UNIVERSITY OF CAPE TOWN
Department of Mechanical Engineering
Rondebosch, Cape Town, South Africa

Numerical Investigation of Dispersion in Hopkinson Pressure Bars

Reuben Ashley Govender

Submitted to the University of Cape Town in partial fulfillment of the degree of MSc in Mechanical Engineering

October 2005

Declaration

I, Reuben Ashley Govender, declare that this dissertation contains my own original work, except where reference and acknowledgment is made to contributions by others. I declare that this material has not been submitted for any purpose or examination to any other Department or University.

Signed this .....21..... day of ..OCTOBER 2005.....

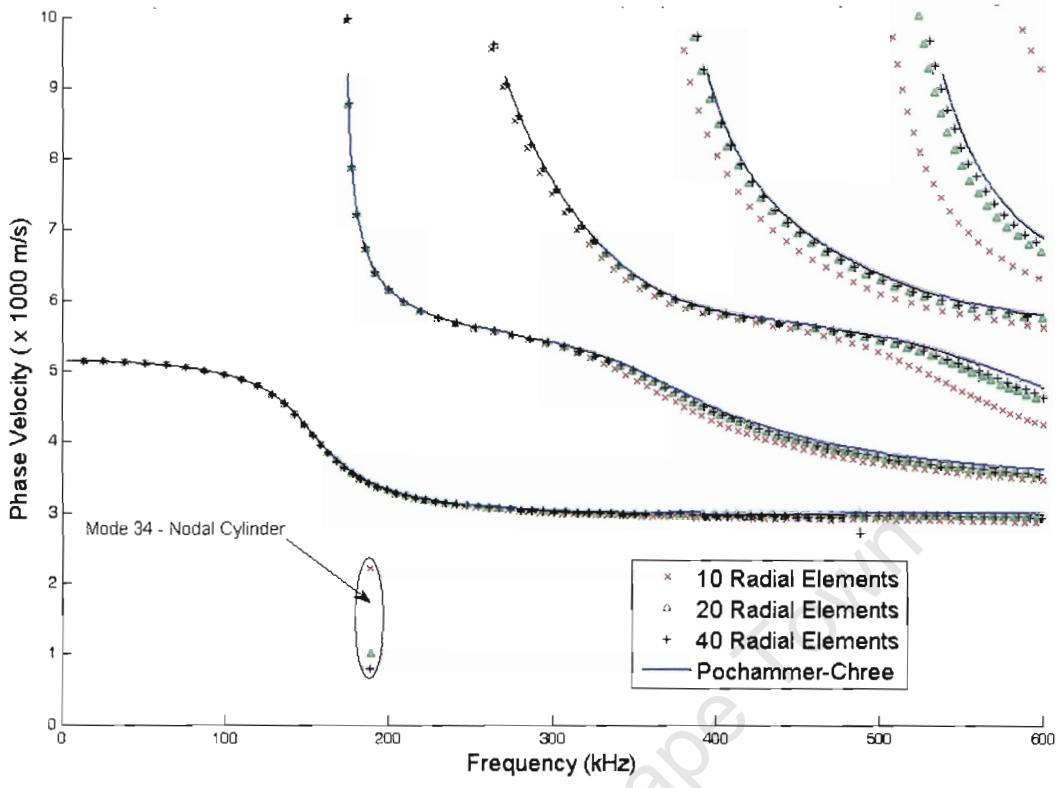
Signed by candidate

Abstract

The Hopkinson Pressure Bar (HPB) is used as a load-time or displacement-time transducer in impact or blast experiments. The Split Hopkinson Pressure Bar (SHPB) is the accepted form of material testing for strain rates between 10^2 s^{-1} and 10^4 s^{-1}. Explicit Finite Element Analysis (FEA) codes are increasingly used to model HPB experiments numerically, due to the complicated boundary conditions imposed by tensile and shear SHPB experiments. However, most publications on numerical modelling of HPB experiments have focussed on the response of the specimen and have paid very little attention to the modelling of the stress wave propagation in the cylindrical bars. This dissertation focuses on the numerical modelling of stress wave propagation in HPBs.

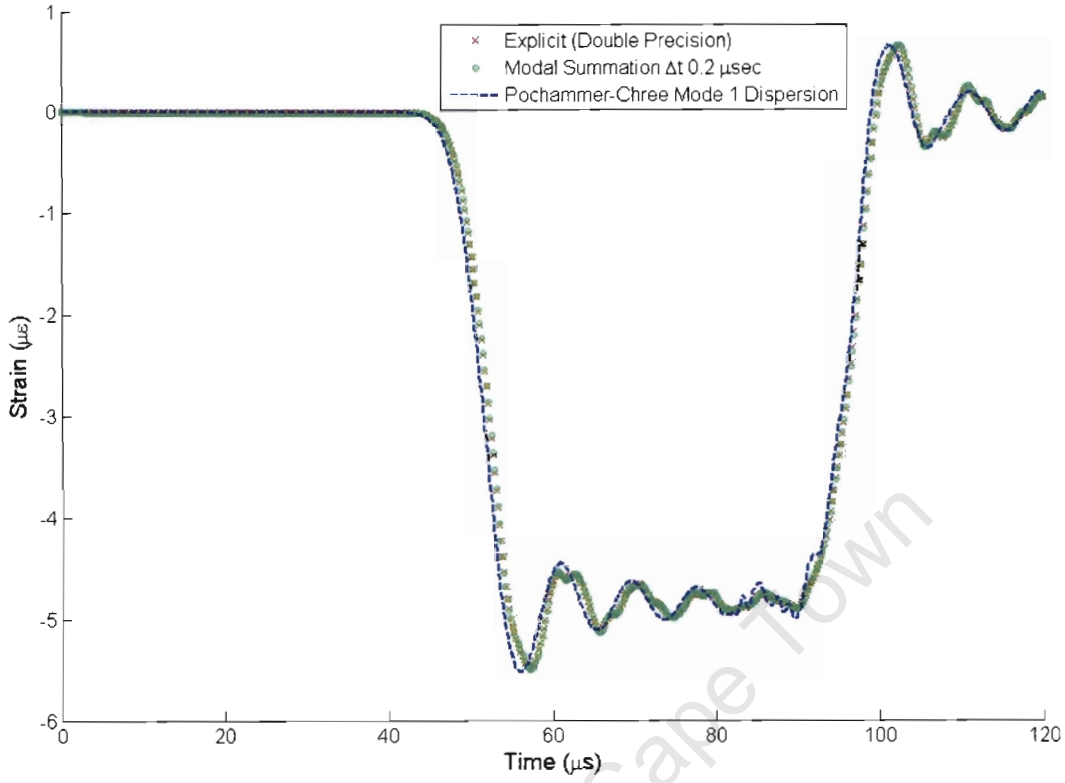
Stress wave propagation in HPBs is dispersive – the wave propagation velocities vary as a function of frequency. Hence the pulse changes shape as it propagates axially. The phase velocity-frequency relationship (C_p-f) derived from the Pochammer-Chree equations for an infinitely long cylindrical bar has been shown to be sufficiently accurate for conventional, compressive SHPB experiments. The Pochammer-Chree equations also permit the displacement of the bar to vary across the radius. Classical SHPB equations assume that the displacement across the radius is uniform.

As there is no analytical solution for vibration of a finite length cylinder, the starting point of the numerical analysis was to model an infinitely long cylinder in Abaqus and compare the results with the Pochammer-Chree equations. The C_p-f relationship derived from the numerical simulation showed very good agreement with the Pochammer-Chree prediction as the mesh was refined (Abstract Figure 1). The numerical and analytical predictions for displacement across the radius were almost identical.



Abstract Figure 1 Comparison of Pochammer-Chree and numerical predictions for Cp-f relationship

The dynamic response of a HPB to trapezoidally ramped pressure load, evenly distributed across one face, was modelled using Abaqus Modal Dynamics and Abaqus/Explicit and is shown in Abstract Figure 2. The numerical predictions showed no significant difference to each other, or when compared to a trapezoidal pulse which was analytically dispersed according to the Pochammer-Chree C_p-f relationship. The peak strains of both numerical predictions have a variation of less than 0.5% compared to the analytical prediction.



Abstract Figure 2 Comparison of numerical and analytical predictions for a trapezoidal pulse propagated 500mm

A Gauss windowed sine wave (GWS) was used as a loading function to investigate the effect of the artificial numerical damping present in Abaqus/Explicit on higher frequency stress wave propagation. The numerical predictions showed a decrease in amplitude of less than 8% and a slight phase shift, when compared to an analytically dispersed GWS, with a sinusoidal frequency f_s of 75 kHz. The numerical prediction using a GWS, with f_s = 200 kHz, differed significantly in shape and amplitude when compared to an analytically dispersed GWS.

This study established the mesh density requirements for accurately modelling stress wave propagation in HPBs, where the propagation was limited to the 1st Pochammer-Chree mode. The artificial numerical damping in Explicit FEA codes has a significant effect on vibrations with frequencies higher than 100 kHz. Hence suitable care should be exercised when modelling HPB experiments where the load contains frequencies higher than 100 kHz.

Acknowledgements

I wish to thank the following people for their contributions and assistance during this project.

Mr. T.J. Cloete, who co-supervised the project, for his invaluable technical knowledge and insight of this work, without which this project wouldn't have begun.

Prof. G.N. Nurick, who co-supervised the project, for his advice, support and understanding of my needs as a student, which helped bring this project to a successful conclusion.

All the other members of BISRU, for their support, advice and tolerance of me.

The staff of the Department of Mechanical Engineering, University of Cape Town, for their support.

The NRF and Denel, for their financial support of this project.

Mr. S. Marais & Mr. B. Prenzlow, for their advice on programming and Fourier analysis.

Mr. V. Balden, Mr. D. Bonorchis and Mr. H. Bowles for their assistance with Abaqus.

My friends, who listened even when they didn't understand what I was saying, and understood when I needed them for emotional support.

My family, who listened and encouraged and whose love and support, are one of the few unconditional, unquestionable things in my life.

My God and creator, who has done more for me than I can ask or imagine.

Table of Contents

1 Introduction 1

2 Background and Theory 4

2.1 Introduction 4
2.2 Origins of the Hopkinson Pressure Bar 4
2.3 One Dimensional Wave Theory 4
2.4 Stress Wave Propagation and Dispersion 5
2.5 The Split Hopkinson Pressure Bar (SHPB) and Material Characterisation 6
2.6 Dispersion Correction in the Frequency Domain for arbitrary pulses 10
2.7 Numerical Simulation of HPB Problems 12
2.7.1 Finite Difference / Volume Schemes 12
2.7.2 Explicit Finite Element Analyses of HPB Experiments 13

3 The Analytical Solution For Vibration of Infinite Cylinders 15

3.1 The Pochammer-Chree solution 15
3.2 Dispersion 20
3.2.1 Effect of different parameters on phase velocity-frequency relationship 22
3.3 Variation of displacement over the radius 25
3.4 Group Velocity 26
3.5 Dispersion Correction Map 28

4 Numerical Analysis 29

4.1 Introduction 29
4.2 Numerical Simulation of Dynamic Problems 30
4.2.1 Modal Summation Method 30
4.2.2 Explicit Finite Element Analysis 32
4.2.3 Comparison of Modal Summation and Explicit FEA 34
4.3 FE Simulation of Vibration of an Infinitely Long Cylindrical Bar 36
4.3.1 Material 37
4.3.2 Geometry 37
4.3.3 Boundary Conditions 37

4.3.4 Mesh 37
4.3.5 Post-Processing of Abaqus Results 38
4.4 Results & Discussion 41
4.4.1 Axial Mesh Dependence 41
4.4.2 Radial Mesh Dependence 42
4.4.3 Radial Bias Factor Dependence 43
4.4.4 Summary of Mesh Density Study 44
4.4.5 Mesh size criterion 44
4.5 Phase Velocity – Frequency Relationship For a Finite Length Bar 46
4.5.1 Material, Geometry and Mesh 46
4.5.2 Boundary Conditions 46
4.5.3 Post-Processing of Abaqus Results 46
4.5.4 Results and Discussion 48
4.5.5 Comparison of Numerical Predictions for C_p-f Relationship for Infinite and Finite Length Cylinders 51
4.6 Dynamic Simulations of Stress Wave Propagation in Pressure Bars 53
4.6.1 Model Inputs 53
4.6.2 Comparison of Modal Summation and Explicit Results 54
4.7 Artificial Damping in Numerical Modelling 60
4.7.1 Parametric investigation of numerical damping 62
4.7.2 Higher Frequency Effects 66
4.8 Concluding Remarks 71

5 Conclusions 72

6 Recommendations 74

7 References 76

Appendix A Computer Code for Solving Pochammer-Chree Dispersion Equation 79

A.1 Generation of multi-mode phase velocity maps for dispersion correction 79
A.2 Modified Dispersion Correction Code 94

Appendix B Modal Analysis of Infinite and Finite Cylindrical Bars using Abaqus 98

B.1 Mode Extraction - Typical Input Data Deck 98

B.2 Extraction of Axial Displacement Along Outer Radius 101
B.3 Determining Wavelength From Axial Displacement Data 103

Appendix C FEA of dynamic loading of bars 108

C.1 Trapezoidal Pressure Load – Explicit Analysis Input Data Deck 108
C.2 Trapezoidal Pressure Load – Modal Summation Method Input Data Deck 112

Appendix D FEA of Oscillatory Loading of Bars 116

D.1 User Defined Load for Gauss Windowed Sine Wave 116
D.2 Typical Input Data Deck for GWS Loading 117

List of Figures

Chapter 1

Figure 1-1 Typical SHPB Setup 1
Figure 2-1 Layout of a SHPB for material characterisation 6
Figure 2-2 Typical strain gauge signals from a SHPB test [13] 7
Figure 2-3 Gauss windowed sine wave 13
Figure 3-1 Change of dispersion equation form with respect to wavelength 17
Figure 3-2 Flowchart of solver algorithm 18
Figure 3-3 Scaled phase velocity vs scaled wavelength for ν = 0.26 19
Figure 3-4 Dispersion of a trapezoidal pulse after 500mm of axial propagation 20
Figure 3-5 Pochammer-Chree phase velocity as a function of frequency, d = 20mm 22
Figure 3-6 Effect of bar diameter on first Pochammer-Chree mode, ν=0.26 23
Figure 3-7 Effect of Poisson's ratio on first Pochammer-Chree mode, d = 20mm 24
Figure 3-8 Variation of axial displacement as a function of radius, for different modes at different frequencies 25
Figure 3-9 Scaled group velocity curves derived from Pochammer-Chree phase velocities 26
Figure 3-10 Phase velocity frequency curve for dispersion correction of a 20mm diameter steel bar, E = 207 GPa, ν = 0.26 28
Figure 4-1 Deformed vibrational mode shape 36
Figure 4-2 Abaqus axial displacement profile compared to a true sine wave of wavelength 40 mm (180 kHz) 39
Figure 4-3 Data flow while determining C_p-f relationship from Abaqus results 40
Figure 4-4 Effect of axial mesh density on numerical C_p-f prediction 41
Figure 4-5 Effect of radial mesh density on numerical phase velocity prediction 42
Figure 4-6 Effect of radial bias factor on numerical phase velocity prediction 43
Figure 4-7 Axial displacement profile of a finite bar (unconstrained ends) for mode correspoding to 180 kHz 47

Figure 4-8 Numerical phase velocity frequency prediction for a 200mm bar 48
Figure 4-9 Axial displacement profile of mode 17 (153 kHz) 49
Figure 4-10 Numerical phase velocity - frequency prediction for a 500mm bar 50
Figure 4-11 Numerical phase velocity - frequency prediction for a 1000mm bar 51
Figure 4-12 Pressure loading of face 53
Figure 4-13 Abaqus Modal Summation and Explicit predictions for a trapezoidal pressure loading 55
Figure 4-14 Comparison of Explicit (Double Precision), Modal Summation and Pochammer-Chree 1st mode predictions for propagation of a trapezoidal wave 56
Figure 4-15 Enlarged view of oscillations from Figure 4-14 57
Figure 4-16 Comparison of numerical predictions with Pochammer-Chree 2nd mode correction for propagation of a trapezoidal pulse 58
Figure 4-17 83 kHz Gauss Windowed Sine Wave 61
Figure 4-18 Amplitude spectra of a trapezoidal pulse and 83 kHz GWS 62
Figure 4-19 Comparison of numerical and analytical predictions for 75 kHz GWS propagated 1000mm 64
Figure 4-20 Enlarged view of peaks from Figure 4-19 64
Figure 4-21 Comparison of numerical and 1st Pochammer-Chree mode analytical prediction for a 200 kHz GWS propagated 1000mm 68
Figure 4-22 Comparison of numerical prediction with no damping, and 1st and 2nd Pochammer-Chree mode dispersion correction for a 200 kHz GWS propagated 1000mm 69

List of Tables

Table 4-1 Mesh Density Parameters 38
Table 4-2 Frequency, phase velocity and mesh geometry at 1% variation between numerical and analytical predictions 45
Table 4-3 Computer specifications 59
Table 4-4 GWS Parameters 61
Table 4-5 HPB Model parameters 63
Table 4-6 Damping parameters for different simulations 63
Table 4-7 Peak strains from Simulations 1 and 2 65
Table 4-8 Summary of viscosity parameter investigation 66

List of Symbols and Abbreviations

Symbol	Description	Units
a	Outer radius of bar	[m]
c	Damping	[Ns.m^{-1}]
C_0	Fundamental wave velocity	[m.s^{-1}]
C_p	Phase velocity	[m.s^{-1}]
C_g	Group Velocity	[m.s^{-1}]
D or d	Diameter of bar	[m]
E	Elastic Modulus	[Pa]
f	Frequency	[Hz]
f_s	Sinusoidal frequency in a Gauss windowed sinusoid	[Hz]
$\underline{F}(t)$	External load vector or any function of time	
$F(\omega)$	Fourier Component	
k	Stiffness	[N.m^{-1}]
m	Mass	[kg]
r	Radius	[m]
t	time	[s]
u	Displacement	[m]
x	Generalised coordinate (unless otherwise stated)	[m]
z	Axial direction of bar	[m]
ε	Strain	
φ	Spatially dependent shape function for FE formulation	
ϕ	Mass normalised Eigen vector for modal summation	
ν	Poisson's Ratio	
ρ	Density	[kg.m^{-3}]
σ	Stress	[MPa]
Λ	Wavelength	[m]

Symbols with an underline (e.g. $\underline{x}$) denote vector quantities.

Capitalised, bold symbols (e.g. **M**) denote a matrix or tensor quantity.

1 Introduction

This dissertation documents the investigation into the numerical modelling of wave propagation in Hopkinson Pressure Bars (HPB), using the Finite Element code Abaqus. It describes the origin and effects of dispersion in Hopkinson Pressure Bars. It documents the solution of the analytical equations and the development of a dispersion correction method that accounts for higher Pochammer-Chree modes of dispersion. It compares the analytical and numerical solutions for vibration of an infinitely long cylindrical bar. It compares the accuracy of dispersion modelled in an infinitely long cylinder using the modal summation method with the Pochammer-Chree analytical solution for dispersion. The modal summation solution is then used to benchmark the accuracy of explicit Finite Element Analysis in the simulation of HPB experiments. This work forms part of research conducted by the Blast Impact and Survivability Research Unit (BISRU), at the Department of Mechanical Engineering, University of Cape Town (UCT). This research encompasses response of structures to impact and explosive loading and high strain rate material modelling.

The Hopkinson Pressure Bar (HPB) was first introduced by, and named after, Hopkinson [1] as a means of measuring the peak pressure and duration of loading due to explosive detonation or impact of bullets. Davies [2] instrumented a HPB with capacitative sensors, permitting the measurement of the pressure time history. The HPB was adapted to high strain rate material testing by Kolsky [3,4]. A relatively short specimen of the material being tested was sandwiched between two relatively long HPBs and loaded impulsively. This apparatus, shown in Figure 1-1, is referred to in the literature as a "Kolsky Bar" or a "Split Hopkinson Pressure Bar" (SHPB). A more complete description of the general principles and operation of the SHPB is given in Chapter 2.

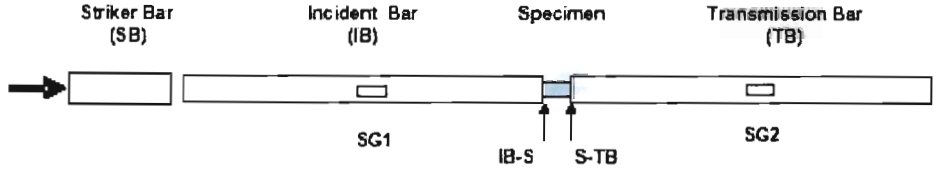


Figure 1-1 Typical SHPB Setup

1 Introduction

The application of the SHPB to high strain material testing permits testing at strain rates approaching 10^4 s^{-1}, for compressive tests, and up to 5 x 10^3 s^{-1} for tension and torsional tests. Conventional screw driven or servo-hydraulic tensile testers can achieve a strain rate of 1 s^{-1} and high capacity servo-hydraulic or pneumatic testers can achieve strain rates of up to 10^2 s^{-1}. The properties of materials at high strain rates are of interest to the following applications:

- Automotive impacts and crashworthiness
- Aerospace impacts, either due to crashes or foreign object damage such as bird strike
- Defense applications such as explosive or projectile loading of military vehicles.
- High speed material forming
- Finite Element or other numerical simulation of any of the above applications where a material model, such as Cowper-Symonds or Johnson-Cook, is necessary to describe the material behaviour over the applicable strain rate range.

Analysis of the data gathered from a HPB requires analysis of the stress wave which propagates from the impact end of the bar. Pochammer [5] and Chree [6] arrived independently at the same solution for wave propagation in cylindrical bars. This solution predicts that the wave propagation velocity is a function of the wavelength. Hence a pulse composed of different wavelengths or frequencies, which is true for all non-sinusoidal pulses, will change shape as it propagates down the bar due to the differing propagation velocities and subsequent phase shifts. The lengthening of the duration of the pulse is known as dispersion and the change in amplitude as attenuation[1]. Hereafter, reference to dispersion implies that attenuation is also present. In order to accurately determine the pulse shape at the specimen or impact face, dispersion must be accounted for. Bancroft [7] and Davies [2] presented the numerical solution to the Pochammer-Chree dispersion equations.

[1] Attenuation is often associated with damping, which does cause a decrease is amplitude of oscillations. However, attenuation in the strictest sense refers to a decrease in amplitude, whether caused by damping or phase shifting of oscillations.

Gorham [8] and Follansbee and Frantz [9] transformed the pulse from it original time domain to its Fourier coefficients in the frequency domain. Dispersion correction was applied in the frequency domain, and the corrected pulse is obtained by applying an inverse Fourier transform to the corrected frequency components.

BISRU utilises HPBs both in material testing in the conventional SHPB rig [10], and measuring plate response in ballistic pendulum rigs [11], subjected to blast loading.

It has been shown that the Pochammer-Chree solution may be applied to dispersion correction of compressive SHPB signals with acceptable accuracy ([8], [9], [10]). The use of HPBs for tensile and shear testing introduce boundary conditions, which differ greatly from those imposed by a compressive SHPB experiment. However, the effect on dispersion of the geometry introduced by the fixtures used for tensile and shear SHPB tests is unknown. As no analytical solution is available for a finite length bar, it is necessary to model these conditions numerically. While various compression SHPB experiments have been simulated numerically, the accuracy with which these simulations capture dispersion has not been verified. This dissertation documents the investigation into modelling of wave propagation in HPBs using numerical techniques. The accuracy with which dispersion can be captured by a numerical model is assessed. The effects of various parameters in Abaqus on wave propagation were investigated.

2 Background and Theory

2.1 Introduction

This chapter discusses the origins of the HPB as an instrument in impact and explosive testing and material characterisation. It examines the theory necessary to analyse the strain-time data obtained from a typical HPB and apply this data to the experiment in question, be it material testing or high rate load measurement.

2.2 Origins of the Hopkinson Pressure Bar

Hopkinson [1] measured the peak pressure and duration of impulsive loading with long cylindrical bars. The loading was achieved by firing a standard rifle bullet at the bar, or detonating a gun-cotton charge a short stand off distance from the end of the bar. The peak pressure and duration of the compressive wave generated by the impact were obtained by measuring the momentum transmitted to short end pieces weakly attached to the end of the long cylindrical bar. However, this method could not determine the shape of the wave (its pressure-time history). It was assumed that the wave propagation in the bars obeyed one dimensional wave theory, as the axial length of the bars was much greater than the diameter.

2.3 One Dimensional Wave Theory

The propagation of an elastic wave, which propagates only axially and not radially, down a bar has been thoroughly investigated and documented [12,14]. One dimensional waves are governed by the well known differential equation:

$$\frac{\partial^2 u}{\partial z^2} = \frac{1}{C_o^2}\frac{\partial^2 u}{\partial t^2} \qquad \textbf{Eq 2-1}$$

The general form of the solution to Eq 2-1 is

$$u(z,t) = f(z - C_0 t) + g(z + C_0 t) \qquad \textbf{Eq 2-2}$$

where f and g give the form of the wave travelling in the positive and negative directions.

C_0 is the fundamental wave speed in the material and is given by

$$C_0 = \sqrt{\frac{E}{\rho}} \qquad \textbf{Eq 2-3}$$

One dimensional wave theory is considered sufficiently accurate for HPBs where the radial dimension is one order of magnitude smaller than the wavelength of the excitation.

2.4 Stress Wave Propagation and Dispersion

One dimensional wave theory is limited by the assumption that there is no variation in the directions perpendicular to the wave propagation (radial for bars) and that the wave does not change shape in the direction of propagation. Pochammer [5] and Chree [6] independently arrived at the same solution for the equation governing transmission of waves along an infinite bar of finite, known diameter, where displacement may vary radially. This result, which is discussed in greater detail in Chapter 3, showed that the propagation velocity (phase velocity C_p) of a disturbance consisting of a single sinusoidal wave is not just a function of the material as assumed by one dimensional theory, but is also a function of the wave length λ, or frequency. In addition, different modes of vibration may be excited by the same frequency and propagate at different phase velocities.

Bancroft [7] and Davies [2] presented numerical solutions to the Pochammer-Chree equations. These showed relationship between frequency and phase velocity, and the variation of axial displacement across the radius of the bar. Davies showed the change in shape of a pulse as it propagates axially (dispersion) by representing a trapezoidal pulse with its Fourier series and phase shifting each term according to the phase velocity. Davies made use of cylindrical and parallel plate capacitors to obtain the deflection of the end of the transmitter bar as a function of time. As the deflection can be related to the stress in the bar, it was possible to obtain the stress-time history of a pulse in the bar.

Davies also noted that the energy of a pulse consisting of different wavelengths will be transmitted at a different velocity to the phase velocity of the individual wavelengths. This velocity is termed the "group velocity" (C_g) and is determined by:

$$C_g = C_p - \lambda \frac{dC_p}{d\lambda} \qquad \textbf{Eq 2-4}$$

2.5 The Split Hopkinson Pressure Bar (SHPB) and Material Characterisation

Kolsky [3] developed a method of material characterisation by sandwiching a relatively short specimen between two long HPBs (Figure 2-1).

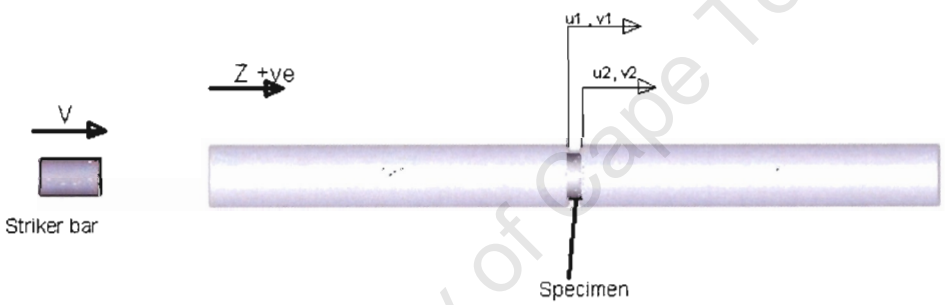


Figure 2-1 Layout of a SHPB for material characterisation

A compressive incident pulse was generated in the incident bar by impact with a striker bar. The velocity of the striker may be measured by means of light traps. The velocity of the striker bar was limited so that the resultant compressive stress wave in the incident bar does not exceed the elastic limit of the bar. The long cylindrical bars are supported by bearings which permit the bars to move freely in the axial direction but prevent radial movement. When the incident pulse reaches the specimen, a portion is transmitted to the specimen and the remainder in reflected. The specimen length is much shorter than the duration of the compressive loading pulse, so it approaches stress equilibrium rapidly due to the multiple reflections of the wave within the specimen. An excellent description of the SHPB and it background theory was written by Gray [12].

Most modern HPBs make use of longitudinal strain gauges, typically mounted mid-way along the bars, to measure the incident $\varepsilon_I(t)$, reflected $\varepsilon_R(t)$ and transmitted $\varepsilon_T(t)$ strain. The analogue signals from the strain gauges are amplified and digitised by suitable equipment [Gorham [8], Follansbee & Frans [9], Marais [10]]. Typical examples of the strain gauge signals captured in SHPB experiment are shown in Figure 2-2.

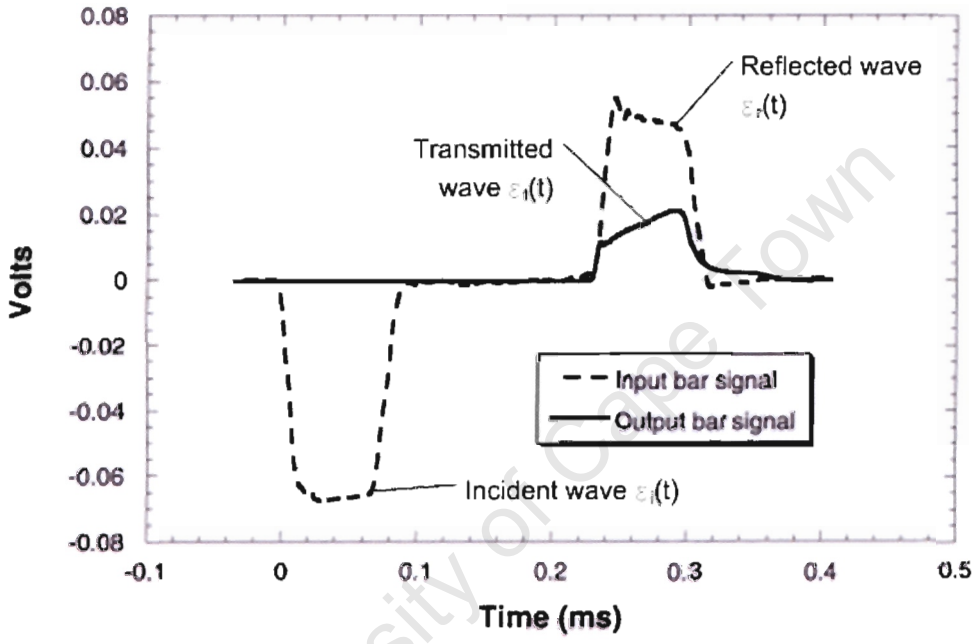


Figure 2-2 Typical strain gauge signals from a SHPB test [13]

The following equations describe how the specimen stress – strain relationship is found from experimentally measured signals and are a summary of those presented by Gray [12].

The axial strain ε_z may be related to the axial deflection u_z by:

$$\varepsilon_z = \frac{\partial u_z}{\partial z} \qquad \textbf{Eq 2-5}$$

Applying Eq 2-2 to Eq 2-5 yields the strain in the incident bar:

$$\varepsilon_1 = \varepsilon_i + \varepsilon_r = \frac{\partial f}{\partial z} + \frac{\partial g}{\partial z} \qquad \textbf{Eq 2-6}$$

Similarly, if the wave in the transmitter bar is given by $h(z-C_0t)$, the transmitter strain is

$$\varepsilon_t = \frac{\partial h}{\partial z} \qquad \textbf{Eq 2-7}$$

Differentiating with respect to time gives the velocities of the faces of the bars.

$$\begin{aligned} v_1 &= \frac{\partial}{\partial t}(u) \\ &= \frac{\partial}{\partial t}(f(z - C_0 t) + g(z + C_0 t)) \\ &= C_0(-\varepsilon_i + \varepsilon_r) \end{aligned} \qquad \textbf{Eq 2-8}$$

and

$$v_2 = \frac{\partial h}{\partial t} = C_0 \varepsilon_t \qquad \textbf{Eq 2-9}$$

Given an initial specimen length l_o and assuming that the ends of the bars remain in contact with the specimen, the instantaneous specimen length is found by

$$l_s(t) = l_0 - \int_0^t (v_1 - v_2)dt \qquad \textbf{Eq 2-10}$$

The instantaneous true strain in the specimen is

$$\varepsilon(t) = \ln\left(\frac{l_s(t)}{l_0}\right) \qquad \textbf{Eq 2-11}$$

The strain rate is

$$\dot{\varepsilon} = \frac{\partial \varepsilon}{\partial t} = \frac{v_1 - v_2}{l_s(t)} \qquad \textbf{Eq 2-12}$$

The forces acting on the faces of the specimen are F_1 on the input bar face and F_2 on the transmitter bar face.

$$F_1 = A_t E_t (\varepsilon_i + \varepsilon_r)$$
$$F_2 = A_t E_t \varepsilon_t \qquad \textbf{Eq 2-13}$$

The flow stress in the specimen may be accurately calculated once F_1 and F_2 are equal (i.e. the specimen has reached equilibrium). The time taken for the specimen to reach equilibrium is termed "ringing up". By assuming that the specimen volume remains constant the instantaneous specimen area is given by

$$A_s(t) = \frac{A_0 l_0}{l_s(t)} \qquad \textbf{Eq 2-14}$$

Hence the true stress in the specimen is

$$\sigma_s(t) = \frac{F_2}{A_s(t)} \qquad \textbf{Eq 2-15}$$

Thus by measuring the strains in the bars $\varepsilon_I(t)$ and $\varepsilon_T(t)$, one may calculate the true stress $\sigma_s(t)$ and strain $\varepsilon_s(t)$ in the specimen.

2.6 Dispersion Correction in the Frequency Domain for arbitrary pulses

Gorham [8] and Follansbee and Franz [9] presented a method for dispersion correction of any arbitrary pulses by correcting for a phase shift in the frequency domain. Any non-periodic pulse in the time domain *f(t)* may be transformed to its Fourier components *F(ω)* in the frequency domain by [15]:

$$f(t) \Leftrightarrow F(\omega) = \int_{-\infty}^{\infty} f(t) e^{-j\omega t} dt \qquad \textbf{Eq 2-16}$$

F(ω) may be transformed to the time domain by:

$$f(t) = \frac{1}{2\pi} \int_{-\infty}^{\infty} F(\omega) e^{j\omega} d\omega \qquad \textbf{Eq 2-17}$$

If a signal is phase shifted in the time domain, then its shifted component in the frequency domain is given by :

$$f(t+\tau) \Leftrightarrow F(\omega) e^{-j\omega\tau} \qquad \textbf{Eq 2-18}$$

Dispersions may be described by the various frequency components of the pulse changing their relative phase. The Pochammer-Chree solution, discussed in greater detail in Chapter 3, may be used to obtain the relation between phase velocity C_p and frequency ω. The phase shift of a given frequency component that has travelled a known axial distance z is given by (Marais [10]):

$$\tau_p(\omega) = \frac{z}{C_p(\omega)} \qquad \textbf{Eq 2-19}$$

The dispersed frequency component $F_p'(\omega)$ is then

$$F_p'(\omega) = F_p(\omega) e^{-j\omega\tau_p(\omega)} \qquad \textbf{Eq 2-20}$$

The dispersed pulse in the time domain $f'(t)$ is given by

$$f'(t) = \frac{1}{2\pi} \int_{-\infty}^{+\infty} F_p'(\omega) e^{-j\omega} d\omega \qquad \textbf{Eq 2-21}$$

The equations presented here are in the Fourier integral form, which require piecewise continuous functions. Dispersion correction is required for correction of digital signals obtained from experiments. As the input is digital data rather than an analytically defined function, the Discrete Fourier Transform (DFT) is used rather than the Fourier integral. The DFT is defined for a set containing N discrete data points and is

$$F_n = \sum_{k=0}^{N-1} f_k e^{\frac{-j2\pi nk}{N}} \quad \text{for } 0 \le n \le N\text{-}1 \qquad \textbf{Eq 2-22}$$

The discrete data points in the time domain are obtained by operating on the frequency domain components:

$$f_k = \frac{1}{N} \sum_{n=0}^{N-1} F_n e^{\frac{j2\pi nk}{N}} \quad \text{for } 0 \le n \le N-1 \qquad \textbf{Eq 2-23}$$

Previous dispersion correction algorithms required the approximation of the measured pulse with an analytically defined shape, typically a trapezoidal or step function pulse. The accuracy of the dispersion correction was then strongly dependent on how well the incident pulse could be approximated with an analytically defined shape. The method proposed by Gorham [8] and Follansbee and Franz [9] permitted dispersion correction of any arbitrary pulse, regardless of its shape, which could be converted to digital data. Gorham's work also reduced the oscillations on the specimen stress-strain curve obtained from SHPB experiments.

Gong, Malvern and Jenkins [16] applied a frequency domain dispersion correction method to signals captured in SHPB testing of concrete. Bancroft's [7] numerically obtained solution to the Pochammer-Chree equations was used as a reference for the phase velocity frequency relationship. A corrected strain time signal was compared to

a signal captured at another strain gauge station on the same bar. Gong *et al* showed that correction using the first Pochammer-Chree mode gave good agreement between the corrected and captured signals, for the given loading.

2.7 Numerical Simulation of HPB Problems

Due to the absence of an analytical solution for wave propagation in a finite length cylindrical bar, many attempts have been made to simulate various HPB problems using numerical techniques. A more detailed account of explicit FEM and numerical modelling techniques is discussed in Chapter 4.

2.7.1 Finite Difference / Volume Schemes

Bertholf [17] used a finite difference, forward approximation scheme to simulate elastic wave propagation in finite cylindrical bars. The simulation was axisymmetric and numerically solved Cauchy's equation of motion at discrete points on a radial-axial mesh for a given set of boundary conditions. A plane harmonic displacement was applied to the end of the bar and allowed to propagate axially. The numerical solution was compared to the predictions using the Pochammer-Chree solution at a distance away from the bar end, where the Pochammer-Chree solution may be assumed to be valid. Bertholf compared his numerical solution to experimental data provided by Miklowitz and Niesewanger [18], and showed good qualitative agreement.

Bertholf and Karnes [19] used a similar code to simulate a full SHPB compression experiment. The simulation was used to demonstrate the effect of specimen-bar friction and radial inertia on the accuracy of measured specimen strain rate sensitivity. The HPB mesh consisted of 12 radial elements along a 25.4mm (1") radius and 160 elements along a 508mm (20") length. The results of their analyses were used to suggest suitable specimen aspect ratios and lubrication in order to minimize radial inertia and friction effects.

2.7.2 Explicit Finite Element Analyses of HPB Experiments

Tyas and Watson [20] used an explicit (forward time stepping) Finite Element Analysis to simulate the response of a bar to a known oscillating input function. The input function was a Gauss windowed sinusoid [Figure 2-3], which permitted the control of the frequency content of the input pulse. A correction factor was derived from the Pochammer-Chree solution that related average axial displacement over the radial cross-section to the axial displacement at the outer radius. An additional "Dynamic Elastic Modulus" factor was derived that accounted for the first Pochammer-Chree mode of dispersion. This modified dispersion correction algorithm was used to correct the surface axial displacement predicted by the explicit FEA back to the input face. The resulting corrected displacement – time curve was used to calculate the applied load on the input face. The calculated input load was compared to the known input load of the simulation. The modified dispersion correction algorithm was shown to give good agreement between the known input load and the calculated input load. Small discrepancies between the two were attributed to higher mode dispersion, attenuation of higher frequency components of the input pulse or the invalidity of the Pochammer-Chree solution due to insufficient wave propagation length.

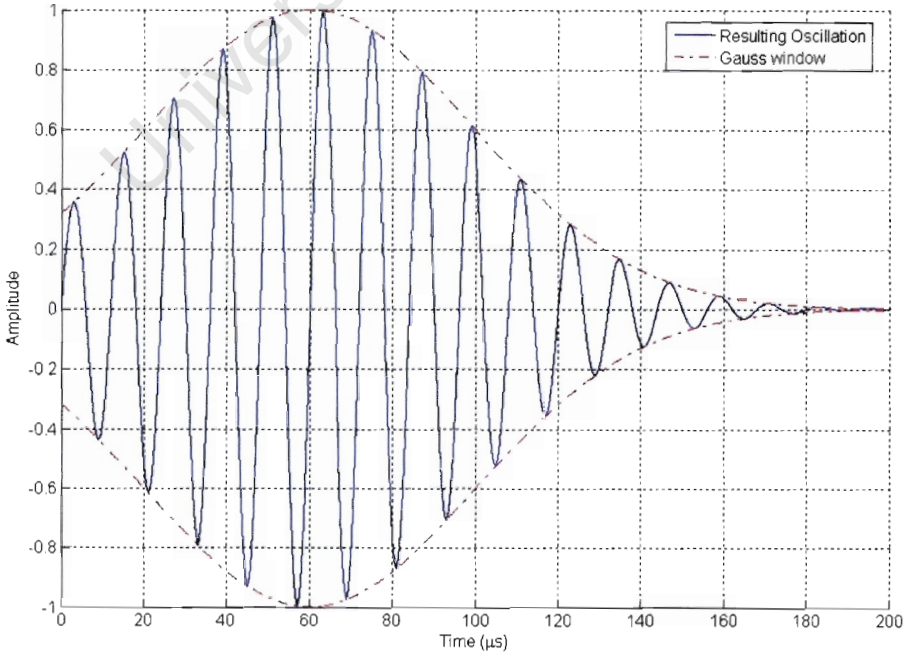


Figure 2-3 Gauss windowed sine wave

Ramirez and Rubio-Gonzalez [21] numerically simulated the propagation of square and trapezoidal waves in cylinders using the LS-DYNA code. The simulations showed that the pulses dispersed as the pulses propagated axially. The degree of dispersion was strongly linked to the frequency content of the pulse. Pulses with a large bandwidth (i.e. containing low and high frequencies) dispersed more than low bandwidth pulses.

Very little was published on numerical modelling of HPBs during the time between the work of Bertholf and Karnes in the 1970s and the work of Tyas in the late 1990s. However, most of the work on frequency domain correction of digitised signals (Gorham [8], Follansbee and Franz [9]) was published during this period.

3 The Analytical Solution For Vibration of Infinite Cylinders

3.1 The Pochammer-Chree solution

Pochammer (1876) [2] and Chree (1889) [6] arrived independently at the same equations for the vibration of an infinitely long cylindrical bar. Their analyses permitted variation of displacement in the axial and radial direction, i.e. a 2 dimensional solution. The Pochammer-Chree solution relates the velocity of propagation (phase velocity C_p) of a sinusoidal wave to the wavelength Λ. Love [22] summarised Pochammer and Chree's work and presented the equation used to determine the phase velocity and hence frequency of a disturbance of a given single wavelength. Bancroft [7] presented Love's equation in the following form:

$$(x-1)^2\phi(ha)-(\beta x-1)[x-\phi(ka)]=0$$

Eq 3-1

The individual terms in Eq 3-1are given by the following equations.

$$\beta=\frac{1-2\upsilon}{1-\upsilon} \qquad x=(1-\upsilon)\left[\frac{C_p}{C_0}\right]^2 \qquad \gamma=\frac{2\pi}{\Lambda}$$

$$h=\gamma\sqrt{\beta x-1} \qquad k=\gamma\sqrt{2x-1} \qquad \phi(ha)=\frac{J_0(ha)}{J_1(ha)}$$

$$\phi(ka)=\frac{J_0(ka)}{J_1(ka)} \qquad J_0=\sum_{m=0}^{\infty}\frac{(-1)^m x^{2m}}{2^{2m}(m!)^2} \qquad J_1=\sum_{m=0}^{\infty}\frac{(-1)^m x^{2m+1}}{2^{2m+1}((m+1)!)^2}$$

J_0 is a Bessel function of the first kind, of order zero and J_1 is a Bessel function of the first kind, of order one.

Eq 3-1 is often referred to in the literature as the "Dispersion Equation" or 'Frequency Equation". For a given bar and wavelength Λ it is possible to find values of x which satisfy Eq 3-1 and hence the phase velocity and frequency. These correspond to the different Pochammer-Chree modes of vibration.

The assumed form of the displacement solution is

$$u_z = U(r)e^{i(\chi z + pt)}$$
$$u_r = W(r)e^{i(\chi z + pt)} \qquad \textbf{Eq 3-2}$$

U(r) and *W(r)* are time invariant functions which describe the variation in either radial or axial displacement as a function of radius. Love showed that possible forms of *U(r)* and *W(r)* are:

$$U(r) = -AhJ_1(hr) + C\chi J_1(kr)$$
$$W(r) = Ai\chi J_o(hr) + CikJ_o(kr) \qquad \textbf{Eq 3-3}$$

A and *C* are arbitrary constants which define the magnitude of the displacement. As *h* and *k* are functions of *x*, *U(r)* and *W(r)* are only defined for those values of *x* which satisfy Eq 3-1. The roots of Eq 3-1 are solved numerically. However, this is complicated by the need to solve these over a range of wavelengths to obtain the solution over the desired frequency range.

The change in form of a plot of the dispersion equation as wavelength changes is shown in Figure 3-1. The wavelength changes from 12mm to 8mm to 4mm in the descending plots. It is evident that as wavelength decreases, more roots of Eq 3-1 (and hence more modes) exist in the domain of interest. The number of roots occurring before the central stationary point also increases as wavelength decreases. In addition, the numerical solver must be able to distinguish between the true roots and the asymptotes present in the domain.

In order to solve for an unknown number of roots over a variable frequency domain, a dedicated solver was written. The Matlab code for this solver is documented in detail in Appendix A .

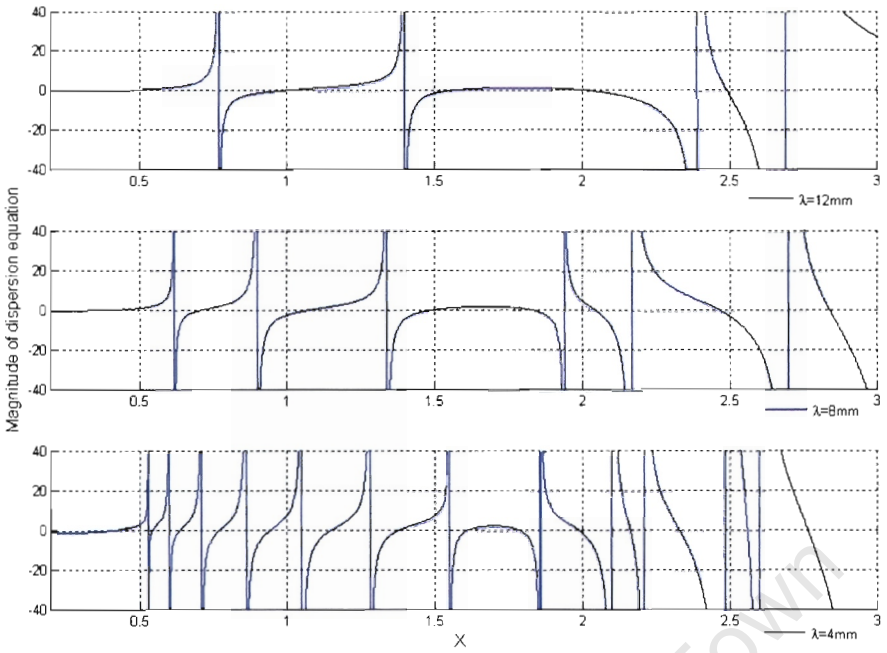


Figure 3-1 Change of dispersion equation form with respect to wavelength

The author's initial attempts at the solver code were computationally inefficient and showed instability at the higher modes. This instability manifested in the solver skipping modes arbitrarily. The solver algorithm was refined to remove these instabilities and make the code more efficient. This algorithm is shown in Figure 3-2.

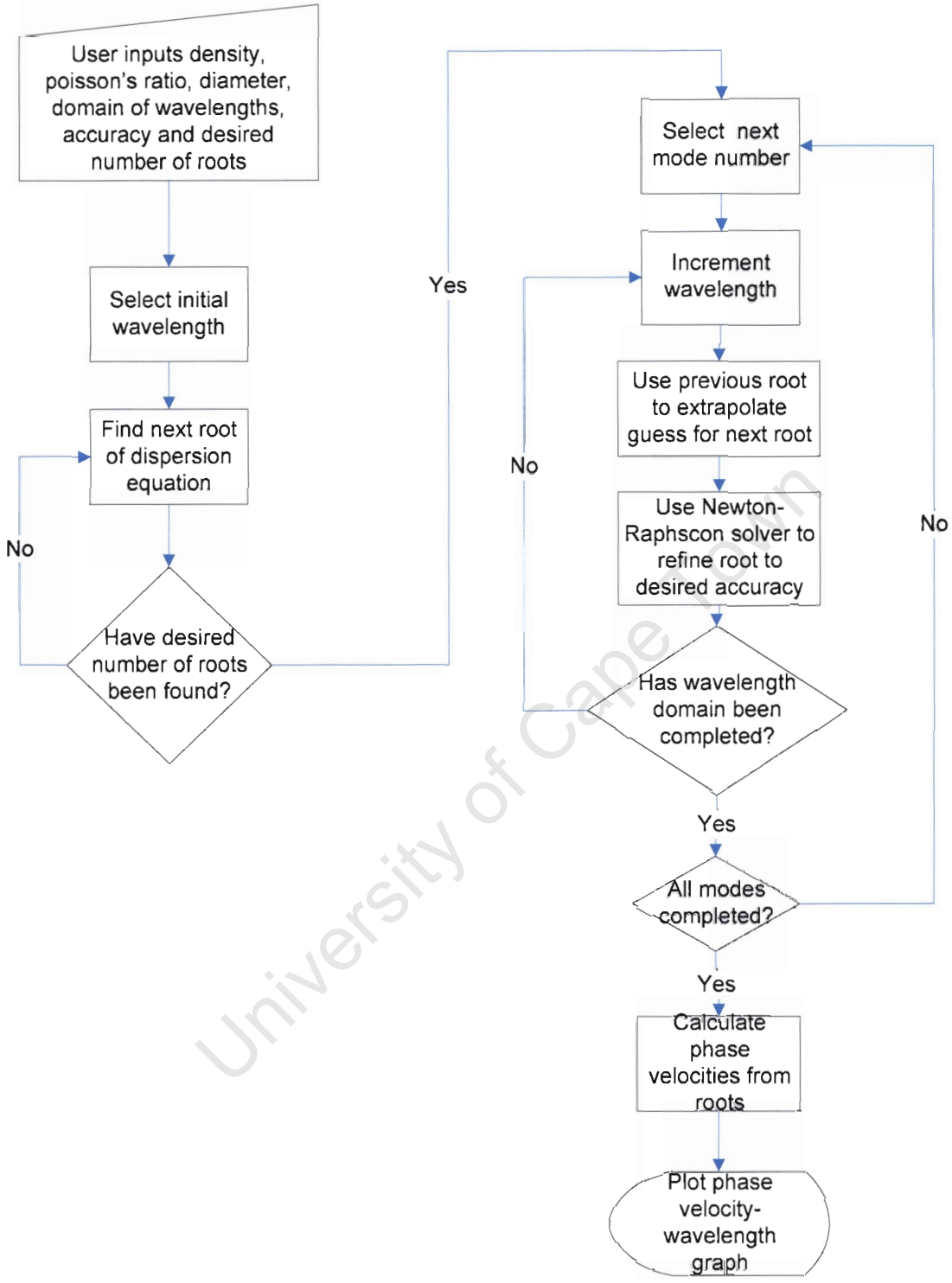


Figure 3-2 Flowchart of solver algorithm

The primary output of the solver is the numerical relationship between phase velocity and wavelength of excitation. Figure 3-3 shows the relationship between scaled phase velocity (C_p/C_0) and scaled wavelength (D/Λ) for the first six Pochammer-Chree modes for $\nu = 0.26$.

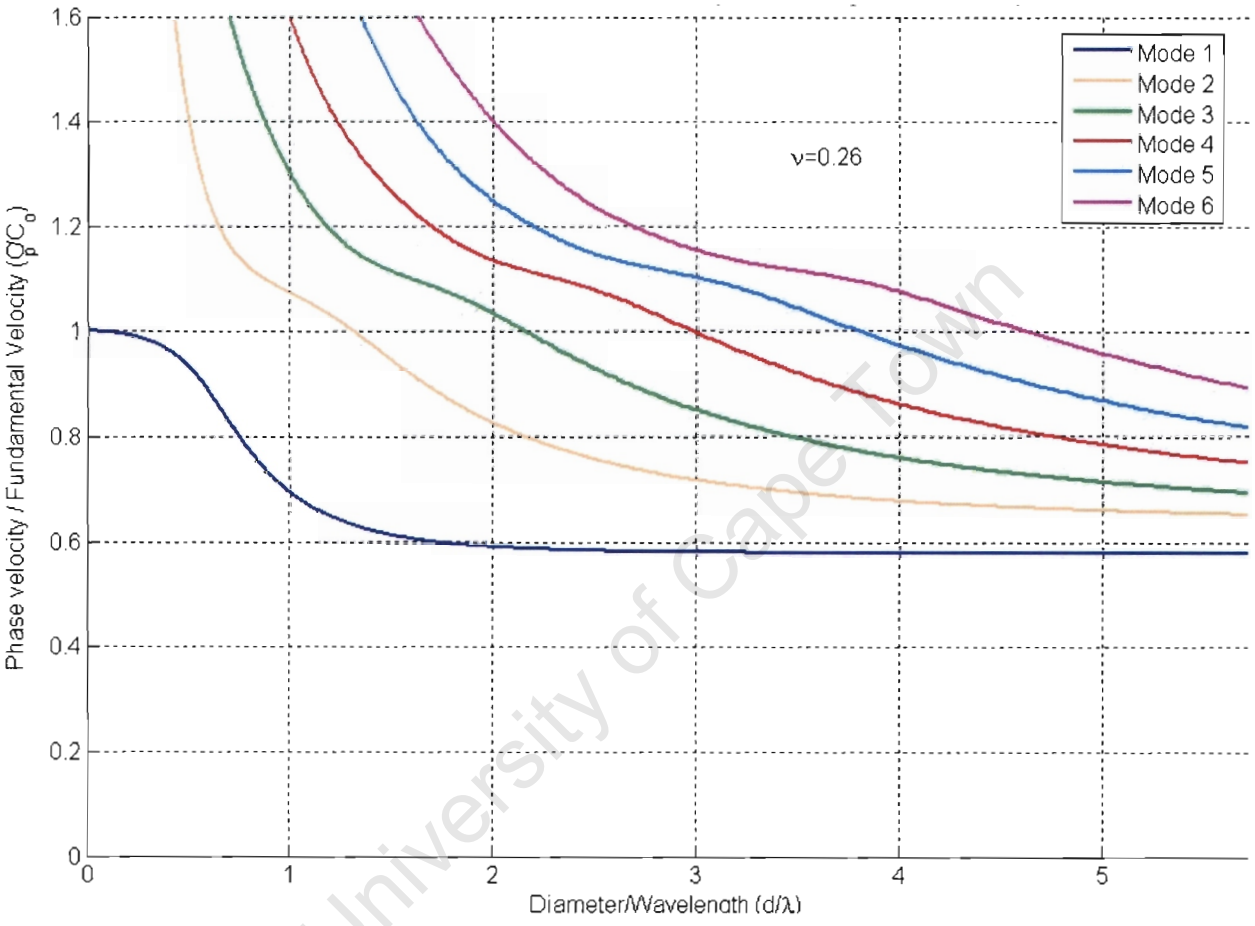


Figure 3-3 Scaled phase velocity vs. scaled wavelength for ν = 0.26

3.2 Dispersion

The Pochammer-Chree solution showed that the propagation velocity of a given sinusoidal wave is a function of the wavelength. Hence a pulse consisting of more than one wavelength will change shape as it propagates because the different wavelengths are moving relative to one another (phase shifting). This phenomenon is known as dispersion (Kolsky [4], Achenbach [14]).

It is evident from Figure 3-3 that as the wavelength decreases, the phase velocity also decreases. Hence the shorter wavelength (or higher frequency) components of a pulse will begin to lag the longer wavelength components as the pulse propagates axially. The dispersion of a trapezoidal pulse is shown in Figure 3-4.

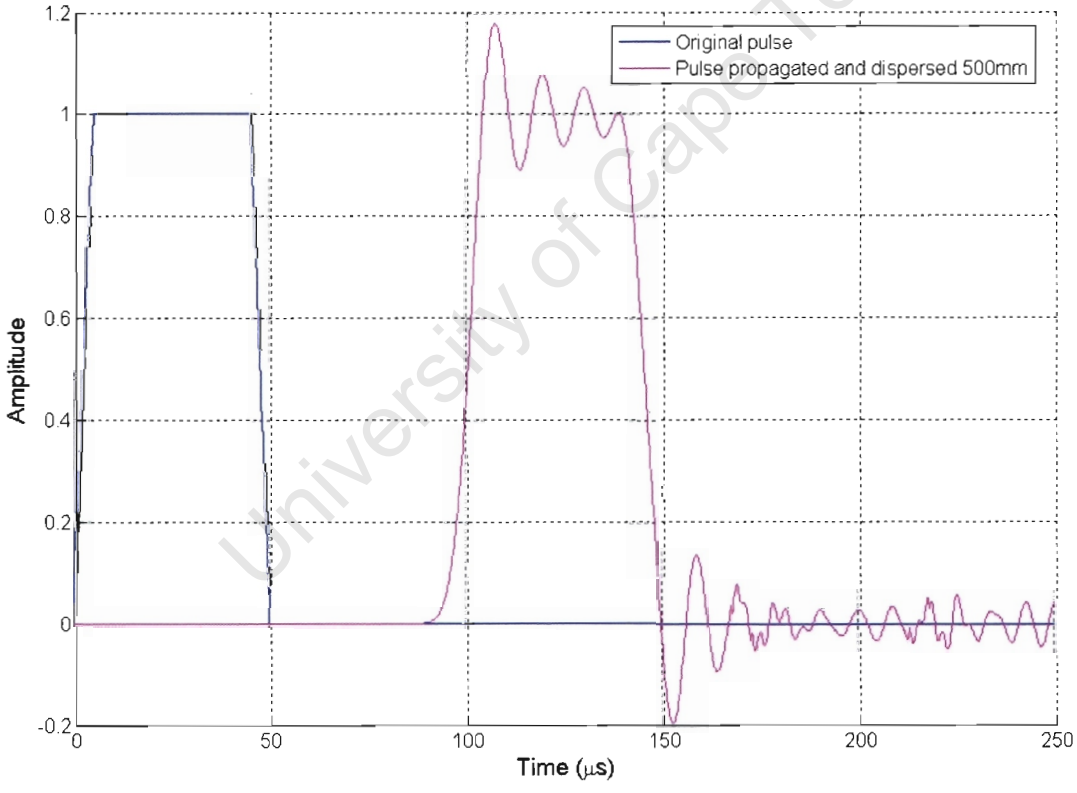


Figure 3-4 Dispersion of a trapezoidal pulse after 500mm of axial propagation

The dispersed pulse has oscillations superimposed on the plateau of the trapezoid, and on the tail where previously the amplitude was zero. Due to the superimposed oscillations, the peak value of the dispersed pulse is greater than that of the un-

dispersed pulse. The original trapezoidal pulse had sharp corners (discontinuous 1st derivative with respect to time) whereas the corners of the dispersed pulse have been smoothed out (1st derivate with respect to time is continuous). Closer inspection of the graph reveals that the rise and decay of the dispersed pulse are not as steep as the un-dispersed pulse.

Dispersion of stress pulses has significant implications for SHPB tests. The strain-time history in the HPBs is measured relatively far away (at least 10 bar diameters, but typically of the order of hundreds of mm) from the specimen-bar interface or the loaded face. Hence the pulse will have dispersed as it propagated axially from the point of measurement to the point where the strain-time history is needed. Failure to account for this dispersion will lead to uncertainties and inaccuracies in the loading of the specimen and hence the stress-strain relationship determined from the test.

The Pochammer-Chree solution presents the phase velocity as a function of wavelength. However, it is necessary to obtain the phase velocity as a function of frequency. Experimental signals from HPBs are measured as functions of time. Applying a Fourier transform to these signals transforms them to the frequency domain. The phase velocity is easily transformed from the scaled wavelength domain by the following:

$$\frac{d}{\lambda} \Rightarrow \frac{d}{\lambda}\frac{C_p}{C_o} = \frac{d}{C_o}\frac{C_p}{\lambda} = \frac{d}{C_o} f \qquad \textbf{Eq 3-4}$$

As d/C_o is constant for all wavelengths / frequencies, it may be removed from the equation. Hereafter the phase velocity is presented in the frequency domain rather than the wavelength domain (Figure 3-5).

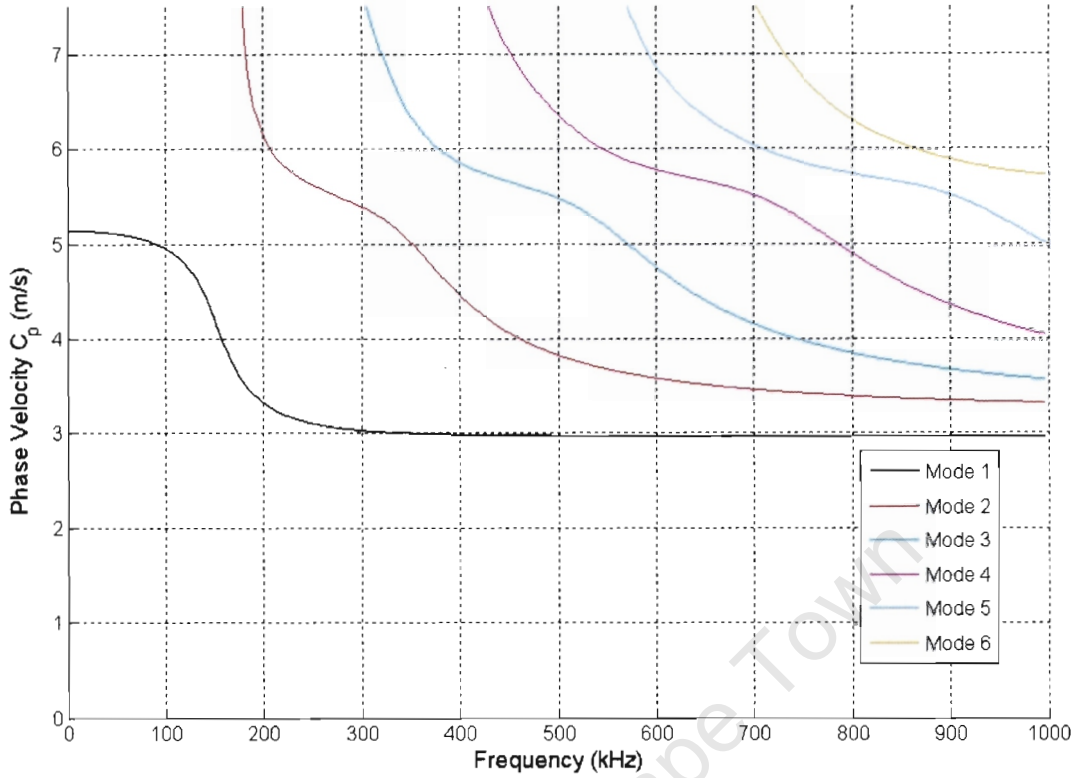


Figure 3-5 Pochammer-Chree phase velocity as a function of frequency, d = 20mm

3.2.1 Effect of different parameters on phase velocity-frequency relationship

The phase velocity-frequency relationship determined from the Pochammer-Chree dispersion equation depends on the material elastic modulus E, density ρ, Poisson's ratio v and the bar diameter d. E and ρ affect only the fundamental velocity C_o, and hence change only the magnitude of the phase velocity-frequency curve but not its shape. The shape of the phase velocity-frequency curve is affected by v and d.

3.2.1.1 Effect of diameter

The change in the phase velocity-frequency relationship for different diameter bars is presented in Figure 3-6. The following parameters were kept constant: E = 207 GPa, ρ = 7830 kg.m^{-3}, ν = 0.26. Only the first Pochammer-Chree mode is presented for clarity.

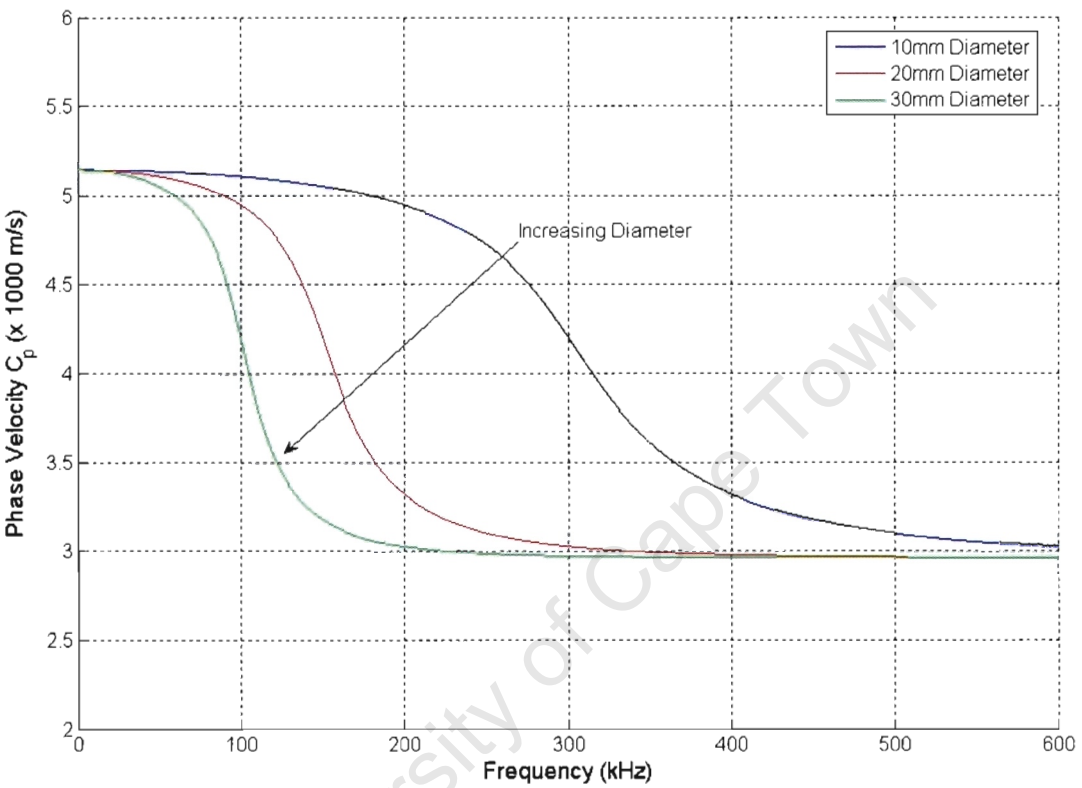


Figure 3-6 Effect of bar diameter on first Pochammer-Chree mode, ν=0.26

As the bar diameter is increased, the decrease in C_p begins at lower frequencies and is steeper. The value of C_p, which the first mode tends to with increasing frequency, is not significantly affected by diameter. Hence as the bar diameter decreases, the phase velocity does not vary much from the fundamental wave velocity for a much larger frequency range (up to 100 kHz). This shows that as the bar becomes more slender, the Pochammer-Chree solution tends towards the one dimensional wave equations, which predict that phase velocity does not vary with frequency.

3.2.1.2 Effect of Poisson's ratio

The effect of ν on the phase velocity-frequency relationship is shown in Figure 3-7, while the following were kept constant: E = 207 GPa, ρ = 7830 kg.m^{-3}, d = 20mm.

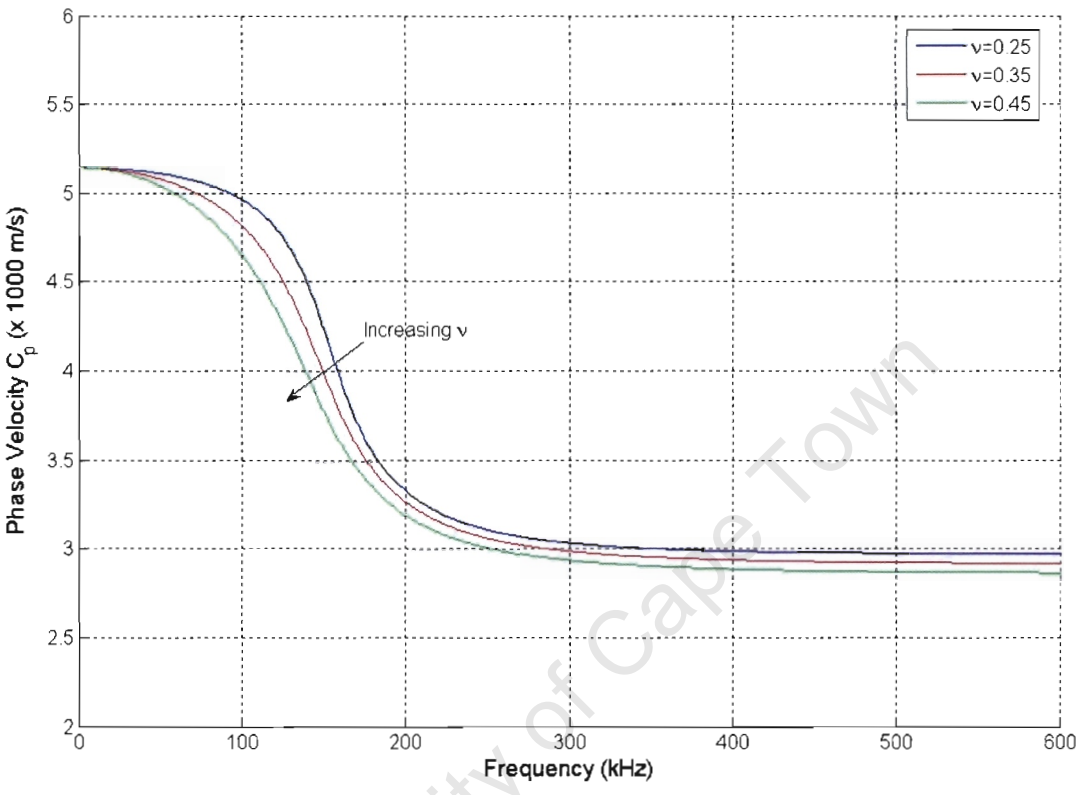


Figure 3-7 Effect of Poisson's ratio on first Pochammer-Chree mode, d = 20mm

As ν increases (the material becomes less compressible), the decrease in C_p begins at lower frequencies but is less steep. Higher values of ν also decrease the value of C_p that the first Pochammer-Chree mode approaches as frequency increases.

3.3 Variation of displacement over the radius

The Pochammer-Chree solution permits the radial and axial displacement to vary as a function of radial and axial positions. These functions are presented in Eq 3-3. However, the shape of these functions can only be determined once the roots of the dispersion equation (Eq 3-1) have been solved. The displacement profile of a given Pochammer-Chree mode changes as a function of frequency. As the higher modes only exist above certain frequency values, it is not possible to plot their displacement profile below these. Examples of the axial displacement as a function of radius, for several different frequencies, are shown in Figure 3-8.

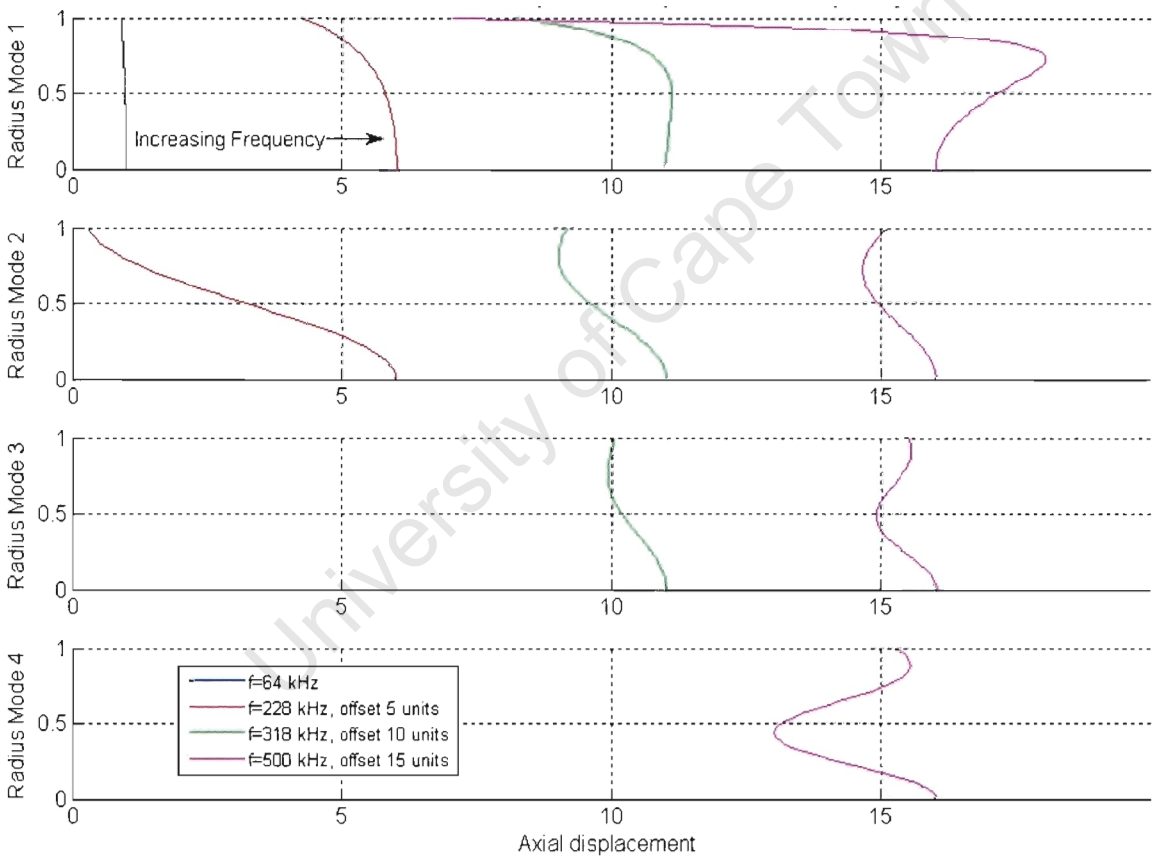


Figure 3-8 Variation of axial displacement as a function of radius, for different modes at different frequencies

The displacement profiles in Figure 3-8 are normalised by dividing by the displacement at the centre of the bar. The higher frequency modes are also offset, as indicated in the legend, for clarity.

3.4 Group Velocity

The Pochammer-Chree dispersion equation gives the phase velocity of a sinusoidal pulse of a single wavelength or frequency. However, most pulses or disturbances are not sinusoidal even though they may be described by a Fourier series of sinusoids. Even when the pulse is described by a Fourier series, it will have many different wavelengths associated with the different terms. The energy contained in the pulse will propagate at the group velocity C_g (Davies [2], Kolsky [4], Achenbach [14]). The group velocity is given by Eq 2-4. The group velocities derived from the Pochammer-Chree phase velocities are shown in Figure 3-9.

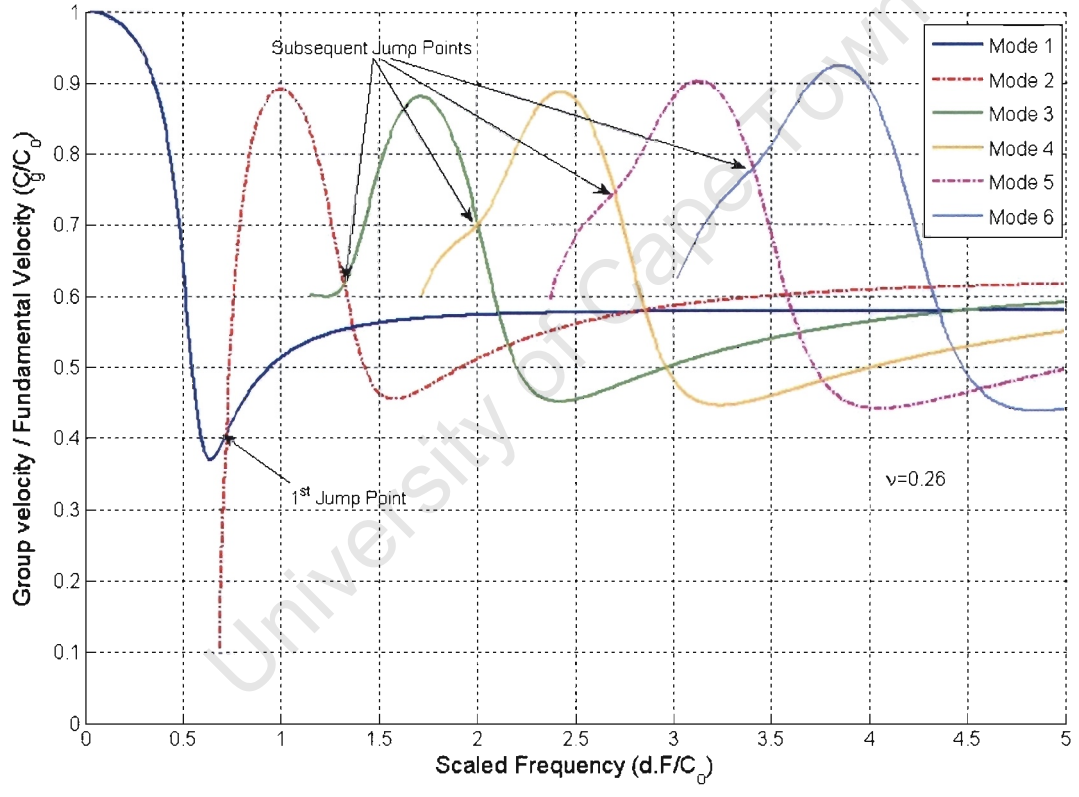


Figure 3-9 Scaled group velocity curves derived from Pochammer-Chree phase velocities

The experimental analysis of Lee and Crawford [28] suggests that the components of a pulse will propagate at the phase velocity of the Pochammer-Chree mode with the

highest group velocity. It is assumed that if more than one mode is present[2] at that frequency, all the modes will propagate at the phase velocity of the dominant mode. As the phase velocity curves of the different modes do not intersect at any point, there is a steep jump from mode to mode as opposed to a smooth transition. These jump points are determined by the intersection values of the group velocity curves, which are indicated in Figure 3-9.

[2] The FFT methods used in the dispersion correction method (§2.6) can only associate one amplitude with a given frequency. Hence it is currently not possible to determine if more than one vibrational mode is present at a given frequency.

3.5 Dispersion Correction Map

Having calculated the jump points, it is possible to construct a dispersion correction map which contains the phase velocity as a function of frequency. A typical dispersion correction map is shown in Figure 3-10.

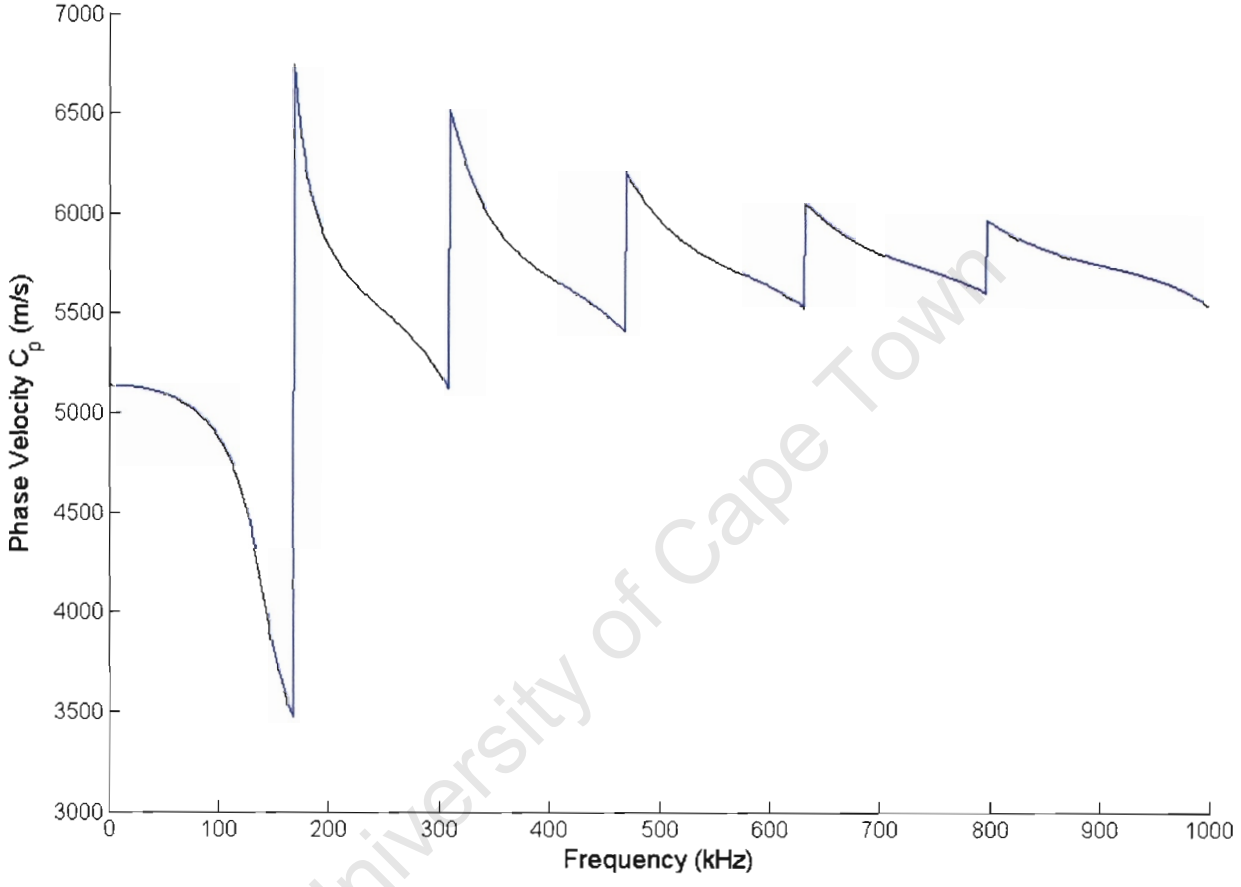


Figure 3-10 Phase velocity frequency curve for dispersion correction of a 20mm diameter steel bar, E = 207 GPa, v = 0.26

The method by which a pulse is corrected for dispersion is given in §2.6. It should be noted that while this dispersion correction method accounts for the presence of higher Pochammer-Chree modes, it only corrects for the mode with the highest group velocity at any given frequency.

4 Numerical Analysis

4.1 Introduction

The Pochammer-Chree solution gives the relationship between phase velocity and frequency for an *infinitely long* bar. Currently, no analytical solution for this relationship for a bar of finite length has been published. Many publications have shown that using the Pochammer-Chree solution to correct for dispersion gives good results for SHPB compression tests (Gorham [8], Follansbee and Frantz [9], Marais [10]). However, SHPB experiments include compression, tensile and shear tests. These tests impose very different boundary conditions on the ends of the bars to those associated with compression SHPB tests. The effect of these boundary conditions on dispersion has not been resolved. Hence it is necessary to model these SHPB experiments numerically, typically with either an explicit FEM code or finite volume code with explicit time integration. This chapter discusses the theory related to numerical simulation of dynamic problems. It documents the comparison of the Pochammer-Chree phase velocity – frequency relationship (C_p-f) with that predicted by the Abaqus v6.5.1 finite element code for an infinitely long bar, obtained using the modal summation method. Abaqus was used to model a bar of finite length. The C_p-f relationship of the finite length bar was derived from its Eigen modes. The response of bars to dynamic loading was modelled using Abaqus/Explicit and Abaqus Modal Dynamics, and compared to waves dispersed according to the Pochammer-Chree theory.

4.2 Numerical Simulation of Dynamic Problems

A dynamic problem involving a linear elastic element may be solved using the well known form of Newton's Third Law.

$$m\ddot{x} + kx = f(t) \qquad \text{Eq 4-1}$$

If the problem domain has *N* degrees of freedom and may be discretised this equation takes the matrix-vector form:

$$\mathbf{M}\underline{\ddot{x}} + \mathbf{K}\underline{x} = \underline{F}(t) \qquad \text{Eq 4-2}$$

Some of the methods used for solving this equation are discussed in §4.2.1 & §4.2.2. Thomson [23] and Rao [24] give a more detailed account of the derivations.

4.2.1 Modal Summation Method

The solution to the homogeneous part of Eq 4-2 may take the form

$$\underline{x}_i = \underline{X}_i(Cos(\omega_i t + \vartheta)) \qquad \text{Eq 4-3}$$

$\underline{X}$ gives the time invariant magnitude of the displacements at each degree of freedom (node) and is referred to as the "mode shape" or the "normal mode". If this is substituted into Eq 4-2, the result is

$$[-\omega_i^2\mathbf{M} + \mathbf{K}]\underline{X}_i(Cos(\omega_i t + \vartheta)) = \underline{0} \qquad \text{Eq 4-4}$$

This reduces to the Eigen value problem

$$\left|\mathbf{K} - \omega_i^2\mathbf{M}\right| = 0 \qquad \text{Eq 4-5}$$

The Eigen values of this equation correspond to the square of the natural frequencies ω_i^2, and the Eigen vectors to the normal modes $\underline{X}_i$. It can be shown that the normal modes form an orthogonal set, i.e. for all $i \neq j$

$$\underline{X}_i^T\mathbf{M}\underline{X}_j = \underline{0}$$

$$\underline{X}_i^T\mathbf{K}\underline{X}_j = \underline{0} \qquad \text{Eq 4-6}$$

Further more,

$$\underline{X}_i^T\mathbf{M}\underline{X}_i = M_{ii}$$

$$\underline{X}_i^T\mathbf{K}\underline{X}_i = K_{ii} \qquad \text{Eq 4-7}$$

M_{ii} and K_{ii} are respectively the generalized mass and stiffness, associated with a given mode. The ortho-normal mode is defined as

$$\underline{\phi}_i = \underline{X}_i \Big/ \sqrt{M_{ii}} \qquad \textbf{Eq 4-8}$$

Hence Eq 4-7 reduces to

$$\begin{aligned} \phi_i^T \mathbf{M} \phi_i &= 1 \\ \phi_i^T \mathbf{K} \phi_i &= \omega_i^2 \end{aligned} \qquad \textbf{Eq 4-9}$$

The ortho-normal mode vectors may then be assembled into the modal matrix $\mathbf{\Phi}$

$$\mathbf{\Phi} = \begin{bmatrix} \underline{\phi}_1 & \underline{\phi}_2 & \dots & \underline{\phi}_N \end{bmatrix} \qquad \textbf{Eq 4-10}$$

If one transforms from the original coordinate system $\underline{x}$ to a new system $\underline{y}$ such that

$$\underline{x} = \mathbf{\Phi}\underline{y} \qquad \textbf{Eq 4-11}$$

Then Eq 4-2 becomes

$$\mathbf{M\Phi}\ddot{\underline{y}} + \mathbf{K\Phi}\underline{y} = \underline{F} \qquad \textbf{Eq 4-12}$$

Pre-multiplying by $\mathbf{\Phi}^{\mathrm{T}}$ yields

$$\mathbf{\Phi^T M\Phi}\ddot{\underline{y}} + \mathbf{\Phi^T K\Phi}\underline{y} = \mathbf{\Phi}^{\mathrm{T}}\underline{F} \qquad \textbf{Eq 4-13}$$

Because the normal modes are orthogonal [Eq 4-6,Eq 4-7,Eq 4-9] the matrices $\mathbf{\Phi^T M\Phi}$ and $\mathbf{\Phi^T K\Phi}$ are purely diagonal and the equation further reduces to

$$\ddot{y}_i + \omega_i^2 y_i = (\mathbf{\Phi}^{\mathbf{T}}\underline{F})_i \qquad \textbf{Eq 4-14}$$

The original coupled multi degree of freedom system has been decoupled to N single degree of freedom equations.

The modal summation method does not cope well with systems where damping plays a significant role. The addition of a damping term to Eq 4-2 prevents the formation of an Eigen value problem, unless certain conditions are fulfilled. The modal summation method may still be used if the damping matrix $\boldsymbol{C}$ may be written as a linear combination of the stiffness and mass matrices, i.e.

$$\mathbf{C} = \alpha\mathbf{M} + \beta\mathbf{K} \qquad \textbf{Eq 4-15}$$

This formation of the damping matrix is also known as Rayleigh damping [23]. It is seldom possible to approximate the damping matrix with sufficient accuracy in this fashion.

In the context of a conventional HPB problem, damping does not play an appreciable role. If the HPB is a metal, it is treated as a linear elastic solid. The Pochammer-Chree solution also ignores damping. Hence the absence of damping from the modal summation method does not prevent if from being applied to a HPB problem.

4.2.2 Explicit Finite Element Analysis

Explicit, or direct integration, finite element methods may be used to solve Eq 4-2. These assume that the displacement solution may be approximated by (Balden [26])

$$\underline{U} = \sum_{j=1}^{N} \varphi_j(\underline{x}) a_j(t) \qquad \textbf{Eq 4-16}$$

The assumed form of the approximated solution has been split into time invariant, spatially dependent "shape functions" φ and the amplitude is controlled by the time dependent coefficients a, which are the values of the nodal displacements. Incorporating damping into Eq 4-1 gives

$$\mathbf{M}\ddot{\underline{x}} + \mathbf{C}\dot{\underline{x}} + \mathbf{K}\underline{x} = \underline{F}(t) \qquad \textbf{Eq 4-17}$$

If the assumed form of the solution [Eq 4-16] is substituted into the governing equation [Eq 4-17] , and the Galerkin Method of Weighted Residuals applied, the result is:

$$\sum_{i=1}^{N} M_{ij} \frac{d^2 a_j(t)}{dt^2} + C_{ij} \frac{da_j(t)}{dt} + K_{ij} a_j(t) = F_i(t) \qquad \textbf{Eq 4-18}$$

Various direction integration schemes may be used to solve for the unknown nodal displacements a_j, given the initial conditions imposed on the system. Some of the more popular schemes are the Central Difference method, the Houboult method, the Wilson-θ method and the Newmark method (Balden [26], Cook [27], Bathe [31]). Abaqus/Explicit v6.5.1 makes use of the Central Difference method, which is summarised in §4.2.2.1.

The stability and accuracy of any direct integration scheme is dependent on the size of the time step Δt. The stability criterion for Δt depends on the scheme being used.

4.2.2.1 Central Difference Method

The Central Difference method solves the displacement, velocity and acceleration at any time t_n,[3] given known values for these quantities at the preceding time steps t_{n-1} and t_{n-2}. It assumes that the displacement, velocity and acceleration vary linearly across the time step. It also assumes that the time step Δt between t_n and the preceding time t_{n-1} and t_{n-1} and its preceding time t_{n-2} is equal. The velocity and acceleration at the intermediate step t_{n-1} may be approximated by [26]:

$$\dot{a}_{n-1} \approx \frac{a_n - a_{n-2}}{2\Delta t} \qquad \textbf{Eq 4-19}$$

$$\ddot{a}_{n-1} \approx \frac{a_n - 2a_{n-1} + a_{n-2}}{\Delta t^2}$$

[3] A subscript n refers to the value of the quantity at an arbitrary time step t_n, whereas subscripts i,j refer to the degree of freedom.

Writing Eq 4-18 for the intermediate time step:

$$\mathbf{M}\ddot{\underline{a}}_{n-1} + \mathbf{C}\dot{\underline{a}}_{n-1} + \mathbf{K}\underline{a}_{n-1} = \underline{F}_{n-1} \qquad \textbf{Eq 4-20}$$

By substituting Eq 4-19 into Eq 4-20, and grouping all terms containing a_n on the left hand side and a_{n-1} and a_{n-2} terms on the right gives:

$$\left(\frac{1}{\Delta t^2}\mathbf{M} + \frac{1}{\Delta t}\mathbf{C}\right)\underline{a}_n = \underline{F}_{n-1} - \left(\mathbf{K} - \frac{2}{\Delta t^2}\mathbf{M}\right)\underline{a}_{n-1} - \left(\frac{1}{\Delta t^2}\mathbf{M} - \frac{1}{2\Delta t}\mathbf{C}\right)\underline{a}_{n-2}$$

Eq 4-21

Hence the displacement at time t_n is defined explicitly in terms of t_{n-1} and t_{n-2}. This permits subsequent time steps to be solved without expensive computational processes such as matrix inversion or decomposition.

4.2.3 Comparison of Modal Summation and Explicit FEA

The modal summation method is limited to problems where the material response remains linear. The mode frequencies and shapes are calculated according to the initial stiffness matrix, which is based on linear material response. If plasticity occurs anywhere in the domain, the stiffness matrix and hence the associated vibrational modes change. The original modes, based on the linear response, cannot be applied to the non-linear behaviour associated with plastic deformation. Hence the modal summation method cannot be applied to the simulation of a full SHPB experiment, as the specimen undergoes bulk plastic deformation. It can however be used to model the wave propagation down the bar. The accuracy with which the modal summation method simulates wave propagation is analysed and discussed in greater detail in § 4.3.

Explicit FEA does not require the stiffness, mass or damping matrices to be constant for the duration of the problem. While direct integration is computationally cheapest while these matrices are constant, the matrices can be recalculated as often as

necessary. Thus any non-linearity, be it geometric or material, is readily incorporated into an explicit simulation. Explicit FEA can be used to model a full SHPB experiment, including the plastic response of the specimen. However, accuracy is strongly linked to the time step chosen. Most schemes calculate a critical time step based on the time for a disturbance moving at the fundamental wave velocity C_o to travel across the smallest element dimension in the domain.

4.3 FE Simulation of Vibration of an Infinitely Long Cylindrical Bar

An axisymmetric model of a infinite cylinder was created in Abaqus 6.5.1, to assess how accurately the FE model simulated the dispersion behaviour predicted by the Pochammer-Chree solution. The form of the Pochammer-Chree displacement solution is given by Eq 3-2:

$$\underline{u} = \underline{U}(r)e^{i(\frac{2\pi z}{\Lambda} + pt)}$$

The form of the complex exponential shows that the displacement will vary sinusoidally along the axial direction. At any given time, the axial location of the maximum and minimum displacements will be determined by the disturbance's wavelength Λ. The displacement solution can be broken down to its time dependent, spatially invariant, and spatially dependent, time invariant components.

$$\begin{aligned} \underline{u} &= \underline{U}(r)e^{i(pt)}e^{i(\frac{2\pi z}{\Lambda})} \\ &= \underline{U}(r)e^{i(pt)}[Cos(\frac{2\pi z}{\Lambda}) + iSin(\frac{2\pi z}{\Lambda})] \end{aligned} \qquad \textbf{Eq 4-22}$$

The portion of the displacement solution which is dependent on z is a complex exponential (i.e. hence it varies sinusoidally). For a given excitation wavelength Λ, there will be values of z at which the complex exponential term is zero and hence the resultant displacement is zero. The axial positions with zero displacement are termed axial nodes.

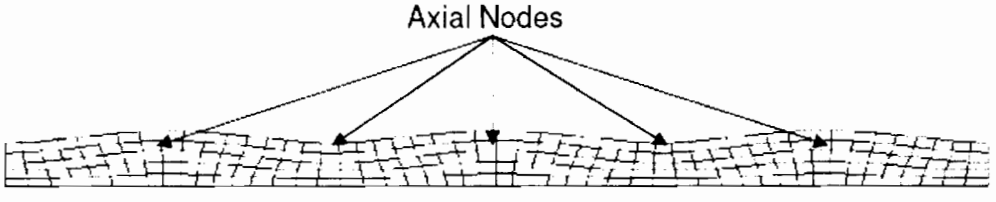


Figure 4-1 Deformed vibrational mode shape

Hence an "infinite cylinder" can be modelled as a cylinder of finite length, with the ends treated as nodes (axial displacement constrained to zero). The model exploits the existence of axial nodes as convenient locations to terminate the model numerically. The model then represents a portion of an infinite cylinder. This constraint does restrict the vibrational modes extracted to those with wavelengths shorter than the length of the modelled cylinder. The wavelengths will also be integral fractions of the model bar length.

4.3.1 Material

The bar material was modelled as an elastic solid. In keeping with the other analyses discussed, the material was steel with an Elastic Modulus E = 207 GPa, density ρ = 7830 kg.m^{-3} and Poisson's Ratio ν = 0.26.

4.3.2 Geometry

The bar was modelled as an axisymmetric cylinder, of diameter 20mm and length 200mm.

4.3.3 Boundary Conditions

As mentioned in §4.3, the axial displacement at both ends of the bar was constrained to zero. The radial displacement along the central axis of the bar was constrained to zero to enforce axisymmetry, while the axial displacement along the centre line was unconstrained. The outer radius of the bar was unconstrained.

4.3.4 Mesh

Axisymmetric continuum stress elements, with 4 nodes, linear shape functions and full integration (CAX4) were used. Higher order shape functions would be expected to yield more accurate results. However, only linear shape function elements are available in Abaqus/Explicit v6.5.1. The results of the modal analysis could only be compared with those of an explicit FEA if the element shape functions (and other parameters) are the same. Initial simulations with reduced integration elements produced "zero strain energy" or "hourglass" modes in the Eigen analysis. To avoid these spurious modes full integration was used. Thirteen simulations were run to investigate the effect of axial and radial mesh density and radial bias factor on the

accuracy of the solution. The mesh parameters used in the different simulations are summarised in Table 4-1.

Table 4-1 Mesh Density Parameters

Simulation	Axial Elements	Radial Elements	Bias Factor
MD 1	100	10	1
MD 2	100	20	1
MD 3	100	40	1
MD 4	200	10	1
MD 5	200	20	1
MD 6	200	40	1
MD 7	400	10	1
MD 8	400	20	1
MD 9	400	40	1
MD 10	400	20	1.5
MD 11	400	20	2.0
MD 12	400	20	3.0
MD 13	400	20	5.0

4.3.5 Post-Processing of Abaqus Results

The vibrational modes are processed in Abaqus by running a "Linear Perturbation – Frequency" step. The C_p-f relationship may be obtained from the following formula:

$$C_p = f_p \Lambda_p \qquad \textbf{Eq 4-23}$$

The frequency values for each mode are readily obtained as Abaqus writes these to a data file. The axial displacement of any point varies sinusoidally as a function of undeformed axial position. The wavelength of this sine wave corresponds to the wavelength of the mode. Due to the axial constraint imposed on the end of the bars, the axial displacement profile had to assume either complete or half sine waves, but could not take on any other fraction of the sine wave.

This was checked by randomly selecting a mode from the results and plotting the axial displacement profile against a true sine wave of the same wavelength. An example of the axial displacement profile corresponding to the mode with a frequency of 180 kHz is shown in Figure 4-2.

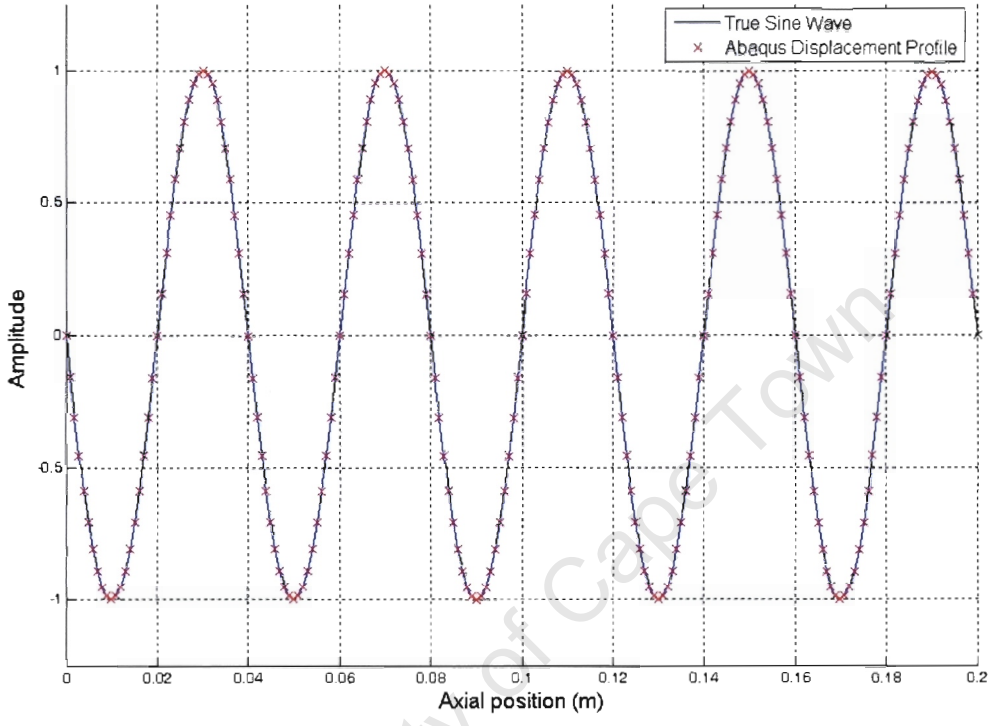


Figure 4-2 Abaqus axial displacement profile compared to a true sine wave of wavelength 40 mm (180 kHz)

The lack of variation between the Abaqus axial displacement profile and the true sine wave confirms the form of the Pochammer-Chree solution.

The extraction of the Eigen mode data from Abaqus required the use of a Python Script macro, due to the large number (>500) of modes. The data files were then processed in Matlab to determine the mode wavelength and hence its phase velocity. As the axial displacement profiles were pure sine waves, the wavelength was most accurately determined by counting zero crossings. The flow of data between Abaqus, Python and Matlab is shown in Figure 4-3. Examples of the Abaqus input data deck, Python script and Matlab code for wavelength determination may be found in Appendix B .

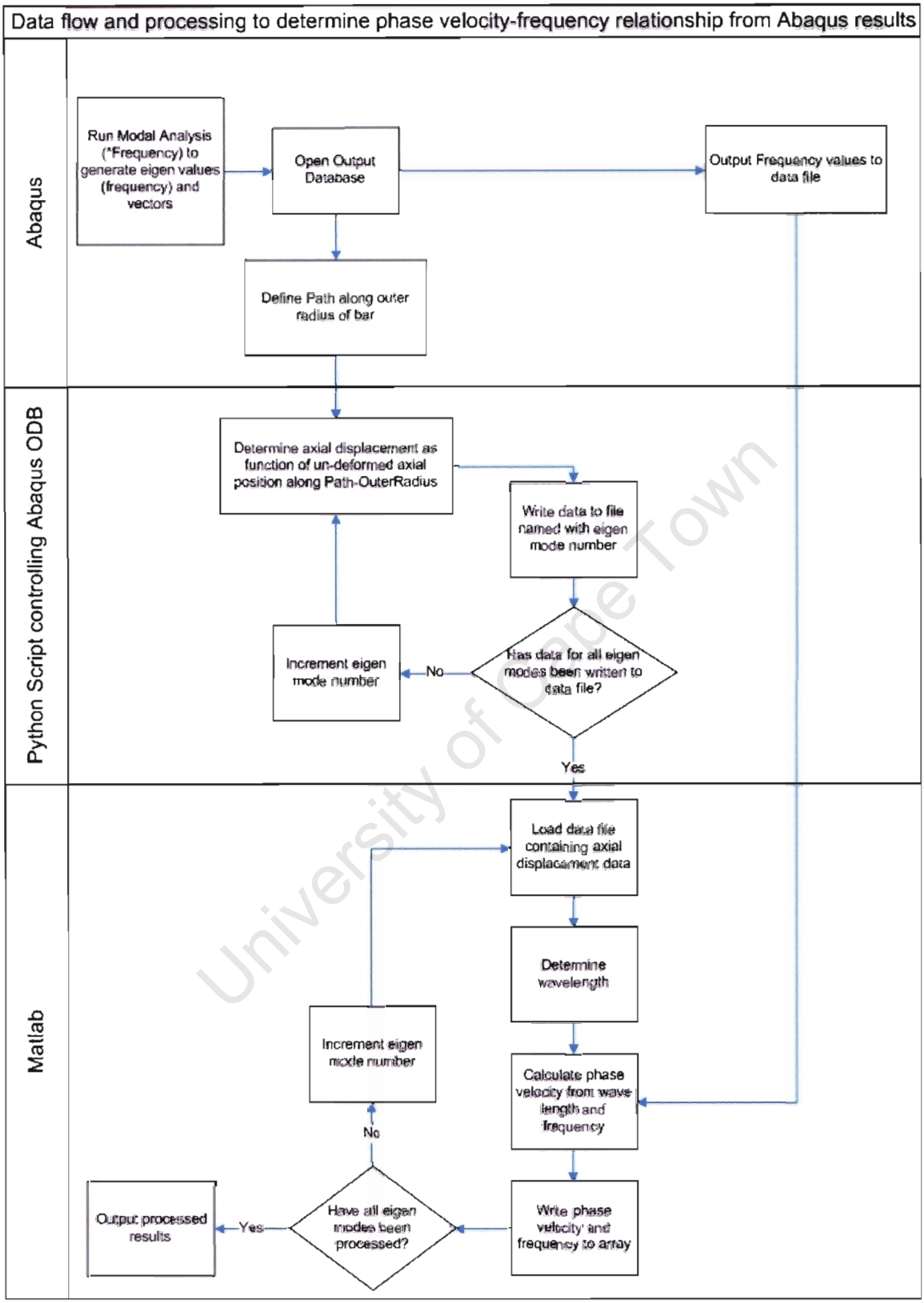


Figure 4-3 Data flow while determining C_p-f relationship from Abaqus results

4.4 Results & Discussion

4.4.1 Axial Mesh Dependence

The axial mesh dependence was determined by fixing the radial mesh at 40 elements and varying the axial mesh with 100, 200 and 400 elements. The results of these simulations are shown in Figure 4-4.

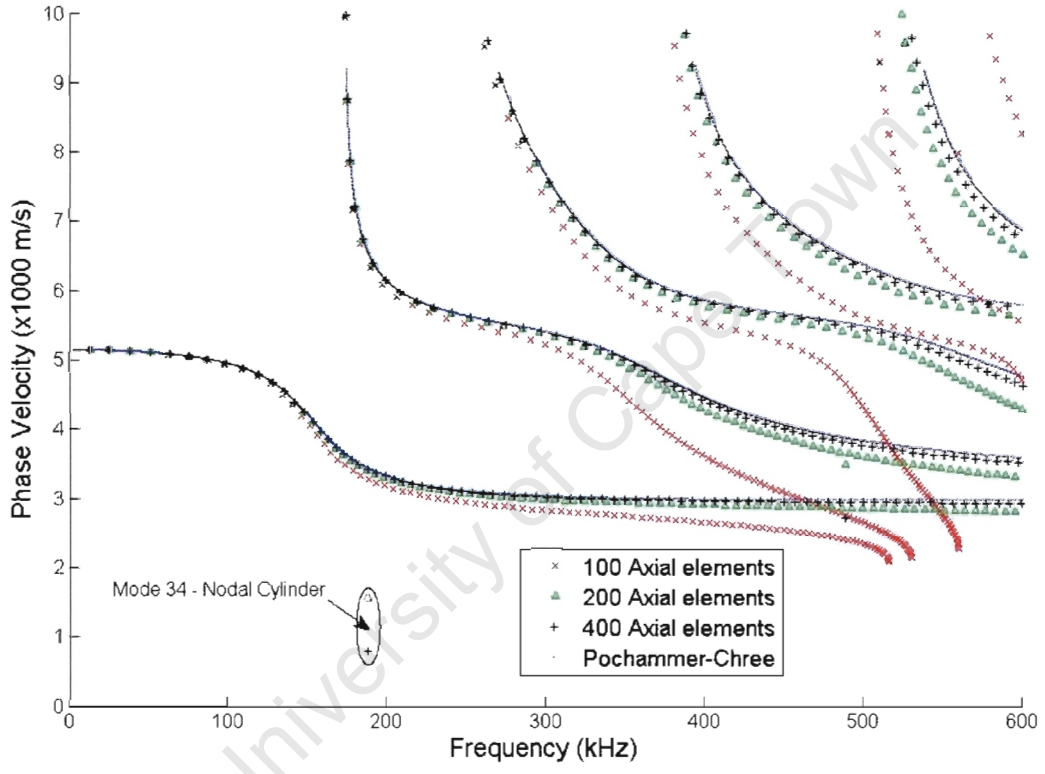


Figure 4-4 Effect of axial mesh density on numerical C_p-f prediction

The results from the 100 axial element model diverge rapidly from the Pochammer-Chree solution above 200 kHz. Increasing the number of axial elements from 100 to 200 reduces the variation between the numerical and Pochammer-Chree solutions by an order of magnitude, at frequencies above 500 kHz. The increase from 200 to 400 axial elements does not make as dramatic a reduction in variation for the first mode but does affect the variation of the higher modes significantly. It is evident that the accuracy of the numerical solution is highly dependent on the axial mesh density.

4.4.2 Radial Mesh Dependence

The numerical solution's radial mesh dependence was determined by fixing the axial mesh at 400 elements and varying the radial mesh over 10, 20 and 40 elements. The effects of radial mesh density on the accuracy of the numerical solution are shown in Figure 4-5.

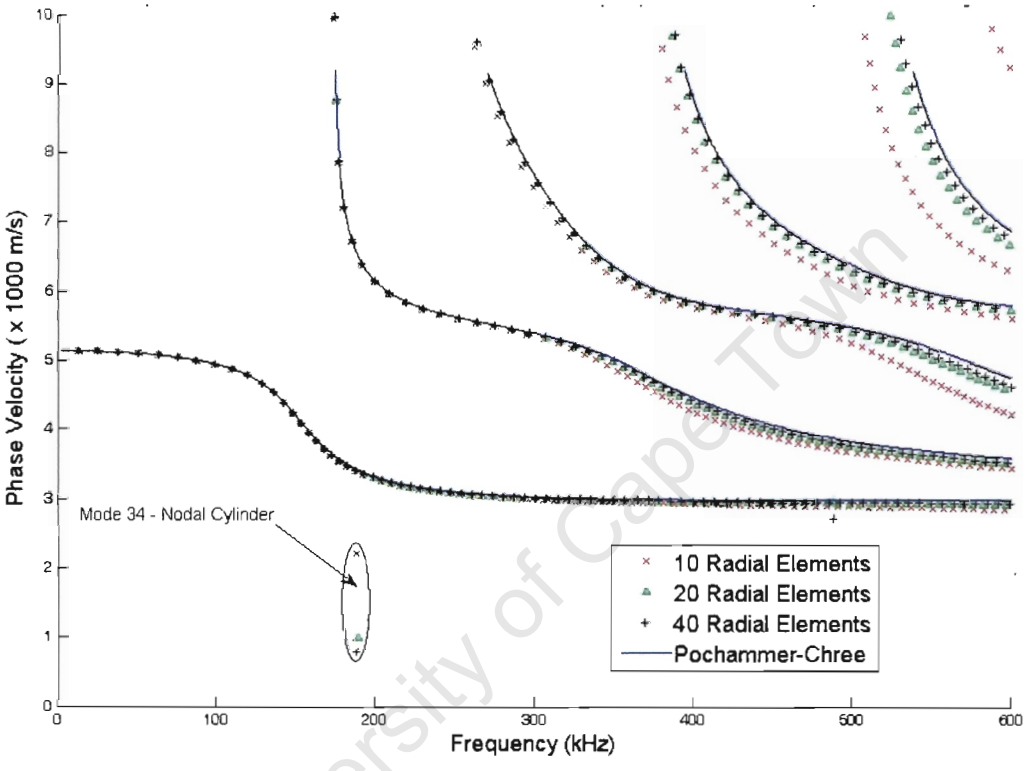


Figure 4-5 Effect of radial mesh density on numerical phase velocity prediction

Visible divergence from the first and second Pochammer-Chree modes is only evident above 300 kHz. Increasing the number of radial elements reduces the variation between the numerical solution and the Pochammer-Chree solution. However, this effect is only significant for the higher Pochammer-Chree modes. All of the radial meshes investigated showed very good agreement with the first Pochammer-Chree mode.

4.4.3 Radial Bias Factor Dependence

The radial mesh was biased with a greater mesh density near the outer radius of the bar. Bias factors of 1.5, 2, 3 and 5 were investigated. 20 radial elements and 400 axial elements were used for all of the simulations. A comparison of the Pochammer-Chree C_p-f relationship with that predicted by Abaqus for different radial bias factors is presented in Figure 4-6.

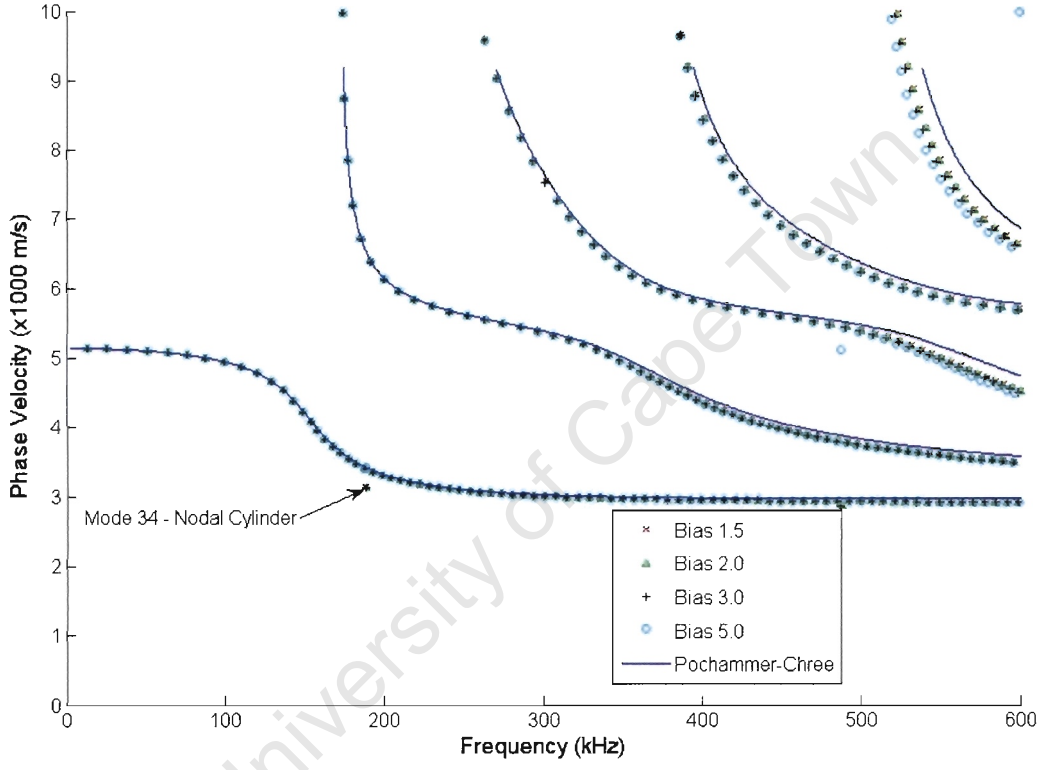


Figure 4-6 Effect of radial bias factor on numerical phase velocity prediction

There is no significant variation between the numerical results and the Pochammer-Chree prediction for frequencies less than 300 kHz. The numerical results show good agreement with the first and second Pochammer-Chree modes over the entire frequency range. For the higher Pochammer-Chree modes, the variation between numerical and analytical results increases as frequency increases. Higher radial bias towards the outer radius resulted in a decrease in variation between the numerical results and the first Pochammer-Chree mode. This was accompanied by an increase in variation between the numerical results and the higher Pochammer-Chree modes.

4.4.4 Summary of Mesh Density Study

The results presented show that the vibrational modes of an infinite cylinder can be modelled in Abaqus with acceptable accuracy. This may be used as a baseline to estimate the mesh density for simulation of a HPB experiment if one knows the frequency content of the applied load. An HPB loaded with a load bandwidth less than 200 kHz may be modelled with only 20 radial (per 10mm of radius) and 1000 axial (per 1000mm of bar length) elements with no detriment to the accuracy. To obtain a similar level of accuracy with a load bandwidth of 500 kHz, 40 radial and 2000 axial elements would be required.

4.4.5 Mesh size criterion

Numerical modelling of a HPB experiment requires some knowledge of either the input load, or a measured strain or displacement at some point on the bar. It is possible to determine an "optimum" mesh size, based on the frequency content of the known signal, that accurately captures the wave propagation while minimizing computational expense. For the purposes of this study, an accuracy of 1% was deemed sufficient. It was not possible to calculate an error or variation curve for all the data points returned by Abaqus as these jumped between Pochammer-Chree modes. It was possible to determine the frequency at which the variation exceeded 1% by visual inspection of the graphs followed by detailed interrogation of the data points. As this was time consuming only a few examples are presented. The following table contains the applicable data from selected simulations at the point where the variation between numerical and analytical phase velocities exceeds 1%.

Table 4-2 Frequency, phase velocity and mesh geometry at 1% variation between numerical and analytical predictions

Radial Elements	40	40	40	10	20
Axial Elements	100	200	400	400	400
Frequency f (kHz)	135	316	656	305	481
Phase Velocity C_p (m/s)	4501	5252	5645	5310	5498
Wavelength/Diameter (λ/D) (mm)	1.67	0.84	0.43	0.87	0.57
Wavelength / Radial Element length (λ /Re)	133.3	67	34.4	17	22
Wavelength / Axial Element length (λ /Ze)	17	17	17	35	22

The author recommends that simulation of HPB experiments use 20 elements across the radius. Elements with a square aspect ratio (radial length = axial length) should be used unless the bar is relatively short. The earlier comparisons showed that an increase in axial mesh density yields better accuracy that a similar increase in radial mesh density, while the increase in computational expense is the same.

4.5 Phase Velocity – Frequency Relationship For a Finite Length Bar

The Pochammer-Chree solution provides the C_p-f relationship for a bar of infinite length. While no analytical solution exists for a bar of finite length, attempts have been made to empirically determine the phase velocity – frequency relationship for a finite length bar (Lee and Crawford [28]). Having shown that a modal analysis can accurately predict Pochammer-Chree modes of vibration in an infinite bar, a similar method was applied to a finite bar model to determine its C_p-f relationship.

4.5.1 Material, Geometry and Mesh

The material, geometry and mesh of the infinite bar model (§4.3.1, §4.3.2) were retained. Further simulations were run on bars of length 500mm and 1000mm, with 20 radial elements with an aspect ratio of 1.

4.5.2 Boundary Conditions

The infinite bar was modelled by treating the ends as nodes and constraining the axial displacement of the ends to zero. By removing the axial constraint at the ends, the model became that of a finite length bar. The radial displacement constraint on the centre line was retained to enforce axisymmetry.

4.5.3 Post-Processing of Abaqus Results

As the axial displacement constraint on the ends of the bar had been removed, the axial displacement profile could now assume shapes other than complete or half sine waves. An example of the typical axial displacement profile extracted from Abaqus is given in Figure 4-7.

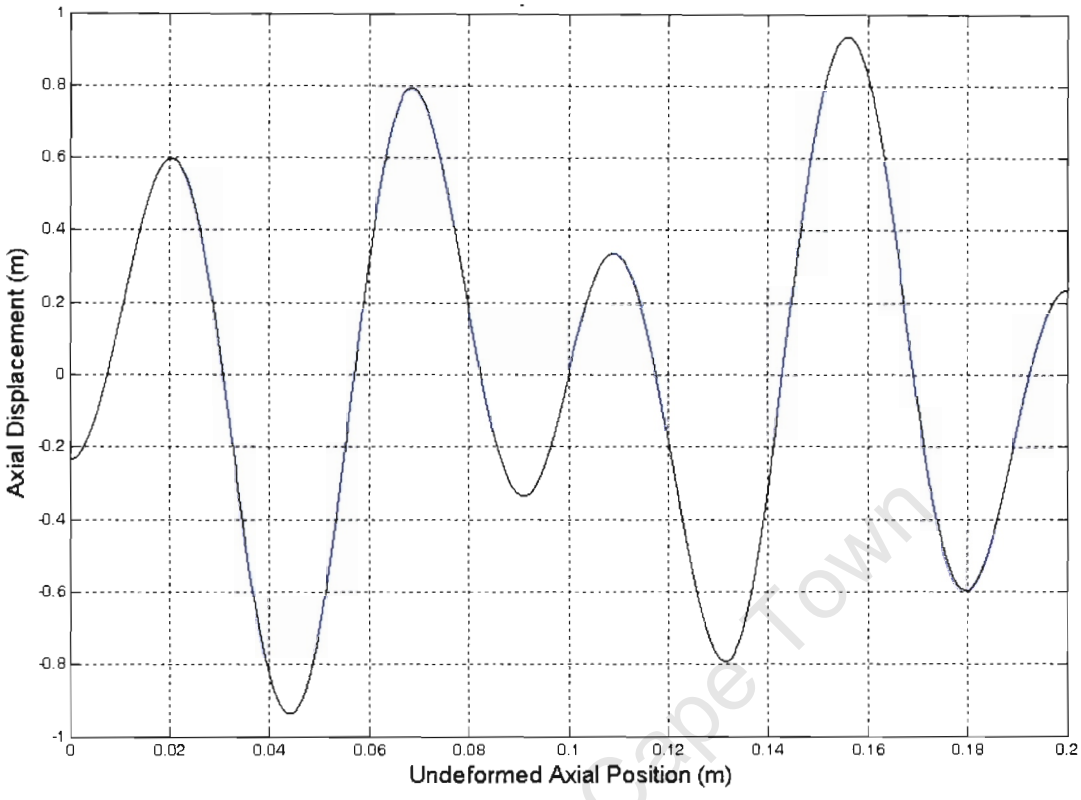


Figure 4-7 Axial displacement profile of a finite bar (unconstrained ends) for mode correspoding to 180 kHz

It is evident from Figure 4-7 that the axial displacement profile of the finite length bar cannot be characterized by a single sine wave. Hence the method of determining the wavelength by counting zero crossings is not applicable to this model. The wavelength was determined by performing a modified Fourier transform on the axial displacement data. The traditional Fourier integral [Eq 2-16] transforms a function from the time domain to the frequency domain. However, the time variable t can be treated as dummy variable and substituted with the spatial dimension z. The Fourier integral then transforms the function from the spatial domain to the wavelength domain. This does not resolve the problem of the mode being characterized by several wavelengths. The characteristic wavelengths were restricted to those with amplitudes greater than half the maximum amplitude in the wavelength spectrum. As the frequency for the mode had already been found, the phase velocity was found for all of the characteristic wavelengths by Eq 4-23.

4.5.4 Results and Discussion

The results of the analysis from the previous section are shown in Figure 4-8.

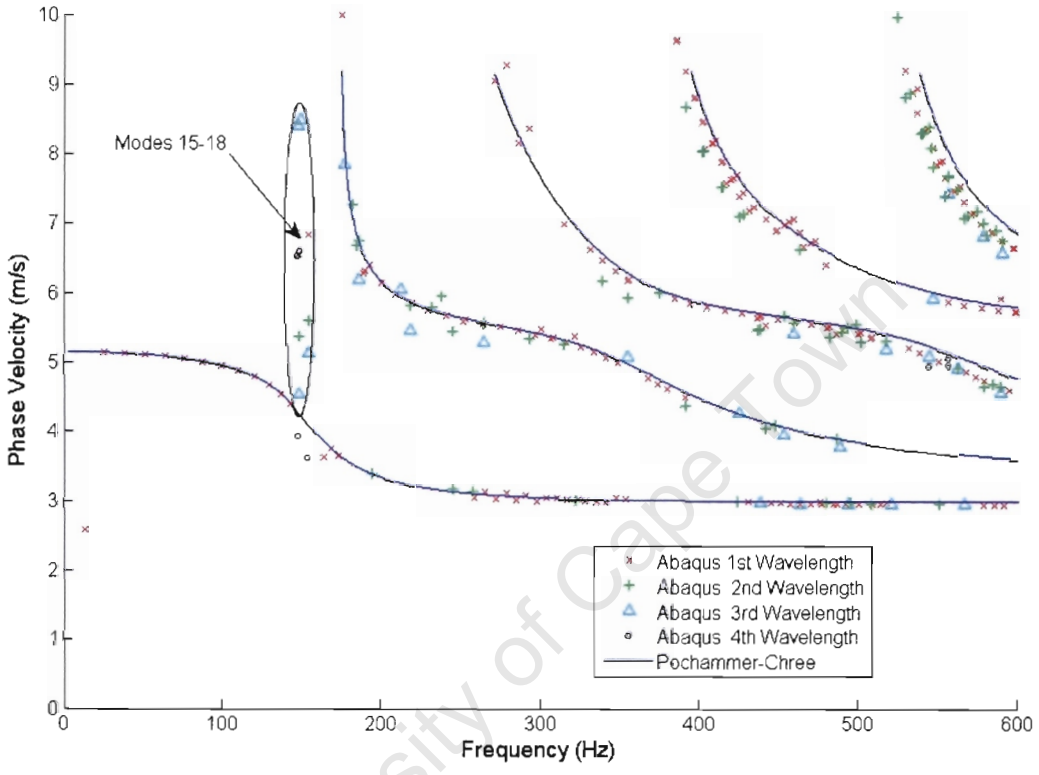


Figure 4-8 Numerical phase velocity frequency prediction for a 200mm bar

The calculated phase velocities are shown for the first 4 characteristic wavelengths of each mode, if these exist. As very few modes had more than 4 wavelengths of significant amplitude, the higher order wavelengths were not plotted. The best agreement between the numerical C_p-f relationship for the finite length bar and the Pochammer-Chree solution is in the region below 150 kHz. The phase velocity prediction for the modes close to 150 kHz (modes 15-18) vary by almost one order of magnitude from the Pochammer-Chree solution. An example of the axial displacement profile of one of these modes is plotted in Figure 4-9.

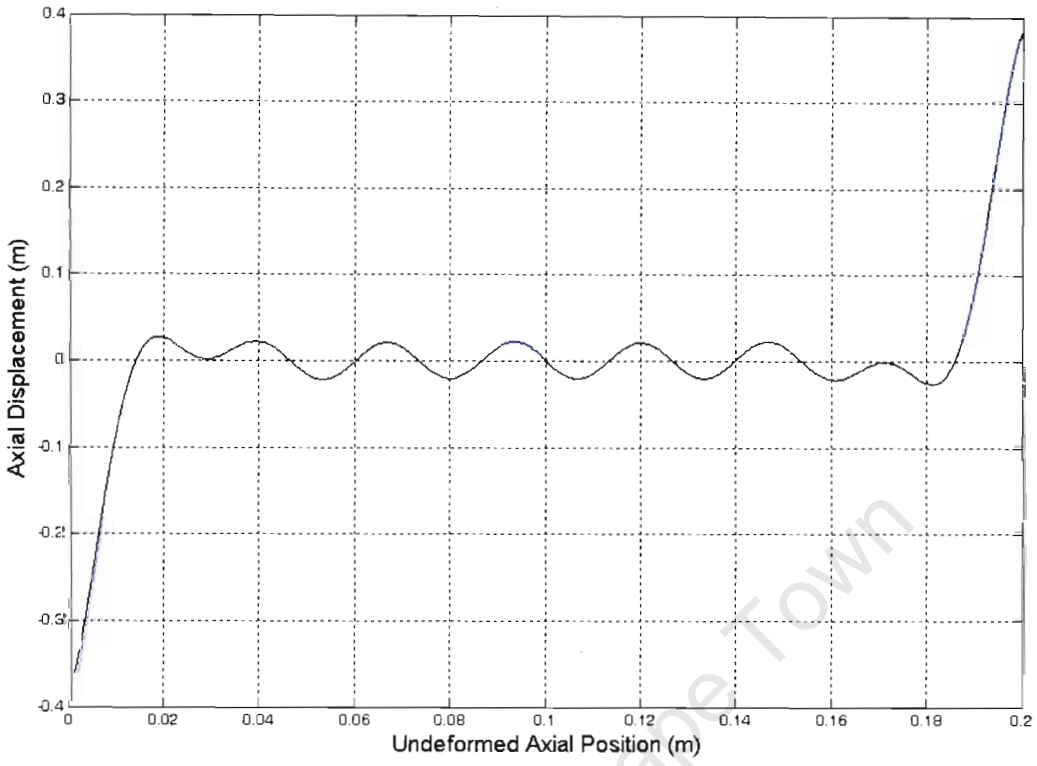


Figure 4-9 Axial displacement profile of mode 17 (153 kHz)

The axial displacement profile of the modes displayed common characteristics, which are very obvious in mode 17. The magnitude of the displacement at the unconstrained ends is much bigger than that in the middle of the bar. When transformed to the wavelength spectrum, these manifest as large amplitudes for wavelengths which are not characteristic of the rest of the bar. As the characteristic wavelengths for the mode are determined according to the largest magnitude, these wavelengths are returned and hence the phase velocity does not converge to the Pochammer-Chree prediction. This behaviour is expected, because the fundamental difference between the finite and infinite bar models is the non-zero displacement at the ends. The effect of relaxing the boundary conditions at the bar ends resulted in these modes which do not correspond to the infinite bar modes of the Pochammer-Chree prediction.

The phase velocity prediction for the first mode is also exactly half what it should be as the wavelength finding algorithm could not find a wavelength that was longer than the bar length. In general the phase velocities predicted for the finite length 200mm

bar approach those of the Pochammer-Chree solution. There is much more scatter in the numerical finite bar prediction than the numerical infinite bar prediction. Additional simulations were run for finite bars of length 500mm and 1000mm, while retaining all other properties. The results for these are shown in Figure 4-10 & Figure 4-11.

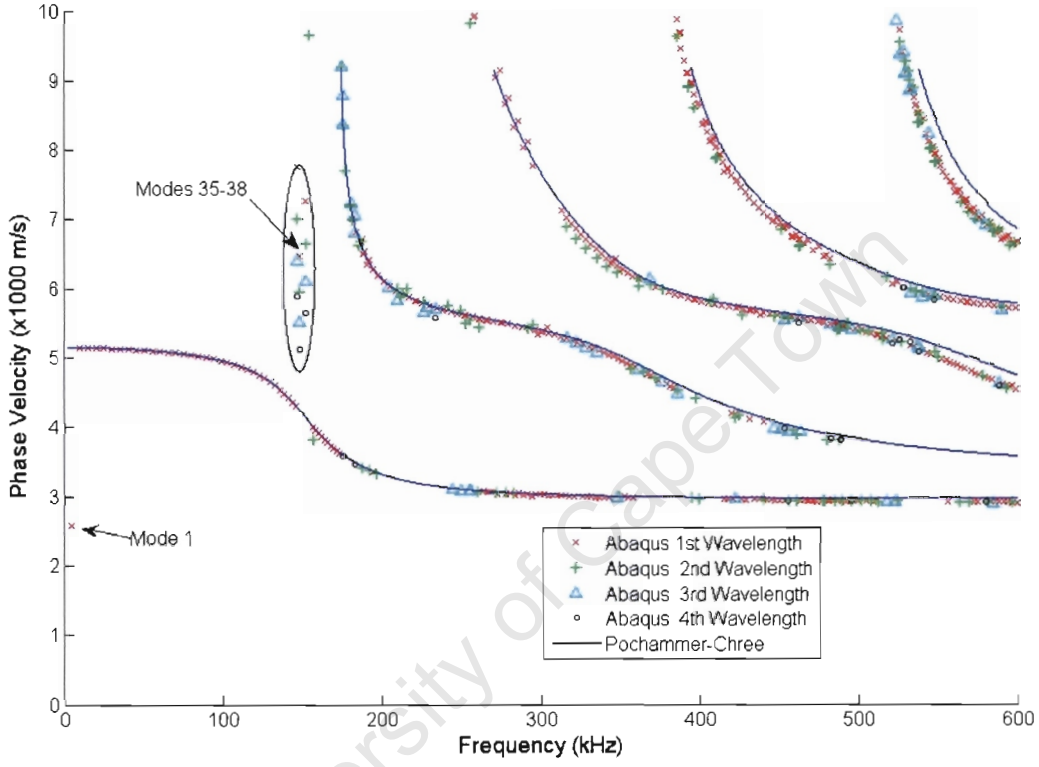


Figure 4-10 Numerical phase velocity - frequency prediction for a 500mm bar

As the length of the bar modelled was increased, while maintaining mesh density, the phase velocity predictions began to approach the Pochammer-Chree solution more closely at higher frequencies. This suggests that as the ratio of the model bar length to the excitation wavelength increases (i.e. scaled frequency increases) any finite bar will begin to approach the infinite bar behaviour predicted by the Pochammer-Chree solution. The modes in the 145-155 kHz frequency range, for the 500 mm and 1000 mm models, exhibited almost identical behaviour to those in the 200mm model.

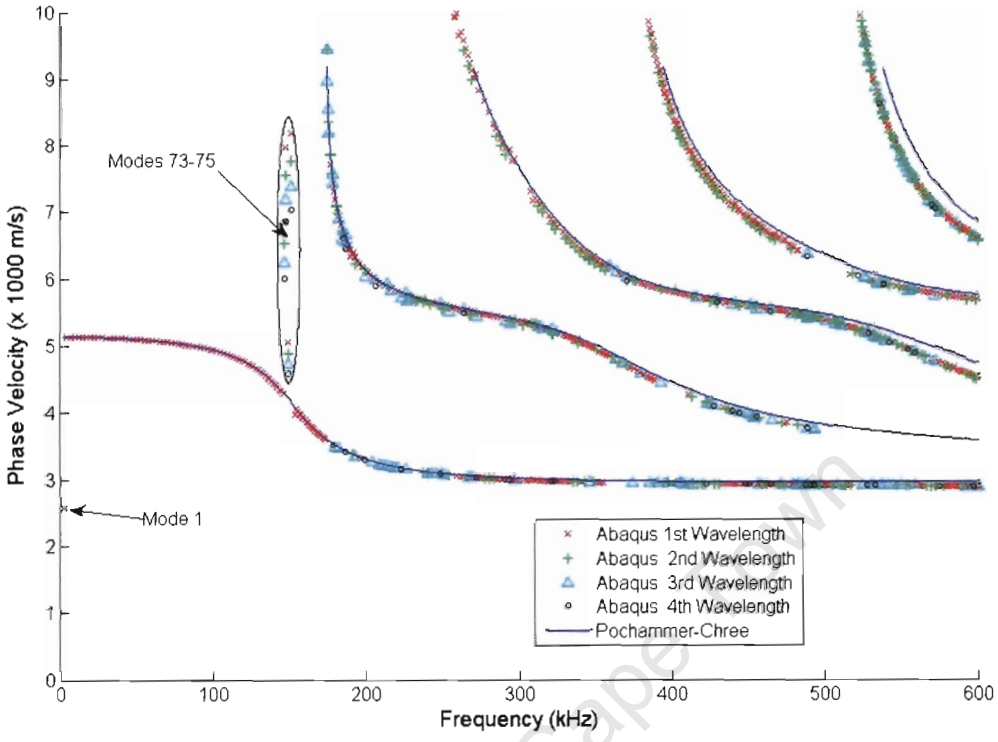


Figure 4-11 Numerical phase velocity - frequency prediction for a 1000mm bar

4.5.5 Comparison of Numerical Predictions for C_p-f Relationship for Infinite and Finite Length Cylinders

The most significant difference between the results of the infinite and finite length models is the association of the Eigen modes with the Pochammer-Chree modes. One Eigen mode for the infinitely long bar corresponded to one Pochammer-Chree mode at that specific frequency.

However, the wavelength analysis of the Eigen modes suggests that one Eigen mode for the finite length bar may be associated with more than one Pochammer-Chree mode. The analysis of the finite bar Eigen modes resulted in wavelengths and hence Cp-f relationships that approached the Pochammer-Chree modes. However, most single finite bar Eigen modes cannot be associated accurately with a single Pochammer-Chree mode. The Eigen vector, which defines the mode shape, does not vary with time. Hence all of the wavelengths associated with that finite length bar

Eigen mode will be coupled in some way, which may affect their propagation. The infinite bar model returns Pochammer-Chree modes with independent Eigen modes hence the propagation of a particular Pochammer-Chree mode should not be affected by the other modes.

The comparison of these results suggests that wave propagation in a finite length cylinder is different from that in infinitely long cylinders. A more detailed investigation of this difference presents an opportunity for future research.

4.6 Dynamic Simulations of Stress Wave Propagation in Pressure Bars

Having ascertained that the modal summation method can accurately predict dispersion, it was possible to compare it with an explicit FE solution for identical inputs. The problem modelled was an axisymmetric cylinder with a pressure applied evenly to one end. The pressure history was a symmetrical trapezoid.

4.6.1 Model Inputs

The modal summation and explicit FE model material and geometry parameters were identical to those used for determining the accuracy of the modal summation solution. (§4.3.1, §4.3.2)

4.6.1.1 Loading

An evenly distributed pressure of 1MPa was applied to one end of the bar. The pressure was varied with respect to time (using the *Amplitude option in ABAQUS) according to a symmetric trapezoid, which is shown in Figure 4-12.

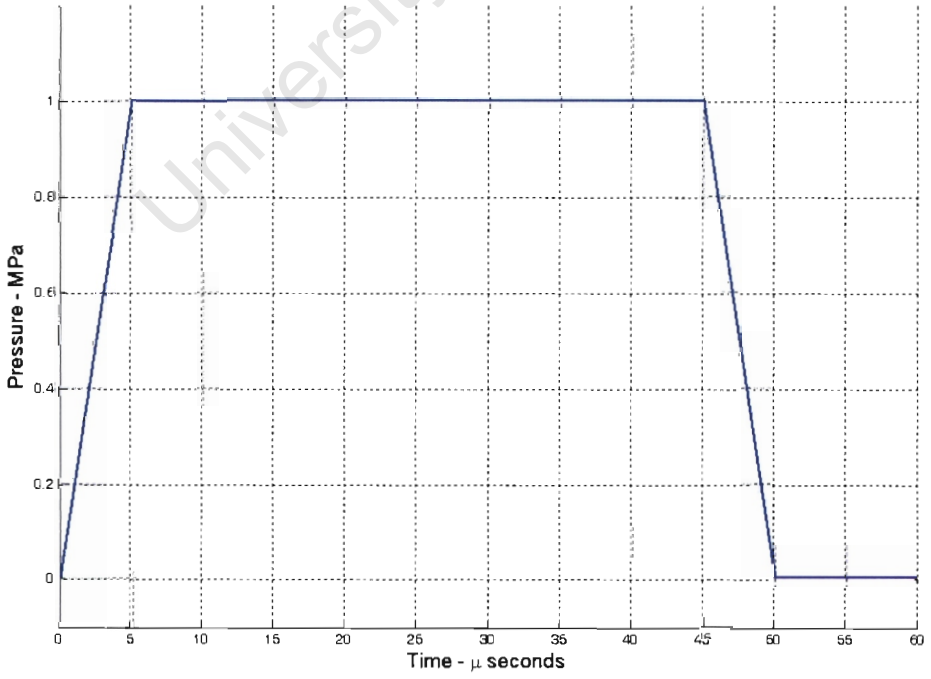


Figure 4-12 Pressure loading of face

4.6.1.2 Boundary Conditions

No axial constraints were imposed on the bar. The nodes at the centre line were constrained radially to preserve axisymmetry but were free to move axially.

4.6.1.3 Mesh

Both the Explicit FE and modal summation models used 20 radial elements and 200 axial elements, with 4 node, linear axisymmetric elements. The modal summation model used full integration elements (CAX4) to avoid spurious zero energy modes. The Explicit FE model used reduced integration elements (CAX4R) as these were the only linear 4 node elements available for an Explicit FE analysis. A dynamic, explicit integration analysis will not be influenced by zero energy modes, which do affect the Eigen value and vector analysis in a modal analysis. While these are different element formulations, the different methods should calculate identical stiffness and mass matrices, for linear shape functions. Hence the results of the Abaqus/Explicit and Abaqus Modal Dynamics analyses should be comparable. The initial explicit simulations were performed with Abaqus' "Enhanced Hourglass Control" option and the default settings for the bulk viscosity parameters.

4.6.1.4 Time Step

The modal summation simulation used a time step Δt of 0.2 μs. Abaqus/Explicit automatically calculated a time step based on the smallest element dimension (Element By Element option). The initial critical time step calculated by Abaqus/Explicit was 0.055 μs.

4.6.2 Comparison of Modal Summation and Explicit Results

The axial strain history was recorded at the outer radius of the bar, 250mm from the loaded face, for both simulations. The modal summation analysis is deemed a purely linear analysis by Abaqus, so the axial engineering strain (E22 data) was recorded. The explicit analysis is deemed to be a non-linear analysis by Abaqus, so the E22 data was unavailable. The logarithmic (true) axial strain (LE22) was recorded instead. As the material only experienced small strains and remained entirely linear in its response, the engineering and true strains may be treated as equivalent.

The results of the initial simulation are shown in Figure 4-13.

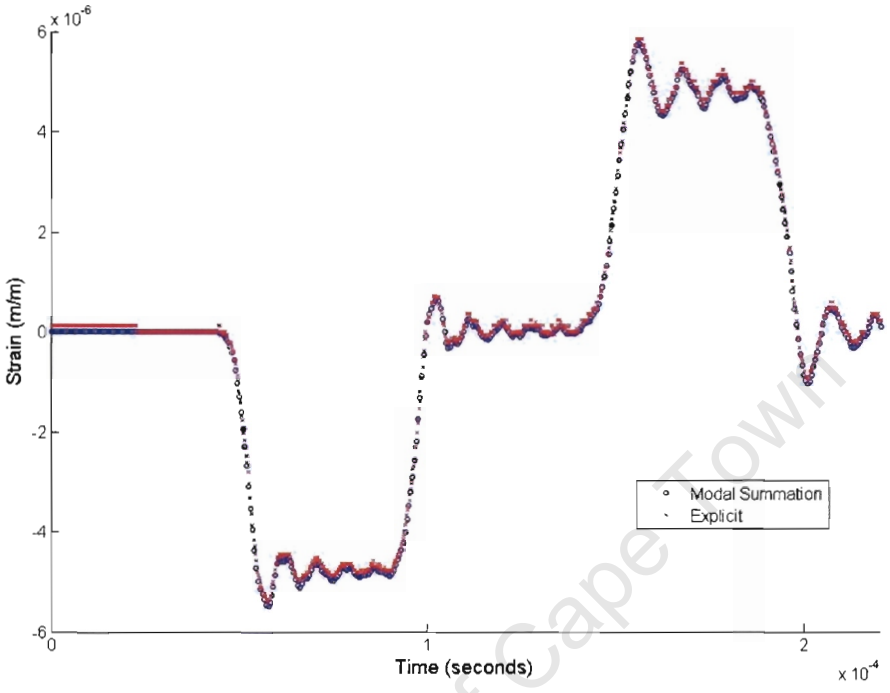


Figure 4-13 Abaqus Modal Summation and Explicit predictions for a trapezoidal pressure loading

The axial strain shown in Figure 4-13 for the modal summation solution is the average of the strain calculated at the four nodes. The explicit solution, using reduced integration, outputs the strain at the integration point which lies at the centroid of the element. The first pass of the pulse is negative as the input load is compressive and hence produces negative strain. The reflected pulse is positive (tensile) as the reflection is off a free surface, which is consistent with 1 dimensional wave theory. There is very good agreement between the explicit and modal summation solutions on the non-oscillating portions of the pulse (the rise and decay). The oscillations are characteristic of dispersion of a trapezoidal pulse in a cylinder. A 1st mode Pochammer-Chree dispersion correction was applied to a trapezoidal pulse to predict its dispersed shape after propagating 250mm. The explicit solution also has a constant non-zero offset for the first 20 μs and some spurious oscillations just prior to the arrival of the main pulse. A comparison of the numerical predictions with the Pochammer-Chree corrected pulse is shown in Figure 4-14.

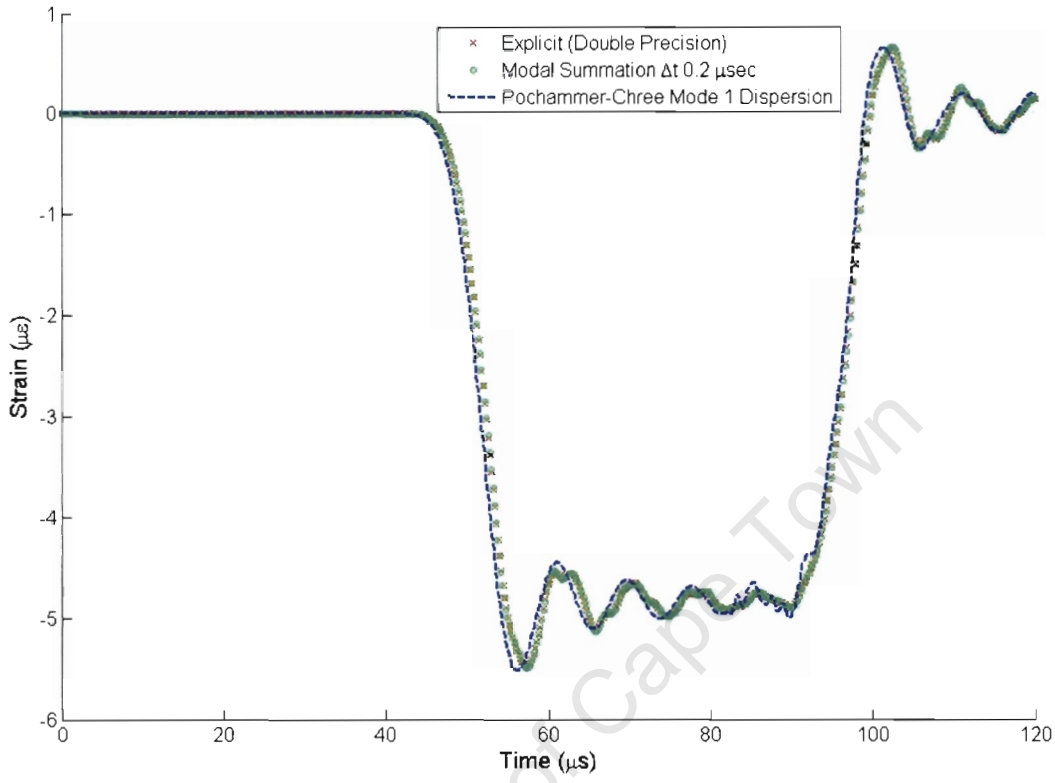


Figure 4-14 Comparison of Explicit (Double Precision), Modal Summation and Pochammer-Chree 1st mode predictions for propagation of a trapezoidal wave

Figure 4-14 shows the numerical and analytical predictions for the propagation of the trapezoidal wave are in good agreement. Variation between the modal summation and explicit FE results is only visible because the results are recorded at slightly different time values. Closer inspection of the numerical predictions (Figure 4-15) showed that the modal summation and explicit FE results do lie on the same curve.

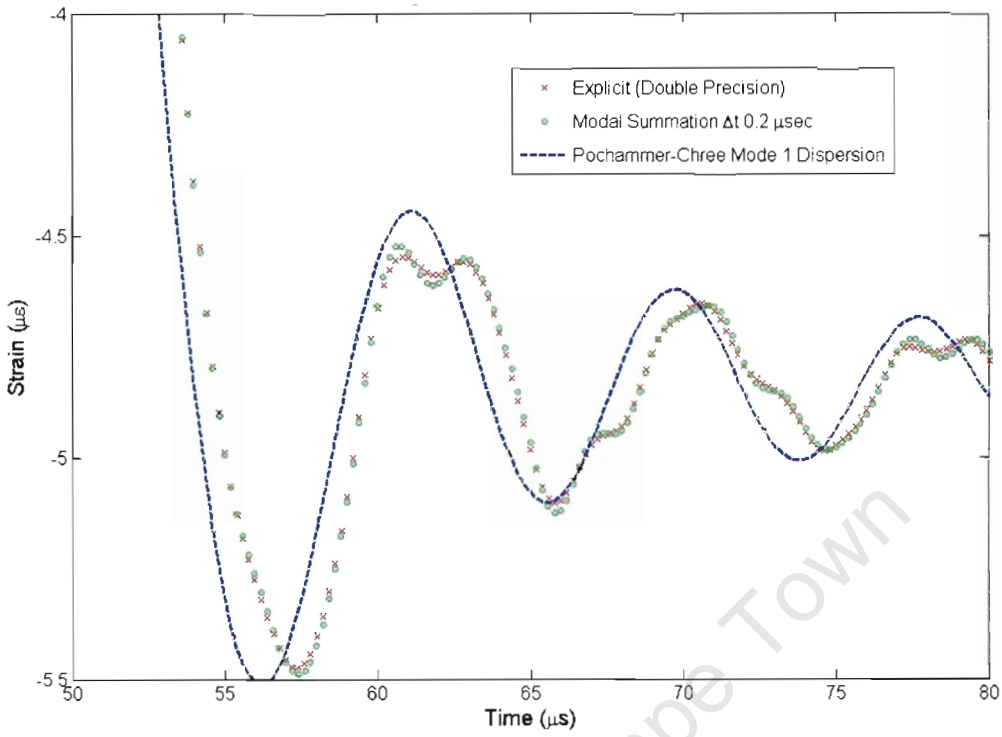


Figure 4-15 Enlarged view of oscillations from Figure 4-14

The Pochammer-Chree prediction reaches its peak 1.5 μs before both the numerical predictions. However, the modal summation and Pochammer-Chree solution predict almost identical amplitudes (Variation<0.5%) for the peaks of the oscillations. The explicit simulation was re-run with the "Double Precision" option, which increases the number of decimal places stored in Abaqus Explicit calculations, thereby increasing accuracy. This resulted in the initial offset present in the previous explicit simulation being eliminated. The "Double Precision" explicit FEA solution (Figure 4-15) also showed much better agreement (Variation<0.3%) with the modal summation solution than the "Single Precision" explicit FEA solution. The Pochammer-Chree prediction has a higher frequency oscillation superimposed on the second half of the trapezoid plateau. This behaviour does not feature in either of the numerical predictions. These oscillations may indicate the presence of higher order Pochammer-Chree modes. As the dispersion correction was performed using the 1st mode, any higher order modes would be phase shifted incorrectly. The numerical predictions were then compared to a Pochammer-Chree dispersion correction utilizing 1st and 2nd mode correction (Figure 4-16).

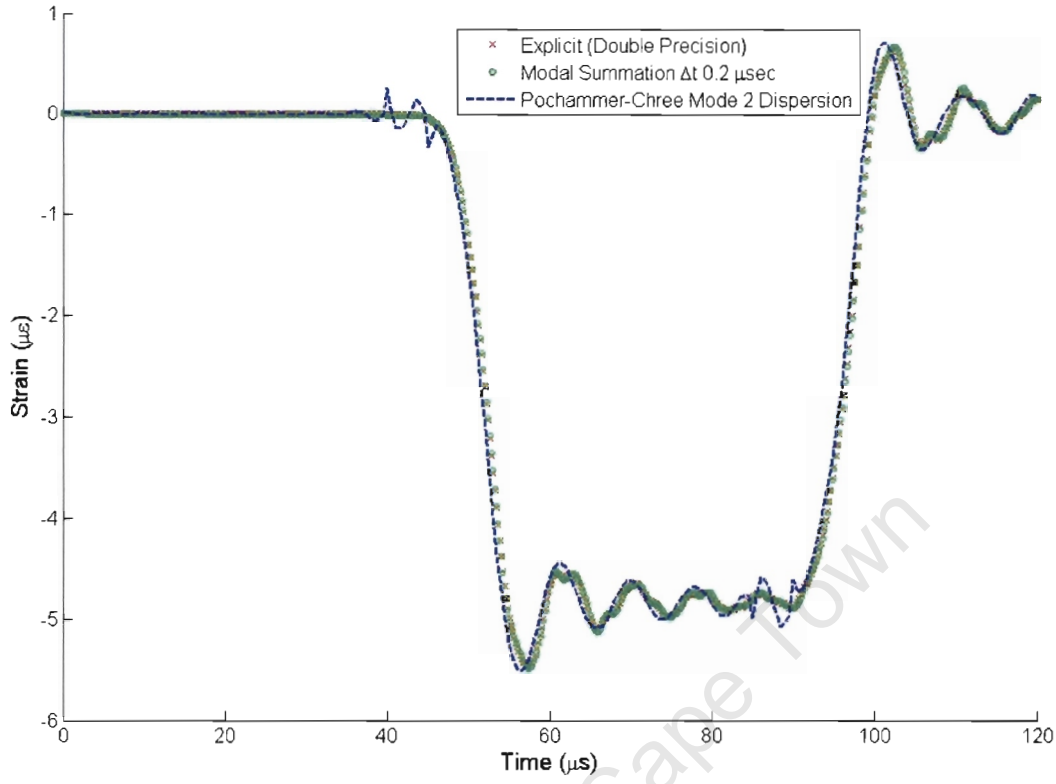


Figure 4-16 Comparison of numerical predictions with Pochammer-Chree 2nd mode correction for propagation of a trapezoidal pulse

The 2nd mode Pochammer-Chree dispersion correction reduced the higher frequency oscillations on the plateau of the trapezoid, but did not eliminate them completely. However, it did introduce oscillations at a similar frequency that precede the arrival of the main pulse, that are clearly not indicative of real wave propagation.

This spurious behaviour raised questions about multi-mode Pochammer-Chree dispersion correction applied to HPB signals. The 1st Pochammer-Chree mode has an axial displacement shape function ($U(r)$) with at most one nodal cylinder. This is consistent with an evenly distributed load or impact with a striker bar of similar diameter to the HPB. However, higher Pochammer-Chree modes have more nodal cylinders. A loading function would have to have a similar radial profile, in addition to be in the correct frequency range, to excite these higher modes. Thus conventional compressive SHPB loading is unlikely to significantly excite higher Pochammer-Chree modes.

4.6.2.1 Computational Expense

The specifications of the computer used for these simulations, and the computational expense, are summarised in Table 4-3.

Table 4-3 Computer specifications

Processor	Pentium™ IV 2.4 GHz
Memory	1 GByte
Operating System	Windows XP ™ Service Pack 2
Abaqus/Explicit simulation time	~6 minutes
Abaqus Modal Dynamics simulation time	~10 hours

The explicit FEA was computationally much cheaper than the modal summation simulation. Computing the Eigen modes for the modal summation took approximately 6 minutes, while completing the transient modal analysis took almost 10 hours.

4.7 Artificial Damping in Numerical Modelling

Explicit FEA codes such as Abaqus include artificial numerical damping. These are intended to improve modelling of high speed events by introducing an artificial pressure in the element proportional to the volumetric strain rate $\dot{\varepsilon}_{vol}$. Abaqus/Explicit v6.5.1 [30] makes use of linear and quadratic artificial damping. The linear damping term is:

$$P_{bv1} = b_1 \rho C_o L_e \dot{\varepsilon}_{vol}$$

Eq 4-24

The quadratic damping term is:

$$P_{bv2} = \rho (b_2 L_e \dot{\varepsilon}_{vol})^2$$

Eq 4-25

The bulk viscosity terms b_1 and b_2 have default values of 0.06 and 1.2 respectively but may be changed by the user.

These artificial damping terms will affect high frequency oscillations more than lower frequencies, as these induce higher volumetric strain rates. The trapezoidal loading function used in previous simulations does not have significant high frequency content. Any high frequency content present in the trapezoid is due to the sharp corners (discontinuous first derivative) and is difficult to control. The Gauss Windowed Sine wave (GWS) used by Tyas [20] is a function that does contain significant high frequency content, but importantly allows the user to control the location, amplitude and bandwidth of the high frequency content. The equation of the GWS is:

$$f(t) = A.Sin(2\pi f_s t).e^{-(\frac{t-to}{\kappa})^2}$$

Eq 4-26

Tyas used the parameters in Table 4-4 for his simulations with a GWS, shown in Figure 4-17.

Table 4-4 GWS Parameters

Parameter	Symbol	Value
Amplitude	A	1
Sinusoid Frequency	f_s	83 kHz
Window Centre	t_o	60 μs
Window Width	κ	56 μs

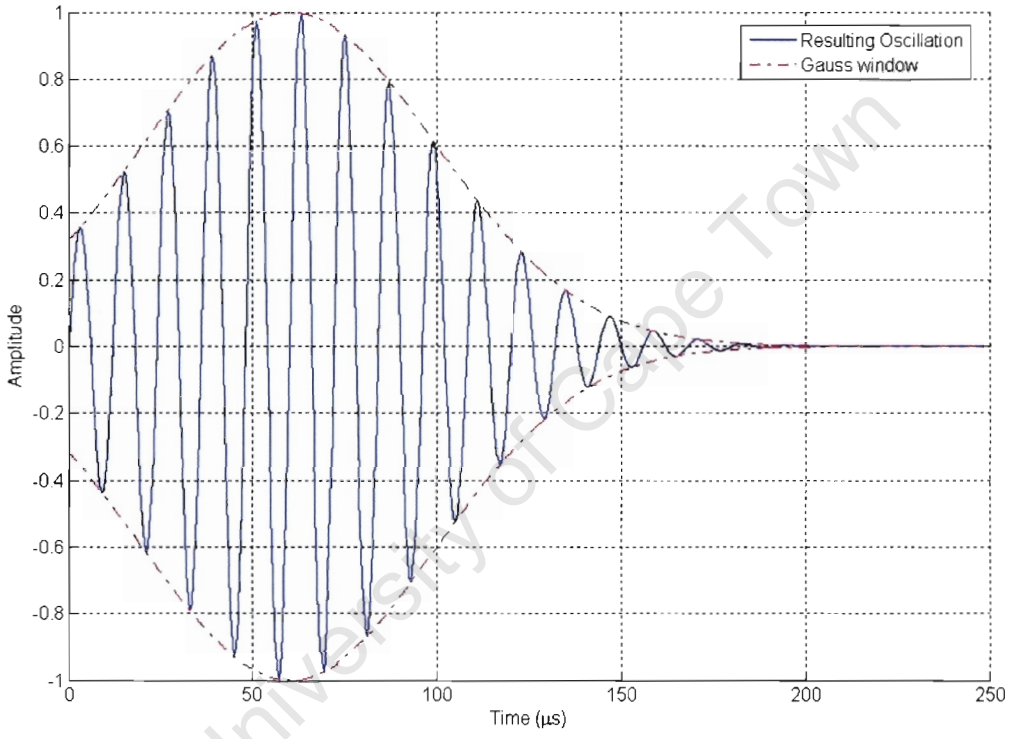


Figure 4-17 83 kHz Gauss Windowed Sine Wave

When transformed to the frequency domain by a FFT, the GWS has a low bandwidth amplitude spectrum when compared to a trapezoidal pulse, as shown in Figure 4-18.

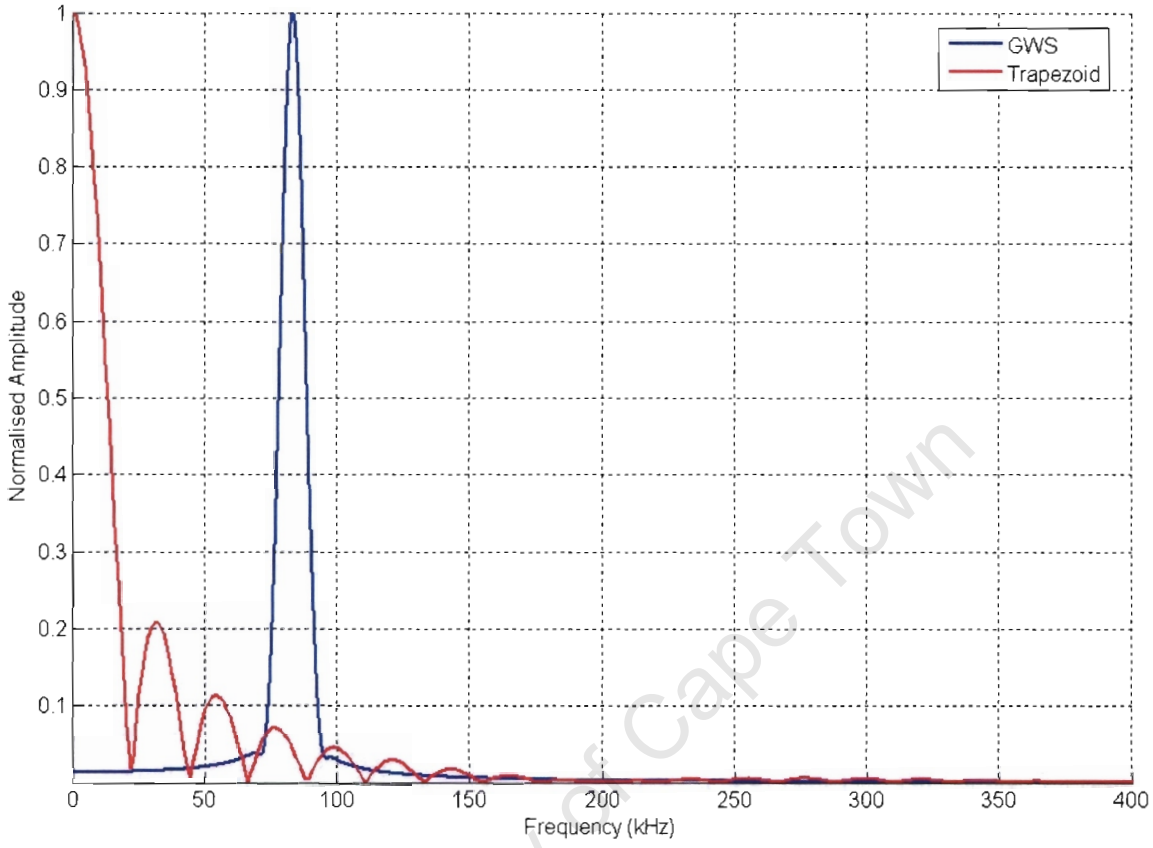


Figure 4-18 Amplitude spectra of a trapezoidal pulse and 83 kHz GWS

The Fourier terms of significant magnitude are centred at the sinusoid frequency f_s. The use of a GWS as a loading function permits analysis, using Abaqus/Explicit, of HPB models' response to high frequencies without significant low frequency loading. While a GWS is not representative of loading in the "real world", which always contains significant low frequency Fourier terms, it is useful for investigating behaviour that is frequency dependent. It is also not possible to load a HPB in the laboratory in such a way that the resulting stress wave is a GWS.

4.7.1 Parametric investigation of numerical damping

The numerical damping included in Abaqus/Explicit is controlled by the parameters b_1 and b_2. A series of simulations, using a GWS as a loading function, were run to quantify the effect of changes to these parameters on dispersion of waves in a HPB. The HPB model parameters are shown in Table 4-5.

Table 4-5 HPB Model parameters

Parameter	Value
Bar length	2500mm
Bar diameter	24mm
Elastic Modulus	207 GPa
Poisson's Ratio	0.26
Radial element length	0.6 mm / element
Axial element length	1 mm / element

Simulations were run using a GWS with damping parameters as summarised in Table 4-6.

Table 4-6 Damping parameters for different simulations

Simulation Number	Bulk Viscosity Factor b_1	Bulk Viscosity Factor b_2
BV 1 (Default values)	0.06	1.2
BV 2	0.03	1.2
BV 3	0.01	1.2
BV 4	0.06	0.6
BV 5	0.01	0.3

The GWS had the same parameters as that in Table 4-4 except for f_s = 75 kHz. The results of simulation BV 1 and BV 2 are compared with a GWS dispersed by Pochammer-Chree theory in Figure 4-19. An attempt was made to simulate the same model and load with Abaqus Modal Dynamics (modal summation) but the simulation was terminated due to insufficient computational resources. No further modal summation simulations were attempted.

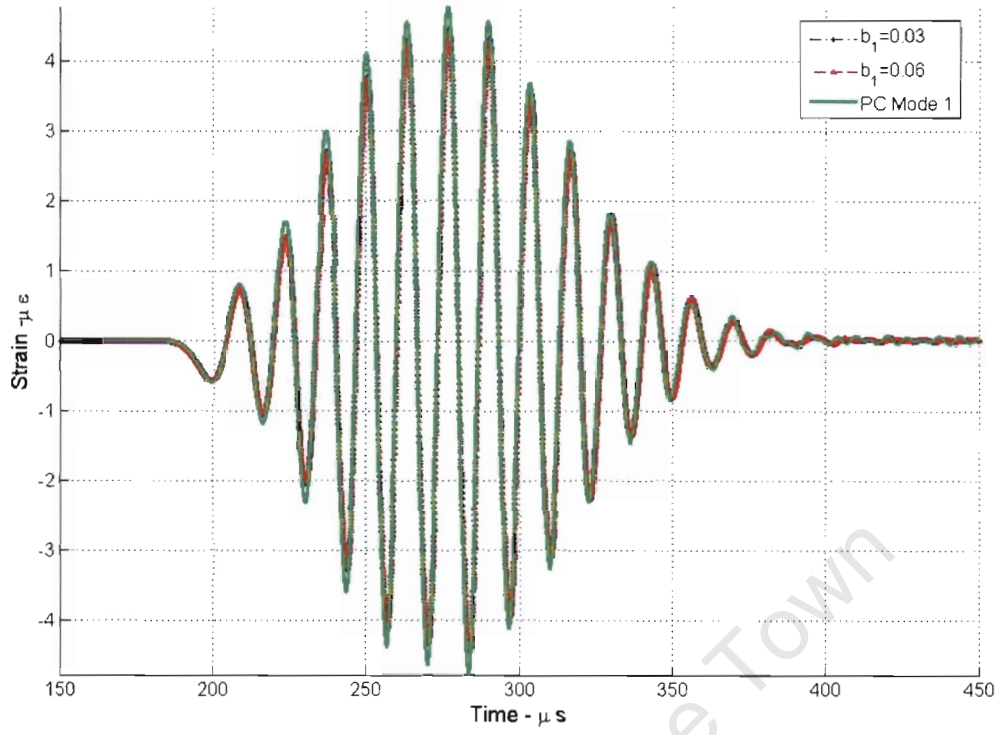


Figure 4-19 Comparison of numerical and analytical predictions for 75 kHz GWS propagated 1000mm

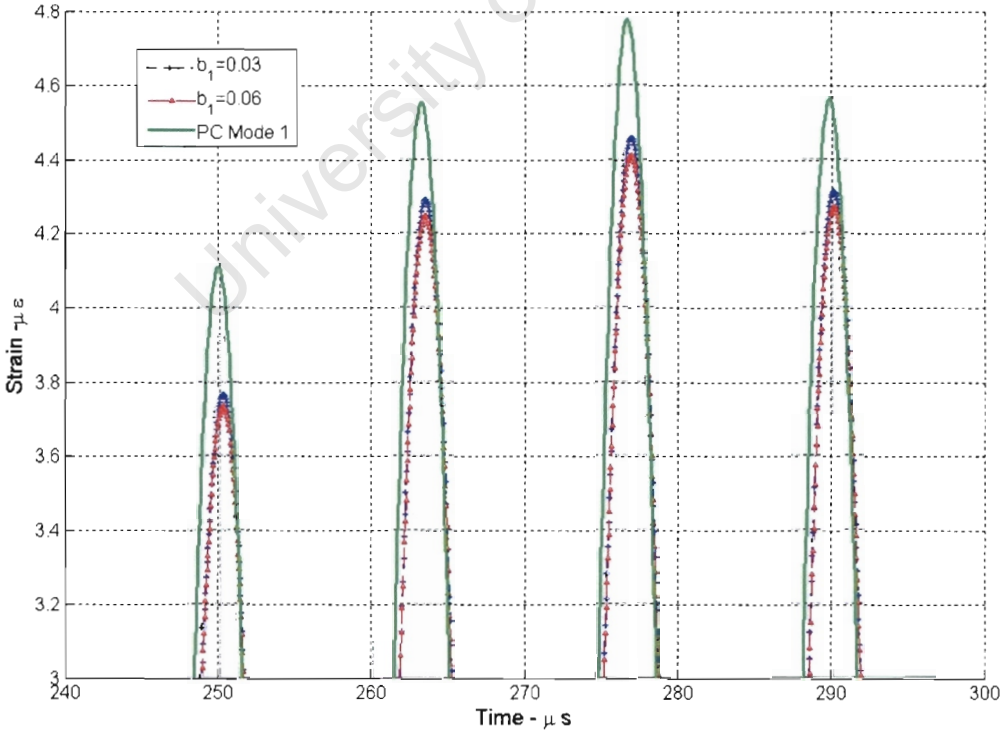


Figure 4-20 Enlarged view of peaks from Figure 4-19

Figure 4-20 shows there is very little variation between the results of simulations BV 1 and BV 2, despite a 50% change in b_1. The peak strains recorded in the different simulations are shown in Table 4-7.

Table 4-7 Peak strains from Simulations 1 and 2

Simulation	Peak Strain (με)	Variation to undispersed pulse
BV 1 Linear viscosity b_1=0.06	4.41	-8.5 %
BV 2 Linear viscosity b_1=0.03	4.46	-7.5 %
Pochammer-Chree dispersed signal	4.77	-1.0%
Undispersed GWS	4.82	N/A

It is evident that the peak amplitudes of the strains recorded by the explicit FEA simulations are significantly less than that of the signal dispersed according to Pochammer-Chree theory. The peak strains of all of the propagated signals are less than that of the undispersed pulse, which is expected. Closer inspection of the peaks revealed that the two numerical results were in phase while the analytically dispersed GWS leads the numerical results slightly. This is also expected as damping also causes the lengthening of the period of the response when compared to the forcing function. Bathe [31] showed that any explicit numerical integration of an oscillating system will exhibit amplitude decay and lengthening of period when compared to the exact solution.

The results of the other simulations do not differ visually from those shown in Figure 4-19. The results are summarised in Table 4-8.

Table 4-8 Summary of viscosity parameter investigation

Simulation	Bulk Viscosity Parameters		Peak Strain	Variation to dispersed pulse
	Linear b_1	Quadratic b_2	με	
BV 1	0.06	1.2	4.41	-7.5%
BV 2	0.03	1.2	4.46	-6.5%
BV 3	0.01	1.2	4.49	-5.8%
BV 4	0.06	0.6	4.41	-7.5%
BV 5	0.01	0.3	4.49	-5.8%
Pochammer-Chree dispersed pulse			4.77	N/A

There is less than 2% variation of the peak strain recorded in all of the numerical simulations. The linear bulk viscosity parameter b_1 had a greater effect than the quadratic factor b_2. The quadratic factor b_2 was changed from 1.2 in simulation BV 3 to 0.3 in simulation BV 5 (75% decrease), yet the difference in peak strain was only differed only after the third decimal place. It is debatable as to whether the differences between the analytical and numerical predictions are due to the artificial numerical damping or the amplitude decay and period lengthening effects normally associated with explicit numerical integration.

4.7.2 Higher Frequency Effects

The simulations performed with f_s=75 kHz showed a significant difference between the numerical and analytical predictions. However, changes to the damping parameters did not significantly affect the results. The artificial damping is proportional to volumetric strain rate. Increasing the frequency of the loading function increases volumetric strain rate. This will cause the artificial numerical damping to have an even greater effect on the simulation results. Conventional gas gun loading of a HPB generally does not excite frequencies higher than 100 kHz. However, using a HPB as a pressure transducer in a blast experiment, or in a Taylor Impact Test, may excite much higher frequencies. Hence artificial damping is likely to have a much larger effect on the results of explicit FE simulations of blast or Taylor Impact experiments.

Further simulations were performed using a GWS loading function with f_s = 200 kHz, with the same damping parameters as outlined in Table 4-6. The value of 200 kHz was chosen as the higher Pochammer-Chree modes are only present above approximately 170 kHz. Earlier investigations (§4.4) had shown that Abaqus Modal Dynamics (modal summation) could accurately predict the frequencies and shapes of the higher Pochammer-Chree modes. However, the trapezoidal pulses used to compare Abaqus/Explicit and Abaqus Modal Dynamics do not have sufficient high frequency content to excite the higher Pochammer-Chree modes. Modelling a 200 kHz GWS loading function ensures sufficient high frequency content to excite higher Pochammer-Chree modes.

4.7.2.1 Comparison of numerical prediction with 1st Pochammer-Chree mode dispersion correction

A simulation was run using a 200 kHz GWS in Abaqus/Explicit, using the same bar model as previous simulations. The damping parameters were b_1 = 0.01 and b_2 = 0.6. The results of the simulation are compared with analytically dispersed pulse, using only the 1st Pochammer-Chree mode for correction, in Figure 4-21.

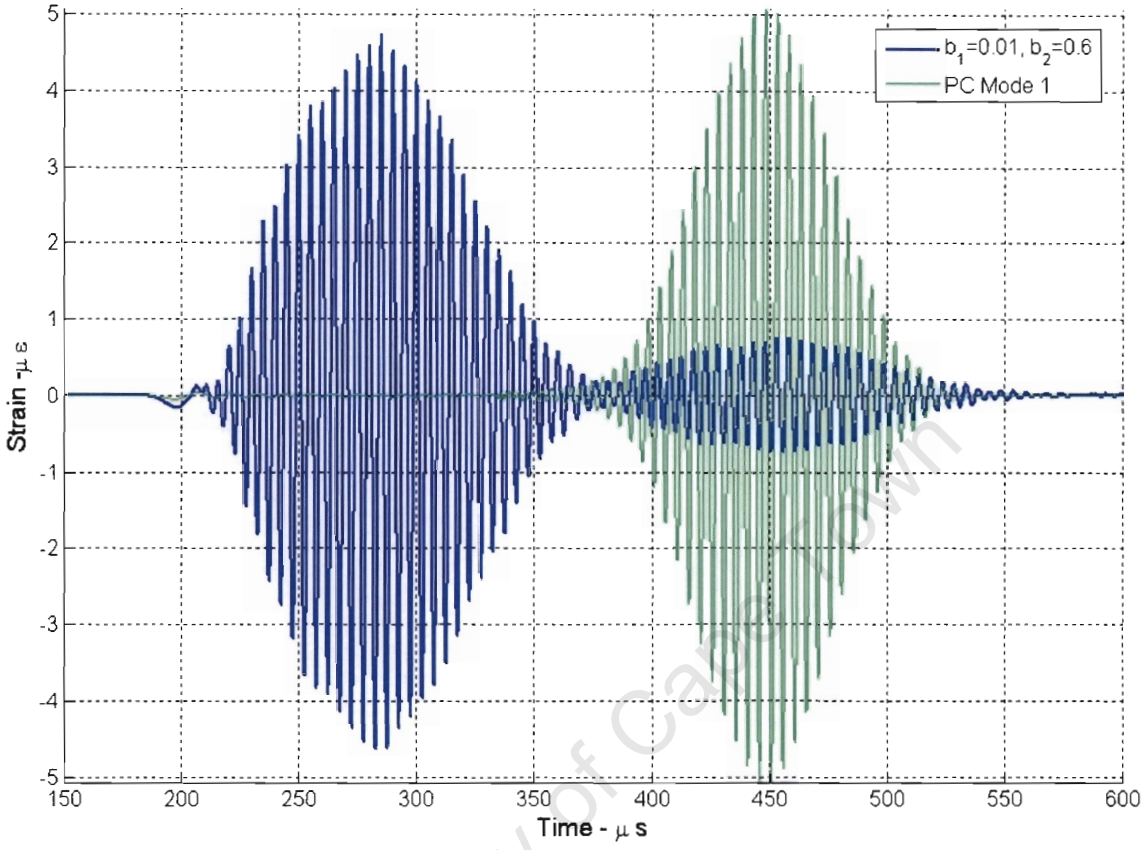


Figure 4-21 Comparison of numerical and 1st Pochammer-Chree mode analytical prediction for a 200 kHz GWS propagated 1000mm

The most obvious discrepancy is that the numerically simulated GWS leads the analytically dispersed pulse by almost 170 μs. This can be attributed to the largest Fourier components of the 200 kHz GWS being in the frequency range where the 2nd Pochammer-Chree mode, not the 1st, is dominant. In addition, the numerical simulation has a second, smaller pulse trailing the main body of the pulse. The peak of the second pulse almost coincides with that of the analytically dispersed pulse, which was corrected using the 1st Pochammer-Chree mode only. This suggests that at least a portion of the pulse was propagating according to the 1st Pochammer-Chree mode. However, it was obvious from the gross differences between analytical and numerical results that using only the 1st Pochammer-Chree mode for dispersion correction was not appropriate.

4.7.2.2 Comparison of numerical prediction with 1st and 2nd mode Pochammer-Chree dispersion correction

The same 200 kHz GWS was correcting analytically using the 1st and 2nd Pochammer-Chree modes. The Abaqus/Explicit simulation was re-run, with $b_1 = 0$ and $b_2 = 0$, in an attempt to remove the numerical damping effects completely. The results are shown in Figure 4-22.

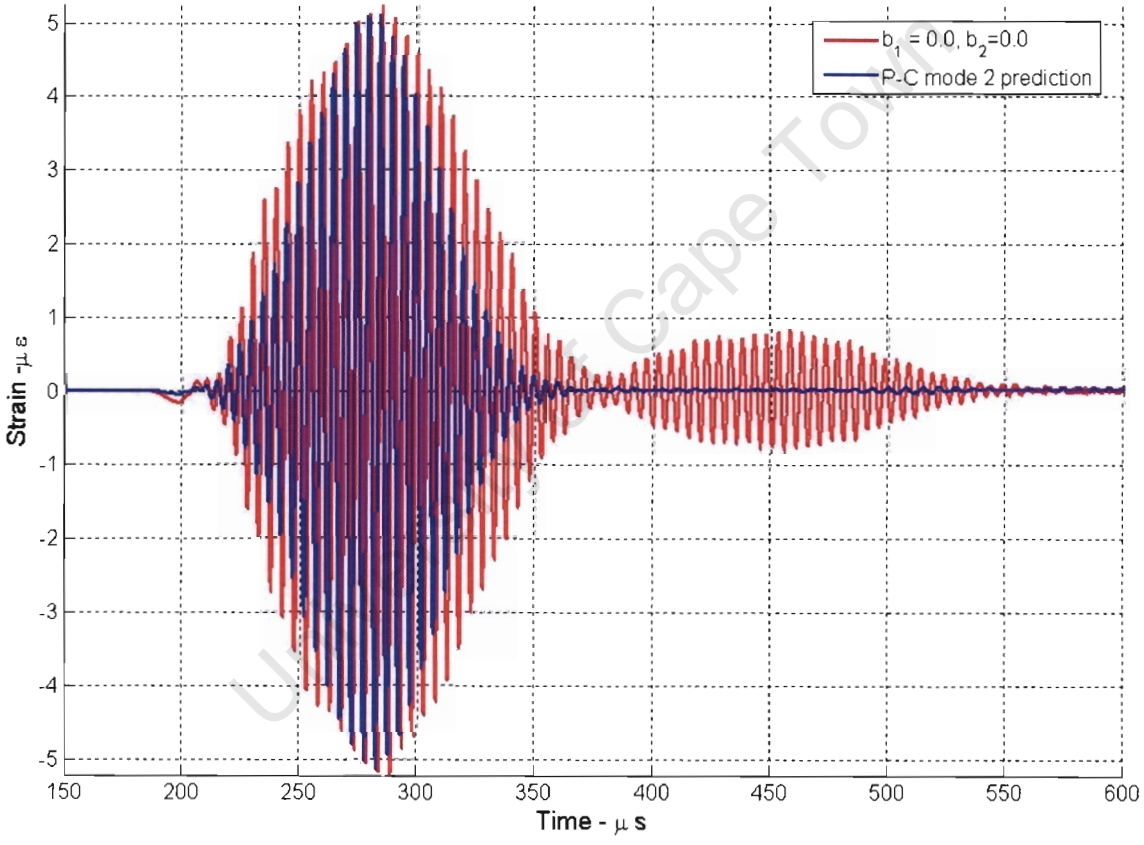


Figure 4-22 Comparison of numerical prediction with no damping, and 1st and 2nd Pochammer-Chree mode dispersion correction for a 200 kHz GWS propagated 1000mm

Using 1st and 2nd Pochammer-Chree modes for dispersion correction resulted in much better alignment of the analytical and numerical predictions. The peak strains of analytically dispersed pulse arrives approximately 4 μs before that of the numerical prediction. The peak strains of the numerical and analytical predictions are 5.24 με

and 5.10 με respectively, which is a difference of 2.7%. However, the shapes of the numerically and analytically predicted results are very different. The numerical prediction still has a second pulse trailing the main body. The numerical damping parameters were both set to 0, so the second trailing pulse cannot be attributed to artificial numerical damping effects.

4.7.3 Differences between the analytical dispersion correction method and the numerical predictions

The analytical dispersion correction curves in use correct the Fourier components of the loading function according to the Pochammer-Chree mode with the highest group velocity, based on the experimental work of Lee and Crawford [28]. The Pochammer-Chree solution permits the presence of two or more modes of vibration at the same frequency, once the higher modes are present. These different modes propagate at different phase velocities. However, the current analytical dispersion correction method corrects all modes present at a given frequency according to one phase velocity. This is due to the limitation of the discrete Fourier methods- each frequency value is associated with only one amplitude. It is not currently possible to determine what portion of this amplitude of the loading function or input pulse belongs to a particular Pochammer-Chree mode. While it is possible to correct for different Pochammer-Chree modes over different frequency ranges, it is not possible to correct for more than one Pochammer-Chree mode at a given frequency.

The second, trailing pulse shown in Figure 4-21, may align with the 1st Pochammer-Chree mode correction because those are the components of the pulse which propagate according to the 1st Pochammer-Chree mode. This suggests that the numerical solution is permitting different modes of vibration to propagate at different, independent phase velocities, which is contrary to Lee and Crawford's [28] conclusions. These uncertainties regarding correction of more than one mode at a given frequency prevent conclusive comparisons of the numerical and analytical results. This phenomenon warrants closer investigation at a later date.

4.8 Concluding Remarks

The modal summation analysis of an infinite cylindrical bar verified that the Pochammer-Chree modes would manifest naturally as the Eigen values and vectors of a discretised numerical model. There was no significant difference between the results of modal summation and explicit FEA simulations of realistic gas gun loading of SHPB. These numerical simulation results agreed well with trapezoidal pulses analytically corrected using the Pochammer-Chree dispersion solutions. Modelling of higher frequency oscillatory loading revealed that the artificial numerical damping incorporated in Abaqus/Explicit does affect the amplitude and phase of wave propagation in HPBs. Changes to the artificial numerical damping parameters do not significantly affect the numerical predictions below 100 kHz. Hence changes to the damping parameters are not recommended for loading functions with no significant frequency content above 100 kHz. The large discrepancies between the numerical and analytical predictions for high frequency (200 kHz) pulses prevent any definite statements about numerical modelling of such pulses.

5 Conclusions

The propagation of high frequency (200 kHz) oscillatory loads, in the form of a Gauss Windowed Sine Wave (GWS), was affected by the numerical damping present in Abaqus/Explicit. The damping definitely affected the amplitude and phase of the higher frequency components significantly. At frequencies less than 100 kHz, it is not possible to separate the error incurred by the artificial numerical damping from that of the explicit numerical integration. At a frequency of 200 kHz, other discrepancies between the numerical results and analytically dispersed pulse persisted, even when the artificial numerical damping was removed completely. It is not possible to state with any certainty whether the analytical or numerical predictions are more correct for high frequency, multiple mode wave propagation. However, conventional SHPB experiments result in pulses with approximately trapezoidal shapes where the significant frequency content does not exceed 100 kHz and the majority of the energy in the pulse is contained in the lower (<25 kHz) frequencies. Hence explicit FEA may be used to accurately model conventional SHPB experiments.

5 Conclusions

The following summarises the conclusions of the investigation:

The code written to solve the Pochammer-Chree dispersion equations for higher modes was efficient and its results were readily incorporated in the existing dispersion correction code. The code may be easily and quickly used to generate dispersion correction maps for linear elastic solid cylinders, for frequencies less than 1.25 MHz.

The phase velocity – frequency (C_p-f) relationship determined from a modal summation (Eigen value) analysis of an infinite bar agreed closely with that predicted by the Pochammer-Chree solution. Each Eigen mode corresponded to a single Pochammer-Chree mode. As the mesh density of the numerical model increased, the variation between the numerical and analytical solutions decreased rapidly. Hence the modal summation method will predict the same mode shapes and phase velocities as the Pochammer-Chree solution.

The modal summation method was applied to a finite length bar to determine its C_p-f relationship, which has no known analytical solution. As the length of bar modelled was increased, the C_p-f relationship tended towards the Pochammer-Chree solution. However, most finite bar Eigen modes contained more than one significant wavelength, which should propagate at different velocities. This suggests that the higher modes of vibration may be coupled, in a currently undetermined manner, in a finite length bar.

Dynamic pressure loading of a HPB was modelled using modal summation (Abaqus Modal Dynamics) and an explicit FEA code (Abaqus/Explicit). For an evenly distributed, trapezoidally ramped load, there was no variation between the modal summation and explicit FEA results. These results were compared with a stress wave dispersed according to the Pochammer-Chree solution. There was no significant variation between the numerical results and the analytical results.

6 Recommendations

This study has established a foundation for future research into numerical modelling of HPB experiments. The following recommendations are made for numerical modelling of HPB experiments:

Realistic SHPB loading, with no significant frequency content above 100 kHz, may be accurately modelled using explicit FEA. The mesh should use square elements, with 20 elements across the radius. Under these conditions, there is little gain in changing the artificial numerical damping from the default parameters.

The following present opportunities for further investigation:

The simulations performed in this investigation were based on evenly distributed pressure loads. Simulation of tensile or shear HPB experiments will require the modelling of non-evenly distributed or concentrated loads. These conditions should be simulated and compared to the analytically dispersed pulses and experimental data. However, a thorough investigation of this requires a large number of numerical simulations involving different distributions of the load. The numerical predictions would also have to be compared to experimental results where the load distribution was varied.

All of the comparisons between numerical simulations and analytically dispersed pulses were based on strains read at the outer radius of the HPB. This is because all of the available experimental data from HPB experiments is obtained from surface mounted strain gauges. However, the primary interest in any HPB experiment is displacement-time history or stress-time history across the radius of the end faces of a HPB, which can be determined from explicit FE models. Analytical correction methods, based on the Pochammer-Chree solution, have been devised to predict the variation of displacement or stress across the radius at the bar end (Tyas and Watson [20], Merle and Zhao [32]). However, the Pochammer-Chree solution is not truly

applicable to the end faces, as it was derived for an infinitely long bar. Hence it is not sensible to compare the FE predictions for the bar ends to these analytical predictions. However, if it were possible to measure the displacement-time history of the unloaded face of a HPB, this could be compared to the results of the explicit FE models. Once the accuracy of the FE models has been determined, the FE results may be used to more accurately determine the radial variation of the deflection or stress at the faces of the specimen. This removes the assumptions regarding uniformity of displacement across the radius of the HPB, which improves the accuracy of the stress-strain relationship derived from SHPB experiments.

The loading function for any numerical model of a HPB is generally assumed to be a certain shape. The accuracy of the numerical predictions will obviously be highly dependent on the accuracy of the loading function. Inverse modelling can be applied to the loading function to increase its accuracy – the parameterised loading function can be modified until the numerical prediction matches that of an experimentally measured strain-time history at one strain gauge station. The model can be verified by comparing the numerical prediction at another axial position to an independent strain gauge station at the corresponding position. This method may be applied to any configuration and loading of a HPB for which two strain gauge readings are available on the same HPB.

The current analytical dispersion correction method cannot account accurately for multiple vibrational modes at a single frequency. This limitation is imposed by the Fourier integral which associates a single amplitude with a single frequency. The possibility of transforming a time domain signal to the frequency domain with multiple amplitudes at a single frequency would require a focused investigation.

7 References

1. Hopkinson, B. "A Method of Measuring the Pressure Produced in the Detonation of High Explosives or by the Impact of Bullets", *Phil. Trans. Roy. Soc. London*, Vol. 213, pp. 437-456, 1914.
2. Davies, R.M. "A Critical Study of the Hopkinson Pressure Bar", *Phil. Trans. Roy. Soc. London*, Vol. 240, pp. 375-457, 1948.
3. Kolsky, H. "An Investigation of the Mechanical Properties of Materials at Very High Rates of Load", *Proc. Roy. Soc. London*, Vol. B62, pp. 676-700, 1949.
4. Kolsky, H. "*Stress Waves in Solids*", New York Dover Publications, 1963.
5. Pochammer, L. "On the Propagation Velocities of Small Oscillations in an Unlimited Isotropic Circular Cylinder" *J. f. Math. (Crelle)* Vol. 81, pp 324-326, 1876.
6. Chree, C. "The Equations of an Isotropic Elastic Solid in Polar and Cylindrical Coordinate, Their Solutions and Applications", *Cambridge Philosophical Society Transactions*, Vol. 14, pp 250-369, 1889.
7. Bancroft, D. "The Velocity of Longitudinal Waves in Cylindrical Bars", *Physical Review* Vol. 59, 1941
8. Gorham, D.A., "A Numerical Method for the Correction of Dispersion in Pressure Bar Signals", *J. Phys. E. Sci. Instum.*, Vol.16, pp. 477-479, 1983.
9. Follansbee, P.S. and Frantz, C, "Wave Propagation in the Split Hopkinson Pressure Bar", *Journal of Engineering Materials and Technology*, Vol. 105, pp. 61-66, 1983.
10. Marais, S.T. *Development and Proving of a Split Hopkinson Pressure Bar used for High Strain Rate Materials Testing*, M.Sc Dissertation, University of Cape Town, 2001
11. Cloete, T.J., Nurick, G.N. and Palmer, R.N "The deformation and shear failure in peripherally clamped centrally supported blast loaded circular plates." *Int. J. Impact Eng.*, Article in Press, 2005.
12. Gray, G.T. (Rusty) III, "Classic Split-Hopkinson Pressure Bar Technique", *ASM Vol. 8 – Mechanical Testing*, 1999.

13. Lennon, A.M. and Ramesh, K.T. "A Technique For Measuring The Dynamic Behaviour Of Materials At High Temperatures", *International Journal of Plasticity*, Vol. 14, pp 1279-1292, 1998.

14. Achenbach, J.D. *Wave Propagation In Elastic Solids*, North-Holland Publishing Company, 1973.

15. Morrison, N. "*Introduction To Fourier Analysis*", John Wiley & Sons, 1994

16. Gong, J.C., Malvern, L.E., Jenkins, D.A., "Dispersion Investigation in the Split Hopkinson Pressure Bar", *Journal of Engineering Materials and Technology*, Vol. 112, pp. 309-314, 1990

17. Bertholf, L.D. "Numerical Solution for Two-Dimensional Elastic Wave Propagation in Finite Bars", *Transactions of the ASME- Journal of Applied Mechanics*, Vol. 34, pp. 725-734, 1967

18. Miklowitz, J., and Niesewanger C.R. "The Propagation of Compressional Waves in a Dispersive Elastic Rod", *Journal of Applied Mechanics*, Vol. 24, Trans. ASME, Vol. 79, pp 240-244, 1957

19. Bertholf, L.D. and Karnes, C.H. "Two Dimensional Analysis Of The Split Hopkinson Pressure Bar System", *J. Mech. Phys. Solids*, Vol. 23, pp. 1-19, 1975

20. Tyas, A. and Watson, A.J. "An Investigation Of Frequency Domain Dispersion Correction Of Pressure Bar Signals", *International Journal of Impact Engineering*, Vol. 25, pp. 87-101, 2001

21. Ramirez, H. and Rubio-Gonzalez, C., "Finite-element simulation of wave propagation and dispersion in Hopkinson bar test", *Materials and Design*, Correct Proof in Press, 2004

22. Love, A.E.H., *A Treatise On The Mathematical Theory Of Elasticity*, New York Dover Publications, 1944. {Originally Cambridge University Press, 1927}

23. Thomson, W.T. "*Theory Of Vibration With Applications*", Chapman & Hall, 1993

24. Rao, S.S. "*Mechanical Vibrations, 2nd Edition*" Addison-Wesley Publishing Company, 1990

25. Daykin, L. "*Numerical Modelling of the Split Hopkinson Pressure Bar Experiment*", Project Report for Post Graduate Course MEC551Z, University of Cape Town, 2001

26. Balden, V. "*MEC564Z Introduction to Finite Elements Course Notes*" , University of Cape Town, 2004

27. Cook, R.D. "*Concepts and Applications of Finite Element Analysis*", 2nd Ed. John Wiley and Sons, 1981.

28. Lee, C.K.B., Crawford, R.C., Mann, K.A., Coleman, P. and Petersen, C., "Evidence of higher Pochammer-Chree modes in an unsplit Hopkinson Bar", *Meas. Sci. Technol.*, Vol. 6, pp. 853-859, 1995

29. Borse, G.J., "*Numerical Methods With Matlab*", PWS Publishing Company, 1997

30. Abaqus v6.5.1 *Documentation & User's Manual* , www.abaqus.com, 2005 (updated September 2005)

31. Bathe, K-L., "*Finite Element Procedures In Engineering Analysis*", Prentice-Hall, pp 540-547,1982

32. Merle, R. and Zhao, H. "On the errors associated with the use of a large diameter SHPB, correction for radially non-uniform distribution of stress and particle velocity in SHPB testing, *International Journal of Impact Engineering*, Vol. 31 (In Press), 2005

Appendix A Computer Code for Solving Pochammer-Chree Dispersion Equation

A.1 Generation of multi-mode phase velocity maps for dispersion correction

The following code is used to generate the phase velocity-frequency by solving the Pochammer-Chree dispersion equation in Matlab. The functions and sub-routines written specifically for use by this code are contained in the subsequent sections.

```
%Finding all  mode roots as function of wavelength
%
clc
clear all
User_variables

%Defining global variables
global a Elmod density poisson wavelength acc_ref MaxSteps
%a=bar outer radius Elmod =Elastic Modulus

%calculated constants
Vo=sqrt(Elmod/density);
 %Fundamental wave velocity
beta=(1-2*poisson)/(1-poisson);

length(wavelength);

disp('What is the desired accuracy of the roots?')
acc_ref=input('Reference accuracy=');

disp('How many iterations should be performed?')
MaxSteps=input('Maximum iterations=');

disp('Maximum Roots=?')
```

```
MaxRoots=input('MaxRoots=?')

Freq_Cut_Off=1e+6;
StartTime=clock;
Begin=clock;

%allocating array for refined values of roots
X_roots=zeros(length(wavelength),MaxRoots);

%give initial values for interval to search on
Smin=0.4;
Smax=4;
   %Use interval search method over first few wavelength values to find
   %refined values of roots for those wavelengths
   %Then use value of root for preceding value of wavelength as first
   %guess for current value of wavelength

   for k=1:3     %A

      %choose value of wavelength
      L=wavelength(k);

      %allocating local array for intervals containing roots of a given frequency
      Freq_Roots=zeros(MaxRoots,2);

      %As L decreases the number of roots on the interval increases
      %and the separation of the roots decreases, so a shorter interval is needed
      %to find all the root containing intervals
      S=[Smin:(L/5):Smax]';    %Defining range of S

      %S is local variable used to represent X

      A=[S(1:(length(S)-1)) S(2:length(S))];
      %Each row of A represents an interval of S

      B=PochChreeVec(A,poisson,L,a);
```

```
%Find function value at each array value

C=B(:,1).*B(:,2)<0;
%Establishing which intervals contain a zero crossing

H=abs(B(:,1).*B(:,2))<25;
%Establishing if crossing is a root or asymptote

IJ=C.*H; %+N;
D=[IJ IJ];
E=A.*D;
F=E(E>0);
G=[F(1:length(F)/2) F(length(F)/2+1:length(F))];
%Array G contains all intervals containing a zero crossing (either
%root or asymptote

 if size(G,1)>MaxRoots      %Z
   Freq_Roots=G(1:MaxRoots,:);
else
   Freq_Roots(1:size(G,1),:)=G(:,:);   %Z
 end                        %Z
%Ignoring higher roots than required

   for p=1:MaxRoots    %E
       if Freq_Roots(p,2)>0    %F
         %Root refining using interval halving

%X_roots(k,p)=PochChreeRootFind(Freq_Roots(p,1),Freq_Roots(p,2),L,a,poisson,a
cc_ref,MaxSteps);

         %Root refining using Newton-Raphson

X_roots(k,p)=PochChreeNewtRaph(Freq_Roots(p,1),Freq_Roots(p,2),L,a,poisson,ac
c_ref,MaxSteps,k,p);
       end    %F
   end    %E
```

```
  end %A

  for p=1:MaxRoots    %P  %Finding all values for a mode by looping over
     %wavelength then repeating for the next mode

     for k=4:length(wavelength)  %K
        tic;
        L=wavelength(k);
        RootGuess=2*X_roots(k-1,p)-X_roots(k-2,p);
        %Linear extrapolation to give good guess for next value of root
        RootGuess2=RootGuess+L/5;

        %Refine root using Newton Raphscon

X_roots(k,p)=PochChreeNewtRaph(RootGuess,RootGuess2,L,a,poisson,acc_ref,Ma
xSteps,k,p);

        if abs(X_roots(k,p)) >= 4
           disp('Exiting mode  ')
           p
           disp(' at d/L =')
           2*a/L
           break
           %if abs(X_roots(k,p)) >= 2.5 implies that the phase
           %velocity will be 2.5 times greater than the fundamental
           %velocity - more likely that the root is complex and the
           %the phase velocity is becoming asymptotic to some value of
           %wavelength
        end

        %Refine Root using interval bisection

%X_roots(k,p)=PochChreeRootFind(RootGuess,RootGuess2,L,a,poisson,acc_ref,M
axSteps);
```

```
        Time(k,2)=toc;

    end %K
  end %P

EndTime=etime(clock,StartTime)

scaled_wavelength=Diameter./wavelength;
scaled_wavelength_1=scaled_wavelength(1:length(scaled_wavelength)-1);
scaled_wavelength_inv=wavelength./Diameter;
scaled_veloc=sqrt(X_roots./(1+poisson));

Wavelen_plot=plot(scaled_wavelength,scaled_veloc,'-..');
        xlabel('Diameter/Wavelength');
        ylabel('Phase velocity / Fundamental Velocity');
        text(5, 0.3, ['Poisson=',num2str(poisson)]);
        grid on;

save('scaled_wavelength.mat','scaled_wavelength');
save('scaled_veloc.mat','scaled_veloc');

%Allocate and initialise arrays
  X = zeros(length(wavelength),MaxRoots);
  scaled_phase_veloc = zeros((length(scaled_veloc)-1),(2*MaxRoots));

  inv_scaled_wavelength = zeros((length(scaled_veloc)-1),(2*MaxRoots));

  Dphase_velocDinv_scaled_wavelength=zeros((length(scaled_veloc)-
1),MaxRoots);

  scaled_group_veloc = zeros(length(scaled_phase_veloc),MaxRoots);

tic;

for p=1:MaxRoots    %H
```

```
   Scaled_freq(:,p) = scaled_wavelength.*scaled_veloc(:,p);

   scaled_phase_veloc(:,(2*p-1):(2*p)) = [scaled_veloc(1:(length(scaled_veloc)-1),p)
scaled_veloc((2:length(scaled_veloc)),p)];

   inv_scaled_wavelength(:,(2*p-1):(2*p))=
scaled_wavelength_inv(1:(length(scaled_veloc)-1))
scaled_wavelength_inv(2:length(scaled_veloc))];

   Dphase_velocDinv_scaled_wavelength(:,p) = (scaled_phase_veloc(:,2*p)-
scaled_phase_veloc(:,2*p-1))./(inv_scaled_wavelength(:,2*p) -
inv_scaled_wavelength(:,2*p-1));

   scaled_group_veloc(:,p) = scaled_phase_veloc(:,2*p-1) -
inv_scaled_wavelength(:,2*p-1).*Dphase_velocDinv_scaled_wavelength(:,p);

end      %H

Group_plot_time=toc

Frequency=Scaled_freq.*Vo./(2*a);
PhaseVelocity=scaled_veloc.*Vo;

save('PC_Frequency.mat','Frequency')
save('PC_PhaseVelocity.mat','PhaseVelocity')

%Eliminate any entries is scaled_group_veloc that are outside of limits due
%to numerical errors
AZ=scaled_group_veloc<=1;
AY=scaled_group_veloc>=0;
scaled_group_veloc=scaled_group_veloc.*AZ.*AY;

%Create a vector X_1 which is the same length as scaled_group_veloc
Scaled_freq_1=Scaled_freq(1:length(Scaled_freq)-1,:);

MaxFreq=real(max(max(Scaled_freq_1)));  %Obtain maximum scaled frequency
of interest
```

```
Scaled_freq_1=real(Scaled_freq_1);  %

% VariableName_int implies that the array has been generated by
% or is used in interpolating from other arrays
Scaled_freq_int=[0:MaxFreq/2000:MaxFreq]';

for r=1:MaxRoots    %J

   Dom_Freq_i=Scaled_freq_1(:,r);  %Extract scaled_freq values

   Dom_Freq=Dom_Freq_i(Dom_Freq_i>0);  %Shrink vector to non-zero values

   Rng_Phs_Veloc_i=scaled_veloc(:,r);  %Extracting phase velocities
   Rng_Phs_Veloc=Rng_Phs_Veloc_i(Rng_Phs_Veloc_i>0);

   len=length(Rng_Phs_Veloc);
   if length(Rng_Phs_Veloc)<length(Dom_Freq)
        Rng_Phs_Veloc(len+1:length(Dom_Freq))=0;
   elseif length(Rng_Phs_Veloc)>length(Dom_Freq)
      Rng_Phs_Veloc(length(Dom_Freq)+1:len)=[];
      %Forcing Rng_Phs_Veloc to be the same length as Dom_Freq
   end

   Rng_Grp_Vel_i=scaled_group_veloc(:,r);  %Extract group velocity values
   Rng_Grp_Vel=Rng_Grp_Vel_i(Rng_Grp_Vel_i>0);

   if length(Rng_Grp_Vel)<length(Dom_Freq) %K
      len=length(Rng_Grp_Vel);
      Rng_Grp_Vel(len+1:length(Dom_Freq))=0;
   elseif length(Rng_Grp_Vel)>length(Dom_Freq)
      Rng_Grp_Vel(length(Dom_Freq)+1:len)=[];
      %Force Rng_Grp_Vel to be the same size as Dom_Freq by replicating
      %its last element
   end    %K

Phase_Veloc_int(:,r)=interp1(Dom_Freq,Rng_Phs_Veloc,Scaled_freq_int,'pchip');
```

```
   %Interpolate phase velocity values to obtain against a standard domain

Group_veloc_int(:,r)=interp1(Dom_Freq,Rng_Grp_Vel,Scaled_freq_int,'pchip','extrap'
);
      %Interpolate Grp_vel values to obtain grp_vel against a standard
      %domain

end %J

%Remove spurious values of Group_veloc_int caused by interpolation method
AA=Group_veloc_int<=1;
AB=Group_veloc_int>=0;
Group_veloc_int=Group_veloc_int.*AA.*AB;

AD=zeros(length(Phase_Veloc_int),MaxRoots);
AE=zeros(length(Phase_Veloc_int),MaxRoots);
AF=zeros(length(Phase_Veloc_int),MaxRoots);
AG=zeros(length(Phase_Veloc_int));
%initialise logical arrays

AG=max(Group_veloc_int,[],2);   %Find max grp velocity for any freq

AD=repmat(AG,1,MaxRoots);   %Replicate AG for comparison with each column
of AE

AE=Group_veloc_int>=AD; %logical array giving column of max grp vel
%which corresponds to mode number of dominant group

Scaled_Phase_Veloc_Map=AE.*Phase_Veloc_int;

Phase_Veloc_Map=Vo*Scaled_Phase_Veloc_Map;

Freq_Map=Vo*Scaled_freq_int/(2*a);


figure;
```

```
        hold on;
        Freq_Int_plot1=plot(Scaled_freq_int,Group_veloc_int,'.','MarkerSize',6);
        xlabel('Scaled Frequency (d.F/Co)');
        ylabel('Group velocity / Fundamental Velocity (Cg/Co)');
        text(5, 0.3, ['Poisson=',num2str(poisson)]);
        title('Scaled Group Velocities');
        %Freq_Int_plot2=plot(Scaled_freq_1,scaled_group_veloc,'-','LineWidth',1);
        grid on;
        hold off;

figure;
        Scaled_Vel_Map=plot(Scaled_freq_int,Scaled_Phase_Veloc_Map)
        xlabel('Scaled Frequency (d.F/Co)');
        ylabel('Phase velocity / Fundamental Velocity (Cp/Co)');
        text(5, 0.3, ['Poisson=',num2str(poisson)]);
        grid on;

figure;
        Vel_Map=plot(Freq_Map,Phase_Veloc_Map);
        xlabel('Frequency (Hz)');
        ylabel('Phase velocity (m/s)');
        text(200e+3, 2500, ['Poisson=',num2str(poisson)]);
        grid on;

Fn=repmat(Freq_Map,1,MaxRoots);

Cn1=Phase_Veloc_Map.*(Fn<=Freq_Cut_Off);
Fn1=Fn.*(Fn<=Freq_Cut_Off);
%Truncate Frequency and Phase Velocity vectors to values less than
%frequency cut off value

Cn2=Cn1(Cn1>0);
%Remove all zero elements of Phase Velocity

Fn2=Fn1(Cn1>0);
%Remove all corresponding elements of Frequency vector
```

```
figure;
Vel_Map3=plot(Fn2,Cn2);
xlabel('Frequency (Hz)');
ylabel('Phase velocity (m/s)');
text(200e+3, 2500, ['Poisson=',num2str(poisson)]);
grid on;

Finish=etime(clock,Begin)

disp('File name to save phase velocity to?')
VelocFile=input('Velocity file name=?','s')

disp('File name to save frequencies  to?')
FreqFile=input('Frequency file name=?','s')

save (VelocFile,'Cn2');
save (FreqFile,'Fn2');

%Generating PC Velocity Maps for comparison with Abaqus data
PC_Veloc=Phase_Veloc_int.*(Phase_Veloc_int>0);
PC_Veloc1=PC_Veloc.*(PC_Veloc<=2);

%plot(Freq_Map,PC_Veloc1)
PC_Veloc1(:,3)=PC_Veloc1(:,3).*(Freq_Map>=2e+5);
PC_Veloc1(:,4)=PC_Veloc1(:,4).*(Freq_Map>=3.6e+5);
PC_Veloc1(:,5)=PC_Veloc1(:,5).*(Freq_Map>=5e+5);
PC_Veloc1(:,6)=PC_Veloc1(:,6).*(Freq_Map>=6e+5);

PC_Veloc=Vo.*real(PC_Veloc1);
PC_Freq=Freq_Map;

plot(PC_Freq,PC_Veloc)
save('PC_Veloc.mat','PC_Veloc');
save('PC_Freq.mat','PC_Freq');
```

A.1.1 User Variables Input

This code is called to obtain the user inputs.

```
%Inputting global variables for use by all files
clear

global a Elmod density poisson wavelength
%a=bar outer radius Elmod =Elastic Modulus

disp('The following values will be used for the given variables in all further calculations.')

disp('Radius units - meters')
a=input('Bar outer radius=?'  )

disp('Elastic Modulus units - Pascals')
Elmod=input('Elastic Modulus=?'  )

disp('Density units - kg/m^3')
density=input('Density=?'  )

poisson=input('Poissons Ratio=?'  )

disp('Wavelength will be defined over a range.')
disp('Enter the lower and upper limits of wavelength (m)')

L_lo=input('shortest wavelength =?  ')
L_hi=input('longest wavelength=?  ')

%equally spaced vector begin
%wavelength=[L_lo:((L_hi-L_lo)/1000):L_hi]';
%equally spaced vector end

%unequally spaced vector begin
   %initialise index counter
%    n=1;
```

```
%   wavelength(1)=L_lo;
%
%    while wavelength(n)<L_hi
%        wavelength(n+1)=wavelength(n)*1.05;
%        n=n+1;
%    end
%
%    N=length(wavelength)
%
%    wavelength(N)=L_hi;
%unequally spaced vector end

%biased vector begins
    R=1.004;
    N=fix((log(L_hi/L_lo)/log(R)+1))
    wavelength=zeros(N,1);
    for n=1:N
        wavelength(n)=L_lo*(R^(n-1));
    end

%biased vector ends
Diameter=repmat(2*a,length(wavelength),1);
```

A.1.2 Dispersion Equation

Generating the dispersion maps requires that the magnitude of the dispersion equation 3-1 and its slope at a point be found repeatedly for differing values of x. Hence a separate function was written to prevent repetition of the lines of code.

```
function y=PochChreeVec(X,poisson,WaveLength,BarRadius)
%using Capital X in function instead of lower case x in attempt to have
%function defined as scalar function rather than array function
beta=(1-2*poisson)/(1-poisson);

gamma=2*pi/WaveLength;

h=gamma.*sqrt(beta.*X-1);

k=gamma.*sqrt(2.*X-1);

phi_ha=BarRadius.*h.*besselj(0,BarRadius.*h)./besselj(1,BarRadius.*h);

phi_ka=BarRadius.*k.*besselj(0,BarRadius.*k)./besselj(1,BarRadius.*k);

y=(X-1).*(X-1).*phi_ha-(beta.*X-1).*(X-phi_ka);

%Dedicated function to find derivate of dispersion function
function
 y= DiffPochChree(X,poisson,WaveLength,BarRadius)
dX=1e-6;
X2=X+dX;

F1=PochChreeVec(X,poisson,WaveLength,BarRadius);
F2=PochChreeVec(X2,poisson,WaveLength,BarRadius);
y=(F2-F1)./dX;
```

A.1.3 Modified Newton-Raphscon Root Solver

The following code refined an initial guess for the root of the dispersion equation using a modified Newton-Raphscon algorithm. The modifications are necessary for refining "touching roots" as the ordinary algorithm is sufficient for finding "crossing roots".

```
function Xr=PochChreeNewtRaph(Sa,Sb,L,a,poisson,acc_ref,MaxSteps,Wave,Mode)
%dedicated function for returning root of Pochammer_Chree using
%Newton-Raphson method
%Eqn in a given interval for a given accuracy

StartTime=clock;
S_init=Sa;
step=0;  %initialise iteration counter
accuracy = 1;  %initialise accuracy

%Entering root finding loop
while (step<=MaxSteps) & (accuracy>acc_ref) %B

   step = step+1;

   if (step>MaxSteps)  %exceeding maximum iterations
     disp('Maximum iterations reached')
      disp('Last value of X obtained:')
      So
      Xr=PochChreeRootFind(S_init,Sb,L,a,poisson,acc_ref,MaxSteps)
   end

   FSa=PochChree(Sa,poisson,L,a);

   dFSa=DiffPochChree(Sa,poisson,L,a);

  %Establish if function is approaching a touching root
   %if (Mode==6)  %Mode 6 displaying off behaviour
    %   if (abs(FSa)<=5e-1) | (abs(dFSa)<=0.75) % | =OR
```

```
    %  Xr=Sa;
    %  break
    %  end
   %end

   So=Sa-(FSa/dFSa);   %Next value of root

   FSo=PochChree(So,poisson,L,a);

   if abs(FSo)<acc_ref %A %Function value is approaching zero
      %format short
      %EndTime=etime(clock,StartTime)
      %step
       Xr=So;
      %FSo

      break
    else
      Sa=So; %USe current value of root as guess
   end   %A

end %B
```

A.2 Modified Dispersion Correction Code

This appendix details the code by written by the author to implement dispersion correction for more than 4 Pochammer-Chree modes. The code is based on the method proposed by Gorham [8] and Follansbee and Frantz [9] and draws largely from the code originally written by Marais [10].

```
%Correct signal for dispersion
%either with pre-existing correction map
% or create map as necessary
clc;
clear all;

dt=1/(10e+6);   %seconds
dZ=0.5;                 %meters
StartTime=1e-15;

disp('Do you wish to perform dispersion correction')
disp('using (1) An existing C_p-F map')
disp(' or (2) Create a new C_p-F map?');

Choice=input('Option=?');

if Choice==1    %A
   disp('Please enter the name of the phase velocity file');
   Vel_file=input('File name=?','s')
   disp('Please enter the name of the frequency file');
   Frequency_file=input('File name=?','s')

   C_PC=load('-ascii',Vel_file);
   F_PC=load('-ascii',Frequency_file);

   disp('The following values will be used for the given variables in all further
calculations.')

   disp('Radius units - meters')
   a=input('Bar outer radius=?'   )
```

```
   disp('Elastic Modulus units - Pascals')
   Elmod=input('Elastic Modulus=?'   )

   disp('Density units - kg/m^3')
   density=input('Density=?'   )

   poisson=input('Poissons Ratio=?'   )

elseif Choice==2
  disp('Warning- the solver may take up to 5 minutes to complete');
   PC_Disp_Solve;

end     %A

%Fundamental velocity
Vo=sqrt(Elmod/density);

%Load input signal
disp('Please enter the file name containing the data to be corrected')
FileName=input('File name = ?','s')
XY1=load('-ascii', FileName);
x=XY1(:,2);

%Clear unnecessary variables
clear XY1;

%Zero signal

%Generate time and frequency vectors
N=length(x);
Time=[StartTime:dt:((N-1)*dt+StartTime)].';
Period=dt.*N;
Fo=1/Period;
Frequency=Fo.*[fix(-N/2):1:fix(N/2)].';
AngFreq=2*pi.*Frequency;
```

```
%Interpolate phase velocity for correction vectors
    %positive side of Frequency vector
    QQ=length(Frequency);
    RR=fix(QQ/2);

    %last value of phase velocity
    Cend=C_PC(length(C_PC));

    Cp_int1=interp1(F_PC,C_PC,Frequency(RR:QQ),'cubic',Cend);

    %initialise phase velocity correction vector
    Cp_corr=repmat(Cend,QQ,1);

    %negative side of frequency vector
    Cp_int2=zeros(RR,1);

    for n=1:(RR-1); %A
       Cp_int2(n)=Cp_int1(RR-n+1);
    end %A

    Cp_corr(1:(RR))=Cp_int2;
    Cp_corr((RR+1):QQ)=Cp_int1(1:(RR+1));
    Cp_corr((RR-1):1:(RR+1))=repmat(Vo,3,1);
%phase velocity for correction vector complete

%Correct for dispersion
Yo=fft(x);
Ys=fftshift(Yo);
PhaseShift=dZ./Cp_corr;
Ys_corr=Ys.*exp(-1.*j.*AngFreq.*PhaseShift);
X_c=ifft(ifftshift(Ys_corr));
X_corr=real(X_c);

DataOut=[Time X_corr];
save ('CorrSig.dat','DataOut','-ascii','-double','-tabs');
disp('The corrected signal has been saved to the file "CorrSig.dat" ')
```

```
disp('Please re-name the file now to prevent it being over-written')

figure;
    plot(F_PC./(1e+3),C_PC);
    grid on;
    set(gca,'FontName','Arial','FontSize',12)
    xlabel('Frequency  (kHz)');
    ylabel('Phase Velocity C_p (m/s)');
    title('Current Dispersion Correction Map','FontSize',14);

figure;
    plot(Frequency./(1e+3),Cp_corr);
    set(gca,'FontName','Arial','FontSize',12)
     xlabel('Frequency  (kHz)');
    ylabel('Phase Velocity C_p (m/s)');
    title('Full Dispersion Correction Map','FontSize',14);

Y_max=max(Ys);
figure;
    plot(Frequency./(1e+3),abs(Ys)./Y_max);
    set(gca,'FontName','Arial','FontSize',12)
     xlabel('Frequency  (kHz)');
    ylabel('Amplitude');
    title('Scaled Amplitude Frequency Spectrum','FontSize',14);
    axis([0 1e+3 -Inf Inf])

figure;
    hold on;
    plot(Time./(1e-6),x,'b');
    plot(Time./(1e-6),X_corr,'m');
    set(gca,'FontName','Arial','FontSize',12)
     xlabel('Time (\mus)');
    ylabel('Amplitude');
    title('Original and dispersed signals','FontSize',14);
    legend({'Original signal','Corrected for dispersion'})
```

Appendix B Modal Analysis of Infinite and Finite Cylindrical Bars using Abaqus

B.1 Mode Extraction - Typical Input Data Deck

The following is an example of an input data deck used to extract the eigen values and modes from a model of an infinite bar. Sections of the nodal coordinate and element connectivity tables have been removed due to space constraints.

```
*Heading
** Job name: Job-200mm20r400a Model name: Model-1
*Preprint, echo=NO, model=NO, history=NO, contact=NO
**
** PARTS
**
*Part, name=Part-Bar
*End Part
**
**
** ASSEMBLY
**
*Assembly, name=Assembly
**
*Instance, name=Part-Bar-1, part=Part-Bar
*Node
      1,          0.,          0.
      2, 0.000500000024,          0.
      3, 0.00100000005,          0.
      4, 0.0014999999,          0.
***Nodal coordinate data removed due to space constraints***
   8419, 0.00899999961,  0.200000003
   8420, 0.00949999969,  0.200000003
   8421, 0.00999999978,  0.200000003
*Element, type=CAX4
```

```
  1,   1,   2,  23,  22
  2,   2,   3,  24,  23
  3,   3,   4,  25,  24
***Element connectivity data removed due to space constraints ***
7998, 8397, 8398, 8419, 8418
7999, 8398, 8399, 8420, 8419
8000, 8399, 8400, 8421, 8420
*Nset, nset=Set-All, generate
   1,  8421,    1
*Elset, elset=Set-All, generate
   1,  8000,    1
** Region: (Section-All:Set-All)
** Section: Section-All
*Solid Section, elset=Set-All, material=Material-Steel
1.,
*End Instance
**
*Nset, nset=_PickedSet4, internal, instance=Part-Bar-1, generate
   1,  8401,   21
*Elset, elset=_PickedSet4, internal, instance=Part-Bar-1, generate
   1,  7981,   20
*Nset, nset=_PickedSet5, internal, instance=Part-Bar-1
   1,   2,   3,   4,   5,   6,   7,   8,   9,  10,  11,  12,  13,  14,  15,  16
  17,  18,  19,  20,  21, 8401, 8402, 8403, 8404, 8405, 8406, 8407, 8408, 8409,
8410, 8411
 8412, 8413, 8414, 8415, 8416, 8417, 8418, 8419, 8420, 8421
*Elset, elset=_PickedSet5, internal, instance=Part-Bar-1
   1,   2,   3,   4,   5,   6,   7,   8,   9,  10,  11,  12,  13,  14,  15,  16
  17,  18,  19,  20, 7981, 7982, 7983, 7984, 7985, 7986, 7987, 7988, 7989, 7990,
7991, 7992
 7993, 7994, 7995, 7996, 7997, 7998, 7999, 8000
*End Assembly
**
** MATERIALS
**
*Material, name=Material-Steel
*Density
```

```
7830.,
*Elastic
 2.07e+11, 0.26
**
** BOUNDARY CONDITIONS
**
** Name: BC-CentreAxis Type: Symmetry/Antisymmetry/Encastre
*Boundary
_PickedSet4, XSYMM
** Name: BC-Ends Type: Symmetry/Antisymmetry/Encastre
*Boundary
_PickedSet5, YSYMM
** ----------------------------------------------------------------
**
** STEP: Step-EigenVals
**
*Step, name=Step-EigenVals, perturbation
*Frequency, eigensolver=Lanczos, acoustic coupling=on,
normalization=displacement, number interval=1, bias=1.
, 100., 1e+06, , ,
**
** OUTPUT REQUESTS
**
*Restart, write, frequency=0
**
** FIELD OUTPUT: F-Output-1
**
*Output, field, variable=PRESELECT
*End Step
```

B.2 Extraction of Axial Displacement Along Outer Radius

Due to the large number of modes, it was necessary to record a macro within Abaqus to extract the axial displacement data. The steps performed in Abaqus are automatically written as Python™ Script commands to a file, designated "abaqusmacros.py". The file is then edited to permit looping over the modes. The author wishes to thank Victor Balden for his assistance with the Python™ Script.

```
# Do not delete the following import lines
from abaqus import *
from abaqusConstants import *

#Macro record begins
def MacroGetWave():
    import part
    import regionToolset
    import displayGroupMdbToolset as dgm
    import material
    import section
    import assembly
    import step
    import interaction
    import load
    import mesh
    import job
    import sketch
    import visualization
    import xyPlot
    import displayGroupOdbToolset as dgo
    o1 = session.openOdb(
        name='E:/Reuben/Axisymmetric Bar/200mm20r400a/200mm20r400a.odb',
        readOnly=FALSE)
    session.viewports['Viewport: 1'].setValues(displayedObject=o1)
    session.viewports['Viewport: 1'].view.setValues(nearPlane=0.45748,
        farPlane=0.67739, width=0.0274935, height=0.0200022, cameraPosition=(
        0.0104092, 0.00879771, 0.566392), cameraTarget=(0.0104092, 0.00879771,
```

```
        0))
    session.Path(name='Path-OuterRadius', type=NODE_LIST, expression=((
        'PART-BAR-1', ('21:8021:21',), )),
        ))
    session.viewports['Viewport: 1'].odbDisplay.setPlotMode(CONTOUR)
    session.viewports['Viewport: 1'].odbDisplay.contourOptions.setValues(
        deformationScaling=UNIFORM, uniformScaleFactor=1e-4, maxValue=1,
        minValue=0)
#Edited code for looping over modes
for mode in range(1,885):
    session.viewports['Viewport: 1'].odbDisplay.setFrame(step=0, frame=mode)
    session.viewports['Viewport: 1'].odbDisplay.setDeformedVariable(
        variableLabel='U')
    session.viewports['Viewport: 1'].odbDisplay.setPrimaryVariable(
        variableLabel='U', outputPosition=NODAL, refinement=(COMPONENT, 'U2'))
    pth = session.paths['Path-OuterRadius']
    session.XYDataFromPath(name='XYData-'+str(mode), path=pth,
includeIntersections=FALSE,
        shape=UNDEFORMED, labelType=TRUE_DISTANCE)
    x0 = session.xyDataObjects['XYData-'+str(mode)]
    session.writeXYReport(fileName='200mm20r400aMode'+str(mode)+'.dat',
appendMode=OFF, xyData=(
        x0, ))
```

B.3 Determining Wavelength From Axial Displacement Data

The axial displacement profiles extracted from Abaqus were post-processed in Matlab to determine the wavelengths of the individual modes. The final method used to determine the wavelength of the mode used the Fourier transform, but in the spatial domain rather than the time domain. As the output of the FFT is a wave number, rather than a frequency as in the Fourier integral, the wavelength is easily found given the model bar length and the wave number.

```
%Read Abaqus report file containing axial displacement along outer radius
%and calculate the wavelength of the relevant mode
clc
clear all

%ModeNum=input('Mode Number=?')
tic;
Modes=891;
WaveNum=zeros(Modes,4);

format short g;

for ModeNum=1:Modes    %A
   %Concatenate Prefix, mode number and suffix to obtain string for filename
   FileName=['FinBar200_Mode',num2str(ModeNum),'.dat'];

   FileID=fopen(FileName); %open file
   XY_Data=textscan(FileID,' %f %f','headerlines',6);
   %textscan reads data to a cell array rather than an ordinary array
   fclose(FileID);

   %Convert data from a cell array to vectors
   X=XY_Data{1,1};
   Y=XY_Data{1,2};

   %The following code obtained the wavelength by counting peaks and troughs
   %but was discarded due to innaccuracies - it has been included for
```

```
   %completeness

   %Wavelength finding by peak and trough counting %H
%           Peak = repmat(max(Y),length(Y),1);
%          Trough=repmat(min(Y),length(Y),1);
%
%              %Count peaks by seeing which values of Y are within a tolerance of
max(Y)
%          Tol=1e-6;
%          AA=Y(abs(Y-Peak)<=Tol);
%
%          %Count troughs
%          AB=Y(abs(Y-Trough)<=Tol);
%
%          if min(Y)==0    %B
%             WaveNum(ModeNum,2)=(length(AA))/2;
%          else
%             WaveNum(ModeNum,2)=(length(AA)+length(AB))/2;
%          end              %B
%
%          WaveNum(ModeNum,1)=ModeNum;
%          end    %A
   %End Wavelength finding by peak and trough counting %H

 %The following code counted zero crossings to obtain wavelengths
 %It was robust and accurate for the infinte bar models but couldn't cope
 %with the finite bar modes

   %Wavelength finding by counting zero crossings %J
%          Y1=Y(1:length(Y)-1);
%          Y2=Y(2:length(Y));
%          AC=(Y1.*Y2<=0);
%          AD=AC(AC>0);
%
%          WaveNum(ModeNum,2)=(length(AD)-1)/2;
%          %Because first and last values of vector are always zero, it is
%          %necessary to remove these from the count
```

```
    %End Wavelength finding by counting zero crossings %J

    %Wavelength finding by Fourier methods  %G
        N=length(Y);
        %Fourier Coefficients of Y
        Y_F=fft(Y);
        %Centre fourier coefficients and normalise
        Y_F1=abs(fftshift(Y_F)./N);

        Coeff_num=[-fix(N/2):fix(N/2)]';
           %Coeff_num truncated because of how fft is found
        Coeff_Num1=Coeff_num(1:length(Coeff_num)-1);

        AA=Coeff_Num1.*(Y_F1>=0.5.*max(Y_F1));
           AB=AA(AA>0);

        if length(AB)==1    %H
           WaveNum(ModeNum,2)=AB;

        else
              for p=1:(length(AB)-1)  %I
                 if (AB(p+1)-AB(p))<=1   %J
                    WaveNum(ModeNum,(p+1))=mean(AB(p:p+1));
                 else
                    WaveNum(ModeNum,(p+1))=AB(p);
                 end    %J
              end    %I
        end    %H

    %End wavelength finding by Fourier  %G

    WaveNum(ModeNum,1)=ModeNum;

end    %A

RunTime=toc
```

```
format short g
WaveNum
Q=input('Mode Number=?')

   %Concatenate Prefix, mode number and suffix to obtain string for filename
   FileName=['FinBar200_Mode',num2str(Q),'.dat'];

   FileID=fopen(FileName); %open file
   XY_Data=textscan(FileID,' %f %f','headerlines',6);
   %textscan reads data to a cell array rather than an ordinary array
   fclose(FileID);

   %Convert data from a cell array to vectors
   X=XY_Data{1,1};
   Y=XY_Data{1,2};

   plot(X,Y)
   xlabel(' Undeformed Axial Position (m)','FontName','Arial', 'FontSize',14)
   ylabel('Axial Displacement (m)','FontName','Arial', 'FontSize',14)
   title('Axial Displacement Profile','FontName','Arial',
'FontSize',14,'FontWeight','demi')
   grid on

   WaveNum(Q,:)
BarLength=0.200;

load Freq200.dat
Freq=Freq200(1:Modes,2);
Freq1=repmat(Freq,1,(size(WaveNum,2)-1));
Wavelength=BarLength./WaveNum(:,2:size(WaveNum,2));
PhaseVeloc=Freq1.*Wavelength;
CalculationTime=toc

load('PC_PhaseVelocity.mat')
load('PC_Frequency.mat')

figure
```

```
hold on
        plot(Freq1(:,1)./1e+3,PhaseVeloc(:,1)./1e+3,'xr','MarkerSize',6)
        plot(Freq1(:,2)./1e+3,PhaseVeloc(:,2)./1e+3,'+g','MarkerSize',6)
        plot(Freq1(:,3)./1e+3,PhaseVeloc(:,3)./1e+3,'^c','MarkerSize',6)
        plot(Freq1(:,4)./1e+3,PhaseVeloc(:,4)./1e+3,'ok','MarkerSize',4)
        plot(Frequency./1e+3,PhaseVelocity./1e+3,'-b','LineWidth',1)
        axis([0 600 0 10])
        set(gca, 'FontName','Arial','FontSize',12)
        legend1 = legend({'Abaqus 1st Wavelength','Abaqus 2nd
        Wavelength','Abaqus 3rd Wavelength','Abaqus 4th
        Wavelength','Pochammer-Chree',}, 'FontName','Arial',...
         'FontSize',12)
        xlabel('Frequency (Hz)','FontName','Arial', 'FontSize',14)
        ylabel('Phase Velocity (m/s)','FontName','Arial', 'FontSize',14)
        title('Comparison of Pochammer-Chree and Finite Bar (200mm) predictions
        for phase velocity','FontName','Arial', 'FontSize',16)
hold off


save ('Freq0727_1.mat','Freq')
save('Wavelength0727_1.mat','Wavelength')
save('PhaseVeloc0727_1.mat','PhaseVeloc')
```

Appendix C FEA of dynamic loading of bars

C.1 Trapezoidal Pressure Load – Explicit Analysis Input Data Deck

```
*Heading
** Job name: Job-Trap_500_5_40_5 Model name: Model-1
*Preprint, echo=NO, model=NO, history=NO, contact=NO
**
** PARTS
**
*Part, name=Part-Bar
*End Part
**
**
** ASSEMBLY
**
*Assembly, name=Assembly
**
*Instance, name=Part-Bar-1, part=Part-Bar
*Node
     1, 0.00999999978,  0.100000001
     2,          0.,  0.100000001
***Nodal coordinate data removed due to space constraints ***
 10520, 0.00899999961,  0.499000013
 10521, 0.00949999969,  0.499000013
*Element, type=CAX4R
   1,    1,    7, 1060,  242
   2,    7,    8, 1061, 1060
***Element connectivity data removed due to space constraints ***
9999, 10520, 10521,  642,  643
10000, 10521,  641,    5,  642
*Nset, nset=Set-All, generate
    1, 10521,     1
*Elset, elset=Set-All, generate
   1, 10000,    1
** Region: (Section-Steel:Set-All), (Controls:EC-1)
```

```
** Section: Section-Steel
*Solid Section, elset=Set-All, controls=EC-1, material=Material-Steel
1.,
*End Instance
**
*Nset, nset=Set-CentreLine, instance=Part-Bar-1
    2,    3-**Datat removed for readability** 9961, 9981
*Elset, elset=_Surf-LoadEnd_S3, internal, instance=Part-Bar-1, generate
 1981,  2000,     1
*Elset, elset=Set-Gauge050, instance=Part-Bar-1
 1001,
*Elset, elset=Set-Gauge250, instance=Part-Bar-1
 5000,
*Elset, elset=Set-Gauge450, instance=Part-Bar-1
 9000,
*Surface, type=ELEMENT, name=Surf-LoadEnd
_Surf-LoadEnd_S3, S3
*Nset, nset=Set-NGauge250, elset=Set-Gauge250, instance=Part-Bar-1
*Nset, nset=Set-NGauge450, elset=Set-Gauge450, instance=Part-Bar-1
*End Assembly
**
** ELEMENT CONTROLS
**
*Section Controls, name=EC-1, hourglass=ENHANCED
1., 1., 1.
*Amplitude, name=Amp-Trapezoid
0., 0., 2.5e-06, 0.5, 5e-06, 1., 7.5e-06, 1.
1e-05, 1., 4e-05, 1., 4.5e-05, 1., 4.75e-05, 0.5
5e-05, 0., 5.25e-05, 0., 5.5e-05, 0., 6e-05, 0.
7e-05, 0.
**
** MATERIALS
**
*Material, name=Material-Steel
*Density
7830.,
*Elastic
```

```
 2.07e+11, 0.26
**
** BOUNDARY CONDITIONS
**
** Name: BC-1 Type: Displacement/Rotation
*Boundary
Set-CentreLine, 1, 1
** ----------------------------------------------------------------
**
** STEP: Step-LoadEx
**
*Step, name=Step-LoadEx
*Dynamic, Explicit, Element By Element, scale factor=0.4
, 0.00025
*Bulk Viscosity
0.06, 1.2
**
** LOADS
**
** Name: Load-EndPressure   Type: Pressure
*Dsload, amplitude=Amp-Trapezoid
Surf-LoadEnd, P, 1e+06
**
** OUTPUT REQUESTS
**
*Restart, write, number interval=1, time marks=NO
**
** FIELD OUTPUT: F-Output-1
**
*Output, field, variable=PRESELECT
**
** HISTORY OUTPUT: H-Output-1
**
*Output, history, time interval = 0.2e-6
*Element output, elset=Set-Gauge050
 E, LE, S
*Element output, elset=Set-Gauge250
```

```
 E, LE, S
*Element output, elset=Set-Gauge450
 E, LE, S
*Node output, nset=Set-NGauge250
 U
*Node output, nset=Set-NGauge450
 U
*End Step
```

C.2 Trapezoidal Pressure Load – Modal Summation Method Input Data Deck

```
*Heading
** Job name: Job-Trap_500_5_40_5MS Model name: Model-1
*Preprint, echo=NO, model=NO, history=NO, contact=NO
**
** PARTS
**
*Part, name=Part-Bar
*End Part
**
**
** ASSEMBLY
**
*Assembly, name=Assembly
**
*Instance, name=Part-Bar-1, part=Part-Bar
*Node
      1, 0.00999999978,  0.100000001
      2,          0.,  0.100000001
***Nodal coordinate data removed due to space constraints ***
  10520, 0.00899999961,  0.499000013
  10521, 0.00949999969,  0.499000013
*Element, type=CAX4
    1,     1,     7,  1060,   242
    2,     7,     8,  1061,  1060
***Element connectivity data removed due to space constraints ***
9999, 10520, 10521,   642,   643
10000, 10521,   641,     5,   642
*Nset, nset=Set-All, generate
     1,  10521,      1
*Elset, elset=Set-All, generate
    1, 10000,     1
** Region: (Section-Steel:Set-All)
** Section: Section-Steel
```

```
*Solid Section, elset=Set-All, material=Material-Steel
1.,
*End Instance
**
*Nset, nset=Set-CentreLine, instance=Part-Bar-1
   2,    3,    6. **Data removed due to space constraints**  1058, 1059
*Elset, elset=Set-CentreLine, instance=Part-Bar-1
  20,   40,   60, . **Data removed due to space constraints**  9961, 9981
*Elset, elset=_Surf-LoadEnd_S3, internal, instance=Part-Bar-1, generate
 1981,  2000,     1
*Elset, elset=Set-Gauge250, instance=Part-Bar-1
 5000,
*Surface, type=ELEMENT, name=Surf-LoadEnd
_Surf-LoadEnd_S3, S3
*End Assembly
*Amplitude, name=Amp-Trapezoid
0., 0., 2.5e-06, 0.5, 5e-06, 1., 7.5e-06, 1.
1e-05, 1., 4e-05, 1., 4.5e-05, 1., 4.75e-05, 0.5
5e-05, 0., 5.25e-05, 0., 5.5e-05, 0., 6e-05, 0.
7e-05, 0.
**
** MATERIALS
**
*Material, name=Material-Steel
*Density
7830.,
*Elastic
 2.07e+11, 0.26
**
** BOUNDARY CONDITIONS
**
** Name: BC-1 Type: Displacement/Rotation
*Boundary
Set-CentreLine, 1, 1
** ----------------------------------------------------------------
**
** STEP: Step-EigenVals
```

```
**
*Step, name=Step-EigenVals, perturbation
*Frequency, eigensolver=Lanczos, acoustic coupling=on,
normalization=displacement, number interval=1, bias=1.
, 100., 500000., , ,
**
** OUTPUT REQUESTS
**
*Restart, write, frequency=0
**
** FIELD OUTPUT: F-Output-1
**
*Output, field, variable=PRESELECT
*End Step
** ----------------------------------------------------------------
**
** STEP: Step-LoadMS
**
*Step, name=Step-LoadMS, perturbation
*Modal dynamic, continue=NO
0.2e-06, 0.00022
**
** LOADS
**
** Name: Load-EndPressure   Type: Pressure
*Dsload, amplitude=Amp-Trapezoid
Surf-LoadEnd, P, 1e+06
**
** OUTPUT REQUESTS
**
**
** FIELD OUTPUT: F-Output-2
**
*Output, field, variable=PRESELECT
**
** HISTORY OUTPUT: H-Output-1
**
```

```
*Output, history, frequency=1
*Element output, elset=Set-Gauge250
 E, LE
*End Step
```

Appendix D FEA of Oscillatory Loading of Bars

D.1 User Defined Load for Gauss Windowed Sine Wave

The use of a GWS as a loading function necessitated writing a "User Defined Function" (vdload) file in Fortran. The author wishes to thank Dean Bonorchis for his assistance with the coding.

```
      subroutine vdload(nblock,ndim,stepTime,totalTime,amplitude,
     1 curCoords,velocity,dirCos,jltyp,sname,value)
     include 'vaba_param.inc'
c
     dimension curCoords(nblock,ndim),velocity(nblock,ndim),
     1 dirCos(nblock,ndim,ndim),value(nblock)
     character*80 sname
c
     do km=1,nblock
        A = 1.0d+6
        f = 83.33333333d3
        t0 = 60.0d-6
        xk = 56.0d-6
c       pi = 3.14159d0
        pi = 2.0d0*asin(1.0d0)
        value(km) = A*sin(2*pi*f*stepTime)*exp(-((stepTime-t0)/xk)
c
 100  format(f16.4)
        return
        end
```

D.2 Typical Input Data Deck for GWS Loading

```
*Heading
** Job name: Job-TyasLoad2 Model name: Model-1
*Preprint, echo=NO, model=NO, history=NO, contact=NO
**
** PARTS
**
*Part, name=Part-Bar
*End Part
**
**
** ASSEMBLY
**
*Assembly, name=Assembly
**
*Instance, name=Part-Bar-1, part=Part-Bar
*Node
     1,          0.,          0.
     2, 0.000500000024,          0.
***Data removed due to space constraints***
125024,  0.011500001,          2.5
 125025, 0.0120000001,          2.5
*Element, type=CAX4R
    1,     1,     2,    27,    26
    2,     2,     3,    28,    27
***Data removed due to space constraints***
119999, 124998, 124999, 125024, 125023
120000, 124999, 125000, 125025, 125024
*Nset, nset=Set-All, generate
     1,  125025,       1
*Elset, elset=Set-All, generate
     1,  120000,       1
** Region: (Section-All:Set-All)
** Section: Section-All
*Solid Section, elset=Set-All, material=Material-Steel
```

```
1.,
*End Instance
**
*Elset, elset=_Surf-LoadEnd_S1, internal, instance=Part-Bar-1, generate
  1,  24,   1
*Surface, type=ELEMENT, name=Surf-LoadEnd
_Surf-LoadEnd_S1, S1
*Elset, elset=Set-Gauge100mm, instance= Part-Bar-1
 4800
*Elset, elset=Set-Gauge500mm, instance= Part-Bar-1
 24000
*Elset, elset=Set-Gauge500mm_2, instance= Part-Bar-1
 23989
*Elset, elset=Set-G1000mm, instance= Part-Bar-1
 48000
*Elset, elset=Set-G1000mm_2, instance= Part-Bar-1
 47989
*Elset, elset=Set-Face7, instance=Part-Bar-1
 7
*Nset, nset=Set-NGauge100mm, elset=Set-Gauge100mm, instance=Part-Bar-1
*Nset, nset=Set-NGauge500mm, elset=Set-Gauge500mm, instance=Part-Bar-1
*End Assembly
**
** MATERIALS
**
*Material, name=Material-Steel
*Density
8000.,
*Elastic
 2.0e+11, 0.29
** ----------------------------------------------------------------
**
** STEP: Step-Load_Ex
**
*Step, name=Step-Load_Ex
*Dynamic, Explicit, element by element
, 0.00075
```

```
*Bulk Viscosity
0.06, 1.2
**
** LOADS
**
** Name: Load-End_Press   Type: Pressure
*Dsload
Surf-LoadEnd, PNU, 1.
**
** OUTPUT REQUESTS
**
*Restart, write, number interval=1, time marks=NO
**
** FIELD OUTPUT: F-Output-1
**
*Output, field,  number interval = 250, variable=PRESELECT
**
** HISTORY OUTPUT: H-Output-1
**
*Output, history, time interval=20e-8
*Element output, elset=Set-Gauge100mm
 E, LE, S
*Element output, elset=Set-Gauge500mm
 E, LE, S
*Element output, elset=Set-Gauge500mm_2
 E, LE, S
*Element output, elset=Set-G1000mm
 E, LE, S
*Element output, elset=Set-G1000mm_2
 E, LE, S
*End Step
```