only valid material matrix is the plain stress matrix as the stress in the hoop should be constant for a specific (r,z) location. Thus the matrix [D] is defined as [30]:

For 3D strain:

$$[D] = \frac{E}{(1+v)(1-2v)}\begin{bmatrix} 1-v & v & v & 0 \\ v & 1-v & v & 0 \\ v & v & 1-v & 0 \\ 0 & 0 & 0 & \frac{1-2v}{2} \end{bmatrix} \qquad \mathbf{101}$$

$$\varepsilon_\theta = \frac{2\pi(r+u)-2\pi r}{2\pi r} = \frac{\Delta circumference}{circumference\ orginal} = \frac{u}{r}$$

C.3 QUAD8

C.3.1 Changes to code

To modify the quad4 code to quad8 code it is first required is to modify the number of columns read in, from 5 to 9, in the element connectivity matrix. This is done using an element type check which determines the number of elm vertices to be read in. The new code is thus:

```
%read the type of element
if (strcmp(lower(elmtype),'tria3'))
    elmVert=3;
elseif (strcmp(lower(elmtype),'quad4'))
    elmVert=4;
elseif (strcmp(lower(elmtype),'quad8'))
    elmVert=4;
elseif (strcmp(lower(elmtype),'tria6'))
    elmVert=6;
else
    warning('elmtype not recognised in ReadInpFileType.m')
    return %exit function
end
```

Thus the element type is read in as follows:

```
% Read element number and element connectivity
dummy = fgetl(fid);%get "Element_connectivity (elm (nodes 1 - n) mat#)"
elnodes = fscanf(fid,'%f \n',[elmVert+3,nelem])';
```

The second item modified is the assembler code to construct the connection of the nodes as indicated in Cook, while maintaining the current node number system of the mesh generators. Thus the code is modified as follows:

```
%Construct assembly array
assembly = zeros(mesh.NumElm,8);
for i=1:mesh.NumElm
    p1 = find(mesh.Nodes==mesh.Elm(i).ElmNodes(1));
    p2 = find(mesh.Nodes==mesh.Elm(i).ElmNodes(2));
    p3 = find(mesh.Nodes==mesh.Elm(i).ElmNodes(3));
    p4 = find(mesh.Nodes==mesh.Elm(i).ElmNodes(4));
    p5 = find(mesh.Nodes==mesh.Elm(i).ElmNodes(5));
    p6 = find(mesh.Nodes==mesh.Elm(i).ElmNodes(6));
    p7 = find(mesh.Nodes==mesh.Elm(i).ElmNodes(7));
    p8 = find(mesh.Nodes==mesh.Elm(i).ElmNodes(8));
    assembly(i,1:16) = [(p1-1)*2+1:(p1-1)*2+2 (p2-1)*2+1:(p2-1)*2+2 ...
                        (p3-1)*2+1:(p3-1)*2+2 (p4-1)*2+1:(p4-1)*2+2 ...
                        (p5-1)*2+1:(p5-1)*2+2 (p6-1)*2+1:(p6-1)*2+2 ...
                        (p7-1)*2+1:(p7-1)*2+2 (p8-1)*2+1:(p8-1)*2+2];
end
```

The third item to be changed is the sizing of the row_vec, col_vec and stiff_vec. These have to be increased from the 8x8 for each element to the 16x16 for each element. This is done as follows:

```
%Initialize global stiffness matrix;
%k_global = zeros(2*nnodes,2*nnodes);
k_global = sparse(2*mesh.NumNodes,2*mesh.NumNodes);
row_vec    = zeros(16*16*mesh.NumElm,1);
col_vec    = zeros(16*16*mesh.NumElm,1);
stiff_vec  = zeros(16*16*mesh.NumElm,1);
```

The next change is the size of B_ALL and EB_ALL which convert too:

```
%Initialize B_ALL and DB_ALL
B_ALL  = zeros(27*mesh.NumElm,16);
EB_ALL = zeros(27*mesh.NumElm,16);
```

This is due to the fact that at each node there will be three stresses, namely σ_1, σ_2 and τ_{12} . However since there are nine Gauss points it is necessary to maintain all the data for every Gauss point and then extrapolate the Gauss points to the eight nodes. Thus the size is 9x3=27.

The next change is the increasing of the number of co-ordinates required, due to the increase in the node number. At the same time the size of the local matrix B, m_elem and k_elem matrices are increased from 3x8, 8x8 and 8x8 to 3x16, 16x16 and 16x16 respectively. Thus the new co-ordinates of the 8 nodes and new sized matrices are given by:

```
% Find coordinates of element nodes
x1 = mesh.Coord(mesh.Elm(i).ElmNodes(1),1);
x2 = mesh.Coord(mesh.Elm(i).ElmNodes(2),1);
x3 = mesh.Coord(mesh.Elm(i).ElmNodes(3),1);
x4 = mesh.Coord(mesh.Elm(i).ElmNodes(4),1);
x5 = mesh.Coord(mesh.Elm(i).ElmNodes(5),1);
x6 = mesh.Coord(mesh.Elm(i).ElmNodes(6),1);
x7 = mesh.Coord(mesh.Elm(i).ElmNodes(7),1);
x8 = mesh.Coord(mesh.Elm(i).ElmNodes(8),1);

y1 = mesh.Coord(mesh.Elm(i).ElmNodes(1),2);
y2 = mesh.Coord(mesh.Elm(i).ElmNodes(2),2);
y3 = mesh.Coord(mesh.Elm(i).ElmNodes(3),2);
y4 = mesh.Coord(mesh.Elm(i).ElmNodes(4),2);
y5 = mesh.Coord(mesh.Elm(i).ElmNodes(5),2);
y6 = mesh.Coord(mesh.Elm(i).ElmNodes(6),2);
y7 = mesh.Coord(mesh.Elm(i).ElmNodes(7),2);
y8 = mesh.Coord(mesh.Elm(i).ElmNodes(8),2);

% Numerical integration loops
B       = zeros(3,16);
k_elem  = zeros(16,16);
m_elem  = zeros(16,16);
H       = zeros(5,5);
```

The fourth major change revolves around the fact that to accurately estimate the quad eight, it is necessary to implement Gauss Quadrature of order 3 (nine points). This is indicated by the figure below:

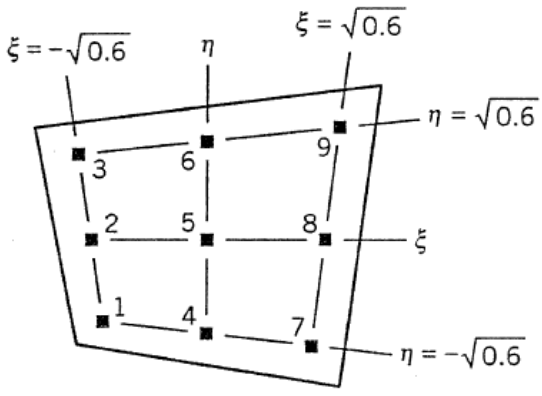


Figure C-7: Third order Gauss Quadrature node locations [28]

C.3.2 Gauss Quadrature Approximate Solution

Consequently the 'Gauss' loop will need to be run for nine points. Thus it will be composed of one loop varying from 1 to nine, where the definition of the Gauss point per loop is given by a CASE statement. The code used to implement the Gauss Quadrature is (approximate solution):

```
for iGauss = 1:9
    Gauss_pos=sqrt(0.6);
    factor=0.0;
    xi=0.0;
    eta=0.0;
    %procedure is as per Concepts and applications of FE analysis -
    %Cook issue 4 pg 214
    %define the values for each gauss point
    switch iGauss
        case (1)
            factor=25/81;
            xi  = -1*Gauss_pos;
            eta = -1*Gauss_pos;
        case (2)
            factor=40/81;
            xi  = -1*Gauss_pos;
            eta = 0.0;
        case (3)
            factor=25/81;
            xi  = -1*Gauss_pos;
            eta = Gauss_pos;
        case (4)
            factor=40/81;
            xi  = 0.0;
            eta = -1*Gauss_pos;
        case (5)
            factor=64/81;
            xi  = 0.0;
            eta = 0.0;
        case (6)
            factor=40/81;
            xi  = 0.0;
            eta = Gauss_pos;
        case (7)
            factor=25/81;
            xi  = Gauss_pos;
            eta = -1*Gauss_pos;
        case (8)
            factor=40/81;
            xi  = Gauss_pos;
            eta = 0.0;
        case (9)
            factor=25/81;
            xi  = Gauss_pos;
            eta = Gauss_pos;
        otherwise
            disp('Should never get here, ERROR (Guass)');
    end
```

Loop for nine Gauss Points

Case Statement as per figure 6.3-1 [28]

The loop will loop over the nine Gauss points selecting the values via the case statement together with the factor. The factor is the constant multiplier indicated in the third order Gauss Quadrature formula. The local stiffness matrix is then found using the third order Gauss Quadrature:

$$I \approx \frac{25}{81}\left(\phi_1 + \phi_3 + \phi_7 + \phi_9\right) + \frac{40}{81}\left(\phi_2 + \phi_4 + \phi_6 + \phi_8\right) + \frac{64}{81}\phi_5 \qquad \textbf{102}$$

This summation is implemented in the code using

```
%[16x16]=[16,16]+t[16x3][3x3][3x16]detJ
k_elem = k_elem + factor*mesh.Elm(i).t*B'*E*B*detJ;
```

C.3.3 Full integration

Using the full integration on page 211 of Cook [28], then the weighting functions are defined as:

- ζ/η=$\pm\sqrt{0.6}$ Weighting factor =5/9
- ζ/η=0 Weighting factor =8/9

This is implemented in Matlab using two for loops which run over three increments and the following arrays are used in the summation of the Gauss Quadrature:

```
%define arrays for xi eta and factor (1 to 9) for nine-point rule
Gauss_pos=sqrt(0.6);
factor_array=[5/9 8/9 5/9
              5/9 8/9 5/9];
xi_array=[-Gauss_pos 0 Gauss_pos];
eta_array=[-Gauss_pos 0 Gauss_pos];
```

Thus the summation of the stiffness matrix over the nine Gaussian points is defined as:

```
%^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
%[16x16]=[16,16]+t[16x3][3x3][3x16]detJ
k_elem = k_elem + factor_array(1,iGauss)*factor_array(2,jGauss)*mesh.Elm(i).t*B'*E*B*detJ;
```

With the corresponding Mass matrix being calculated using:

```
%calculate the mass for each element
rho=mat(mesh.Elm(i).Mat,4);
%            N=B2*N;%onver N for x and y to N for xi and eta
m_elem=m_elem+factor_array(1,iGauss)*factor_array(2,jGauss)*mesh.Elm(i).t*rho*N'*N*detJ;
```

These equations are based on equation 6.3-5 pg. 211 of Cook [28].

C.3.4 Further common Changes from Quad4 code

The next substantial change is the shape functions. Using reference [28] page 214, the shape functions are defined for the following node numbering:

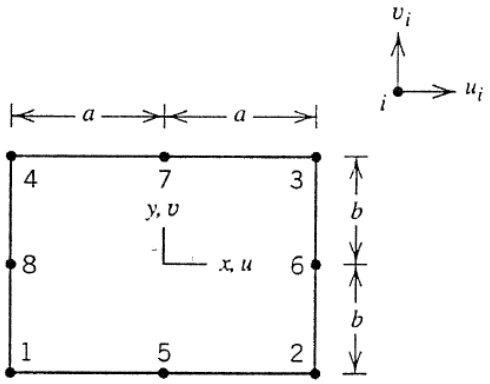


Figure C-8: General node number of quad8 element

The Isoparametric shape functions are given by:

$$N_1 = \frac{1}{4}(1-\xi)(1-\eta) - \frac{1}{2}(N_8 + N_5)$$

$$N_2 = \frac{1}{4}(1+\xi)(1-\eta) - \frac{1}{2}(N_6 + N_5)$$

$$N_3 = \frac{1}{4}(1+\xi)(1+\eta) - \frac{1}{2}(N_6 + N_7)$$

$$N_4 = \frac{1}{4}(1-\xi)(1+\eta) - \frac{1}{2}(N_8 + N_7)$$

$$N_5 = \frac{1}{2}(1-\xi^2)(1-\eta)$$

$$N_6 = \frac{1}{2}(1+\xi)(1-\eta^2)$$

$$N_7 = \frac{1}{2}(1-\xi^2)(1+\eta)$$

$$N_8 = \frac{1}{2}(1-\xi)(1-\eta^2)$$

These shape functions are implemented in the code as follows:

```
%define shape function
N5=0.25*(1-xi^2)*(1-eta);
N6=0.25*(1+xi)*(1-eta^2);
N7=0.25*(1-xi^2)*(1+eta);
N8=0.25*(1-xi)*(1-eta^2);
N1=0.25*(1-xi)*(1-eta)-0.25*(N8+N5);
N2=0.25*(1+xi)*(1-eta)-0.25*(N5+N6);
N3=0.25*(1+xi)*(1+eta)-0.25*(N6+N7);
N4=0.25*(1-xi)*(1+eta)-0.25*(N7+N8);
N=[ N1 0 N2 0 N3 0 N4 0 N5 0 N6 0 N7 0 N8 0
    0 N1 0 N2 0 N3 0 N4 0 N5 0 N6 0 N7 0 N8];
```

This will be used to calculate the mass matrix.

```
%calculate the mass for each element
rho=mat(mesh.Elm(i).Mat,4);
%            N=B2*N;%onver N for x and y to N for xi and eta
m_elem=m_elem+mesh.Elm(i).t*rho*N'*N*detJ;
```

It is well known that for an Isoparametric element the same shape functions are used to interpolate for both the x, y system as the ξ, η system. Thus using equation 6.2-5 (pg. 207) of reference [28], the Jacobian matrix is:

$$[J]=\begin{bmatrix} x,_{\xi} & y,_{\xi} \\ x,_{\eta} & y,_{\eta} \end{bmatrix}=\begin{bmatrix} \sum N_{i,\xi}x_i & \sum N_{i,\xi}y_i \\ \sum N_{i,\eta}x_i & \sum N_{i,\eta}y_i \end{bmatrix} \qquad \textbf{104}$$

This can then be written in matrix form:

$$\therefore [J]=\begin{bmatrix} N_{1,\eta} & N_{2,\eta} & N_{2,\eta} & N_{4,\eta} & N_{5,\eta} & N_{6,\eta} & N_{7,\eta} & N_{8,\eta} \\ N_{1,\xi} & N_{2,\xi} & N_{3,\xi} & N_{4,\xi} & N_{5,\xi} & N_{6,\xi} & N_{7,\xi} & N_{8,\xi} \end{bmatrix} \begin{bmatrix} x_1 & y_1 \\ x_2 & y_2 \\ x_3 & y_3 \\ x_4 & y_4 \\ x_5 & y_5 \\ x_6 & y_6 \\ x_7 & y_7 \\ x_8 & y_8 \end{bmatrix} = \begin{bmatrix} J_{11} & J_{12} \\ J_{21} & J_{22} \end{bmatrix} \qquad \textbf{105}$$

The derivatives with respect to (wrt) η are given by:

$$N_{1,\eta} = -\frac{1}{4}(1-\xi) - \frac{1}{2}\left(N_{8,\eta} + N_{5,\eta}\right)$$

$$N_{2,\eta} = -\frac{1}{4}(1+\xi) - \frac{1}{2}\left(N_{6,\eta} + N_{5,\eta}\right)$$

$$N_{3,\eta} = \frac{1}{4}(1+\xi) - \frac{1}{2}\left(N_{6,\eta} + N_{7,\eta}\right)$$

$$N_{4,\eta} = \frac{1}{4}(1-\xi) - \frac{1}{2}\left(N_{8,\eta} + N_{7,\eta}\right) \quad \textbf{106}$$

$$N_{5,\eta} = -\frac{1}{2}\left(1-\xi^2\right)$$

$$N_{6,\eta} = -\eta(1+\xi)$$

$$N_{7,\eta} = \frac{1}{2}\left(1-\xi^2\right)$$

$$N_{8,\eta} = -\eta(1-\xi)$$

The derivatives in wrt ξ are given by:

$$N_{1,\xi} = -\frac{1}{4}(1-\eta) - \frac{1}{2}\left(N_{8,\xi} + N_{5,\xi}\right)$$

$$N_{2,\xi} = \frac{1}{4}(1-\eta) - \frac{1}{2}\left(N_{6,\xi} + N_{5,\xi}\right)$$

$$N_{3,\xi} = \frac{1}{4}(1+\eta) - \frac{1}{2}\left(N_{6,\xi} + N_{7,\xi}\right)$$

$$N_{4,\xi} = -\frac{1}{4}(1+\eta) - \frac{1}{2}\left(N_{8,\xi} + N_{7,\xi}\right) \quad \textbf{107}$$

$$N_{5,\xi} = -\xi(1-\eta)$$

$$N_{6,\xi} = \frac{1}{2}\left(1-\eta^2\right)$$

$$N_{7,\xi} = -\xi(1+\eta)$$

$$N_{8,\xi} = -\frac{1}{2}\left(1-\eta^2\right)$$

This is implemented in the code as follows:

```
%define the derivatives of each shape function
der_SF=zeros(2,8);
%Xi derivatives (der wrt Xi)(defined out of order as needed)
der_SF(1,5)=-1*xi*(1-eta);
der_SF(1,6)=0.5*(1-eta^2);
der_SF(1,7)=-1*xi*(1+eta);
der_SF(1,8)=-0.5*(1-eta^2);
der_SF(1,1)=-0.25*(1-eta)-0.5*(der_SF(1,8)+der_SF(1,5));
der_SF(1,2)=0.25*(1-eta)-0.5*(der_SF(1,6)+der_SF(1,5));
der_SF(1,3)=0.25*(1+eta)-0.5*(der_SF(1,6)+der_SF(1,7));
der_SF(1,4)=-0.25*(1+eta)-0.5*(der_SF(1,8)+der_SF(1,7));
%Eta derivatives (der wrt Eta)(defined out of order as needed)
der_SF(2,5)=-0.5*(1-xi^2);
der_SF(2,6)=-1*eta*(1+xi);
der_SF(2,7)=0.5*(1-xi^2);
der_SF(2,8)=-1*eta*(1-xi);
der_SF(2,1)=-0.25*(1-xi)-0.5*(der_SF(2,8)+der_SF(2,5));
der_SF(2,2)=-0.25*(1+xi)-0.5*(der_SF(2,6)+der_SF(2,5));
der_SF(2,3)=0.25*(1+xi)-0.5*(der_SF(2,6)+der_SF(2,7));
der_SF(2,4)=0.25*(1-xi)-0.5*(der_SF(2,8)+der_SF(2,7));

%define the shape function derivative matrix
%[4x16]
D = [ der_SF(1,1) 0  der_SF(1,2) 0 der_SF(1,3) 0 der_SF(1,4) 0 ...
    der_SF(1,5) 0  der_SF(1,6) 0 der_SF(1,7) 0 der_SF(1,8) 0;
    der_SF(2,1)  0 der_SF(2,2)  0 der_SF(2,3)  0  der_SF(2,4)  0 ...
    der_SF(2,5)  0 der_SF(2,6)  0 der_SF(2,7)  0  der_SF(2,8)  0;
    0 der_SF(1,1) 0  der_SF(1,2) 0 der_SF(1,3) 0 der_SF(1,4) ...
    0 der_SF(1,5) 0  der_SF(1,6) 0 der_SF(1,7) 0 der_SF(1,8);
    0 der_SF(2,1)  0 der_SF(2,2)  0 der_SF(2,3)  0  der_SF(2,4)  ...
    0 der_SF(2,5)  0 der_SF(2,6)  0 der_SF(2,7)  0  der_SF(2,8)];
```

The next change is that of the Jacobian. This is due to the fact that the derivatives of the shape function matrix are now a 4x16. Consequently for the nodal locations in parallel columns it is required to multiply the non-zero terms with each column. Thus the implementation is:

```
%define the Jacobian
 J = D(1:2,1:2:15)*[x1 y1
                    x2 y2
                    x3 y3
                    x4 y4
                    x5 y5
                    x6 y6
                    x7 y7
                    x8 y8];
```

The standard derivation of the k_elem matrix is maintained as is from the Quad4 file. However, the generation of EB_ALL and B_ALL has to be updated to take into account the increase in node number thus it is modified too:

```
%update the matrices to account for 9 gauss points
B_ALL((i-1)*27+Gausspt*3+1:(i-1)*27+Gausspt*3+3,:)=B;
EB_ALL((i-1)*27+Gausspt*3+1:(i-1)*27+Gausspt*3+3,:)=E*B;
Gausspt = Gausspt + 1;
```

The next change is done when the local stiffness matrix is assembled into the global stiffness matrix. This is accomplished using:

```
%     k_global(pg,pg) = k_global(pg,pg) + k_elem;
      for ii=1:16
          for jj=1:16
              pos_vec = pos_vec + 1;
              row_vec(pos_vec) = pg(ii);
              col_vec(pos_vec) = pg(jj);
              stiff_vec(pos_vec) = k_elem(ii,jj);
          end
      end
```

Basically the implementation is the same as for the Quad4 code; however the loops now go to 16 instead of 8.

C.4 TRIA3

The next element type to be considered in this dissertation is the TRIA3 triangular element. Once this element is derived and verified against Nastran, then the TRIAX or axisymmetric element needed for the analysis of the horn and specimen can be developed and verified.

C.4.1 CST (Using analytical integration instead of numerical)

The linear triangle's displacement in 2 degrees of freedom is given by [29]:

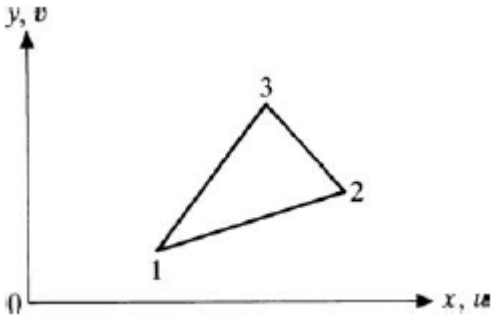


Figure C-9: Linear triangle

Thus the displacements u and v can be written in terms of unknown constants:

$$u = \alpha_1 + \alpha_2 x + \alpha_3 y \qquad \textbf{108}$$

$$v = \alpha_4 + \alpha_5 x + \alpha_6 y$$

This can be re-written in matrix form as:

$$\begin{bmatrix} u_1 \\ u_2 \\ u_3 \end{bmatrix} = [A] \begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \alpha_3 \end{bmatrix}$$

Similarly for the displacement, v:

$$\begin{bmatrix} v_1 \\ v_2 \\ v_3 \end{bmatrix} = [A] \begin{bmatrix} \alpha_4 \\ \alpha_5 \\ \alpha_6 \end{bmatrix}$$

Where

$$[A] = \begin{bmatrix} 1 & x_1 & y_1 \\ 1 & x_2 & y_2 \\ 1 & x_3 & y_3 \end{bmatrix}$$

Consequently

$$\begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \alpha_3 \end{bmatrix} = [A]^{-1} \begin{bmatrix} u_1 \\ u_2 \\ u_3 \end{bmatrix}$$

And

$$\begin{bmatrix} \alpha_4 \\ \alpha_5 \\ \alpha_6 \end{bmatrix} = [A]^{-1} \begin{bmatrix} v_1 \\ v_2 \\ v_3 \end{bmatrix}$$

Where

$$[A]^{-1} = \frac{1}{2A} \begin{bmatrix} A_1^0 & A_2^0 & A_3^0 \\ a_1 & a_2 & a_3 \\ b_1 & b_2 & b_3 \end{bmatrix}$$

In which

$A_1^0 = x_2 y_1 - x_1 y_2$

$A_2^0 = x_3 y_2 - x_2 y_3$

$A_3^0 = x_1 y_3 - x_3 y_1$

$a_1 = y_2 - y_1$

$a_2 = y_3 - y_2$

$a_3 = y_1 - y_3$

$b_1 = x_2 - x_1$

$b_2 = x_3 - x_2$

$b_3 = x_1 - x_3$

The determinant of [A] is in fact the area of A as proven below:

$$Area = \frac{1}{2}\det([A]) = \frac{1}{2}(A_1^0 + A_2^0 + A_3^0) = \frac{1}{2}(\mathrm{a_1b_2 - a_2b_1})$$

Based on the cross product and the fact that any triangle with one point at the origin and sides a, b with angle alpha subtended has an area of $\frac{1}{2}a.b.sin\alpha$ which is equivalent to the magnitude of the cross product in plane {the cross product of two vectors can be thought of as the area of a parallelogram}

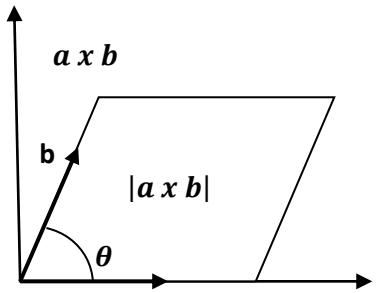


Figure C-10: Area of parallelogram is the magnitude of the cross product

Consequently the following is obtained:

$$\begin{bmatrix}\alpha_1\\ \alpha_2\\ \alpha_3\end{bmatrix} = \frac{1}{2A}\begin{bmatrix}A_1^0 & A_2^0 & A_3^0\\ a_1 & a_2 & a_3\\ b_1 & b_2 & b_3\end{bmatrix}\begin{bmatrix}u_1\\ u_2\\ u_3\end{bmatrix}$$

But from equation 108

$$u = \alpha_1 + \alpha_2 x + \alpha_3 y = [1 \quad x \quad y]\begin{bmatrix}\alpha_1\\ \alpha_2\\ \alpha_3\end{bmatrix} = [1 \quad x \quad y]\frac{1}{2A}\begin{bmatrix}A_1^0 & A_2^0 & A_3^0\\ a_1 & a_2 & a_3\\ b_1 & b_2 & b_3\end{bmatrix}\begin{bmatrix}u_1\\ u_2\\ u_3\end{bmatrix}$$

In addition generally for shape functions the form is:

$$u = [N]\begin{bmatrix}u_1\\ u_2\\ u_3\end{bmatrix}$$

Thus the shape functions are:

$$[N] = [1 \quad x \quad y]\frac{1}{2A}\begin{bmatrix}A_1^0 & A_2^0 & A_3^0\\ a_1 & a_2 & a_3\\ b_1 & b_2 & b_3\end{bmatrix} = \frac{1}{2A}[A_1^0 + a_1x + b_1y \quad A_2^0 + a_2x + b_2y \quad A_3^0 + a_3x + b_3y]$$

Hence,

$$N_i = \frac{1}{2A}(A_i^0 + a_ix + b_iy)$$

Thus for both u and v:

$$\begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} N_1 & 0 & N_2 & 0 & N_3 & 0 \\ 0 & N_1 & 0 & N_2 & 0 & N_3 \end{bmatrix} \begin{Bmatrix} u_1 \\ v_1 \\ u_2 \\ v_2 \\ u_3 \\ v_3 \end{Bmatrix}$$

Where x and y are any points inside the element. If the x and y location is on one of the sides then one of the shape functions will be zero. To define the mass matrix the kinetic energy is used:

$$Ek = \frac{1}{2}\int_v \langle \dot{\varepsilon}(x,y) \rangle \rho \langle \dot{\varepsilon}(x,y) \rangle dV$$

$$Ek = \frac{1}{2}\int_v \langle \ddot{u}(x,y) \rangle \rho \langle \ddot{u}(x,y) \rangle dV$$

Thus for constant thickness

$$Ek = \frac{1}{2}\int_v \langle \ddot{u}(x,y) \rangle \rho \langle \ddot{u}(x,y) \rangle t dx dy$$

Using the fact that:

$$\begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} N_1 & 0 & N_2 & 0 & N_3 & 0 \\ 0 & N_1 & 0 & N_2 & 0 & N_3 \end{bmatrix} \begin{Bmatrix} u_1 \\ v_1 \\ u_2 \\ v_2 \\ u_3 \\ v_3 \end{Bmatrix} = [N]\{u_e\}$$

Consequently

$$Ek = \{\ddot{u_e}\}^T \int_{x,y} t\rho [N]^T [N]\, dxdy \{\ddot{u_e}\}$$

Thus the mass matrix per element is defined as:

$$[m]_e = \int_{x,y} t\rho [N]^T [N]\, dxdy$$

Thus expanding:

$$[m]_e = t\rho \iint \begin{bmatrix} N_1N_1 & 0 & N_1N_2 & 0 & N_1N_3 & 0 \\ 0 & N_1N_1 & 0 & N_1N_2 & 0 & N_1N_3 \\ N_2N_1 & 0 & N_2N_2 & 0 & N_2N_3 & 0 \\ 0 & N_2N_1 & 0 & N_2N_2 & 0 & N_2N_3 \\ N_3N_1 & 0 & N_3N_2 & 0 & N_3N_3 & 0 \\ 0 & N_3N_1 & 0 & N_3N_2 & 0 & N_3N_3 \end{bmatrix} dA$$

This can be solved using the fact that, integration over a triangle area is given by:

$$\int L^k{}_i \, L_j{}^m L_l{}^n dA = \frac{k!\, m!\, n!}{(2+k+m+n)!} 2A$$

Thus

$$\int N_1 N_1 \, dA = \int N_1{}^2 \, dA = \frac{2!\, 0!\, 0!}{(2+2+0+0)!} 2A = \frac{A}{6}$$

And

$$\int N_1 N_2 \, dA = \frac{1!\, 1!\, 0!}{(2+1+1+0)!} 2A = \frac{1}{12} A$$

Consequently

$$[m]_e = \frac{At\rho}{12} \begin{bmatrix} 2 & 0 & 1 & 0 & 1 & 0 \\ 0 & 2 & 0 & 1 & 0 & 1 \\ 1 & 0 & 2 & 0 & 1 & 0 \\ 0 & 1 & 0 & 2 & 0 & 1 \\ 1 & 0 & 2 & 0 & 2 & 0 \\ 0 & 1 & 0 & 1 & 0 & 2 \end{bmatrix}$$

Similarly for the stiffness matrix the potential (strain) energy of an element is considered; hence the stiffness matrix of each element is given by:

$$[k]_e = \int_A t[B]^T [D]\,[B] dA$$

Where

$$[B] = \begin{bmatrix} \frac{\partial}{\partial \mathrm{x}} & 0 \\ 0 & \frac{\partial}{\partial \mathrm{y}} \\ \frac{\partial}{\partial \mathrm{y}} & \frac{\partial}{\partial \mathrm{x}} \end{bmatrix} \begin{bmatrix} N_1 & 0 & N_2 & 0 & N_3 & 0 \\ 0 & N_1 & 0 & N_2 & 0 & N_3 \end{bmatrix}$$

Function defined in area co-ordinates can be solved in Cartesian co-ordinates using:

$$\frac{\partial}{\partial x} = \sum_{i=1}^{3} \frac{\partial L_i}{\partial x} \frac{\partial}{\partial L_i} \qquad and \qquad \frac{\partial}{\partial y} = \sum_{i=1}^{3} \frac{\partial L_i}{\partial y} \frac{\partial}{\partial L_i}$$

If

$$N_i = L_i = \frac{1}{2A}\left(A_i^0 + a_i x + b_i y\right)$$

$$\frac{\partial}{\partial x} = \frac{1}{2A}\sum_{i=1}^{3} a_i \frac{\partial}{\partial L_i} \qquad and \qquad \frac{\partial}{\partial y} = \frac{1}{2A}\sum_{i=1}^{3} b_i \frac{\partial}{\partial L_i}$$

Therefore,

$\frac{\partial N_j}{\partial x} = \frac{1}{2A}\sum_{i=1}^{3} a_i \frac{\partial N_j}{\partial N_i} = \frac{a_i}{2A} = \frac{a_j}{2A}$
$\frac{\partial N_j}{\partial y} = \frac{1}{2A}\sum_{i=1}^{3} b_i \frac{\partial N_j}{\partial N_i} = \frac{b_i}{2A} = \frac{b_j}{2A}$ Since $\begin{cases}\frac{\partial N_j}{\partial N_i} = 1 \ (i = j)\\ \frac{\partial N_j}{\partial N_i} = 0 \ (i \neq j)\end{cases}$

As a result

$$[B] = \frac{1}{2A}\begin{bmatrix} a_1 & 0 & a_2 & 0 & a_3 & 0 \\ 0 & b_1 & 0 & b_2 & 0 & b_3 \\ b_1 & a_1 & b_2 & a_2 & b_3 & a_3 \end{bmatrix}$$

Hence the element stiffness is:

$$E_p = \frac{1}{2}\int_v \{\varepsilon\}^T [E]\{\varepsilon\} dV$$

$$E_p = \frac{t}{2}\int_v \{d\}^T [B]^T [E][B]\{d\} dA$$

But all variables are not functions of x or y thus,

$$E_p = \frac{t}{2}\{d\}^T \int_v dA\, [B]^T [E][B]\{d\}$$

$$E_p = \frac{At}{2}\{d\}^T [B]^T [E][B]\{d\}$$

Thus the stiffness is extracted as:

$$[k]_e = \frac{t}{4A}\begin{bmatrix} a_1 & 0 & a_2 & 0 & a_3 & 0 \\ 0 & b_1 & 0 & b_2 & 0 & b_3 \\ b_1 & a_1 & b_2 & a_2 & b_3 & a_3 \end{bmatrix}^{\mathrm{T}} [D] \begin{bmatrix} a_1 & 0 & a_2 & 0 & a_3 & 0 \\ 0 & b_1 & 0 & b_2 & 0 & b_3 \\ b_1 & a_1 & b_2 & a_2 & b_3 & a_3 \end{bmatrix}$$

Note: areas cancel with [B] area and the half value (1/2) from energy equation is not part of the stiffness.

C.4.2 Linear triangle as per Kwon

The shape functions are as per Kwo ("The finite element method using Matlab", CRC Press 1996):

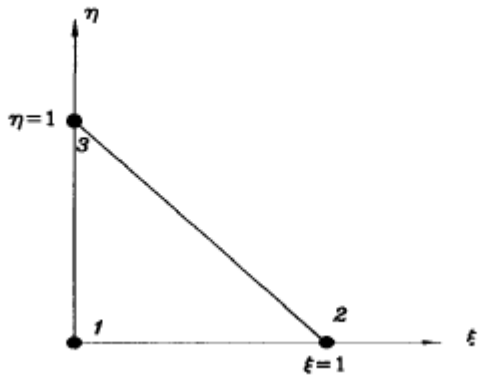


Figure C-11: Three node triangular element

Thus the four shape functions are defined by; see reference [28] pg. 206:

$$N_1 = 1 - \xi - \eta$$
$$N_2 = \xi$$
$$N_3 = \eta \qquad \textbf{109}$$

Thus for the 2d case the strain is defined as:

$$\{\varepsilon\} = \begin{Bmatrix} \varepsilon_{xx} \\ \varepsilon_{yy} \\ \gamma_{xy} \end{Bmatrix} = \begin{bmatrix} \frac{\partial}{\partial x} & 0 \\ 0 & \frac{\partial}{\partial y} \\ \frac{\partial}{\partial y} & \frac{\partial}{\partial x} \end{bmatrix} \begin{Bmatrix} \sum_{i=1}^{3} N_i . u_i \\ \sum_{i=1}^{3} N_i . w_i \end{Bmatrix} = [\partial][N(x,y]\{d\} \qquad \textbf{110}$$

[3x1] [3x2] [2x1]

This is equivalent to:

$$\{\varepsilon\} = \begin{Bmatrix} \varepsilon_{xx} \\ \varepsilon_{yy} \\ \gamma_{xy} \end{Bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{bmatrix} \begin{Bmatrix} \frac{\partial u}{\partial x} \\ \frac{\partial u}{\partial y} \\ \frac{\partial v}{\partial x} \\ \frac{\partial v}{\partial y} \end{Bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{bmatrix} \begin{bmatrix} \frac{\partial}{\partial x} & 0 \\ \frac{\partial}{\partial y} & 0 \\ 0 & \frac{\partial}{\partial x} \\ 0 & \frac{\partial}{\partial y} \end{bmatrix} \begin{Bmatrix} \sum_{i=1}^{3} N_i . u_i \\ \sum_{i=1}^{3} N_i . w_i \end{Bmatrix} = [\partial][N(x,y]\{d\}$$

[3x4] [4x1] = [3x4] [4x2] [2x1]

$$\{\varepsilon\} = \begin{Bmatrix} \varepsilon_{xx} \\ \varepsilon_{yy} \\ \gamma_{xy} \end{Bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{bmatrix} \begin{Bmatrix} \frac{\partial u}{\partial x} \\ \frac{\partial u}{\partial y} \\ \frac{\partial v}{\partial x} \\ \frac{\partial v}{\partial y} \end{Bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{bmatrix} \begin{bmatrix} \frac{\partial}{\partial x} & 0 \\ \frac{\partial}{\partial y} & 0 \\ 0 & \frac{\partial}{\partial x} \\ 0 & \frac{\partial}{\partial y} \end{bmatrix} \begin{bmatrix} N_1 & 0 & N_2 & 0 & N_3 & 0 \\ 0 & N_1 & 0 & N_2 & 0 & N_3 \end{bmatrix} \begin{Bmatrix} u_1 \\ v_1 \\ u_2 \\ v_2 \\ u_3 \\ v_3 \end{Bmatrix}$$

This can be converted to the functions of (ξ,η) using the Jacobian (i.e. u=∅)

$$\begin{Bmatrix} \frac{\partial \varphi}{\partial \xi} \\ \frac{\partial \varphi}{\partial \eta} \end{Bmatrix} = [J] \begin{Bmatrix} \frac{\partial \varphi}{\partial x} \\ \frac{\partial \varphi}{\partial y} \end{Bmatrix} = \begin{bmatrix} \frac{\partial x}{\partial \xi} & \frac{\partial y}{\partial \xi} \\ \frac{\partial x}{\partial \eta} & \frac{\partial y}{\partial \eta} \end{bmatrix} \begin{Bmatrix} \frac{\partial \varphi}{\partial x} \\ \frac{\partial \varphi}{\partial y} \end{Bmatrix}$$

The scalar field can be any field such as x and y.

Where

$$[J] = \begin{bmatrix} \frac{\partial x}{\partial \xi} & \frac{\partial y}{\partial \xi} \\ \frac{\partial x}{\partial \eta} & \frac{\partial y}{\partial \eta} \end{bmatrix} = \begin{bmatrix} \sum_{i=1}^{3} \frac{\partial}{\partial \xi} N_i(\xi,\eta).x_i & \sum_{i=1}^{3} \frac{\partial}{\partial \xi} N_i(\xi,\eta).y_i \\ \sum_{i=1}^{3} \frac{\partial}{\partial \eta} N_i(\xi,\eta).x_i & \sum_{i=1}^{3} \frac{\partial}{\partial \eta} N_i(\xi,\eta).y_i \end{bmatrix} \quad \textbf{111}$$

This can be visualized as:

$$[J] = \begin{bmatrix} \frac{\partial x}{\partial \xi} & \frac{\partial y}{\partial \xi} \\ \frac{\partial x}{\partial \eta} & \frac{\partial y}{\partial \eta} \end{bmatrix} = \begin{bmatrix} \frac{\partial}{\partial \xi} N_1(\xi,\eta) & \frac{\partial}{\partial \xi} N_2(\xi,\eta) & \frac{\partial}{\partial \xi} N_3(\xi,\eta) \\ \frac{\partial}{\partial \eta} N_2(\xi,\eta) & \frac{\partial}{\partial \eta} N_2(\xi,\eta) & \frac{\partial}{\partial \eta} N_3(\xi,\eta) \end{bmatrix} \begin{bmatrix} x_1 & y_1 \\ x_2 & y_2 \\ x_3 & y_3 \end{bmatrix} \quad \textbf{112}$$

[2x2] [2x3] [3x2]

The inverse of the Jacobian is given by:

$$[J]^{-1} = \frac{1}{det([J])} \begin{bmatrix} \frac{\partial y}{\partial \eta} & -\frac{\partial y}{\partial \xi} \\ -\frac{\partial x}{\partial \eta} & \frac{\partial x}{\partial \xi} \end{bmatrix} \quad \textbf{113}$$

This inverse will be sufficient for:

$$\begin{Bmatrix} \frac{\partial u}{\partial x} \\ \frac{\partial u}{\partial y} \end{Bmatrix} = [J]^{-1} \begin{Bmatrix} \frac{\partial u}{\partial \xi} \\ \frac{\partial u}{\partial \eta} \end{Bmatrix}$$

However in this case both u and v are required, thus the following [J]* matrix defined:

$$\{du\} = \begin{Bmatrix} \frac{\partial u}{\partial x} \\ \frac{\partial u}{\partial y} \\ \frac{\partial v}{\partial x} \\ \frac{\partial v}{\partial y} \end{Bmatrix} = \frac{1}{det([J])} \begin{bmatrix} \frac{\partial y}{\partial \eta} & -\frac{\partial y}{\partial \xi} & 0 & \\ -\frac{\partial x}{\partial \eta} & \frac{\partial x}{\partial \xi} & & \\ & & \frac{\partial y}{\partial \eta} & -\frac{\partial y}{\partial \xi} \\ 0 & & -\frac{\partial x}{\partial \eta} & \frac{\partial x}{\partial \xi} \end{bmatrix} \begin{Bmatrix} \frac{\partial u}{\partial \xi} \\ \frac{\partial u}{\partial \eta} \\ \frac{\partial v}{\partial \xi} \\ \frac{\partial v}{\partial \eta} \end{Bmatrix} = [J]^*[\partial N]\{d\}$$

[4x1] [4x4] [4x1] = [4x4] [4x6] [6x1]

Where

$$[J]^* = \frac{1}{det([J])} \begin{bmatrix} \frac{\partial y}{\partial \eta} & -\frac{\partial y}{\partial \xi} & 0 & \\ -\frac{\partial x}{\partial \eta} & \frac{\partial x}{\partial \xi} & & \\ & & \frac{\partial y}{\partial \eta} & -\frac{\partial y}{\partial \xi} \\ 0 & & -\frac{\partial x}{\partial \eta} & \frac{\partial x}{\partial \xi} \end{bmatrix}$$

$$[\partial N] = \begin{bmatrix} \frac{\partial}{\partial \xi}N_1(\xi,\eta) & 0 & \frac{\partial}{\partial \xi}N_2(\xi,\eta) & 0 & \frac{\partial}{\partial \xi}N_3(\xi,\eta) & 0 \\ \frac{\partial}{\partial \eta}N_1(\xi,\eta) & 0 & \frac{\partial}{\partial \eta}N_2(\xi,\eta) & 0 & \frac{\partial}{\partial \eta}N_3(\xi,\eta) & 0 \\ 0 & \frac{\partial}{\partial \xi}N_1(\xi,\eta) & 0 & \frac{\partial}{\partial \xi}N_2(\xi,\eta) & 0 & \frac{\partial}{\partial \xi}N_3(\xi,\eta) \\ 0 & \frac{\partial}{\partial \eta}N_1(\xi,\eta) & 0 & \frac{\partial}{\partial \eta}N_2(\xi,\eta) & 0 & \frac{\partial}{\partial \eta}N_3(\xi,\eta) \end{bmatrix}$$

$$\{d\} = \begin{Bmatrix} u_1 \\ v_1 \\ u_2 \\ v_2 \\ u_3 \\ v_3 \end{Bmatrix}$$

$$\frac{\partial}{\partial \xi}N_1(\xi,\eta) = -1$$
$$\frac{\partial}{\partial \eta}N_1(\xi,\eta) = -1$$
$$\frac{\partial}{\partial \xi}N_2(\xi,\eta) = 1$$
$$\frac{\partial}{\partial \eta}N_2(\xi,\eta) = 0$$
$$\frac{\partial}{\partial \xi}N_3(\xi,\eta) = 0$$
$$\frac{\partial}{\partial \eta}N_3(\xi,\eta) = 1$$

Simplifying

$$[\partial N] = \begin{bmatrix} -1 & 0 & 1 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 & 1 & 0 \\ 0 & -1 & 0 & 1 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 & 1 \end{bmatrix}$$

Thus the stiffness matrix is calculated based on the potential (strain) energy of the structure:

$$K = \frac{1}{2}\int_{v} \langle \varepsilon(x,y) \rangle [D] \{\varepsilon(x,y)\}\, dV$$

For constant thickness

$$K = \frac{1}{2}\iint_{A} \langle \varepsilon(x,y) \rangle [D] \{\varepsilon(x,y)\} t\, dxdy$$

Thus to convert to parametric element it is necessary to use the determinant which is assumed to be the physical incremental area for the element. However in the case of triangle elements the determinant is 2A thus determinant needs to be reduced by half.

$$K = \frac{1}{2}\int_{\substack{\xi=-1 \\ \eta=-1}}^{\substack{\xi=1 \\ \eta=1}} \langle \varepsilon(x,y) \rangle [D] \{\varepsilon(x,y)\} t\, [detA] \frac{1}{2} d\xi d\eta$$

C.5 Axis-symmetric QUAD4 element

The main equations for the Axis-symmetric element are the same for the 2d elements; however the integrals are now carried out over the normal two dimensions (x and y for 2d and r and z for axis-symmetric case) with the addition of a third dimension for axis symmetric being the angular component. Thus the following integrals are defined for the element stiffness and mass matrices respectively [28]:

$$[K]_{elm} = \int_{-b}^{b}\int_{-a}^{a}\int_{-\pi}^{\pi} [B]^T [E][B]\, r d\theta dx dy$$

Where

a - is the half width in the x-axis direction of the quad4 element

b - is the half height of the quad4 element in the y-direction

$$[M]_{elm} = \int_{-b}^{b}\int_{-a}^{a}\int_{-\pi}^{\pi} [N]^T [N]\, r d\theta dx dy$$

The matrices for the shape functions remain the same as before i.e.:

$$[N] = \begin{bmatrix} N_1 & 0 & N_2 & 0 & N_3 & 0 & N_4 & 0 \\ 0 & N_1 & 0 & N_2 & 0 & N_3 & 0 & N_4 \end{bmatrix}$$

The degrees of freedom u and v and now in terms of axes r and z respectively. The calculation of the material matrix [E] however now has a fourth row to allow for strain in the hoop (theta direction) of the element. Thus the isotropic material the new [E] matrix is:

$$[E] = \frac{E(1-v)}{(1+v)(1-2v)} \begin{bmatrix} 1 & \frac{v}{1-v} & \frac{v}{1-v} & 0 \\ \frac{v}{1-v} & 1 & \frac{v}{1-v} & 0 \\ \frac{v}{1-v} & \frac{v}{1-v} & 1 & 0 \\ 0 & 0 & 0 & \frac{1-2v}{2-2v} \end{bmatrix}$$

The stress vector is thus of the form:

$$\{\sigma\} = \begin{Bmatrix} \sigma_r \\ \sigma_\theta \\ \sigma_z \\ \tau_{zr} \end{Bmatrix}$$

Similarly the strain is defined by:

$$\{\varepsilon\} = \begin{Bmatrix} \varepsilon_r \\ \varepsilon_\theta \\ \varepsilon_z \\ \gamma_{zr} \end{Bmatrix}$$

Where the circumferential strain is defined as the change in the circumference divided by the original circumference:

$$\varepsilon_\theta = \frac{2\pi(r+u) - 2\pi r}{2\pi r} = \frac{u}{r}$$

As a result the strain as calculated from the displacements is given by:

$$\{\varepsilon\} = \begin{Bmatrix} \varepsilon_r \\ \varepsilon_\theta \\ \varepsilon_z \\ \gamma_{zr} \end{Bmatrix} = [\partial] \begin{Bmatrix} u \\ w \end{Bmatrix}$$

Where

$$[\partial] = \begin{bmatrix} \frac{\partial}{\partial r} & 0 \\ \frac{1}{r} & 0 \\ 0 & \frac{\partial}{\partial z} \\ \frac{\partial}{\partial z} & \frac{\partial}{\partial r} \end{bmatrix}$$

And

$$\begin{Bmatrix} u \\ w \end{Bmatrix} = \begin{bmatrix} N_1 & 0 & N_2 & 0 & N_3 & 0 & N_4 & 0 \\ 0 & N_1 & 0 & N_2 & 0 & N_3 & 0 & N_4 \end{bmatrix} \begin{Bmatrix} u_1 \\ v_1 \\ u_2 \\ \vdots \\ v_4 \end{Bmatrix}$$

$$[B] = [\partial][N]$$

However if the second row of $[\partial]$ is removed then the formulation is the same as that for the quad4 element as done above. But r replaces x and y is replaced by z. Thus the modified code for axis symmetric analysis is a direct copy of the Quad4 code except that the Gauss Quadrature will now be completed over the three integrals.

Note: along the axis of revolution the displacement in the x/r direction i.e. u needs to remain ZERO else there will be a discontinuity or pin hole. Thus in the code the displacements of all the nodes at r=0 must be zeroed. The typical element co-ordinate system is indicated in the figure below:

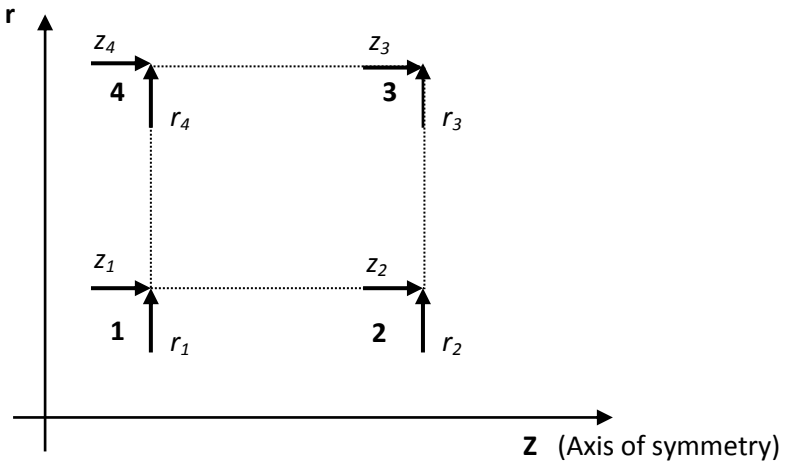


Figure C-12: Axis definition for an element

$$[N] = \begin{bmatrix} N_1 & 0 & N_2 & 0 & N_3 & 0 & N_4 & 0 \\ 0 & N_1 & 0 & N_2 & 0 & N_3 & 0 & N_4 \end{bmatrix}$$

The standard shape functions are given by:

$$[N] = \frac{1}{4ab}\begin{bmatrix} (a-x)(b-y) & 0 & (a+x)(b-y) & 0 & (a+x)(b+y) & 0 & (a-x)(b+y) & 0 \\ 0 & (a-x)(b-y) & 0 & (a+x)(b-y) & 0 & (a+x)(b+y) & 0 & (a-x)(b+y) \end{bmatrix}$$

In axis-symmetric models:

$$x = r$$

$$y = z$$

Now if the element is considered as isoperimetric

$$a = 1$$

$$b = 1$$

$$\xi = r$$

$$\eta = z$$

$$[N] = \frac{1}{4}\begin{bmatrix} (1-\xi)(1-\eta) & 0 & (1+\xi)(1-\eta) & 0 & (1+\xi)(1+\eta) & 0 & (1-\xi)(1+\eta) & 0 \\ 0 & (1-\xi)(1-\eta) & 0 & (1+\xi)(1-\eta) & 0 & (1+\xi)(1+\eta) & 0 & (1-\xi)(1+\eta) \end{bmatrix}$$

$$[J] = \begin{bmatrix} x,_{\xi} & y,_{\xi} \\ x,_{\eta} & y,_{\eta} \end{bmatrix} = \begin{bmatrix} \sum N_{i,\xi} x_i & \sum N_{i,\xi} y_i \\ \sum N_{i,\eta} x_i & \sum N_{i,\eta} y_i \end{bmatrix} = \begin{bmatrix} r,_{\xi} & z,_{\xi} \\ r,_{\eta} & z,_{\eta} \end{bmatrix} = \begin{bmatrix} \sum N_{i,\xi} r_i & \sum N_{i,\xi} z_i \\ \sum N_{i,\eta} r_i & \sum N_{i,\eta} z_i \end{bmatrix} \quad \textbf{115}$$

Thus for the first term of the matrix above the following is obtained:

J11

$$\sum \mathrm{N}_{\mathrm{i},\xi}\mathrm{x}_{\mathrm{i}} = \frac{\partial}{\partial\xi}\left[\frac{1}{4}(1-\xi)(1-\eta)\right]\mathrm{x}_1 + \frac{\partial}{\partial\xi}\left[\frac{1}{4}(1+\xi)(1-\eta)\right]\mathrm{x}_2 + \frac{\partial}{\partial\xi}\left[\frac{1}{4}(1+\xi)(1+\eta)\right]\mathrm{x}_3 + \frac{\partial}{\partial\xi}\left[\frac{1}{4}(1-\xi)(1+\eta)\right]\mathrm{x}_4$$

$$\therefore \sum \mathrm{N}_{\mathrm{i},\xi}\mathrm{x}_{\mathrm{i}} = \left[\frac{-1}{4}(1-\eta)\right]\mathrm{x}_1 + \left[\frac{1}{4}(1-\eta)\right]\mathrm{x}_2 + \left[\frac{1}{4}(1+\eta)\right]\mathrm{x}_3 + \left[\frac{-1}{4}(1+\eta)\right]\mathrm{x}_4 \quad \textbf{116}$$

Similarly,

J12

$$\therefore \sum \mathrm{N}_{\mathrm{i},\xi}\mathrm{y}_{\mathrm{i}} = \left[\frac{-1}{4}(1-\eta)\right]\mathrm{y}_1 + \left[\frac{1}{4}(1-\eta)\right]\mathrm{y}_2 + \left[\frac{1}{4}(1+\eta)\right]\mathrm{y}_3 + \left[\frac{-1}{4}(1+\eta)\right]\mathrm{y}_4 \quad \textbf{117}$$

The second row of derivative is done with respect to η, hence:

J21

$$\sum \mathrm{N}_{\mathrm{i},\eta}\mathrm{x}_{\mathrm{i}} = \frac{\partial}{\partial\eta}\left[\frac{1}{4}(1-\xi)(1-\eta)\right]\mathrm{x}_1 + \frac{\partial}{\partial\eta}\left[\frac{1}{4}(1+\xi)(1-\eta)\right]\mathrm{x}_2 + \frac{\partial}{\partial\eta}\left[\frac{1}{4}(1+\xi)(1+\eta)\right]\mathrm{x}_3 + \frac{\partial}{\partial\eta}\left[\frac{1}{4}(1-\xi)(1+\eta)\right]\mathrm{x}_4$$

$$\therefore \sum \mathrm{N}_{\mathrm{i},\eta}\mathrm{x}_{\mathrm{i}} = \left[\frac{-1}{4}(1-\xi)\right]\mathrm{x}_1 + \left[\frac{-1}{4}(1+\xi)\right]\mathrm{x}_2 + \left[\frac{1}{4}(1+\xi)\right]\mathrm{x}_3 + \left[\frac{1}{4}(1-\xi)\right]\mathrm{x}_4 \quad \textbf{118}$$

Similarly,

J22

$$\therefore \sum \mathrm{N}_{\mathrm{i},\eta}\mathrm{y}_{\mathrm{i}} = \left[\frac{-1}{4}(1-\xi)\right]\mathrm{y}_1 + \left[\frac{-1}{4}(1+\xi)\right]\mathrm{y}_2 + \left[\frac{1}{4}(1+\xi)\right]\mathrm{y}_3 + \left[\frac{1}{4}(1-\xi)\right]\mathrm{y}_4 \quad \textbf{119}$$

This can then be written in matrix form:

$$\therefore [\mathrm{J}] = \frac{1}{4}\begin{bmatrix} -1+\eta & 1-\eta & 1+\eta & -1-\eta \\ -1+\xi & -1-\xi & 1+\xi & 1-\xi \end{bmatrix}\begin{bmatrix} \mathrm{x}_1 & \mathrm{y}_1 \\ \mathrm{x}_2 & \mathrm{y}_2 \\ \mathrm{x}_3 & \mathrm{y}_3 \\ \mathrm{x}_4 & \mathrm{y}_4 \end{bmatrix} = \begin{bmatrix} \mathrm{J}_{11} & \mathrm{J}_{12} \\ \mathrm{J}_{21} & \mathrm{J}_{22} \end{bmatrix} \qquad \textbf{120}$$

But in axis-symmetric models:

$$x = r$$

$$y = z$$

$$\therefore [J] = \frac{1}{4}\begin{bmatrix} -1+\eta & 1-\eta & 1+\eta & -1-\eta \\ -1+\xi & -1-\xi & 1+\xi & 1-\xi \end{bmatrix}\begin{bmatrix} r_1 & z_1 \\ r_2 & z_2 \\ r_3 & z_3 \\ r_4 & z_4 \end{bmatrix} = \begin{bmatrix} J_{11} & J_{12} \\ J_{21} & J_{22} \end{bmatrix} \qquad \textbf{121}$$

$$\therefore [J] = \begin{bmatrix} J_{11} & J_{12} \\ J_{21} & J_{22} \end{bmatrix} = \frac{1}{4}\begin{bmatrix} (-1+\eta)r_1 + (1-\eta)r_2 + (1+\eta)r_3 + (-1-\eta)r_4 & (-1+\eta)z_1 + (1-\eta)z_2 + (1+\eta)z_3 + (-1-\eta)z_4 \\ (-1+\xi)r_1 + (-1-\xi)r_2 + (1+\xi)r_3 + (1-\xi)r_4 & (-1+\xi)z_1 + (-1-\xi)z_2 + (1+\xi)z_3 + (1-\xi)z_4 \end{bmatrix}$$

If :

$r_{ij} = r_i - r_j$

$z_{ij} = z_i - z_j$

$$\therefore [J] = \frac{1}{4}\begin{bmatrix} r_{21} + r_{34} + (r_{12} + r_{34})\eta & z_{21} + z_{34} + (z_{12} + z_{34})\eta \\ r_{32} + r_{41} + (r_{14} + r_{32})\xi & z_{32} + z_{41} + (z_{14} + z_{32})\xi \end{bmatrix}$$

This is equivalent to:

$$\therefore [J] = \frac{1}{4}\begin{bmatrix} r_{21} + r_{34} + (r_{12} + r_{34})\eta & z_{21} + z_{34} + (z_{12} + z_{34})\eta \\ r_{32} + r_{41} + (r_{12} + r_{34})\xi & z_{32} + z_{41} + (z_{12} + z_{34})\xi \end{bmatrix}$$

The determinant of the Jacobian is given by:

$$\therefore \det[J] = J_{11}J_{22} - J_{12}J_{21}$$

$$\therefore \det[J] = \frac{1}{4}[r_{21} + r_{34} + (r_{12} + r_{34})\eta]\frac{1}{4}[z_{32} + z_{41} + (z_{12} + z_{34})\xi] - \frac{1}{4}[z_{21} + z_{34} + (z_{12} + z_{34})\eta]\frac{1}{4}[r_{32} + r_{41} + (r_{12} + r_{34})\xi]$$

$$\therefore \det[J] =$$

$$\frac{1}{16}[r_{21}z_{32} + r_{21}z_{41} + r_{21}(z_{12} + z_{34})\xi + r_{34}z_{32} + r_{34}z_{41} + r_{34}(z_{12} + z_{34})\xi + z_{32}(r_{12} + r_{34})\eta + z_{41}(r_{12} + r_{34})\eta + (z_{12} + z_{34})\xi(r_{12} + r_{34})\eta]$$

$$-\frac{1}{16}[z_{21}r_{32} + z_{21}r_{41} + z_{21}(r_{12} + r_{34})\xi + z_{34}r_{32} + z_{34}r_{41} + z_{34}(r_{12} + r_{34})\xi + r_{32}(z_{12} + z_{34})\eta + r_{41}(z_{12} + z_{34})\eta + (z_{12} + z_{34})\eta(r_{12} + r_{34})\xi]$$

If the terms are grouped according to those dependent on each variable then the above can be simplified too:

$\eta\xi$

$$\frac{1}{16}\left[(z_{12}+z_{34})\xi(r_{12}+r_{34})\eta\right]-\frac{1}{16}\left[(z_{12}+z_{34})\eta(r_{12}+r_{34})\xi\right]$$
$$=\frac{1}{16}\xi\eta\left[(z_{12}+z_{34})(r_{12}+r_{34})-(z_{12}+z_{34})(r_{12}+r_{34})\right]$$
$$=0$$

η

$$\frac{1}{16}\left[z_{32}(r_{12}+r_{34})\eta+z_{41}(r_{12}+r_{34})\eta\right]-\frac{1}{16}\left[r_{32}(z_{12}+z_{34})\eta+r_{41}(z_{12}+z_{34})\eta\right]$$
$$=\frac{1}{16}\eta\left[z_{32}(r_{12}+r_{34})+z_{41}(r_{12}+r_{34})-r_{32}(z_{12}+z_{34})-r_{41}(z_{12}+z_{34})\right]$$
$$=\frac{1}{16}\eta\left[(r_{12}+r_{34})(z_{32}+z_{41})-(z_{12}+z_{34})(r_{32}+r_{41})\right]$$

This can be expanded too:

$$=\frac{1}{16}\eta\left[r_{12}z_{32}+r_{12}z_{41}+r_{34}z_{32}+r_{34}z_{41}-(z_{12}r_{32}+z_{12}r_{41}+z_{34}r_{32}+z_{34}r_{41})\right]$$

$$=\frac{1}{16}\eta\left[r_{12}z_{32}+r_{12}z_{41}+r_{34}z_{32}+r_{34}z_{41}-z_{12}r_{32}-z_{12}r_{41}-z_{34}r_{32}-z_{34}r_{41}\right]$$

$$=\frac{1}{16}\eta\left[r_{12}z_{32}+r_{12}z_{41}+r_{34}z_{32}+r_{34}z_{41}-r_{32}z_{12}-r_{41}z_{12}-r_{32}z_{34}-r_{41}z_{34}\right]$$

Expanding the r functions and grouping the following is obtained:

$$=\frac{1}{16}\eta\left[2r_{14}z_{32}+2r_{32}z_{41}\right]$$
$$=\frac{1}{8}\eta\left[r_{23}z_{14}-r_{14}z_{23}\right]$$

ξ

$$\frac{1}{16}\left[r_{21}(z_{12}+z_{34})\xi+r_{34}(z_{12}+z_{34})\xi\right]-\frac{1}{16}\left[z_{21}(r_{12}+r_{34})\xi+z_{34}(r_{12}+r_{34})\xi\right]$$
$$=\frac{1}{16}\xi\left[(r_{21}+r_{34})(z_{12}+z_{34})-(z_{21}+z_{34})(r_{12}+r_{34})\right]$$

This can be expanded too:

$$= \frac{1}{16}\xi\left[r_{21}z_{12} + r_{21}z_{34} + r_{34}z_{12} + r_{34}z_{34} - z_{21}r_{12} - z_{21}r_{34} - z_{34}r_{12} - z_{34}r_{34}\right]$$

Expanding the r functions and grouping:

$$= \frac{1}{16}\xi\left[2r_{21}z_{34} + 2z_{12}r_{34}\right]$$

$$= \frac{1}{8}\xi\left[r_{21}z_{34} + z_{12}r_{34}\right]$$

Constants

$$= \frac{1}{16}\left[r_{21}z_{32} + r_{21}z_{41} + r_{34}z_{32} + r_{34}z_{41}\right] - \frac{1}{16}\left[z_{21}r_{32} + z_{21}r_{41} + z_{34}r_{32} + z_{34}r_{41}\right]$$

$$= \frac{1}{16}\left[r_{21}z_{32} + r_{21}z_{41} + r_{34}z_{32} + r_{34}z_{41} - z_{21}r_{32} - z_{21}r_{41} - z_{34}r_{32} - z_{34}r_{41}\right]$$

Expanding the r functions and grouping:

$$= \frac{1}{16}\left[2r_2z_{31} + 2r_1z_{24} + 2r_3z_{42} + 2r_4z_{31}\right]$$

$$= \frac{1}{8}\left[r_{31}z_{42} + r_{24}z_{31}\right]$$

Thus A_0, A_1 and A_2 can be defined such that:

$$\therefore \det[J] = \frac{1}{4}\left(A_0 + A_1\xi + A_2\eta\right)$$

Where the terms are defined by the above calculated terms:

$$A_0 = \frac{1}{2}\left[r_{31}z_{42} + r_{24}z_{31}\right]$$

$$A_1 = \frac{1}{2}\left[r_{21}z_{34} + z_{12}r_{34}\right]$$

$$A_2 = \frac{1}{2}\left[r_{23}z_{14} - r_{14}z_{23}\right]$$

Finally the derivatives with respect to r and z can be obtained using the inverse Jacobian.

C.6 Axis-symmetric TRIA3 element

C.6.1 CST (using analytical integration instead of numerical)

The displacement function is defined as per that of a plain Tria3 element except that x and y are exchanged for r and z [28].

$$\{\mathrm{u}\} = \begin{bmatrix}\mathrm{u(r,z)}\\ \mathrm{w(r,z)}\end{bmatrix} = [\mathrm{N(r,z)}]\{\mathrm{u_e}\}$$

Where the shapes functions for a triangle element are:

$$[N] = \begin{bmatrix}N_1 & 0 & N_2 & 0 & N_3 & 0\\ 0 & N_1 & 0 & N_2 & 0 & N_3\end{bmatrix}$$

$$\{u_e\} = [u_1 \quad w_1 \quad u_2 \quad w_2 \quad u_3 \quad w_3]$$

The strain is given by:

$$\{\varepsilon_e\} = \begin{Bmatrix}\varepsilon_r\\ \varepsilon_\theta\\ \varepsilon_z\\ \gamma_{zr}\end{Bmatrix} = [\partial \mathrm{N(r,z)}]\{u_e\} = [B]\{u_e\}$$

Where

$$[\partial] = \begin{bmatrix}\frac{\partial}{\partial r} & 0\\ \frac{1}{r} & 0\\ 0 & \frac{\partial}{\partial z}\\ \frac{\partial}{\partial z} & \frac{\partial}{\partial r}\end{bmatrix}$$

The [B] matrix is defined by:

$$[B] = \begin{bmatrix}\frac{\partial}{\partial r} & 0\\ \frac{1}{r} & 0\\ 0 & \frac{\partial}{\partial z}\\ \frac{\partial}{\partial z} & \frac{\partial}{\partial r}\end{bmatrix}\begin{bmatrix}N_1 & 0 & N_2 & 0 & N_3 & 0\\ 0 & N_1 & 0 & N_2 & 0 & N_3\end{bmatrix}$$

Thus the stiffness per element is given by:

$$[K_e] = 2\pi \int [B]^T [C]^T [B] r \, dA$$

The element shape functions are based on the linear triangle formulation (Turners triangle). Thus the following is valid:

$$N_i(r,z) = \frac{\alpha_i + \beta_i r + \gamma_i z}{2A_e}$$

Where as per the CST method using numerical integration:

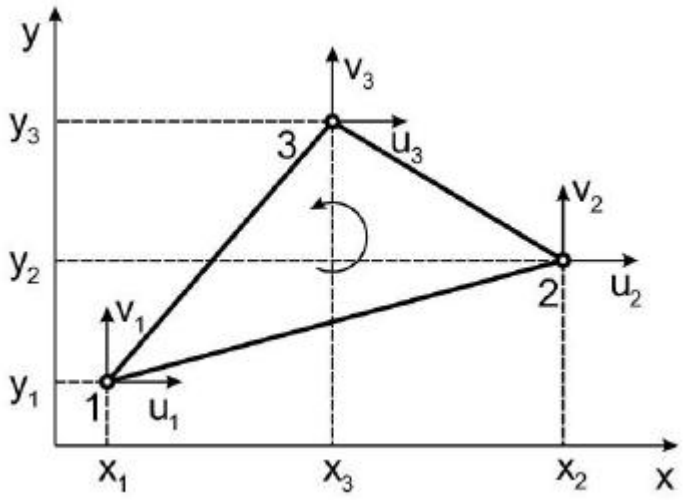


Figure C-13: linear triangle element nodal co-ordinates and displacement [30]

The coefficients below are the same as the section C.4.1 however they have been re-ordered in some cases but the results should remain the same.

$\alpha_1 = r_2 z_3 - r_3 z_2$	$\beta_1 = z_2 - z_3$	$\gamma_1 = r_3 - r_2$
$\alpha_2 = -r_3 z_1 + r_1 z_3$	$\beta_2 = z_3 - z_1$	$\gamma_2 = r_1 - r_3$
$\alpha_3 = r_1 z_2 - r_2 z_1$	$\beta_3 = z_1 - z_2$	$\gamma_3 = r_2 - r_1$

The determinant of [A] is in fact the area of A as proven below:

$$Area = \frac{1}{2}\det([A]) = abs\left(\frac{1}{2}\begin{vmatrix} 1 & r_1 & z_1 \\ 1 & r_2 & z_2 \\ 1 & r_3 & z_3 \end{vmatrix}\right) = \frac{1}{2}(\alpha_1 + \alpha_2 + \alpha_3)$$

The method for obtaining the above alpha, beta and gamma equations are generated using the method where constants are defined such that:

$$u(r_i, z_i) = a_1 + a_2 r_i + a_3 z_i$$

$$v(r_i, z_i) = a_4 + a_5 r_i + a_6 z_i$$

Thus in matrix form;

$$\{\Psi_i\} = \begin{Bmatrix} u_i \\ v_i \end{Bmatrix} = \begin{bmatrix} 1 & r & z & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & r & z \end{bmatrix} \begin{Bmatrix} a_1 \\ a_2 \\ a_3 \\ a_4 \\ a_5 \\ a_6 \end{Bmatrix}$$

The solution for the constant {a} is given by:

$$\begin{Bmatrix} u_1 \\ u_2 \\ u_3 \end{Bmatrix} = \begin{bmatrix} 1 & r_1 & z_1 \\ 1 & r_2 & z_2 \\ 1 & r_3 & z_3 \end{bmatrix} \begin{Bmatrix} a_1 \\ a_2 \\ a_3 \end{Bmatrix} \Longrightarrow \{a\} = [x]^{-1}\{u\}$$

Similarly

$$\begin{Bmatrix} v_1 \\ v_2 \\ v_3 \end{Bmatrix} = \begin{bmatrix} 1 & r_1 & z_1 \\ 1 & r_2 & z_2 \\ 1 & r_3 & z_3 \end{bmatrix} \begin{Bmatrix} a_4 \\ a_5 \\ a_6 \end{Bmatrix} \Longrightarrow \{a\} = [x]^{-1}\{v\}$$

Thus alpha, beta and gamma are introduced in the inversion:

$$[x]^{-1} = \frac{1}{2A} \begin{bmatrix} \alpha_1 & \alpha_2 & \alpha_3 \\ \beta_1 & \beta_2 & \beta_3 \\ \gamma_1 & \gamma_2 & \gamma_3 \end{bmatrix}$$

The solution too {a} is:

$$\begin{Bmatrix} a_1 \\ a_2 \\ a_3 \end{Bmatrix} = \frac{1}{2A} \begin{bmatrix} \alpha_1 & \alpha_2 & \alpha_3 \\ \beta_1 & \beta_2 & \beta_3 \\ \gamma_1 & \gamma_2 & \gamma_3 \end{bmatrix} \begin{Bmatrix} u_1 \\ u_2 \\ u_3 \end{Bmatrix}$$

$$\begin{Bmatrix} a_4 \\ a_5 \\ a_6 \end{Bmatrix} = \frac{1}{2A} \begin{bmatrix} \alpha_1 & \alpha_2 & \alpha_3 \\ \beta_1 & \beta_2 & \beta_3 \\ \gamma_1 & \gamma_2 & \gamma_3 \end{bmatrix} \begin{Bmatrix} v_1 \\ v_2 \\ v_3 \end{Bmatrix}$$

Expanding:

$$\{u(r,z)\} = [1 \quad r \quad z] \begin{Bmatrix} a_1 \\ a_2 \\ a_3 \end{Bmatrix} = [1 \quad r \quad z] \frac{1}{2A} \begin{bmatrix} \alpha_1 & \alpha_2 & \alpha_3 \\ \beta_1 & \beta_2 & \beta_3 \\ \gamma_1 & \gamma_2 & \gamma_3 \end{bmatrix} \begin{Bmatrix} u_1 \\ u_2 \\ u_3 \end{Bmatrix}$$

$$\{v(r,z)\} = [1 \quad r \quad z] \begin{Bmatrix} a_1 \\ a_2 \\ a_3 \end{Bmatrix} = [1 \quad r \quad z] \frac{1}{2A} \begin{bmatrix} \alpha_1 & \alpha_2 & \alpha_3 \\ \beta_1 & \beta_2 & \beta_3 \\ \gamma_1 & \gamma_2 & \gamma_3 \end{bmatrix} \begin{Bmatrix} v_1 \\ v_2 \\ v_3 \end{Bmatrix}$$

If the above function is expanded it is noted that the alpha, beta and gamma constants define the shape functions:

$$\{u(r,z)\} = \frac{1}{2A}\{(\alpha_1 + \beta_1 r + \gamma_1 z)u_1 + (\alpha_2 + \beta_2 r + \gamma_2 z)u_2 + (\alpha_3 + \beta_3 r + \gamma_3 z)u_3\}$$

The same is valid for [v(r,z)]

$$N_1 = \frac{1}{2A}(\alpha_1 + \beta_1 r + \gamma_1 z)$$

$$N_2 = \frac{1}{2A}(\alpha_2 + \beta_2 r + \gamma_2 z)$$

$$N_3 = \frac{1}{2A}(\alpha_3 + \beta_3 r + \gamma_3 z)$$

Thus

$$\{\Psi\} = \begin{Bmatrix} u(r,z) \\ v(r,z) \end{Bmatrix} = [N]\{d\}$$

Where

$$\begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} N_1 & 0 & N_2 & 0 & N_3 & 0 \\ 0 & N_1 & 0 & N_2 & 0 & N_3 \end{bmatrix} \begin{Bmatrix} r_1 \\ z_1 \\ r_2 \\ z_2 \\ r_3 \\ z_3 \end{Bmatrix}$$

Note: r and z are any points inside the element.

Thus if the r and z location is on one of the sides then one of the shape functions will be zero.

$$\{\varepsilon_e\} = \begin{Bmatrix} \varepsilon_r \\ \varepsilon_\theta \\ \varepsilon_z \\ \gamma_{zr} \end{Bmatrix} = \begin{Bmatrix} \frac{\partial u}{\partial r} \\ \frac{u}{r} \\ \frac{\partial v}{\partial z} \\ \frac{\partial u}{\partial z} + \frac{\partial v}{\partial r} \end{Bmatrix} = \begin{Bmatrix} a_2 \\ \frac{a_1}{r} + a_2 + \frac{a_3 z}{r} \\ a_6 \\ a_3 + a_6 \end{Bmatrix}$$

This can be written as:

$$\{\varepsilon_e\} = \begin{Bmatrix} \varepsilon_r \\ \varepsilon_\theta \\ \varepsilon_z \\ \gamma_{zr} \end{Bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ \frac{1}{r} & 1 & \frac{z}{r} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 1 \end{bmatrix} \begin{Bmatrix} a_1 \\ a_2 \\ a_3 \\ a_4 \\ a_5 \\ a_6 \end{Bmatrix}$$

If additional substitution is done then the following is obtained:

$$\{\varepsilon_e\} = \begin{Bmatrix} \varepsilon_r \\ \varepsilon_\theta \\ \varepsilon_z \\ \gamma_{zr} \end{Bmatrix} = \frac{1}{2A} \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ \frac{1}{r} & 1 & \frac{z}{r} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \alpha_1 & \alpha_2 & \alpha_3 & & & \\ \beta_1 & \beta_2 & \beta_3 & & 0 & \\ \gamma_1 & \gamma_2 & \gamma_3 & & & \\ & & & \alpha_1 & \alpha_2 & \alpha_3 \\ & 0 & & \beta_1 & \beta_2 & \beta_3 \\ & & & \gamma_1 & \gamma_2 & \gamma_3 \end{bmatrix} \begin{Bmatrix} r_1 \\ z_1 \\ r_2 \\ z_2 \\ r_3 \\ z_3 \end{Bmatrix}$$

[4x1] = [4x6] [6x6] [6x1]

Similarly for the stiffness matrix the potential (strain) energy of an element is considered; hence the stiffness matrix of each element is given by:

$$[k]_e = \frac{1}{2} \int_v \{\varepsilon\}^T [D] \{\varepsilon\} dV$$

$$[k]_e = \pi \int_A \{\varepsilon_e\}^T [D] \{\varepsilon_e\} dA$$

Where

$$\{\varepsilon_e\} = \begin{Bmatrix} \varepsilon_r \\ \varepsilon_\theta \\ \varepsilon_z \\ \gamma_{zr} \end{Bmatrix} = \frac{1}{2A} \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ \frac{1}{r} & 1 & \frac{z}{r} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \alpha_1 & \alpha_2 & \alpha_3 & & & \\ \beta_1 & \beta_2 & \beta_3 & & 0 & \\ \gamma_1 & \gamma_2 & \gamma_3 & & & \\ & & & \alpha_1 & \alpha_2 & \alpha_3 \\ & 0 & & \beta_1 & \beta_2 & \beta_3 \\ & & & \gamma_1 & \gamma_2 & \gamma_3 \end{bmatrix} \begin{Bmatrix} r_1 \\ z_1 \\ r_2 \\ z_2 \\ r_3 \\ z_3 \end{Bmatrix}$$

Consequently

$$[k]_e = \frac{\pi}{4A^2} \{d\}^T \iint [B]^T [D] [B] \, dr dz \{d\}$$

Where $[B] = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ \frac{1}{r} & 1 & \frac{z}{r} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \alpha_1 & \alpha_2 & \alpha_3 & & & \\ \beta_1 & \beta_2 & \beta_3 & & 0 & \\ \gamma_1 & \gamma_2 & \gamma_3 & & & \\ & & & \alpha_1 & \alpha_2 & \alpha_3 \\ & 0 & & \beta_1 & \beta_2 & \beta_3 \\ & & & \gamma_1 & \gamma_2 & \gamma_3 \end{bmatrix}$

$$[B] = \begin{bmatrix} & \beta_1 & & & \beta_2 & & & \beta_3 & & 0 & 0 & 0 \\ \frac{\alpha_1}{r} + \beta_1 + \frac{z}{r}\gamma_1 & & & \frac{\alpha_2}{r} + \beta_2 + \frac{z}{r}\gamma_2 & & & \frac{\alpha_3}{r} + \beta_3 + \frac{z}{r}\gamma_3 & & 0 & 0 & 0 \\ & 0 & & & 0 & & & 0 & & \gamma_1 & \gamma_2 & \gamma_3 \\ & \gamma_1 & & & \gamma_2 & & & \gamma_3 & & \gamma_1 & \gamma_2 & \gamma_3 \end{bmatrix}$$

So because of row 2 of the first matrix the whole matrix needs to be integrated to get the final solution. It is not possible to just remove the matrices and get the area as the result.

The elements of the strain in the theta orientation are given by:

$$\varepsilon_\theta \iint [B]^T[D][B]\, drdz$$

To define the mass matrix the kinetic energy is used:

$$Ek = \frac{1}{2}\int_v \langle \dot{\varepsilon}(r,z)\rangle \rho \langle \dot{\varepsilon}(r,z)\rangle dV$$

$$Ek = \frac{1}{2}\int_v \langle \ddot{u}(r,z)\rangle \rho \langle \ddot{u}(r,z)\rangle dV$$

Thus for constant Axisymmetric element

$$Ek = \frac{1}{2}\int_v \langle \ddot{u}(r,z)\rangle \rho \langle \ddot{u}(r,z)\rangle d\theta dr dz$$

Using the fact that:

$$\begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} N_1 & 0 & N_2 & 0 & N_3 & 0 \\ 0 & N_1 & 0 & N_2 & 0 & N_3 \end{bmatrix} \begin{Bmatrix} r_1 \\ z_1 \\ r_2 \\ z_2 \\ r_3 \\ z_3 \end{Bmatrix} = [N]\{u_e\}$$

Consequently

$$Ek = \{\ddot{u}_e\}^T 2\pi \int_{r,z} \rho [N]^T[N]\, drdz \{\ddot{u}_e\}$$

Thus the mass matrix per element is defined as:

$$[m]_e = \int_{r,z} \rho [N]^T[N]\, drdz$$

Thus expanding:

$$[m]_e = \pi\rho \iint \begin{bmatrix} N_1N_1 & 0 & N_1N_2 & 0 & N_1N_3 & 0 \\ 0 & N_1N_1 & 0 & N_1N_2 & 0 & N_1N_3 \\ N_2N_1 & 0 & N_2N_2 & 0 & N_2N_3 & 0 \\ 0 & N_2N_1 & 0 & N_2N_2 & 0 & N_2N_3 \\ N_3N_1 & 0 & N_3N_2 & 0 & N_3N_3 & 0 \\ 0 & N_3N_1 & 0 & N_3N_2 & 0 & N_3N_3 \end{bmatrix} dA$$

This can be solved using the fact that, integration over a triangle area is given by:

$$\int {L^k}_i\, {L_j}^m {L_l}^n dA = \frac{k!\, m!\, n!}{(2+k+m+n)!} 2A$$

Thus

$$\int N_1 N_1 \, dA = \int {N_1}^2 \, dA = \frac{2!\,0!\,0!}{(2+2+0+0)!} 2A = \frac{A}{6}$$

And

$$\int N_1 N_2 \, dA = \frac{1!\,1!\,0!}{(2+1+1+0)!} 2A = \frac{1}{12} A$$

Consequently

$$[m]_e = \frac{\pi A \rho}{12} \begin{bmatrix} 2 & 0 & 1 & 0 & 1 & 0 \\ 0 & 2 & 0 & 1 & 0 & 1 \\ 1 & 0 & 2 & 0 & 1 & 0 \\ 0 & 1 & 0 & 2 & 0 & 1 \\ 1 & 0 & 2 & 0 & 2 & 0 \\ 0 & 1 & 0 & 1 & 0 & 2 \end{bmatrix}$$

C.6.2 Linear Triangle using Isoparametric Method

The displacement function is defined as per that of a plain Tria3 element except that x and y are exchanged for r and z.

$$\{\mathrm{u}\} = \begin{bmatrix} \mathrm{u(r,z)} \\ \mathrm{w(r,z)} \end{bmatrix} = [\mathrm{N(r,z)}]\{\mathrm{u_e}\}$$

Where the shapes functions for a triangle element are:

$$[N] = \begin{bmatrix} N_1 & 0 & N_2 & 0 & N_3 & 0 \\ 0 & N_1 & 0 & N_2 & 0 & N_3 \end{bmatrix}$$

$$\{u_e\} = [u_1 \quad w_1 \quad u_2 \quad w_2 \quad u_3 \quad w_3]$$

The strain is given by:

$$\{\varepsilon_e\} = \begin{Bmatrix} \varepsilon_r \\ \varepsilon_\theta \\ \varepsilon_z \\ \gamma_{zr} \end{Bmatrix} = [\partial \mathrm{N(r,z)}]\{u_e\} = [B]\{u_e\}$$

Where

$$[\partial] = \begin{bmatrix} \dfrac{\partial}{\partial r} & 0 \\ \dfrac{1}{r} & 0 \\ 0 & \dfrac{\partial}{\partial z} \\ \dfrac{\partial}{\partial z} & \dfrac{\partial}{\partial r} \end{bmatrix}$$

The [B] matrix is defined by:

$$[B] = \begin{bmatrix} \frac{\partial}{\partial r} & 0 \\ \frac{1}{r} & 0 \\ 0 & \frac{\partial}{\partial z} \\ \frac{\partial}{\partial z} & \frac{\partial}{\partial r} \end{bmatrix} \begin{bmatrix} N_1 & 0 & N_2 & 0 & N_3 & 0 \\ 0 & N_1 & 0 & N_2 & 0 & N_3 \end{bmatrix}$$

Thus the stiffness per element is given by:

$$[K_e] = 2\pi \int [B]^T [C]^T [B] r \, dA$$

However if Isoparametric triangular elements are used then the shape functions are given by:

$$N_1 = 1 - \xi - \eta$$

$$N_2 = \xi$$

$$N_3 = \eta$$

Thus since the [B] matrix is defined in terms of r and z, it is required to change the variables using the Jacobian. This unfortunately cannot be done as for the 2D case by just multiplying the [B] by the Jacobian due to the presence of row 2.

$$[B] = \begin{bmatrix} \frac{\partial N_1}{\partial r} & 0 & \frac{\partial N_2}{\partial r} & 0 & \frac{\partial N_3}{\partial r} & 0 \\ \frac{N_1}{r} & 0 & \frac{N_2}{r} & 0 & \frac{N_3}{r} & 0 \\ 0 & \frac{\partial N_1}{\partial z} & 0 & \frac{\partial N_2}{\partial z} & 0 & \frac{\partial N_3}{\partial z} \\ \frac{\partial N_1}{\partial z} & \frac{\partial N_1}{\partial r} & \frac{\partial N_2}{\partial z} & \frac{\partial N_2}{\partial r} & \frac{\partial N_3}{\partial z} & \frac{\partial N_3}{\partial r} \end{bmatrix}$$

Thus the following needs to be noted and implemented for parametric elements (ξ,η):

$$r(\xi, \eta) = N_1(\xi, \eta) r_1 + N_2(\xi, \eta) r_2 + N_3(\xi, \eta) r_3$$

$$z(\xi, \eta) = N_1(\xi, \eta) z_1 + N_2(\xi, \eta) z_2 + N_3(\xi, \eta) z_3$$

Thus the above equation are converted to functions of (ξ,η) using the Jacobian (i.e. u=Ø)

$$\begin{Bmatrix} \frac{\partial \varphi}{\partial \xi} \\ \frac{\partial \varphi}{\partial \eta} \end{Bmatrix} = [J] \begin{Bmatrix} \frac{\partial \varphi}{\partial r} \\ \frac{\partial \varphi}{\partial z} \end{Bmatrix} = \begin{bmatrix} \frac{\partial r}{\partial \xi} & \frac{\partial z}{\partial \xi} \\ \frac{\partial r}{\partial \eta} & \frac{\partial z}{\partial \eta} \end{bmatrix} \begin{Bmatrix} \frac{\partial \varphi}{\partial r} \\ \frac{\partial \varphi}{\partial z} \end{Bmatrix}$$

The scalar field can be any field such as x and y.

Where

$$[J] = \begin{bmatrix} \frac{\partial r}{\partial \xi} & \frac{\partial z}{\partial \xi} \\ \frac{\partial r}{\partial \eta} & \frac{\partial z}{\partial \eta} \end{bmatrix} = \begin{bmatrix} \sum_{i=1}^{3} \frac{\partial}{\partial \xi} N_i(\xi,\eta).r_i & \sum_{i=1}^{3} \frac{\partial}{\partial \xi} N_i(\xi,\eta).z_i \\ \sum_{i=1}^{3} \frac{\partial}{\partial \eta} N_i(\xi,\eta).r_i & \sum_{i=1}^{3} \frac{\partial}{\partial \eta} N_i(\xi,\eta).z_i \end{bmatrix} \quad \textbf{122}$$

Thus using the shape functions above:

$$[J] = \begin{bmatrix} \frac{\partial r}{\partial \xi} & \frac{\partial z}{\partial \xi} \\ \frac{\partial r}{\partial \eta} & \frac{\partial z}{\partial \eta} \end{bmatrix} = \begin{bmatrix} -r_1 + r_2 & -z_1 + z_2 \\ -r_1 + r_3 & -z_1 + z_3 \end{bmatrix} \quad \textbf{123}$$

This can be visualized as:

$$[J] = \begin{bmatrix} J_{11} & J_{12} \\ J_{21} & J_{22} \end{bmatrix} = \begin{bmatrix} \frac{\partial r}{\partial \xi} & \frac{\partial z}{\partial \xi} \\ \frac{\partial r}{\partial \eta} & \frac{\partial z}{\partial \eta} \end{bmatrix} = \begin{bmatrix} \frac{\partial}{\partial \xi} N_1(\xi,\eta) & \frac{\partial}{\partial \xi} N_2(\xi,\eta) & \frac{\partial}{\partial \xi} N_3(\xi,\eta) \\ \frac{\partial}{\partial \eta} N_2(\xi,\eta) & \frac{\partial}{\partial \eta} N_2(\xi,\eta) & \frac{\partial}{\partial \eta} N_3(\xi,\eta) \end{bmatrix} \begin{bmatrix} r_1 & z_1 \\ r_2 & z_2 \\ r_3 & z_3 \end{bmatrix} \quad \textbf{124}$$

[2x2] [2x3] [3x2]

The inverse of the Jacobian is given by:

$$[J]^{-1} = \frac{1}{det([J])} \begin{bmatrix} \frac{\partial z}{\partial \eta} & -\frac{\partial z}{\partial \xi} \\ -\frac{\partial r}{\partial \eta} & \frac{\partial r}{\partial \xi} \end{bmatrix} = \frac{1}{det([J])} \begin{bmatrix} J_{22} & -J_{12} \\ -J_{21} & J_{11} \end{bmatrix} \quad \textbf{125}$$

This inverse will be sufficient for, u,w N_1 etc . As an example for the shape function N_1:

$$\begin{Bmatrix} \frac{\partial N_1}{\partial r} \\ \frac{\partial N_1}{\partial z} \end{Bmatrix} = [J]^{-1} \begin{Bmatrix} \frac{\partial N_1}{\partial \xi} \\ \frac{\partial N_1}{\partial \eta} \end{Bmatrix}$$

Thus [B] can be re-written as:

$$[B] = \begin{bmatrix} \frac{\partial N_1}{\partial r} & 0 & \frac{\partial N_2}{\partial r} & 0 & \frac{\partial N_3}{\partial r} & 0 \\ \frac{N_1}{r} & 0 & \frac{N_2}{r} & 0 & \frac{N_3}{r} & 0 \\ 0 & \frac{\partial N_1}{\partial z} & 0 & \frac{\partial N_2}{\partial z} & 0 & \frac{\partial N_3}{\partial z} \\ \frac{\partial N_1}{\partial z} & \frac{\partial N_1}{\partial r} & \frac{\partial N_2}{\partial z} & \frac{\partial N_2}{\partial r} & \frac{\partial N_3}{\partial z} & \frac{\partial N_3}{\partial r} \end{bmatrix}$$

$$[B]=[C]\begin{bmatrix}\left(J_{22}\frac{\partial N_1}{\partial\xi}-J_{12}\frac{\partial N_1}{\partial\eta}\right) & 0 & \left(J_{22}\frac{\partial N_2}{\partial\xi}-J_{12}\frac{\partial N_2}{\partial\eta}\right) & 0 & \left(J_{22}\frac{\partial N_3}{\partial\xi}-J_{12}\frac{\partial N_3}{\partial\eta}\right) & 0\\ \frac{N_1}{r} & 0 & \frac{N_2}{r} & 0 & \frac{N_3}{r} & 0\\ 0 & \left(-J_{21}\frac{\partial N_1}{\partial\xi}+J_{11}\frac{\partial N_1}{\partial\eta}\right) & 0 & \left(-J_{21}\frac{\partial N_2}{\partial\xi}+J_{11}\frac{\partial N_2}{\partial\eta}\right) & 0 & \left(-J_{21}\frac{\partial N_3}{\partial\xi}+J_{11}\frac{\partial N_3}{\partial\eta}\right)\\ \left(-J_{21}\frac{\partial N_1}{\partial\xi}+J_{11}\frac{\partial N_1}{\partial\eta}\right) & \left(J_{22}\frac{\partial N_1}{\partial\xi}-J_{12}\frac{\partial N_1}{\partial\eta}\right) & \left(-J_{21}\frac{\partial N_2}{\partial\xi}+J_{11}\frac{\partial N_2}{\partial\eta}\right) & \left(J_{22}\frac{\partial N_2}{\partial\xi}-J_{12}\frac{\partial N_2}{\partial\eta}\right) & \left(-J_{21}\frac{\partial N_3}{\partial\xi}+J_{11}\frac{\partial N_3}{\partial\eta}\right) & \left(J_{22}\frac{\partial N_3}{\partial\xi}-J_{12}\frac{\partial N_3}{\partial\eta}\right)\end{bmatrix}$$

Where [C] is a coefficient matrix to help in fitting the matrix on one page:

$$[C]=\begin{bmatrix}\frac{1}{det([J])} & 0 & 0 & 0\\ 0 & 1 & 0 & 0\\ 0 & 0 & \frac{1}{det([J])} & 0\\ 0 & 0 & 0 & \frac{1}{det([J])}\end{bmatrix}$$

Thus [B] is now a function of (ξ,η) and hence can be integrated where:

$$\frac{\partial}{\partial\xi}N_1(\xi,\eta)=-1$$

$$\frac{\partial}{\partial\eta}N_1(\xi,\eta)=-1$$

$$\frac{\partial}{\partial\xi}N_2(\xi,\eta)=1$$

$$\frac{\partial}{\partial\eta}N_2(\xi,\eta)=0$$

$$\frac{\partial}{\partial\xi}N_3(\xi,\eta)=0$$

$$\frac{\partial}{\partial\eta}N_3(\xi,\eta)=1$$

Thus the [B] matrix is simplified too:

$$[B]=[C]\begin{bmatrix}(-J_{22}+J_{12}) & 0 & J_{22} & 0 & -J_{12} & 0\\ \frac{N_1}{r} & 0 & \frac{N_2}{r} & 0 & \frac{N_3}{r} & 0\\ 0 & (J_{21}-J_{11}) & 0 & -J_{21} & 0 & J_{11}\\ (J_{21}-J_{11}) & (-J_{22}+J_{12}) & -J_{21} & J_{22} & J_{11} & -J_{12}\end{bmatrix}$$

Thus the stiffness matrix is calculated based on the potential (strain) energy of the structure:

$$K=\frac{1}{2}\int_v\langle\varepsilon(x,y)\rangle[D]\{\varepsilon(x,y)\}\,dV$$

For constant radial elements (that is for a closed cylinder)

$$K=\pi\iint_A\langle\varepsilon(r,z)\rangle[D]\{\varepsilon(r,z)\}r\,drdz$$

However as indicated previously

$$\langle \varepsilon(r,z) \rangle = [B]\{u_e\}$$

Thus to convert to parametric element the determinant is used which is assumed to be the physical incremental area for the element. In the case of triangle elements the determinant is 2A thus the determinant is reduced by half.

$$K = \{u_e\}\pi \underbrace{\int_{\eta=-1}^{\eta=1}\int_{\xi=-1}^{\xi=1}[B]^t[D][B]\,[detJ]rd\xi d\eta}_{\textbf{\textit{Stiffness Matrix}}}\{u_e\}$$

This can be numerically integrated using numerical integration. The formula to be applied is:

$$\int_A \varphi(r,z)\,dA \approx \sum_{i=1}^{n} \varphi_i(\xi,\eta)\,J_i W_i$$

Where

n - Is the number of sampling points,

$J_i = \frac{1}{2}|[J(\xi,\eta)]|$ (0.5 is because the area of the triangle is reference co-ordinates is 0.5)

W – Is the weighting factors given in the following table as taken from Cook page 267 [28].

TABLE 7.4-1 SELECTED FORMULAS FOR NUMERICAL INTEGRATION OVER A TRIANGULAR AREA, EQ. 7.4-1 [6.4]

No. of points	Figure	Degree of precision	Coordinates (r_i, s_i)	Weights W_i
1	7.4-1a	1	$\left(\frac{1}{3},\frac{1}{3}\right)$	1.0
3	7.4-1b	2	$\left(\frac{2}{3},\frac{1}{6}\right),\left(\frac{1}{6},\frac{1}{6}\right),\left(\frac{1}{6},\frac{2}{3}\right)$	$\frac{1}{3}$
3	7.4-1c	2	$\left(\frac{1}{2},0\right),\left(0,\frac{1}{2}\right),\left(\frac{1}{2},\frac{1}{2}\right)$	$\frac{1}{3}$
4	7.4-1d	3	$\left(\frac{1}{3},\frac{1}{3}\right)$ $\left(\frac{3}{5},\frac{1}{5}\right),\left(\frac{1}{5},\frac{1}{5}\right),\left(\frac{1}{5},\frac{3}{5}\right)$	$-\frac{27}{48}$ $\frac{25}{48}$

Figure C-14: Numerical integration factors for triangle elements [28]

C.7 FEM Verification

To verify that the Matlab code created is working correctly a verification analysis was performed on one, four and an eight element model. This was in addition to the cylindrical model checks done in section 4.1.2 and each horn and specimen check.

C.7.1 Four Element Model

The four element mode was created as indicated in Figure C-15

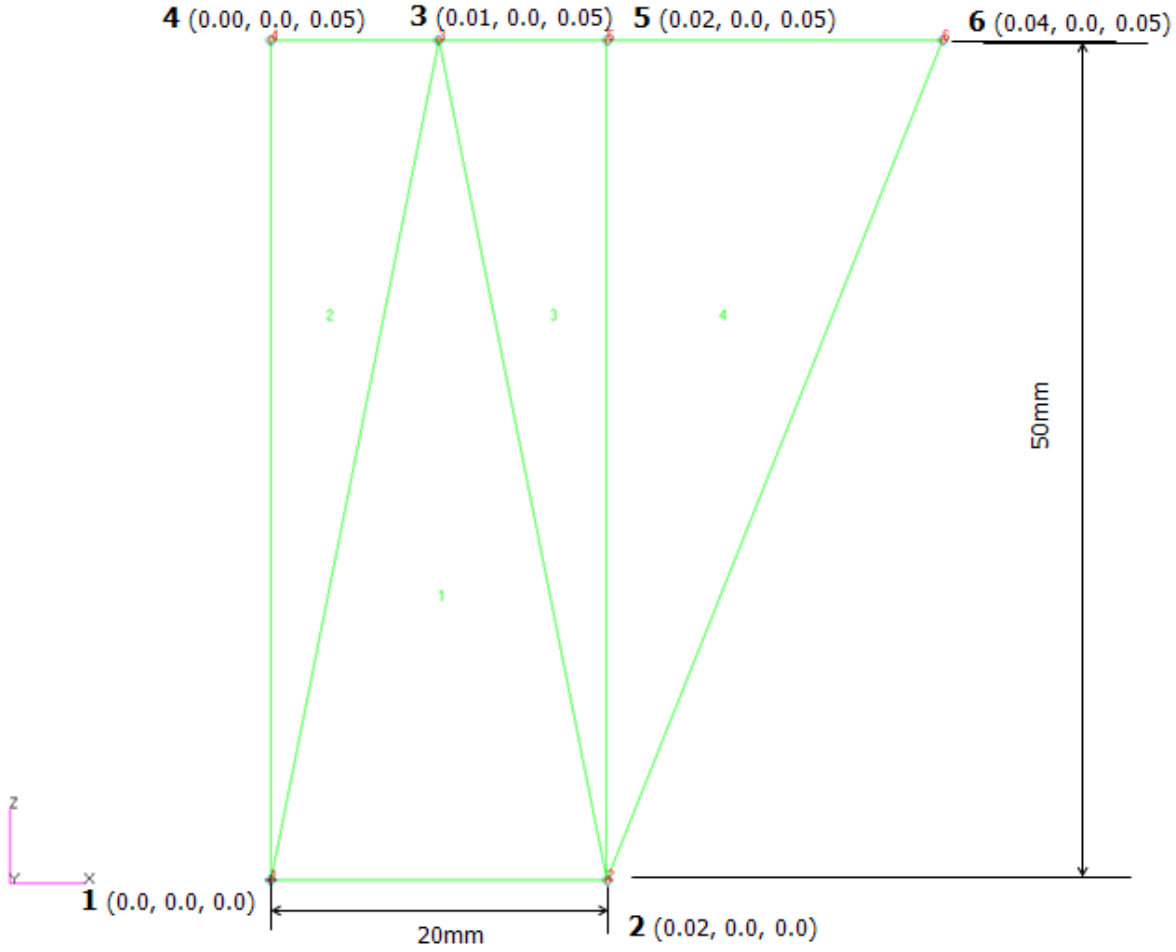


Figure C-15: Four element verification model

Two mass matrices were considered namely the lumped and coupled mass matrix. The following tables indicate the correspondence of the results for the different integration methods as compared to the results obtained from Nastran.

Table C-1: Comparison of four element model using lumped mass matrix

Solver	Nastran	Matlab		
Integration method	**Standard**	**Full**	**Reduced**	**Area**
Mode # 1	0	0	0	0
Mode # 2	28781	28635	19962	6780
Mode # 3	39179	38637	32336	8640
Mode # 4	52862	52637	48235	11060
Mode # 5	55079	54990	52121	15870
Mode # 6	72380	72289	71426	21110
Mode # 7	81063	81003	78806	26630
Mode # 8	108749	108242	81401	31320
Mode # 9	112671	111860	102129	56830
Mode # 10	155565	150483	117345	64540
Mode # 11	188655	177362	160715	127800
Mode # 12	411039	336552	182711	135000

Similarly for the coupled mass matrix the following results are obtained:

Table C-2: Comparison of four element model using coupled mass matrix

Solver	Nastran	Matlab		
Integration method	**Standard**	**Full**	**Reduced**	**Area**
Mode # 1	0	0	0	0
Mode # 2	41851	41596	39686	7090
Mode # 3	59879	59647	63268	10500
Mode # 4	77747	77015	100483	14220
Mode # 5	94304	94193	156296	22530
Mode # 6	110793	110802	156707	34790
Mode # 7	144584	145754	229610	47300
Mode # 8	183123	183852	305825	60180
Mode # 9	197786	198767	-	98660
Mode # 10	277177	268165	-	117170
Mode # 11	338807	323303	-	205730
Mode # 12	662729	553858	-	223940

The results indicate that the full integration method corresponds well with the Nastran results. The reduced and area methods show different levels of confidence when compared to Nastran. The lumped mass matrix indicated that for this configuration the reduced integration method is sufficient to obtain data of the same order of magnitude. However for the coupled matrix the

reduced integration method is not as effective and the method also had some modes that did not reflect after integration. Overall the four element model indicates that the Full integration method provides the best data and hence this method will be used for all indicated results. It should be noted that two sets of coefficients were identified for the full integration namely that of Cook [28] and the other was from Rice University: *http://www.mems.rice.edu/~akin/Elsevier/Chap_10.pdf (2015/03/16.)*

C.7.2 Eight Element Model

The eight element mode was created as indicated in Figure C-16.

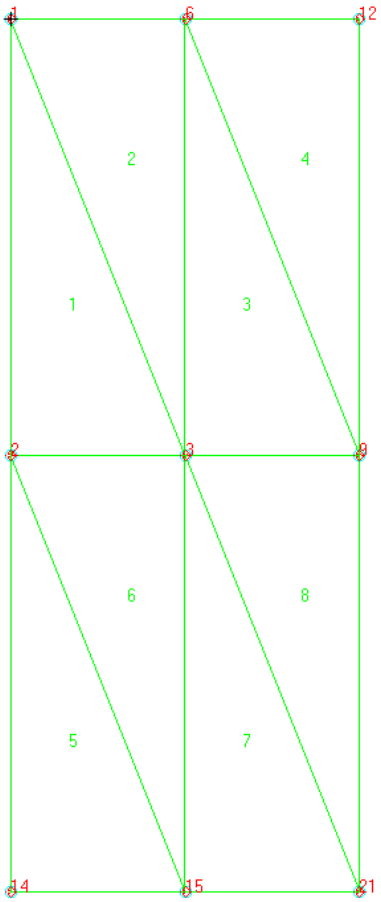


Figure C-16: Eight element verification model

Comparisons were done directly with Nastran using the Matlab code indicated in Appendix G.5.17, whereby the mass and stiffness matrix of Nastran were output and compared directly to the Matlab obtained matrices.

Two mass matrices were considered namely the lumped and coupled mass matrix. The following tables indicate the correspondence of the results for the different integration methods as compared to the results from Nastran.

Table C-3: Comparison of eight element model using lumped mass matrix

Solver	Nastran	Matlab		
Integration method	Standard	Full	Reduced	Area
Mode # 1	0	0	0	0
Mode # 2	22 211	22 208	21 518	0
Mode # 3	30 247	30 070	22 653	3 558
Mode # 4	33 286	32 640	28 090	8 490
Mode # 5	39 494	39 465	37 769	11 173
Mode # 6	45 813	45 780	44 508	15 016
Mode # 7	50 745	50 649	46 572	18 476
Mode # 8	53 863	53 723	48 219	25 076
Mode # 9	54 275	54 190	53 785	27 007
Mode # 10	57 000	56 729	54 200	27 623
Mode # 11	59 399	59 308	56 685	38 096
Mode # 12	65 597	65 278	58 305	46 715
Mode # 13	94 519	93 568	73 862	49 589
Mode # 14	98 474	97 905	83 124	54 668
Mode # 15	110 688	110 073	94 584	56 283
Mode # 16	136 699	128 027	100 478	61 576
Mode # 17	164 183	141 626	105 181	75 389
Mode # 18	213 149	172 943	112 564	316 034

Similarly for the coupled mass matrix the following results are obtained:

Table C-4: Comparison of eight element model using coupled mass matrix

Solver	Nastran	Matlab		
Integration method	Standard	Full	Reduced	Area
Mode # 1	0	0	0	0
Mode # 2	26 860	26 860	28 877	0
Mode # 3	41 449	41 100	40 829	3 846
Mode # 4	52 087	51 680	52 320	11 694
Mode # 5	55 200	54 962	55 554	16 516
Mode # 6	58 125	57 850	67 661	20 208
Mode # 7	70 536	70 330	86 726	27 712
Mode # 8	79 016	78 663	111 310	35 315
Mode # 9	98 312	98 099	122 306	43 034
Mode # 10	104 514	104 519	128 849	44 652
Mode # 11	114 205	114 515	143 773	51 025
Mode # 12	121 550	124 711	163 250	72 058
Mode # 13	147 431	146 061	209 122	81 246
Mode # 14	165 630	162 811	297 584	83 649
Mode # 15	193 961	191 327	-	97 929
Mode # 16	254 105	238 959	-	102 737
Mode # 17	287 123	253 563	-	136 264
Mode # 18	339 751	290 041	-	523 437

APPENDIX D - INITIAL POWER SUPPLY

The in initial power supply was modified from a typical ATX PC power supply to a test power supply using online information [31]. The ATX power supply is capable of supplying +12V, -12V, +5V, -5V and 3.3V directly or with combinations between the voltage supplies other summation values are possible i.e. connecting +12V to -12V can produce a 24V output if required. The use of the ATX power supply was necessary to complete the project before the funding was obtained for AISI as a typical laboratory test power supply can cost approximately in excess of a couple of thousand Rand, however this simple setup cost less than R300 in total. The ATX power supply and typical manufacturing steps are indicated in the images below:

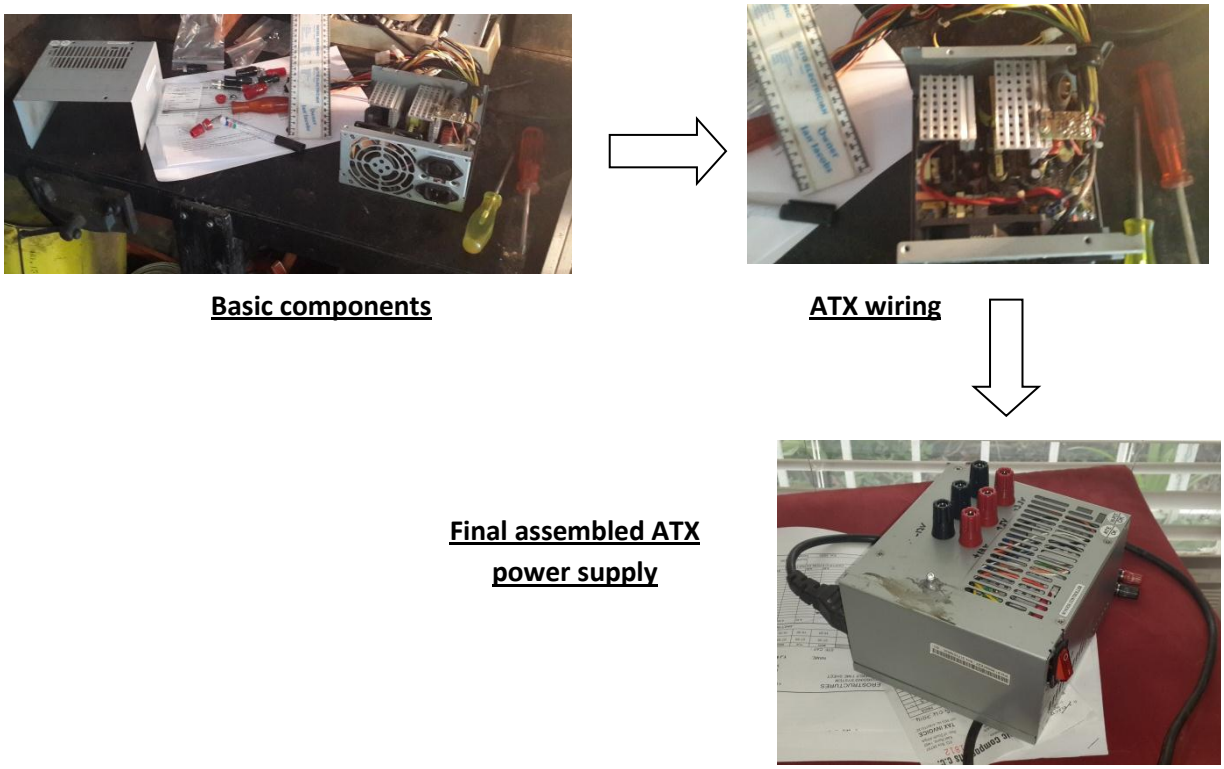


Figure D-1: ATX power supply for strain gauge measurement system

The ATX power was used in-conjunction with the Op-amp system from Denel Dynamics and the resulting amplification factor was gauged via Matlab and found to be 300. This factor is used when determining the final strain measurement from the strain gauge if direct results are implemented as per equations of section B.2.

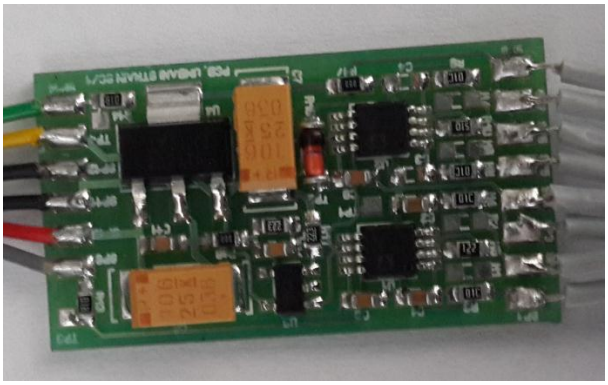


Figure D-2: Strain gauge signal conditioning system

The op-amp circuitry obtained from Denel Dynamics initially consisted of a RC low-pass filter. However this filter would filter out all high frequency signals, which are required in this application since the input signal here is 20 kHz. Thus the two capacitors of the RC filter were removed. This was checked using Matlab's electronics toolbox to verify that the circuitry is now applicable to this current design. To do this an embedded theoretical sinusoidal signal was create of two sign waves, one at a high frequency of 20 kHz and the other at 20 Hz. The separate sinusodial signals are inidcated in Figure D-3 with the combined signal being indicated in Figure D-4.

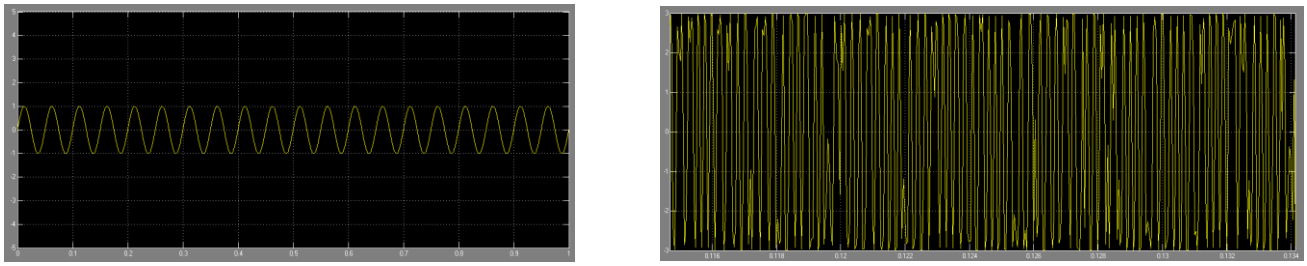


Figure D-3: Two input signals

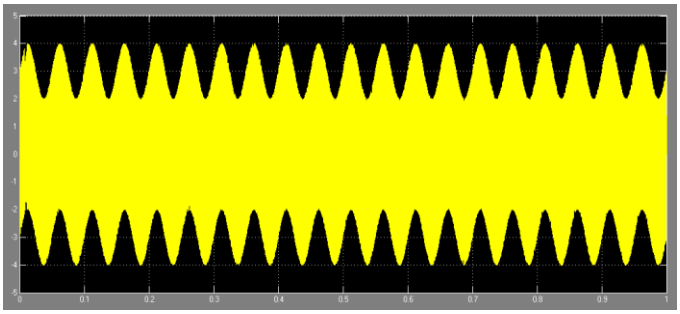


Figure D-4: Combined input signal

The circuits were then symbolically created in Matlab using the electronics tool box by means of resistors and capacitors. The signal was then passed through the two systems and the output monitored. This is indicated in Figure D-5 where the circuit on top is the one with both capacitors removed and the lower circuit is the original low pass filter circuit with the capacitors intact.

The figures clearly indicted the effect of the low pass filter on the input signal, with the first circuit having no effect on the overall signal as used in the Mark 1 design. Thus if an op-amp is inserted in

the first circuit an amplification of the entire signal will be created with no filtering of the higher frequency data.

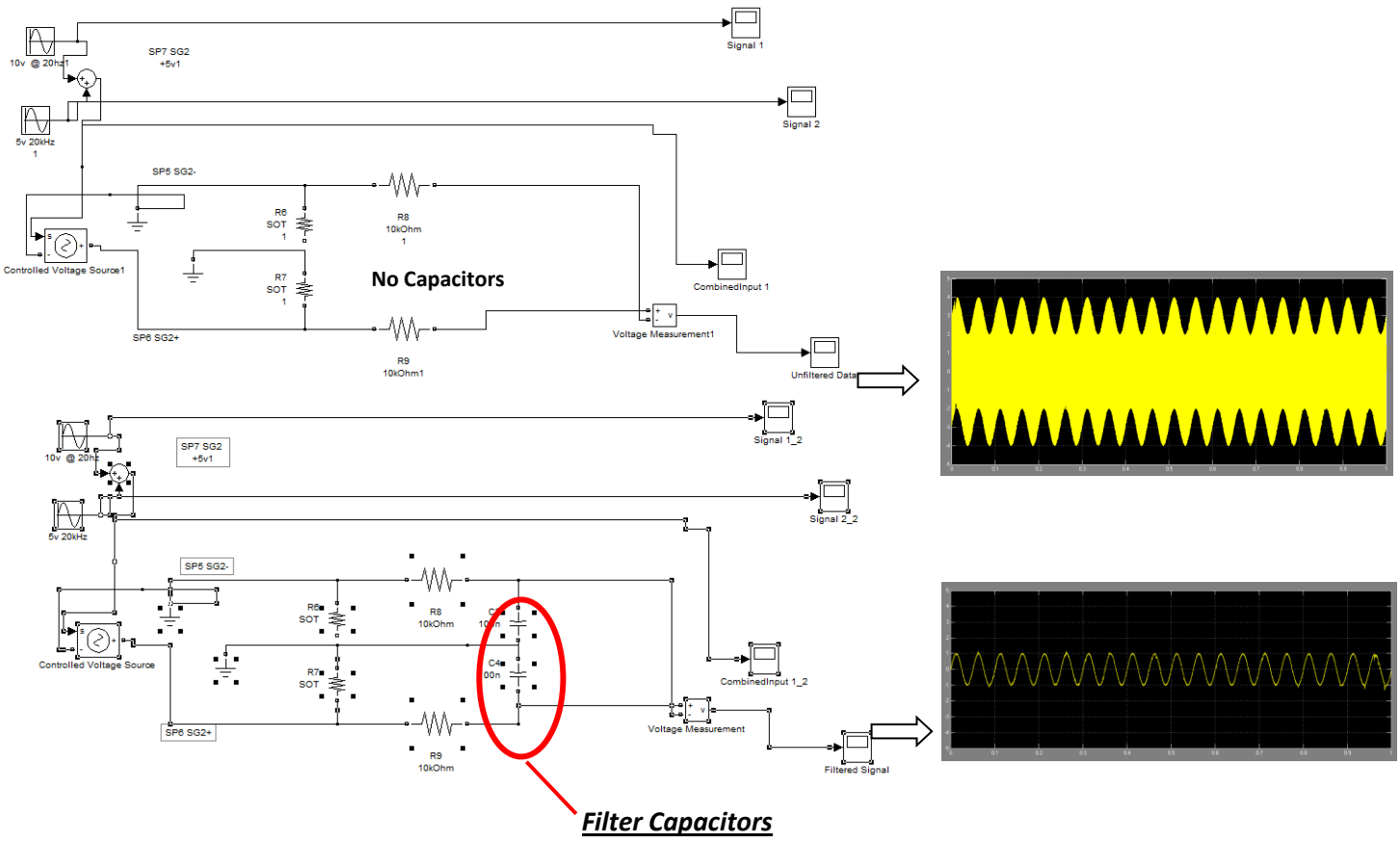


Figure D-5: Test of Circuitry

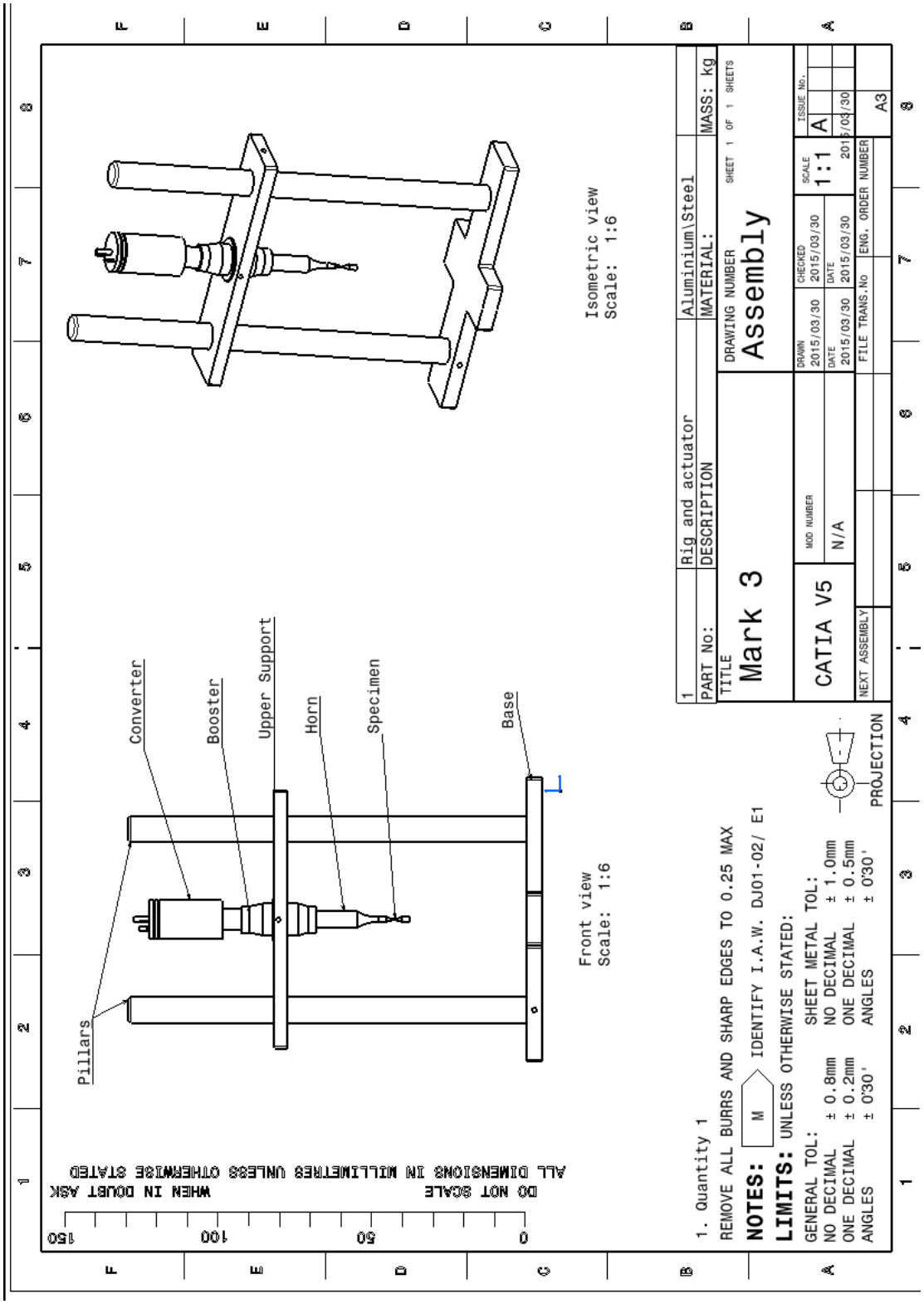
If the second circuit was implemented all the high frequency data would have been filtered which would not be valid for the strain gauge data required.

–

APPENDIX E DESIGN DRAWINGS

This appendix contains all the released drawings for the design of the UHC fatigue testing machine.

E.1 Full Mark 3



The full Mark 3 system is indicated in the following schematic exploded view which indicates all the manufactured and purchased components together with the final measurement system:

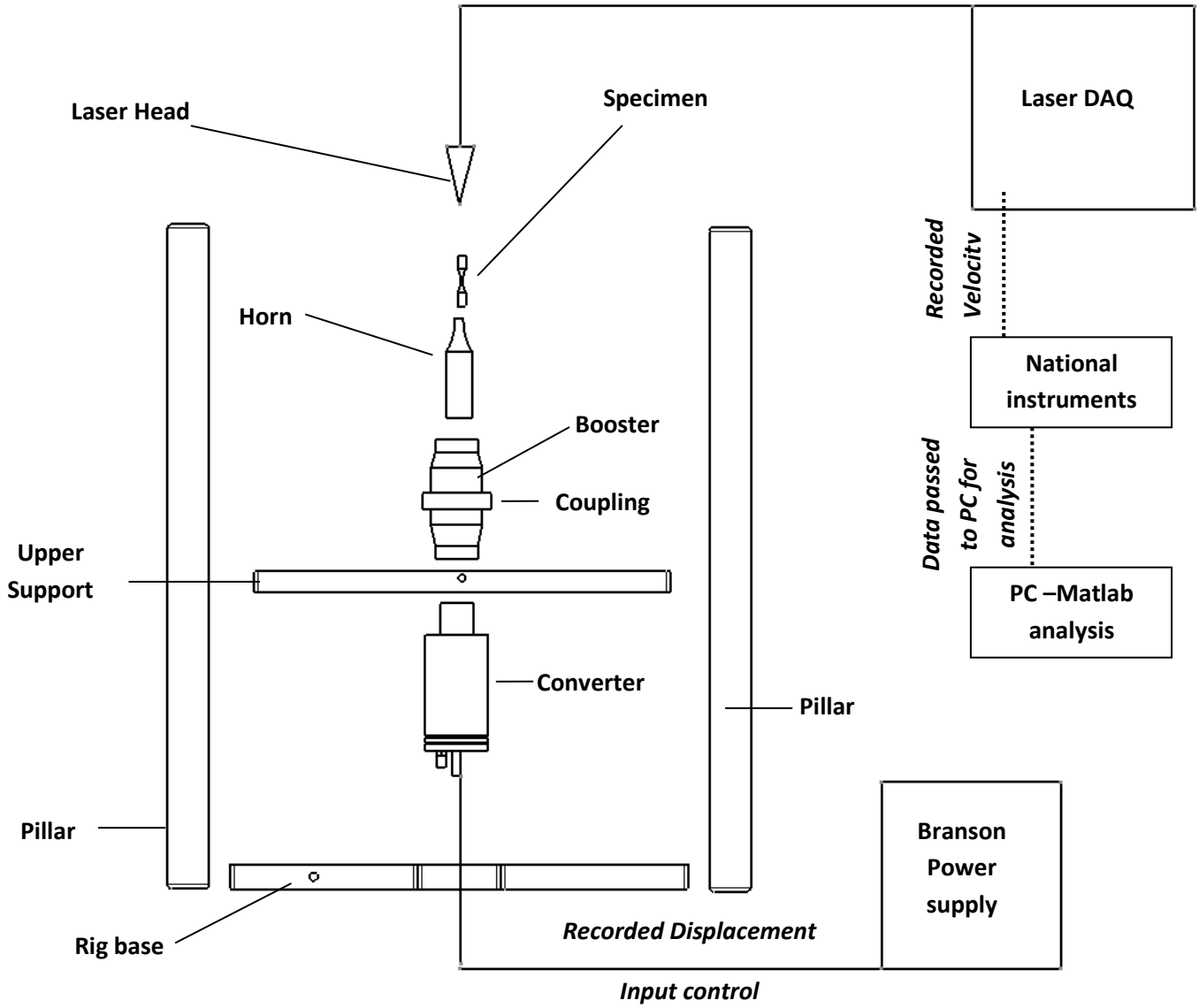
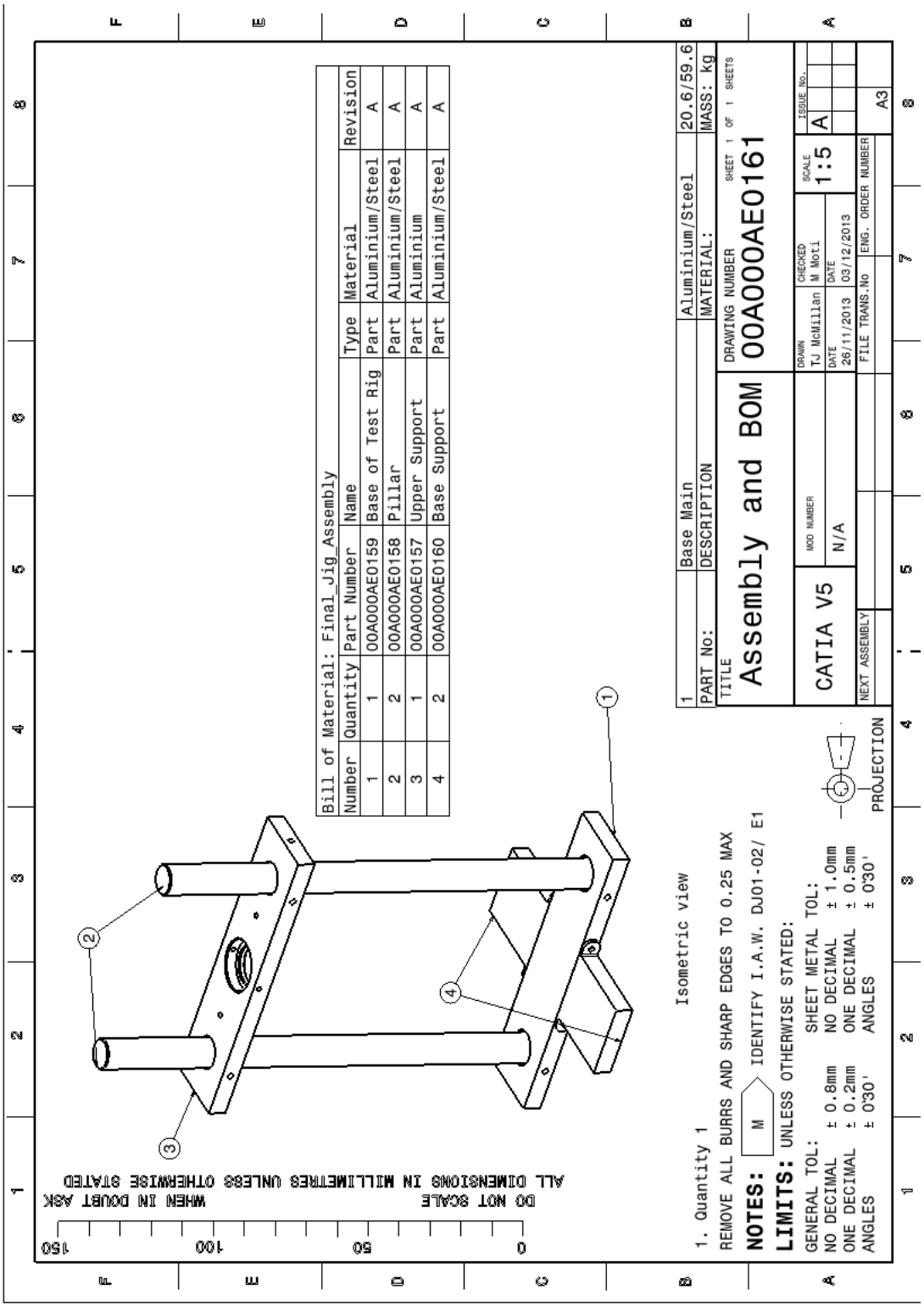
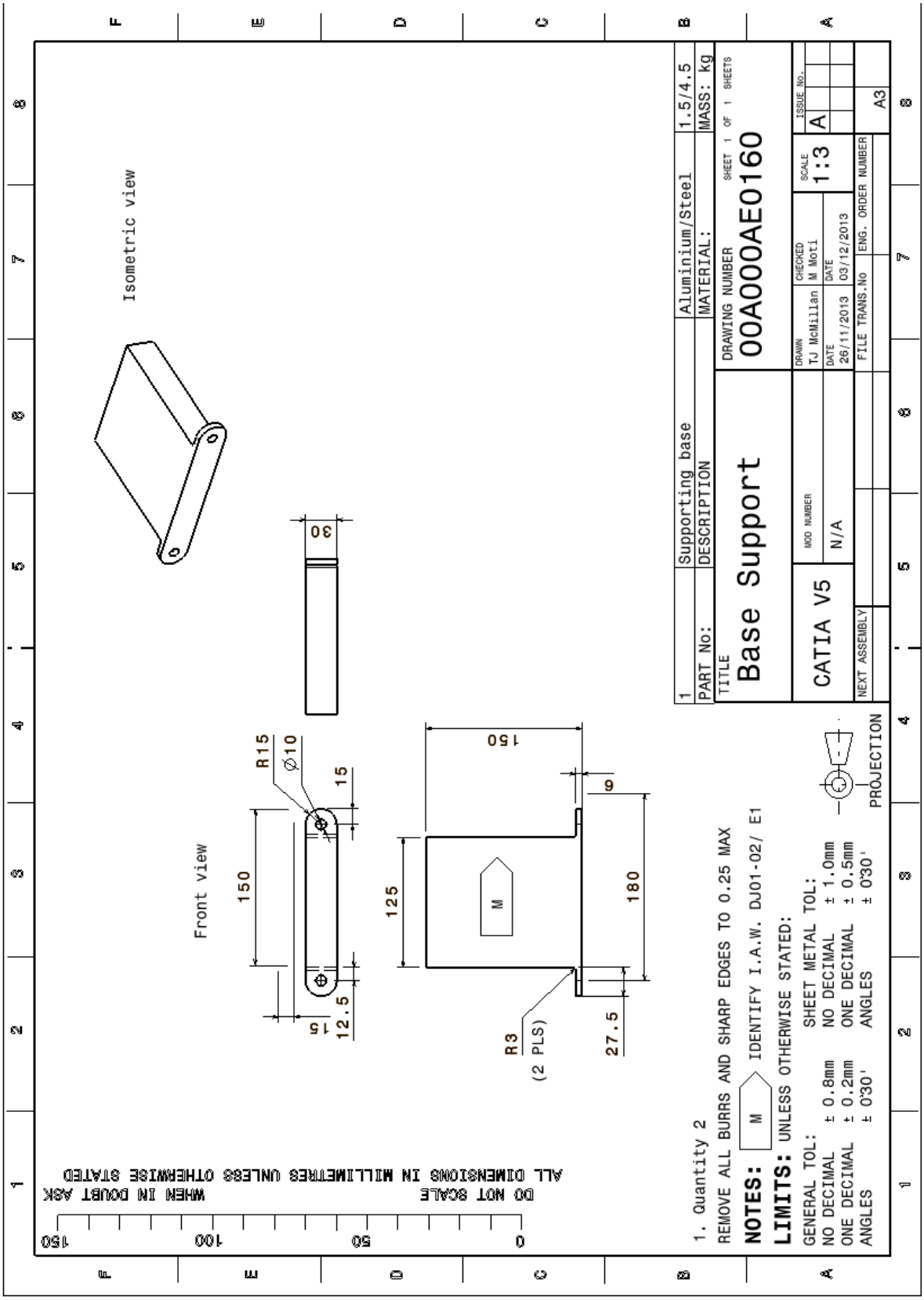


Figure E-1: Schematic exploded view of full Mark 3 system

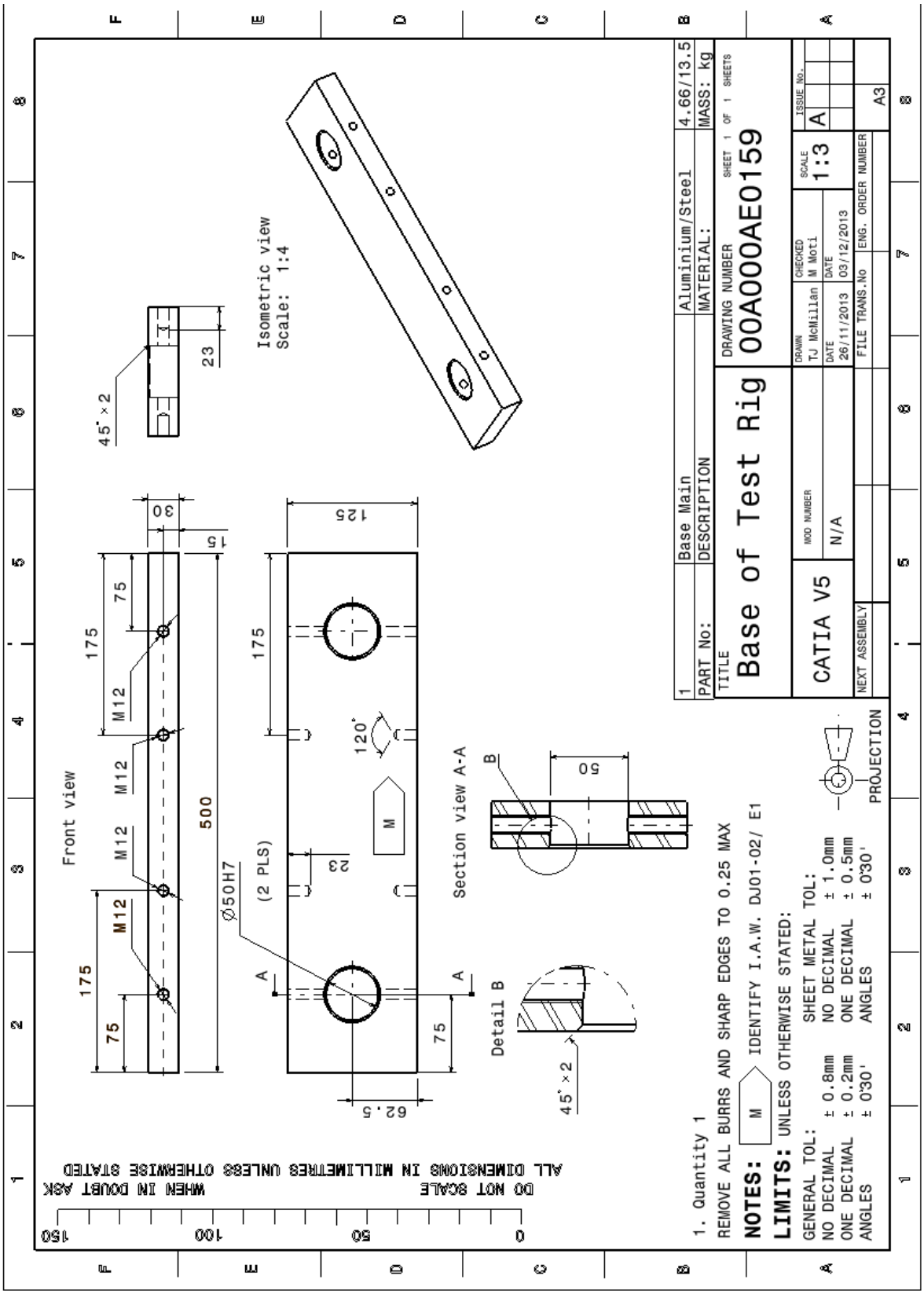
E.2 Support Assembly



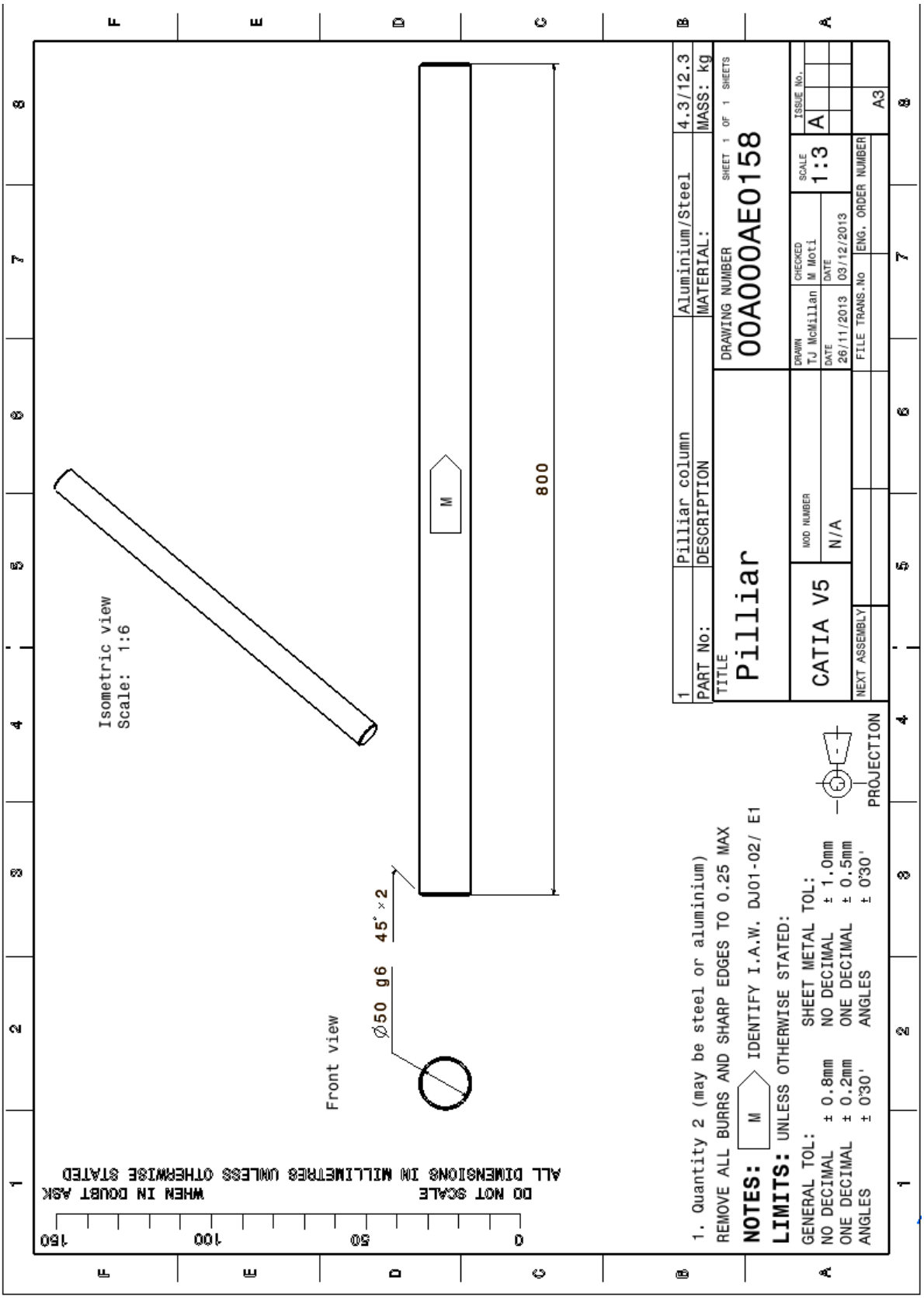
E.3 Base Support



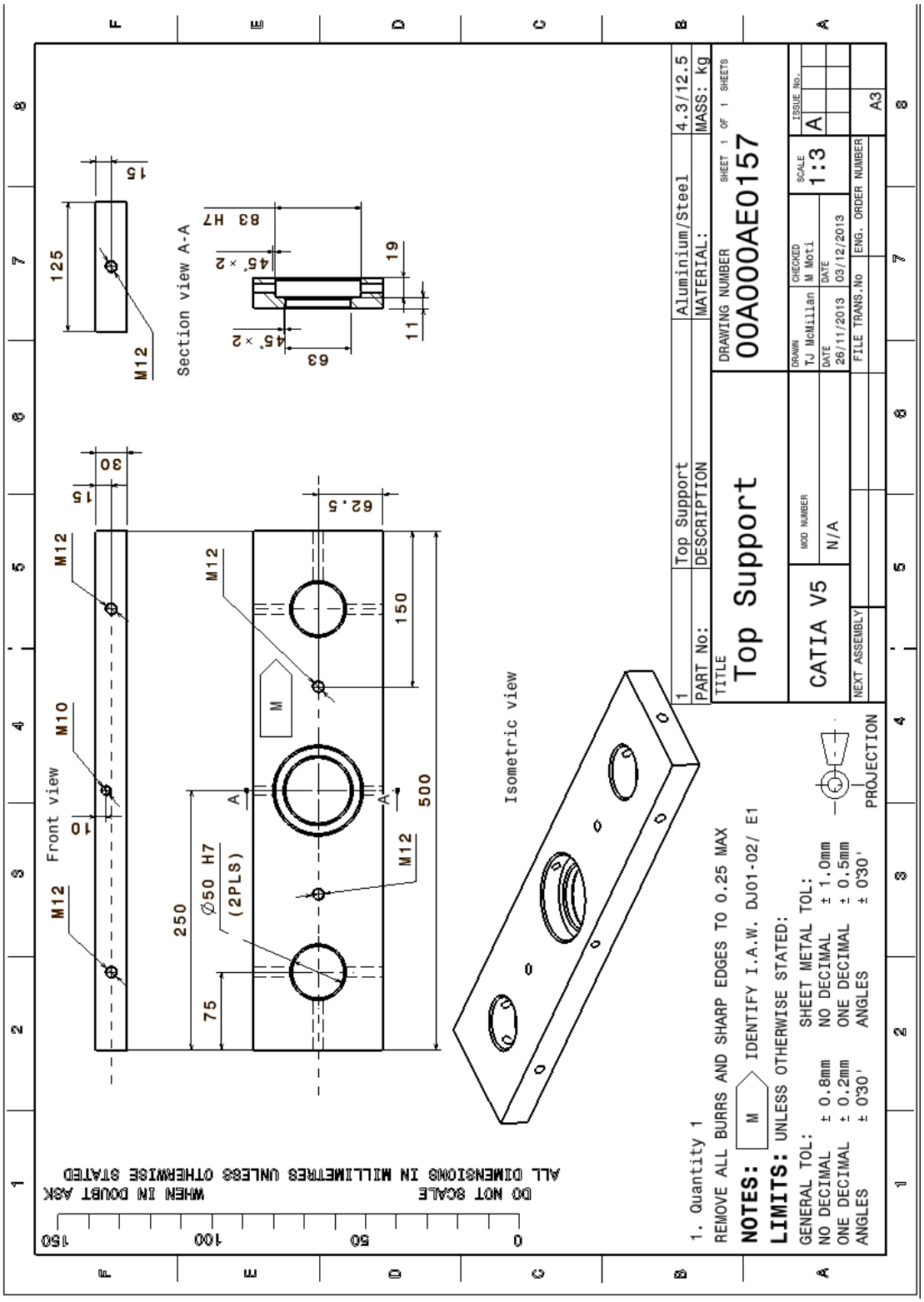
E.4 Base of Test Rig



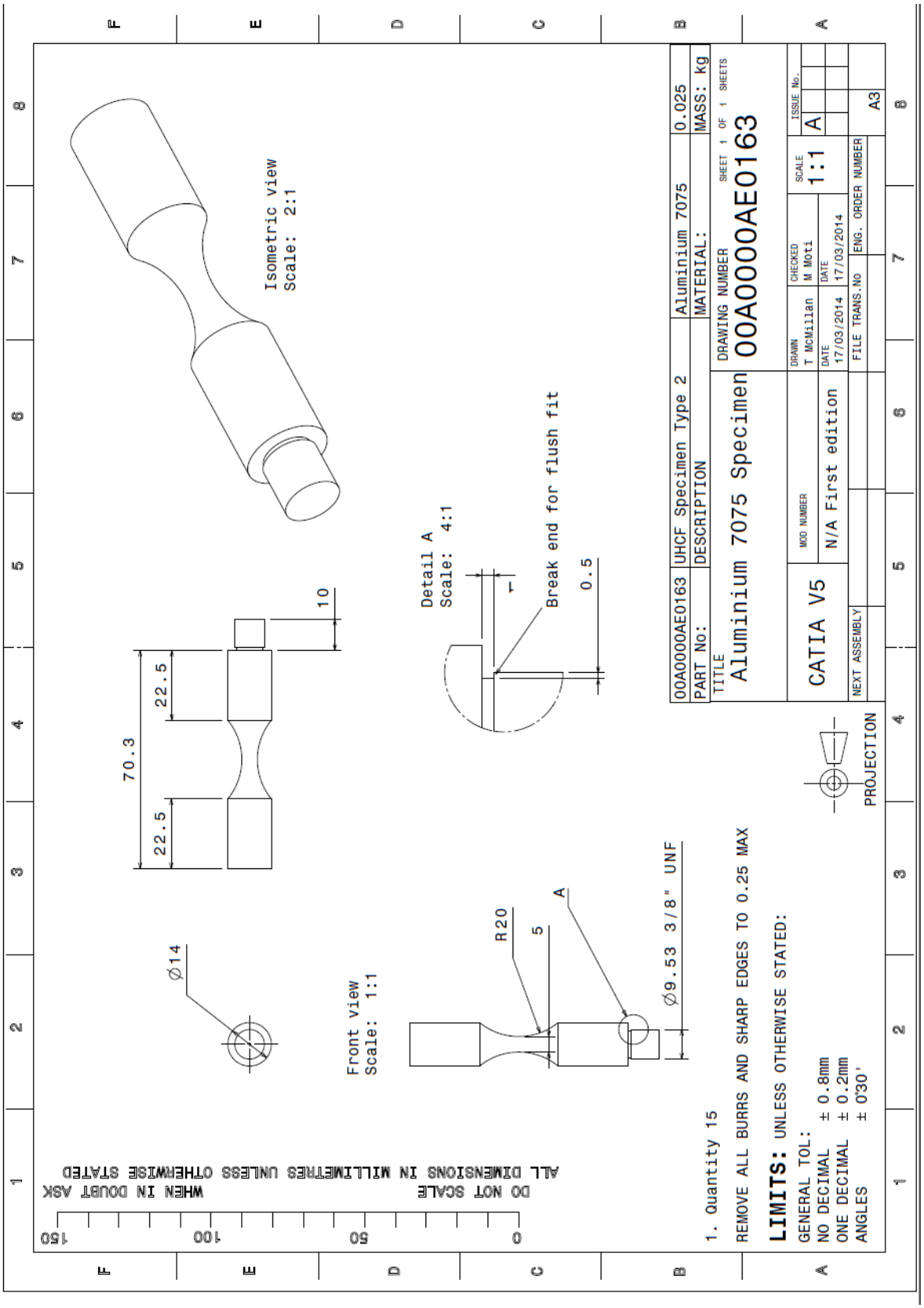
E.5 Pillar


Isometric view
Scale: 1:6
Front view
Ø50 g6
45° × 2
800
M
1. Quantity 2 (may be steel or aluminium)
REMOVE ALL BURRS AND SHARP EDGES TO 0.25 MAX
NOTES: M IDENTIFY I.A.W. DJ01-02/ E1
LIMITS: UNLESS OTHERWISE STATED:
GENERAL TOL: SHEET METAL TOL:
NO DECIMAL ± 0.8mm NO DECIMAL ± 1.0mm
ONE DECIMAL ± 0.2mm ONE DECIMAL ± 0.5mm
ANGLES ± 0°30' ANGLES ± 0°30'
PROJECTION
DO NOT SCALE
WHEN IN DOUBT ASK
ALL DIMENSIONS IN MILLIMETRES UNLESS OTHERWISE STATED
1 Pilliar column Aluminium/Steel 4.3/12.3
PART No: DESCRIPTION MATERIAL: MASS: kg
TITLE
Pilliar
DRAWING NUMBER
00A000AE0158
SHEET 1 OF 1 SHEETS
CATIA V5
MOD NUMBER
N/A
DRAWN TJ McMillan DATE 26/11/2013
CHECKED M Moti DATE 03/12/2013
SCALE 1:3
ISSUE No. A
NEXT ASSEMBLY
FILE TRANS.No
ENG. ORDER NUMBER
A3

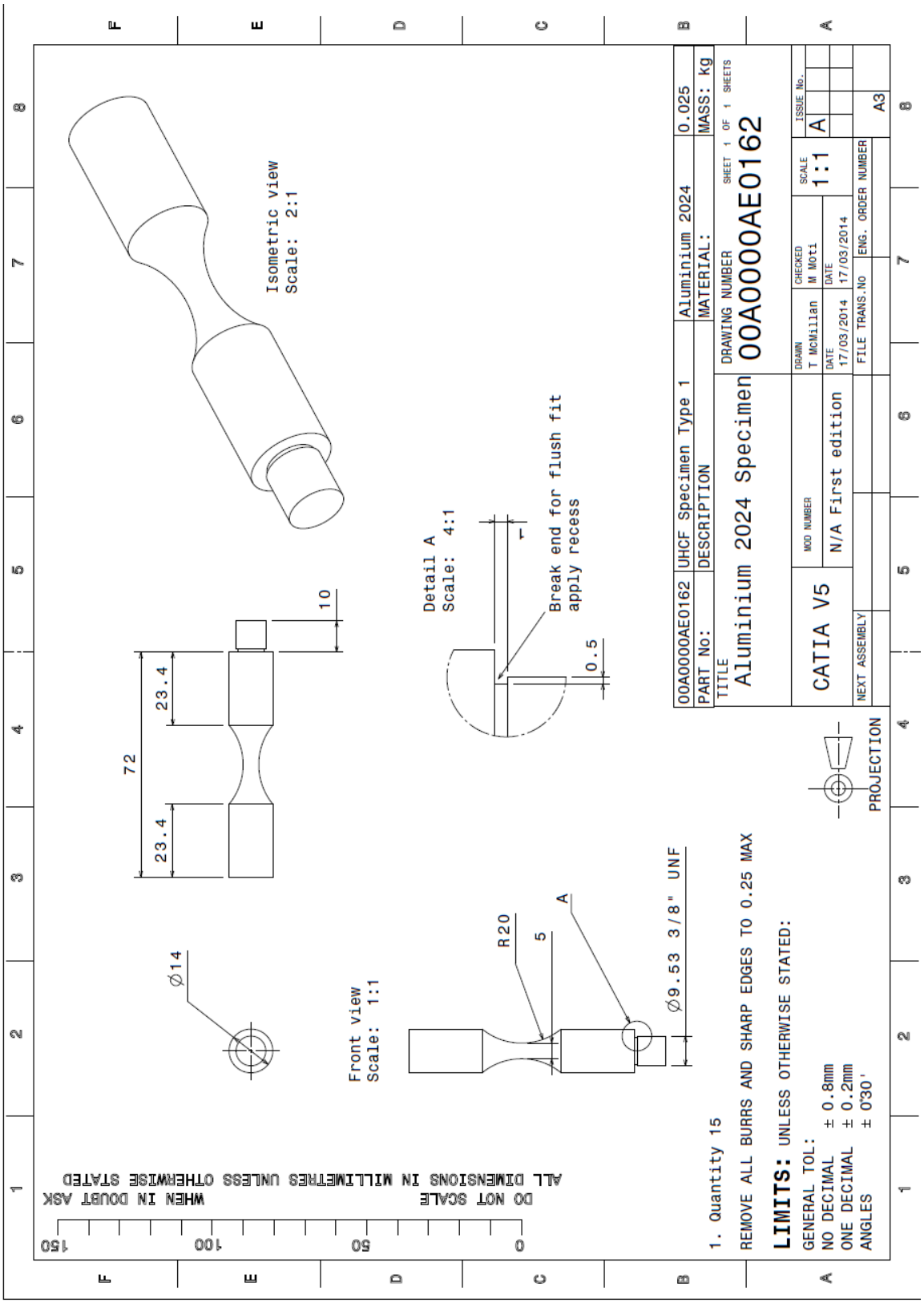
E.6 Top Support



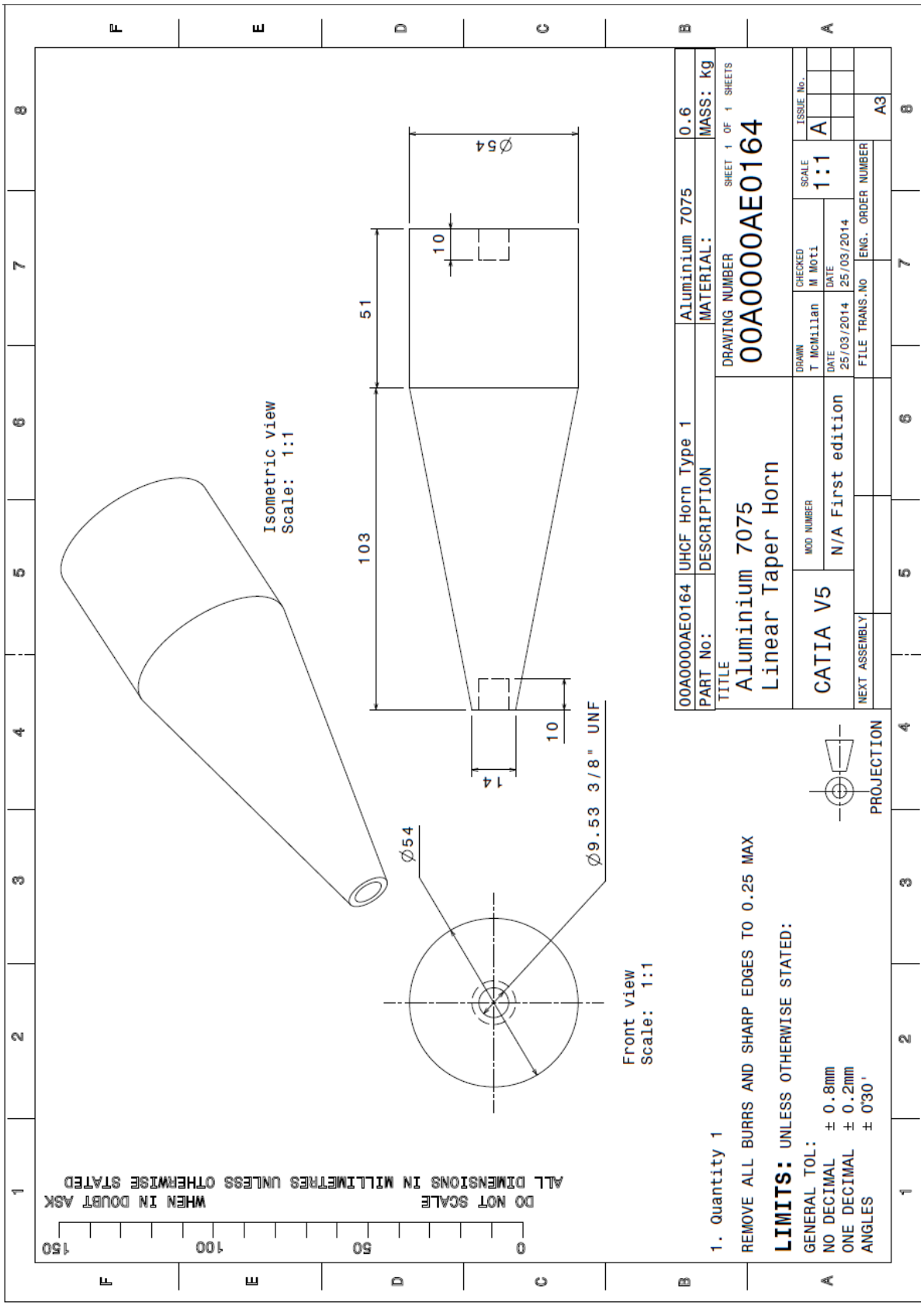
E.7 AL7075 specimen



E.8 AL2024 specimen



E.9 AL7075 Horn



APPENDIX F - STATISTICAL METHODS IMPLEMENTED

F.1 Least squares solution

In static analysis of materials the B-basis is defined as a 10-percentile value $X_{b.}$ This value is a value which predicts 90% of all possible results for this specific test and failure method will exceed this value with a confidence level of 95%. Similarly the A-basis is a 1-percentile value $X_{a.}$ This value is a value which predicts 99% of all possible results for this specific test and failure method will exceed this value with a confidence level of 95%.

The basis values are normally calculated using:

The sample mean as:

$$\bar{X} = \frac{1}{n}\sum_{i=1}^{n} X_i$$

The standard deviation:

$$s_x = \frac{1}{n-1}\sqrt{\sum_{i=1}^{n}\left(X_i - \bar{X}\right)^2}$$

Where n is the number of samples.

Thus the A-basis and B-bases are respectively:

$$X_a = \mu_x - 2.3264 \cdot \sigma_x \qquad X_b = \mu_x - 1.2816 \cdot \sigma_x$$

Where

μ_x – is the mean of the sample

σ_x – is the standard deviation of the sample

X_a and X_b are the A and B basis values.

However in fatigue the idea of a singular basis value is not inherent. Thus the data is normally curve fit according to residuals as per Military handbook 15 see reference [32].

In the statistical analysis of fatigue data, regression analysis is used to obtain the required confidence level. Typically the 'best fit' approach is used to linear regression such as that of the least squares method. This is the typical method implemented by Excel which reduces/minimises the residual error by solving for the unknown constants of the given curve type.

The typical linear least squares curve fit can be obtained using the following formulae and Matlab is can be solved using Matlab's Fminunc function:

$$R = \sum_{i=1}^{n} [O_i - (x_1 I_i + x_2)]^2$$

Where

O_i	- Is i-th output data point value
I_i	- Is the i-th input data point value
X1 and x2	- Are constants of the linear curve fit
R	- Is the overall residual error squared
n	- Is the number of data points in the set

The typical Matlab code to use is:

```
x=[1;1];
options=optimset('TolCon',1e-8,'TolX',1e-8, 'Display','iter');
[x,fval]=fminunc('Least_Squares_Eqn',x,options,O,I)
```

Where the Least_Square_Eqn function is defined by:

```
function f= Least_Squares_Eqn (x,O,I)
    x1=x(1);
    x2=x(2);
    f=0;
    for i = 1: size(O,2)
        f=f+(O(i)-(x1*I(i)+x2))^2;
    end
end
```

Similarly this can be done for the power curve fit. All that changes is the Least_Squares_Eqn function which needs to be equivalent too:

$$R = \sum_{i=1}^{n} [O_i - (x_1 {I_i}^{x_2})]^2$$

The variables definitions are as above. Thus the following curve fit methods will be used in-conjunction with Excels built in options.

The Excel trend lines use the R-squared ratio as an indication of the accuracy of the regression curve fit. The formulations behind the R-squared factor are given by:

$$\overline{y} = \sum_{i=1}^{n} y_i$$

$$SSreg = \sum_{i=1}^{n} [f_i - \overline{y}]^2$$

$$SSres = \sum_{i=1}^{n} [y_i - f_i]^2$$

$$SStot = \sum_{i=1}^{n} [y_i - \overline{y}]^2$$

Where

y_i - Is the know i-th data point
$\overline{y}$ - Is the mean of the known data
f_i - Is the corresponding predicted i-th value

Then the R-square value is given by:

$$R^2 = 1 - \frac{SSres}{SStot}$$

This is implemented in Matlab as:

```
y_bar=mean(O)
SSreg=0;
SSres=0;
SStot=0;
for i=1:size(O,1)
    SStot=SStot+(O(i,1)-y_bar)^2;
    SSreg=SSreg+(a*I(i,1)^b-y_bar)^2;
    SSres=SSres+(O(i,1)-a*I(i,1)^b)^2;
end
R_squared=1-SSres/SStot
```

APPENDIX G - MATLAB CODE

This appendix details all the code need for the FEM models together with the analytical code for the solution of the second order differential wave equation. If possible the typical output is indicated to given an idea of what would be expected from the executed code.

G.1 Main code

The code is called Calculations_TaperedHorn_SpecimenAL7075.m for the solution of the analytical wave equation.

Brief description

```
%Code to calculate the general equations of Ultrasonic fatigue
%Specimen has:
%               a hyperbolic cosine distribution.
%               a linear horn
%               two cylinder section R3,L4 and R2,L2
%
clc
clear
% close all

%define global variables to be used in ode
global k
global R
global L

%Assume material
%--------------------------------------------------------------------------
```

MATERIAL PROPERTIES

```
%Al7075
rho=2796; %kg/m^3
Ed=71.016*10^9;%Pa
v=0.33;%poissons ration
%
%--------------------------------------------------------------------------
```

Inputs

```
%define the A_input constant (Input Displacement)
disp('Input Displacement - A_Input constant')
A_input=0.01*10^-3%10 micro meters
%define the A_input constant (Input Displacement)
disp('Input Frequency - f')
%General constant
f=20*10^3;%[Hz]define the frequency of the wave 20 kHz
%--------------------------------------------------------------------------
```

```
Input Displacement - A_Input constant

A_input =

   1.0000e-05

Input Frequency - f
```

Known Geometry

```
%GEOMETRY PROPERTIES (From pg30 Gigacycle fatigue) ENDURANCE SPECIMEN
%see note book Tab final design:
L=[51;103;22.7;70.7-22.7;22.7]*10^-3;

%define radi
R=[54;14;14;5;14]/2*10^-3;
%--------------------------------------------------------------------------
```

Calculate wave frequency and speed

```
%define the radial frequency
w=2*pi*f;

%calculate the wave velocity
disp('Wave Velocity - Ignore poissons effect')
c=sqrt(Ed/rho)


%Define total length of speciment
Ltot=L(1)+L(2)+L(3)+L(4)+L(5);

%Create a vecotr to creat plots
Stepsize=(Ltot/1000);
y_upper=0;

%solve the differential equation of the tapered section
%define initial conditions of the system as zero disp, zero vel
u0=[A_input;0];%[u_0;u_dot_0]

%define the specific region for the taper
x_range=[0;Ltot];

%define k as per ref 1 (Giga cycle fatigue Bathius) pg 8/41
k=w/c;

%--------------------------------------------------------------------------
```

```
Wave Velocity - Ignore poissons effect

c =

   5.0398e+03
```

Solve circle equation

```
%solve the circle equation
options=optimset('Display','iter');
z=[L(1)+L(2)+L(3);L(1)+L(2)+L(3)+L(4)/2;L(1)+L(2)+L(3)+L(4)]
y=[R(3);R(4);R(5)]
x0=[20*10^-3;z(1);y(1)];%r,a,b
%test unit circle at orgin
% z=[-1;0;1]
% y=[0;-1;0]
% x=[1;0;0]
[const,fval]=fsolve(@getCircleConstants,x0,options,z,y)
```

```
z =

    0.1767
    0.2007
    0.2247


y =

    0.0070
    0.0025
    0.0070
```

```
                                         Norm of      First-order   Trust-region
 Iteration  Func-count     f(x)            step         optimality    radius
     0          4     3.82373e-06                       0.000192              1
     1          5     3.82373e-06       0.0670005       0.000192              1
     2          9     5.63349e-07       0.0167501       4.49e-05         0.0168
     3         10     5.63349e-07       0.0418753       4.49e-05         0.0419
     4         14     1.38255e-07       0.0104688       7.82e-06         0.0105
     5         15     1.38255e-07       0.0261721       7.82e-06         0.0262
     6         19     1.04075e-07      0.00654301       9.57e-06        0.00654
     7         23     7.40793e-08      0.00654301       1.19e-06        0.00654
     8         27     4.24618e-08       0.0163575       1.51e-05         0.0164
     9         31      8.8233e-09       0.0163575       3.63e-06         0.0164
    10         35     9.08306e-10       0.0171003        6.9e-06         0.0409
    11         39     8.99146e-16     0.000131584       6.88e-09         0.0409

Equation solved.

fsolve completed because the vector of function values is near zero
as measured by the default value of the function tolerance, and
the problem appears regular as measured by the gradient.



const =

    0.0663
    0.2007
    0.0688


fval =

   1.0e-07 *

    0.1731
    0.1731
    0.1731
```

Solve the wave equation for current horn + specimen setup

```
%define better variable for circle geometry
CircData=const;
%calculate the displacement under applied for
[x_region,U_32]=ode45(@Tapered_Ode_Solver,x_range,u0,options,L,R,k,CircData);

%need to flip U_32
% U_23=fliplr(U_32')';
```

Plot the geometry

```
%create the x array with the tappered array inbedded
x=0:Stepsize:Ltot;
%~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
%DEFINE SPECIMENT SURFACES
for i=1:size(x,2)
    %L2<abs(x)<=L
    if (abs(x(i))<L(1) & abs(x(i))>=0)
        y_upper(i)=R(1);
    %linear taper section
    elseif (abs(x(i))<(L(1)+L(2)) & abs(x(i))>=L(1))
        a=(R(2)-R(1))/L(2);
        b=R(1)-L(1)*a;
        y_upper(i)=a*x(i)+b;
    elseif (abs(x(i))<(L(1)+L(2)+L(3)) & abs(x(i))>=(L(1)+L(2)))
        y_upper(i)=R(3);
    %circle section
    elseif (abs(x(i))<(L(1)+L(2)+L(3)+L(4)) & abs(x(i))>=(L(1)+L(2)+L(3)))
         r=const(1);
         a=const(2);
         b=const(3);
         y_upper(i)=-sqrt(r^2-(x(i)-a)^2)+b;
    elseif (abs(x(i))<=(Ltot) & abs(x(i))>=(L(1)+L(2)+L(3)+L(4)))
        y_upper(i)=R(5);
    else
        disp('No Y value for current x location:');
        x(i)
```

```
    end
    % sigma(i)=epsilon(i)*Ed;
end
%!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
%CONVERT DATA TO MM
x=x*1000;
y_upper=y_upper*1000;
y_lower=-1*y_upper;
%--------------------------------------------------------------------------
%PLOT THE SPECIMEN
%
%plot the upper and lower surfaces with a mirror
figure(1);
subplot(4,1,1);
fac=1.5;
%subplot(4,1,1)
plot(x,y_upper,'-b','LineWidth',1.5);
hold on;
%plot lower surface
plot(x,y_lower,'-b','LineWidth',1.5);
%Plot verticle lines at step changes
VertLine_x=[L(1);L(1)]*1000;
VertLine_y=[-R(1);R(1)]*1000*fac;
plot(VertLine_x,VertLine_y,'-r','LineWidth',2);
%Plot verticle lines at step changes
VertLine_x=[(L(1)+L(2));(L(1)+L(2))]*1000;
VertLine_y=[-R(1);R(1)]*1000*fac;
plot(VertLine_x,VertLine_y,'-r','LineWidth',2);
%Plot verticle lines at step changes
VertLine_x=[(L(1)+L(2)+L(3));(L(1)+L(2)+L(3))]*1000;
VertLine_y=[-R(1);R(1)]*1000*fac;
plot(VertLine_x,VertLine_y,'-r','LineWidth',2);
%plot at centre of specimen
VertLine_x=[(L(1)+L(2)+L(3)+L(4)/2);(L(1)+L(2)+L(3)+L(4)/2)]*1000;
VertLine_y=[-R(1);R(1)]*1000*fac;
plot(VertLine_x,VertLine_y,'-r','LineWidth',2);
%plot at end of curve on specimen
VertLine_x=[(L(1)+L(2)+L(3)+L(4));(L(1)+L(2)+L(3)+L(4))]*1000;
VertLine_y=[-R(1);R(1)]*1000*fac;
plot(VertLine_x,VertLine_y,'-r','LineWidth',2);
hold off;
xlabel('Location [mm]','FontSize', 9);
ylabel('Displacement of outer edge [mm]','FontSize', 9);
title('Ultra-sonic Fatigue Test Specimen','Fontweight','bold','FontSize', 16);
axis([min(x) max(x) min(y_lower)*fac max(y_upper)*fac]);
grid on;

% Plot the displacment
%plot the upper and lower surfaces with a mirror
subplot(4,1,2);
plot(x_region*1000,U_32(:,1),'k','LineWidth',1.5);
Umax=max(U_32);
Umin=min(U_32);
hold on;
%Plot verticle lines at step changes
VertLine_x=[L(1);L(1)]*1000;
VertLine_y=[Umin(1);Umax(1)]*fac;
plot(VertLine_x,VertLine_y,'-r','LineWidth',2);
%Plot verticle lines at step changes
VertLine_x=[(L(1)+L(2));(L(1)+L(2))]*1000;
plot(VertLine_x,VertLine_y,'-r','LineWidth',2);
%Plot verticle lines at step changes
VertLine_x=[(L(1)+L(2)+L(3));(L(1)+L(2)+L(3))]*1000;
plot(VertLine_x,VertLine_y,'-r','LineWidth',2);
%plot at centre of specimen
VertLine_x=[(L(1)+L(2)+L(3)+L(4)/2);(L(1)+L(2)+L(3)+L(4)/2)]*1000;
plot(VertLine_x,VertLine_y,'-r','LineWidth',2);
%plot at end of curve on specimen
VertLine_x=[(L(1)+L(2)+L(3)+L(4));(L(1)+L(2)+L(3)+L(4))]*1000;
plot(VertLine_x,VertLine_y,'-r','LineWidth',2);
hold off;
xlabel('Location [mm]','FontSize', 9);
ylabel('Displacement [mm]','FontSize', 9);
title('Displacement','Fontweight','bold','FontSize', 12);
axis([min(x_region*1000) max(x_region*1000) min(U_32(:,1))*fac max(U_32(:,1))*fac]);
grid on;

% Plot the strain
%plot the upper and lower surfaces with a mirror
subplot(4,1,3);
U_32(:,2)=U_32(:,2)*10^6; %strain to micro strain
plot(x_region*1000,U_32(:,2),'-g','LineWidth',1.5);
Umax=max(U_32);
Umin=min(U_32);
hold on;
```

```
%Plot verticle lines at step changes
VertLine_x=[L(1);L(1)]*1000;
VertLine_y=[Umin(2);Umax(2)]*fac;
plot(VertLine_x,VertLine_y,'-r','LineWidth',2);
%Plot verticle lines at step changes
VertLine_x=[(L(1)+L(2));(L(1)+L(2))]*1000;
plot(VertLine_x,VertLine_y,'-r','LineWidth',2);
%Plot verticle lines at step changes
VertLine_x=[(L(1)+L(2)+L(3));(L(1)+L(2)+L(3))]*1000;
plot(VertLine_x,VertLine_y,'-r','LineWidth',2);
%plot at centre of specimen
VertLine_x=[(L(1)+L(2)+L(3)+L(4)/2);(L(1)+L(2)+L(3)+L(4)/2)]*1000;
plot(VertLine_x,VertLine_y,'-r','LineWidth',2);
%plot at end of curve on specimen
VertLine_x=[(L(1)+L(2)+L(3)+L(4));(L(1)+L(2)+L(3)+L(4))]*1000;
plot(VertLine_x,VertLine_y,'-r','LineWidth',2);
hold off;
xlabel('Location [mm]','FontSize', 9);
ylabel('Strain [\mu\epsilon]','FontSize', 9);
title('Strain','Fontweight','bold','FontSize', 12);
axis([min(x_region*1000) max(x_region*1000) min(U_32(:,2))*fac max(U_32(:,2))*fac]);
grid on;

%define max strain
max_strain=max(U_32(:,2))

%max strain
max_stress=Ed*max_strain*10^-6*10^-6 %convert micro strain back to strain covert Pa to MPa

%calculate estimated life using MMPDS
ksiToMPa=6.894759087;%MPa
StressRatio=-1
Seq=max_stress/ksiToMPa*(1-StressRatio)^0.49
Nf=10^(14.86-5.80*log(Seq))

% Create Voltage vs displacement curve data (Note strain is in microstrain)
[x_volt,Volt]=GetVoltageFromStrain(x_region*1000,U_32(:,2)*10^-6);

% Plot the voltage
%plot the upper and lower surfaces with a mirror
%figure(2)
subplot(4,1,4);
plot(x_volt,Volt*1000,'-c','LineWidth',1.5);
hold on;
%Plot verticle lines at step changes
VertLine_x=[L(1);L(1)]*1000;
VertLine_y=[min(Volt*1000);max(Volt*1000)]*fac;
plot(VertLine_x,VertLine_y,'-r','LineWidth',2)
%Plot verticle lines at step changes
VertLine_x=[(L(1)+L(2));(L(1)+L(2))]*1000;
plot(VertLine_x,VertLine_y,'-r','LineWidth',2)
%Plot verticle lines at step changes
VertLine_x=[(L(1)+L(2)+L(3));(L(1)+L(2)+L(3))]*1000;
plot(VertLine_x,VertLine_y,'-r','LineWidth',2)
%plot at centre of specimen
VertLine_x=[(L(1)+L(2)+L(3)+L(4)/2);(L(1)+L(2)+L(3)+L(4)/2)]*1000;
plot(VertLine_x,VertLine_y,'-r','LineWidth',2)
%plot at end of curve on specimen
VertLine_x=[(L(1)+L(2)+L(3)+L(4));(L(1)+L(2)+L(3)+L(4))]*1000;
plot(VertLine_x,VertLine_y,'-r','LineWidth',2)
hold off;
xlabel('Location [mm]','FontSize', 9);
ylabel('Voltage [mV]','FontSize', 9);
title('Voltage vs Displacement','Fontweight','bold','FontSize', 12);
axis([min(x_volt) max(x_volt) min(Volt*1000)*fac max(Volt*1000)*fac]);
grid on;
```

```
max_strain =

   2.8473e+03

max_stress =

  202.2017

StressRatio =

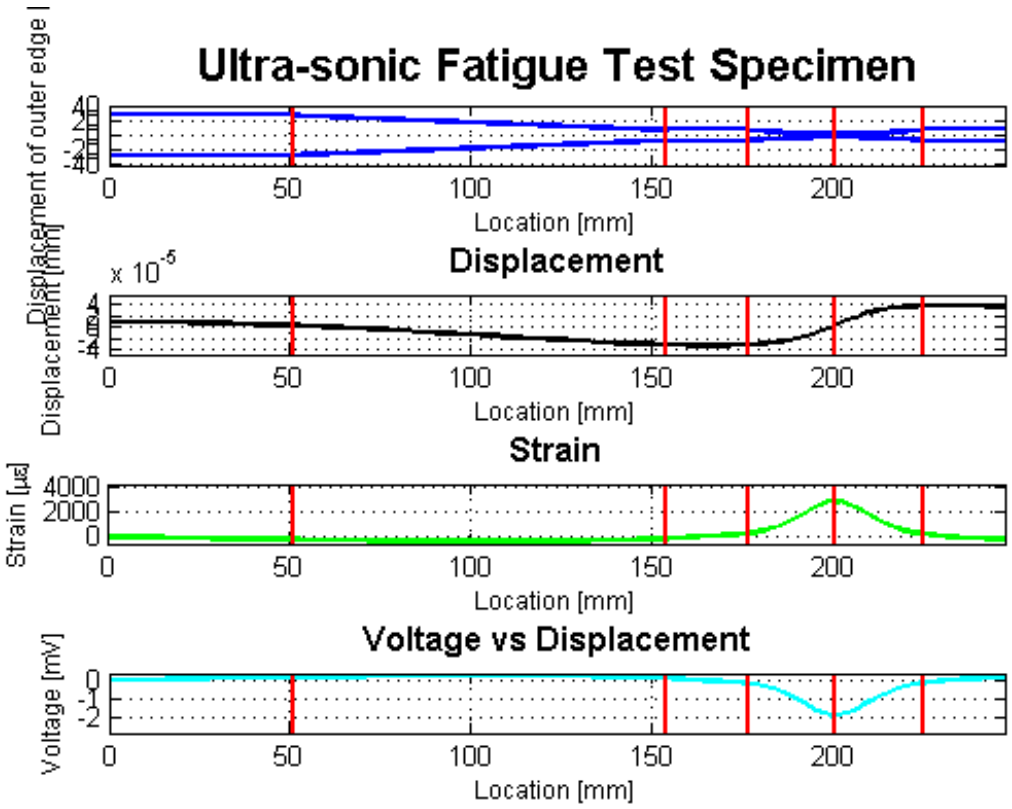
    -1

Seq =
```

```
    41.1880

Nf =

    1.9713e-07
```



Get voltage value at a location

```
%get voltage at a x loction
xloc=50
Vloc=interp1(x_volt,Volt*1000,xloc) %output on mv
```

```
xloc =

    50

Vloc =

    0.1542
```

G.1.1 Tapered_ODE_Solver function

```
%%Define the E matrix where
% y_dot={y_dot; y_ddot}=[E]y=[0 1;-k 2*a/(ax+b)]{y y_dot} + [Q]
function udot=Tapered_Ode_Solver(fx,fu0,L,R,k,CircData)
%%define global variables
% global k
% global R
% global L
Ltot=sum(L);

%%Depending on location along x the function to solve will vary
if (abs(fx)<L(1) & abs(fx)>=0)
    %%in the first flat cylinder region
    a=0;
    b=R(1);
    Emat=[0 1; -k^2 -2*a/(a*fx+b)];
    %define constant matrix which is nothing in this case
    Qmat=[0 0]';
 elseif (abs(fx)<(L(1)+L(2)) & abs(fx)>=L(1))
    %%in the first linear region
    a=(R(2)-R(1))/L(2);
    b=R(1)-L(1)*a;
    Emat=[0 1; -k^2 -2*a/(a*fx+b)];
    %define constant matrix which is nothing in this case
    Qmat=[0 0]';
elseif (abs(fx)<(L(1)+L(2)+L(3)) & abs(fx)>=(L(1)+L(2)))
    %%in the 2nd flat cylinder region of specimen
    a=0;
    b=R(2);
    Emat=[0 1; -k^2 -2*a/(a*fx+b)];
    %define constant matrix which is nothing in this case
    Qmat=[0 0]';
elseif (abs(fx)<(L(1)+L(2)+L(3)+L(4)) & abs(fx)>=(L(1)+L(2)+L(3)))
    %left hand curve of circle
    r=CircData(1);
    a=CircData(2);
    b=CircData(3);
    r_1=-sqrt(r^2-(fx-a)^2)+b;
    Pofx=2*(fx-a)/(r_1*sqrt(r^2-(fx-a)^2));
    Emat=[0 1; -k^2 -Pofx];
    %define constant matrix which is nothing in this case
    Qmat=[0 0]';
elseif (abs(fx)<=(Ltot) & abs(fx)>=(L(1)+L(2)+L(3)+L(4)))
    %right hand curve of circle
    r=CircData(1);
    a=CircData(2);
    b=CircData(3);
    r_1=-sqrt(r^2-(fx-a)^2)+b;
    Pofx=2*(fx-a)/(r_1*sqrt(r^2-(fx-a)^2));
    Emat=[0 1; -k^2 -Pofx];
     %define constant matrix which is nothing in this case
    Qmat=[0 0]';
else
    %%in the last flat cylinder region of specimen
    a=0;
    b=R(5);
    Emat=[0 1; -k^2 -2*a/(a*fx+b)];
    %define constant matrix which is nothing in this case
    Qmat=[0 0]';
end

udot=Emat*fu0+Qmat;

end %end function
```

G.1.2 getCircleConstants function

Function to solve for a circle using 3 points defined in the zy plane

```
function F=getCircleConstants(fx,z,y)
r=fx(1);
a=fx(2);
b=fx(3);

F=[ r^2-(z(1)-a)^2-(y(1)-b)^2
    r^2-(z(2)-a)^2-(y(2)-b)^2
    r^2-(z(3)-a)^2-(y(3)-b)^2
        ];
```

G.1.3 GetVoltageFromStrain function

Function to generate the Voltage for a given strain

```
function [x,V]=GetVoltageFromStrain(x_volt,Strain,Vinput,v,GainStrainGauge,BridgeType,GainSignalCond)
%Variable definition
%x_volt - corresponding x location (row or col vector/single value)
%Vinput - Input voltage
%v      - poissons ration of material
%Strain - Strain corresponding to x locations if col the votage is col if
%           row then voltage is row vector
%GainStrainGauge - strain gauge gain
%BridgeType - Type of bridge used
%               1. (dfault) Full bridge for tension specimen (poison gauges
%                   on onpossite sides of bridge
%               2. Half bridge as per report
%               3. full bridge for bending only strain on each gauge
%               4. full bridge poisons gauges on one side for full bridge
%GainSignalCond
```

Input variable checks

```
if (nargin == 2)
    Vinput=1;
    v=0.3;
    GainStrainGauge=1;
    BridgeType=1;
    GainSignalCond=1;
elseif (nargin == 3)
    v=0.3;
    GainStrainGauge=1;
    BridgeType=1;
    GainSignalCond=1;
elseif (nargin == 4)
    GainStrainGauge=1;
    GainSignalCond=1;
    BridgeType=1;
elseif (nargin == 5)
    GainSignalCond=1;
    BridgeType=1;
elseif (nargin == 6)
    GainSignalCond=1;
elseif (nargin == 7)
    %no missing arguments
else
   disp('Erron in GetVoltageFromStrain input arguments')
   return
end
```

Check both vectors are row vector

```
x_voltFlip=false;
if ~(size(x_volt,1)>=size(x_volt,2))
    x_volt=x_volt';
    x_voltFlip=true;
end
StrainFlip=false;
if ~(size(Strain,1)>=size(Strain,2))
    Strain=Strain';
    StrainFlip=true;
end
```

Get Voltages

```
x=x_volt;
GF=GainStrainGauge;
for i=1:size(Strain,1)
    switch BridgeType
        case 1 % full bridge axial
            V(i)=Vinput*GF*GainSignalCond*(Strain(i)*(v+1))/(Strain(i)*GF*(v-1)-2);
        case 2 % half bridge configuration 3.12.1.5 report
            V(i)=Vinput*GF*GainSignalCond*Strain(i)/2;
        case 3 % full bridge axial only strain (comp/tens) on gauges
            V(i)=Vinput*GF*GainSignalCond*Strain(i);
        case 4 % full bridge bending posisons on one side only
            Vinput*GF*GainSignalCond*Strain(i)/2;
        otherwise
            disp('Unknow BridgeType specified')
            return
    end %end case
end% end for
```

Flip row vectors to columns if that was received

```
if(x_voltFlip)
    x=x';
end
if(StrainFlip)
    V=V';
end
```

G.2 AL7075 Horn and AL2024 Specimen Analytical Code

This code differs slightly from that of the AL7075 code in that the horn material properties differ from the specimen. Thus the ODE solver needs to use the cored material properties depending on where it is analysing along the stack.

Brief description

```
%Code to calculate the general equations of Ultrasonic fatigue
%Specimen has:
%               a hyperbolic cosine distribution.
%               a linear horn
%               two cylinder section R3,L4 and R2,L2
%
```

```
clc
clear
% close all

%define global variables to be used in ode
global k
global R
global L

%Assume material
%-------------------------------------------------------------------------
```

MATERIAL PROPERTIES

```
%Al7075
rho(1)=2796; %kg/m^3
Ed(1)=71.016*10^9;%Pa
v(1)=0.33;%poissons ration

%Al2024
rho(2)=2767.99; %kg/m^3
Ed(2)=72.395*10^9;%Pa
v(2)=0.33;
%to save run
%path='Q:\Shared_Pool\Tristan\TJ_McMillan_Masters\Data\TestData\AL2024\H1_S2_3\20140805_025353_PM\'
%saveas(hh,[path 'Horn&Specimen_stress_ai0'],'jpg')
%-------------------------------------------------------------------------
```

Inputs

```
%define the A_input constant (Input Displacement)
disp('Input Displacement - A_Input constant')
A_input=0.9e-06%10 micro meters
%define the A_input constant (Input Displacement)
disp('Input Frequency - f')
%General constant
f=20*10^3;%[Hz]define the frequency of the wave 20 kHz
%-------------------------------------------------------------------------
```

Known Geometry

```
%GEOMETRY PROPERTIES (From pg30 Gigacycle fatigue) ENDURANCE SPECIMEN
%see note book Tab final design:
L=[51;103;23.4;72.0-23.4*2;23.4]*10^-3;

%define radi
R=[54;14;14;5;14]/2*10^-3;
%-------------------------------------------------------------------------
```

Calculate wave frequency and speed

```
%define the radial frequency
w=2*pi*f;

%calculate the wave velocity
disp('Wave Velocity - Ignore poissons effect')
c(1)=sqrt(Ed(1)/rho(1))
c(2)=sqrt(Ed(2)/rho(2))

%Define total length of speciment
Ltot=L(1)+L(2)+L(3)+L(4)+L(5);

%Create a vecotr to creat plots
Stepsize=(Ltot/1000);
y_upper=0;

%solve the differential equation of the tapered section
%define initial conditions of the system as zero disp, zero vel
u0=[A_input;0];%[u_0;u_dot_0]

%define the specific region for the taper
x_range.horn=[0;L(1)+L(2)];
x_range.spec=[L(1)+L(2);Ltot];

%define k as per ref 1 (Giga cycle fatigue Bathius) pg 8/41
k(1)=w/c(1);
k(2)=w/c(2);
%------------------------------------------------------------------------
```

Solve circle equation

```
%solve the circle equation
options=optimset('Display','iter');
z=[L(1)+L(2)+L(3);L(1)+L(2)+L(3)+L(4)/2;L(1)+L(2)+L(3)+L(4)]
y=[R(3);R(4);R(5)]
x0=[10*10^-3;z(1);y(1)];%r,a,b
%test unit circle at orgin
% z=[-1;0;1]
% y=[0;-1;0]
% x=[1;0;0]
[const,fval]=fsolve(@getCircleConstants,x0,options,z,y)
```

Solve the wave equation for current horn + specimen setup

```
%define better variable for circle geometry
CircData=const;
%calculate the displacement for horn region
[xhorn,Uhorn]=ode45(@Tapered_Ode_SolverHorn,x_range.horn,u0,options,L,R,k(1));
%calculate the displacement for specimen region
u0=[Uhorn(end,1);Uhorn(end,2)*Ed(1)/Ed(2)];
[xspec,Uspec]=ode45(@Tapered_Ode_SolverSpecimen,x_range.spec,u0,options,L,R,k(2),CircData);
```

```
x_region=[xhorn;xspec];
U_32=[Uhorn;Uspec];
lengthHorn=size(xhorn,1);
%need to flip U_32
% U_23=fliplr(U_32')';
```

Plot the geometry

```
%create the x array with the tappered array inbedded
x=0:Stepsize:Ltot;
%~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
%DEFINE SPECIMENT SURFACES
for i=1:size(x,2)
    %L2<abs(x)<=L
    if (abs(x(i))<L(1) & abs(x(i))>=0)
        y_upper(i)=R(1);
    %linear taper section
    elseif (abs(x(i))<(L(1)+L(2)) & abs(x(i))>=L(1))
        a=(R(2)-R(1))/L(2);
        b=R(1)-L(1)*a;
        y_upper(i)=a*x(i)+b;
    elseif (abs(x(i))<(L(1)+L(2)+L(3)) & abs(x(i))>=(L(1)+L(2)))
        y_upper(i)=R(3);
    %circle section
    elseif (abs(x(i))<(L(1)+L(2)+L(3)+L(4)) & abs(x(i))>=(L(1)+L(2)+L(3)))
         r=const(1);
         a=const(2);
         b=const(3);
         y_upper(i)=-sqrt(r^2-(x(i)-a)^2)+b;
    elseif (abs(x(i))<=(Ltot) & abs(x(i))>=(L(1)+L(2)+L(3)+L(4)))
        y_upper(i)=R(5);
    else
        disp('No Y value for current x location:');
        x(i)
    end
    % sigma(i)=epsilon(i)*Ed;
end
%!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
%CONVERT DATA TO MM
x=x*1000;
y_upper=y_upper*1000;
y_lower=-1*y_upper;
```

```
%PLOT THE SPECIMEN
%
%plot the upper and lower surfaces with a mirror
hh=figure(1);
subplot(4,1,1);
fac=1.5;
%subplot(4,1,1)
% plot(x,y_upper,'-b','LineWidth',1.5);
area(x,y_upper);
hold on;
%plot lower surface
area(x,y_lower);
% plot(x,y_lower,'-b','LineWidth',1.5);
```

```
%Plot verticle lines at step changes
VertLine_x=[L(1);L(1)]*1000;
VertLine_y=[-R(1);R(1)]*1000*fac;
plot(VertLine_x,VertLine_y,'-r','LineWidth',2);
%Plot verticle lines at step changes
VertLine_x=[(L(1)+L(2));(L(1)+L(2))]*1000;
VertLine_y=[-R(1);R(1)]*1000*fac;
plot(VertLine_x,VertLine_y,'-r','LineWidth',2);
%Plot verticle lines at step changes
VertLine_x=[(L(1)+L(2)+L(3));(L(1)+L(2)+L(3))]*1000;
VertLine_y=[-R(1);R(1)]*1000*fac;
plot(VertLine_x,VertLine_y,'-r','LineWidth',2);
%plot at centre of specimen
VertLine_x=[(L(1)+L(2)+L(3)+L(4)/2);(L(1)+L(2)+L(3)+L(4)/2)]*1000;
VertLine_y=[-R(1);R(1)]*1000*fac;
plot(VertLine_x,VertLine_y,'-r','LineWidth',2);
%plot at end of curve on specimen
VertLine_x=[(L(1)+L(2)+L(3)+L(4));(L(1)+L(2)+L(3)+L(4))]*1000;
VertLine_y=[-R(1);R(1)]*1000*fac;
plot(VertLine_x,VertLine_y,'-r','LineWidth',2);
hold off;
xlabel('Location [mm]','FontSize', 9);
ylabel('Location of outer edge [mm]','FontSize', 9);
title('Ultra-sonic Fatigue Test Specimen','Fontweight','bold','FontSize', 16);
axis([min(x) max(x) min(y_lower)*fac max(y_upper)*fac]);
grid on;
```

```
U_32=real(U_32); %convert imaginary to real
% Plot the displacment
%plot the upper and lower surfaces with a mirror
subplot(4,1,2);
plot(x_region*1000,U_32(:,1),'k','LineWidth',1.5);
Umax=max(U_32);
Umin=min(U_32);
hold on;
%Plot verticle lines at step changes
VertLine_x=[L(1);L(1)]*1000;
VertLine_y=[Umin(1);Umax(1)]*fac;
plot(VertLine_x,VertLine_y,'-r','LineWidth',2);
%Plot verticle lines at step changes
VertLine_x=[(L(1)+L(2));(L(1)+L(2))]*1000;
plot(VertLine_x,VertLine_y,'-r','LineWidth',2);
%Plot verticle lines at step changes
VertLine_x=[(L(1)+L(2)+L(3));(L(1)+L(2)+L(3))]*1000;
plot(VertLine_x,VertLine_y,'-r','LineWidth',2);
%plot at centre of specimen
VertLine_x=[(L(1)+L(2)+L(3)+L(4)/2);(L(1)+L(2)+L(3)+L(4)/2)]*1000;
plot(VertLine_x,VertLine_y,'-r','LineWidth',2);
%plot at end of curve on specimen
VertLine_x=[(L(1)+L(2)+L(3)+L(4));(L(1)+L(2)+L(3)+L(4))]*1000;
plot(VertLine_x,VertLine_y,'-r','LineWidth',2);
hold off;
xlabel('Location [mm]','FontSize', 9);
ylabel('Displacement [mm]','FontSize', 9);
title('Displacement','Fontweight','bold','FontSize', 12);
axis([min(x_region*1000) max(x_region*1000) min(U_32(:,1))*fac max(U_32(:,1))*fac]);
grid on;
```

```
% Plot the strain
%plot the upper and lower surfaces with a mirror
subplot(4,1,3);
U_32(:,2)=U_32(:,2)*10^6; %strain to micro strain
%Just plot strain
%plot(x_region*1000,U_32(:,2),'-g','LineWidth',1.5);
%plot Strain and stress
stress=[Ed(1)*10^-12*U_32(1:lengthHorn,2);Ed(1)*10^-12*U_32(lengthHorn+1:end,2)];
[hax,hl1,hl2]=plotyy(x_region*1000,U_32(:,2),x_region*1000,stress,'plot','plot');
set(hl1,'LineStyle','-','LineWidth',1.5,'color','b');
set(hl2,'LineStyle','o','LineWidth',1.5,'color','g');
Umax=max(U_32);
Umin=min(U_32);
hold('on');
%Plot verticle lines at step changes
VertLine_x=[L(1);L(1)]*1000;
VertLine_y=[Umin(2);Umax(2)]*fac;
plot(VertLine_x,VertLine_y,'-r','LineWidth',2);
%Plot verticle lines at step changes
VertLine_x=[(L(1)+L(2));(L(1)+L(2))]*1000;
plot(VertLine_x,VertLine_y,'-r','LineWidth',2);
%Plot verticle lines at step changes
VertLine_x=[(L(1)+L(2)+L(3));(L(1)+L(2)+L(3))]*1000;
plot(VertLine_x,VertLine_y,'-r','LineWidth',2);
%plot at centre of specimen
VertLine_x=[(L(1)+L(2)+L(3)+L(4)/2);(L(1)+L(2)+L(3)+L(4)/2)]*1000;
plot(VertLine_x,VertLine_y,'-r','LineWidth',2);
%plot at end of curve on specimen
VertLine_x=[(L(1)+L(2)+L(3)+L(4));(L(1)+L(2)+L(3)+L(4))]*1000;
plot(VertLine_x,VertLine_y,'-r','LineWidth',2);
hold('off');
xlabel(hax(2),'Location [mm]','FontSize', 9);
ylabel(hax(1),'Strain [\mu\epsilon]','FontSize', 9);
ylabel(hax(2),'Stress [MPa]','FontSize', 9);
title('Strain','Fontweight','bold','FontSize', 12);
% axis(hax(1),[min(x_region*1000) max(x_region*1000) floor(min(U_32(:,2))*fac)
ceil(max(U_32(:,2))*fac)]);
% axis(hax(2),[min(x_region*1000) max(x_region*1000) floor(min(stress)*fac) ceil(max(stress)*fac)]);
% %axis(hax(1),'tight');
% %axis(hax(2),'tight');
% grid(hax(1),'minor');
% grid(hax(1),'on');
% axis(hax(1),'tight');
axis(hax(1),[min(x_region*1000) max(x_region*1000) floor(min(U_32(:,2))*fac)
ceil(max(U_32(:,2))*fac)]);
axis(hax(2),[min(x_region*1000) max(x_region*1000) floor(min(stress)*fac) ceil(max(stress)*fac)]);
% axis(hax(2),'tight');
NumYgrid=5;% this is the number of y-measurments required
ymin=floor(min(U_32(:,2)));
ymin=round(ymin/NumYgrid)*NumYgrid;
ymax=ceil(max(U_32(:,2)));
ymax=round(ymax/NumYgrid)*NumYgrid;
set(hax(1),'ytick',[ymin:(ymax-ymin)/NumYgrid:ymax]);
ymin=floor(min(stress));
ymin=round(ymin/NumYgrid)*NumYgrid;
```

```
ymax=ceil(max(stress))
ymax=round(ymax/NumYgrid)*NumYgrid;
set(hax(2),'ytick',[ymin:(ymax-ymin)/NumYgrid:ymax]);
grid(hax(1),'minor');
grid(hax(1),'on');

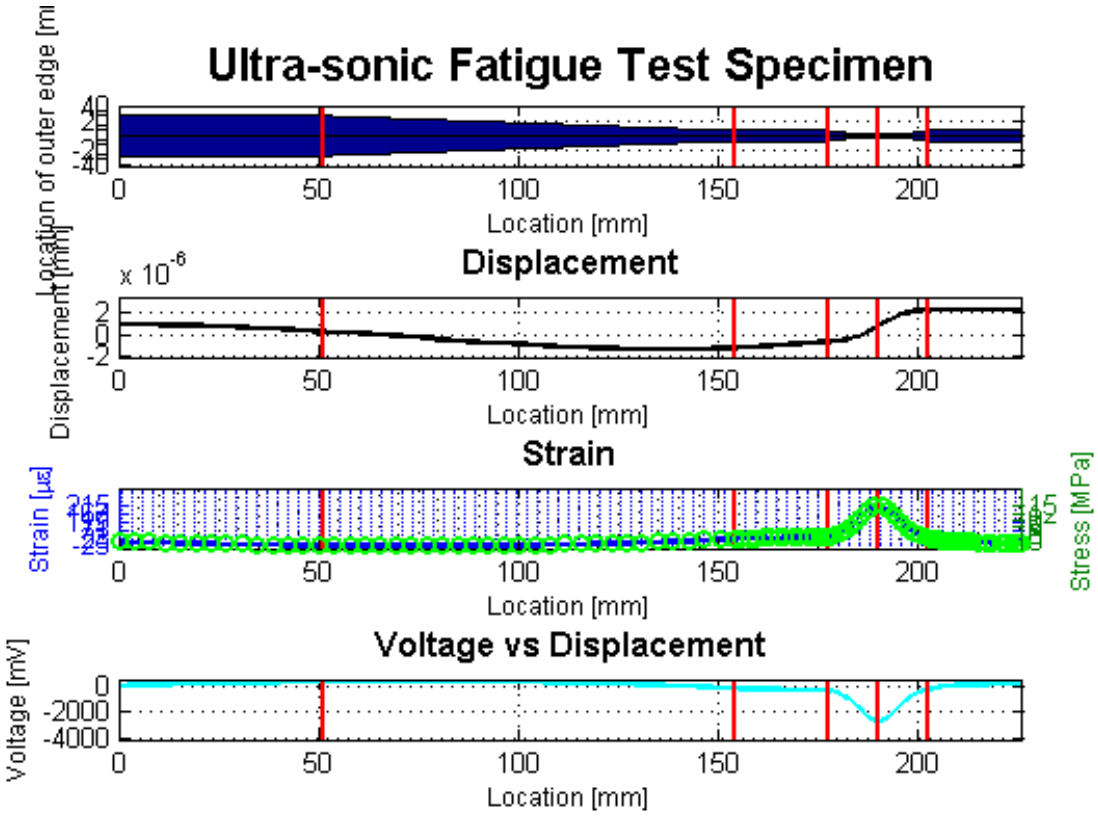
%define max strain
disp('Max strain in microstrain');
max_strain=max(U_32(:,2))

%max strain (max strain should always be in specimen)
disp('Max stress in MPa');
max_stress=Ed(2)*max_strain*10^-6*10^-6 %convert micro strain back to strain covert Pa to MPa

%calculate estimated life using MMPDS
ksiToMPa=6.894759087;%MPa
StressRatio=-1
Seq=max_stress/ksiToMPa*(1-StressRatio)^0.49
disp('Estimate of cycles based on SN data');
Nf=10^(14.86-5.80*log10(Seq))
disp('Time needed at 20 kHz in seconds');
TimeN=Nf/(20000)
if TimeN>60
    disp('Time needed at 20 kHz in minutes');
    TimeN=Nf/(20000*60)
end
if TimeN>60
    disp('Time needed at 20 kHz in hours');
    TimeN=Nf/(20000*60*60)
end
% Create Voltage vs displacement curve data (Note strain is in microstrain)
[x_volt,Volt]=GetVoltageFromStrain(x_region*1000,U_32(:,2)*10^-6,15,0.33,2.1,1,604);
```

```
% Plot the voltage
%plot the upper and lower surfaces with a mirror
%figure(2)
subplot(4,1,4);
plot(x_volt,Volt*1000,'-c','LineWidth',1.5);
hold on;
%Plot verticle lines at step changes
VertLine_x=[L(1);L(1)]*1000;
VertLine_y=[min(Volt*1000);max(Volt*1000)]*fac;
plot(VertLine_x,VertLine_y,'-r','LineWidth',2)
%Plot verticle lines at step changes
VertLine_x=[(L(1)+L(2));(L(1)+L(2))]*1000;
plot(VertLine_x,VertLine_y,'-r','LineWidth',2)
%Plot verticle lines at step changes
VertLine_x=[(L(1)+L(2)+L(3));(L(1)+L(2)+L(3))]*1000;
plot(VertLine_x,VertLine_y,'-r','LineWidth',2)
%plot at centre of specimen
VertLine_x=[(L(1)+L(2)+L(3)+L(4)/2);(L(1)+L(2)+L(3)+L(4)/2)]*1000;
plot(VertLine_x,VertLine_y,'-r','LineWidth',2)
%plot at end of curve on specimen
VertLine_x=[(L(1)+L(2)+L(3)+L(4));(L(1)+L(2)+L(3)+L(4))]*1000;
plot(VertLine_x,VertLine_y,'-r','LineWidth',2)
hold off;
xlabel('Location [mm]','FontSize', 9);
```

```
ylabel('Voltage [mV]','FontSize', 9);
title('Voltage vs Displacement','Fontweight','bold','FontSize', 12);
axis([min(x_volt) max(x_volt) min(Volt*1000)*fac max(Volt*1000)*fac]);
grid on;
```



Get voltage value at a location

```
%get voltage at a x loction
disp('Location in mm to obtain voltage estimate');
xloc=22
%Removes steps
disp('Voltage at above location in mV')
[x,indexLeft,indexRemoved]=unique(x_volt);
V=Volt(1,indexLeft)*1000;
Vloc=interp1(x,V,xloc) %output on mV

disp('Strain at Strain gauge location:')
[x_region,indexLeft,indexRemoved]=unique(x_region);
U=U_32(indexLeft,2);
Sloc=interp1(x_region*1000,U,xloc)
```

G.3 Calibration Code

G.3.1 AL2024 - Calc_TaperedHorn_Spec_GenerateCalibrationCurve_AL2024.m

Brief description

```
%Code to calculate the general equations of Ultrasonic fatigue
%Specimen has:
%               a hyperbolic cosine distribution.
%               a linear horn
%               two cylinder section R3,L4 and R2,L2
% code calculate the displacement at the end and at the strain at the
%strain gauge location for each input and output to a text file for manual
%input to excel from where the anlysis and calibration is completed.
%the code is specific for AL2024 specimen and AL7075 Horn.

clc
clear
close all

%Assume material
%--------------------------------------------------------------------------
```

MATERIAL PROPERTIES

```
%Al7075
rho(1)=2796; %kg/m^3
Ed(1)=71.016*10^9;%Pa
v(1)=0.33;%poissons ration

%Al2024
rho(2)=2767.99; %kg/m^3
Ed(2)=72.395*10^9;%Pa
v(2)=0.33;
%--------------------------------------------------------------------------
```

Inputs

```
%define the A_input constant (Input Displacement)
disp('Input Displacement - A_Input constant')
A_input=[-12e-6:0.2e-6:12e-6]%10 micro meters
%define the A_input constant (Input Displacement)
disp('Input Frequency - f')
%General constant
f=20*10^3;%[Hz]define the frequency of the wave 20 kHz
%define the strain gauge location in mm
StrainGaugeLoc=25;%mm
%--------------------------------------------------------------------------
```

Known Geometry

```
%GEOMETRY PROPERTIES (From pg30 Gigacycle fatigue) ENDURANCE SPECIMEN
%see note book Tab final design:
L=[51;103;23.4;72.0-23.4*2;23.4]*10^-3;

%define radi
R=[54;14;14;5;14]/2*10^-3;
%--------------------------------------------------------------------------
```

Calculate wave frequency and speed

```
%define the radial frequency
w=2*pi*f;

%calculate the wave velocity
disp('Wave Velocity - Ignore poissons effect')
c(1)=sqrt(Ed(1)/rho(1))
c(2)=sqrt(Ed(2)/rho(2))

%Define total length of speciment
Ltot=L(1)+L(2)+L(3)+L(4)+L(5);

%Create a vecotr to creat plots
Stepsize=(Ltot/1000);
y_upper=0;

%define the specific region for the taper
x_range.horn=[0;L(1)+L(2)];
x_range.spec=[L(1)+L(2);Ltot];

%define k as per ref 1 (Giga cycle fatigue Bathius) pg 8/41
k(1)=w/c(1);
k(2)=w/c(2);
%--------------------------------------------------------------------------
```

Solve circle equation

```
%solve the circle equation
options=optimset('Display','iter');
z=[L(1)+L(2)+L(3);L(1)+L(2)+L(3)+L(4)/2;L(1)+L(2)+L(3)+L(4)]
y=[R(3);R(4);R(5)]
x0=[10*10^-3;z(1);y(1)];%r,a,b
%test unit circle at orgin
% z=[-1;0;1]
% y=[0;-1;0]
% x=[1;0;0]
[const,fval]=fsolve(@getCircleConstants,x0,options,z,y)

output(1).A=0;
output(1).midStress=0;
output(1).endDisp=0;
output(1).gaugeStrain=0;
```

```
for ii=1:size(A_input,2)
```

```
    %solve the differential equation of the tapered section
    %define initial conditions of the system as zero disp, zero vel
    u0=[A_input(1,ii);0];%[u_0;u_dot_0]
```

Solve the wave equation for current horn + specimen setup

```
    %define better variable for circle geometry
    CircData=const;
    %calculate the displacement for horn region
    [xhorn,Uhorn]=ode45(@Tapered_Ode_SolverHorn,x_range.horn,u0,options,L,R,k(1));
    %calculate the displacement for specimen region
    u0=[Uhorn(end,1);Uhorn(end,2)*Ed(1)/Ed(2)];
    [xspec,Uspec]=ode45(@Tapered_Ode_SolverSpecimen,x_range.spec,u0,options,L,R,k(2),CircData);

    x_region=[xhorn;xspec];
    U_32=[Uhorn;Uspec];
    lengthHorn=size(xhorn,1);

    strain=[U_32(1:lengthHorn,2);U_32(lengthHorn+1:end,2)];
    stress=Ed(1)*strain;

    %get the input displacement
    output(ii).A=A_input(1,ii)*10^6;
    %get stress at centre
    xloc=L(1)+L(2)+L(3)+L(4)/2;
    method='linear';
    [x_unique,ia,ic]=unique(x_region);
    midStress=interp1(x_unique,stress(ia),xloc,method);
    output(ii).midStress=midStress*10^-6;

    %get displacement at end of horn+specimen
    output(ii).endDisp=U_32(end,1)*10^6;

    %get the strain at the strain gauge location
    xloc=StrainGaugeLoc*10^-3; %conver mm to meters
    method='linear';
    strainAtLoc=interp1(x_unique,strain(ia),xloc,method);
    output(ii).gaugeStrain=strainAtLoc*10^6; %micro strain
```

```
end

fid=fopen('Calibration Data.txt','w');
fprintf(fid,['Point Number' '\t' 'Input Displacement [mu m]' '\t' ...
    'Displacement at End [micrometers]' '\t' 'Stress at Centre [MPa]' ...
    '\t' 'Strain at gaugle location (' num2str(StrainGaugeLoc) 'mm) [microstrain]' '\n']);
for i=1:size(output,2)
    fprintf(fid,[num2str(i) '\t']);
    fprintf(fid,'%g\t',output(i).A);
    fprintf(fid,'%g\t',output(i).endDisp);
    fprintf(fid,'%g\t',output(i).midStress);
    fprintf(fid,'%g\n',output(i).gaugeStrain);
end
fclose(fid);
```

```
display('Code Complete')
```

```
Code Complete
```

G.3.2 AL7075 - Calc_TaperedHorn_Spec_GenerateCalibrationCurve_AL7075.m

Brief description

```
%Code to calculate the general equations of Ultrasonic fatigue
%Specimen has:
%               a hyperbolic cosine distribution.
%               a linear horn
%               two cylinder section R3,L4 and R2,L2
% code calculate the displacement at the end and at the strain at the
%strain gauge location for each input and output to a text file for manual
%input to excel from where the anlysis and calibration is completed.
%the code is specific for AL7075 specimen and AL7075 Horn.

clc
clear
% close all

%define global variables to be used in ode
global k
global R
global L

%Assume material
%--------------------------------------------------------------------------
```

MATERIAL PROPERTIES

```
%Al7075
rho=2796; %kg/m^3
Ed=71.016*10^9;%Pa
v=0.33;%poissons ration
%
%--------------------------------------------------------------------------
```

Inputs

```
%define the A_input constant (Input Displacement)
disp('Input Displacement - A_Input constant')
A_input=[-12e-6:0.2e-6:12e-6]%10 micro meters
%define the A_input constant (Input Displacement)
disp('Input Frequency - f')
%General constant
f=20*10^3%[Hz]define the frequency of the wave 20 kHz
%define the strain gauge location in mm
```

```
StrainGaugeLoc=25;%mm
%--------------------------------------------------------------------------
```

Known Geometry

```
%GEOMETRY PROPERTIES (From pg30 Gigacycle fatigue) ENDURANCE SPECIMEN
%see note book Tab final design:
L=[51;103;22.7;70.7-22.7*2;22.7]*10^-3;

%define radi
R=[54;14;14;5;14]/2*10^-3;
%--------------------------------------------------------------------------
```

Calculate wave frequency and speed

```
%define the radial frequency
w=2*pi*f;

%calculate the wave velocity
disp('Wave Velocity - Ignore poissons effect')
c=sqrt(Ed/rho)


%Define total length of speciment
Ltot=L(1)+L(2)+L(3)+L(4)+L(5);

%Create a vector to creat plots
Stepsize=(Ltot/1000);
y_upper=0;

%define the specific region for the taper
x_range=[0;Ltot];

%define k as per ref 1 (Giga cycle fatigue Bathius) pg 8/41
k=w/c;

%--------------------------------------------------------------------------
```

Solve circle equation

```
%solve the circle equation
options=optimset('Display','iter');
z=[L(1)+L(2)+L(3);L(1)+L(2)+L(3)+L(4)/2;L(1)+L(2)+L(3)+L(4)]
y=[R(3);R(4);R(5)]
x0=[20*10^-3;z(1);y(1)];%r,a,b
%test unit circle at orgin
% z=[-1;0;1]
% y=[0;-1;0]
% x=[1;0;0]
[const,fval]=fsolve(@getCircleConstants,x0,options,z,y)
```

Solve the wave equation for current horn + specimen setup

```
%define better variable for circle geometry
CircData=const;
output(1).A=0;
output(1).midStress=0;
output(1).endDisp=0;
output(1).gaugeStrain=0;

for ii=1:size(A_input,2)
    %solve the differential equation of the tapered section
    %define initial conditions of the system as zero disp, zero vel
    u0=[A_input(1,ii);0];%[u_0;u_dot_0]

    %calculate the displacement under applied for
    [x_region,U_32]=ode45(@Tapered_Ode_Solver,x_range,u0,options,L,R,k,CircData);

    stress=Ed(1)*real(U_32(:,2));

    %get the input displacement
    output(ii).A=A_input(1,ii)*10^6;
    %get stress at centre
    xloc=L(1)+L(2)+L(3)+L(4)/2;
    method='linear';
    %[x_unique,ia,ic]=unique(x_region);
    midStress=interp1(x_region,stress(:),L(1)+L(2)+L(3)+L(4)/2,method);

    output(ii).midStress=midStress*10^-6;
    %get displacement at end of horn+specimen
    output(ii).endDisp=U_32(end,1)*10^6;

    %get the strain at the strain gauge location
    xloc=StrainGaugeLoc*10^-3; %conver mm to meters
    method='linear';
    strainAtLoc=interp1(x_region,real(U_32(:,2)),xloc,method);
    output(ii).gaugeStrain=strainAtLoc*10^6; %micro strain
end

fid=fopen('Calibration Data.txt','w');
fprintf(fid,['Point Number' '\t' 'Input Displacement [mu m]' '\t' ...
    'Displacement at End [micrometers]' '\t' 'Stress at Centre [MPa]' ...
    '\t' 'Strain at gaugle location (' num2str(StrainGaugeLoc) 'mm) [microstrain]' '\n']);
for i=1:size(output,2)
    fprintf(fid,[num2str(i) '\t']);
    fprintf(fid,'%g\t',output(i).A);
    fprintf(fid,'%g\t',output(i).endDisp);
    fprintf(fid,'%g\t',output(i).midStress);
    fprintf(fid,'%g\n',output(i).gaugeStrain);
end
fclose(fid);
display('Code Complete')
```

```
Code Complete
```

G.4 Error Tolerance Code

G.4.1 Calculations_TaperedHorn_Specimen_loopTillEndDispMatches.m

Brief description

```
%Code to calculate the general equations of Ultrasonic fatigue
%Specimen has:
%               a hyperbolic cosine distribution.
%               a linear horn
%               two cylinder section R3,L4 and R2,L2
%
clc
clear
% close all

%define global variables to be used in ode
global k
global R
global L

%Assume material
%--------------------------------------------------------------------------
```

Material Properties

```
%Al7075
rho(1)=2796; %kg/m^3
Ed(1)=(71.016-0)*10^9;%Pa
v(1)=0.33;%poissons ration

%Al2024 (changed for AL7075 specimen)
rho(2)=2767.99; %kg/m^3
Ed(2)=(72.395-0)*10^9;%Pa
v(2)=0.33;
%--------------------------------------------------------------------------
```

Define end displacement

```
readExcelData=false;
%save the plot
savePlot=false;
disp_end=0;
if ~readExcelData
    disp_end=(10-0.0)*10^-6;
else
    path='E:\TJ_McMillan_Masters\Data\TestData\AL2024\H1_S2_13_CSIR16\';
    excelFilename='WeldDataAL2024.xlsx';
    excelDispLoc='b9'; %output location for stress is one row below this
    disp_end=xlsread([path excelFilename],'Results',excelDispLoc);
end
error_tol=0.0001; %error in difference as percentage of disp_end must be less than error
                %abs(disp_end-disp_endCalulated)/disp_end*100 < error_tol
%--------------------------------------------------------------------------
```

Inputs

```
%define bounding range for input displacement
Abnd=[-100,100]*10^-6; %meters
%define the A_input constant (Input Displacement)
disp('Input Frequency - f')
%General constant
f=20*10^3-50;%[Hz]define the frequency of the wave 20 kHz
%--------------------------------------------------------------------------
```

Known Geometry

```
%GEOMETRY PROPERTIES (From pg30 Gigacycle fatigue) ENDURANCE SPECIMEN
%see note book Tab final design:
L=[51;103;23.4;72.0-23.4*2;23.4]*10^-3-0.0*10^-3; %change for AL7075 specimen

%define radi
R=[54;14;14;5;14]/2*10^-3-0.0*10^-3; %change for AL7075 specimen
%--------------------------------------------------------------------------
```

Calcualte wave frequency and speed

```
%define the radial frequency
w=2*pi*f;

%calculate the wave velocity
disp('Wave Velocity - Ignore poissons effect')
c(1)=sqrt(Ed(1)/rho(1))
c(2)=sqrt(Ed(2)/rho(2))

%Define total length of speciment
Ltot=L(1)+L(2)+L(3)+L(4)+L(5);

%Create a vecotr to creat plots
Stepsize=(Ltot/1000);
y_upper=0;

%define the specific region for the taper
x_range.horn=[0;L(1)+L(2)];
x_range.spec=[L(1)+L(2);Ltot];

%define k as per ref 1 (Giga cycle fatigue Bathius) pg 8/41
k(1)=w/c(1);
k(2)=w/c(2);
%--------------------------------------------------------------------------
```

Solve circle equation

```
%solve the circle equation
options=optimset('Display','iter');
z=[L(1)+L(2)+L(3);L(1)+L(2)+L(3)+L(4)/2;L(1)+L(2)+L(3)+L(4)]
y=[R(3);R(4);R(5)]
x0=[20*10^-3;z(1);y(1)];%r,a,b
%test unit circle at orgin
% z=[-1;0;1]
```

```
% y=[0;-1;0]
% x=[1;0;0]
[const,fval]=fsolve(@getCircleConstants,x0,options,z,y)
%define better variable for circle geometry
CircData=const
% (CircData(3)-CircData(1))*1000
```

Loop by changing A_input till end displacement matches

```
%define initail numerical steps size
step_num=10;
Ainc=(Abnd(2)-Abnd(1))/step_num;
A_guess=Abnd(1);
i=1;
j=1;
error_1=1;
while (abs(error_1)> error_tol & i <= step_num+1)
    % Solve the wave equation for current horn + specimen setup

    %define initial conditions of the system as zero disp, zero vel
    u0=[A_guess(i);0];%[u_0;u_dot_0]

    %calculate the displacement for horn region
    [xhorn,Uhorn]=ode45(@Tapered_Ode_SolverHorn,x_range.horn,u0,options,L,R,k(1));
    %calculate the displacement for specimen region
    u0=[Uhorn(end,1);Uhorn(end,2)*Ed(1)/Ed(2)];
    [xspec,Uspec]=ode45(@Tapered_Ode_SolverSpecimen,x_range.spec,u0,options,L,R,k(2),CircData);

    x_region=[xhorn;xspec];
    U_32=[Uhorn;Uspec];
    lengthHorn=size(xhorn,1);

    %get end disp
    disp_endCalulated(i,j)=real(U_32(end,1));
%     i

    %define loop stopping critera
    error_1=(disp_end-disp_endCalulated(i,j))/disp_end;
    error_sign(i)=sign(error_1);
    if i>1
        %if we find new bound
        if(error_sign(i-1) ~=  error_sign(i))
            Abnd=[A_guess(i-1),A_guess(i)];
            Ainc=(Abnd(2)-Abnd(1))/step_num;
            i=1;
            A_guess=0;
            A_guess(i)=Abnd(1);
            j=j+1;
            error_sign=0;
        else
            %get new guess
            A_guess(i+1)=A_guess(i)+Ainc;
            if (A_guess > Abnd(2))
                disp('No solution in defined input displacement range (1)')
                break
            end
```

```
            %increments counter
            i=i+1;
        end
    else
        %get new guess
        A_guess(i+1)=A_guess(i)+Ainc;
        %increments counter
        i=i+1;
    end

    %need to flip U_32
    % U_23=fliplr(U_32')';

end %while
%deifne final input displacement
A_input=(Abnd(2)+Abnd(1))/2 %get mean of the bound
%define initial conditions of the system as zero disp, zero vel
u0=[A_input;0];%[u_0;u_dot_0]
%calculate the displacement for horn region
[xhorn,Uhorn]=ode45(@Tapered_Ode_SolverHorn,x_range.horn,u0,options,L,R,k(1));

%calculate the displacement for specimen region
u0=[Uhorn(end,1);Uhorn(end,2)*Ed(1)/Ed(2)];
[xspec,Uspec]=ode45(@Tapered_Ode_SolverSpecimen,x_range.spec,u0,options,L,R,k(2),CircData);

x_region=[xhorn;xspec];
U_32=[Uhorn;Uspec];
lengthHorn=size(xhorn,1);
```

Plot the geometry

```
%create the x array with the tappered array inbedded
x=0:Stepsize:Ltot;
%~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
%DEFINE SPECIMENT SURFACES
for i=1:size(x,2)
    %L2<abs(x)<=L
    if (abs(x(i))<L(1) & abs(x(i))>=0)
        y_upper(i)=R(1);
        %linear taper section
    elseif (abs(x(i))<(L(1)+L(2)) & abs(x(i))>=L(1))
        a=(R(2)-R(1))/L(2);
        b=R(1)-L(1)*a;
        y_upper(i)=a*x(i)+b;
    elseif (abs(x(i))<(L(1)+L(2)+L(3)) & abs(x(i))>=(L(1)+L(2)))
        y_upper(i)=R(3);
        %circle section
    elseif (abs(x(i))<(L(1)+L(2)+L(3)+L(4)) & abs(x(i))>=(L(1)+L(2)+L(3)))
        r=const(1);
        a=const(2);
        b=const(3);
        y_upper(i)=-sqrt(r^2-(x(i)-a)^2)+b;
    elseif (abs(x(i))<=(Ltot) & abs(x(i))>=(L(1)+L(2)+L(3)+L(4)))
        y_upper(i)=R(5);
    else
        disp('No Y value for current x location:');
        x(i)
```

```
    end
    % sigma(i)=epsilon(i)*Ed;
end

%!!!
```

```
hold on;
%Plot verticle lines at step changes
VertLine_x=[L(1);L(1)]*1000;
VertLine_y=[Umin(1);Umax(1)]*fac;
plot(VertLine_x,VertLine_y,'-r','LineWidth',2);
%Plot verticle lines at step changes
VertLine_x=[(L(1)+L(2));(L(1)+L(2))]*1000;
plot(VertLine_x,VertLine_y,'-r','LineWidth',2);
%Plot verticle lines at step changes
VertLine_x=[(L(1)+L(2)+L(3));(L(1)+L(2)+L(3))]*1000;
plot(VertLine_x,VertLine_y,'-r','LineWidth',2);
%plot at centre of specimen
VertLine_x=[(L(1)+L(2)+L(3)+L(4)/2);(L(1)+L(2)+L(3)+L(4)/2)]*1000;
plot(VertLine_x,VertLine_y,'-r','LineWidth',2);
%plot at end of curve on specimen
VertLine_x=[(L(1)+L(2)+L(3)+L(4));(L(1)+L(2)+L(3)+L(4))]*1000;
plot(VertLine_x,VertLine_y,'-r','LineWidth',2);
hold off;
xlabel('Location [mm]','FontSize', 9);
ylabel('Displacement [mm]','FontSize', 9);
title('Displacement','Fontweight','bold','FontSize', 12);
axis([min(x_region*1000) max(x_region*1000) min(U_32(:,1))*fac max(U_32(:,1))*fac]);
grid on;
```

```
% Plot the strain
%plot the upper and lower surfaces with a mirror
subplot(4,1,3);
U_32(:,2)=U_32(:,2)*10^6; %strain to micro strain
%Just plot strain
%plot(x_region*1000,U_32(:,2),'-g','LineWidth',1.5);
%plot Strain and stress
stress=[Ed(1)*10^-12*U_32(1:lengthHorn,2);Ed(1)*10^-12*U_32(lengthHorn+1:end,2)];
%write data to excel file if required
if readExcelData
    method='linear';
    [x_unique,ia,ic]=unique(x_region);
    midStress=interp1(x_unique,stress(ia),L(1)+L(2)+L(3)+L(4)/2,method);
    xlswrite([path excelFilename],midStress,'Results',[excelDispLoc(1)
num2str(str2num(excelDispLoc(2))+1)]);
end
[hax,hl1,hl2]=plotyy(x_region*1000,U_32(:,2),x_region*1000,stress,'plot','plot');
set(hl1,'LineStyle','-','LineWidth',1.5,'color','b');
set(hl2,'LineStyle','o','LineWidth',1.5,'color','g');
Umax=max(U_32);
Umin=min(U_32);
hold('on');
%Plot verticle lines at step changes
VertLine_x=[L(1);L(1)]*1000;
VertLine_y=[Umin(2);Umax(2)]*fac;
plot(VertLine_x,VertLine_y,'-r','LineWidth',2);
%Plot verticle lines at step changes
VertLine_x=[(L(1)+L(2));(L(1)+L(2))]*1000;
plot(VertLine_x,VertLine_y,'-r','LineWidth',2);
%Plot verticle lines at step changes
VertLine_x=[(L(1)+L(2)+L(3));(L(1)+L(2)+L(3))]*1000;
plot(VertLine_x,VertLine_y,'-r','LineWidth',2);
%plot at centre of specimen
VertLine_x=[(L(1)+L(2)+L(3)+L(4)/2);(L(1)+L(2)+L(3)+L(4)/2)]*1000;
```

```
plot(VertLine_x,VertLine_y,'-r','LineWidth',2);
%plot at end of curve on specimen
VertLine_x=[(L(1)+L(2)+L(3)+L(4));(L(1)+L(2)+L(3)+L(4))]*1000;
plot(VertLine_x,VertLine_y,'-r','LineWidth',2);
hold('off');
xlabel(hax(2),'Location [mm]','FontSize', 9);
ylabel(hax(1),'Strain [\mu\epsilon]','FontSize', 9);
ylabel(hax(2),'Stress [MPa]','FontSize', 9);
title('Strain','Fontweight','bold','FontSize', 12);
axis(hax(1),[min(x_region*1000) max(x_region*1000) floor(min(U_32(:,2))*fac)
ceil(max(U_32(:,2))*fac)]);
axis(hax(2),[min(x_region*1000) max(x_region*1000) floor(min(stress)*fac) ceil(max(stress)*fac)]);
% axis(hax(2),'tight');
NumYgrid=5;% this is the number of y-measurments required
ymin=floor(min(U_32(:,2)));
ymin=round(ymin/NumYgrid)*NumYgrid;
ymax=ceil(max(U_32(:,2)));
ymax=round(ymax/NumYgrid)*NumYgrid;
set(hax(1),'ytick',[ymin:(ymax-ymin)/NumYgrid:ymax]);
ymin=floor(min(stress));
ymin=round(ymin/NumYgrid)*NumYgrid;
ymax=ceil(max(stress))
ymax=round(ymax/NumYgrid)*NumYgrid;
set(hax(2),'ytick',[ymin:(ymax-ymin)/NumYgrid:ymax]);
grid(hax(1),'minor');
grid(hax(1),'on');

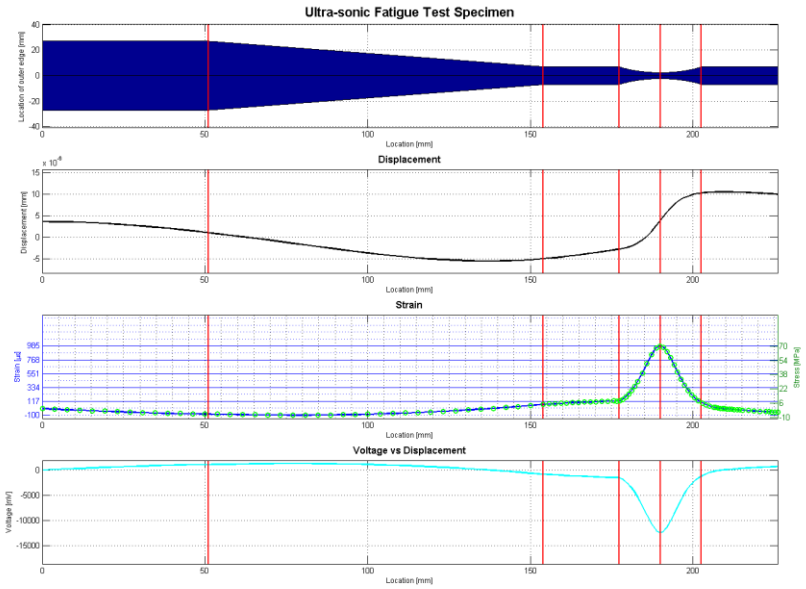
%define max strain
disp('Max strain in microstrain');
max_strain=max(U_32(:,2))

%max strain (max strain should always be in specimen)
disp('Max stress in MPa');
max_stress=Ed(2)*max_strain*10^-6*10^-6 %convert micro strain back to strain covert Pa to MPa

%calculate estimated life using MMPDS
ksiToMPa=6.894759087;%MPa
StressRatio=-1
Seq=max_stress/ksiToMPa*(1-StressRatio)^0.49
disp('Estimate of cycles based on SN data');
Nf=10^(14.86-5.80*log10(Seq))
disp('Time needed at 20 kHz in seconds');
TimeN=Nf/(20000)
if TimeN>60
    disp('Time needed at 20 kHz in minutes');
    TimeN=Nf/(20000*60)
end
if TimeN>60
    disp('Time needed at 20 kHz in hours');
    TimeN=Nf/(20000*60*60)
end
% Create Voltage vs displacement curve data (Note strain is in microstrain)
[x_volt,Volt]=GetVoltageFromStrain(x_region*1000,U_32(:,2)*10^-6,15,0.33,2.1,1,604);
```

```
% Plot the voltage
%plot the upper and lower surfaces with a mirror
%figure(2)
```

```
subplot(4,1,4);
plot(x_volt,Volt*1000,'-c','LineWidth',1.5);
hold on;
%Plot verticle lines at step changes
VertLine_x=[L(1);L(1)]*1000;
VertLine_y=[min(Volt*1000);max(Volt*1000)]*fac;
plot(VertLine_x,VertLine_y,'-r','LineWidth',2)
%Plot verticle lines at step changes
VertLine_x=[(L(1)+L(2));(L(1)+L(2))]*1000;
plot(VertLine_x,VertLine_y,'-r','LineWidth',2)
%Plot verticle lines at step changes
VertLine_x=[(L(1)+L(2)+L(3));(L(1)+L(2)+L(3))]*1000;
plot(VertLine_x,VertLine_y,'-r','LineWidth',2)
%plot at centre of specimen
VertLine_x=[(L(1)+L(2)+L(3)+L(4)/2);(L(1)+L(2)+L(3)+L(4)/2)]*1000;
plot(VertLine_x,VertLine_y,'-r','LineWidth',2)
%plot at end of curve on specimen
VertLine_x=[(L(1)+L(2)+L(3)+L(4));(L(1)+L(2)+L(3)+L(4))]*1000;
plot(VertLine_x,VertLine_y,'-r','LineWidth',2)
hold off;
xlabel('Location [mm]','FontSize', 9);
ylabel('Voltage [mV]','FontSize', 9);
title('Voltage vs Displacement','Fontweight','bold','FontSize', 12);
axis([min(x_volt) max(x_volt) min(Volt*1000)*fac max(Volt*1000)*fac]);
grid on;
%Save plot hh
if (savePlot)
saveas(hh,[path 'Horn&Specimen_stress'],'jpg')
end
```



Get voltage value at a location

```
%get voltage at a x loction
disp('Location in mm to obtain voltage estimate');
xloc=22
%Removes steps
disp('Voltage at above location in mV')
[x,indexLeft,indexRemoved]=unique(x_volt);
V=Volt(1,indexLeft)*1000;
Vloc=interp1(x,V,xloc) %output on mV

disp('Strain at Strain gauge location:')
[x_region,indexLeft,indexRemoved]=unique(x_region);
U=U_32(indexLeft,2);
Sloc=interp1(x_region*1000,U,xloc)

close all
disp(['max_stress is: ', num2str(max_stress)])
```

G.5 FEM Formulation Functions

The FEM solution is comprised of a large amount of functions which can be used depending on what the needs are. That is you can generate a mesh manually or read a mesh in via Nastran and then apply the analysis. This section of the Appendix contains all the functions to perform both options.

G.5.1 InsertElements.m

This function is used to insert the CQUAD and CTRIA lines in the BDF. The code is simply the following:

```
function [status]=InsertElments(fid, Elm, ElmType)
%This function writes in the bdf file
%--------------------------------------------------------------------------
%Created by: TJ McMillan
%Date: 28/10/2013
%Version: 1.0
%Updates:
%1.0    - Creation of code to write to the BDF the element information
%--------------------------------------------------------------------------
   %check the arguments sent to this function
   if nargin ==2
      ElmType='quad4';
   end
   %first determine the element type of the standard default
   if  strcmp(lower(ElmType),lower('TRIA3'))
      Leader='CTRIA3';
      numGrid=3;
   elseif  strcmp(ElmType,lower('TRIA6'))
      Leader='CTRIA6';
      numGrid=6;
   elseif  strcmp(ElmType,lower('QUAD8'))
      Leader='CQUAD8';
      numGrid=8;
   else
      Leader='CQUAD4';
      numGrid=4;
   end


   %print the intial definition of material section
   fprintf(fid,'$ ============================================================\n');
   fprintf(fid,'$ ELEMENTS\n');
   line='';
```

```
    for elmNum=1:size(Elm,2)
        line=[Leader ',' num2str(elmNum) ',' Elm(elmNum).mat(2:end)];
        for i=1:numGrid
            line=[line ',' num2str(Elm(elmNum).connect(i))] ;
            if (i==6 & ~strcmp(leader,'CTRIA6'))
                line=[line ',\n'];
            end
        end
        fprintf(fid, [line ',,\n']);
    end
    fprintf(fid,'$ ================================================================\n');
    status='Completed adding the elements to the BDF';
end
```

G.5.2 InsertGrids.m

```
function [status]=InsertGrids(fid, Elm)
%This function writes in the bdf file
%-------------------------------------------------------------------------
%Created by: TJ McMillan
%Date: 25/10/2013
%Version: 1.0
%Updates:
%1.0    - write the grid points to the bdf file
%-------------------------------------------------------------------------
    %determin the element type
    %first determine the element type of the standard default
    if  strcmp(lower(Elm(1).type),lower('TRIA3'))
        numGrid=3;
    elseif  strcmp(lower(Elm(1).type),lower('TRIA6'))
        numGrid=6;
    elseif  strcmp(lower(Elm(1).type),lower('QUAD8'))
        numGrid=8;
    else
        numGrid=4;
    end
    %get the grids from the elements
    elmNum=1;
    %initialise Grid to null
    Grid=[];
    while (elmNum <= size(Elm,2))
      for  i=1:numGrid
        %check if value has not already been written
        if i==1
          %write value to Grid
          Grid(Elm(elmNum).connect(i)).Xloc=Elm(elmNum).Grid(1,1);
          Grid(Elm(elmNum).connect(i)).Yloc=Elm(elmNum).Grid(1,2);
        elseif Elm(elmNum).connect(i) <= size(Grid,2)
          %only write if value is not []
%           if isempty(Grid(Elm(elmNum).connect(i)).Xloc)
            %write value to Grid
            Grid(Elm(elmNum).connect(i)).Xloc=Elm(elmNum).Grid(1,1);
            Grid(Elm(elmNum).connect(i)).Yloc=Elm(elmNum).Grid(1,2);
%           end
        else
          %write value to Grid
          Grid(Elm(elmNum).connect(i)).Xloc=Elm(elmNum).Grid(1,1);
          Grid(Elm(elmNum).connect(i)).Yloc=Elm(elmNum).Grid(1,2);
        end
      end
      elmNum=elmNum+1
    end
    %print the initial definition of material section
    fprintf(fid,'$ ================================================================\n');
    fprintf(fid,'$ GRIDS\n');
    for pt=1:size(Grid,2)
        fprintf(fid,['GRID,' num2str(pt) ',,' num2str(Grid(pt).Xloc) ',' num2str(Grid(pt).Yloc) ',0.0,,,,\n']);
    end
    fprintf(fid,'$ ================================================================\n');
    status='Insertion of grids completed'

end
```

G.5.3 InsertParameters.m

```
function [status]=InsertParameters(fid, subcase, param)
%This function write the parameters to the passed file id
%------------------------------------------------------------------------
%Created by: TJ McMillan
%Date: 24/10/2013
%Version: 1.0
%Updates:
%1.0    - Creation of bdf writer
%------------------------------------------------------------------------
status='Parameters failed to write';

if nargin==1
    paramPrint=0; %default
elseif nargin==2 & ~subcase==''
    paramPrint=1;%write subcases but default parameters
elseif nargin==2 & ~subcase=='' & ~param==''
    paramPrint=1;%write subcases and parameters
else
    paramPrint=0; %default
end

if paramPrint == 0%hard code the parameter section to default below
    fprintf(fid,'$ NASTRAN input file created by the TJ McMillan code InserParametrs\n');
    fprintf(fid,['$ On ' datestr(now,1) '\n']);
    fprintf(fid,'SOL 103\n');
%     fprintf(fid,'$DIAG 14\n');
%     fprintf(fid,'$ Direct Text Input for Executive Control\n');
%     fprintf(fid,'COMPILE PHASE1DR\n');
%     fprintf(fid,'ALTER "CALL.*PHASE1B" $ AFTER CALL PHASE1B\n');
%     fprintf(fid,'$ print matrix kaa\n');
%     fprintf(fid,'MATPRT KAA/ $\n');
%     fprintf(fid,'$$MATPRN KAA/ $\n');
%     fprintf(fid,'$ print matrix maa\n');
%     fprintf(fid,'MATPRT MAA/ $\n');
%     fprintf(fid,'$$MATPRN MAA/ $\n');
    fprintf(fid,'$$$\n');
    fprintf(fid,'CEND\n');
    fprintf(fid,'$ Direct Text Input for Global Case Control Data\n');
    fprintf(fid,['TITLE = MSC.Nastran job created on ' date ' at ' datestr(rem(now,1)) '\n']);
    fprintf(fid,'ECHO = NONE\n');
    fprintf(fid,'RESVEC = NO\n');
    fprintf(fid,'SUBCASE 1\n');
    fprintf(fid,'$ Subcase name : Run\n');
    fprintf(fid,'    SUBTITLE=Default\n');
    fprintf(fid,'    METHOD = 1\n');
    fprintf(fid,'    SPC = 2\n');
    fprintf(fid,'    VECTOR(SORT1,REAL)=ALL\n');
    fprintf(fid,'    SPCFORCES(SORT1,REAL)=ALL\n');
    fprintf(fid,'$=========================================================\n');
    fprintf(fid,'$$$\n');
    fprintf(fid,'BEGIN BULK\n');
    fprintf(fid,'$=========================================================\n');
    fprintf(fid,'$ PARAMETERS\n');
    fprintf(fid,'$=========================================================\n');
    fprintf(fid,'$ Direct Text Input for Bulk Data\n');
    fprintf(fid,'PARAM    POST    0\n');
    fprintf(fid,'PARAM    AUTOSPC NO\n');
    fprintf(fid,'PARAM   COUPMASS 1\n');
    fprintf(fid,'PARAM    WTMASS 1.\n');
    fprintf(fid,'PARAM   PRTMAXIM YES\n');
    fprintf('EIGRL    1                    10      0\n');
    status='Parameters and command section created';
elseif paramPrint == 1 %hard code the parameter section to default below
    fprintf(fid,'$ NASTRAN input file created by the TJ McMillan code InserParametrs\n');
    fprintf(fid,['$ On ' datestr(now,1) '\n']);
    fprintf(fid,'SOL 103\n');
%     fprintf(fid,'$DIAG 14\n');
%     fprintf(fid,'$ Direct Text Input for Executive Control\n');
%     fprintf(fid,'COMPILE PHASE1DR\n');
%     fprintf(fid,'ALTER "CALL.*PHASE1B" $ AFTER CALL PHASE1B\n');
%     fprintf(fid,'$ print matrix kaa\n');
%     fprintf(fid,'MATPRT KAA/ $\n');
```

```
%     fprintf(fid,'$$MATPRN KAA/ $\n');
%     fprintf(fid,'$ print matrix maa\n');
%     fprintf(fid,'MATPRT MAA/ $\n');
%     fprintf(fid,'$$MATPRN MAA/ $\n');
    fprintf(fid,'$$$\n');
    fprintf(fid,'CEND\n');
    fprintf(fid,'$ Direct Text Input for Global Case Control Data\n');
    fprintf(fid,['TITLE = MSC.Nastran job created on ' date ' at ' datestr(rem(now,1)) '\n']);
    fprintf(fid,'ECHO = NONE\n');
    fprintf(fid,'RESVEC = NO\n');
    fprintf(fid,'SUBCASE 1\n');
    fprintf(fid,['$ Subcase name : ' subcase '\n']);
    fprintf(fid,['   SUBTITLE=' subcase '\n']);
    fprintf(fid,'    METHOD = 1\n');
    fprintf(fid,'    SPC = 2\n');
    fprintf(fid,'    VECTOR(SORT1,REAL)=ALL\n');
    fprintf(fid,'    SPCFORCES(SORT1,REAL)=ALL\n');
    fprintf(fid,'BEGIN BULK\n');
    fprintf(fid,'$ Direct Text Input for Bulk Data\n');
    fprintf(fid,'PARAM    POST    0\n');
    fprintf(fid,'PARAM    AUTOSPC NO\n');
    fprintf(fid,'PARAM   COUPMASS 1\n');
    fprintf(fid,'PARAM    WTMASS 1.\n');
    fprintf(fid,'PARAM   PRTMAXIM YES\n');
    fprintf('EIGRL    1                    10      0\n');
    status='Parameters and command section created';
else %write parameters as received from form
    fprintf(fid,'$ NASTRAN input file created by the TJ McMillan code InserParametrs\n');
    fprintf(fid,['$ On ' datestr(now,1) '\n']);
    fprintf(fid,'SOL 103\n');
    fprintf(fid,'$$$\n');
    fprintf(fid,'CEND\n');
    fprintf(fid,'$ Direct Text Input for Global Case Control Data\n');
    fprintf(fid,['TITLE = MSC.Nastran job created on ' date ' at ' datestr(rem(now,1)) '\n']);
    fprintf(fid,'ECHO = NONE\n');
    fprintf(fid,'RESVEC = NO\n');
    fprintf(fid,'SUBCASE 1\n');
    fprintf(fid,['$ Subcase name : ' subcase '\n']);
    fprintf(fid,['   SUBTITLE=' subcase '\n']);
    fprintf(fid,'    METHOD = 1\n');
    fprintf(fid,'    SPC = 2\n');
    fprintf(fid,'    VECTOR(SORT1,REAL)=ALL\n');
    fprintf(fid,'    SPCFORCES(SORT1,REAL)=ALL\n');
    fprintf(fid,'BEGIN BULK\n');
    fprintf(fid,'$ Direct Text Input for Bulk Data\n');
    for i = 1: size(Param,1)
       fprintf(fid,['PARAM,' Param(1).name ',' Param(1).vlaue '\n']);
    end
    fprintf(fid,'EIGRL    1                    10      0\n');
    status='Parameters and command section created';
end
end
```

G.5.4 InsetPropAndMats.m

```
function [status]=InsertPropAndMats(fid, mat, propMat,thickness)
%This function writes the properties and mat id of the mesh
%-------------------------------------------------------------------------
%Created by: TJ McMillan
%Date: 25/10/2013
%Version: 1.0
%Updates:
%1.0    - Creation of bdf writer
%thickness needs to be the corresponding thickness to each mat
%propMat   need to be the material number corresponding to the given
%         thickness
%mat      is a strucutre for 1 to number of materials with E, rho, type and
%         epsilon as built is variables
%-------------------------------------------------------------------------
%print the intial definition of material section
fprintf(fid,'$ ============================================================\n');
fprintf(fid,'$ Referenced Material Records\n');
fprintf(fid,['$ Description of Material : Date: ' date ' Time:' datestr(rem(now,1)) '\n']);
for numMat=1:size(mat,1)
   if mat(numMat).type=='m'
      materialDef='Default material';
   else
      materialDef='Fastener material';
   end
   %print material number and name
   fprintf(fid,['$ Material Record :' num2str(numMat) ' Material: ' mat.name ' materialDef\n']);
   %print mat1 detials
   fprintf(fid,['MAT1,' num2str(numMat) ',' num2str(mat(numMat).E) ',' num2str(mat(numMat).epsilon) ',' num2str(mat(numMat).rho) '.\n']);
end
fprintf(fid,'$ Elements and Element Properties for region\n');
%print the pshell detials
for numMat=1:size(mat,1)
   fprintf(fid,['PSHELL,' num2str(numMat) ',' num2str(propMat(numMat)) ',' num2str(thickness(numMat)) '\n']);
end
fprintf(fid,'$ ============================================================\n');
status = ' Completed inserting Material and properties';
end
```

G.5.5 WriteBDF.m

```
function [status]=WriteBDF(path,filename,mesh,mat, subcase, param)
%This function writes a new bdf file
%-------------------------------------------------------------------------
%Created by: TJ McMillan
%Date: 28/10/2013
%Version: 1.0
%Updates:
%1.0    - Creation of code to write the BDF
%Variables
%   subcase     - name of subcase
%   param       - is a structure of the parameter names (.name) and values
%                 (.value)
%   -Mat
%       Mat.E       - youngs modulus
%       Mat.rho     - density
%       Mat.epsilon - poissons ratio
%       Mat.name    - name of material
%       Mat.secname - f1 - fastener section 1 or m1 - general material
%   path        - path name
%   filename    - name of the new BDF
%-------------------------------------------------------------------------
%check number of argument
if nargin == 4
    subcase='';
    param='';
elseif narge == 5
    param='';
end

%set the default status
status='Error';
 %open a new file for writing
%open file
display('****************************************');
display('Began writing BDF file');
display(['Location: ' path]);
display(['File name: ' filename]);
fid = fopen(filename,'w');

%insert the command section and parameters
status=InsertParameters(fid, subcase, param)

%insert the materials and properties
status=InsertPropAndMats(fid, mat, 1,1.0)

%insert the elements
status=InsertElments(fid, mesh.Elm, mesh.Elm(1).type)

%insert the grids
status=InsertGrids(fid, mesh.Elm)

%TODO
%still need to add boundary conditions

%TODO
%still need to add possible other reference frames
%if no reference co-ordinates then only the global is maintained

%end the file
fprintf(fid,'ENDDATA\n');

%check final status
if strcmp(status,'Error')
    status='error';
else
    status='BDF created succesfully';
end
display('FEM write BDF code completed');
%close the file
fclose(fid);
end
```

G.5.6 RunNastran.m

```
function [status]=runNastran(FilePath,FileName,Nastranpath,Exe_filename)
%This function runs the Nastran on the filename and waits for the results
%--------------------------------------------------------------------------
%Created by: TJ McMillan
%Date: 28/10/2013
%Version: 1.0
%Updates:
%1.0    - Creation of code run nastran within matlab
%--------------------------------------------------------------------------
close all
clear('java');
clear('global');
fclose('all');

if nargin==2
    %set default Nastran Path
    Nastranpath='C:\MSC.Software\MSC.Nastran2005r3\bin\';
    Exe_filename='nastran.exe';
end
system([Nastranpath Exe_filename ' ' FilePath FileName])

%check for fatal errors in .f06
[path,name,ext,versn]=fileparts([FilePath FileName]);

display('**********check for Fatal Errors************************');
%get the fatal errors for output
errorFound=false;
fid=fopen([path '\' name '.f06']);
fseek(fid, 0, 'eof');
filesize = ftell(fid);
fclose(fid);
fid=fopen([path '\' name '.f06']);
while (~(filesize == ftell(fid)))
    tline=fgetl(fid);
    %only break the loop when there is no more data to be read
    if ~ischar(tline) ,disp('Break1'), break, end;
    if strfind(lower(tline),'fatal')
        tline
        tline=fgetl(fid);
        tline
        errorFound=true;
    end
end
fclose(fid);
%++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
%Delete the dball, f04 and master files
delete([path '\' name '.dball']);
delete([path '\' name '.master']);
delete([path '\' name '.f04']);
%--------------------------------------------------------------------------
%OUTPUT Status if matlab run
if errorFound
    status='Matlab run complete but file includes FATAL errors, see above';
else
    status='Matlab run complete. No FATAL errors';
end
end
```

G.5.7 ReadInpType.m

```
function [mesh,mat]=ReadInpFileType(fFilename)
%This function reads in a text formatted "INP" to create the model
%The format is based on a set order of headings which give node numbers and grid point
%Typical headings are:{note: % is not from txt file}
%   Number_of_nodes
%   Nodal_coordinates (radial,along axis)=(r,z)
%   Number_of_elements
%   Plane_stress_or_strain
%   Element Type (tria3, tria6 quad4 quad8)
%   Element_connectivity (elm (nodes 1 - n) mat#)
%   Number of materials
```

```
%   Material_properties (E,v,rho)
%   Number_of_prescribed_displacements
%   Prescribed_displacements (node dof disp)
%   Number_of_nodal_loads
%   Nodal_loads
% file type is *.inp for input files
%-------------------------------------------------------------------------
%Created by: TJ McMillan
%Date: 15/05/2013
%Version: 1.0
%Updates:
%1.0    - Creation of file reader
%-------------------------------------------------------------------------
%open file
fid = fopen(fFilename,'r');

tic;
disp('Begin to read input file on type .inp');

% Read number of nodes
dummy = fgetl(fid);%get "Number_of_nodes"
nnodes = fscanf(fid,'%d \n',1);

%Read nodal coordinates
dummy = fgetl(fid);% get "Nodal_coordinates (radial,along axis)=(r,z)"
ndcoor = fscanf(fid,'%f \n',[3,nnodes])';%col x rows

% Store node numbers in nodes
nodes = round(ndcoor(:,1));

% Store coordinates in coord (should all be metres)
coor = ndcoor(:,2:3);

% Read number of elements
dummy = fgetl(fid);%get "Number_of_elements "
nelem = fscanf(fid,'%5d \n',1);

% Read if elements are plane stress or plain strain"
dummy = fgetl(fid);%get "Plane_stress_or_strain"
plane = fscanf(fid,'%1d \n',1);

% Read element type"
dummy = fgetl(fid);%get "Element Type (tria3, tria6 quad4 quad8)"
elmtype = fgetl(fid);

%remove white space
elmtype = deblank(elmtype);

%read the type of element
if (strcmp(lower(elmtype),'tria3'))
    elmVert=3;
elseif (strcmp(lower(elmtype),'quad4'))
    elmVert=4;
elseif (strcmp(lower(elmtype),'quad8'))
    %NOTE:  node order should be as per page 212 in Cook for each element
    elmVert=8;
elseif (strcmp(lower(elmtype),'tria6'))
    elmVert=6;
else
    warning('elmtype not recognised in ReadInpFileType.m')
    return %exit function
end

% Read element number and element connectivity
dummy = fgetl(fid);%get "Element_connectivity (elm (nodes 1 - n) mat#)"
elnodes = fscanf(fid,'%f \n',[elmVert+3,nelem])';

%read number of materials
dummy = fgetl(fid);%get "Number of materials"
nMat = fscanf(fid,'%1d \n',1);

% Read material constants
dummy = fgetl(fid);%get "Material_properties (E,v,rho)"
mat = fscanf(fid,'%f\n',[4, nMat]);%[matNum E v rho]
mat=mat';
```

```
% Read number of prescribed displacements
dummy = fgetl(fid);%get "Number_of_prescribed_displacements "
ndispl = fscanf(fid,'%5d \n',1);

% Read prescribed displacements;
dummy = fgetl(fid);%get "Prescribed_displacements (node dof disp)"
displ = fscanf(fid,'%f \n',[3,ndispl])';
displ = sortrows(displ);

% Read number of nodal loads
dummy = fgetl(fid);%get "Number_of_nodal_loads "
ncload = fscanf(fid,'%5d \n',1);

% Read nodal loads
dummy = fgetl(fid);%get "Nodal_loads"
cload = fscanf(fid,'%f \n',[3,ncload])';
fclose(fid);

%write mesh data to mesh
mesh.Coord=coor;
mesh.NumNodes=nnodes;
mesh.ElmType=elmtype;
mesh.NumElm=nelem;
mesh.NumDispl=ndispl;
mesh.Displ=displ;
mesh.NumCload=ncload;
mesh.Nodes=nodes;
mesh.Cload=cload;
mesh.Plane=plane;
for i=1:nelem
    mesh.Elm(i).ElmNodes=elnodes(i,2:end-2);
    mesh.Elm(i).Mat=elnodes(i,end-1);
    mesh.Elm(i).t=elnodes(i,end);
end
finish = toc;
disp(['Done reading input file         : ',num2str(finish),' seconds'])

tic;
end
```

G.5.8 ReadBDFFileType.m

```
function [mesh,mat]=ReadBDFFileType(fPath,fFilename)
%This function reads in a BDF file created via patran
%The format is based on the typical BDF format
%-------------------------------------------------------------------------
%Created by: TJ McMillan
%Date: 24/10/2013
%Version: 1.0
%Updates:
%1.0    - Creation of file reader
%-------------------------------------------------------------------------
%*************************************************************************
%Get the length of BDF
%*************************************************************************
fid = fopen([fPath  fFilename]);
fseek(fid, 0, 'eof');
filesize = ftell(fid);
fclose(fid);
 %----
%loop through the file and read in the data
%----
%open the BDF for reading
fid = fopen([fPath  fFilename],'r');
 %initialise the variables mesh and mat
mesh=[];
mat=[];
 %define a line counter
line=0;
 %define counter for each variable
i_para=1; %counter for parameters
 while (~(filesize == ftell(fid)))
    %get first line
    tline=fgetl(fid);
    line=line+1;
    %check if line starts with a comment symbol - $
    if tline(1)=='$'
       comment=true;
    else
       comment=false;
    end
    if ~comment
       %Two possible BDF lines exist: 1 with commas 2 without commas
       %check if line contains a comma
       LineIsCommaFormat=[];
       LineIsCommaFormat=strfind(tline,',');
       if ~isempty(LineIsCommaFormat) & size(LineIsCommaFormat,2) > 1
          [lineData,status]=ReadBDFLineCommaFormat(tline);
       else
          [lineData,status]=ReadBDFLine8charFormat(tline);
          if strcmp(status,'failed')
             [lineData,status]=ReadBDFLine16charFormat(tline);
              %if still not working this exit routine with error
              if strcmp(status,'failed')
                 error('Failed to read line in BDF, no regonised format')
              end
          end
       end
       %store comment in solution, parameter, grid, quad, mat, pcomp
       if strcmp(lower(lineData(1).str),'param')
          Param(i_para).Name= lineData(2).str;
          Param(i_para).Value= lineData(3).str;
          i_para=i_para+1;
       elseif strcmp(lower(lineData(1).str),'cquad4')
          i_elm=str2num(lineData(2).str);
          mesh.Elm(i_elm).mat=str2num(lineData(3).str);
          mesh.Elm(i_elm).type='cquad4';
          array_connect=[];
          for i=1:4
             array_connect(1,i)=str2num(lineData(i+3).str);
          end
          mesh.Elm(i_elm).connect=array_connect;
          mesh.Elm(i_elm).SubSetNum=0;%not sure which section element belongs too
       elseif strcmp(lower(lineData(1).str),'ctria3')
          i_elm=str2num(lineData(2).str);
```

```
        mesh.Elm(i_elm).mat=str2num(lineData(2).str);
        mesh.Elm(i_elm).type='ctria3';
        array_connect=[];
        for i=1:3
          array_connect(1,i)=str2num(lineData(i+2).str);
        end
        mesh.Elm(i_elm).connect=array_connect;
        mesh.Elm(i_elm).SubSetNum=0;%not sure which section element belongs too
      elseif strcmp(lower(lineData(1).str),'grid')
        i_node=str2num(lineData(2).str);
        mesh.X(i_node,1)= GetNumberOnly(lineData(4).str)
        mesh.Y(i_node,1)= GetNumberOnly(lineData(5).str)
%         i_elm=str2num(lineData(2).str);
%         mesh.Elm(i_elm).Grid(:,1)=mesh.X(i_node);
%          mesh.Elm(i_elm).Grid(:,2)=mesh.Y(i_node);
      elseif strcmp(lower(lineData(1).str),'mat1')
       i_mat=str2num(lineData(2).str);
       %replace + with e+
       test_num=lineData(3).str;
       test_num = strrep(test_num, '+', 'e+');
       mat(i_mat).E=str2num(test_num);
       mat(i_mat).epsilon=str2num(lineData(5).str);
       mat(i_mat).rho=str2num(lineData(6).str);
      elseif strcmp(lower(lineData(1).str),'pshell')
        i_mat=str2num(lineData(2).str);
        mat(i_mat).thickness=str2num(lineData(3).str);
      end
    end
 end % end while (~(filesize == ftell(fid)))

 %generate grid part of mesh using node values
 for elm_num=1:size(mesh.Elm,2)
   for j=1:size(mesh.Elm(elm_num).connect,2)
     mesh.Elm(elm_num).Grid(j,1)=mesh.X(mesh.Elm(elm_num).connect(1,j));
     mesh.Elm(elm_num).Grid(j,2)=mesh.Y(mesh.Elm(elm_num).connect(1,j));
   end
 end
 %close file
 fclose(fid);

end
```

G.5.9 GetNumberOnly.m

This code is used to read in the number from the Nastran BDF and create the number in Matlab.

```
%function to check sign
function [num]=GetNumberOnly(fStr)
    newStr='';
    %first check if there is a e
    if isempty(strfind(lower(fStr),'e'))
        if ~isempty(strfind(fStr,'+'))
           Loc = strfind(fStr, '+');
            for i=1:size(Loc)
                if ~(Loc(i) == 1)
                    newStr=strrep(fStr, '+', 'e+');
                end
            end
        elseif ~isempty(strfind(fStr,'-'))
           Loc = strfind(fStr, '-');
            for i=1:size(Loc)
                if ~(Loc(i) == 1)
                    newStr=strrep(fStr, '-', 'e-');
                end
            end
        else
            newStr=fStr;
        end
    else
        newStr=fStr;
    end
    num=str2num(newStr);
end
```

G.5.10 ReadBDFLineCommaFormat.m

Depending on the type of tline the correct corresponding function needs to be called to decipher the information. This function reads a line in comma format and splits the data into LineData together with a status if the line was analysed correctly.

```
function [LineData,status]=ReadBDFLineCommaFormat(tline)
%This function reads a BDF line in comma format and returns the data in a
%structure form of strings
% tline is the line read from the file
% LineData is the return data in an structure form of strings
%   e.g: LineData(1).str=....
%--------------------------------------------------------------------------
%Created by: TJ McMillan
%Date: 30/10/2013
%Version: 1.0
%Updates:
%1.0    - Creation of line reader
%--------------------------------------------------------------------------
%initialise the staus to failed
status='failed';

%define the first string
startblock=1;
endblock=0;

%initialise string
string='';
count=1;

%loop through the line
for i=1:length(tline)
    if (tline(i) == ',' | i==length(tline) ) %incase there is no comma at the end
        if i==length(tline)
            endblock=i;
        else
            endblock=i-1;
        end
        string=tline(startblock:endblock);
        %trim the white space from the string
        string=strtrim(string);
        %define this string to LineData
        LineData(count).str=string;
        %increment counter
        count=count+1;
        startblock=i+1;
    end
end

%modify status if the line data obtained
if (size(LineData,1)>0)
    status='Line split using comma seperation';
end
end
```

G.5.11 ReadBDFLine8CharFormat.m

This function reads a line space in 8 characters format and splits the data into LineData together with a status if the line was analysed correctly.

```
function [LineData, status]=ReadBDFLine8charFormat(tline)
%This function reads a BDF line in 16 format and returns the data in a
%structure form of strings
% tline is the line read from the file
% LineData is the return data in an structure form of strings
%   e.g: LineData(1).str=....
%--------------------------------------------------------------------------
%Created by: TJ McMillan
%Date: 30/10/2013
%Version: 1.0
%Updates:
%1.0    - Creation of line reader
%--------------------------------------------------------------------------
%initialise loop control variable
endofLine=false;

%initial counter
i=1;
 %define the first string
 startblock=(i-1)*8+1;
 endblock=startblock+7;

%initialise string
string='';

%loop through the line
while (~endofLine & startblock<length(tline))
    if (endblock >= length(tline))
        endblock=length(tline);
        endofLine=true;
    end
    string=tline(startblock:endblock);
    %trim the white space from the string
    string=strtrim(string);
    %check if still have space split via spaces
    CharToSplitBy=' ';
    splitstr = SplitStringIntoArray(string,CharToSplitBy,true);
    for j=1:size(splitstr,2)
        %define this string to LineData
        LineData(i+j-1).str=splitstr(j).str;
    end
    %increment counter
    i=i+j;
    %define the first string
    startblock=(i-1)*8+1;
    endblock=startblock+7;
end

status='Line split using 8 char spacing';
end
```

G.5.12 ReadBDFLine16CharFormat.m

This function read a line space in 16 characters format and splits the data into LineData together with a status if the line was analysed correctly.

```
function [LineData, status]=ReadBDFLine16charFormat(tline)
%This function reads a BDF line in 16 format and returns the data in a
%structure form of strings
% tline is the line read from the file
% LineData is the return data in an structure form of strings
%   e.g.: LineData(1).str=....
%-------------------------------------------------------------------------
%Created by: TJ McMillan
%Date: 12/11/2013
%Version: 1.0
%Updates:
%1.0    - Creation of line reader
%-------------------------------------------------------------------------
%initialise loop control variable
endofLine=false;

%initial counter
i=1;
 %define the first string
 startblock=(i-1)*16+1;
 endblock=startblock+7;

%initialise string
string='';

%loop through the line
while (~endofLine & startblock<length(tline))
    if (endblock >= length(tline))
        endblock=length(tline);
        endofLine=true;
    end
    string=tline(startblock:endblock);
    %trim the white space from the string
    string=strtrim(string);
    %check if still have space split via spaces
    CharToSplitBy=' ';
    splitstr = SplitStringIntoArray(string,CharToSplitBy,true);
    for j=1:size(splitstr,2)
        %define this string to LineData
        LineData(i+j-1).str=splitstr(j).str;
    end
    %increment counter
    i=i+j;
    %define the first string
    startblock=(i-1)*16+1;
    endblock=startblock+7;
end

status='Line split using 16 char spacing';
end
```

G.5.13 GetMeshArbShape.m (version 2.0)

```
function [X,Y,Elm]=GetMeshArbShape_v2(coord,L,numElmV,CurveTypeTop,CurveTypeBot,PlusOne,matType,elmType)
%This function calculates the node points for an arb shaped region
%flat at the bottom and has a curve for the top surface
%node are equivalently spaced where possible
%coord
%   - 4 end vertices labelled in counter clockwise direction
%L
%   - Length of the base of the section
%numElmV
%   - number of elements in the vertical direction between coord 1 and 4
%numElmH
%   - number of elements in the horizontal direction between coord 1 and 2
%Outputs
%X - x-coordinated grid points
%Y - y - coordinated grid points
%Elm
%   Elm.conect  -elm connectivity rows elm number cols are 3 points tri or 4
%   pts quad
%   Elm.mat     - elm material
%--------------------------------------------------------------------------
%Created by: TJ McMillan
%Date: 14/03/2013
%Version: 1.0
%Updates:
%1.0    - Creation of file reader
%2.0    - removed 'quad' and 'tris' and replaced with 'quad4', 'tria3' etc
%       - replaced Grid1 etc with Grid(1,1) ....
%       - fixed quad4 connection
%       - fixed tria3 connections
%--------------------------------------------------------------------------
%check number of vars
if nargin ==3
    CurveTypeTop='str';
    CurveTypeBot='';
    PlusOne=0;
    matType='m1';
    elmType='tria3';
elseif nargin ==4;
    CurveTypeBot='';
    PlusOne=0;
    matType='m1';
    elmType='tria3';
elseif nargin ==5;
    PlusOne=0;
    matType='m1';
    elmType='tria3';
elseif nargin ==6;
    matType='m1';
    elmType='tria3';
elseif nargin ==7;
    elmType='tria3';
end
%get the initial element size in y-direction
StepY=(coord(4,2)-coord(1,2))/numElmV;
%
% %define the number of horizontal elements
% numElmH=round(L/StepY);

%defined loop terminator
EndOfSec=false;
%init stepX
StepX=0;
col=1;
while (col<1000 & ~EndOfSec) %not more than 1000 elms per section
    %get the std step in x-direction
    %need to taper stepX down as elements shrink to keep taper tight
    Lleft=L-sum(StepX);
    if (Lleft > 0)
        StepX(col)=Lleft/round(Lleft/StepY);%try keep elements square
    else
        StepX(col)=0;
    end
```

```
    %define the x coordinate of each elm
    x=coord(1,1)+sum(StepX(1:col-1));
    %check if its the end of the section
    if (x>coord(2,1))
       EndOfSec=true;
       x=coord(2,1);
       StepX(col)=Lleft;
    elseif (x==coord(2,1))
       EndOfSec=true;
       StepX(col)=StepX(col-1);
    end
    %for step 1 should have
    %StepY=(coord(4,2)-coord(1,2))/numElm;
    %else use the curve
    if strcmp(lower(CurveTypeTop),'str')
        m=(coord(4,2)-coord(3,2))/(coord(4,1)-coord(3,1));
        c=coord(4,2);
        ylow=0;
        yup=0;
        if strcmp(lower(CurveTypeBot),'')
            yup=m*x+c;
            ylow=0;
        elseif strcmp(lower(CurveTypeBot),'str')
            mb=(coord(1,2)-coord(2,2))/(coord(1,1)-coord(2,1));
            cb=coord(1,2);
            yup=m*x+c;
            ylow=mb*x+cb;
        end
        %get the spaced vertical Colum of y data
        [X(:,col),Y(:,col),StepY]=getVerticleArray(numElmV,x,ylow,yup,PlusOne);
    elseif strcmp(lower(CurveTypeTop),'cosh')
        R1=(coord(3,2)-coord(2,2));%generally small side on the right
        R2=(coord(4,2)-coord(1,2));%generally large side on left
        %check if profile left to right
        R=0;
        if R1<R2
            alpha=1/L*acosh(R2/R1);
            x2=L-x+coord(1,1);%as the equation works big radii to small radii see pg8/41 ref1
            R=R1;
        else
           alpha=1/L*acosh(R1/R2);
            x2=x-coord(1,1);
            R=R2;
        end
        ylow=0;
        yup=0;
        if strcmp(lower(CurveTypeBot),'')
            yup=R*cosh(alpha*x2);
            ylow=0;
        elseif strcmp(lower(CurveTypeBot),'str')
            mb=(coord(1,2)-coord(2,2))/(coord(1,1)-coord(2,1));
            cb=coord(1,2);
            yup=R*cosh(alpha*x2);
            ylow=mb*x+cb;
        end
        %get the spaced vertical Colum of y data
        [X(:,col),Y(:,col),StepY]=getVerticleArray(numElmV,x,ylow,yup,PlusOne);
    elseif strcmp(lower(CurveTypeTop),'exp')
        %TBD
        disp('Error: Message')
        disp('Exponetial Curve Type not created yet in GetMeshArbShape.m')
        break
    else
        disp('Error: Message')
        disp('Curve Type not recognised in GetMeshArbShape.m')
        break
    end
   %increment col counter
   col=col+1;
end %while
    %define the connectivity
    numelms=0;
    if strcmp(lower(elmType),'quad4') %assume anti clockwise connect
        for col=1:size(X,2)-1
```

```
        for row=1:size(X,1)-1
            numelms=numelms+1;
            inner=row+(col-1)*size(X,1);
            outer=row+col*(size(X,1));
            Elm(numelms).connect=[inner,outer,outer+1,inner+1] %this is perfect anti-clockwise
            Elm(numelms).SubSetNum=numelms;
            Elm(numelms).type='quad4';
            Elm(numelms).Grid(1,1:2)=[X(row,col),Y(row,col)];
            Elm(numelms).Grid(2,1:2)=[X(row,col+1),Y(row,col+1)];
            Elm(numelms).Grid(3,1:2)=[X(row+1,col+1),Y(row+1,col+1)];
            Elm(numelms).Grid(4,1:2)=[X(row+1,col),Y(row+1,col)];
        end
      end
   else%assume its tria3
     for col=1:size(X,2)-1
        for row=1:size(X,1)-1
            numelms=numelms+1;
            inner=row+(col-1)*size(X,1);
            outer=row+col*(size(X,1));
            Elm(numelms).connect=[inner,outer,outer+1];
            Elm(numelms).SubSetNum=numelms;
            Elm(numelms).type='tria3';
            Elm(numelms).Grid(1,1:2)=[X(row,col),Y(row,col)];
            Elm(numelms).Grid(2,1:2)=[X(row,col+1),Y(row,col+1)];
            Elm(numelms).Grid(3,1:2)=[X(row+1,col+1),Y(row+1,col+1)];
            %second element in overall
            numelms=numelms+1;
            Elm(numelms).connect=[inner,outer+1,inner+1];
            Elm(numelms).SubSetNum=numelms;
            Elm(numelms).type='tria3';
            Elm(numelms).Grid(1,1:2)=[X(row,col),Y(row,col)];
            Elm(numelms).Grid(2,1:2)=[X(row+1,col+1),Y(row+1,col+1)];
            Elm(numelms).Grid(3,1:2)=[X(row+1,col),Y(row+1,col)];
        end
      end
   end
   %define the material for each element
   for row=1:numelms
      %Define the material type constant per section
      Elm(row).Mat=matType;
   end
end
```

G.5.14 GenerateMesh.m (version 2.0)

```
function [mesh]=GenerateMesh_v2(Geom,plotGridPoints,Mat,elmType)
%This function calculates the node points and elms for the model
%Geom
%   - Geom.Elm (1 to elmnumber)
%       Geom.Elm(i).Nodes(1-j) (1 to number of nodes 3 for tri 4 for quad
%       Geom.Elm(i).matNum
%   - Geom.Section (1 to number of sections
%       Geom.Section(i).name
%       Geom.Section(i).curveTop
%       Geom.Section(i).curveBot
%    -Geom.NumFasteners (1 to num fasteners)
%       Geom.NumFasteners(i).R;%radius fastener i
%       Geom.NumFasteners(i).StartLocX;%X location that fasterner block starts at
%       Geom.NumFasteners(i).StartLocY;%Y location that fasterner block starts at
%       Geom.NumFasteners(i).L;%total length fastener i
%                       %note first fastener should always start at (0,0) else all is offset
%                       %by thus amount to the left
%   -Mat
%       Mat.E       - youngs modulus
%       Mat.rho     - density
%       Mat.epsilon - poissons ratio
%       Mat.name    - name of material
%       Mat.type - f - fastener material or m - general materal
%                   section 1
%       (1 to number of materials)

%------------------------------------------------------------------------
%Created by: TJ McMillan
%Date: 14/03/2013
%Version: 1.0
```

```
%Updates:
%1.0    - Creation of file reader
%2.0    - replaced Grid1 etc by Grid(1,1:2)
%       - added option for only sections no fasteners
%       - fixed connect error where elements repeat (in getArbshape)
%--------------------------------------------------------------------------
display('Generation of Mesh Started');
tic
%check vars
if nargin == 2
    plotGridPoints=false;
end
%get the dimensions
%Lengths
L=0;
for i=1:size(Geom.L,2)
    L(i)=Geom.L(i);%L(1) is length at end section 1to2
end

%Radii
R=0;
for i=1:size(Geom.R,2)
    R(i)=Geom.R(i);%R(1) is radius at start section 1to2
end


%Get the number of elements wanted per fastener height
if isfield(Geom,'NumElmsInFastener')
    if Geom.NumElmsInFastener ~= 0
        NumElmFastener=Geom.NumElmsInFastener;
    else
        NumElmFastener=3;
    end
else
    NumElmFastener=4;%if no fasteners default is 10 elms
    numElms=NumElmFastener;
end

%GetFastenerDetails
if isfield(Geom,'NumberOfFasteners')
    NumFasteners=Geom.NumberOfFasteners;
else
    NumFasteners=0;
end

L_fast=0;
R_fast=0;
Start_fast=0;

%check if there is geom else assume one cylinder required
if ~isfield(Geom,'Sections')
    %add section for a cylinder
    Geom.Sections(1).name='1to2'
    Geom.Sections(1).curveTop='str'
    Geom.Sections(1).curveBot='str'
end

%loop over the sections and complete mesh
XsecStart=0;%xcoord section start
XsecEnd=0;%xcoord section end
mesh.X=[];
mesh.Y=[];
Elm=[];
for secnum=1:size(Geom.Sections,2)
    %rest subsec to zero
    SubSec=[];
    %get the section start coord
    if secnum==1
        XsecStart=0;
    else
        XsecStart=XsecStart+Geom.L(secnum-1);
    end
    %get the section end coordinate
    XsecEnd=XsecEnd+Geom.L(secnum);
```

```
%check if there is a fastener in this section
FastenerInSec=false;
type='none';
numSubSec=1;
i=1;
FastenerInSec=true;%always run once even if there isn't a fastener
%check if while statement is run
whileStatementRun=false;
while (i <= NumFasteners & FastenerInSec)
   XstartFastener=Geom.NumFasteners(i).StartLocX;
   XendFastener=Geom.NumFasteners(i).StartLocX+Geom.NumFasteners(i).L;
   if (XstartFastener == XsecStart & XendFastener < XsecEnd)
      type='left';
      numSubSec=4;
      SubSec(1).curveTop=Geom.Sections(secnum).curveTop;
      SubSec(1).curveBot='str';
      SubSec(2).curveTop='str';
      SubSec(2).curveBot=Geom.Sections(secnum).curveBot;
      SubSec(3).curveTop=Geom.Sections(secnum).curveTop;
      SubSec(3).curveBot=Geom.Sections(secnum).curveBot;
      SubSec(4).curveTop=Geom.Sections(secnum).curveTop;
      SubSec(4).curveBot=Geom.Sections(secnum).curveBot;

      %get coordinates of edges
      %subsec1
      SubSec(1).coord(1,1)=sum(L(1:secnum-1));
      SubSec(1).coord(1,2)=Geom.NumFasteners(i).R;
      SubSec(1).coord(2,1)= SubSec(1).coord(1,1)+Geom.NumFasteners(i).L;
      SubSec(1).coord(2,2)=SubSec(1).coord(1,2);
      x1=SubSec(1).coord(1,1);
      x2=sum(L(1:secnum));
      y1=R(secnum);
      y2=R(secnum+1);
      xf=SubSec(1).coord(1,1)+Geom.NumFasteners(i).L;
      SubSec(1).coord(3,2)=findTopSurfacePt(SubSec(1).curveTop,xf,x1,x2,y1,y2);
      SubSec(1).coord(3,1)=SubSec(1).coord(2,1);
      SubSec(1).coord(4,2)=R(secnum+1);
      SubSec(1).coord(4,1)=sum(L(1:secnum));
      %subsec2
      SubSec(2).coord(1,1)=0+sum(L(1:secnum-1));
      SubSec(2).coord(1,2)=0;
      SubSec(2).coord(2,1)=Geom.NumFasteners(i).L;
      SubSec(2).coord(2,2)=0;
      SubSec(2).coord(3,2)=Geom.NumFasteners(i).R;
      SubSec(2).coord(4,2)=Geom.NumFasteners(i).R;
      SubSec(2).coord(3,1)=SubSec(2).coord(2,1);
      SubSec(2).coord(4,1)=SubSec(2).coord(1,1);
      %subsec3
      SubSec(3).coord(1,1)=0+sum(L(1:secnum-1))+Geom.NumFasteners(i).L;
      SubSec(3).coord(1,2)=0;
      SubSec(3).coord(2,1)=L(secnum);
      SubSec(3).coord(2,2)=0;
      SubSec(3).coord(3,2)=Geom.NumFasteners(i).R;
      SubSec(3).coord(4,2)=Geom.NumFasteners(i).R;
      SubSec(3).coord(3,1)=SubSec(3).coord(2,1);
      SubSec(3).coord(4,1)=SubSec(3).coord(1,1);
       %subsec4
      SubSec(4).coord(1,1)=0+sum(L(1:secnum-1))+Geom.NumFasteners(i).L;
      SubSec(4).coord(1,2)=Geom.NumFasteners(i).R;
      SubSec(4).coord(2,1)=SubSec(3).coord(2,1);
      SubSec(4).coord(2,2)=Geom.NumFasteners(i).R;
      SubSec(4).coord(3,2)=R(secnum+1);
      SubSec(4).coord(4,2)= SubSec(1).coord(3,2);
      SubSec(4).coord(3,1)=SubSec(4).coord(2,1);
      SubSec(4).coord(4,1)=SubSec(4).coord(1,1);

      %two bottom subsection have one less points
      SubSec(1).PlusOne=1;%this is used to determine number of points in verticle col
      SubSec(2).PlusOne=1;%this is used to determine number of points in verticle col
      SubSec(3).PlusOne=1;%this is used to determine number of points in verticle col
      SubSec(4).PlusOne=1;%this is used to determine number of points in verticle col

      %define materials per section
      SubSec(2).matnum=['f',num2str(secnum)];
```

```
        %Material
        SubSec(1).matnum=['m',num2str(secnum)];
        SubSec(3).matnum=['m',num2str(secnum)];
        SubSec(4).matnum=['m',num2str(secnum)];


        %calculate element length (section 2 fastener)
        %verticle element number
        Hfastener=SubSec(2).coord(4,2)-SubSec(2).coord(1,2);
        Helm=Hfastener/NumElmFastener;
        SubSec(2).numelmsV=NumElmFastener;
        SubSec(1).numelmsV=round((SubSec(1).coord(4,2)-SubSec(1).coord(1,2))/Helm);
        SubSec(3).numelmsV=SubSec(2).numelmsV;
        SubSec(4).numelmsV=SubSec(1).numelmsV;

    elseif (XstartFastener > XsecStart & XendFastener == XsecEnd)
        type='right';
        numSubSec=4;
        SubSec(1).curveTop=Geom.Sections(secnum).curveTop;
        SubSec(1).curveBot=Geom.Sections(secnum).curveBot;
        SubSec(2).curveTop=Geom.Sections(secnum).curveTop;
        SubSec(2).curveBot=Geom.Sections(secnum).curveBot;
        SubSec(3).curveTop='str';
        SubSec(3).curveBot=Geom.Sections(secnum).curveBot;
        SubSec(4).curveTop=Geom.Sections(secnum).curveTop;
        SubSec(4).curveBot='str';
        %subsec1
        SubSec(1).coord(1,1)=0+sum(L(1:secnum-1));
        SubSec(1).coord(1,2)=0;
        SubSec(1).coord(2,1)= SubSec(1).coord(1,1)+Geom.NumFasteners(i).L;
        x1=SubSec(1).coord(1,1);
        x2=SubSec(1).coord(2,1)
        y1=L(secnum);
        y2=L(secnum+1);
        xf=SubSec(1).coord(1,1)+Geom.NumFasteners(i).L;
        SubSec(1).coord(3,2)=findTopSurfacePt(SubSec(1).curveTop,xf,x1,x2,y1,y2);
        SubSec(1).coord(3,1)=SubSec(1).coord(2,1);
        SubSec(1).coord(4,2)=L(secnum);
        SubSec(1).coord(4,1)=SubSec(1).coord(1,1);
        %subsec2
        SubSec(2).coord(1,1)=0+sum(L(1:secnum-1));
        SubSec(2).coord(1,2)=Geom.NumFasteners(i).R;
        SubSec(2).coord(2,1)=Geom.NumFasteners(i).L;
        SubSec(2).coord(2,2)=Geom.NumFasteners(i).R;
        SubSec(2).coord(3,2)=L(secnum);
        SubSec(2).coord(4,2)=L;
        SubSec(2).coord(3,1)=SubSec(2).coord(2,1);
        SubSec(2).coord(4,1)=SubSec(2).coord(1,1);
        %subsec3
        SubSec(3).coord(1,1)=0+sum(L(1:secnum-1))+Geom.NumFasteners(i).L;
        SubSec(3).coord(1,2)=0;
        SubSec(3).coord(2,1)=SubSec(3).coord(1,1)+L(secnum+1);
        SubSec(3).coord(2,2)=0;
        SubSec(3).coord(3,2)=Geom.NumFasteners(i).R;
        SubSec(3).coord(4,2)=Geom.NumFasteners(i).R;
        SubSec(3).coord(3,1)=SubSec(3).coord(2,1);
        SubSec(3).coord(4,1)=SubSec(3).coord(1,1);
        %subsec4
        SubSec(4).coord(1,1)=0+sum(L(1:secnum-1))+Geom.NumFasteners(i).L;
        SubSec(4).coord(1,2)=Geom.NumFasteners(i).R;
        SubSec(4).coord(2,1)=SSubSec(3).coord(2,1);
        SubSec(4).coord(2,2)=Geom.NumFasteners(i).R;
        SubSec(4).coord(3,2)=R(secnum+1);
        SubSec(4).coord(4,2)= SubSec(1).coord(3,2);
        SubSec(4).coord(3,1)=SubSec(4).coord(2,1);
        SubSec(4).coord(4,1)=SubSec(4).coord(1,1);
        %calculate element length (section 2 fastener)
        %verticle element number
        Hfastener=SubSec(3).coord(4,2)-SubSec(3).coord(1,2);
        Helm=Hfastener/NumElmFastener;
        SubSec(3).numelmsV=NumElmFastener;
        SubSec(1).numelmsV=round((SubSec(1).coord(4,2)-SubSec(1).coord(1,2))/Helm);
        SubSec(2).numelmsV=SubSec(3).numelmsV;
```

```
        SubSec(4).numelmsV=SubSec(1).numelmsV;

        %two bottom subsection have one less points
        SubSec(1).PlusOne=1;%this is used to determine number of points in verticle col
        SubSec(2).PlusOne=1;%this is used to determine number of points in verticle col
        SubSec(3).PlusOne=1;%this is used to determine number of points in verticle col
        SubSec(4).PlusOne=1;%this is used to determine number of points in verticle col

        %define materials per section
         %define materials per section
        SubSec(3).matnum=['f',num2str(secnum)];

        %Material
        SubSec(1).matnum=['m',num2str(secnum)];
        SubSec(2).matnum=['m',num2str(secnum)];
        SubSec(4).matnum=['m',num2str(secnum)];

    elseif (XstartFastener == XsecStart & XendFastener == XsecEnd)
        type='full';
        FastenerInSec=false;%no more fasteners
        numSubSec=2;
        SubSec(1).curveTop=Geom.Sections(secnum).curveTop;
        SubSec(1).curveBot='str';
        SubSec(2).curveTop='str';
        SubSec(2).curveBot=Geom.Sections(secnum).curveBot;
        %subsec1
        SubSec(1).coord(1,1)=0+sum(L(1:secnum-1));
        SubSec(1).coord(1,2)=0;
        SubSec(1).coord(2,1)= SubSec(1).coord(1,1)+Geom.NumFasteners(i).L;
        SubSec(1).coord(2,2)=0;
        SubSec(1).coord(3,2)=Geom.NumFasteners(i).R;
        SubSec(1).coord(3,1)=SubSec(1).coord(2,1);
        SubSec(1).coord(4,2)=Geom.NumFasteners(i).R;
        SubSec(1).coord(4,1)=SubSec(1).coord(1,1);
        SubSec(1).PlusOne=0;%this is used to determine number of points in verticle col
        %subsec2
        SubSec(2).coord(1,1)=SubSec(1).coord(1,1);
        SubSec(2).coord(1,2)=SubSec(1).coord(4,2);
        SubSec(2).coord(2,1)=SubSec(1).coord(2,1);
        SubSec(2).coord(2,2)=SubSec(1).coord(3,2);
        SubSec(2).coord(3,2)=R(secnum);
        SubSec(2).coord(4,2)=R(secnum+1);
        SubSec(2).coord(3,1)=SubSec(2).coord(2,1);
        SubSec(2).coord(4,1)=SubSec(2).coord(1,1);
        SubSec(2).PlusOne=1;%this is used to determine number of points in verticle col
        %calculate element length (section 2 fastener)
        %verticle element number
        Hfastener=SubSec(2).coord(4,2)-SubSec(2).coord(1,2);
        Helm=Hfastener/NumElmFastener;
        SubSec(2).numelmsV=NumElmFastener;
        SubSec(1).numelmsV=round((SubSec(1).coord(4,2)-SubSec(1).coord(1,2))/Helm);

        %define materials per section
        SubSec(2).matnum=['f',num2str(secnum)];

        %Material
        SubSec(1).matnum=['m',num2str(secnum)];

    else
        type='none';
        numSubSec=1;
        SubSec(1).curveTop=Geom.Sections(secnum).curveTop;
        SubSec(1).curveBot=Geom.Sections(secnum).curveBot;
        %subsec1
        SubSec(1).coord(1,1)=0+sum(L(1:secnum-1));
        SubSec(1).coord(1,2)=0;
        SubSec(1).coord(2,1)= SubSec(1).coord(1,1)+L(secnum);
        SubSec(1).coord(2,2)=0;
        SubSec(1).coord(3,2)=R(secnum+1);
        SubSec(1).coord(3,1)=SubSec(1).coord(2,1);
        SubSec(1).coord(4,2)=R(secnum);
        SubSec(1).coord(4,1)=SubSec(1).coord(1,1);
        SubSec(1).PlusOne=1;%this is used to determine number of points in verticle col
        %verticle element number
```

```
        if (isempty(mesh.X))%if this is first section then 'elm nun in fastner'=vert num elm
            SubSec(1).numelmsV=NumElmFastener;
        else
           SubSec(1).numelmsV=size(unique(mesh.Y(:,1)),1)-1;
        end

       %Material
        SubSec(1).matnum=['m',num2str(secnum)];
        whileStatementRun=true;
    end
    %increment fastner
    i=i+1;
end %1 to number of fasteners (WHILE LOOP_

%if there is no fastener then define SubSec
%note: FastenerInSec true to run above while statement
if FastenerInSec & ~whileStatementRun
   for i_sec=1:size(Geom.Sections)
       SubSec(i_sec).curveTop=Geom.Sections(i_sec).curveTop;
       SubSec(i_sec).curveBot=Geom.Sections(i_sec).curveBot;
       SubSec(i_sec).coord(1,1)=0+sum(L(1:secnum-1));
       SubSec(i_sec).coord(1,2)=0;
       SubSec(i_sec).coord(2,1)= SubSec(1).coord(1,1)+Geom.L(i_sec);
       SubSec(i_sec).coord(2,2)=0;
       SubSec(i_sec).coord(3,2)=Geom.R(i_sec);
       SubSec(i_sec).coord(3,1)=SubSec(i_sec).coord(2,1);
       SubSec(i_sec).coord(4,2)=Geom.R(i_sec);
       SubSec(i_sec).coord(4,1)=SubSec(i_sec).coord(1,1);;
       SubSec(i_sec).curveTop=Geom.Sections(i_sec).curveBot;
       SubSec(i_sec).coord(1,1)=0+sum(L(1:i_sec-1));
       SubSec(i_sec).coord(1,2)=0;
       SubSec(i_sec).coord(2,1)= SubSec(i_sec).coord(1,1)+Geom.L(i);
       SubSec(i_sec).coord(2,2)=0;
       SubSec(i_sec).coord(3,2)=Geom.R(i_sec+1);
       SubSec(i_sec).coord(3,1)=SubSec(i_sec).coord(2,1);
       SubSec(i_sec).coord(4,2)=Geom.R(i_sec);
       SubSec(i_sec).coord(4,1)=SubSec(i_sec).coord(1,1);
       %Material
        SubSec(i_sec).matnum=['m',num2str(i_sec)];
        %define number of verticle elements
        SubSec(i_sec).numelmsV=numElms;
        SubSec(i_sec).PlusOne=1;%this is used to determine number of points in verticle col
   end
end
%complete the meshing of the section
for i=1:numSubSec
    %init elm type
    Elmtmp=[];
    %get the length of the section
    l=SubSec(i).coord(2,1)-SubSec(i).coord(1,1);
    %generate mesh of small quadralaterial
    [X(i).meshX,Y(i).meshY,Elmtmp]=GetMeshArbShape_v2(SubSec(i).coord,l,SubSec(i).numelmsV, SubSec(i).curveTop,...
        SubSec(i).curveBot,SubSec(i).PlusOne, SubSec(i).matnum,elmType);
    %combine all element subset
    currentElm=size(Elm,2);
    if currentElm==[]
        currentElm=0;
    end
    for r=1:size(Elmtmp,2)
        Elm(currentElm+r).mat=Elmtmp(r).Mat;
        Elm(currentElm+r).connect=Elmtmp(r).connect;
        Elm(currentElm+r).SubSetNum=Elmtmp(r).SubSetNum;
        Elm(currentElm+r).type=Elmtmp(r).type;
        if strcmp(lower(elmType),'quad4')
            Elm(currentElm+r).Grid(1,1:2)=Elmtmp(r).Grid(1,1:2);
            Elm(currentElm+r).Grid(2,1:2)=Elmtmp(r).Grid(2,1:2);
            Elm(currentElm+r).Grid(3,1:2)=Elmtmp(r).Grid(3,1:2);
            Elm(currentElm+r).Grid(4,1:2)=Elmtmp(r).Grid(4,1:2);
        else
            Elm(currentElm+r).Grid(1,1:2)=Elmtmp(r).Grid(1,1:2);
            Elm(currentElm+r).Grid(2,1:2)=Elmtmp(r).Grid(2,1:2);
            Elm(currentElm+r).Grid(3,1:2)=Elmtmp(r).Grid(3,1:2);
        end
    end
```

```
    end%end for 1 to number of sub sections
    %combine the meshes into one complete mesh for the sections
    if (size(SubSec,2)==1)
        mesh.X=[mesh.X ,X(1).meshX];
        mesh.Y=[mesh.Y ,Y(1).meshY];
    elseif (size(SubSec,2)==2)
        mesh.X=[mesh.X ,[X(2).meshX;X(1).meshX]];
        mesh.Y=[mesh.Y ,[Y(2).meshY;Y(1).meshY]];
    else
        mesh.X=[mesh.X ,[[X(2).meshX(1:end-1,:);X(1).meshX],[X(3).meshX(1:end-1,:);X(4).meshX]]];%Note: remove last row as should be
duplicate in other sub sections
        mesh.Y=[mesh.Y ,[[Y(2).meshY(1:end-1,:);Y(1).meshY],[Y(3).meshY(1:end-1,:);Y(4).meshY]]];
    end
end %for secnum

%PLOT----------------------------------------------------------------------
if (plotGridPoints)
    figure(1)
    hold on
    for i=1:size(Elm,2)%for number of elements
        if strcmp(Elm(i).mat(1),'m')
            pointType='-xb';
        else
            pointType='-or';
        end
        if strcmp(lower(Elm(i).type),'quad4')
            x=[Elm(i).Grid(1,1),Elm(i).Grid(2,1),Elm(i).Grid(3,1),Elm(i).Grid(4,1),Elm(i).Grid(1,1)];
            y=[Elm(i).Grid(1,2),Elm(i).Grid(2,2),Elm(i).Grid(3,2),Elm(i).Grid(4,2),Elm(i).Grid(1,2)];
            plot(x,y,pointType);
        else
            x=[Elm(i).Grid(1,1),Elm(i).Grid(2,1),Elm(i).Grid(3,1),Elm(i).Grid(1,1)];
            y=[Elm(i).Grid(1,2),Elm(i).Grid(2,2),Elm(i).Grid(3,2),Elm(i).Grid(1,2)];
            plot(x,y,pointType);
        end
    end
    %plot fastener
    plot([0;L_fast(1)],[R_fast(1);R_fast(1)],'-c','Linewidth',1.5);
    plot([L_fast(1);L_fast(1)],[0;R_fast(1)],'-c','Linewidth',1.5);
    hold off
    xlabel('X [length unit]','FontWeight','bold');
    ylabel('Y [length unit]','FontWeight','bold');
    title('Grid Points','FontSize',14,'FontWeight','bold');
end

mesh.Elm=Elm;

display('Generation of Mesh finished');
toc

%TIME----------------------------------------------------------------------
timeElapsed=toc;
fprintf('Time taken to complete mesh creation: %f\n',timeElapsed);
end
```

G.5.15 SplitStringIntoArray.m

This function takes a string and splits it using the given chars required to be split in sequence. Thus if 'CharToSplitBy'=' ,.' It will first split the string using spaces then split the subsequent strings using the comma and finally the full stop.

```
function splitstr_final = SplitStringIntoArray(StrToSplit,CharToSplitBy,RemoveDuplicateDelimintors)
%split the string for many chars e.g: ' .,'
%-------------------------------------------------------------------------
%Created by: TJ McMillan
%Date: 8/11/2013
%Version: 1.0
%Updates:
%1.0    - Creation of string splitter for multiple chars
%-------------------------------------------------------------------------
%StrToSplit  is the string we want to split
%CharToSplitBy is a string of chars that we will split the string by
%splitstr is a sturcture of strings 1-as many strings e.g: splitstr(i).str
%
   splitstr_final=[];
   for i = 1:length(CharToSplitBy)
      if i==1 %for first char will split string into string structure
         splitstr_final = SplitString(StrToSplit,CharToSplitBy(i),RemoveDuplicateDelimintors);
      else %else for if i>1
         %define new temporay structure
         temp=splitstr_final;
         splitstr_final=[];
         %define a counter
         cnt_start=0;
         for j=1:size(temp,2)
            str_j = SplitString(temp(j).str,CharToSplitBy(i),RemoveDuplicateDelimintors)
            for k=1:size(str_j,2)
             splitstr_final(cnt_start+k).str=str_j(k).str;
            end
            if isempty(k)
               k=0;
            end
            cnt_start=cnt_start+k;
         end
      end
    end %i
end
```

G.5.16 SplitString.m

This function only takes a single char and splits the string according to that char alone, see code below:

```
function splitstr_final = SplitString(StrToSplit,CharToSplitBy,RemoveDuplicateDelimintors)
%split the string for ONE specific char
%--------------------------------------------------------------------------
%Created by: TJ McMillan
%Date: 8/11/2013
%Version: 1.0
%Updates:
%1.0    - Creation of string splitter
%--------------------------------------------------------------------------
%StrToSplit  is the string we want to split
%CharToSplitBy is a SINGLE of char that we will split the string by
%splitstr is a sturcture of strings 1-as many strings e.g: splitstr(i).str
%
cnt = 1;
prevSplitNo = 1;
splitstr=[];
for j=1:length(StrToSplit)
    if (lower(StrToSplit(j)) == lower(CharToSplitBy))
        %               if (cnt >1)
        if (prevSplitNo>j-1)%case of duplicate seperators
            splitstr(cnt).str = StrToSplit(prevSplitNo:(j));
        else
            splitstr(cnt).str = StrToSplit(prevSplitNo:(j-1));
        end
        %               else
        %                   if (prevSplitNo>j-1)%case of duplicate seperators
        %                   splitstr(cnt).str = StrToSplit(prevSplitNo:(j-1));
        %               end
        cnt = cnt +1;
        prevSplitNo = j+1;
    elseif j==length(StrToSplit)%if its the last character
        splitstr(cnt).str = StrToSplit(prevSplitNo:length(StrToSplit));
    end
    %check if we want to remove duplicates then we must check here
    %and modiy cnt
    if (RemoveDuplicateDelimintors & ~isempty(splitstr) & cnt >1)
        if strcmp(splitstr(cnt-1).str,CharToSplitBy)
            %decrement counter
            cnt=cnt-1;
            %make the string null or empty
            splitstr(cnt).str=[];
        end
    end
end
%check if strucutre ends in '' then remove it
if isempty(splitstr(cnt).str)
    splitstr_final=[];
    for j=1:size(splitstr,2)-1
        splitstr_final(j).str=splitstr(j).str;
    end
else
    splitstr_final=splitstr;

end%if
end
```

G.5.17 ReadF06_MandK_FileType.m

This function takes a F06 file which has been created using the following Nastran PCL code to obtain the Kaa and Maa matrices in any order.

```
SOL 103
$DIAG 14
$ Direct Text Input for Executive Control
COMPILE PHASE1DR
ALTER 'CALL.*PHASE1B' $ AFTER CALL PHASE1B
$ print matrix kaa
MATPRT KAA/ $
$$MATPRN KAA/ $
$ print matrix maa
MATPRT MAA/ $
$$MATPRN MAA/ $
$$$
CEND
```

Call for Kaa and Maa matrix to be printed

Thus the following code was written:

```
function [K,M, status]=ReadF06_MandK_FileType(fFilename)
%This function uses the read the F06 file name and return the matrix K and
%M. If K and M are not located then the K and M matrices will be null ([])
%and the status will inidcate that no stiffness or mass matrices were found
%in the F06 file submitted
%-------------------------------------------------------------------------
%Created by: TJ McMillan
%Date: 06/01/2014
%Version: 1.0
%Updates:
%1.0    - Creation of file to read f06 and debugging using two standard F06
%         files
%-------------------------------------------------------------------------
%*************************************************************************
%Get the length of F06
%*************************************************************************
fid = fopen(fFilename);
fseek(fid, 0, 'eof');
filesize = ftell(fid);
fclose(fid);

%define a line counter
line=0;

%define flag for finding K and M matrices
MNotFound=true;
KNotFound=true;

%Initialise K and M
K=[];
M=[];
status='';

%open the file
fid = fopen(fFilename);

 while (~(filesize == ftell(fid)) & (KNotFound | MNotFound))
    %get first line
    tline=fgetl(fid);
    line=line+1;

   %check if line starts with a comment symbol - $ and its not ''
    if isempty(tline)
        lineNull=true;
    else %if the line is not null sized
```

```
        lineNull=false;
        if tline(1)=='$'
            comment=true;
        else
            comment=false;
        end
    end
    if ~comment & ~lineNull
        %look for KAA or MAA
        if (~isempty(strfind(lower(tline),'. kaa')) & KNotFound) %other kaa in executive statement
            %updated status
            status=[status ' K found '];
            %get next line till column
            tline=fgetl(fid);
            tline=fgetl(fid);
            line=line+2;
            %initiate loop stopper
            endOfK=false;
            columnNum=0;
            %get first line
            tline=fgetl(fid);
            line=line+1;
            if findstr('column', lower(tline))
              splitstr = SplitStringIntoArray(tline,' ',true);
              colNum=str2num(splitstr(2).str);
            end
             %loop till all of K is obtained
            while ~endOfK
                %loop until next blank tline is reached
                 tline=fgetl(fid)
                 line=line+1
%                TlineIsEmpty=isempty(tline)
                while ~(isempty(tline)) & isempty(strfind(lower(tline),'msc.nast'))
%                    disp('here1')
                    splitstr = SplitStringIntoArray(tline,' ',true);
                    rowNum=str2num(splitstr(1).str);
                    for i=2:size(splitstr,2)-1
                        K(rowNum+i-2,colNum)=str2num(splitstr(i).str);
                    end %end for
                    tline=fgetl(fid);
                    line=line+1;
%                    if isempty(tline)
%                        disp('empty tline')
%                    end
                end %end while ~(isempty(tline))
                %check if endo of Kaa area in F06
                tline=fgetl(fid);
                line=line+1;
                if ~isempty(strfind(lower(tline),'column'))
                     splitstr = SplitStringIntoArray(tline,' ',true);
                     colNum=str2num(splitstr(2).str);
                elseif ~isempty(strfind(lower(tline),'maa')) | ~isempty(strfind(lower(tline),'***'))
                    endOfK=true;
                else %if after 3 lines still no maa (extra matrx) or column then we assume all matrix is found)
                    splitstr = SplitStringIntoArray(tline,' ',true);
                    FoundKaaOrNextCol=false;
                    i=1;
                  while ~FoundKaaOrNextCol & i <=4
                    if ~isempty(strfind(lower(tline),'column'))
                       splitstr = SplitStringIntoArray(tline,' ',true);
                       colNum=str2num(splitstr(2).str);
                       FoundKaaOrNextCol=true;
                    elseif ~isempty(strfind(lower(tline),'kaa'))
                       %get next line till column
                       tline=fgetl(fid);
                       tline=fgetl(fid);
                       tline=fgetl(fid); %should be column
                       line=line+3;
                       FoundKaaOrNextCol=true;
                       splitstr = SplitStringIntoArray(tline,' ',true);
                       colNum=str2num(splitstr(2).str);
                       break; %exit while loop without reading next line
                    elseif ~isempty(strfind(lower(tline),'maa'))
                       FoundKaaOrNextCol=false;
```

```
                    break; %exit while loop without reading next line
                  end
                  tline=fgetl(fid);
                  line=line+1;
                  i=i+1;
                end %While ~TerminateKfind
                %check if no Kaa found or next column then terminate the
                %K matrix
                if ~FoundKaaOrNextCol
                   endOfK=true;
                end
              end
            end %end while ~endOfK
            %K matrix found
            KNotFound=false;
          end %end if findstr('. kaa', lower(tline))
          if (~isempty(strfind(lower(tline),'. maa')) & MNotFound) %other maa in executive statement
            %updated status
            status=[status ' M found '];
            %get next line till column
            tline=fgetl(fid);
            tline=fgetl(fid);
            line=line+2;
            %initiate loop stopper
            endOfM=false;
            columnNum=0;
            %get first line
            tline=fgetl(fid);
            line=line+1;
            if ~isempty(strfind(lower(tline),'column'))
              splitstr = SplitStringIntoArray(tline,' ',true);
              colNum=str2num(splitstr(2).str);
            end
             %loop till all of K is obtained
            while ~endOfM
               %loop until next blank tline is reached
               tline=fgetl(fid);
               line=line+1;
               while ~(isempty(tline)) & isempty(strfind(lower(tline),'msc.nast'))
                  splitstr = SplitStringIntoArray(tline,' ',true);
                  rowNum=str2num(splitstr(1).str);
                  for i=2:size(splitstr,2)-1
                     M(rowNum+i-2,colNum)=str2num(splitstr(i).str);
                  end %end for
                  tline=fgetl(fid);
                  line=line+1;
               end %end while ~(isempty(tline))
              %check if endo of Kaa area in F06
              tline=fgetl(fid);
              line=line+1;
              if ~isempty(strfind(lower(tline),'column'))
                  splitstr = SplitStringIntoArray(tline,' ',true);
                  colNum=str2num(splitstr(2).str);
              elseif ~isempty(strfind(lower(tline),'kaa')) | ~isempty(strfind(lower(tline),'***'))
                  endOfM=true;
              else %if two subsequent line 0 then 0 in first place then exit
                  splitstr = SplitStringIntoArray(tline,' ',true);
                  FoundMaaOrNextCol=false;
                  i=1;
                 while ~FoundMaaOrNextCol & i <=4
                  if ~isempty(strfind(lower(tline),'column'))
                     splitstr = SplitStringIntoArray(tline,' ',true);
                     colNum=str2num(splitstr(2).str);
                     FoundMaaOrNextCol=true;
                  elseif ~isempty(strfind(lower(tline),'maa'))
                     %get next line till column
                     tline=fgetl(fid);
                     tline=fgetl(fid);
                     tline=fgetl(fid); %should be column
                     line=line+3;
                     FoundMaaOrNextCol=true;
                     splitstr = SplitStringIntoArray(tline,' ',true);
                     colNum=str2num(splitstr(2).str);
                     break; %exit while loop without reading next line
```

```
                elseif ~isempty(strfind(lower(tline),'kaa'))
                    FoundMaaOrNextCol=false;
                    break; %exit while loop without reading next line
                end
                tline=fgetl(fid);
                line=line+1;
                i=i+1;
            end %While ~TerminateKfind
            %check if no Kaa found or next column then terminate the
            %K matrix
            if ~FoundMaaOrNextCol
                endOfM=true;
            end
        end
    end %end while ~endOfM
    %M matrix found
    MNotFound=false;
  end %end if findstr('. maa', lower(tline))
 else %if the line is a comment
  %just move to the next line
 end
end
if strcmp(status,'')
 status='No K and M found';
end
%close file
fclose(fid);

end %for function
legend('Orginal Data','Least Squares Fit')
hold off
grid on

%check using exact solution
y bar=mean(O)
SSreg=0;
SSres=0;
SStot=0;
for i=1:size(O,1)
    SStot=SStot+(O(i,1)-y bar)^2;
    SSreg=SSreg+(a*I(i,1)^b-y bar)^2;
    SSres=SSres+(O(i,1)-a*I(i,1)^b)^2;
end
R squared=1-SSres/SStot

%using matlab fit finction
[curve, goodness]=fit(I,O, 'power1')
figure(1)
hold on
plot(curve,'-r')
hold off
```

G.5.18 ReadBDFandRunModalAnalysis.m

```
clc
clear all
close all
```

Define Variables

```
%define know variables
pathdir='Q:\Shared_Pool\Tristan\TJ_McMillan_Masters\FEM_Models\Horns\Al2024Horn_Mark1\'
BDFname='HornAl2024AndGrub.bdf'
```

Read BDF

```
[mesh,mat]=ReadBDFFileType(pathdir,BDFname)

%%Remove mesh.Elm with null in them (generall from Patran numbering issues
%%where elment numbers dont start at 1
temp=[];
j=1;
for i=1:size(mesh.Elm,2)
    if ~isempty(mesh.Elm(i).Mat)
       temp.Elm(j)=mesh.Elm(i);
       j=j+1;
    end
end
%Convert mm to m
FactorGeom=1000;
%in Nastran X - radius Z- axis of revolution
temp.X=(mesh.X)./FactorGeom; %make all meters
temp.Y=(mesh.Y)./FactorGeom; %make all meters
%Modify mesh.Elm(:).grid
for i=1:size(temp.Elm,2)
    Grid=temp.Elm(i).Grid./FactorGeom;
%     Grid=fliplr(Grid);
    temp.Elm(i).Grid=Grid;
end
temp.NumNodes=mesh.NumNodes;

%conver E,and density to kg meters
for i=1:size(mat,2)
    mat(i).E=mat(i).E.*10^6;
    mat(i).Rho=mat(i).Rho.*10^9;
end

clear mesh
mesh=temp;
clear temp
```

Complete Modal Analysis

```
[freq,modeShapes]=GetModalAnalysis(mesh,mat);
cnt1=1;
freqInRng=0;
for i=1:size(freq,1)
   if freq(i)>10000 & freq(i)<30000
       NumInRange=i
       freq(NumInRange)
       freqInRng(cnt1)=freq(i);
       cnt1=cnt1+1;
   end
end
%%Plot the mesh
% error=PlotMesh(mesh,'b')
%%PlotModes
i=2
```

```
freq(i)
error=PlotModes(mesh,modeShapes,i,-0.1);
freq_all=sort(abs(freq))
save([pathdir 'Test.mat'], 'mesh','modeShapes','freq_all')
%Read Nastran K matrix
% [K_lumped,M_lumped,status]=ReadF06_MandK_FileType([pathdir '\' 'ctria_8elm_renum' '_lumped.f06']);
% [K_const,M_const,status]=ReadF06_MandK_FileType([pathdir '\' 'ctria_8elm_renum' '_consistent.f06']);
[Freq, status]=ReadF06_Freq_FileType([pathdir 'HornAl2024AndGrub' '.f06'])

% load('StiffnessAndMass_8elmTria3X_Const_Method4.mat')
% K_full=full(K)
% M_full=full(M)
```

G.5.19 GetModalAnalysis.m

```
function [freq,modeShapes]=GetModalAnalysis(mesh,mat,meshform,K,M)
%This function obtains the
%stiffness and mass matrices from the correct function based on element type
%from this the eigen values and mode shapes are calculated
%------------------------------------------------------------------------
%Created by: TJ McMillan
%Date: 15/05/2013
%Version: 1.1
%Updates:
%1.0    - Creation of modal analyser
%1.1    - All GetKandM functions updated to version 2
%------------------------------------------------------------------------
if nargin==2
    meshform='axissymetric';
end

if nargin==2 | nargin==3
    %initialise the global mass and stiffness matrices to null
    M=0;
    K=0;

    %first determine which method to use
    if strcmp(lower(meshform),'axissymetric')
        %get the stiffness matrix and mass matrix
         if (strcmp(lower(mesh.Elm(1).Type),'quadx4'))
            [K,M]=GetKandM_Quad4AxiSym_v2(mesh,mat);
         else
            [K,M]=GetKandM_Tria3AxiSym_v2(mesh,mat,1,4);
         end
    else %we assume only a 2d mesh
        %get the stiffness matrix and mass matrix
        %updated to version 2
        if (strcmp(lower(mesh.Elm(1).Type),'quad4'))
            [K,M]=GetKandM_Quad4_v2(mesh,mat);
        elseif (strcmp(lower(mesh.Elm(1).Type),'quad8'))
            [K,M]=GetKandM_Quad8_v2(mesh,mat);
        elseif (strcmp(lower(mesh.Elm(1).Type),'tria3'))
            [K,M]=GetKandM_Tria3_v2(mesh,mat);
        elseif (strcmp(lower(mesh.Elm(1).Type),'tria6'))
            [K,M]=GetKandM_Tria6_v2(mesh,mat);
        else
            disp('Element type not recognised in GetModalAnalysis.m')
            return
        end
    end
end

%calculate the eigen values and mode shapes
% save('StiffnessAndMass.mat','K','M')
useLargeMatrixMethod=true;
if ~useLargeMatrixMethod
    [modeShapes,eigenValDiag]=eig(full(K),full(M));
else
    [modeShapes,eigenValDiag]=eigs(K,M,8,'sm'); %get 8 smallest eigen values
end
%get frequencies from diagonal matrix
omega_squared=diag(eigenValDiag); %but this is the omega squared
omega=sqrt(omega_squared);%this is omega
freq=omega./(2*pi);
end
```

APPENDIX H RESIDUAL LOAD DATA

The following are the residual load data for each type of specimen.

H.1 AL2024

H.1.1 S2-5 and S2-6

DENEL SAAB AEROSTRUCTURES
PRODUCTION LABORATORY
INSTRON 5500R SN J4969

Round test piece Aluminium
ASTM B557

Test type: Tensile
Operator name: Walter Fransman
Sample Identification: TRISTAN S2-5-6
Interface Type: 5500

Instron Corporation
Series IX Automated Materials Testing System 8.25.00
Test Date: Wednesday, November 26, 2014

Sample Rate (pts/secs): 1.0000
Crosshead Speed: 1.0000 mm/min
Second Speed: 3.0000 mm/min
Third Speed: 0.0000 mm/min

Full Scale Load Range: 100.0000 kN

Humidity (%): 29
Temperature: 23 C

Spec: U.T.S. (Min)

Spec: Proof (Min)

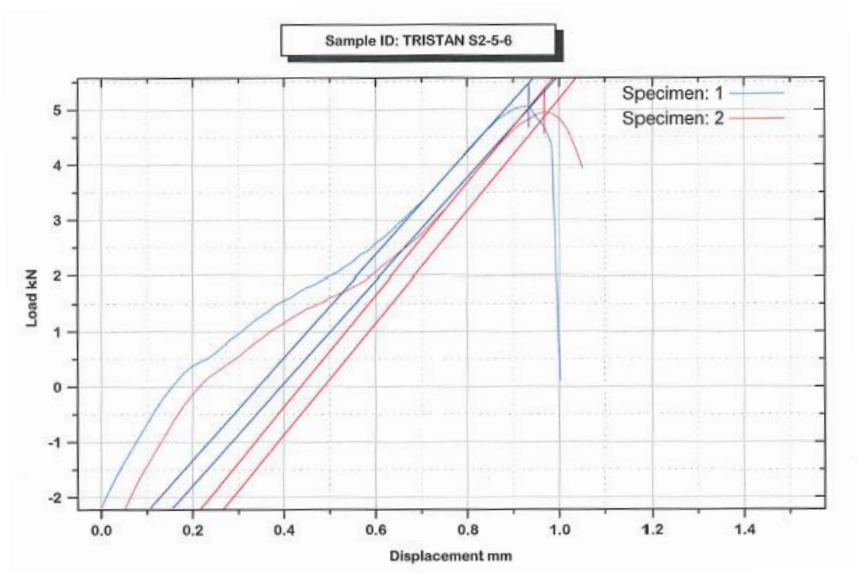
Spec:Elongation(Min)

Spec: R of A (Min)

Sample comments:

Tensile broke in gauge length.

	Load at Max.Load (N)	Stress at Max.Load (MPa)	Load at offset Yield 1 (N)	Stress at offset Yield 1 (MPa)	Diameter (mm)	Area (mm^2)
S2-5	5067	254.0	4859	243.5	5.040	19.950
S2-6	4958	252.5	4931	251.1	5.000	19.635



H.1.2 S2-7 to S2-10

DENEL SAAB AEROSTRUCTURES
PRODUCTION LABORATORY
INSTRON 5500R SN J4969

Round test piece Aluminium
ASTM B557

Test type: Tensile
Operator name: Walter Fransman
Sample Identification: TRISTAN S2-7-10
Interface Type: 5500

Instron Corporation
Series IX Automated Materials Testing System 8.25.00
Test Date: Tuesday, November 25, 2014

Sample Rate (pts/secs): 1.0000
Crosshead Speed: 1.0000 mm/min
Second Speed: 3.0000 mm/min
Third Speed: 0.0000 mm/min

Full Scale Load Range: 100.0000 kN

Humidity (%): 29
Temperature: 23 C

Spec: U.T.S. (Min)

Spec: Proof (Min)

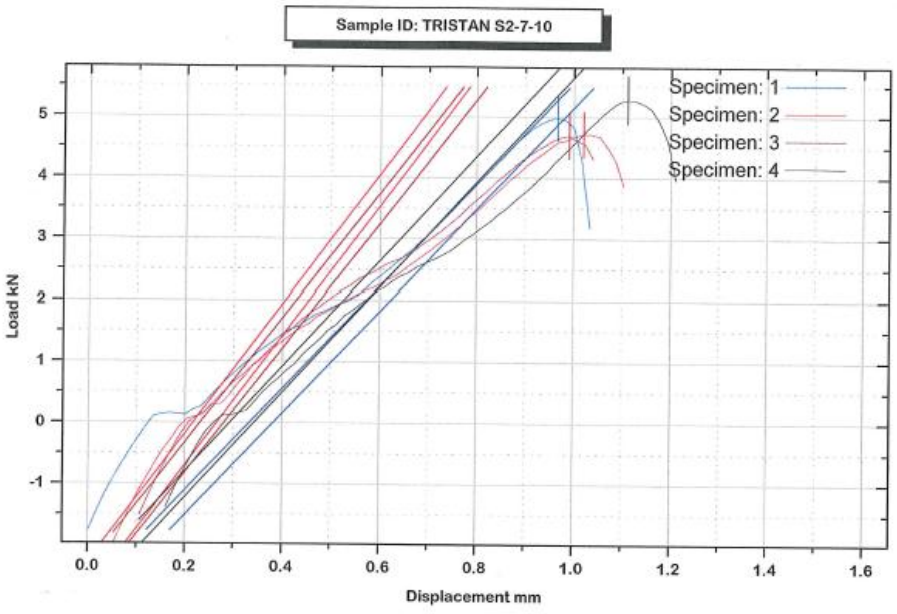
Spec:Elongation(Min)

Spec: R of A (Min)

Sample comments:

Tensile broke in gauge length.

	Load at Max.Load (N)	Stress at Max.Load (MPa)	Load at offset Yield 1 (N)	Stress at offset Yield 1 (MPa)	Diameter (mm)	Area (mm^2)
S2-7	4982	249.7	4948	248.0	5.040	19.950
S2-8	4719	237.5	1248	62.8	5.030	19.871
S2-9	4695	237.2	1879	95.0	5.020	19.792
S2-10	5281	266.8	2136	107.9	5.020	19.792



H.1.3 S2-11 to S2-15

DENEL SAAB AEROSTRUCTURES
PRODUCTION LABORATORY
INSTRON 5500R SN J4969

Round test piece Aluminium
ASTM B557

Test type:	Tensile		Instron Corporation	
Operator name:	Walter Fransman		Series IX Automated Materials Testing System	8.25.00
Sample Identification:	TRISTAN S2-11-15		Test Date: Tuesday, November 25, 2014	
Interface Type:	5500			
Sample Rate (pts/secs):	1.0000			
Crosshead Speed:	1.0000	mm/min	Humidity (%): 29	
Second Speed:	3.0000	mm/min	Temperature: 23 C	
Third Speed:	0.0000	mm/min		
Full Scale Load Range:	100.0000	kN		

Spec: U.T.S. (Min)

Spec: Proof (Min)

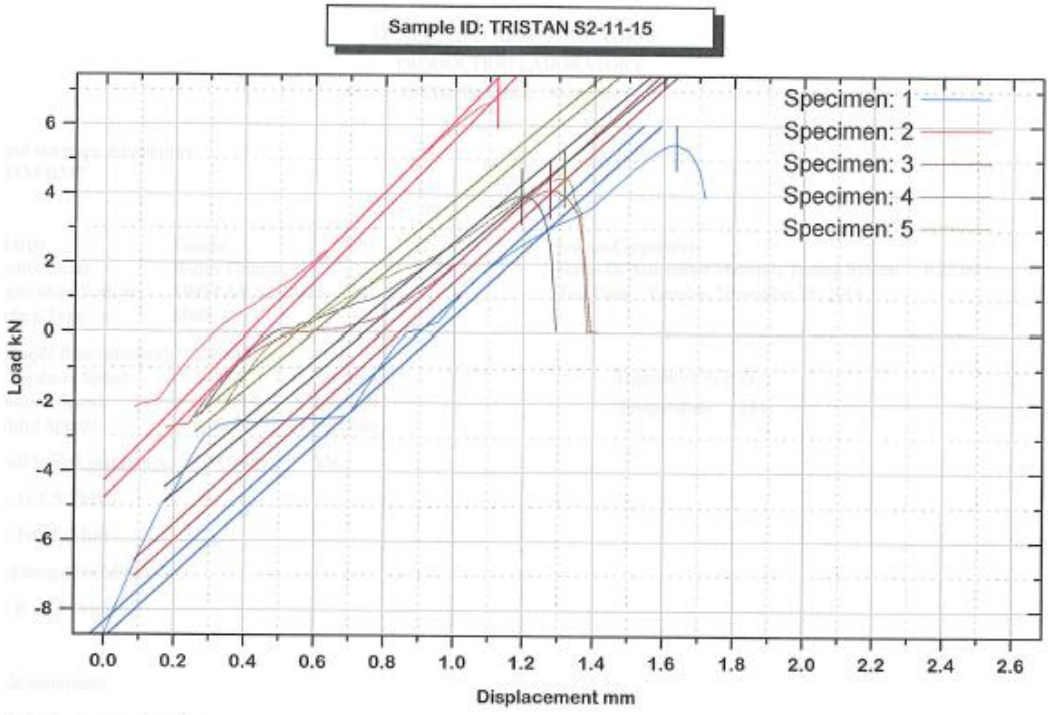
Spec:Elongation(Min)

Spec: R of A (Min)

Sample comments:

Tensile broke in gauge length.

	Load at Max.Load (N)	Stress at Max.Load (MPa)	Load at offset Yield 1 (N)	Stress at offset Yield 1 (MPa)	Diameter (mm)	Area (mm^2)
S2 - 11	5472	274.3	2967	148.7	5.040	19.950
S2 - 12	6766	339.1	6766	339.1	5.040	19.950
S2 - 13	4133	208.0	4114	207.0	5.030	19.871
S2 - 14	3955	199.0	3936	198.1	5.030	19.871
S2 - 15	4471	232.3	1404	73.0	4.950	19.244



H.1.4 S2-16 to S2-25

DENEL SAAB AEROSTRUCTURES
PRODUCTION LABORATORY
INSTRON 5500R SN J4969

Round test piece Aluminium
ASTM B557

Test type: Tensile
Operator name: Walter Fransman
Sample Identification: TRISTON Al 2024 A-J
Interface Type: 5500

Instron Corporation
Series IX Automated Materials Testing System 8.25.00
Test Date: Tuesday, March 03, 2015

Sample Rate (pts/secs): 1.0000
Crosshead Speed: 1.0000 mm/min
Second Speed: 3.0000 mm/min
Third Speed: 0.0000 mm/min

Full Scale Load Range: 100.0000 kN

Humidity (%): 29
Temperature: 23 C

Spec: U.T.S. (Min) MPa

Spec: Proof (Min) MPa

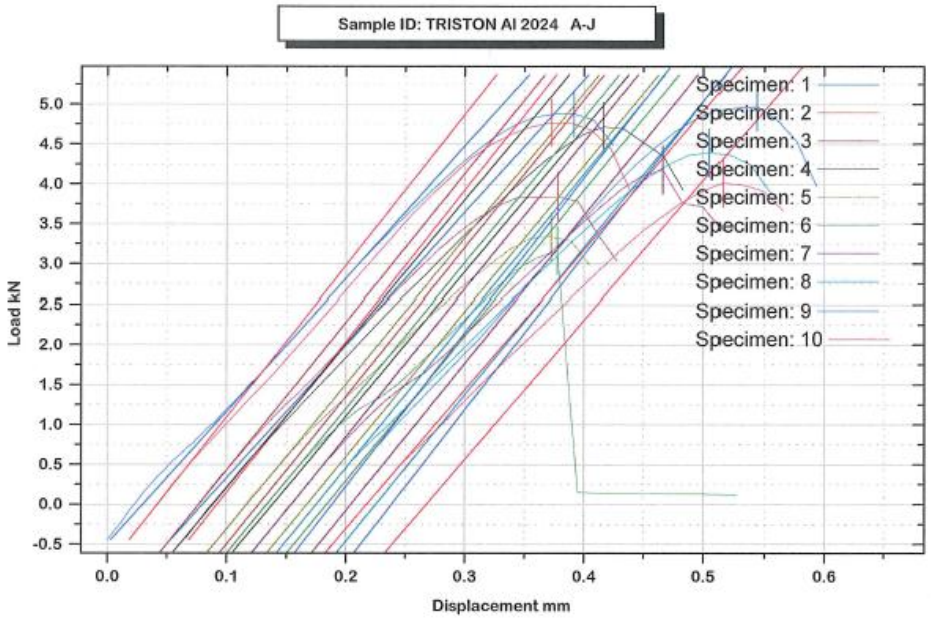
Spec:Elongation(Min)%

Spec: R of A (Min)

Sample comments: ALUMINIUM 2024 A-J

Tensile broke in gauge length.

	Load at Max.Load (N)	Stress at Max.Load (MPa)	Load at offset Yield 1 (N)	Stress at offset Yield 1 (MPa)	Diameter (mm)	Area (mm^2)
A	4884	255.9	4881	255.7	4.930	19.089
B	4769	255.0	4625	247.3	4.880	18.704
C	3844	198.9	3835	198.5	4.960	19.322
D	4721	246.3	4624	241.3	4.940	19.167
E	3336	168.5	3324	168.0	5.020	19.792
F	3170	158.9	3081	154.4	5.040	19.950
G	4177	219.7	3521	185.2	4.920	19.012
H	4396	227.5	4101	212.2	4.960	19.322
I	4976	256.5	4817	248.3	4.970	19.400
J	4015	210.4	3908	204.7	4.930	19.089



H.2 AL7075

H.2.1 S1-1, S1-8 to S1-11

DENEL SAAB AEROSTRUCTURES
PRODUCTION LABORATORY
INSTRON 5500R SN J4969

Round test piece Aluminium
ASTM B557

Test type: Tensile
Operator name: Walter Fransman
Sample Identification: TRISTAN S1-1-11
Interface Type: 5500

Instron Corporation
Series IX Automated Materials Testing System 8.25.00
Test Date: Tuesday, November 25, 2014

Sample Rate (pts/secs): 1.0000
Crosshead Speed: 1.0000 mm/min
Second Speed: 3.0000 mm/min
Third Speed: 0.0000 mm/min

Full Scale Load Range: 100.0000 kN

Humidity (%): 29
Temperature: 23 C

Spec: U.T.S. (Min)

Spec: Proof (Min)

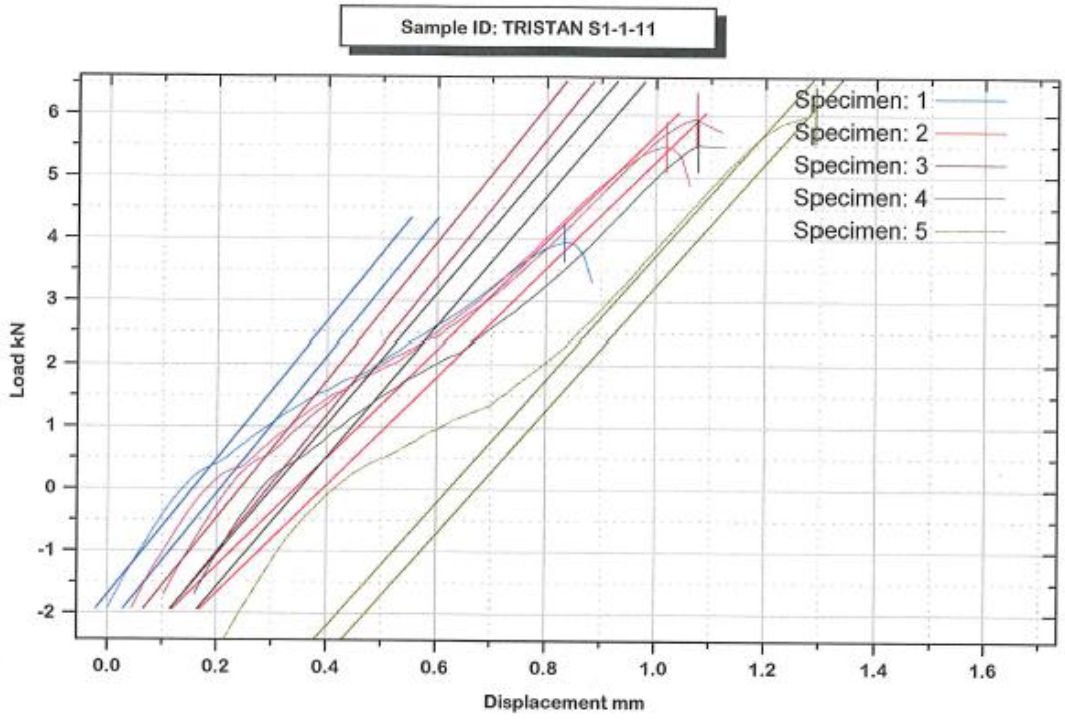
Spec:Elongation(Min)

Spec: R of A (Min)

Sample comments:

Tensile broke in gauge length.

	Load at Max.Load (N)	Stress at Max.Load (MPa)	Load at offset Yield 1 (N)	Stress at offset Yield 1 (MPa)	Diameter (mm)	Area (mm^2)
S1 - 1	3949	191.1	1169	56.6	5.130	20.669
S1 - 8	5491	274.1	5482	273.7	5.050	20.030
S1 - 7	5941	302.6	1603	81.7	5.000	19.635
S1 - 10	5513	305.9	1440	79.9	4.790	18.020
S1 - 11	6004	310.7	6004	310.7	4.960	19.322



H.2.2 S1-5

DENEL AEROSTRUCTURES
PRODUCTION LABORATORY
Instron 3369/J4973 (50 kN)

Round test pieces
ASTM E8M

Test type:	Tensile		Instron Corporation	
Operator name:	Elsa Pretorius		Series IX Automated Materials Testing System	8.25.00
Sample Identification:	Tristan S1-5		Test Date: Wednesday, September 10, 2014	
Interface Type:	TT			
Sample Rate (pts/secs):	1.0000			
Crosshead Speed:	0.1500	mm/min	Humidity (%): 37	
Second Speed:	1.5000	mm/min	Temperature: 23 C	
Third Speed:	0.0000	mm/min		
Full Scale Load Range:	50.0000	kN		

Spec:U.T.S.(Min) MPa

Spec:Proof.(Min) MPa

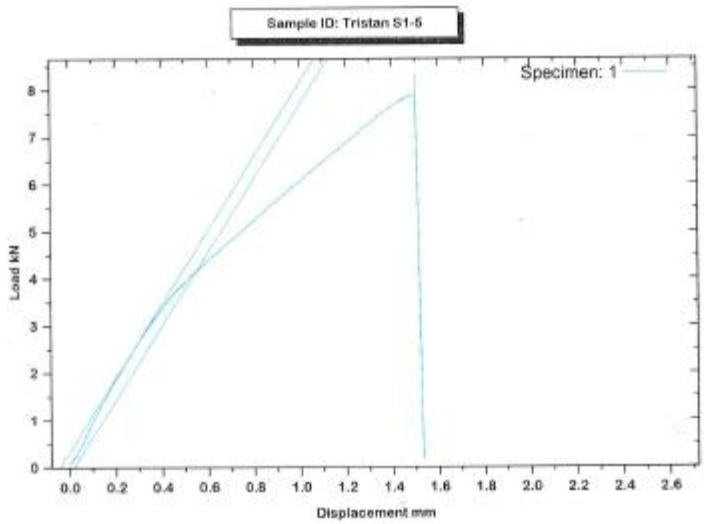
Spec:Elongation(Min)%

Spec:R of A(Min)

Sample comments:

Broke in gauge length.

	Load at Max.Load (N)	Stress at Max.Load (MPa)	Load at offset Yield 1 (N)	Stress at offset Yield 1 (MPa)	Diameter (mm)	Area (mm^2)
1	7879	401.3	4296	218.8	5.000	19.635



H.2.3 S1-6

DENEL AEROSTRUCTURES
PRODUCTION LABORATORY
Instron 3369/J4973 (50 kN)

Round test pieces
ASTM E8M

Test type: Tensile
Operator name: Elsa Pretorius
Sample Identification: Tristan S1-6
Interface Type: TT

Instron Corporation
Series IX Automated Materials Testing System 8.25.00
Test Date: Wednesday, September 10, 2014

Sample Rate (pts/secs): 1.0000
Crosshead Speed: 0.1500 mm/min
Second Speed: 1.5000 mm/min
Third Speed: 0.0000 mm/min

Full Scale Load Range: 50.0000 kN

Humidity (%): 37
Temperature: 23 C

Spec:U.T.S.(Min) MPa

Spec:Proof.(Min) MPa

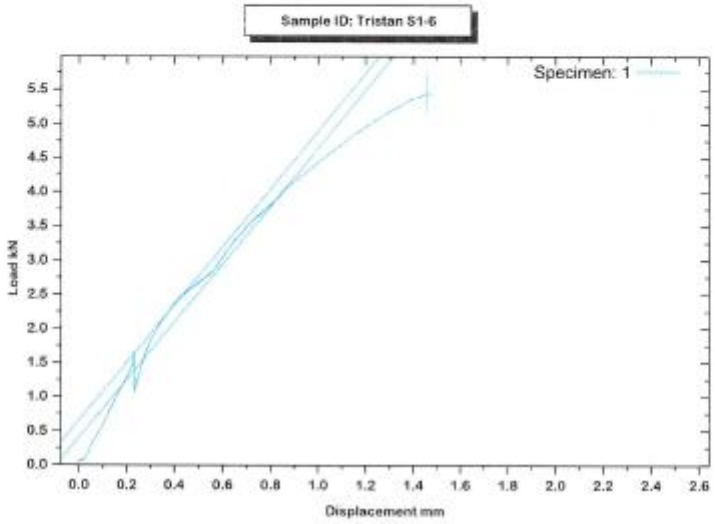
Spec:Elongation(Min)%

Spec:R of A(Min)

Sample comments:

Broke in gauge length.

	Load at Max.Load (N)	Stress at Max.Load (MPa)	Load at offset Yield 1 (N)	Stress at offset Yield 1 (MPa)	Diameter (mm)	Area (mm^2)
1	5445	277.3	3928	200.0	5.000	19.635



H.2.4 S1-12 to S1-15

DENEL SAAB AEROSTRUCTURES
PRODUCTION LABORATORY
INSTRON 5500R SN J4969

Round test piece Aluminium
ASTM B557

Test type: Tensile
Operator name: Walter Fransman
Sample Identification: TRISTAN S1-12-16
Interface Type: 5500

Instron Corporation
Series IX Automated Materials Testing System 8.25.00
Test Date: Tuesday, November 25, 2014

Sample Rate (pts/secs): 1.0000
Crosshead Speed: 1.0000 mm/min
Second Speed: 3.0000 mm/min
Third Speed: 0.0000 mm/min

Full Scale Load Range: 100.0000 kN

Humidity (%): 29
Temperature: 23 C

Spec: U.T.S. (Min)

Spec: Proof (Min)

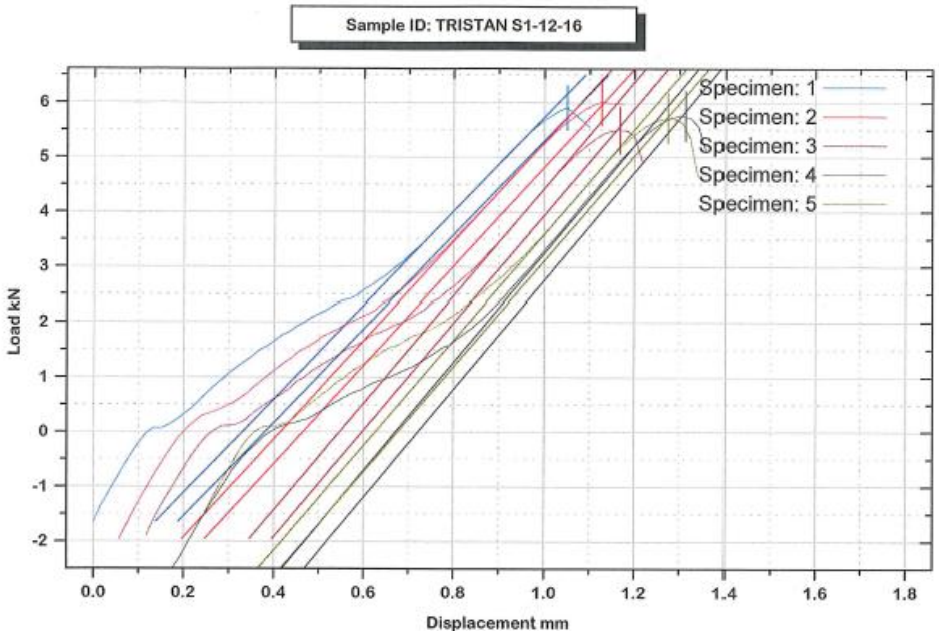
Spec:Elongation(Min)

Spec: R of A (Min)

Sample comments:

Tensile broke in gauge length.

	Load at Max.Load (N)	Stress at Max.Load (MPa)	Load at offset Yield 1 (N)	Stress at offset Yield 1 (MPa)	Diameter (mm)	Area (mm^2)
S1-12	5913	301.1	5556	282.9	5.000	19.635
S1-13	6016	302.7	5978	300.8	5.030	19.871
S1-16	5501	287.0	5501	287.0	4.940	19.167
S1-14	5755	295.4	5755	295.4	4.980	19.478
S1-15	5723	296.2	5723	296.2	4.960	19.322



H.2.5 S1-16 to S1-25

DENEL SAAB AEROSTRUCTURES
PRODUCTION LABORATORY
INSTRON 5500R SN J4969

Round test piece Aluminium
ASTM B557

Test type: Tensile
Operator name: Walter Fransman
Sample Identification: TRISTON Al 7075 1-8 1-10
Interface Type: 5500

Instron Corporation
Series IX Automated Materials Testing System 8.25.00
Test Date: Tuesday, March 03, 2015

Sample Rate (pts/secs): 1.0000
Crosshead Speed: 1.0000 mm/min
Second Speed: 3.0000 mm/min
Third Speed: 0.0000 mm/min

Full Scale Load Range: 100.0000 kN

Humidity (%): 29
Temperature: 23 C

Spec: U.T.S. (Min) MPa

Spec: Proof (Min) MPa

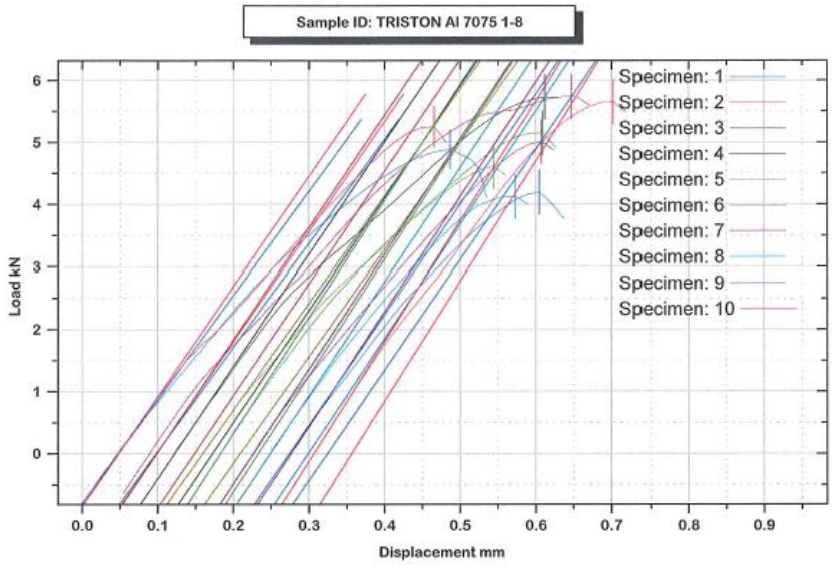
Spec:Elongation(Min)%

Spec: R of A (Min)

Sample comments: ALUMINIUM 7075 1-10 SPECIMENS

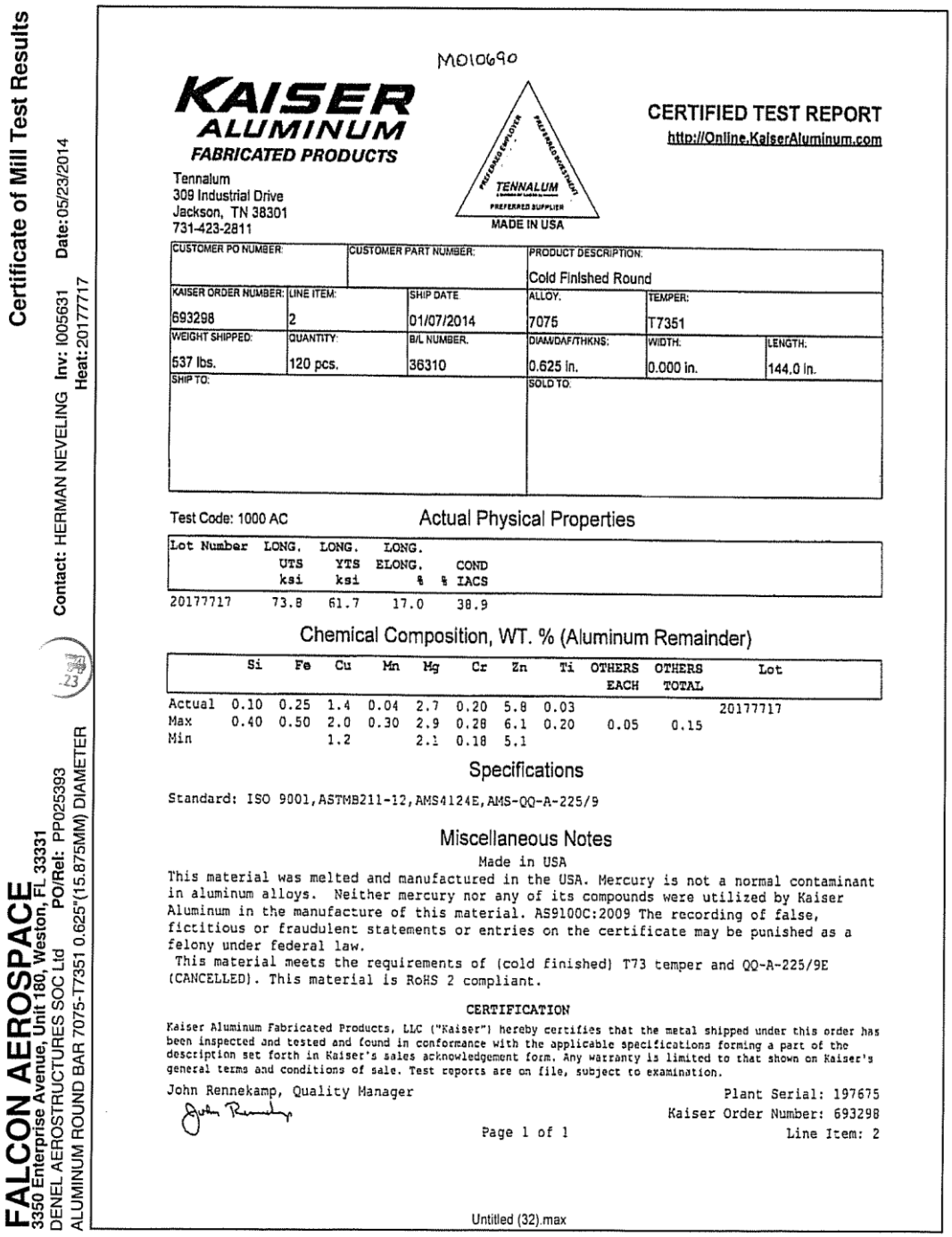
Tensile broke in gauge length.

	Load at Max.Load (N)	Stress at Max.Load (MPa)	Load at offset Yield 1 (N)	Stress at offset Yield 1 (MPa)	Diameter (mm)	Area (mm^2)
1	4885	253.9	3763	195.6	4.950	19.244
2	5256	269.8	3832	196.7	4.980	19.478
3	5742	291.3	3530	179.1	5.010	19.714
4	5732	286.2	3683	183.9	5.050	20.030
5	4614	234.1	3457	175.4	5.010	19.714
6	5156	269.0	3842	200.5	4.940	19.167
7	5014	257.4	3929	201.7	4.980	19.478
8	4128	206.1	3975	198.4	5.050	20.030
9	4201	211.4	3763	189.4	5.030	19.871
10	5655	296.2	5322	278.8	4.930	19.089



APPENDIX I MATERIAL DELIVERY TEST REPORTS

The material's used in the verification of the UHCF testing machine, have the following material data specifications, as received from the supplied Kaiser Aluminium.



Certificate of Mill Test Results

Contact: HERMAN NEVELING Inv: I005631 Date: 05/23/2014

Heat: 20177717

FALCON AEROSPACE
3350 Enterprise Avenue, Unit 180, Weston, FL 33331
DENEL AEROSTRUCTURES SOC Ltd PO/Rel: PP025393
ALUMINUM ROUND BAR 7075-T7351 0.625"(15.875MM) DIAMETER

M010690

KAISER ALUMINUM
FABRICATED PRODUCTS

Tennalum
309 Industrial Drive
Jackson, TN 38301
731-423-2811

TENNALUM
PREFERRED SUPPLIER
MADE IN USA

CERTIFIED TEST REPORT
http://Online.KaiserAluminum.com

CUSTOMER PO NUMBER:	CUSTOMER PART NUMBER:	PRODUCT DESCRIPTION:		
		Cold Finished Round		
KAISER ORDER NUMBER:	LINE ITEM: / SHIP DATE	ALLOY:	TEMPER:	
693298	2 / 01/07/2014	7075	T7351	
WEIGHT SHIPPED:	QUANTITY: / B/L NUMBER:	DIAM/DAF/THKNS:	WIDTH:	LENGTH:
537 lbs.	120 pcs. / 36310	0.625 in.	0.000 in.	144.0 in.
SHIP TO:		SOLD TO:		

Test Code: 1000 AC

Actual Physical Properties

Lot Number	LONG. UTS ksi	LONG. YTS ksi	LONG. ELONG. %	COND % IACS
20177717	73.8	61.7	17.0	38.9

Chemical Composition, WT. % (Aluminum Remainder)

	Si	Fe	Cu	Mn	Mg	Cr	Zn	Ti	OTHERS EACH	OTHERS TOTAL	Lot
Actual	0.10	0.25	1.4	0.04	2.7	0.20	5.8	0.03			20177717
Max	0.40	0.50	2.0	0.30	2.9	0.28	6.1	0.20	0.05	0.15	
Min			1.2		2.1	0.18	5.1				

Specifications

Standard: ISO 9001,ASTMB211-12,AMS4124E,AMS-QQ-A-225/9

Miscellaneous Notes

Made in USA

This material was melted and manufactured in the USA. Mercury is not a normal contaminant in aluminum alloys. Neither mercury nor any of its compounds were utilized by Kaiser Aluminum in the manufacture of this material. AS9100C:2009 The recording of false, fictitious or fraudulent statements or entries on the certificate may be punished as a felony under federal law.

This material meets the requirements of (cold finished) T73 temper and QQ-A-225/9E (CANCELLED). This material is RoHS 2 compliant.

CERTIFICATION

Kaiser Aluminum Fabricated Products, LLC ("Kaiser") hereby certifies that the metal shipped under this order has been inspected and tested and found in conformance with the applicable specifications forming a part of the description set forth in Kaiser's sales acknowledgement form. Any warranty is limited to that shown on Kaiser's general terms and conditions of sale. Test reports are on file, subject to examination.

John Rennekamp, Quality Manager

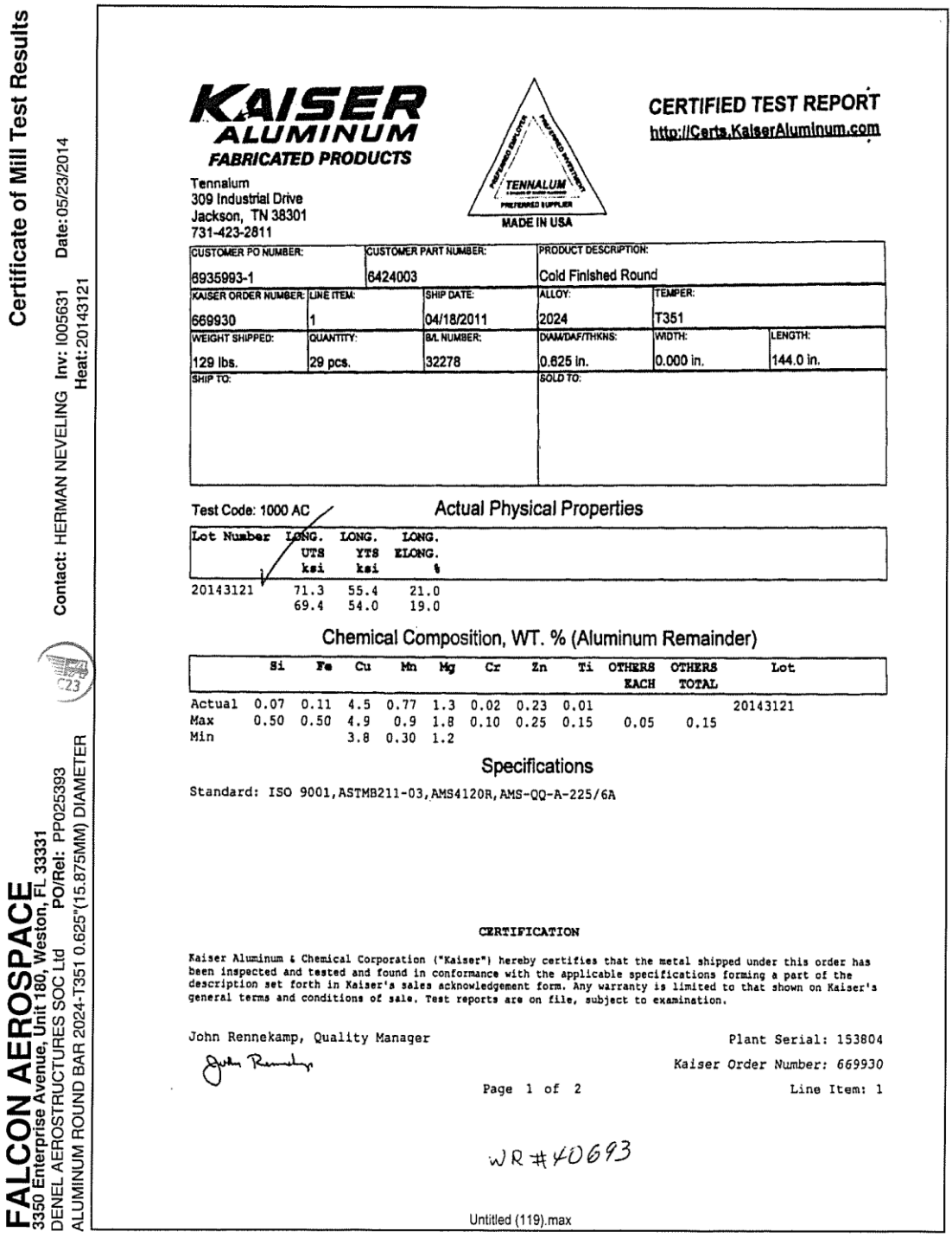
Plant Serial: 197675
Kaiser Order Number: 693298
Line Item: 2

Page 1 of 1

Untitled (32).max

Figure I-1: Certified Test report from Kaiser Aluminium for AL7075-T735

Similarly for AL2024 the following was received:



Certificate of Mill Test Results

Contact: HERMAN NEVELING Inv: I005631 Date: 05/23/2014

Heat: 20143121

FALCON AEROSPACE
3350 Enterprise Avenue, Unit 180, Weston, FL 33331
DENEL AEROSTRUCTURES SOC Ltd PO/Rel: PP025393
ALUMINUM ROUND BAR 2024-T351 0.625"(15.875MM) DIAMETER

KAISER ALUMINUM
FABRICATED PRODUCTS

Tennalum
309 Industrial Drive
Jackson, TN 38301
731-423-2811

TENNALUM
PREFERRED SUPPLIER
MADE IN USA

CERTIFIED TEST REPORT
http://Certs.KaiserAluminum.com

CUSTOMER PO NUMBER:	CUSTOMER PART NUMBER:	PRODUCT DESCRIPTION:		
6935993-1	6424003	Cold Finished Round		
KAISER ORDER NUMBER:	LINE ITEM:	SHIP DATE:	ALLOY:	TEMPER:
669930	1	04/18/2011	2024	T351
WEIGHT SHIPPED:	QUANTITY:	B/L NUMBER:	DIAM/DAF/THKNS:	WIDTH: / LENGTH:
129 lbs.	29 pcs.	32278	0.625 in.	0.000 in. / 144.0 in.
SHIP TO:			SOLD TO:	

Test Code: 1000 AC

Actual Physical Properties

Lot Number	LONG. UTS ksi	LONG. YTS ksi	LONG. ELONG. %
20143121	71.3	55.4	21.0
	69.4	54.0	19.0

Chemical Composition, WT. % (Aluminum Remainder)

	Si	Fe	Cu	Mn	Mg	Cr	Zn	Ti	OTHERS EACH	OTHERS TOTAL	Lot
Actual	0.07	0.11	4.5	0.77	1.3	0.02	0.23	0.01			20143121
Max	0.50	0.50	4.9	0.9	1.8	0.10	0.25	0.15	0.05	0.15	
Min			3.8	0.30	1.2						

Specifications

Standard: ISO 9001,ASTMB211-03,AMS4120R,AMS-QQ-A-225/6A

CERTIFICATION

Kaiser Aluminum & Chemical Corporation ("Kaiser") hereby certifies that the metal shipped under this order has been inspected and tested and found in conformance with the applicable specifications forming a part of the description set forth in Kaiser's sales acknowledgement form. Any warranty is limited to that shown on Kaiser's general terms and conditions of sale. Test reports are on file, subject to examination.

John Rennekamp, Quality Manager

Plant Serial: 153804
Kaiser Order Number: 669930
Line Item: 1

Page 1 of 2

WR#40693

Untitled (119).max

Figure I-2: Certified Test report from Kaiser Aluminium for AL2024-T351

APPENDIX J COST ANALYSIS

A cost analysis of the Rig was created based on the purchase cost and manufacturing costs of each component. The man hours used in the design are not added into the cost below.

Table J-1: Equipment cost estimate

Item No	Part	Supplier	Quantity	Comment	Cost per item [Rand]	Local Cost [Rand]
1	Frequency generator Branson 2.5 DCX	Ultraplast	1	COTS	R 84 803.40	R 84 803.00
2	CJ20 Piezo Electric Converter	Ultraplast	1	COTS	R 19 080.50	R 19 080.50
3	AL7075 Horn	Kaiser	1	Material	R 4 000.00	R 4 000.00
				Manufacture	R1 500.00	R1 500.00
4	AL2024 Specimens	Kaiser	21	Material	R 210.00	R 4 410.00
				Manufacture	R 300.00	R 6 300.00
5	AL7075 Specimens	Kaiser	21	Material	R 225.00	R 4 725.00
				Manufacture	R 300.00	R 6 300.00
6	Aluminium Booster	Ultraplast		COTS	R12 000.00	R12 000.00
7	Strain Gauges (axial)	Kyowa	2	10 per packet	R 1 697.00	R 3 394.00
8	Laser Vibrometer	CSIR	1	Leased	-	-
9	Test Support Material and manufacture	DAe Steel and Aluminium	1	Material	R 27 500.00	R 27 500.00
				Manufacture	R 12 500.00	R 12 500.00
10	Fastener and miscellaneous	Turbox Fasteners	18	COTS	R 16.50	R 297.00
Total						**R 173 309.50**

Notes:

COTS – consumable off the shelf (components are pruchases as working items from a given supplier).

All prices are exclusive of VAT and thus 14% will be added to the final costing.