1-1-2002

Structural vibrations and internal ballistic modelling of a star-grain solid rocket motor

Sonny Loncaric
Ryerson University

In compliance with the
Canadian Privacy Legislation
some supporting forms
may have been removed from
this dissertation.

While these forms may be included
in the document page count,
their removal does not represent
any loss of content from the dissertation.

STRUCTURAL VIBRATIONS AND INTERNAL BALLISTIC MODELLING OF A STAR-GRAIN SOLID ROCKET MOTOR

by

Sonny Loncaric

A thesis
presented to Ryerson University
in partial fulfillment of the
requirement for the degree of
Master of Applied Science
in the program of
Mechanical Engineering

Toronto, Ontario, Canada, 2002

Your file Votre référence
ISBN: 0-612-87161-4
Our file Notre référence
ISBN: 0-612-87161-4

I hereby declare that I am the sole author of this thesis

I authorize Ryerson University to lend this thesis to other institutions or individuals for the purpose of scholarly research.

Sonny Loncaric

I further authorize Ryerson University to reproduce this thesis by photocopying or by other means, in total or in part, at the request of other institutions or individuals for the purpose of scholarly research.

Sonny Loncaric

Ryerson University requires the signature of all persons using or photocopying this thesis. Please sign below, and give address and date.

STRUCTURAL VIBRATIONS AND INTERNAL BALLISTIC MODELLING OF A STAR-GRAIN SOLID ROCKET MOTOR

MASc, 2003, Sonny Loncaric, Mechanical Engineering, Ryerson University.

Abstract

A numerical model is developed to solve the governing equations for the structural dynamics and internal ballistics of a solid rocket motor (SRM). An explicit finite element method is used to solve for the structural response, and an explicit finite volume method is used to solve for the internal ballistic flow. Together, these two numerical solutions are coupled to model the nonsteady behaviour of axial combustion instability in sleeved cylindrical- and star-grain SRMs.

The simulation model is used to predict the axial instability in star-grain SRMs. A parametric analysis is made to record the effects of various parameters on the simulation model. These parameters include the numerical dissipation constant, damping ratio and pulsing strength. It is found that both the numerical dissipation constant and damping ratio can, both artificially and physically, affect the finite element structural response of the motor. The pulsing strength affects only the rate at which the dc pressure rises as well as how quickly the limiting wave amplitude is reached.

The detailed analysis of simulated star-grain SRM axial instability reveals the effect of structural vibrations on burning rate augmentation and wave development in nonsteady operation. The variation in oscillation frequencies about a given grain section periphery, and along the grain with different levels of burnback, influences the means by which the local acceleration drives the combustion and flow behavior. The amount of damping also plays a role in influencing the predicted instability symptoms of the motor.

Acknowledgements

I would first like to thank Jerry Karpynczyk for his encouragement, and always keeping spirits up. I've learnt many things about rockets, experimental testing and manufacturing that weren't a part of my thesis, of which I'm glad I had that opportunity to learn in any case. I would like to thank Hamid Ghaemi for the many hours of discussion on structural mechanics and finite element analysis, which helped with the development of the simulation; and for time and discussion that did not relate to my thesis work, but was just as interesting. I would also like to thank Dr. Zouheir Fawaz, my thesis co-supervisor, for helping give me the opportunity to study for my master's degree at Ryerson, as well as providing useful input in the development of my simulation. I would also like to thank Meredith Swithers for her never-ending support and encouragement. She has helped me emotionally and spiritually, as well as providing useful criticism in the writing of this thesis. Finally, I would thank my supervisor Dr. David Greatrix for his encouragement and help with this thesis. I appreciate his support and scrutiny with the work I've done, as well as his expedience and experience in helping me when I needed it. But, most importantly, I would like to thank Dr. Greatrix for giving me the chance to study in this field, as I probably could not have done it without him.

Table of Contents

List of Tables

List of Figures

Nomenclature

Mathematical Symbols

$[\ ]$, $[\ ]_D$	Rectangular matrix or square matrix, diagonal matrix
$\{\ \}$, $\lfloor\ \rfloor$	Column vector, row vector
$[\ ]^{-1}$	Matrix inverse
$[\ ]^T$, $\{\ \}^T$	Matrix transpose (i.e. $\{\ \}=\lfloor\ \rfloor^T$), vector transpose
$\|\ \|$, $\|\|\ \|\|$	Absolute value, norm of a matrix or vector
$\vec{a}, \dot{a}$	Vector, time differentiation; i.e. $\dot{a} = da/dt$, $\ddot{a} = d^2a/dt^2$

Latin Symbols

a	sound speed; acceleration
a_n	normal acceleration
a_{lat}	lateral acceleration
a_l	longitudinal acceleration
a_x, a_y	Cartesian components of acceleration
A	port area
A_e	element area
$\{B\}$	burning vector
$\lfloor B \rfloor$	spatial derivatives of shape functions
c	damping coefficient; wave speed
$[C]$, $[c]$	global damping matrix, elemental
C, C_o	burning rate coefficient, at reference temperature
$C_n, C_{n_{FE}}, C_{n_{IBF}}$	Courant number; FE Courant number; IBF Courant number
C_p	gas heat capacity
C_s	propellant heat capacity
CB	control boundary
CFD	computational fluid dynamics
CFL	Courant-Friedrichs-Lewy
d	hydraulic diameter
d_t	throat diameter
D	particulate drag
D_p	lag parameter
$[D]$	dynamic matrix
E	total specific energy; elastic modulus (Young's modulus)
$[E]$	elastic modulus matrix
f	Darcy-Weisbach friction factor; external force
f_c	centrifugal force
f_n	natural or resonant frequency ($f = \omega/2\pi$)

$\{F\}$, $\{f\}$	global external force vector, elemental
$\{f_b\}$	body force vector
$\{F_{IN}\}$	global internal force vector
FE	finite element
G_a	accelerative mass flux component
G_{ao}	peak accelerative mass flux component
h	convection heat transfer coefficient
h^*	convective heat transfer coefficient for zero-transpiration
i, j, k, n	indexing variables
$[I]$	identity matrix
IBF	internal ballistic flow
$[J]$	Jacobian matrix
k	thermal conductivity of gas; stiffness
$[K]$, $[k]$	global stiffness matrix of structure, elemental
K	orientation correction factor
l_{ij}	length of element side $i-j$
l_{min}	minimum element side length
$l_{segment}$	segment length
$[L]$	differential operator; congruence matrix
(L, K)	segment circle center
L_c	length of nozzle convergence
L_p	length of propellant grain
m	mass
m_p	particle mass
$[M]$, $[m]$	global mass matrix, elemental
n	pressure exponent
$\vec{n}$	normal vector
$\{N\}$	directional vector
$\lfloor N \rfloor$	shape functions
p	local static pressure
p_c	mean combustion chamber pressure
p_{FE}, p_{IBF}	pressure used in FE calculations, pressure output from IBF solution
p_x, p_y	Cartesian components of p
Pr	Prandtl number
$\{P\}$	segment vector
Q	heat transfer rate from gas to particulate
$\{Q\}$, $\{q\}$	generalized (nodal) global displacement vector, elemental
$\{\dot{Q}\}$, $\{\dot{q}\}$	generalized (nodal) global velocity vector, elemental
$\{\ddot{Q}\}$, $\{\ddot{q}\}$	generalized (nodal) global acceleration vector, elemental
r	time step synchronizing index; interpolated radius; radius
r_a	propellant inner radius
r_b	overall burning rate; aluminium casing inner radius

r_{b_i}	segment burning rate
r_c	steel sleeve inner radius
r_o	base burning rate
r_p	pressure-dependent burning rate
R	specific gas constant; ratio between IBF half time step and FE time step
R_1, R_2	segment circle radius 1, radius 2
Re_d	Reynolds number based on hydraulic diameter
S	surface or surface area
SRM	solid rocket motor
t	time
t_{Al}	aluminium casing thickness
t_e	element thickness
t_{Prop}	propellant thickness
t_{St}	steel sleeve thickness
$\{T\}$	traction vector
T_i	initial propellant temperature
T_{io}	reference propellant temperature
T_f	flame temperature
T_s	surface temperature
u	Cartesian x-displacement component; local gas velocity; displacement
u_p	particulate velocity
$\{u\}$	displacement vector (i.e. $\{u, v\}^T$)
$\{\dot{u}\}$	velocity vector
$\{\ddot{u}\}$	acceleration vector
$\{u\}_i$	segment directional unit vector
U	gas velocity
v	Cartesian y-displacement component
v_w	flame front velocity
V	volume
x	distance from the head-end of the motor
x, y	Cartesian coordinates

Greek Symbols

α	shape function exponent
α_p	particulate fraction
β	heat flux coefficient; Algorithmic stability constant; shape function exponent
δ_0	reference energy film thickness
ΔH_s	net surface heat of reaction
Δl_{min}	minimum element side length
Δt	time step interval
Δx	distance between nodes
ε	surface roughness; strain; percent error
$\{\varepsilon\}$	strain vector

ϕ	acceleration orientation angle; arbitrary field variable
ϕ_d	displacement angle
$\{\phi\}$	arbitrary field variable vector
γ	gas specific heat ratio; numerical damping constant; shape function exponent
η, ξ	natural coordinates
κ	local wall dilatation
λ	arbitrary flow property
μ	gas absolute viscosity
θ, θ_o	interpolation angle, segment reference angle
ρ	mass density; local gas density
ρ_p	particulate density
σ	stress
σ_p	burning rate temperature sensitivity
σ_r	radial stress
σ_θ	circumferential stress
$\{\sigma\}$	stress vector
υ	Poisson's ratio
ω	angular velocity; eigenvalue
ω_d	damped oscillation frequency [s^{-1}]
ω_n	natural or resonant frequency [s^{-1}]
ξ_L	axial damping ratio
ζ	structural damping ratio; random position

1 Introduction

1.1 Introduction

Combustion instability has always been a factor in the understanding of current, and development and design of new, rocket motors. Combustion instability symptoms usually arise when a rocket motor is somehow disturbed from its normal operating conditions. These disturbances typically come from two possible sources: internal sources such as when pockets of unburned propellant suddenly ignite; or when cracks (in the case of solid propellant or fuel) form or voids are exposed, rapidly increasing the burning surface area; or when an igniter, insulation or unburned propellant fragment is expelled through the nozzle. Alternatively, a motor may be disturbed via an external source such as a blast wave, vibration or failure of a structural component either in a flight vehicle or on a test stand. A normally stable motor could become potentially unstable if exposed to any of these disturbances, possibly leading to a loss in performance, or even catastrophic failure of the motor.

Combustion instability has traditionally been categorized into two forms – linear and nonlinear.[1] A motor is linearly unstable when a relatively weak initial disturbance in the motor causes small amplitude waves to begin oscillating throughout the combustion chamber, eventually developing into stronger sustained pressure waves. Nonlinear combustion instability is initiated when there are relatively stronger initial disturbances present. In either case, sustained pressure waves traveling throughout the combustion chamber, may evolve into strong, shock-fronted compression waves that oscillate throughout the combustion chamber.

Nonlinear axial combustion instability is of key interest in this thesis. As already mentioned, nonlinear instability is commonly identified by sustained, large-amplitude, pressure waves that oscillate in the combustion chamber. These compression or shock waves are typically limited in their amplitude, and may be accompanied by an

overall increase in the mean (also termed time-averaged or dc) chamber pressure. There are two types of waves found during the unstable operation of a motor. These are longitudinal and transverse waves. Longitudinal waves run up and down the long axis of the motor and oscillate typically at a relatively low frequency (resonant frequency in the gas cavity in the longitudinal direction), while transverse waves run laterally and oscillate at a much higher frequency. In order to sustain these waves in the combustion chamber, the variables governing wave development must somehow be coupled to other parameters related to the instability inside the motor.

Several experimental studies on solid rocket motors (SRMs) have been made in the past, which have shown that structural oscillations have an influence on the development of sustained pressure waves in the combustion chamber.[2,3,4] Unsteady flow fields (due to wave development) will interact with the combustion process, and potentially cause a change in the propellant pyrolysis or burning rate through pressure and gas velocity changes. This will cause a change in local mass flow addition and pressure. This change in pressure will in turn affect the structure possibly leading to oscillations that may further augment the propellant burning rate.

Recent progress in calculating the propellant burning rate in the presence of an acceleration field has allowed the analysis of the effects of structural vibrations in SRM wave development.[5,6,7,8] As a motor oscillates, an acceleration field will be generated throughout the motor. It is the local accelerations experienced by the burning surface of the propellant, which augments the burning rate. Consequently, it is imperative that the structural characteristics of the SRM be known, as well as the coupling between the structural oscillations and wave development.

Earlier work done by Harris, Wong and de Champlain focused on the structural characteristics of both cylindrical and star-grain SRMs and their response to the passing of a longitudinal shock wave.[3] The influence of structural vibrations on pulse-triggered instability was demonstrated in their experiments. Later work done by Greatrix involved a comprehensive model for a cylindrical grain SRM that combined a simple

structural model with the internal ballistics of the motor.[2,8] This work showed how structural oscillations couple with the internal ballistics, influencing axial wave development in the SRM. However, this model was limited to cylindrical-grain SRMs due to the simple structural model used.

1.2 Present Study

This thesis involves the prediction and analysis of the unsteady internal ballistics of solid propellant rocket motors of various propellant grain geometries. In particular, there is an emphasis on the numerical prediction and analysis of how structural vibrations couple with the internal ballistic, flow to produce symptoms which are related to nonlinear instabilities within SRMs. The effect of including this vibration within the framework of the internal ballistic simulation model is made evident through analysis of cylindrical and star-grain motors. The induced acceleration fields from structural vibration and their effect on the pyrolysis rate of the propellant will be modelled, as will the structural response to the passing of a pressure wave. A structural analysis of the star-grain motor structure will also be carried out to try to understand how the response of the motor structure affects wave development.

1.3 The Model

The numerical model is comprised of two separate components – the internal ballistic flow solution and the SRM structural solution. The internal ballistic flow solution follows the work of Greatrix for the analysis of the unsteady internal ballistic flow in a cylindrical-grain SRM.[2] This analysis is quasi-one-dimensional in nature and uses a computational fluid dynamic (CFD) method called the Random Choice Method (Section 3.3). Employing a CFD solver that can produce a solution for the internal ballistic flow given the port geometry and acceleration field as inputs, a suitable method is then required to accurately model the structural response of a cylindrical- or star-grain SRM, and provide these inputs such that the effects of these two "modules"

may be coupled through the acceleration-augmented burning rate, to complete the simulation.

The most common method of numerical structural analysis is the finite element (FE) method, which is a suitable choice for this simulation. In order to model a star-grain SRM structure with a minimum number of assumptions, a three-dimensional model could be used. However, for this preliminary study, a series of two-dimensional FE structural solutions is evaluated at the nodes of the quasi-one-dimensional CFD module (see Figure 1-1). This simplifies passing boundary conditions while providing a correct solution to the cross-sectional dynamic response of the cylindrical- or star-grain SRM. There are of course assumptions that affect the accuracy of this type of model, which are discussed in Section 2.4.1. Nevertheless, this model should describe to some degree the coupling expected between the structure and the internal flow, allowing for the prediction of the onset, development and sustaining of symptoms associated with pulse-triggered nonlinear axial instability.

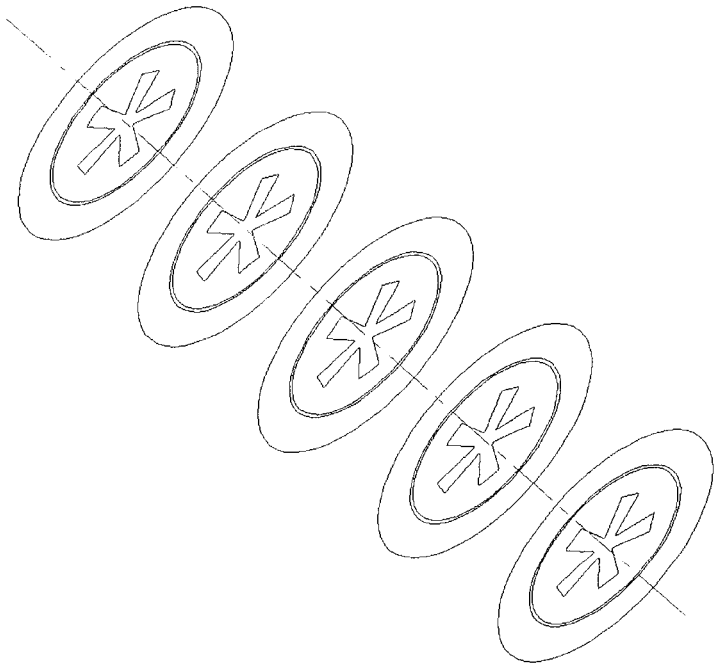


Figure 1-1 – Simplified schematic of proposed simulation model

2 Structural Modelling

2.1 Introduction

In this chapter, the development of all the Finite Element (FE) modelling components is presented in detail. The chapter begins with the generation of the required geometry and grids, and follows through to a list of assumptions used in the creation of the FE model, to the derivation of the elements themselves. Boundary conditions are explored and defined, and a methodology for a harmonic analysis of the geometry is presented. Some FE results are obtained for a cylindrical-grain motor and compared to analytical results, as well as commercial FE packages to determine solution accuracy. The dynamic response of the system is also compared to other test cases, which use commercial packages. A method to simulate propellant regression is also discussed.

2.2 Grid Generation/Grids

The choice of element geometry for any FE analysis depends on a few factors. In particular, one is concerned with the accuracy of solution and the ability to easily fit the required geometry. Triangular elements were selected for this simulation model for a few reasons. Some of these reasons involve the creation of the element and the solution algorithm; this will be discussed in Section 2.4.3. For grid generation purposes, triangular elements were selected because a triangle is the simplest two-dimensional shape, so it can easily fit into any given geometry, thus simplifying grid generation.

The grid generation for this simulation is accomplished using an unstructured triangular grid generator originally developed by Galyukov & Voinovich.[9] The program was restructured and additions made such that the necessary geometry could be input, and the newly created grid would be in a suitable form of output for the FE simulation. The generator takes a series of points that represent the boundary(s), and creates nodes

within this boundary. Triangular elements are then fitted/associated to the nodes. The initial elements are structured; however, an iterative smoothing algorithm is run on the interior nodes. This rearranges the elements to suit a set of criteria (maximum & minimum size, aspect ratio and surface area ratio) that defines the ideal element. These criteria are manually set by the user and require a little skill and experience to best generate a grid. The initial structured elements could be used for a grid in certain circumstances; however, rarely, if ever, will the elements match the boundary in a suitable fashion. It is however possible with experience and patience to retain some of the structure in the grid. Having structure in the grid is desirable as it reduces numerical noise in the solution, especially with the higher derivatives (i.e. acceleration). It can be done with the proper settings, but this does require a lot of trial and error for every grid, and even then, only with the coarsest grids. The finer the grid, the less chance there is in maintaining structure; and it is nearly impossible if anything but the simplest geometry is used (i.e., cylindrical-grain core).

Grid generation begins with the creation of the boundaries. The input to the grid generator requires a series of points that represent the boundary(s) as shown in Figure 2-1 for a cylindrical grain motor (simplified representation – actual input would have many more points along the boundary). For the sake of clarity, the process of creating the boundaries will be termed "dotting". Since the grid generator does not identify regions representing different materials (differing colors in Figure 2-1), grids for these regions must be made separately. The various grids are combined into one grid when the simulation initializes itself. An advantage of this is that grids of various components may be swapped (for either a coarser or finer one) without creating a new grid. However, the key limitation here is that the nodes along the boundaries of the meshes of two differing materials must be both equal in number and position.

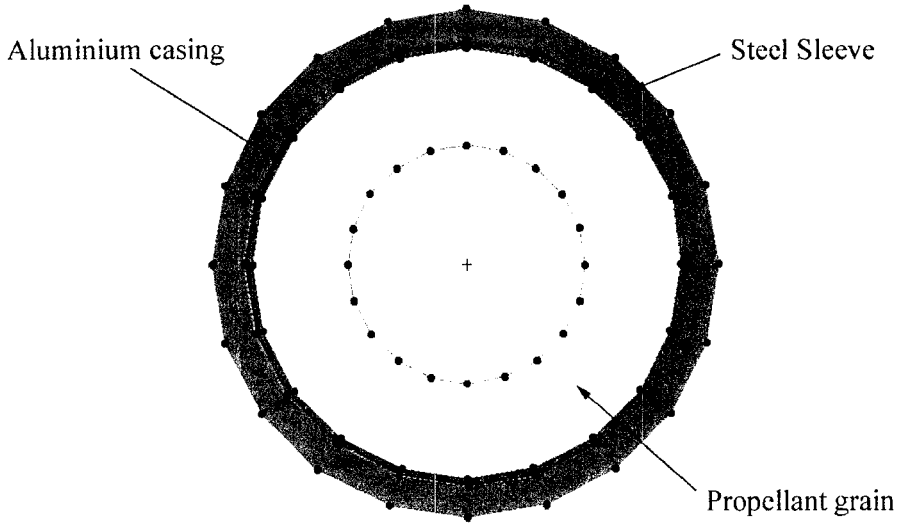


Figure 2-1 – Grid generator input

Therefore, if there are 20 nodes along the interior of one grid, there must be 20 matching nodes along the exterior of the other (see Figure 2-2). This matching must be done before the boundaries are input into the grid generator. Once the initialization routines are run, the grids are merged and duplicate nodes are deleted.

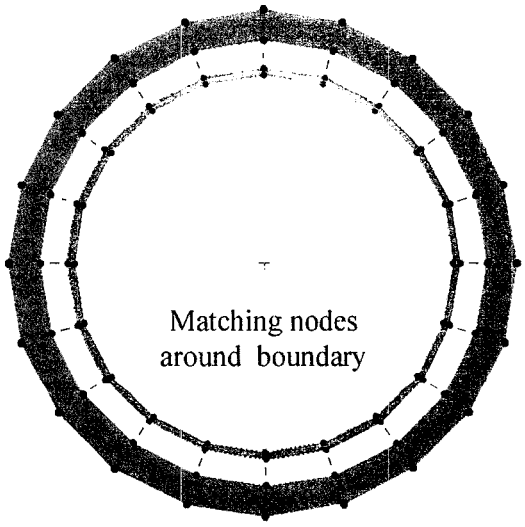


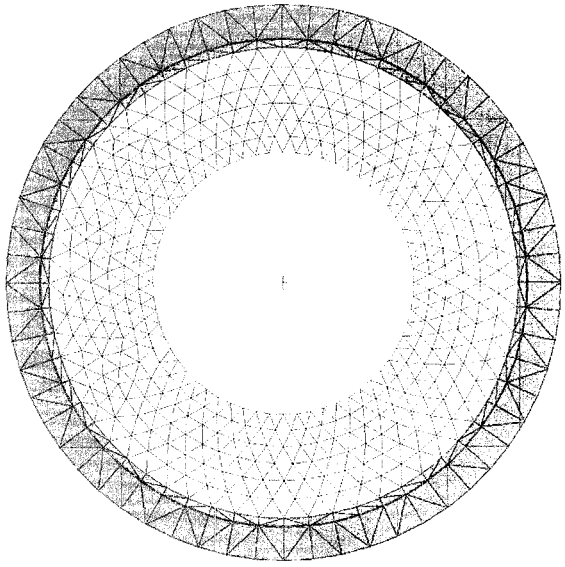
Figure 2-2 – Node matching along boundary

The majority of the boundaries in this simulation are circular, so the dotting of these boundaries is straightforward. A simple program easily accomplishes this. Circular grids include those for the steel sleeve (Steel sleeve employed for static tests only, not in flight), the aluminum casing and for cylindrical grains. Non-cylindrical propellant grains are still dotted in the same manner, but the interior boundary of the grain is no longer circular; a simple computer program to dot the boundary is no longer sufficient. For example, the star-grain core shapes used in later simulations were first drawn in AutoCAD and exported as .DXF files. A translation routine was written to convert the .DXF output into a useable form. Since the star-grain core shape is the internal boundary, it does not need to match with another grid; only the circular outer boundary does.

A variety of grids was created at varying levels of refinement, both with cylindrical and five-point star-grain core shapes. For the cylindrical grain motors, the dimensions of the motor were taken from previous experimental / computational work done.[2] This was done in order to correlate the output from the simulation with previous research. The structural dimensions and material properties for the thin sleeved cylindrical grain motors used in Reference 2 are shown in Table 2-1. The grids shown in Figure 2-3 to Figure 2-6 are for the cylindrical grain at varying levels of refinement. The outer steel sleeve (cyan) is 4.67 mm thick, the aluminum casing (magenta) is 1.27 mm thick and the propellant (yellow) is 14.4 mm thick with an inner radius of 18 mm. Here, as one goes from Figure 2-3 to Figure 2-6, the grids become finer and finer, as well as there being less structure in the mesh. As mentioned earlier, the finer the grid, the more difficult it is to maintain element structure. Only in Figure 2-3 is there any significant structure in the mesh. Also, note that although all components of the motor are shown, they are still separate grids at this point. It can be seen how the nodes along the contacting boundaries match with each other. An important point here is that different grids can be used with each other so long as the boundary nodes match, so if the propellant grid is not sufficiently smooth, it may be swapped for a finer grid without changing the aluminum casing grid or the steel sleeve grid.

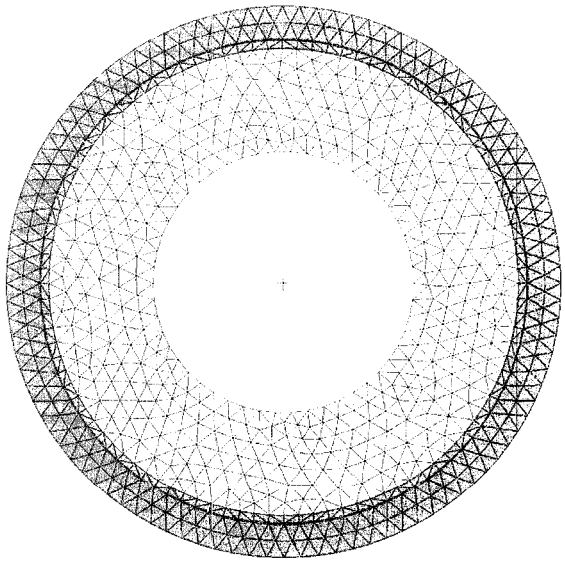
Steel Sleeve		Units
Inner wall radius (r_c)	33.67	mm
Thickness (t_{st})	4.67	mm
Elastic modulus (E_{st})	200	GPa
Poisson's ratio (υ_{st})	0.300	-
Density (ρ_{st})	7850.0	kg/m^3
Aluminium Casing		Units
Inner wall radius (r_b)	32.40	mm
Thickness (t_{Al})	1.27	mm
Elastic modulus (E_{Al})	80	GPa
Poisson's ratio (υ_{Al})	0.330	-
Density (ρ_{Al})	2700.0	kg/m^3
Propellant	Color used in all Figures	Units
Inner wall radius (r_a)	18.00	mm
Thickness (t_{Prop})	14.40	mm
Elastic modulus (E_{Prop})	0.045	GPa
Poisson's ratio (υ_{Prop})	0.497	-
Density (ρ_{Prop})	1730.0	kg/m^3

Table 2-1 – Cylindrical-grain geometry and material properties



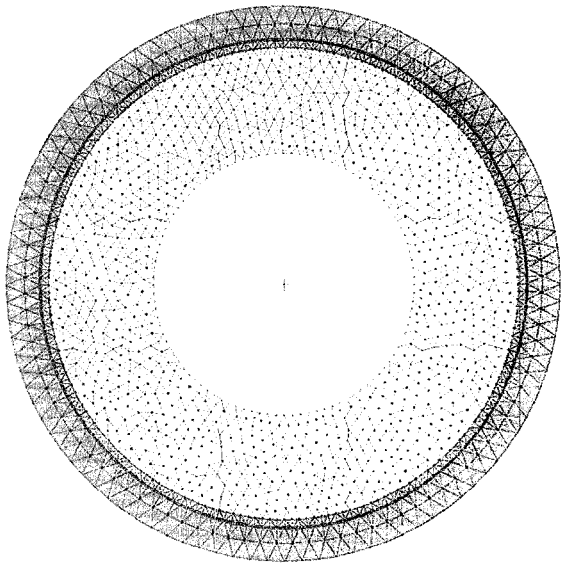
556 Nodes
992 Elements

Figure 2-3 – Cylindrical-grain mesh CG1



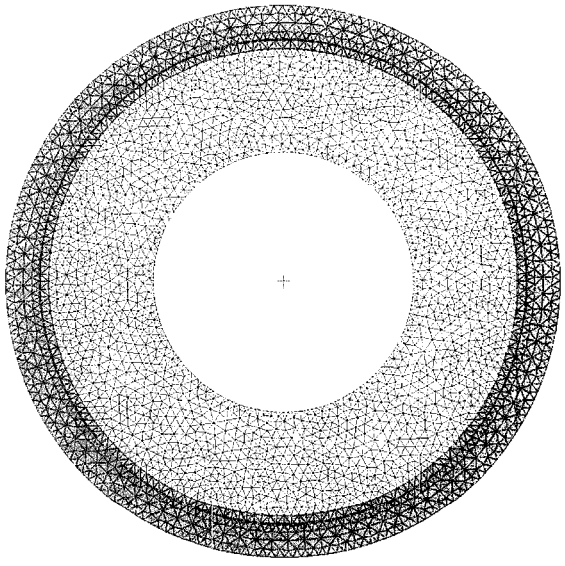
784 Nodes
1418 Elements

Figure 2-4 – Cylindrical-grain mesh CG2



1673 Nodes
3146 Elements

Figure 2-5 – Cylindrical-grain mesh CG3



2957 Nodes
5514 Elements

Figure 2-6 – Cylindrical-grain mesh CG4

Propellant grids for the star-grain core shape are later shown in Figure 2-8 to Figure 2-11, with the level of refinement increasing towards Figure 2-11. In these figures, the motor geometry is similar to the star-grain motors used in Reference 3; again to correlate the simulation output with previous experimental and numerical data (note these motors have a larger 14.1 mm steel sleeve, and identical aluminum casing) Dimensions for $1/10^{th}$ of the initial (pre-burn) five-point star port shape is shown below in Figure 2-7.

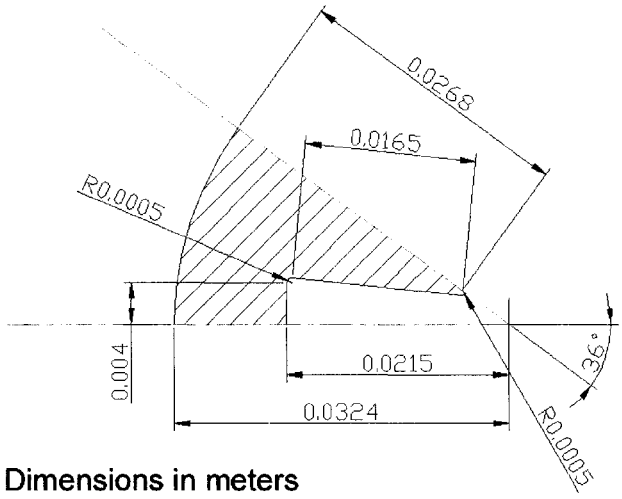


Figure 2-7 – Star shape dimensions

In Figure 2-8 to Figure 2-11, there is no longer any visible structure in the propellant mesh due to the more complex geometry. Note how the triangular elements easily fit the given geometry and how they are concentrated in regions where a lot of definition is required. This increase in definition or refinement is not due to the need to have a finer grid in regions of stress concentrations (although this is desired, the grid generator has no way of knowing where these stress concentrations are beforehand), but due to the increase in the number of points that define the boundary in that region (Figure 2-12). Therefore, it is possible to control where there will be a greater density of elements in the grid by dotting a density of points at specific regions on the boundary before being input into the grid generator. This method of mesh refinement must be carefully coordinated with the generator criteria settings mentioned earlier, and may be a cause for concern due to the trial and error nature of this process.

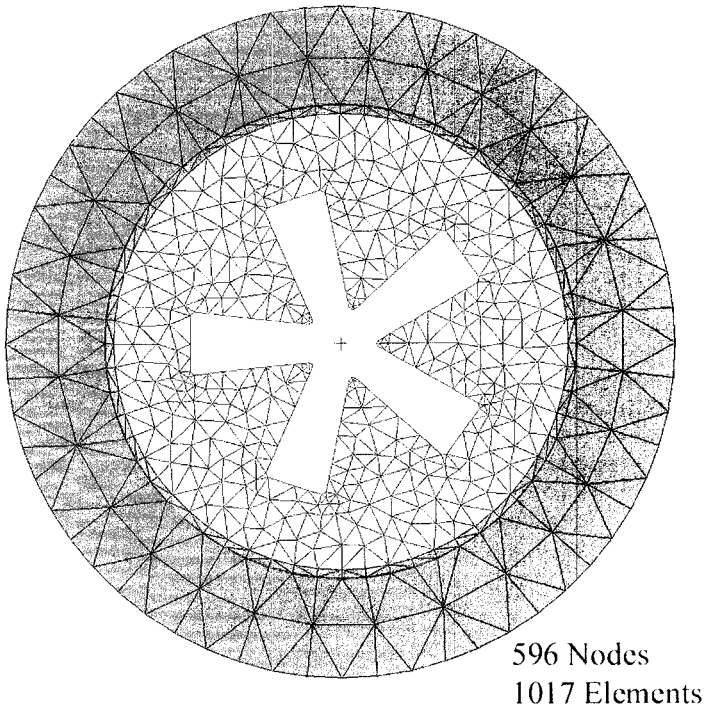


Figure 2-8 – Star-grain mesh SG1

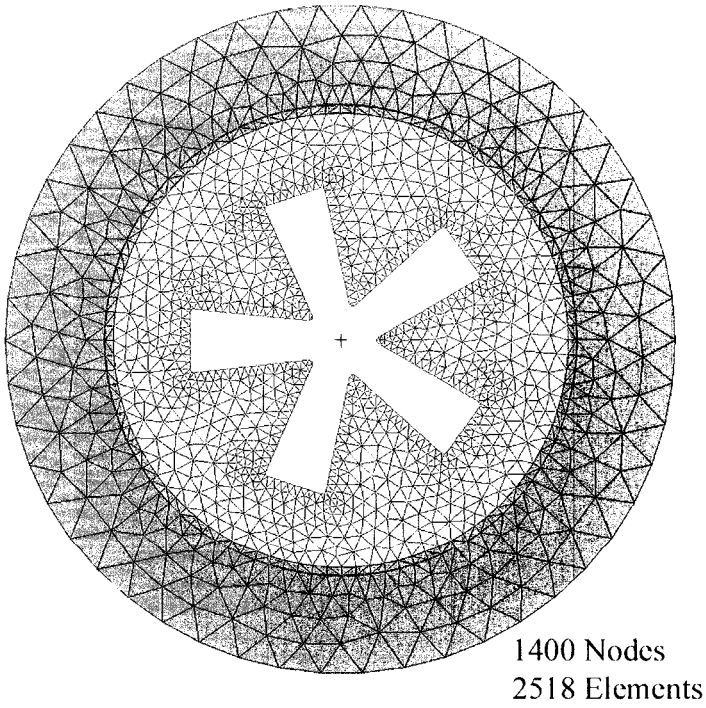


Figure 2-9 – Star-grain mesh SG2

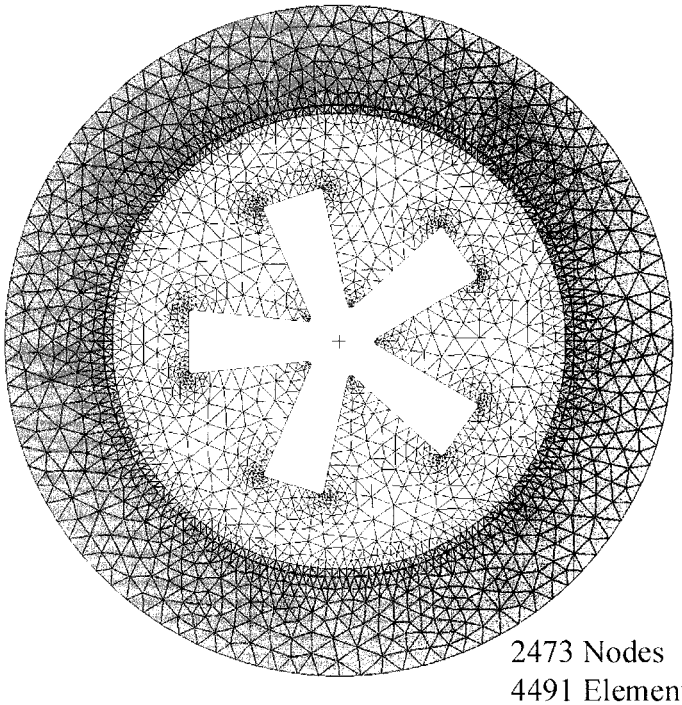


Figure 2-10 – Star-grain mesh SG3

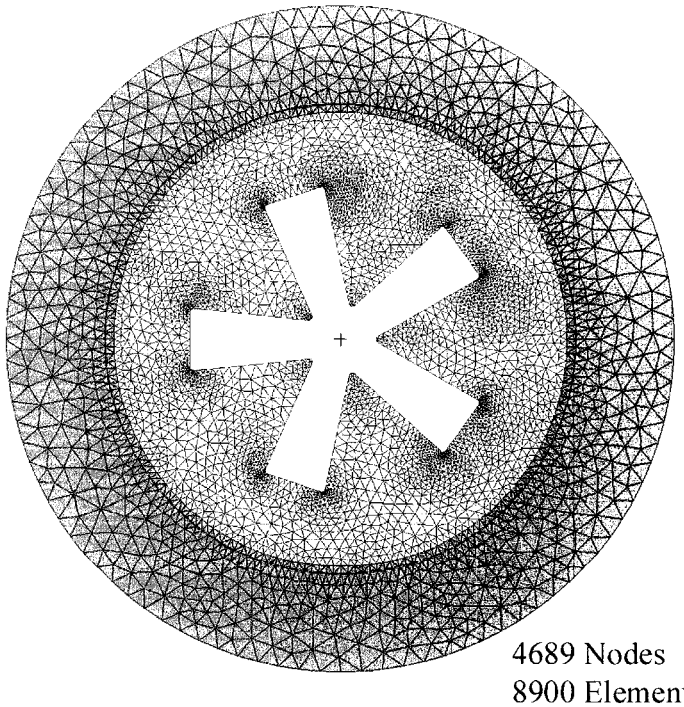


Figure 2-11 – Star-grain mesh SG4

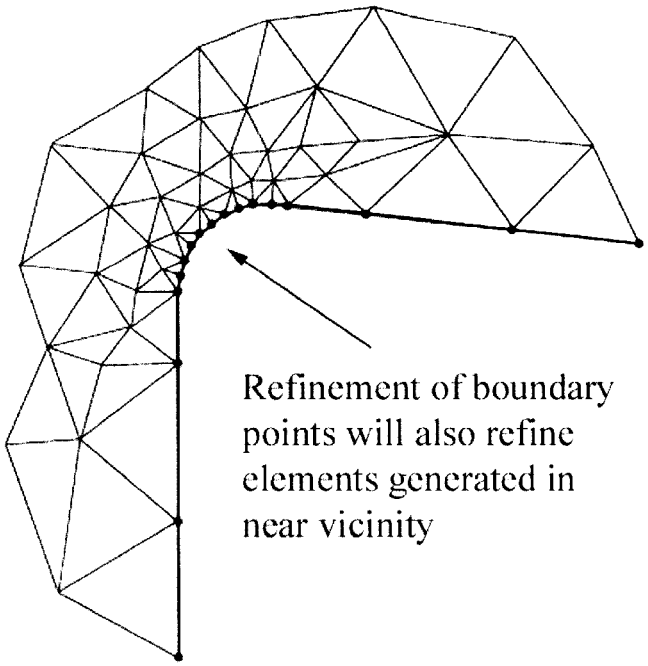


Figure 2-12 – Grid refinement

An important observation is that the number of nodes and elements is greatly increased with the more complicated geometry, and hence so are the total degrees of freedom for the grid. This means that the star-grain grids will require more total computing time than the cylindrical-grain grids. This is mainly due to the increased refinement in the star-grain meshes at corners and points. Selective boundary dotting techniques can be used to minimize the number of nodes and elements in the grid. A creative touch along with patience helps in this process. Reducing the number of nodes and elements in a grid does two things – both will reduce computational time. A reduction in nodes reduces the degrees of freedom in the system. A reduction in elements reduces the number of required calculations per degree of freedom in the system. Care must be taken not to compromise the accuracy of the solution by having too coarse a grid. The level of refinement at which point making the grid any finer does not reduce the error in the solution is called the point of grid independence. Grid independence is found by comparing the FE solution of a given geometry with itself using progressively finer grids. Once there is a negligible difference in the solution using finer grids, one can say grid independence is achieved. However, the level of refinement at which the solution error is independent of the grid (and the computational

time) may be quite high depending on the geometry. So, coarser grids are utilized to reduce computer time at the expense of an increase in error. This is why many grids are generated to see which grid gives the best results with the least expensive computational requirements. Although there is a little trial and error in doing this, the most suitable grid is easily found this way.

2.3 Grid Symmetry

As one may see in Figure 2-3 to Figure 2-6 and Figure 2-8 to Figure 2-11, the geometry exhibits a certain level of symmetry. The exploitation of symmetry in a given geometry has perhaps the greatest impact in reducing the total degrees of freedom in the system. Mentioned earlier, the total degrees of freedom in the FE system have a marked impact on both computer time and accuracy. The previously mentioned methods used to reduce the degrees of freedom all tend to affect accuracy in some way (though not always in an adverse manner). Correctly using symmetry in the grid reduces the total degrees of freedom, but does not affect solution accuracy. The key trick in exploiting symmetry is the proper implementation of the necessary boundary conditions on the given geometry (see Section 2.4.5).

In order to simplify the grid generation process, the task of utilizing geometric symmetry is given to the initialization routines in the simulator, and not to the grid generator. Therefore, whole grids are always input into the simulator, and are "cut down" by a prescribed amount. Doing this allows quick changes in the amount of symmetry used in a simulation, without going through the trouble of creating another grid. Via this approach, only complete grids as shown in Figure 2-3 to Figure 2-6 and Figure 2-8 to Figure 2-11 are meshed using the grid generator; breaking up the whole grids into symmetrical parts is done upon program execution.

Since all the geometry used in this thesis is essentially radial in section, any symmetry will come in the shape of pie slices of the circle. To break the grid up into symmetrical parts, the amount of symmetry in the geometry must be determined by the

user. To best determine where to "cut" the grid, one would try to find symmetry lines in the grid where there is only radial motion (or displacement). For example, in a cylindrical grain, there is literally an indefinite amount of symmetry, so any value will nominally suffice (from experience a $1/20^{th}$ section is the smallest that should be used; otherwise, the total degrees of freedom becomes too low). However, for a five-point star-grain port shape, either a $1/5^{th}$, or better yet a $1/10^{th}$, pie slice must be used. To break up the grid, consider the SRM geometry in Figure 2-3. Knowing the amount of symmetry in the grid, symmetry lines can be created at the appropriate angles such that only the elements to be retained are bound between them. Of course the symmetry lines will cross through a few elements, but the nodes of these elements can easily be moved to lie on the symmetry lines such that the entire element is retained. The remaining elements and nodes may be deleted. This process is illustrated Figure 2-13.

Care must be taken with certain grids such that the reshaped elements along the symmetry lines are not poorly formed (no large obtuse angles in the elements or "needle"-shaped elements, or elements with a very small area). A simple algorithm is used to alleviate any poorly shaped elements, but it is not always reliable, and poor elements do at times arise. So, some discretion must be used in the process of selecting symmetry with any specific grid. With the exploitation of symmetry in these grids, appropriate boundary conditions must be enforced to ensure the grid is properly constrained to prevent lateral rigid body motion. For the type of symmetry encountered in the grids used in this thesis, the only boundary condition that needs to be considered, is that the nodes that lie <u>on</u> the symmetry lines can only move along the symmetry line (radially from the center of the complete structure). This boundary condition of course is what defines the amount of symmetry in the grid in the first place, and is easily implemented in the transient calculations. This boundary condition is described in detail in Section 2.4.5. Two grids of five-point star-grain motors that exploit symmetry are shown below in Figure 2-14 and Figure 2-15. These are $1/10^{th}$ pie slices of the grids in Figure 2-9 and Figure 2-11 respectively. By the node and element count in these grids, the advantages of using symmetry are clear.

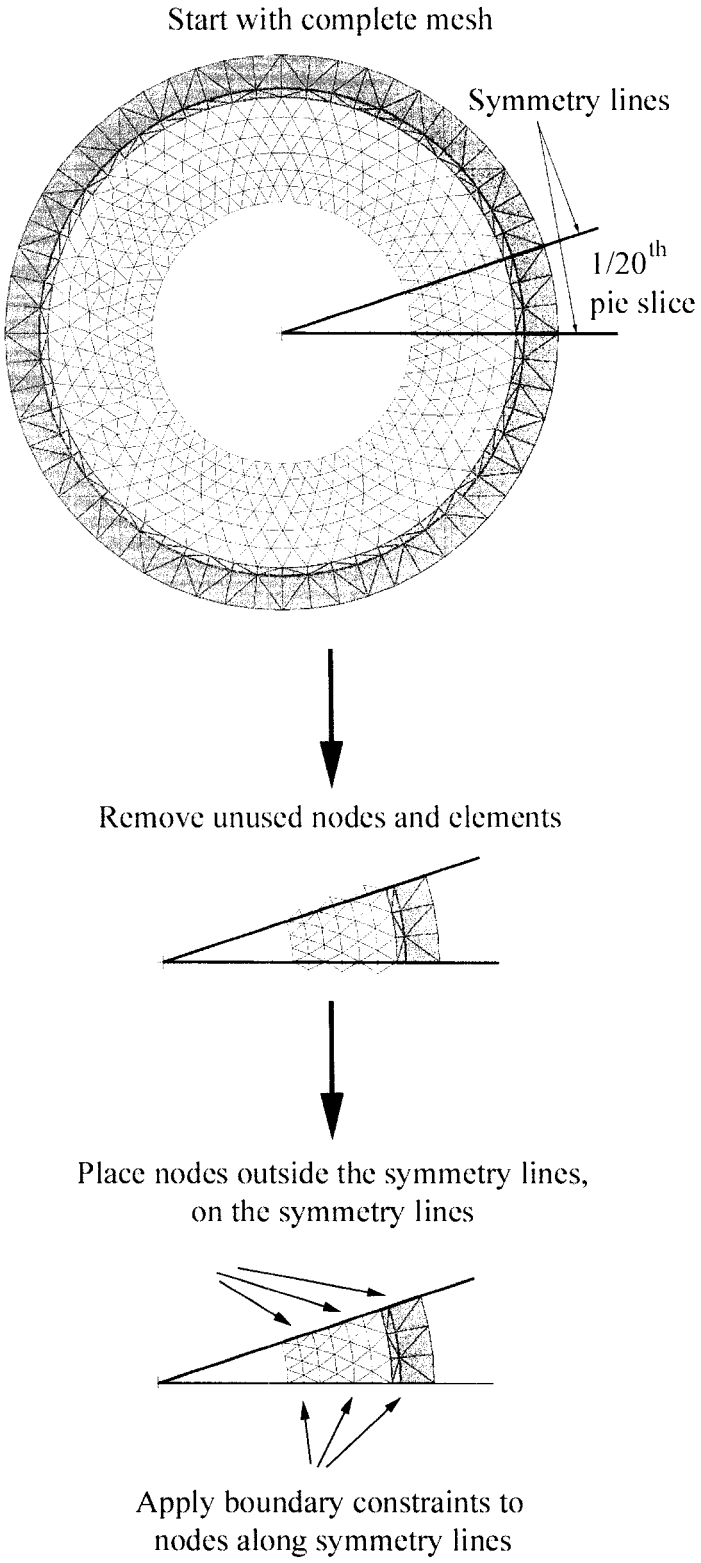


Figure 2-13 – Exploiting symmetry

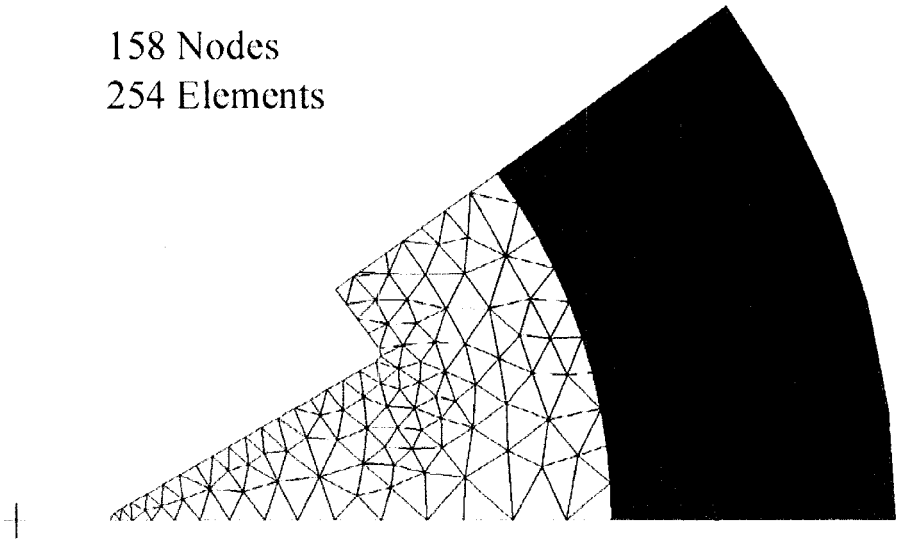


Figure 2-14 – Star-grain mesh SG2-10

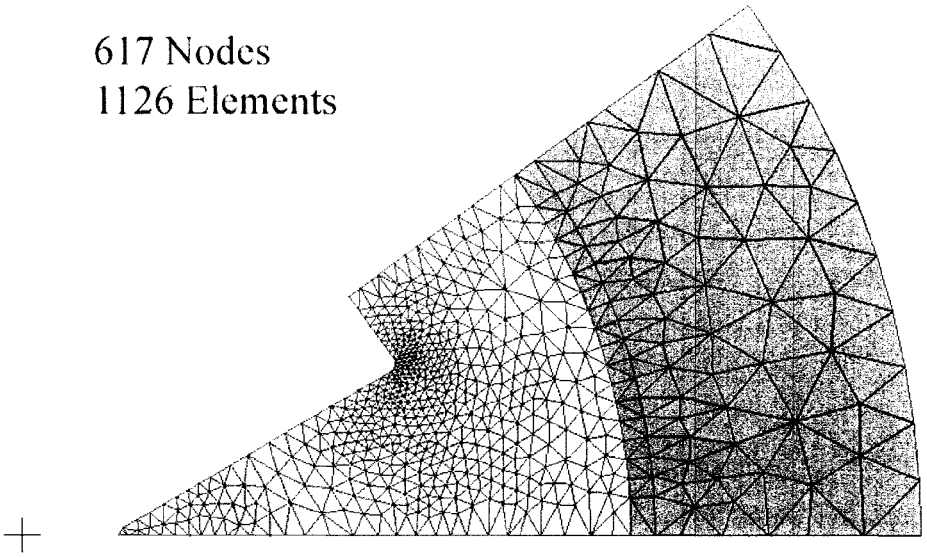


Figure 2-15 – Star-grain mesh SG4-10

2.4 Structural Modelling

2.4.1 Assumptions

As with most engineering solutions, there are assumptions that have to be made in order to simplify the system, otherwise the complexity (and computational requirements if a numerical method is used) may be too great to handle in a practical manner. In this simulation model, there are a few assumptions used to simplify the system. These are listed in the following paragraphs.

As this is not a true three-dimensional simulation, but an arrangement of two-dimensional sections along an orthogonal axis, some approximations inherent in a two-dimensional FE analysis must be used. The basic structure of the motor is essentially a long cylinder, so it's reasonable to use a plain strain analysis (where there is zero strain in the direction normal to the 2-D section). However, this then leads to some other imposed assumptions. First, using a plain strain analysis, the motor is assumed infinitely long. This of course is not true, but for the plain strain model to be consistent, this must be assumed. This means that the end effects in the motor structure are ignored. In reality, the plain strain model is only true near the center of the motor. At the ends, a plain stress (where there is zero stress in the direction normal to the 2-D section) approximation would be the most appropriate. Nevertheless, since only one of these approximations can be used, a plain strain analysis is deemed the better of the two, as its solution is valid over a larger part of the motor. Secondly, as these are a series of two-dimensional sections, there is no structural connection between the sections. There is of course a connection between sections when it comes to the internal ballistic flow calculations, but the sections do not affect each other in the FE solution.

Another important assumption used in this simulation is that all materials behave in a linear elastic manner. This is a good assumption for any metal materials in the motor as both aluminum and steel exhibit this behaviour below their yield stress. As for the propellant, this assumption is less accurate as this rubber-like material exhibits a

visco-elastic behaviour. Although it is possible to model a visco-elastic material using an FE analysis, current methods require a complex nonlinear analysis along with other experimental and/or empirical material data. This would substantially increase the simulation complexity along with the required computer power. To model the propellant a nonlinear large deformation analysis may be used, where the stiffness properties are a function of the displacements. For the transient FE analysis in this simulation, a basic viscous structural damping model is employed based on the resonant frequency of the structure. For a study of this nature, this is deemed the best choice. Structural damping is discussed in detail in Section 2.4.4.

In a typical SRM, internal temperatures are very high. This means heat fluxes are high into the exposed materials. The propellant surface is exposed to the high temperatures in any one section along the interior boundary. Here, at or near the burning surface, thermally induced strains in a material can potentially alter the final FE solution. However, the thermal conductivity of the propellant is very low, and burn-back of the grain reduces the heat penetration to a relatively thin zone, so neglecting heat transfer in the FE simulation is a reasonable approximation.

Other assumptions for the FE model, involve the propellant regression or burn-back. The burning rate and direction is assumed constant over a given time step. Since the time step in the transient FE analysis is small, this is a reasonable approximation for the current study. Furthermore, the direction of propellant regression or "burn vector" is considered normal to the local surface. Again, in the absence of a more sophisticated model, this is considered a reasonable choice. Propellant regression and burning rate calculations are discussed in Sections 2.7 and 3.4 respectively.

Finally, with respect to propellant regression affecting the FE solution; at the time of writing, the technology of elements having the capability of changing shape while a transient solution is being run (thus having transient stiffness, mass and damping properties) is still in its infancy. The FE method used here does not completely account for a changing mesh. Its mass, damping and stiffness properties are

frozen at the point where FE calculations are commenced. This technique creates adequate output over a short period of burning time – which is all that is required.

As mentioned earlier, many of these assumptions are made to simplify the system. Nevertheless, there are forced assumptions employed as well, that are limited by the current models available to simulate various properties of the system. These assumptions must be used for lack of a more sophisticated model. The following sections show how these assumptions affect the development and construction of the simulation model.

2.4.2 Governing Equations

In problems of static elasticity, the static equilibrium equations for a two-dimensional analysis in matrix form are listed below (dynamic terms will be introduced in Section 2.4.4):[10]

$$[L]^T\{\sigma\}+\{f_b\}=0 \tag{2.1}$$

where $[L]$ is the differential operator, $\{\sigma\}$ is the stress vector and $\{f_b\}$ is the body force vector. These are defined as

$$[L]=\begin{bmatrix} \dfrac{\partial}{\partial x} & 0 \\ 0 & \dfrac{\partial}{\partial y} \\ \dfrac{\partial}{\partial y} & \dfrac{\partial}{\partial x} \end{bmatrix} \tag{2.2}$$

$$\{\sigma\}=\{\sigma_x \quad \sigma_y \quad \tau_{xy}\}^T \tag{2.3}$$

$$\{f_b\}=\begin{Bmatrix} f_{bx} \\ f_{by} \end{Bmatrix} \tag{2.4}$$

where σ_x and σ_y are the stresses in the x and y directions respectively and τ_{xy} is the shear stress. The body forces in the x and y directions will be elucidated further on in this section. In addition to Equation (2.1) are expressions to relate stresses to strains (Hooke's Law) and strains to displacements. Hooke's Law, neglecting any initial stresses or strains, is identified as

$$\{\sigma\}=[E]\{\varepsilon\} \tag{2.5}$$

where $[E]$ is the elastic modulus matrix and $\{\varepsilon\}$ is the strain vector. These are defined as[10]

$$[E]=\frac{E}{(1+\upsilon)(1-2\upsilon)}\begin{bmatrix} 1 & \upsilon & 0 \\ \upsilon & 1 & 0 \\ 0 & 0 & \dfrac{1-2\upsilon}{2} \end{bmatrix} \tag{2.6}$$

$$\{\varepsilon\}=\{\varepsilon_x \quad \varepsilon_y \quad \gamma_{xy}\}^T \tag{2.7}$$

where υ is Poisson's ratio and E is the elastic modulus for the material; ε_x and ε_y are the strains in the x and y directions and γ_{xy} is the shear strain. Note that the elastic modulus matrix (Equation (2.6)) is for a plain strain analysis as opposed to a plain stress analysis. It is important to note this difference, as it is part of the original assumptions of this simulation.

The expressions to relate strains to displacements are based on small displacement strain assumption, and are expressed as (large displacement analysis is discussed later in this section) [10]

$$\varepsilon_x = \frac{\partial u}{\partial x} \qquad \varepsilon_y = \frac{\partial v}{\partial y} \qquad \gamma_{xy} = \frac{\partial v}{\partial x} + \frac{\partial u}{\partial y} \tag{2.8}$$

$$\therefore \{\varepsilon\} = \left\{ \frac{\partial u}{\partial x} \quad \frac{\partial v}{\partial y} \quad \frac{\partial v}{\partial x} + \frac{\partial u}{\partial y} \right\} \tag{2.9}$$

If the displacement vector is defined as $\{u\} = \{u \quad v\}^T$, where u and v are the displacements in the x and y directions, then the strain vector is better expressed as

$$\{\varepsilon\} = [L]\{u\} \tag{2.10}$$

Therefore, by combining Equations (2.10) and (2.5) into (2.1), the governing differential equations in matrix form can be expressed as

$$[L]^T [E][L]\{u\} + \{f_b\} = 0 \tag{2.11}$$

2.4.3 FE Stiffness Derivations

The finite element method is a numerical method that takes a system (that typically cannot be solved for by analytical means), and breaks it up into more manageable and simpler "finite elements". The governing equations are then solved for over the simpler elements in a piece-wise manner using an approximating polynomial over the element. In the beginning of this chapter, the use of triangular elements to define the physical domain was noted, partially because of the ease with which a triangle can fit into a given geometry. Other reasons to use a triangular element are to reduce the degrees of freedom in an element and to reduce the computational time. Being a two-dimensional analysis, there are two degrees of freedom per node so a triangle having a node at each corner will have six degrees of freedom (higher order elements may have mid-side nodes as well increasing the total degrees of freedom to twelve or higher). However, reducing the degrees of freedom per element has the

tendency of reducing the accuracy in the solution. So in order to maintain accuracy in the solution with a lower order element, more elements must be used. This problem is commonly dealt with by using higher order elements that have more degrees of freedom per element (and so less elements overall). In this simulation, three-node (six degrees of freedom per element) triangles are used effectively.

Using three-node triangles allows the use of a linear approximating polynomial – or shape function – to approximate the displacement field over the element. A displacement-field-based element is used, as the second time derivative (acceleration) is of importance in this simulation. As there are three nodes in the element, three shape functions are required. It is simpler to create a set of coordinates that are local or "natural" to the element in order to evaluate the shape functions. This will also allow the mapping of the discretized equations from local to global coordinates, and make integration of the shape functions simpler afterwards. A triangle, being a two-dimensional shape, only requires two natural coordinates (ξ, η) to define it. These coordinates axes are typically in-line with two of the triangles sides (Figure 2-16), and the length ranges from 0 to 1.

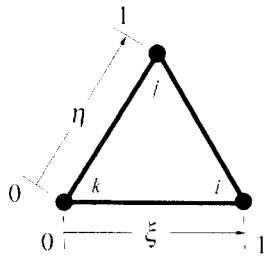


Figure 2-16 – Natural coordinates of an element

Any variable inside the element must be represented in terms of the nodal values. Using an arbitrary variable $\phi(x,y)$, this is expressed as[11,12]

$$\phi = N_i\phi_i + N_j\phi_j + N_k\phi_k = \lfloor N \rfloor \{\phi\} \tag{2.12}$$

where $\lfloor N \rfloor = \lfloor N_i \quad N_j \quad N_k \rfloor$ and is the shape functions of node i, j and k respectively; and $\{\phi\}$ are the nodal values of ϕ. If $\{\phi\}$ and ϕ are set to unity, Equation (2.12) becomes[11,12]

$$1 = N_i + N_j + N_k \tag{2.13}$$

There are three shape functions and only two nodal coordinates, therefore only two of the shape functions are linearly independent, while the third is a function of the other two. Thus, Equation (2.13) is an important property of the shape functions as it links the third shape function to the other two. Looking at Equation (2.12), at node i, ϕ must equal ϕ_i, therefore N_i must equal 1 while N_j and N_k equal 0. This follows similarly for nodes j and k as well. Taking this into account it is safe to say that any shape function may vary from 0 to 1. As the two natural coordinates also vary from 0 to 1, it is simple enough to evaluate the first two shape functions along these coordinates such that $N_i = \xi$ and $N_j = \eta$. Using Equation (2.13), the third shape function may be also evaluated. Therefore, the three linear shape functions are[12]

$$N_i = \xi \qquad N_j = \eta \qquad N_k = 1 - \xi - \eta \tag{2.14}$$

Using Equation (2.12), the displacement field {u} of the element may be expressed as

$$u = N_i u_i + N_j u_j + N_k u_k \tag{2.15a}$$

$$v = N_i v_i + N_j v_j + N_k v_k \tag{2.15b}$$

It is simpler to express the nodal displacements in the u and v directions in Equations (2.15a) and (2.15b) in terms of an elemental displacement vector $\{q\}$:[12]

$$\begin{Bmatrix} u_i \\ v_i \\ u_j \\ v_j \\ u_k \\ v_k \end{Bmatrix} = \begin{Bmatrix} q_1 \\ q_2 \\ q_3 \\ q_4 \\ q_5 \\ q_6 \end{Bmatrix} = \{q\}$$

where $\{u\}$ may now be expressed in matrix form as

$$\{u\} = \begin{bmatrix} N_i & 0 & N_j & 0 & N_k & 0 \\ 0 & N_i & 0 & N_j & 0 & N_k \end{bmatrix} \{q\} = \lfloor N \rfloor \{q\} \tag{2.16}$$

Equation (2.16) defines the linear displacement field inside the element knowing only the nodal displacements.

Having a linear displacement field defined, the governing equations must now be discretized over the domain. In this thesis, the governing equations are discretized using Galerkin's method. Galerkin's method is based on the method of weighted residuals where the shape function is used as the weighting function. The approximation is resolved by setting the residual of a function relative to the weighting function to zero and integrating across the domain:[11]

$$\iiint_V \lfloor N \rfloor^T (R) dV = 0 \tag{2.17}$$

where R is the residual, which in this case is Equation (2.1) or (2.11). The divergence theorem integrated by parts,[13]

$$\iiint_\Omega \alpha \nabla \cdot \vec{V} d\Omega = \iint_\Gamma \alpha \vec{n} \cdot \vec{V} d\Gamma - \iiint_\Omega \nabla \alpha \cdot \vec{V} d\Omega \tag{2.18}$$

is applied to Equation (2.17) becoming (where $\alpha \equiv \lfloor N \rfloor^T$ and $\nabla \cdot \vec{V} \equiv [L]^T \{\sigma\}$ from Equation (2.1), and $\Omega \equiv V$ and $\Gamma \equiv S$):[12]

$$0 = \iiint_V \lfloor N \rfloor^T \left([L]^T \{\sigma\} + \{f_b\} \right) dV \tag{2.19}$$

$$\therefore 0 = \iint_S \lfloor N \rfloor^T \vec{n} \cdot \{\sigma\} dS - \iiint_V \lfloor N \rfloor^T [L]^T \{\sigma\} dV + \iiint_V \lfloor N \rfloor^T \{f_b\} dV \tag{2.20}$$

In Equation (2.20), S represents the surface area of the boundaries of the element and V the volume of the element. The first term $\vec{n} \cdot \{\sigma\}$ represents the vector components of all the total stresses and forces acting along the boundary. These include pressure and shear forces, and point loads. Pressure and shear force vectors are commonly combined together and called a traction force $\{T\}$, where $\{T\} = \{T_x \quad T_y\}^T$ and T_x and T_y are the x and y traction force components respectively. There are no point loads in this simulation, so all non-relevant terms are dropped for clarity. The traction term has relevance only at the physical boundaries of the system, as the term cancels out in between elements. The traction forces will be elaborated upon in Section 2.4.5. Inserting Equations (2.5), (2.10) and (2.16) into (2.20), the final form of the governing equations discretized over one element is

$$\iiint_V \lfloor N \rfloor^T [L]^T [E][L] \lfloor N \rfloor \{q\} dV = \iint_S \lfloor N \rfloor^T \{T\} dS + \iiint_V \lfloor N \rfloor^T \{f_b\} dV \tag{2.21}$$

In Equation (2.21) in the left-hand term, the elemental displacement vector {q} may be factored out leaving $\iiint_V \lfloor N \rfloor^T [L]^T [E][L] \lfloor N \rfloor dV$. This remaining term is commonly called the stiffness of the matrix or stiffness matrix; and analogous to spring stiffness, is combined into one term $[k]$, where $[k]\{q\}$ would represent the internal force in the element. The right-hand terms in Equation (2.21) are the traction forces and the body forces. These two forces constitute the externally applied forces over the element,

which are combined into one term $\{f\}$. Equilibrium dictates that the internal forces must equal the externally applied forces, therefore

$$[k]\{q\}=\{f\} \tag{2.22}$$

where,

$$[k]=\iiint_V [N]^T [L]^T [E][L][N]dV \tag{2.23}$$

$$\{f\}=\iint_S [N]^T \{T\}dS + \iiint_V [N]^T \{f_b\}dV \tag{2.24}$$

Equations (2.23) and (2.24) are the product of the discretization process. However, to be able to apply and integrate these equations across any arbitrarily shaped element, a mapping process must be defined for the differential operator $[L]$. The terms in the differential operator are differentials with respect to the Cartesian x and y coordinates. The differential operator operates on the shape functions, which are defined in natural coordinates. Therefore, these differentials must be transformed into the natural ξ and η coordinates of an element. The transformation will allow the geometry of a triangular element of arbitrary shape to be mapped to a so-called "master element" (see Figure 2-16). The master element, defined in natural coordinates, will allow Equations (2.23) and (2.24) to be integrated without any great difficulty.

The transformation involves changing the Cartesian coordinate differential operators into natural coordinate differential operators. Using Equation (2.6), it can be shown that the geometry of an element defined in x and y coordinates can be represented as a function of its nodal coordinates and a function of natural coordinates using the shape functions:[12]

$$x = N_i x_i + N_j x_j + N_k x_k \tag{2.25a}$$

$$y = N_i y_i + N_j y_j + N_k y_k \tag{2.25b}$$

$$\therefore x = (x_i - x_k)\xi + (x_j - x_k)\eta + x_k \tag{2.26a}$$

$$\therefore y = (y_i - y_k)\xi + (y_j - y_k)\eta + y_k \tag{2.26b}$$

Using the notation $x_{ij} = x_i - x_j$ and $y_{ij} = y_i - y_j$ (and permutations thereof), the above equations may be written as

$$x = x_{ik}\xi + x_{jk}\eta + x_k \tag{2.27a}$$

$$y = y_{ik}\xi + y_{jk}\eta + y_k \tag{2.27b}$$

Equations (2.27a) and (2.27b) show that the Cartesian x and y coordinates are functions of ξ and η. To transform the differential terms in $[L]$, the chain rule for partial derivatives is used to express the differentials with respect to the natural coordinates ξ and η:[12,14]

$$\frac{\partial}{\partial \xi} = \frac{\partial}{\partial x}\frac{\partial x}{\partial \xi} + \frac{\partial}{\partial y}\frac{\partial y}{\partial \xi} \tag{2.28}$$

$$\frac{\partial}{\partial \eta} = \frac{\partial}{\partial x}\frac{\partial x}{\partial \eta} + \frac{\partial}{\partial y}\frac{\partial y}{\partial \eta} \tag{2.29}$$

which can be expressed in matrix form as

$$\begin{Bmatrix} \frac{\partial}{\partial \xi} \\ \frac{\partial}{\partial \eta} \end{Bmatrix} = \begin{bmatrix} \frac{\partial x}{\partial \xi} & \frac{\partial y}{\partial \xi} \\ \frac{\partial x}{\partial \eta} & \frac{\partial y}{\partial \eta} \end{bmatrix} \begin{Bmatrix} \frac{\partial}{\partial x} \\ \frac{\partial}{\partial y} \end{Bmatrix} = [J] \begin{Bmatrix} \frac{\partial}{\partial x} \\ \frac{\partial}{\partial y} \end{Bmatrix} \tag{2.30}$$

The transformation matrix $[J]$ is commonly called the *Jacobian* of the transformation. Taking the derivatives of Equations (2.27a) and (2.27b) with respect to the natural coordinates, the derivative terms in $[J]$ may be filled in:

$$[J]=\begin{bmatrix} x_{ik} & y_{ik} \\ x_{jk} & y_{jk} \end{bmatrix} \tag{2.31}$$

Using Equations (2.30) and (2.31), a transformation is now available for expressing a set of differentials in one coordinate system in terms of another using only the nodal coordinates of the element. Equation (2.30) must be rearranged in order to express the Cartesian coordinate differentials in terms of the natural coordinate differentials:

$$\begin{Bmatrix} \frac{\partial}{\partial x} \\ \frac{\partial}{\partial y} \end{Bmatrix} = [J]^{-1} \begin{Bmatrix} \frac{\partial}{\partial \xi} \\ \frac{\partial}{\partial \eta} \end{Bmatrix} \tag{2.32}$$

where the inverse of the Jacobian matrix is expressed as

$$[J]^{-1} = \frac{1}{|\det[J]|} \begin{bmatrix} y_{jk} & -y_{ik} \\ -x_{jk} & x_{ik} \end{bmatrix} \tag{2.33}$$

$$\det[J] = x_{ik} y_{jk} - x_{kj} y_{ik} \tag{2.34}$$

Note that the determinant of $[J]$ is equal to twice the area of an arbitrary triangular element (A_e).[12,15] Thus,

$$2A_e = |\det[J]| \tag{2.35}$$

The determinant is a positive value only when the nodes i, j and k are set in a counterclockwise order around the element, where in that case the magnitude sign | | is not needed.

Combining Equations (2.32) through (2.35), it follows that

$$\begin{Bmatrix} \frac{\partial}{\partial x} \\ \frac{\partial}{\partial y} \end{Bmatrix} = \frac{1}{2A_e} \begin{Bmatrix} y_{jk}\frac{\partial}{\partial \xi} - y_{ik}\frac{\partial}{\partial \eta} \\ -x_{jk}\frac{\partial}{\partial \xi} + x_{ik}\frac{\partial}{\partial \eta} \end{Bmatrix} \tag{2.36}$$

Equation (2.36) presents the transformation required to change the Cartesian differentials in the differential operator $[L]$ into natural coordinate based differentials. Thus $[L]$ expressed in natural differentials is

$$[L] = \frac{1}{2A_e} \begin{bmatrix} y_{jk}\frac{\partial}{\partial \xi} - y_{ik}\frac{\partial}{\partial \eta} & 0 \\ 0 & -x_{jk}\frac{\partial}{\partial \xi} + x_{ik}\frac{\partial}{\partial \eta} \\ -x_{jk}\frac{\partial}{\partial \xi} + x_{ik}\frac{\partial}{\partial \eta} & y_{jk}\frac{\partial}{\partial \xi} - y_{ik}\frac{\partial}{\partial \eta} \end{bmatrix} \tag{2.37}$$

Combining the terms $[L]$ and $\lfloor N \rfloor$ from Equation (2.23) into a single term $\lfloor B \rfloor$ and using Equation (2.37) for $[L]$ results in (note that $x_{ij} = -x_{ji}$ and $y_{ij} = -y_{ji}$)

$$\lfloor B \rfloor = [L]\lfloor N \rfloor \tag{2.38}$$

$$= \frac{1}{2A_e} \begin{bmatrix} y_{jk}\frac{\partial}{\partial \xi} - y_{ik}\frac{\partial}{\partial \eta} & 0 \\ 0 & -x_{jk}\frac{\partial}{\partial \xi} + x_{ik}\frac{\partial}{\partial \eta} \\ -x_{jk}\frac{\partial}{\partial \xi} + x_{ik}\frac{\partial}{\partial \eta} & y_{jk}\frac{\partial}{\partial \xi} - y_{ik}\frac{\partial}{\partial \eta} \end{bmatrix} \begin{bmatrix} \xi & 0 & \eta & 0 & 1-\xi-\eta & 0 \\ 0 & \xi & 0 & \eta & 0 & 1-\xi-\eta \end{bmatrix}$$

$$\therefore \lfloor B \rfloor = \frac{1}{2A_e} \begin{bmatrix} y_{jk} & 0 & y_{ki} & 0 & y_{ij} & 0 \\ 0 & x_{kj} & 0 & x_{ik} & 0 & x_{ji} \\ x_{kj} & y_{jk} & x_{ik} & y_{ki} & x_{ji} & y_{ij} \end{bmatrix} \tag{2.39}$$

In order to properly evaluate the stiffness of the softer propellant, a nonlinear analysis is used where the elemental stiffness matrix in Equation (2.23) is augmented by the geometric changes in stiffness resulting from deformation. This is a large deformation analysis where the elements in matrix $\lfloor B \rfloor$ and the element area are calculated using the nodal coordinates plus the displacement vector. This is most efficiently implemented by updating the coordinate differences x_{kj}, x_{ik}, x_{ji}, y_{jk}, y_{ki} and y_{ij} in Equations (2.33), (2.34) and (2.39) with the current displacement vector. Therefore, $x_{ij} = (x_i + u_i) - (x_j + u_j)$ and $y_{ij} = (y_i + v_i) - (y_j + v_j)$ (and permutations thereof) are changed in Equations (2.33), (2.34) and (2.39).

Equation (2.39) may be substituted into Equation (2.23) to make

$$[k] = \iiint_V \lfloor B \rfloor^T [E] \lfloor B \rfloor dV \tag{2.40}$$

Note that $\lfloor N \rfloor^T [L]^T = \lfloor B \rfloor^T$. In Equation (2.40), the terms $\lfloor B \rfloor^T [E] \lfloor B \rfloor$ are all constants, therefore taking note that $\iiint_V dV = A_e t_e$, where t_e is the element thickness, Equation (2.40) becomes

$$[k] = A_e t_e \lfloor B \rfloor^T [E] \lfloor B \rfloor \tag{2.41}$$

Equation (2.41) is the final form of the stiffness matrix that is in a format suitable for computer coding and as there is no integration necessary, Equation (2.41) may be computed very quickly. Again, note that $\lfloor B \rfloor$ and A_e are functions of the displacement vector $\{q\}$, thus an iterative solution method would be suggested. However, this is not the case, as once transient terms are introduced, the solution method can be linearized (see Section 2.4.4).

Recall that Equation (2.10) expresses the strain inside the element. Upon combining Equations (2.10) and (2.16) results in

$$\{\varepsilon\}=[L][N]\{q\}=[B]\{q\} \tag{2.42}$$

when expanded through becomes,

$$\{\varepsilon\}=\frac{1}{2A_e}\begin{Bmatrix} y_{jk}q_1+y_{ki}q_3+y_{ij}q_5 \\ x_{kj}q_2+x_{ik}q_4+x_{ji}q_6 \\ y_{jk}q_1+x_{kj}q_2+y_{ki}q_3+x_{ik}q_4+y_{ij}q_5+x_{ji}q_6 \end{Bmatrix} \tag{2.43}$$

Note that the strain in Equation (2.43) is a vector that expresses the strain throughout the element. Having only x, y and shear terms in the vector, the strain in a three-node triangular element is constant throughout the element, hence the name Constant Strain Triangle (CST). The stresses inside the element abide by Equation (2.5), thus the elemental stresses are also constant. This is a consequence of the first order nature of the linear shape functions used. Looking at the external force component in Equation (2.22), Equation (2.24) does not use the differential operator $[L]$, thus no transformations are required. However, Equation (2.24) must still be integrated. Integration methods for integrating Equation (2.24) will be discussed in Section 2.4.5.

Once the stiffness matrix and externally applied force vector are evaluated for all the elements that define the system, Equation (2.22) is summed up for each element such that a *global* stiffness matrix, *global* displacement vector and *global* external force vector are assembled for the system:

$$[K]\leftarrow\sum_{elements}[k] \qquad \{Q\}\leftarrow\sum_{elements}\{q\} \qquad \{F\}\leftarrow\sum_{elements}\{f\}$$

$$\therefore[K]\{Q\}=\{F\} \tag{2.44}$$

Equation (2.44) ultimately defines the overall finite element system. Knowing the applied loads on the system, the nodal displacements $\{Q\}$ may be solved for, and the stresses and strains may then be solved. However, before this can happen, the necessary boundary conditions must be applied (see Section 2.4.5).

2.4.4 Transient Structural FE Derivations

The overall internal ballistic simulation model includes the effects of structural vibrations on the burning rate and combustion chamber wave development. Therefore, it is necessary to be able to model transient effects in the structure. Recall Newton's second law, $F=ma$. Taking the sum of the forces around any body will be equal to the net inertia of the body. Figure 2-17 illustrates the summing up of the forces of a one-dimensional system with a mass, spring and damper.[16,17]

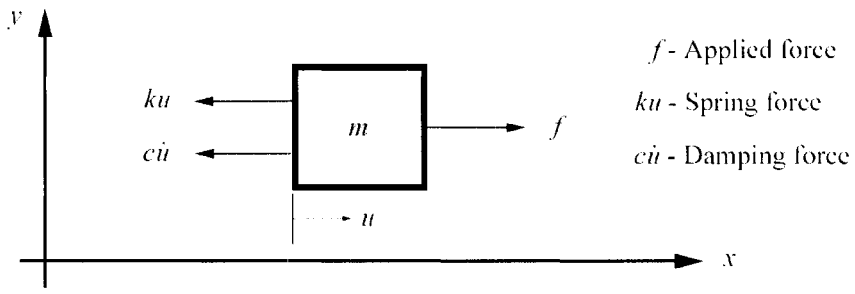


Figure 2-17 – Summation of the forces about a mass (m)

Thus using Newton's second law and using a conventional notation where the acceleration of the mass $a = \ddot{u} = \frac{d^2u}{dt^2}$, the sum of the forces results in

$$m\ddot{u} = f - ku - c\dot{u}$$

or,

$$m\ddot{u} + c\dot{u} + ku = f \tag{2.45}$$

Equation (2.45) expresses the motion of any body subjected to an external force (f), where $m\ddot{u}$ is the inertial force, $c\dot{u}$ is the damping force and ku is the spring force. The spring force in Equation (2.45) is the one-dimensional representation to the internal force presented in Section 2.4.3 and thus analogous to it. The inertial and damping forces in the structure due to its movement could be modelled in the same manner as the internal and external forces in the previous section using Galerkin's method of discretization:[11]

$$\text{Inertial force per unit volume} = \iiint_V \lfloor N \rfloor^T (\rho \{\ddot{u}\}) dV \tag{2.46}$$

$$\text{Damping force per unit volume} = \iiint_V \lfloor N \rfloor^T (c \{\dot{u}\}) dV \tag{2.47}$$

where c is an arbitrary damping coefficient per unit volume and $\{\dot{u}\}$ and $\{\ddot{u}\}$ are the first and second time derivatives of the displacement vector $\{u\}$. Note that $\{\dot{u}\}$ and $\{\ddot{u}\}$ may be expressed in terms of a nodal velocity and acceleration vector linearly interpolated over the element as done in Equation (2.16) such that,

$$\begin{Bmatrix} \dot{u}_i \\ \dot{v}_i \\ \dot{u}_j \\ \dot{v}_j \\ \dot{u}_k \\ \dot{v}_k \end{Bmatrix} = \begin{Bmatrix} \dot{q}_1 \\ \dot{q}_2 \\ \dot{q}_3 \\ \dot{q}_4 \\ \dot{q}_5 \\ \dot{q}_6 \end{Bmatrix} = \{\dot{q}\} \quad \text{and} \quad \begin{Bmatrix} \ddot{u}_i \\ \ddot{v}_i \\ \ddot{u}_j \\ \ddot{v}_j \\ \ddot{u}_k \\ \ddot{v}_k \end{Bmatrix} = \begin{Bmatrix} \ddot{q}_1 \\ \ddot{q}_2 \\ \ddot{q}_3 \\ \ddot{q}_4 \\ \ddot{q}_5 \\ \ddot{q}_6 \end{Bmatrix} = \{\ddot{q}\} \tag{2.48}$$

Thus,

$$\frac{d}{dt}\{u\} = \{\dot{u}\} = \begin{bmatrix} N_i & 0 & N_j & 0 & N_k & 0 \\ 0 & N_i & 0 & N_j & 0 & N_k \end{bmatrix} \{\dot{q}\} = \lfloor N \rfloor \{\dot{q}\} \tag{2.49}$$

$$\frac{d^2}{dt^2}\{u\}=\{\ddot{u}\}=\begin{bmatrix} N_i & 0 & N_j & 0 & N_k & 0 \\ 0 & N_i & 0 & N_j & 0 & N_k \end{bmatrix}\{\ddot{q}\}=\lfloor N \rfloor\{\ddot{q}\} \tag{2.50}$$

So, substituting Equations (2.49) and (2.50) into Equations (2.46) and (2.47) and factoring out the elemental velocity and acceleration vectors and other constants, the inertial and damping forces may be expressed as

$$\iiint_V \rho \lfloor N \rfloor^T \lfloor N \rfloor \{\ddot{q}\} dV = \rho \iiint_V \lfloor N \rfloor^T \lfloor N \rfloor dV \{\ddot{q}\} = [m]\{\ddot{q}\} \tag{2.51}$$

$$\iiint_V c \lfloor N \rfloor^T \lfloor N \rfloor \{\dot{q}\} dV = c \iiint_V \lfloor N \rfloor^T \lfloor N \rfloor dV \{\dot{q}\} = [c]\{\dot{q}\} \tag{2.52}$$

where $[m]$ and $[c]$ represent the mass and damping matrices of the element. These two matrices will be elaborated upon further in this section. Equations (2.51) and (2.52) can be added to Equation (2.22) which results in an expression analogous to Equation (2.45):

$$[m]\{\ddot{q}\}+[c]\{\dot{q}\}+[k]\{q\}=\{f\} \tag{2.53}$$

Equation (2.53) governs the dynamic response of the element.

To find the dynamic response of the <u>whole</u> system, the mass and damping matrices as well as the velocity and acceleration vectors must be assembled into *global* matrices and vectors as was done for Equation (2.44):

$$[M] \leftarrow \sum_{elements}[m] \qquad \{\ddot{Q}\} \leftarrow \sum_{elements}\{\ddot{q}\} \qquad [C] \leftarrow \sum_{elements}[c] \qquad \{\dot{Q}\} \leftarrow \sum_{elements}\{\dot{q}\}$$

So, recalling Equation (2.44), Equation (2.53) can be restated using global matrices and vectors, which represents the whole system:

$$\therefore [M]\{\ddot{Q}\}+[C]\{\dot{Q}\}+[K]\{Q\}=\{F\} \tag{2.54}$$

The dynamic response of the motor as defined by Equation (2.54) must be integrated through time to produce a solution. A commonly used integration method is the Newmark method. The Newmark method involves using a set of relations that approximate the time derivatives of the global displacement vector $\{Q\}$ to find displacement and velocity values at the next time step:[11]

$$\{Q\}_{t+\Delta t} = \{Q\}_t + \Delta t\{\dot{Q}\}_t + \frac{\Delta t^2}{2}\left[(1-2\beta)\{\ddot{Q}\}_t + 2\beta\{\ddot{Q}\}_{t+\Delta t}\right] \tag{2.55a}$$

$$\{\dot{Q}\}_{t+\Delta t} = \{\dot{Q}\}_t + \Delta t\left[(1-\gamma)\{\ddot{Q}\}_t + \gamma\{\ddot{Q}\}_{t+\Delta t}\right] \tag{2.55b}$$

where Δt is the time step, γ is the amount of numerical dissipation or damping, and β helps control the stability of the algorithm and determines if the solution is implicit or explicit. For a working algorithm, β must be ≥ 0 and γ must be ≥ 0.5. If $\gamma = 0.5$, second order accuracy of the algorithm is assured. If $\gamma > 0.5$, there will be less noise in the solution, but only first order accuracy can be assured. A similar, more general method called the α – method, can ensure second order accuracy even with $\gamma > 0.5$ but, requires more computer time per time step.[11] Higher order methods will be briefly discussed further in this section.

The acceleration is found from Equation (2.54) at time $t + \Delta t$, knowing $\{Q\}_{t+\Delta t}$ and $\{\dot{Q}\}_{t+\Delta t}$ from Equations (2.55a) and (2.55b):

$$[M]\{\ddot{Q}\}_{t+\Delta t} + [C]\left(\{\dot{Q}\}_t + \Delta t\left[(1-\gamma)\{\ddot{Q}\}_t + \gamma\{\ddot{Q}\}_{t+\Delta t}\right]\right) + \tag{2.56}$$

$$[K]\left(\{Q\}_t + \Delta t\{\dot{Q}\}_t + \frac{\Delta t^2}{2}\left[(1-2\beta)\{\ddot{Q}\}_t + 2\beta\{\ddot{Q}\}_{t+\Delta t}\right]\right) = \{F\}$$

Factoring terms,

$$\{\ddot{Q}\}_{t+\Delta t} = \left(\Delta t^2\beta[K] + \Delta t\gamma[C] + [M]\right)^{-1} \times \tag{2.57}$$

$$\left(\{F\}_{t+\Delta t} - [K]\left(\{Q\}_t + \Delta t\{\dot{Q}\}_t + \frac{\Delta t^2}{2}(1-2\beta)\{\ddot{Q}\}_t\right) - [C]\left(\{\dot{Q}\}_t + \Delta t(1-\gamma)\{\ddot{Q}\}_t\right)\right)$$

This method requires a rather large square matrix to be inverted, requiring a considerable amount of computer power per time step. Considering this, and fact that the internal ballistic flow calculations (Chapter 3) are explicit and already use a small time step, the simplest thing to do is to use an explicit method for the transient FE analysis as well. To do this with the Newmark method, β is set to zero, therefore Equation (2.57) becomes,[11]

$$\{\ddot{Q}\}_{t+\Delta t} = \left(\Delta t\gamma[C] + [M]\right)^{-1} \times$$

$$\left(\{F\}_{t+\Delta t} - [K]\left(\{Q\}_t + \Delta t\{\dot{Q}\}_t + \frac{\Delta t^2}{2}\{\ddot{Q}\}_t\right) - [C]\left(\{\dot{Q}\}_t + \Delta t(1-\gamma)\{\ddot{Q}\}_t\right)\right) \tag{2.58}$$

The difference between the externally applied force $\{F\}_{t+\Delta t}$, and the internal force $[K]\left(\{Q\}_t + \Delta t\{\dot{Q}\}_t + \frac{\Delta t^2}{2}\{\ddot{Q}\}_t\right)$, is to be combined into one term labelled a "residual force" $\{R\}_{t+\Delta t}$ – to reduce the size of Equation (2.58):

$$\{R\}_{t+\Delta t} = \{F\}_{t+\Delta t} - [K]\left(\{Q\}_t + \Delta t\{\dot{Q}\}_t + \frac{\Delta t^2}{2}\{\ddot{Q}\}_t\right) = \{F\}_{t+\Delta t} - \{F_{in}\}_{t+\Delta t} \tag{2.59}$$

Therefore, the Newmark method becomes essentially the same as the central difference method,[11]

$$\{\ddot{Q}\}_{t+\Delta t} = (\Delta t\gamma[C]+[M])^{-1}(\{R\}_{t+\Delta t} - [C](\{\dot{Q}\}_t + \Delta t(1-\gamma)\{\ddot{Q}\}_t)) \tag{2.60}$$

However, in Equation (2.60), unless the mass and damping matrices are diagonal in their topology (a diagonal matrix is defined as a square matrix whose entries are zero everywhere except along the principal diagonal), a large square matrix must still be inverted. If the mass and damping matrices are diagonal, inverting these matrices is a trivial operation, and solving for $\{\ddot{Q}\}_{t+\Delta t}$ becomes simple vector algebra. If Equations (2.51) and (2.52) are used in the creation of the mass and damping matrices, then these matrices will be symmetric but not diagonal. In that case, the matrices are called "consistent" matrices – being consistent with the derivation procedure of the stiffness matrix and external force vector as in the previous section using the same shape function. In order to create diagonal mass and damping matrices, a technique called lumping is used.

The process of lumping the mass matrix for any one element is straightforward. In its simplest form used in this simulation, the total mass of an element is divided by the number of nodes, and that value is assigned to each degree of freedom in the element (note: this method of lumping is only valid for three-node triangular elements):[11]

$$[m] = \frac{\rho A_e t_e}{3}\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} = \frac{\rho A_e t_e}{3}[1 \quad 1 \quad 1 \quad 1 \quad 1 \quad 1]_D \tag{2.61}$$

Equation (2.61) is used instead of the expression for $[m]$ in Equation (2.51). Mass lumping in general is not necessarily less accurate than using consistent matrices in any respect; in some cases it provides better accuracy depending on the system.[11] Trial FE simulations were run to compare the results using consistent and lumped mass matrices. The differences in the results for a cylindrical-grain motor were negligible. The primary difference is that consistent mass matrices required far more computer time, because the mass matrix is no longer diagonal. In this simulation model, mass lumping is used on the sole basis of simplifying the integration and increasing the computational efficiency.

Damping in the motor structure, as mentioned in Section 2.4.1, in this simulation is modelled as viscous in effect (proportional to velocity). It is based on an FE device called *proportional damping* or *Rayleigh damping*. Proportional damping defines the damping matrix as a linear combination of the mass and stiffness matrices. In this simulation model however, the damping matrix is made proportional only to the mass matrix (i.e. mass proportional) for simplicity. Drawing from a single degree of freedom system as illustrated in Figure 2-17 as an analogy, the damping coefficient may be expressed as[16]

$$c = 2\zeta\omega_n m \tag{2.62}$$

where ζ is the damping ratio and ω_n is the resonant frequency of the structure ($\omega_n = 2\pi f_n$). The damping ratio is a variable set to control the amount and behaviour of damping in the FE structure. The resonant frequency is the dominant fundamental frequency of the structure and is discussed and calculated in Section 2.5. Note that the damping coefficient in Equation (2.62) is a function of mass as well. Thus if m is replaced with the elemental mass matrix $[m]$, Equation (2.62) becomes the elemental damping matrix, which is simply the mass portion of the Rayleigh damping device:[11]

$$[c] = 2\zeta\omega_n[m] \tag{2.63}$$

If the mass matrix is diagonal, then the damping matrix will be diagonal as well using Equation (2.63) instead of the expression for [c] in Equation (2.52). Having a diagonal mass and damping matrix, the integration in Equation (2.60) is simplified. Again, note that the natural frequency and damping ratio apply to the entire structure and does not vary from element to element. In addition, the resonant frequency is kept constant throughout the simulation, even though the propellant is burning away. It is not practical to re-compute the resonant frequency as the propellant regresses, as the resonant frequency analysis is time consuming.

Having a damping matrix as a function of the mass matrix further simplifies the application of Equation (2.60). Both the lumped mass and damping matrices are assembled into global matrices in the same fashion as the consistent form of these matrices. Using the lumped form of the mass and damping matrices, Equation (2.60) reduces to

$$\{\ddot{Q}\}_{t+\Delta t} = \frac{1}{2\Delta t\gamma\zeta\omega_n + 1}[M]^{-1}\left(\{R\}_{t+\Delta t} - 2\zeta\omega_n[M]\left(\{\dot{Q}\}_t + \Delta t(1-\gamma)\{\ddot{Q}\}_t\right)\right) \quad (2.64)$$

Thus, the only matrix that must be inverted is the mass matrix, and this is a simple task if the mass matrix is diagonal (the diagonal entries are inverted). So restating Equation (2.55a), Equation (2.55b) and Equation (2.60), one finds the displacement, velocity and acceleration respectively via:

$$\{Q\}_{t+\Delta t} = \{Q\}_t + \Delta t\{\dot{Q}\}_t + \frac{\Delta t^2}{2}\{\ddot{Q}\}_t \quad (2.65a)$$

$$\{\dot{Q}\}_{t+\Delta t} = \{\dot{Q}\}_t + \Delta t\left[(1-\gamma)\{\ddot{Q}\}_t + \gamma\{\ddot{Q}\}_{t+\Delta t}\right] \quad (2.65b)$$

$$\{\ddot{Q}\}_{t+\Delta t} = \frac{1}{2\Delta t\gamma\zeta\omega_n + 1}[M]^{-1}\left(\{F\}_{t+\Delta t} - \{F_{in}\}_{t+\Delta t} - 2\zeta\omega_n[M]\left(\{\dot{Q}\}_t + \Delta t(1-\gamma)\{\ddot{Q}\}_t\right)\right) \quad (2.65c)$$

The internal force component $\{F_{\text{int}}\}$ in Equation (2.65c), which is part of the residual force described by Equation (2.59), contains the only non-diagonal (symmetric) square matrix $[K]$ that needs to be assembled and stored in computer memory. However, this is not the case. At the beginning of every new time step, the global displacement vector $\{Q\}$ is updated using Equation (2.65a). Instead of multiplying the global terms, $[K]\{Q\}=\{F_{in}\}$, it is more computationally efficient to start with the elemental displacement components $\{q\}$ and compute the internal force component $[k]\{q\}=\{f_{\text{int}}\}$ for the new time step on an element-by-element level. This also has the effect of not calculating any zero terms, which would normally litter the stiffness matrix. Then, the *global* internal force vector may be assembled and stored. This method relieves the computer's memory of having to store the large n x n global stiffness matrix.

Since the stiffness matrix is a function of the displacement (using a large deformation analysis as discussed in Section 2.4.3), the elemental stiffness matrices must be recalculated every time step. Doing so will create some computational overhead, but will provide a new stiffness matrix for each time step; therefore, the solution does not need to be iterated each time step as the displacement vector for the next time step is known using Equation (2.65a). Thus, the transient solution procedure has the effect of linearizing the solution of the internal force vector, thus simplifying the solution process.

Of quick note at this point is that there are algorithms that guarantee higher order accuracy such as the second order α – method and fourth order Runge – Kutta methods. Each of these methods was instigated in the early development of the FE solver to see how they affected the accuracy of the FE output. The fourth order Runge – Kutta method as one would expect produced good results however at the cost of a very small time step (10^{-2} of the Newmark method). The second order α – method is essentially a generalized form of the Newmark method. The α – method, produced results having little or no difference to the output from the Newmark method – even

with a large amount of algorithmic damping. The Newmark method produced good results with a minimum of computer power required and thus was chosen for this simulation.

Explicit methods, unlike implicit methods, are only conditionally stable. This stability criterion is dependant on the highest natural frequency of the structure, which is a function of the maximum speed at which information may travel through any point in the system. This criterion is commonly called the Courant-Friedrichs-Lewy (CFL) condition. This speed is associated with the sound speed of the materials in the system. The sound speed (a) through a solid is expressed as, [11]

$$a = \sqrt{\frac{E}{\rho(1-\upsilon^2)}} \tag{2.66}$$

with material properties υ, ρ and E. Thus, the maximum allowable time step the FE solution may be stably integrated with is as follows:

$$\Delta t_{\max_{FE}} \leq C_{n_{FE}} \frac{\Delta l_{\min}}{a_{\max}} \quad \text{, where } C_{n_{FE}} \leq 1 \tag{2.67}$$

In Equation (2.67), $\Delta t_{\max_{FE}}$ is the time it takes a wave of speed $a_{\max}$ to travel distance $\Delta l_{\min}$. So, to find the maximum allowable time step that Equations (2.65a), (2.65b) and (2.65c) may be integrated in stable manner, it is necessary to find the smallest element and determine which side is the smallest. The shorter of either the x or y component of the length of that side determines $\Delta l_{\min}$. Then the sound speed must be evaluated for each material and the largest value taken as $a_{\max}$. Knowing $\Delta l_{\min}$ and $a_{\max}$, $\Delta t_{\max_{FE}}$ may be found for the system. $C_{n_{FE}}$ is the Courant number for the FE calculations, which scales down the time step, as running the integration routines at the maximum allowable time step has the general tendency of blowing up the simulation under certain circumstances. The Courant number is set to the highest possible value that permits a

stable run, as the closer to $\Delta t_{\max_{FE}}$ the simulation is run, the more accuracy is maintained and less computer time is needed. Further discussion on the time step of the simulation may be found in Section 4.3.

2.4.5 FE Boundary Conditions

The boundary conditions for the FE calculations come from an analysis of the physical system that is being simulated. A section of the motor is displayed in Figure 2-18 below. The most significant boundary condition for the simulation is that of the internal chamber pressure. It is of course of prime importance, as pressure wave development inside the motor will affect the motor structure. Other boundary conditions include the external pressure (which reduces with altitude if simulating a flight case), centrifugal forces due to the motor spinning (either in flight or on a test stand) and a physical constraint to prevent lateral rigid-body motion. Propellant regression is not a required boundary condition for FE calculations and will be discussed in Section 2.7.

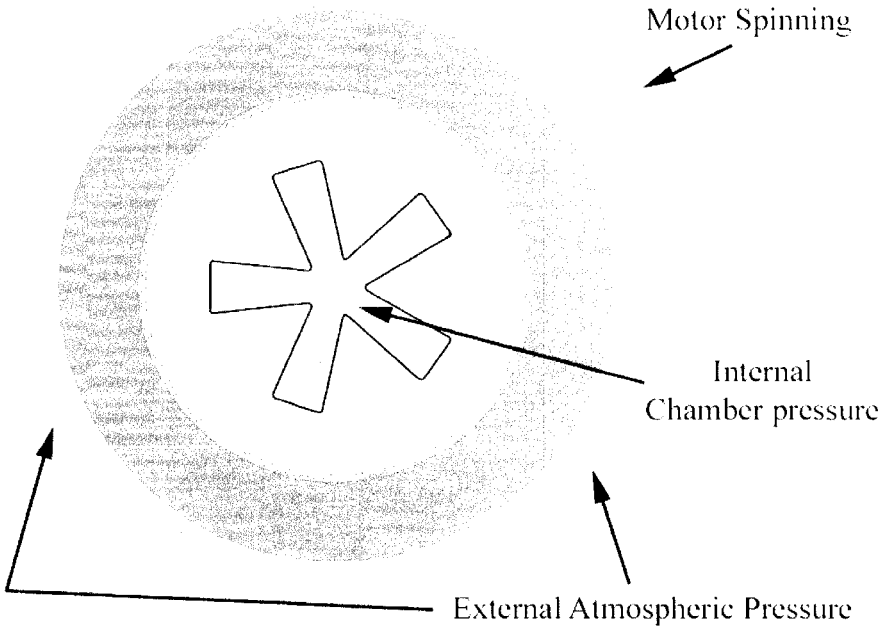


Figure 2-18 – Applied forces affecting the structure

All pressure forces are gauge pressures with respect to standard sea level pressure, in order to avoid initial stresses and strains in the system. Therefore, if the external pressure is not changing, it should be set to zero. The traction forces in Equation (2.24) in this simulation consist only of the pressure forces. Viscous forces acting along the boundaries due to the motion of the gas are neglected, and are in general of a much lower magnitude than the pressure forces. In addition, the pressure does not vary along the boundary due to the one-dimensional internal flow calculations; only one pressure value is assigned to any one motor section, so the pressure is constant from one element to the next along the pressure boundary for any one section.

Looking at the traction term in Equation (2.24), $\{T\}$ represents the x and y components of pressure acting along a surface such that $\{T\}=\{p_x \quad p_y\}^T$. Noting that $dS = t_e dl$ for a two-dimensional analysis, the traction term is written as

$$\iint_S [N]^T \{T\} dS = t_e \int_l \begin{bmatrix} N_i & 0 \\ 0 & N_i \\ N_j & 0 \\ 0 & N_j \\ N_k & 0 \\ 0 & N_k \end{bmatrix} \begin{Bmatrix} p_x \\ p_y \end{Bmatrix} dl \tag{2.68}$$

In most cases, there are pressure forces acting only along one side of the element. If the pressure force acts along side *i-j*, Equation (2.68) becomes

$$\iint_S [N]^T \{T\} dS = t_e \int_l \begin{bmatrix} N_i & 0 \\ 0 & N_i \\ N_j & 0 \\ 0 & N_j \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \begin{Bmatrix} p_x \\ p_y \end{Bmatrix} dl \tag{2.69}$$

The shape function relating the node that isn't a part of the affected side is dropped from Equation (2.68). If the pressure force acts along a different side, Equation (2.69) is permutated such that the necessary nodes are included. The extrapolation of the x and y components of pressure force (p) is illustrated in Figure 2-19 and expressed as follows:[11,12]

$$p_x = -p\sin\theta \tag{2.70a}$$

$$p_y = -p\cos\theta \tag{2.70b}$$

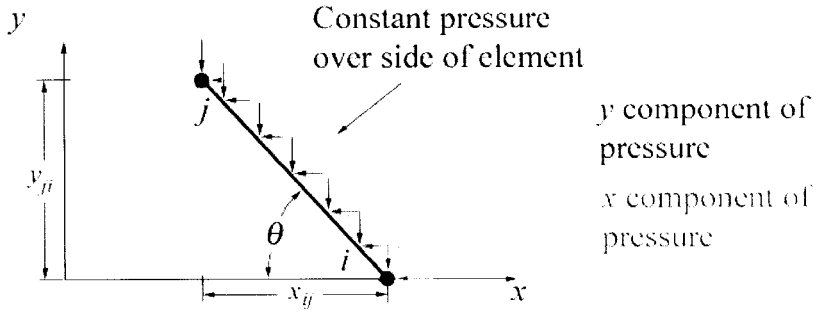


Figure 2-19 – Pressure force components

The trigonometric terms above may be represented using ratios based on the geometry of the element:

$$\sin\theta = \frac{y_{ji}}{l_{ij}} \tag{2.71a}$$

$$\cos\theta = \frac{x_{ij}}{l_{ij}} \tag{2.71b}$$

where l_{ij} is the length of the side i-j. Substituting Equations (2.71a) and (2.71b) into Equations (2.70a) and (2.70b), and recalling that $x_{ij} = -x_{ji}$ and $y_{ji} = -y_{ij}$, result in the traction force vector:

$$\{T\} = \begin{Bmatrix} p_x \\ p_y \end{Bmatrix} = \frac{p}{l_{ij}} \begin{Bmatrix} y_{ij} \\ x_{ji} \end{Bmatrix} \tag{2.72}$$

Equation (2.72) represents the total traction forces in this simulation. Note that all the terms in Equation (2.72) are constants and may be factored out of the integral in Equation (2.69) such that

$$\iint_S \lfloor N \rfloor^T \{T\} dS = t_e \int_l \begin{bmatrix} N_i & 0 \\ 0 & N_i \\ N_j & 0 \\ 0 & N_j \\ 0 & 0 \\ 0 & 0 \end{bmatrix} dl \begin{Bmatrix} p_x \\ p_y \end{Bmatrix} \tag{2.73}$$

The remaining shape functions in the integral may be integrated using a simple method appropriate for integrating linear shape functions:[14]

$$\int_l N_i^\alpha N_j^\beta dl = \frac{\alpha! \beta!}{(\alpha + \beta + 1)!} l_{ij} \tag{2.74}$$

where l_{ij} is the distance between nodes i and j, and α and β (for Equation (2.74) only) are the exponents of the shape functions and must be positive integers. Integrating Equation (2.73) using Equation (2.74) and multiplying through all the terms results in

$$\iint_S \lfloor N \rfloor^T \{T\} dS = t_e \frac{l_{ij}}{2} \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 1 & 0 \\ 0 & 1 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \begin{Bmatrix} p_x \\ p_y \end{Bmatrix} \tag{2.75}$$

$$\iint_S \lfloor N \rfloor^T \{T\} dS = \frac{t_e p}{2} \begin{Bmatrix} y_{ij} & x_{ji} & y_{ij} & x_{ji} & 0 & 0 \end{Bmatrix}^T \tag{2.76}$$

Equation (2.76) is the final useable form of the traction component of Equation (2.24). Again, if sides of the element other than *i-j* are exposed to a pressure force, Equation (2.67) may be permutated to accommodate the necessary side.

The only potential body forces experienced by the system are that of the centrifugal forces due to the motor being spun either on the test stand or in flight. Gravity forces are neglected as mentioned in Section 2.4.1 and are of a negligible magnitude in comparison to the centrifugal forces in either case. The centrifugal body forces per unit volume (f_c) experienced at any point is expressed as,

$$f_c = \rho\omega^2 R \tag{2.77}$$

where ρ is the density of the material, ω is the angular velocity of the motor and R is the distance of node n from the center of rotation. The body force vector in Equation (2.24) requires the body forces to be expressed in x and y components. Therefore, Equation (2.77) is broken up into its components by breaking up R into its x and y terms (see Figure 2-20).

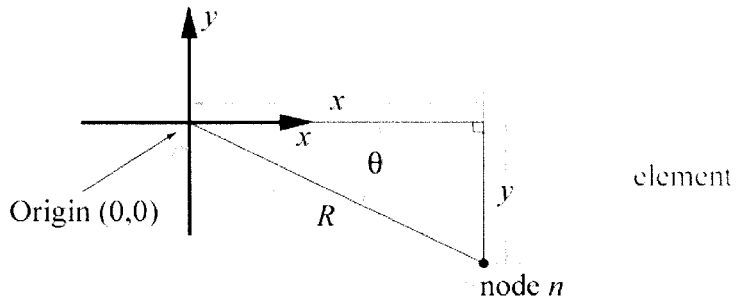


Figure 2-20 – Body force components

The center of rotation in this simulation is at $x = y = 0$, thus using trigonometric relations similar to Equations (2.71a) and (7.51b), one gets

$$R\sin\theta = x_n \tag{2.78a}$$

$$R\cos\theta = y_n \tag{2.78b}$$

where x_n and y_n are the x and y coordinates of a particular node. Therefore, breaking up Equation (2.77) into its component terms using Equations (2.78a) and (2.78b), the body force vector is expressed as

$$\{f_b\} = \begin{Bmatrix} f_{bx} \\ f_{by} \end{Bmatrix} = \rho\omega^2 \begin{Bmatrix} R\sin\theta \\ R\cos\theta \end{Bmatrix} = \rho\omega^2 \begin{Bmatrix} x_n \\ y_n \end{Bmatrix} \tag{2.79}$$

Substituting Equation (2.79) into the body force term in Equation (2.24), it is evident that the body force terms may all be factored out, as they are constants:

$$\iiint_V [N]^T \{f_b\} dV = \rho\omega^2 \iiint_V [N]^T dV \begin{Bmatrix} x_n \\ y_n \end{Bmatrix} \tag{2.80}$$

The remaining terms to be integrated in Equation (2.80) are the shape functions. This is done in a manner similar to what was done with the traction force integration. However, the integration is now done over the volume of the element and not the boundary surface area. This also means that all three shape functions are integrated, instead of only two as in the traction component. Therefore, Equation (2.74) is not quite applicable in this case, however a different variant may be used:[14]

$$\iint_S N_i^\alpha N_j^\beta N_k^\gamma dS = \frac{\alpha!\beta!\gamma!}{(\alpha+\beta+\gamma+2)!} 2A_e \tag{2.81}$$

where A_e is the area bounded by nodes i, j and k (area of the element) and α, β and γ (for Equation (2.81) only) are the exponents of the shape functions and must be positive integers. Noting that for a two-dimensional analysis $dV = t_e dS$, Equation (2.80) is integrated using Equation (2.81) as follows:

$$\rho\omega^2 \iiint_V [N]^T dV \begin{Bmatrix} x_n \\ y_n \end{Bmatrix} = \rho\omega^2 t_e \iint_S \begin{bmatrix} N_i & 0 \\ 0 & N_i \\ N_j & 0 \\ 0 & N_j \\ N_k & 0 \\ 0 & N_k \end{bmatrix} dS \begin{Bmatrix} x_n \\ y_n \end{Bmatrix} \tag{2.82}$$

$$\therefore \iiint_V [N]^T \{f_b\} dV = \rho\omega^2 t_e \frac{A_e}{3} \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 1 & 0 \\ 0 & 1 \\ 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{Bmatrix} x_n \\ y_n \end{Bmatrix} \tag{2.83}$$

Multiplying through,

$$\therefore \iiint_V [N]^T \{f_b\} dV = \frac{\rho\omega^2 t_e A_e}{3} \{x_i \quad y_i \quad x_j \quad y_j \quad x_k \quad y_k\}^T \tag{2.84}$$

Equation (2.84) represents the final usable form of the body force term in Equation (2.24). However, there are other considerations to think about when dealing with spinning structures such as stress stiffening or spin softening, or the introduction of Coriolis forces into the solution. These are not considered here since spinning of the SRM is not a concern of this thesis. The centrifugal body force term presented in Equation (2.84) is done so for a certain level of completeness in the applied force term, and to make the reader somewhat aware of its implementation. If spinning is to be used in the FE solution, the above-mentioned factors will have to be taken into account in addition to Equation (2.84).

Thus, by combining Equations (2.76) and (2.84), Equation (2.24) then becomes

$$\{f\}=\frac{t_e p}{2}\begin{Bmatrix} y_{ij} \\ x_{ji} \\ y_{ij} \\ x_{ji} \\ 0 \\ 0 \end{Bmatrix}+\frac{\rho\omega^2 t_e A_e}{3}\begin{Bmatrix} x_i \\ y_i \\ x_j \\ y_j \\ x_k \\ y_k \end{Bmatrix} \tag{2.85}$$

In Equation (2.85), the only time varying term is the pressure (p). However, the propellant is a rubbery-like material that under the high-pressure environment of the combustion chamber will undergo a much larger deformation than the metals in the casing or the sleeve. Nonlinear corrections applicable to explicit methods are made to the external force terms. The nodal coordinates for the elements for the most part do not change (except for those along the burning surface). However, the larger deformations will increase (or decrease) the length of the element sides, which are exposed to the chamber pressure, and move the nodes a little further from the rotational center increasing the centrifugal force they experience. To account for large deformations, the nodal coordinates are updated with the displacements of the nodes for the current time step, such that Equation (2.85) becomes,

$$\{f\}=\frac{t_e p}{2}\left(\begin{Bmatrix} y_{ij} \\ x_{ji} \\ y_{ij} \\ x_{ji} \\ 0 \\ 0 \end{Bmatrix}+\begin{Bmatrix} u_{ij} \\ v_{ji} \\ u_{ij} \\ v_{ji} \\ 0 \\ 0 \end{Bmatrix}\right)+\frac{\rho\omega^2 t_e A_e}{3}\left(\begin{Bmatrix} x_i \\ y_i \\ x_j \\ y_j \\ x_k \\ y_k \end{Bmatrix}+\begin{Bmatrix} u_i \\ v_i \\ u_j \\ v_j \\ u_k \\ v_k \end{Bmatrix}\right) \tag{2.86}$$

where $u_{ij}=u_i-u_j$ and $v_{ji}=v_j-v_i$. Equation (2.86) provides a correction for the external force for any large geometric deformations in the solution.

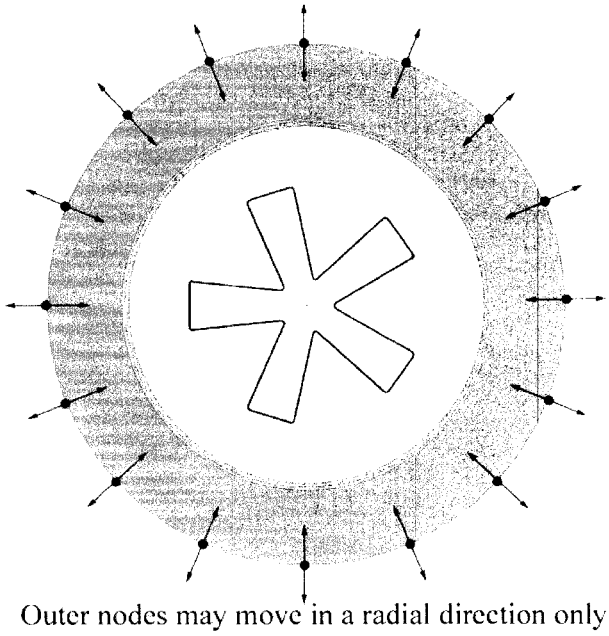


Figure 2-21 – Radial constraint on a full mesh

The only other boundary conditions used for the FE analysis in this simulation are the physical constraints used to prevent lateral rigid body motion. This physical constraint allows movement of certain nodes only in the radial direction. In the case where a complete grid is used, the nodes along the outer most boundary of the motor (the exposed part of the steel sleeve) are constrained to radial movement only (see Figure 2-21). In the case when symmetry is exploited, only the nodes that lie along the symmetry lines are constrained to move radially (see Figure 2-22). Note that only symmetric vibration modes can be captured if geometric symmetry is used.

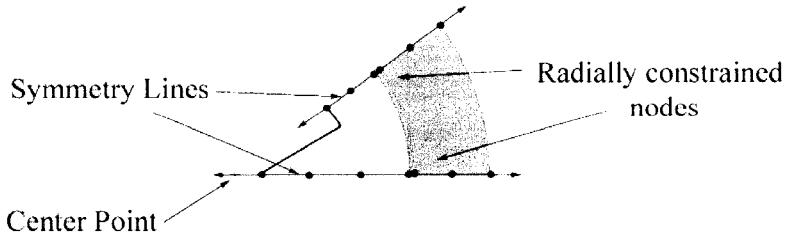


Figure 2-22 – Radial constraint on a partial mesh

After the calculations for one time step are completed, the nodes along a symmetry line – or in the case of a full grid – the nodes along the outer boundary of the mesh, must be repositioned in such a way that any tangential displacement is set to zero. This is done by first calculating the angle (θ) the node lies at with respect to some datum, which in Figure 2-23 is the positive x-axis. Then, the displacement vector for the node is transformed from Cartesian coordinates (x,y) into cylindrical coordinates (r,θ) to find the radial and tangential components of the nodal displacement:

$$\begin{Bmatrix} q_r \\ q_\theta \end{Bmatrix} = \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix} \begin{Bmatrix} q_x \\ q_y \end{Bmatrix} \tag{2.87}$$

where q_r is the radial component and q_θ the tangential component of the displacement of a node; q_x and q_y are the Cartesian components.

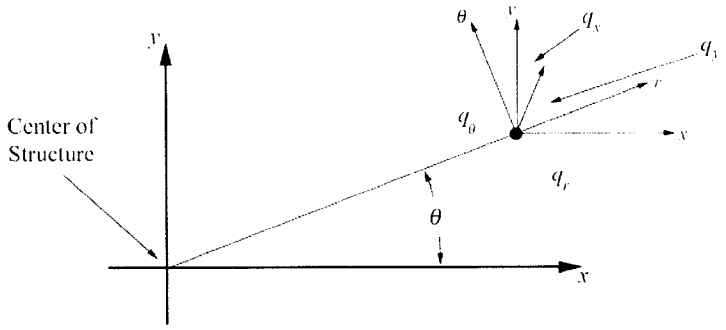


Figure 2-23 – Coordinate transformation of displacement vector

As this constraint restricts the displacement of the node to the radial direction only, the tangential component (q_θ) must be set to zero. Knowing the nodal displacement in the radial direction (q_r), the Cartesian components of the constrained nodal displacement may be extracted:

$$\begin{Bmatrix} q_r \\ 0 \end{Bmatrix} = \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix} \begin{Bmatrix} q'_x \\ q'_y \end{Bmatrix} \tag{2.88}$$

where q'_x and q'_y are the updated constrained nodal displacement components in Cartesian coordinates. Solving for q'_x and q'_y gives

$$\begin{Bmatrix} q'_x \\ q'_y \end{Bmatrix} = \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix}^{-1} \begin{Bmatrix} q_r \\ 0 \end{Bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \begin{Bmatrix} q_r \\ 0 \end{Bmatrix} \tag{2.89}$$

$$\therefore \begin{Bmatrix} q'_x \\ q'_y \end{Bmatrix} = \begin{Bmatrix} \cos\theta \\ \sin\theta \end{Bmatrix} q_r \tag{2.90}$$

Substituting in q_r from Equation (2.87) the above becomes,

$$\begin{Bmatrix} q'_x \\ q'_y \end{Bmatrix} = \begin{Bmatrix} q_x \cos^2\theta + q_y \sin\theta\cos\theta \\ q_x \sin\theta\cos\theta + q_y \sin^2\theta \end{Bmatrix} = \begin{bmatrix} \cos^2\theta & \sin\theta\cos\theta \\ \sin\theta\cos\theta & \sin^2\theta \end{bmatrix} \begin{Bmatrix} q_x \\ q_y \end{Bmatrix} \tag{2.91}$$

Thus, Equation (2.91) provides the necessary transformation to constrain the required nodes such that they are able to have displacements only in the radial direction. This transformation is introduced into the calculations after every time step in order to keep the structure properly constrained, and to prevent it from having any lateral rigid-body motion.

2.5 Harmonic Testing and the Natural Frequency

In order to use the damping model for the transient FE calculations, the resonant frequency of the whole structure must be known. The resonant frequency of a system is usually defined as the natural frequency that produces the most prominent response of that system. To find the natural frequencies of the system, an eigenvalue analysis is done. To find the dominant natural frequency, a harmonic test of the system is made. This is done by making a transient analysis of an undamped system (without damping, all the degrees of freedom in the structure move in phase with one another at the same frequency), which undergoes forced vibration at several specified frequencies (typically the natural frequencies of the system). The dominant natural frequency will be the

forcing frequency that produces the most significant displacements. Eigenvalue analysis and harmonic testing are presented in this section in that order.

Making the damping matrix and the external force vector zero, Equation (2.54) becomes,

$$[M]\{\ddot{Q}\}+[K]\{Q\}=0 \tag{2.92}$$

The steady-state solution of Equation (2.92) is $\{Q\}=0$. However, in a state of free vibration after an initial disturbance, $\{Q\}$ represents excursions from the steady-state solution. Vibratory motion consists of nodal displacements that vary sinusoidally with time relative to some equilibrium displacement, which in this case is zero. Thus, the displacement vector $\{Q\}$ and acceleration vector $\{\ddot{Q}\}$ may be represented as[16]

$$\{Q\}=\{\overline{Q}\}\sin\omega t \tag{2.93a}$$

and

$$\{\ddot{Q}\}=-\omega^2\{\overline{Q}\}\sin\omega t \tag{2.93b}$$

where $\{\overline{Q}\}$ is the maximum displacement due to vibration. Substituting Equations (2.93a) and (2.93b) into Equation (2.92) results in the eigenproblem,[11,12]

$$-\omega^2[M]\{\overline{Q}\}\sin\omega t+[K]\{\overline{Q}\}\sin\omega t=0 \tag{2.94}$$

$$\therefore([K]-\omega^2[M])\{\overline{Q}\}=0 \tag{2.95}$$

where ω^2 is an eigenvalue of the system, and ω is the associated natural frequency. In order for the eigenvector $\{\overline{Q}\}$ in Equation (2.95) to be nontrivial, the required condition is that

$$\det\left|[K]-\omega^2[M]\right|=0 \tag{2.96}$$

Simplifying where $[D]=[M]^{-1}[K]$,

$$\det\left|[D]-\omega^2[I]\right|=0 \tag{2.97}$$

$[D]$ is termed the "dynamic" matrix and $[I]$ is an identity matrix. In order to keep $[D]$ symmetric, a congruence transformation is applied where $[M]$ is decomposed such that $[M]=[L][L]^T$ (if $[M]$ is diagonal, then the terms in $[L]$ are found by taking the square root of the corresponding terms in $[M]$).[15] As a result, the dynamic matrix can be evaluated as:

$$[D]=[L]^{-1}[K][L]^{-1^T} \tag{2.98}$$

Equation (2.97) represents a standard eigenvalue problem, thus the eigenvalues of the system may be extracted using a suitable algorithm.

There are as many natural frequencies as there are degrees of freedom in the system. Finding the characteristic polynomial to extract the eigenvalues is the common practice, but is quite impractical in larger FE systems as well as inaccurate due to the inefficient root extraction methods for higher order polynomials. The only practical methods for larger FE systems are either a vector iteration method or a transformation method. In this thesis, a transformation method is used to extract the eigenvalues. Transformation methods are efficient at extracting all the eigenvalues in a system. Although only some of the lowest eigenvalues are required, the ready availability of computer algorithms for the transformation method available to the author made transformation methods the best choice.

The basic approach in transformation methods is to transform the dynamic matrix into a simpler form, and then determine the eigenvalues. A common method in use for large systems is the QL method. This method uses Householder matrix transformations to first reduce the dynamic matrix to a tri-diagonal form, and then the QL method is used to determine the eigenvalues. Householder's method is used typically to find a symmetric tri-diagonal matrix that is similar to a given symmetric matrix. The QL method will extract the eigenvalues of the dynamic matrix without using Householder's transformations, however the number of operations required on a full square matrix is $2n^3$, where n is the dimension of the n x n matrix. If Householder's transformations are used to reduce the dynamic matrix, only 2n operations are required, making this method quite efficient. The details of Householder's method and the QL method are omitted for brevity's sake, but the details and algorithms may be found in References 18 and 19 respectively.

For the eigenvalue analysis, a full mesh is usually used, as breaking up the grid into symmetric pieces may remove vibration modes that may exhibit a lower frequency. In practice, this may be desired, as exploiting symmetry for the transient FE analysis will eliminate these modes anyhow. Once the eigenvalues for the system are determined, they are sorted from lowest to highest and the natural frequencies are computed via: $f_n = \sqrt{\omega_n^2}/2\pi$. In some cases, there will be a few natural frequencies with values near zero. These are modes related to lateral rigid-body motion of the structure. Unless the system is properly constrained against rigid-body motion, there will always be some near zero eigenvalues. Penalty constraints are applied to the exterior of the steel sleeve to prevent rotation of the grid. If symmetry is used, penalty constraints are applied to the nodes that lie along the symmetry lines to allow displacement in the radial direction only. Penalty constraints impose extra stiffness or "stiffness penalty" to the required nodes in the required direction such that it represents a region of very stiff material.[11,12] No or negligible displacement of the nodes in that direction may happen. Penalty constraints are not suitable for the transient analysis as the greater stiffness imposed by the penalty shortens the time step to an unacceptably small value (see Section 2.4.4).

With the eigenvalues for the system extracted and the natural frequencies known and sorted from lowest to highest, harmonic testing must be used to find which natural frequency is dominant in the system. A dominant natural frequency is a natural frequency that produces significant displacements in the system when exposed to an external periodic excitation force. In this case, the harmonic tests use a sinusoidally varying internal pressure with a maximum/minimum value of ±1 kPa. The excitation frequencies of the internal pressure are the natural frequencies from the eigenvalue analysis. Setting the damping ratio to zero and beginning from the lowest frequency, the structure is allowed to resonate and the displacements are recorded. This is repeated for all the natural frequencies in the system until the natural frequencies that produce the most prominent responses are determined. The natural frequency that produces the largest response is the resonant frequency. However, when a system is being analyzed that has a great number of degrees of freedom, this method is quite inefficient.

A simpler, less tedious method is to first take the system and apply an impulse pressure or a step increase in pressure to the system with no damping, and record its response. The first two or so prominent frequencies should present themselves from this response. The above process of harmonic testing may then be done using only the natural frequencies closest to the frequencies observed in the impulse response. This method will drastically reduce the amount of time required to find the dominant frequencies. An interesting additional benefit from this methodology is that depending on where in the system the impulse responses were recorded from, the natural frequency of that local region would be superimposed on the resonant frequency of the whole system. For example, in an SRM, the natural frequency of just the casing would be higher than the whole system with the propellant, as it is much stiffer. If the impulse response were recorded from the casing, the resonant frequency of the whole structure would show in the response, but with no damping, the natural frequency of the casing should appear superimposed on the resonant response. This benefit will become useful when used on the star-grain SRM analysis, as the various parts of the grain may have

different resonant frequencies. This method of harmonic testing will allow any point in the system to be analyzed for its dominant response.

As a result, the most dominant natural frequency of the system is the resonant frequency of the system, which is used in the damping model. The resonant frequency is evaluated for both cylindrical and five-point star-grain shapes. These values are presented and used in Chapters 5 and 6 respectively. In Section 2.6, the resonant frequencies for a variety of cylindrical grains is evaluated in order to determine if a more refined mesh affects the result. As seen in the results presented in Table 2-2 in Section 2.6, the evident answer is no. Having more degrees of freedom in the system that is above some basic minimum generally does not reduce any error in the computed natural frequency.[11] It does however greatly increase the computing time. The resonant frequencies of the system do in general take some time to evaluate. However, they only need to be calculated once for any given geometry. Once the resonant frequency is determined for a structure, it may be used even with grids that are more refined – as long as it has the same geometry. Care must however be taken once the propellant begins to burn away, as the structural characteristics of the system will change. The resonant frequency of the system must be recalculated if there is any significant change in the motor geometry due to propellant regression.

2.6 FE results for a Cylindrical Grain Motor

As with any numerical model, its results must be verified by comparing them to solutions of simpler systems generated by other methods in order to ensure that any output that the FE simulation model generates is reliable and considered accurate. This is done by either using analytical solutions or other reliable numerical solutions – typically from commercial packages – and comparing the output from the simulation to them. In this thesis, verification of the FE results is done on a cylindrical-grain motor. The simpler geometry allows both analytical and numerical solutions to be generated and used in comparison. The FE model verification process is broken up into two components – static testing and dynamic testing.

Static results are generated by allowing the simulation to run until the maximum magnitude of acceleration in the system is below some minimum value (typically less than 10^{-6} m/s^2). These results are then compared to analytical solutions and solutions generated by the ANSYS commercial finite element program. For the sake of simplifying the analytical analysis and the ANSYS model, the aluminium casing in the cylindrical-grain SRM will be replaced by a steel casing of the same dimensions. This should have no effect on the accuracy of the results. Therefore, all grids used in the static FE testing will look like the one shown in Figure 2-24. Note that the casing is still modelled, however it is modelled using the material properties of steel, not aluminium, as in Figure 2-3.

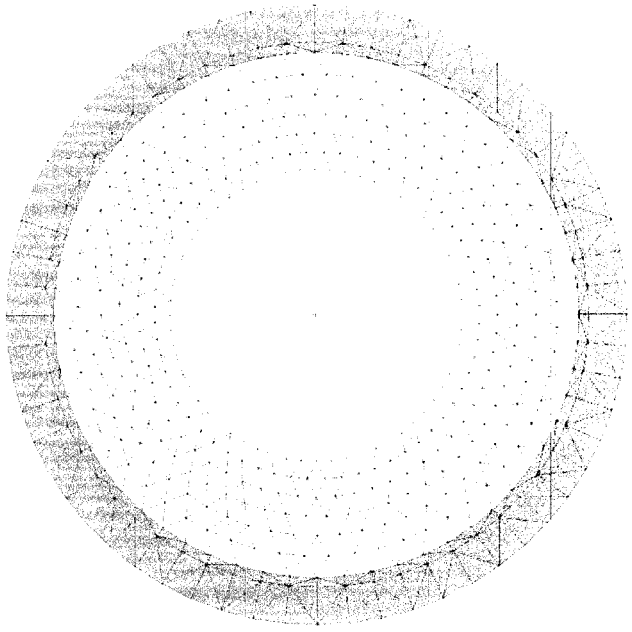


Figure 2-24 – Static test material layout

The static test case of a cylindrical-grain motor may be thought of as a thick-walled cylinder under pressure. Therefore, an analytical solution may be found using thick wall theory for the stress analysis of a long cylinder (plain strain analysis). This analysis assumes that there is no change in the field variables in the circumferential direction, thus making this a one-dimensional analysis (of field variables in the radial direction).

The governing equation for the displacement field of a cross section of a cylinder neglecting thermal and body forces is,[10]

$$\frac{d}{dr}\left[\frac{1}{r}\frac{d}{dr}(ru)\right]=\frac{d^2u}{dr^2}+\frac{1}{r}\frac{du}{dr}-\frac{u}{r^2}=0 \tag{2.99}$$

where r is the axis and u is the radial displacement of a point in the r direction. Equation (2.99) may be integrated to obtain a homogenous solution for u:

$$u=C_1 r+\frac{C_2}{r} \tag{2.100}$$

where C_1 and C_2 are arbitrary constants. The radial and circumferential stress components may then be found for a plain strain analysis:[10]

$$\sigma_r=\frac{C_1 E}{(1+\upsilon)(1-2\upsilon)}-\frac{E}{1+\upsilon}\frac{C_2}{r^2} \tag{2.101a}$$

$$\sigma_\theta=\frac{C_1 E}{(1+\upsilon)(1-2\upsilon)}+\frac{E}{1+\upsilon}\frac{C_2}{r^2} \tag{2.101b}$$

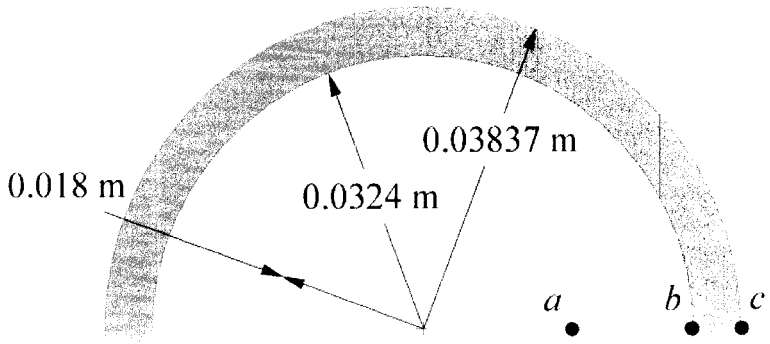


Figure 2-25 – Analytical model schematic

As mentioned earlier, the system as shown in Figure 2-24 is simplified such that only the steel sleeve and the propellant are modelled. A labelled schematic of the system for the analytical analysis is shown in Figure 2-25. Thus, Equation (2.100) will exist for both the steel sleeve and the propellant. This means that there are not two arbitrary constants, but four such that Equation (2.100) results in

$$u_1 = C_1 r + \frac{C_2}{r} \text{ , for } b \le r \le c \text{ (steel sleeve)} \tag{2.102a}$$

$$u_2 = C_3 r + \frac{C_4}{r} \text{ , for } a \le r \le b \text{ (propellant)} \tag{2.102b}$$

for each material. Equation (2.101a) is also similarly applied for both materials such that:

$$\sigma_{r_1} = \frac{C_1 E}{(1+\upsilon)(1-2\upsilon)} - \frac{E}{1+\upsilon}\frac{C_2}{r^2}, \text{ for } b \le r \le c \text{ (steel sleeve)} \tag{2.103a}$$

$$\sigma_{r_2} = \frac{C_3 E}{(1+\upsilon)(1-2\upsilon)} - \frac{E}{1+\upsilon}\frac{C_4}{r^2}, \text{ for } a \le r \le b \text{ (propellant)} \tag{2.103b}$$

It is not necessary to apply Equation (2.101b) to both materials, as it is not required to solve the system. However, if the circumferential stresses are needed, Equation (2.101b) is easily applied.

For labeling purposes, the steel sleeve is labelled as material 1 and the propellant as material 2. The elastic modulus and Poisson's ratio are values typical for steel and the propellant used and found in Table 2-1. As shown in Figure 2-25, point a lies on the inner propellant surface (r = 0.018 m), point b lies at the point where the steel sleeve and the propellant surface meet (r = 0.0324 m) and point c lies on the outer steel sleeve (r = 0.03837 m). The boundary conditions for the above differential equation may now be stated. At point a, the stress in the radial direction is equal to the applied pressure (p). At point b, the radial stress in the propellant is equal to the radial stress in the steel sleeve as is the displacement. At point c, there is no applied pressure above ambient conditions (p = 0), so the radial stress is zero. These boundary conditions may be restated as,

$$\sigma_{r_2}(a) = -p \qquad \sigma_{r_2}(b) = \sigma_{r_1}(b) \tag{2.104a}$$

$$\sigma_{r_1}(c) = 0 \qquad u_1(b) = u_2(b) \tag{2.104b}$$

Implementing these boundary conditions into Equation (2.102a), Equation (2.102b), Equation (2.103a) and Equation (2.103b), there will exist four equations and four unknowns (C_1, C_2, C_3, C_4). Therefore, the solution process is non-trivial and straightforward. The evaluation of the analytical displacements is taken at point a, and point c, as data from these points is commonly used in other literature and so used here for consistency. Using an applied internal pressure of 10.5 MPa as a test case, the propellant displacement at point a is 100.217 microns and the outer steel sleeve displacement at point c is 8.972 microns. These values are tabulated in Table 2-2.

Using the ANSYS version 5.4 commercial finite element program, the above test case is modelled using six-node triangular elements with two degrees of freedom

per node. The total degrees of freedom in the system is 10286. Upon solution of the system, the radial displacement at points a and c are 100.0 microns and 8.950 microns respectively. These values are nearly identical to those found from the analytical solution. To find the percent error in the ANSYS solution (taking the analytical solution as exact), a basic error calculation is used, the results of which are tabulated in Table 2-2:

$$\varepsilon = \left| \frac{u_{FE} - u_{theory}}{u_{theory}} \right| \times 100\% \tag{2.105}$$

Equation (2.105) is also used to evaluate the error of the FE results with respect to the analytical results.

The grids used for the FE tests, are the same as those found in Figure 2-3 through Figure 2-6 labelled CG1 through CG4 respectively. Again, note that the aluminium casing is replaced with a steel casing such that there are only two materials (see Figure 2-24). To distinguish these grids from those in Figure 2-3 through Figure 2-6, the suffix 's' is placed in the grid label (i.e. CG1 becomes CG1s). The displacements and error of the FE tests are tabulated in Table 2-2 along with the theoretical and ANSYS results as a comparison. In addition, the resonant frequencies for the grids are calculated and compared to each other.

Analysis	u_a **(μm)**	u_b **(μm)**	ε_a	ε_b	f_n **(Hz)**
Theory	100.217	8.972	0%	0%	N/A
ANSYS	100.000	8.950	0.22%	0.25%	N/A
CG1s	101.390	8.811	1.17%	1.79%	14808
CG2s	99.365	8.947	0.85%	0.28%	14874
CG3s	100.899	9.002	0.68%	0.33%	14847
CG4s	100.878	9.009	0.66%	0.41%	14892

Table 2-2 – Cylindrical-grain static FE results

As was expected, the error in the FE computations is reduced by using progressively finer grids. However, only to a certain level. As discussed in Section 2.2, a point of grid independence will be reached where refining the grid any more will not further reduce the error. Grid independence is reached using grid CG3s. Grid CG4s, having many more degrees of freedom does not produce any significant reduction in error. Also, note that these tests were run both with full grids and with symmetric grids, both of which having produced similar results, thus only the results of the full grids is presented in Table 2-2. The ANSYS results in comparison have a smaller error than the FE output; however, this would be expected as higher-order six-node elements are used, thus the system may be solved for with a greater accuracy – albeit with a longer computational time. The error in the FE static test cases is reasonably small and with one exception, less than 1%. This is more than satisfactory for the purposes of this thesis. It should be noted again that although the grids with more degrees of freedom have less error, they take up significantly more computer time when used. Thus, the small increase in error offered by using a coarser grid may be acceptable, considering the significant reduction in computer time presented.

As discussed in Section 2.5, the natural frequencies of the system should not vary appreciably from one mesh to the next. The results for the natural frequencies of the FE grids confirm this. All the frequencies tend to lie at approximately 14.85 kHz. This natural frequency may be used as the resonant frequency for damping calculations in this particular system.

To test the transient solution components of the FE solver, a test case of a cylindrical grain SRM is used such that a 10.5 MPa step increase in the internal pressure is applied to this system. Note that for the dynamic tests, the aluminium casing is modelled – unlike the static tests. Grid CG3 (see Figure 2-5) is used in all the results discussed in this section. The resonant frequency for this system used for the damping model is found to be 14.337 kHz. The damping ratio ζ is set to 0.1 – a value that should reduce the amplitude of the displacement off the equilibrium value by about half each

cycle for a particular resonant frequency.[16] In addition, the numerical dissipation constant γ in Equation (2.41), which controls the amount of numerical dissipation, is set to 0.74. Although this value can only guarantee at least first order accuracy, it is necessary to reduce some of the high frequency noise in the solution, particularly in the acceleration terms. In the Newmark method, γ dampens the higher frequency modes; and although it does have an effect on the lower modes as well, it is small while $\gamma < 1.0$. Figure 2-26 shows the response of the FE system, using grid CG3.

In Figure 2-26, the displacement versus time is plotted for both the outer surface (point *c*) and the propellant surface (point *a*). Note that the response settles after about 0.6 ms to roughly the equilibrium value. The primary oscillation frequency is approximately 14.2 kHz, which is close to the expected value of 14.265 kHz as defined by,[16]

$$f_d = \sqrt{1-\zeta^2}\, f_n \tag{2.106}$$

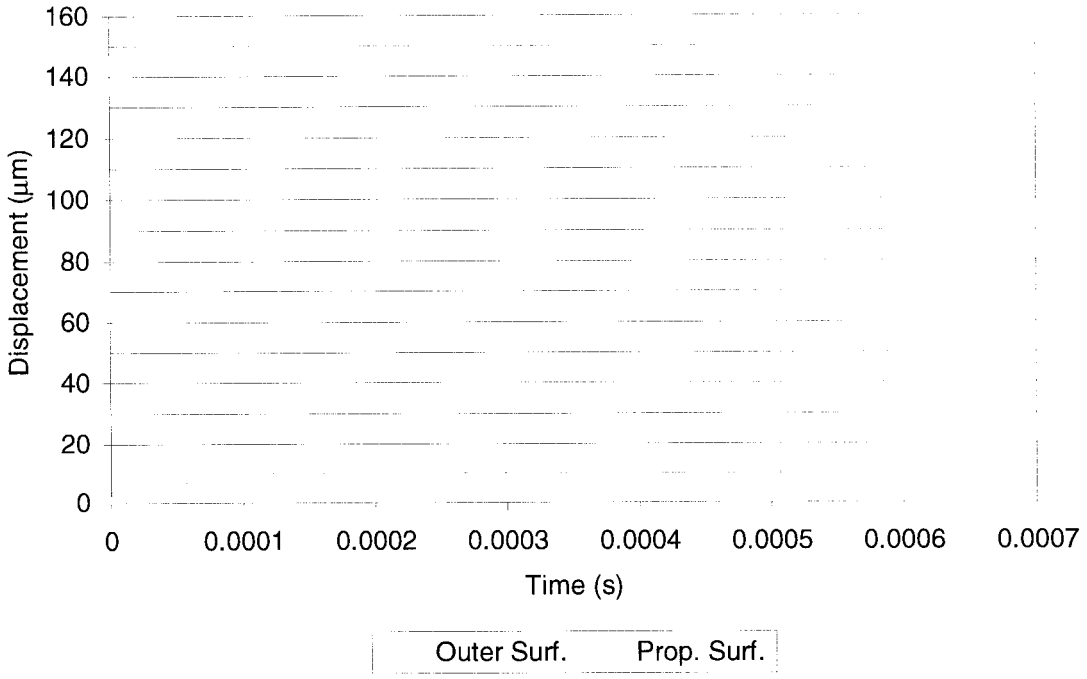


Figure 2-26 – Response of grid CG3 to a 10.5 MPa step pressure increase

where f_d is the damped resonant frequency and f_n is the resonant frequency of grid CG3. As expected, the amplitude of the dynamic displacement decays by approximately half each cycle. Looking at Figure 2-26, it is evident that there are other vibration modes present in the response. These modes may be inherent to the response of the system or may be a product of the first-order nature of the constant-strain elements. However, these modes are very small in amplitude in comparison to the primary mode and aren't considered a problem. Grids CG1 through CG4 all exhibited nearly the same response. The only variations came in the form of very small differences in the displacements at any time, the primary oscillation frequency and any secondary modes. These variations are due to the subtle differences in the grids.

To verify that the response for this system may be relied upon, a similar test was

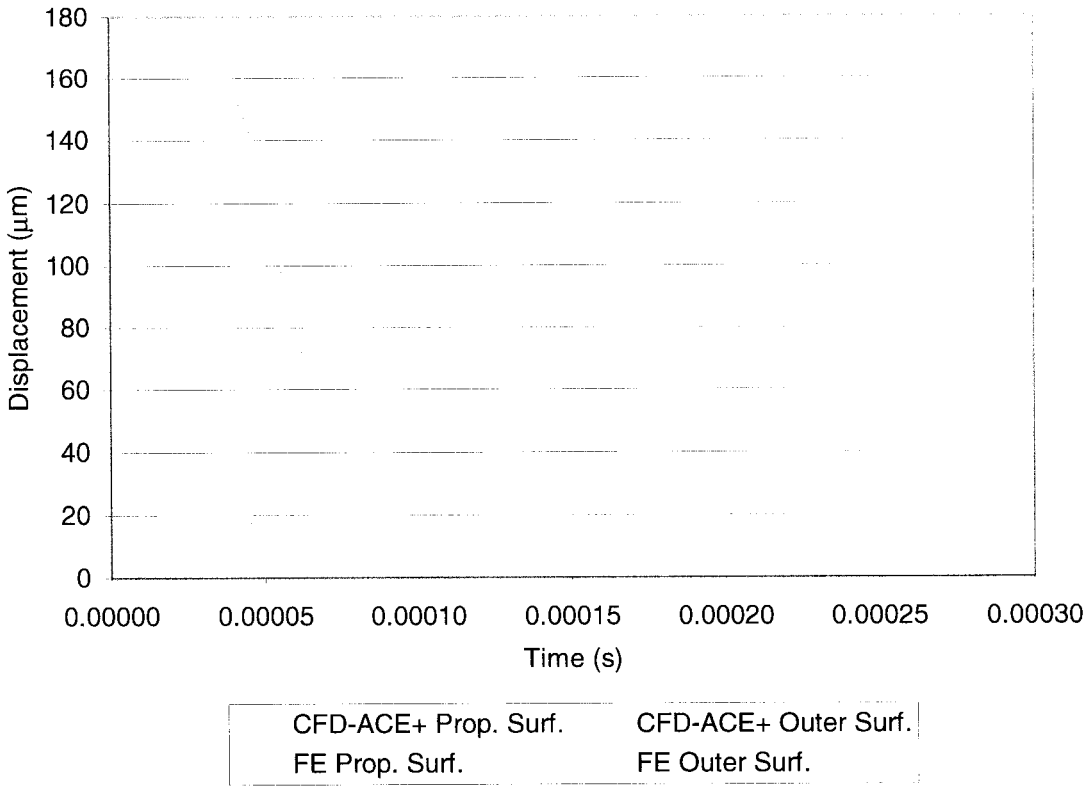


Figure 2-27 – Comparison of CFD-ACE+ and FE transient results

done via another commercial program (CFD-ACE+) by Greatrix and Kudriavtsev.[20] CFD-ACE+ is a comprehensive multi-physics flow solver developed by CFD Research Corporation. It also has a finite element module (FEMSTRESS) to model surrounding structures and can include fluid-structure interactions. CFD-ACE+ was used to model a section of a cylindrical grain setup, applying a 2-D plain strain analysis of the motor. The CFD-ACE+ model was subjected to the same 10.5 MPa step increase in pressure as the FE analysis was, with a damping ratio of 0.1. In Figure 2-27, the CFD-ACE+ results are displayed on a displacement versus time plot. The FE results as shown in Figure 2-26 are overlaid in Figure 2-27 as a comparison.

From the results displayed in Figure 2-27, it is evident that the FE response does resemble the CFD-ACE+ response closely. There is on average a 2.1% difference between the maximum and minimum displacement values per cycle. Both the FE model and the CFD-ACE+ model oscillate at approximately 14.2 kHz. Although there is a slight difference in the oscillation frequencies, it does not seem very significant. The small differences in the responses of the FE and CFD-ACE+ systems may be due to different types of elements being used in the modelling or possibly a difference in the damping model. Additionally, using lumped or consistent mass matrices in CFD-ACE+ will tend to produce slightly different natural frequencies, which would also affect the damping model.[11]

Using static and dynamic tests, the FE component of the simulation is verified through a comparison of analytical and other numerical data (ANSYS and CFD-ACE+), which modelled nearly identical systems. Static FE tests agreed well with theory and with output from ANSYS, producing typically less than 1% error (with respect to theory). Dynamic tests showed that the system oscillates at a frequency nearly the same as predicted by the harmonic analysis of the system, and that the damping model works as predicted. Although the dynamic response of the CFD-ACE+ and FE models are not in exact correlation, they are close enough to suggest that the FE program does produce results within the accuracy expected for this investigation.

2.7 Propellant Regression

Propellant regression affects the inertial, damping and elastic characteristics of the structure, as well as the port geometry required for the Internal Ballistic Flow (IBF) calculations. This is all manifested through the grid, as the grid defines the mass and stiffness matrices (the damping matrix is a function of the mass matrix), and must model the port geometry through its inner boundary. As the propellant slowly burns away, the propellant mesh must be continually updated and adapted to compensate for the lost propellant. The propellant regression algorithm is broken up into two parts – the burn-back of the interior boundary, which will be discussed first, and the modification of the propellant grid topology resulting from the movement of this boundary, which will be discussed afterwards.

For the purposes of this thesis, the interior boundary is labelled the "Control Boundary" (CB), as this boundary will control how much modification the FE mesh will undergo, as well as being a boundary that must be regulated so as to keep its shape true to reality. The CB represents the burning surface of the propellant. This surface must be accurately modelled since from this boundary the port area and length are calculated, as are the local surface accelerations required for the burning rate calculations. Since the FE grid uses straight-sided triangles, the CB is made up of a series of straight-line segments. Straight lines don't represent curves very well (concave curves in the CB will underestimate the port area and the perimeter length). However as the element size decreases, so do the segment size, thereby better approximating the port geometry and reducing the error in the port area and perimeter length calculations (see Section 4.2).

The nodes in the CB are the only points that are actually placed on the physical burning surface; the exception is of course along straight sides, where any spot along the line segment also lies along the physical burning surface. These nodes are the points where the burning rate is calculated in the IBF component of the simulation. It is also where the burn-back algorithm is applied which will modify the CB. The burn-back

algorithm uses a simple vector relationship, where a burn vector $\{B\}_i$ is vectorially added to the current position of the node (see Figure 2-28). The burn vector represents the distance and direction a node has "burned" over an increment of time. A unique burn vector is calculated for every i^{th} segment at every time step via,

$$\{B\}_i = r_{b_i}\{u\}_i \Delta t \tag{2.107}$$

where r_{b_i} is the segment burning rate, $\{u\}_i$ is the segment directional unit vector which represents the burning direction, and Δt is the current time increment.

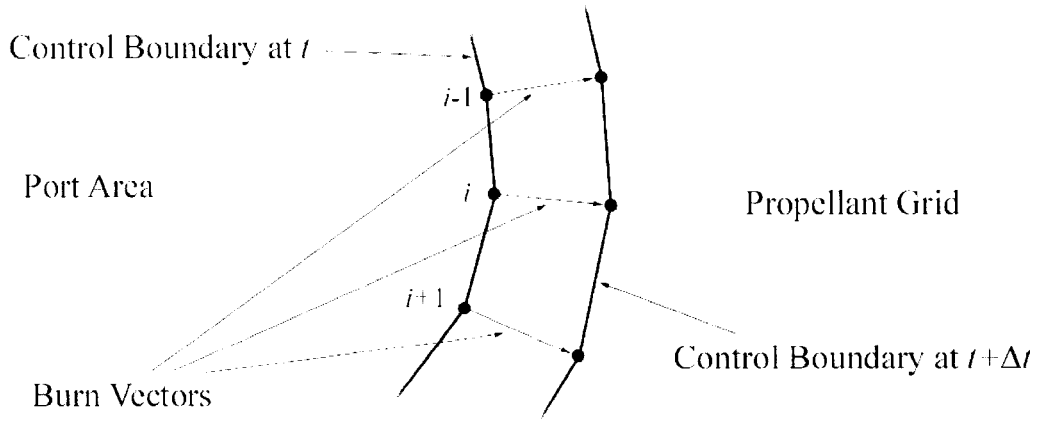


Figure 2-28 – Burn-back vector addition

As mentioned in Section 2.4.1, the burning rate and direction are constant over each FE time step, so Equation (2.107) is trivial in its solution. The exception is at nodes that sit at points and those along convex curves in the CB. At these points, an artificial augmentation of the burning rate is required so these nodes don't lag behind the rest of the nodes in the CB. Figure 2-29 illustrates this idea using burn vectors of equal magnitude. Using vector projection, a suitable augmentation can be formulated. First, a second burn vector labelled $\{B_2\}_i$ is created at the point such that its direction is

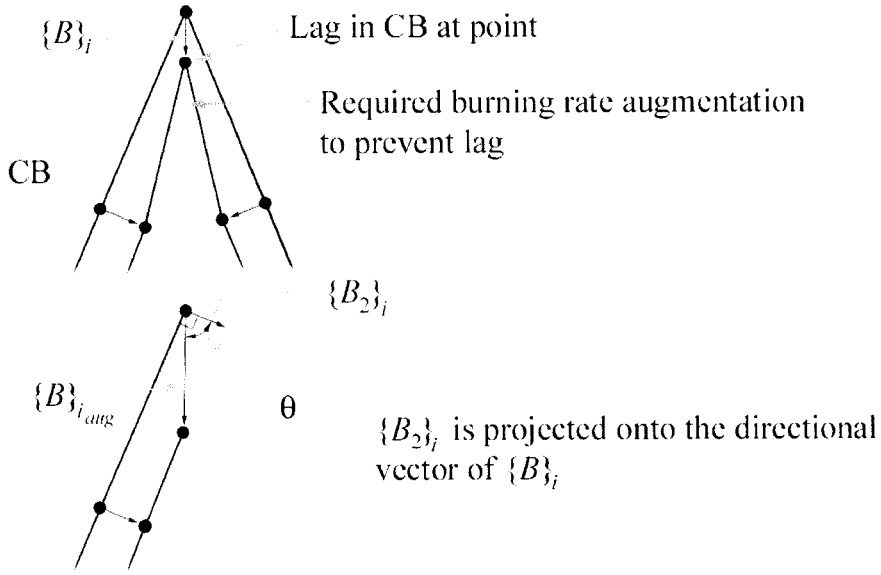


Figure 2-29 – Burn rate augmentation at points and convex curves in the CB

normal to the segment it represents, but it has the same magnitude as the original $\{B\}_i$ (i.e. $\|B_2\|_i = \|B\|_i = r_{b_i}\Delta t$). Then the inverse projection of $\{B_2\}_i$ onto $\{u\}_i$ (the directional unit vector for the segment – see Equation (2.78)) is taken to find the augmented burn vector $\{B\}_{i_{aug}}$, where $\|B\|_{i_{aug}}$ is the augmented burning rate. Algebraically this projection is expressed as:

$$\{B\}_{i_{aug}} = \frac{\|B_2\|_i^2}{\{u\}_i \cdot \{B_2\}_i}\{u\}_i = \frac{r_{b_i}\Delta t}{\cos\theta}\{u\}_i = r_{b_{i_{aug}}}\{u\}_i \tag{2.108}$$

The angle θ in Equation (2.108) is the angle between $\{u\}_i$ and $\{B_2\}_i$. The augmented burning rate is expressed as:

$$r_{b_{i_{aug}}} = \|B\|_{i_{aug}} = \frac{r_{b_i}\Delta t}{\cos\theta} \tag{2.109}$$

Using the augmented burn vector for the necessary nodes in the CB will prevent any lag in these nodes and maintain the accuracy of the CB.

An important property of the burn vector is that its directional unit vector must remain normal to the CB surface at that point. When the CB is straight, this is a straightforward task where the segment directional unit vector is simply a unit vector that is normal to a vector that defines the segment. However, when the CB is curved, this process is not so straightforward. In order to identify which direction is normal to the burning surface at the i^{th} node, the curvature of the burning surface must be locally approximated. There are of course numerous curve-fitting schemes available that can approximate a curve given a series of points; however, to keep things simple, the normals are found using a weighted averaging technique that does not require a curve to be fitted to the whole CB (or a section thereof). This method approximates a normal based on the information in the two adjacent segments. If vector $\{P\}_i$ is a "segment vector" that represents the i^{th} segment, and vector $\{P\}_{i-1}$ the segment before the i^{th}, then a directional vector $\{N\}_i$ for node i will be equal to the weighted average of the normals of $\{P\}_i$ and $\{P\}_{i-1}$ where the weights are the lengths of these vectors. This is illustrated in Figure 2-30 and expressed below in Equation (2.110):

$$\{N\}_i = \begin{Bmatrix} N_{x_i} & N_{y_i} \end{Bmatrix} = \frac{\begin{Bmatrix} -\left(P_{y_i} + P_{y_{i-1}}\right) & P_{x_i} + P_{x_{i-1}} \end{Bmatrix}}{\|P\|_i + \|P\|_{i-1}} \tag{2.110}$$

P_x and P_y are the x and y components of the segment vector $\{P\}$. $\{N\}_i$ is then turned into the directional unit vector $\{u\}_i$ by performing the simple operation:

$$\{u\}_i = \frac{\{N\}_i}{\|N\|_i} \tag{2.111}$$

The directional unit vectors must also always point into the propellant. This is done by normalizing all the directional unit vectors on the CB with a simple check statement.

The CB segment vectors will run normally clockwise, so the sign of the cross product between the directional unit vector and the segment vector must be positive for all directional unit vectors. If the unit vector is pointing the wrong way, it is inverted.

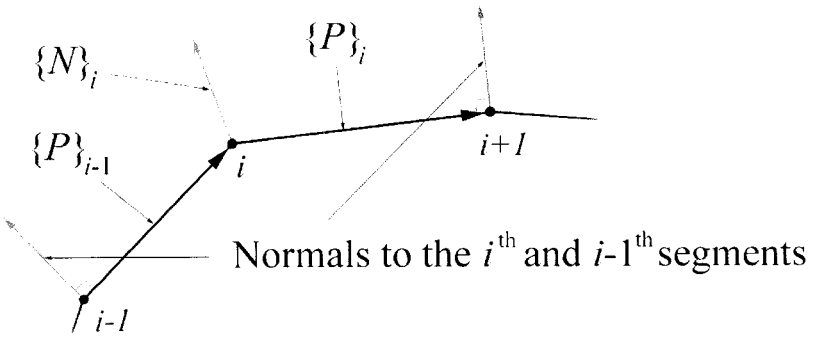


Figure 2-30 – Weighted averaging method for the directional unit vectors

This method of obtaining the directional burn vector is efficient and proven to be accurate. It also may be used on both curved and straight parts of the CB. Care should be taken when dealing with nodes that lay on both the CB and a symmetry line. In these cases, the directional unit vector should be collinear with the symmetry line. Artificial burning rate augmentation as described above may be necessary at these points depending on how the CB intersects with the symmetry line. Other approximation methods for finding the directional unit vector that involved curve fitting were attempted and produced nearly the same results over curved segments, but required special treatment for straight sides (curve-fitting along a straight segment which is adjacent to other curved segments tends to introduce "wiggles" into the fitted curve) or were too complicated and so were dismissed. Once the directional unit vector is found for a node on the CB, it is not changed unless there is a grid modification (discussed below). This is suggested by Greatrix so that the CB is not "smeared", causing error in the definition of the CB with the regression of the propellant.[7]

Having a suitable boundary that clearly defines the internal burning surface is the first step in the propellant regression algorithm in this simulation. The next step must involve modifying the FE propellant grid topology. As the CB moves outward, it

will change/delete elements and absorb nearby nodes and thus altering the mass and stiffness properties of the system. This of course makes physical sense; as propellant is burned away, there will be less mass in the system; also, having less propellant, which is many orders of magnitude less stiff than the surrounding casing and sleeve, will effectively make the structure stiffer. A geometric change to any of the nodes or elements in the system will cause a change in mass and stiffness properties; therefore, the elemental mass and stiffness matrices must be reevaluated for any modified elements.

In the simplest scenario where the grid must be modified, no elements are deleted or nodes absorbed into the CB. Only the nodes in the CB are stepped back by their respective burn vectors. This has the effect of only altering the shape of the elements that border the CB, and so no real adaptation of the grid to the new geometry is needed. This case is illustrated in Figure 2-31.

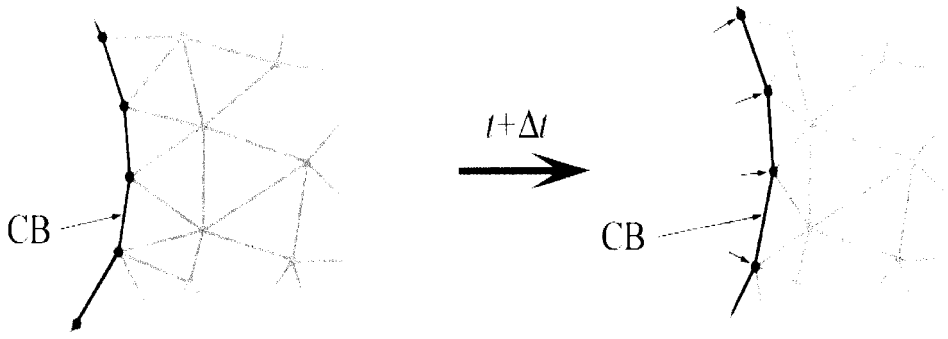


Figure 2-31 – Simple case for grid modification

As the propellant regresses, elements bordering the CB will eventually become small and their shapes distorted. Once an element shrinks by a certain amount, the grid modification routines must be run to adapt the grid to the new geometry. The check statement for this relies on the area ratio between the current "burnt away" element area and the original element area upon its creation. A typical cutoff area ratio would be set at 45% of the original element area, but may vary between 20% and 60% depending on

the grid. The cutoff area ratio should not be too small to maintain elements that are as close to being equilateral as possible, and not to produce distorted elements (elements that are needle-like or very obtuse). The check statement is run every time step after the CB is moved and after any grid modifications have been done.

When the check statement finds an element that must be modified, the grid adaptation routines are run. There are a great many possible scenarios in how and which elements are modified; and how the related nodes are repositioned – it all depends on the local grid topology. Using unstructured grids, the number of element modification scenarios increases as the grid topology is apparently random. In this thesis, sixteen different element modification scenarios were identified. There are, as already mentioned, many more possibilities; fortunately, these sixteen cases were all that was required in the grids used. It would be tedious to describe and discuss all sixteen cases, so instead the cases are represented graphically in Figure 2-32, Figure 2-33 and Figure 2-34. Only the nodes and elements that are involved in the modification or relevant to the case's topology are drawn to save space. Symmetry lines are drawn in blue, the CB segment(s) in black and elements that are to be deleted are shaded in pink and nodes in red. The element that initially failed the check statement is marked with an 'x'. Arrows show where certain nodes are to be relocated. Each case is shown both before and after modification. Cases #1 through #4 have dashed gray curves, which represent an approximated burning surface in between CB nodes. These dashed lines represent the possible places that nodes, which are relocated to the CB, can be placed. Node relocation onto the CB will be discussed later in this section.

Note that some of the cases are drawn as concave surfaces. This is done arbitrarily and is for the sake of consistency in the illustrations. These cases also work just as well for convex and straight-sided surfaces as well. The exception is of course Case #16 where the grid is modified at a point. Many of the above cases could be thought of as similar or the same; what makes each case different from any other is the unique identifying features of that case. Each case in Figure 2-32 through Figure 2-34,

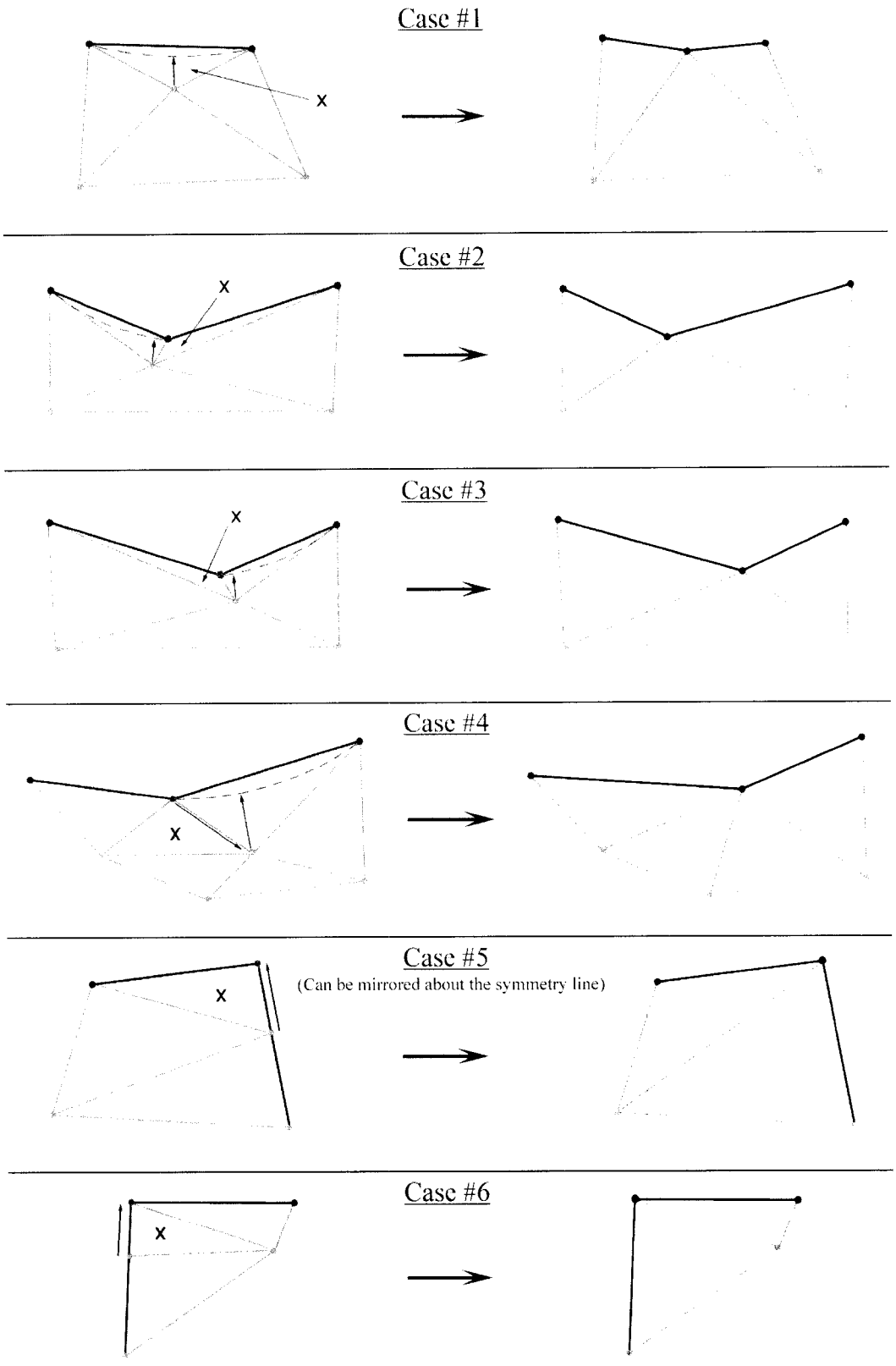


Figure 2-32 – Grid modification schemes (Cases #1 – #6)

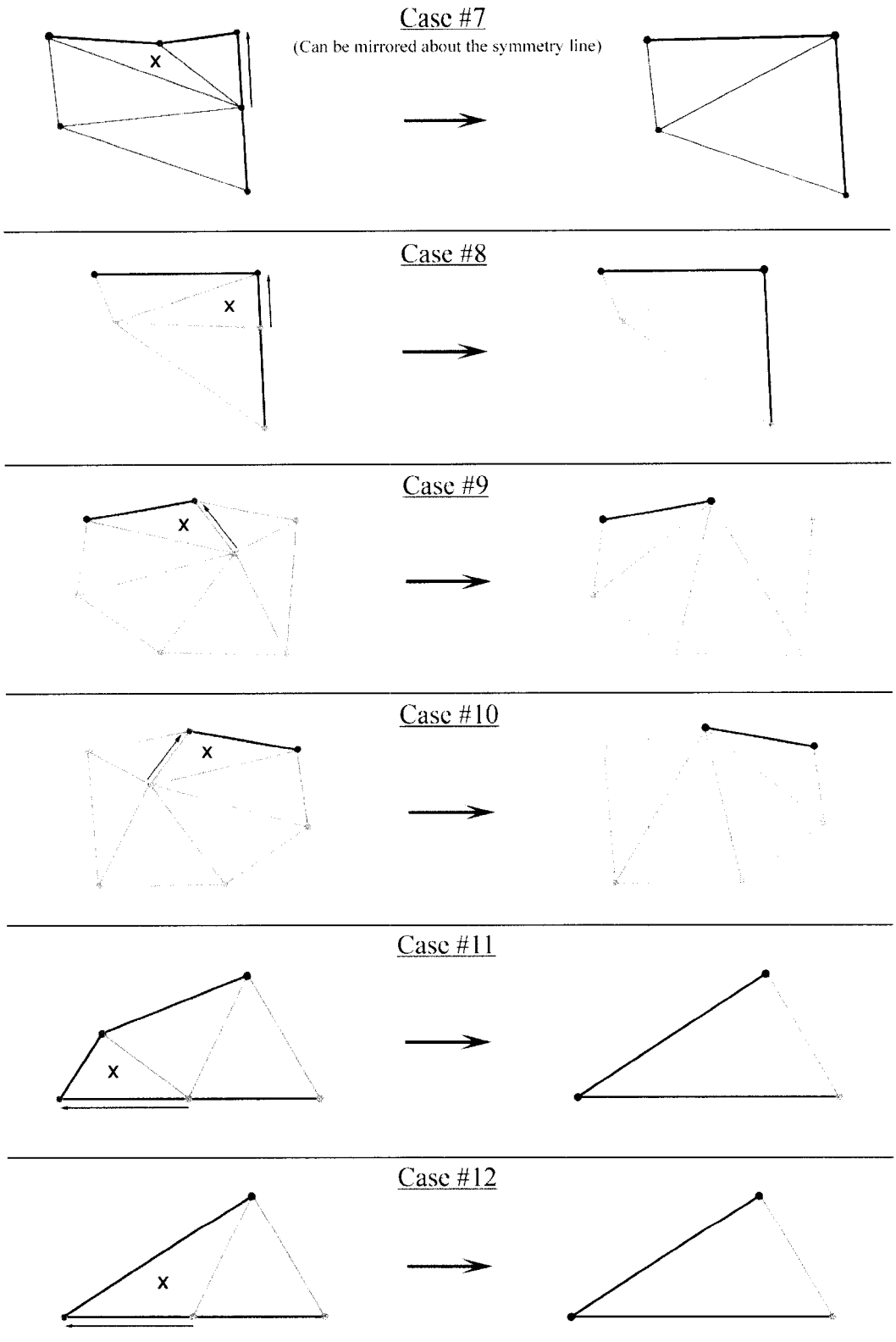


Figure 2-33 – Grid modification schemes (Cases #7 – #12)

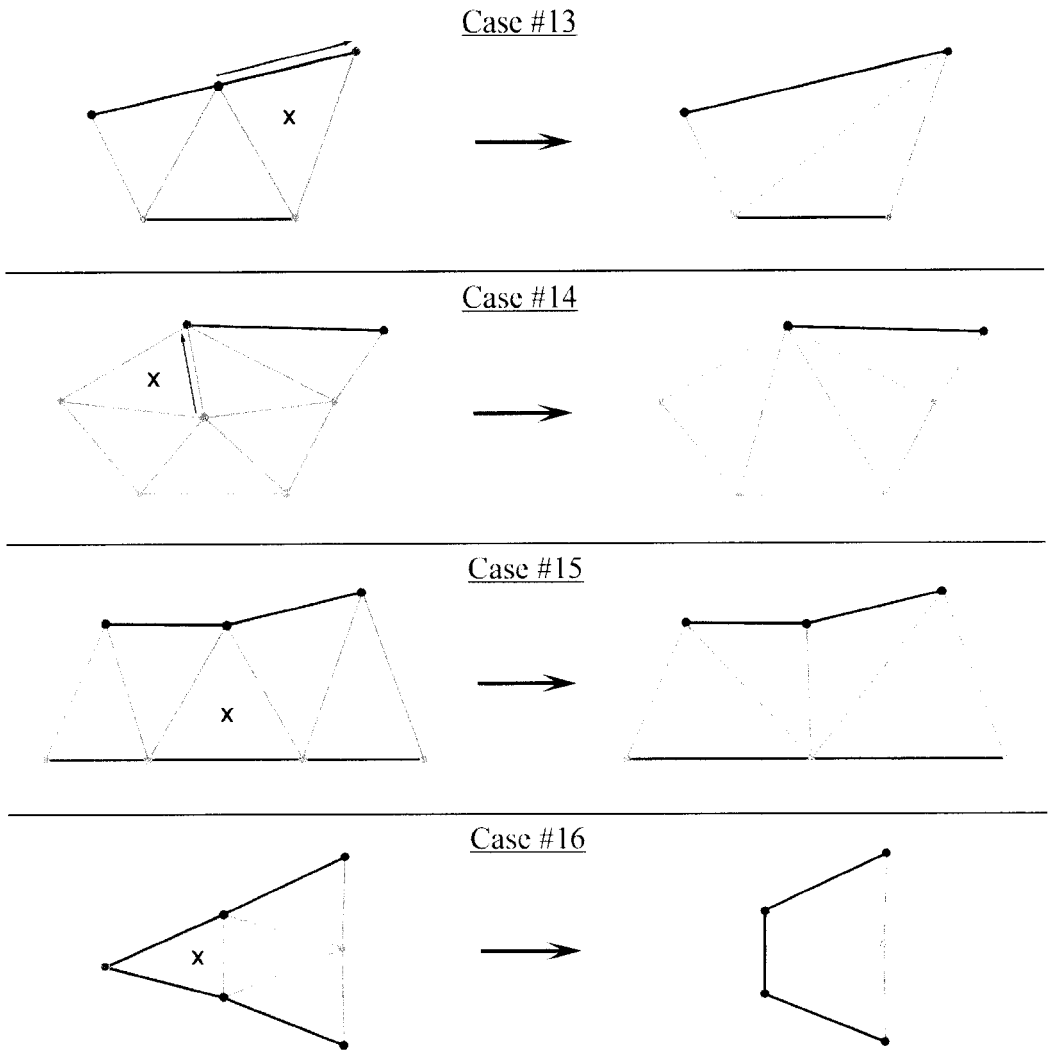


Figure 2-34 – Grid modification schemes (Cases #13 – #16)

has a set of unique identifying features that allow a computer algorithm to identify them. To list the features unique to each case here would be long, tedious, and unnecessary. The identifying features for any case are self-explanatory for the most part by looking at the appropriate illustration in Figure 2-32 through Figure 2-34. Many differences between cases rely on if a node(s) is/are laying on a symmetry line or on the CB, or if the element is obtuse or acute in shape. The grid modification algorithm is designed to analyze the local topology and decide on which case(s) suit the topology. Then the algorithm will decide which case (if more than one) will produce the best elements (whichever case will produce elements shaped closest to an equilateral

triangle), and uses the selection to modify the grid. This is also why some cases are nearly the same – just to give the above algorithm some choices in its case selections. Some cases, such as Case #5 and #7, can be mirrored (about the symmetry line), as their identifying features are the same either way; unlike Cases #2 and #3, and #9 and #10, where although they are opposites to each other, they each have differing identifying features. Cases may also be used one after another since sometimes when an element is modified, it will fail the second area ratio check statement, and as a result, the grid must be further modified.

The most important thing in the modification of the FE propellant grid is to maintain the integrity of the CB. In most of the cases, the CB is not interfered with in the modification process. However, there are cases where the CB must be modified in order to perform the grid modification. This must be done without unacceptably distorting the local elements too much. Cases #7, #11, #13 and #16 change the CB, but the interior nodes are moved to the CB only to replace nodes that were originally there (the exception is Case #16 where no nodes are moved). These cases don't change the integrity of the CB (points that were known to be on the burning surface have not changed/moved), although they can reduce the resolution of it (some nodes are deleted from the CB, so the position of a particular point on the burning surface is lost). This reduction in CB resolution due to Cases #7, #11, #13 and #16, has shown itself to generally not be a problem in the grids used, since the reduction is balanced by an increase in resolution from Case #1. (Case #1 was found to be the most utilized case by an appreciable amount, tending to increase the overall CB resolution as the propellant burned away. Generally speaking, the resolution of the CB depends on the local element density in the grid as the CB regresses past a point – absorbing the nodes into it as it moves, increasing its resolution.)

Cases #1 through #4 also modify the CB, and these cases do affect the integrity of the CB. In these cases, interior nodes are also moved onto the CB, however these nodes are moved to a location in-between the nodes on the CB. This means that the burning surface must somehow be approximated so that the interior node can be placed

accurately, maintaining the integrity of the CB (gray dashed curves in Cases #1 through #4 in Figure 2-32). This is a trivial task if the burning surface is straight, but the majority of the burning surface is not straight. Again, various curve-fitting schemes to approximate the burning surface were evaluated, but none produced satisfactory results for this task. A technique was created that was found to work very well. This approximation uses the linear interpolation between circular arcs based on the angle between directional unit vectors over a segment in the CB. To aid in the explanation of this method, refer to Figure 2-35.

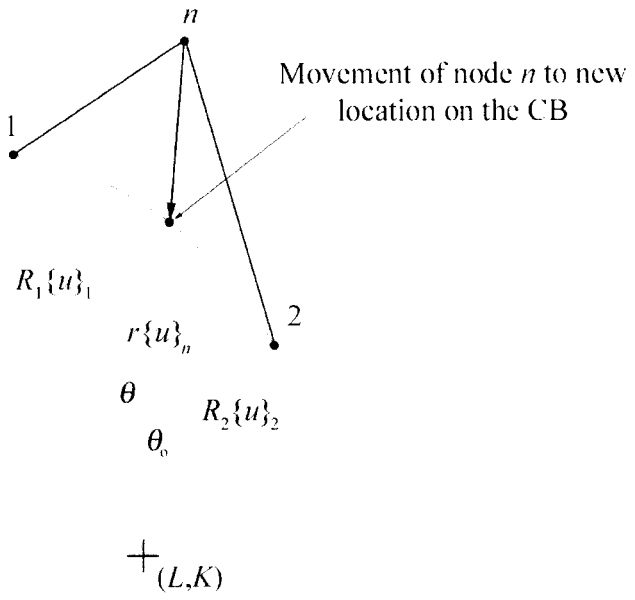


Figure 2-35 – CB curvature approximation

The approximation begins with one segment in the CB with its nodes labelled 1 and 2. The intersection point (L, K) must be found for two vectors that are linearly dependent, with the directional unit vectors of the first and second node in the CB segment. These two vectors have magnitudes R_1 and R_2, and may be expressed as $R_1\{u\}_1$ and $R_2\{u\}_2$ (recall that $\{u\}_i$ is the directional unit vector for node i). This intersection point will represent the center of two circles having radii R_1 and R_2. If the two directional unit vectors are parallel (their cross product being zero), then the burning surface is a straight line (discussed later in this section). Having these two

vectors originating from the intersection point, the two nodes in the CB segment may be expressed as vector combinations of the intersection point and either vector:

$$\begin{Bmatrix} L \\ K \end{Bmatrix} + R_1 \{u\}_1 = \begin{Bmatrix} x_1 \\ y_1 \end{Bmatrix} \rightarrow \begin{Bmatrix} L \\ K \end{Bmatrix} = \begin{Bmatrix} x_1 \\ y_1 \end{Bmatrix} - R_1 \begin{Bmatrix} u_{x_1} \\ u_{y_1} \end{Bmatrix} \tag{2.112a}$$

$$\begin{Bmatrix} L \\ K \end{Bmatrix} + R_2 \{u\}_2 = \begin{Bmatrix} x_2 \\ y_2 \end{Bmatrix} \rightarrow \begin{Bmatrix} L \\ K \end{Bmatrix} = \begin{Bmatrix} x_2 \\ y_2 \end{Bmatrix} - R_2 \begin{Bmatrix} u_{x_2} \\ u_{y_2} \end{Bmatrix} \tag{2.112b}$$

Equations (2.112a) and (2.112b) are rearranged such that they each leave the intersection point on the left-hand side. Doing so, Equations (2.112a) and (2.112b) may be equated to each other to solve for R_1 and R_2:

$$\begin{Bmatrix} L \\ K \end{Bmatrix} = \begin{Bmatrix} x_1 - R_1 u_{x_1} \\ y_1 - R_1 u_{y_1} \end{Bmatrix} = \begin{Bmatrix} x_2 - R_2 u_{x_2} \\ y_2 - R_2 u_{y_2} \end{Bmatrix} \tag{2.113}$$

Combining like terms and following through, Equation (2.113) becomes:

$$\begin{bmatrix} u_{x_2} & -u_{x_1} \\ u_{y_2} & -u_{y_1} \end{bmatrix} \begin{Bmatrix} R_2 \\ R_1 \end{Bmatrix} = \begin{Bmatrix} x_2 - x_1 \\ y_2 - y_1 \end{Bmatrix} \tag{2.114}$$

Solving for the two radii in Equation (2.114) is a simple matter with the result shown below:

$$R_1 = \frac{u_{x_2}(y_2 - y_1) - u_{y_2}(x_2 - x_1)}{u_{y_2} u_{x_1} - u_{y_1} u_{x_2}} \tag{2.115}$$

$$R_2 = \frac{u_{x_1}(y_2 - y_1) - u_{y_1}(x_2 - x_1)}{u_{y_2} u_{x_1} - u_{y_1} u_{x_2}} \tag{2.116}$$

Having solved for the two radii, the intersection point (L, K) can be obtained from either Equation (2.112a) or (2.112b).

The radii of the two concentric circles and their center point are now established, based on the intersection of the directional unit vectors of nodes 1 and 2. To place the interior node n on the burning surface, a third vector $r\{u\}_n$, is required, whose magnitude will be linearly interpolated between R_1 and R_2. Vector $r\{u\}_n$, is a vector that has a magnitude r, and a unit vector $\{u\}_n$ that points from point (L, K) towards node n. The unit vector $\{u\}_n$, is expressed as,

$$\{u\}_n = \left\{ \frac{x_n - L}{\sqrt{(x_n - L)^2 + (y_n - K)^2}} \quad \frac{y_n - K}{\sqrt{(x_n - L)^2 + (y_n - K)^2}} \right\} \tag{2.117}$$

where x_n and y_n are the coordinates of node n prior to its movement to the CB. The magnitude r is found using the expression:

$$r = R_1 - \frac{\theta}{\theta_0}(R_2 - R_1) \tag{2.118}$$

The angles θ and θ_0 in Equation (2.118) are the angles between unit vectors $\{u\}_n$ and $\{u\}_1$, and unit vectors $\{u\}_1$ and $\{u\}_2$ respectively. These can be found using the expression given below, replacing the vectors $\vec{A}$ and $\vec{B}$ for the necessary unit vector pairs:

$$\theta = \tan^{-1} \sqrt{\left(\frac{\|A\| \|B\|}{\vec{A} \cdot \vec{B}} \right)^2 - 1} \tag{2.119}$$

Using Equations (2.112) through (2.119), a vector $r\{u\}_n$ and an origin point (L, K) can be found that will reposition node n to a suitable place along the CB that approximates the actual burning surface. Thus the new position of node n is:

$$\begin{Bmatrix} x_n \\ y_n \end{Bmatrix}_{new} = \begin{Bmatrix} L \\ K \end{Bmatrix} + r\{u\}_n \tag{2.120}$$

This method accurately places interior nodes onto the burning surface allowing them to be absorbed into the CB. As mentioned earlier, if the side is straight, a slightly different methodology must be used. This is because the denominator in Equations (2.115) and (2.116) will become zero if the directional unit vectors of nodes 1 and 2 are parallel. The general idea remains the same, except now the "approximated curve" does not need to be approximated as it lies exactly on the segment vector. The interior node is moved to lie directly on the segment vector at a suitable position.

The intersection point (L, K) changes only when the directional unit vector of one of the two nodes in a CB segment is changed. However, the radius R_i for node i, must be continually updated by adding the burn-back distance $r_{b_i}\Delta t$ every time step. Doing so ensures that when another node is absorbed into the CB, the burning surface is accurately modelled.

Using the methods described in this section, the simulation of propellant grain regression can be accurately modelled. Having the FE grid adapted to the new geometry, the new elemental mass and stiffness properties may be evaluated at any given time step. Normally the FE solution is not run while the burn-back routines are being run, because the changes in grid topology create spurious noise in the FE solution at the location of change (recall from Section 2.4.1 that the elements used in this simulation cannot handle grid changes). However, if there are no changes to the grid topology (as shown in the simple case of burn-back as illustrated in Figure 2-31), then an FE solution could be successfully obtained while updating the mass and stiffness

properties every time step. The problem lies in the fact that over the simulated firing time of the motor where the FE calculations are done (typically 30 – 60 ms), there is a high probability that the grid will require a modification in its topology. Therefore, FE calculations are not run while the grid modification routines are used. Further details on how the FE calculations and burn-back routines are handled are discussed in Section 4.5.

Figure 2-36, Figure 2-37 and Figure 2-38 on the following pages, show the effects of the discussed propellant regression algorithm, using grid SG2-10 (see Figure 2-14). Figure 2-36 shows the grid initially, Figure 2-37 shows the grid with 28% of the propellant mass fraction "burnt" away, and Figure 2-38 shows the grid with 60% of the propellant burned away. Note that the directional unit vectors are shown in each of the figures; each pointing into and normal to the surface, except the two directional unit vectors that lie along the symmetry line (radially from the center of the motor). These two directional unit vectors are collinear with the symmetry lines. These two burnt back grids were created by setting the burning rate and time step to a constant value and allowing only the burn-back algorithms to run.

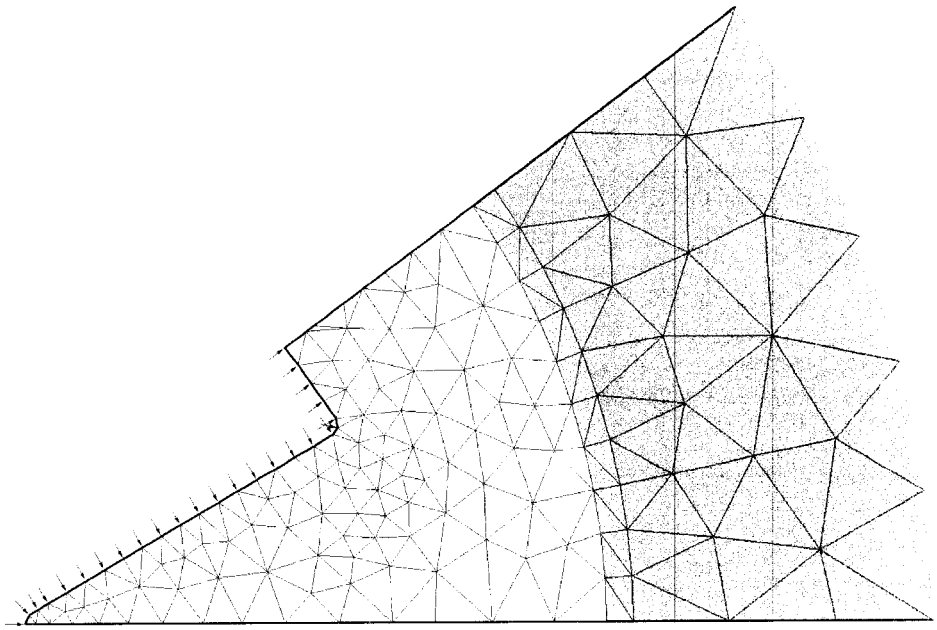


Figure 2-36 – 0% propellant mass fraction using grid SG2-10

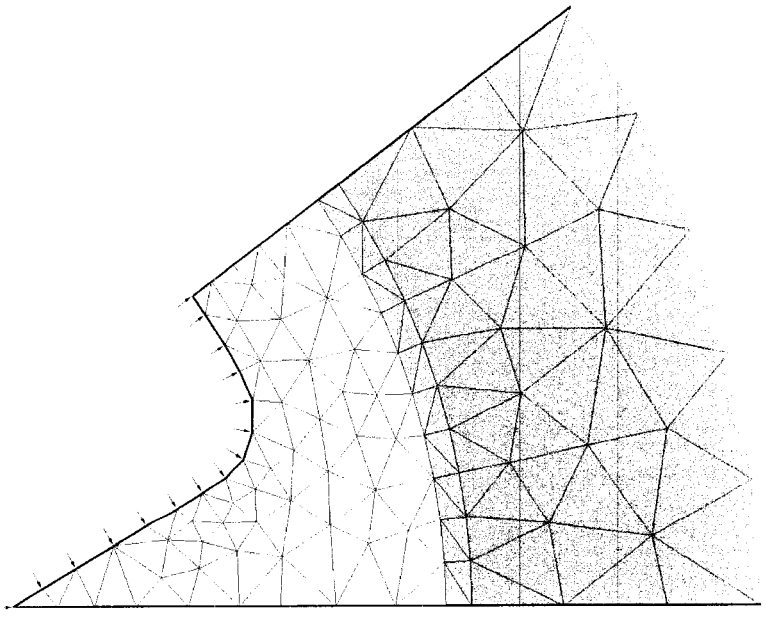


Figure 2-37 – 28% propellant mass fraction using grid SG2-10

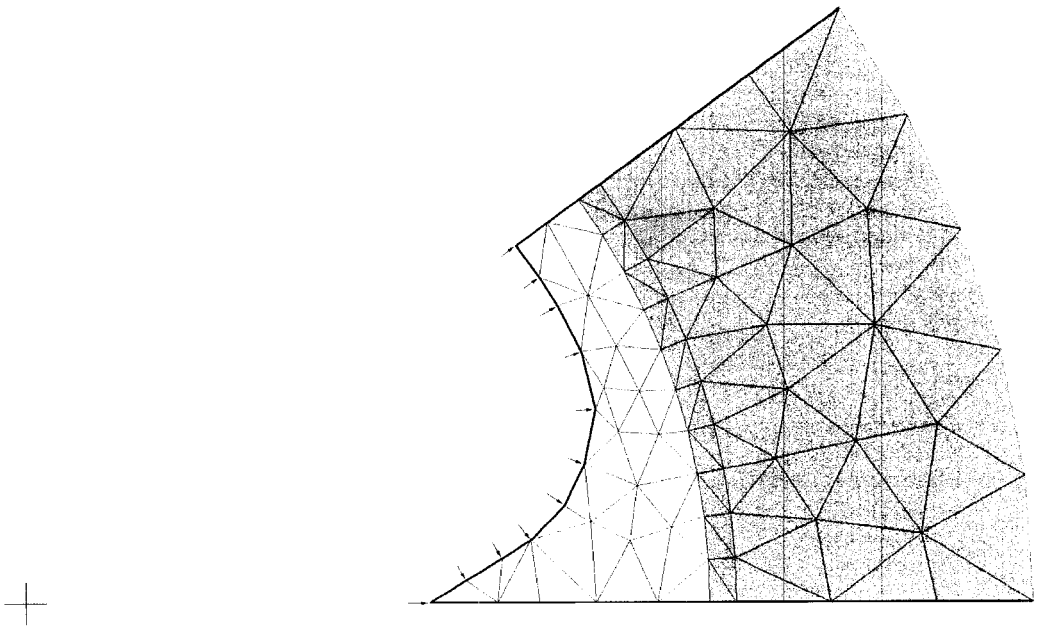


Figure 2-38 – 60% propellant mass fraction using grid SG2-10

3 Internal Ballistic Flow Modelling

3.1 Introduction

This chapter deals with the internal flow components of the simulation model. The internal flow inside the motor is simulated via a flow solver that uses a higher order version of the Random-Choice Method. This computational method was employed by Greatrix and Gottlieb to solve for the internal ballistics of an SRM.[21] In the following sections, the governing equations and a general outline of the method of solution will be discussed. A separate section is reserved for the equations and solution of the pyrolysis rate (burning rate) of the propellant. The burning rate is an important component of the internal ballistics, as well as being vital to the propellant regression calculations in the FE portion of the simulation.

3.2 Governing Equations

In comparison to typical one-dimensional gas dynamic conservation equations for duct flows, those used for the internal ballistics inside an SRM require additional terms to account for mass, momentum and energy addition from the propellant combustion. In addition, the flow inside an SRM typically consists of two phases – the gas phase and the particulate phase. Additional conservation equations are required to account for the particulate phase. However, in this study, the particulate phase was not accounted for as the experimental SRM data used for comparison involved a non-aluminized propellant, which has a low particulate fraction. Thus, the total number of equations and variables are reduced. For this thesis, the governing quasi-one-dimensional hydrodynamic conservation equations for the gas flow are:[21]

$$\frac{\partial \rho}{\partial t} + \frac{\partial (\rho u)}{\partial x} = -\frac{1}{A}\frac{\partial A}{\partial x}\rho u + (1-\alpha_p)\rho_s \frac{4r_b}{d} - (\frac{4r_b}{d} + \kappa)\rho \qquad (3.1)$$

$$\frac{\partial(\rho u)}{\partial t}+\frac{\partial}{\partial x}(\rho u^2+p)=-\frac{1}{A}\frac{\partial A}{\partial x}\rho u^2-(\frac{4r_b}{d}+\kappa)\rho u-\rho a_l-\frac{\rho_p}{m_p}D \tag{3.2}$$

$$\begin{aligned}\frac{\partial(\rho E)}{\partial t}+\frac{\partial}{\partial x}(\rho uE+up)&=-\frac{1}{A}\frac{\partial A}{\partial x}(\rho uE+up)-(\frac{4r_b}{d}+\kappa)\rho E\\&+(1-\alpha_p)\rho_s\frac{4r_b}{d}(C_pT_f+\frac{v_w^2}{2})-\rho ua_l-\frac{\rho_p}{m_p}(u_pD+Q)\end{aligned} \tag{3.3}$$

Note that there are some particulate phase parameters in the above equations (particle mass fraction α_p, flow density ρ_p, mass m_p, velocity u_p, drag force D and heat transfer Q). As mentioned, the particulate phase will not be accounted for in this thesis (hence α_p = 0); these terms are left in the equations for completeness. In Equations (3.1) to (3.3), the principle gas flow variables are ρ, p, u, and E, where ρ is the flow density, p is the pressure, u is the velocity and E is the total specific energy where $E=\frac{p}{(\gamma-1)\rho}+\frac{u^2}{2}$. The additional variables in Equations (3.1) to (3.3) such as the port area (A), port dilatation above that due to burn-back (κ), hydraulic diameter (d), longitudinal acceleration of the gas (a_l) and propellant burning rate (r_b) are considered separately from the above equations and discussed below. C_p, T_f, v_w and ρ_s are constants and refer to the heat capacity of the gas, flame temperature, flame front velocity and propellant density respectively.

The port area, hydraulic diameter and the port dilatation are calculated from the FE output of the simulation. The calculation of port area and hydraulic diameter are discussed in Section 4.2. The port dilatation is the dilatation of the flow at a given section due to structural deformation of the propellant and <u>not</u> due to propellant regression and is defined as $\kappa=\frac{1}{A}\frac{\partial A}{\partial t}$. The longitudinal acceleration is the acceleration of the gas resulting from structural oscillations in this direction arising from axial wave motion inside the combustion chamber. The burning rate is intrinsically coupled to the

conservation equations, and must be solved in conjunction with these equations. The burning rate equations are listed and discussed in Section 3.4.

The governing differential equations for the internal ballistics of an SRM have two basic components – the homogeneous and inhomogeneous parts, where here the homogeneous parts are on the left hand side and the inhomogeneous parts are on the right of Equations (3.1) to (3.3). The homogeneous components are essentially the same set of equations as those seen in constant-area duct flow problems. The inhomogeneous components are largely specific to SRMs and are related to the mass influx of gas from the propellant burning away, changes in the port area due to propellant regression and nozzle area transition, port dilatation from structural vibrations, and acceleration effects.

3.3 Random-Choice Method

The method of solution of the governing equations involves using a higher order explicit finite-volume method called the Random-Choice Method (RCM).[22] It is a technique for integrating hyperbolic sets of partial differential equations. The main difference between the RCM and typical Godunov based methods is that the RCM uses a pseudo-random sampling of flow properties at a given position within a wave profile as opposed to flow-averaging across an elemental flow section, as typically found in a Godunov scheme. This helps in reducing background noise, which is prominent in this application due to the spatial transitions in port area and mass influx from the inhomogeneous terms in the conservation equations.

The RCM selects a random position ζ in-between two nodes at the beginning of a set time step ($0 < \zeta < 1$). Flow property gradients (gas density, pressure and velocity) are then established from both left and right nodes to the intermediate random position using quasi-steady flow equations (Equations (3.1) to (3.3) with all time derivatives set to zero). Under transient conditions, there will likely be a jump discontinuity at the intermediate point producing some wave motion and the quasi-steady equations may

produce a choked flow condition before reaching the intermediate position. If a choked condition should arise, the quasi-steady flow solution would be replaced with either an isentropic compression or rarefaction wave (depending on the case) of certain strength such that the flow is just sonic at the intermediate position. This procedure is pictured in Figure 3-1.

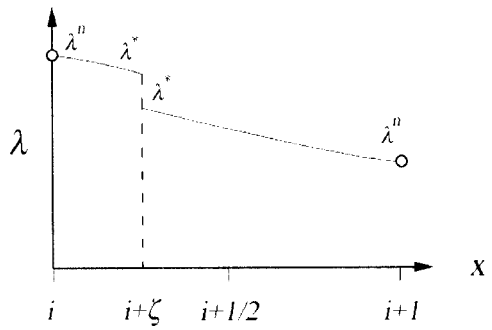


Figure 3-1 – Illustration of flow discontinuity

Then, a planar Riemann solution of any wave motion resulting from the jump discontinuity is calculated and the flow properties are ascertained at the mid-point between the left and right node and at the end of the half time step as shown in Figure 3-2. Of special note is that the nodes shift by $\Delta x/2$ (to the mid-point) for every half time step, so care must be taken not to lose track of the flow properties, as this would induce noise in the solution.

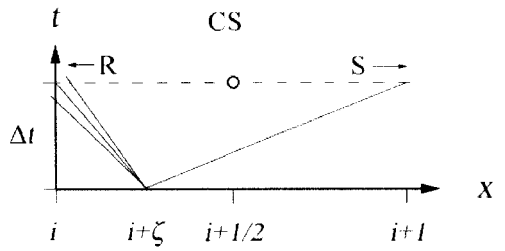


Figure 3-2 – Selection of flow properties

From here, a method described by Ben-Artzi and Falcovitz is used to incorporate the inhomogeneous terms into the planar wave solution.[23] This method

allows the wave solution to approach second-order accuracy as the time step becomes small. To calculate the time derivatives, a weak-wave Eulerian scheme is used. This simplifies matters, as dealing with left and right-running characteristic waves is simpler than dealing with the rarefaction wave and shock wave used in the Riemann solution. Therefore, with the inclusion of inhomogeneous terms, flow conditions are matched through the weak waves from left to right and a contact surface allows for the calculation of the time derivatives arising from the random position ζ. Since ζ will generally not coincide with the mid-point, the influence of the time derivatives that correct for wave-based inhomogeneous effects will occur over a shorter time span (Δt_{eff}) than the full half time step Δt. This is illustrated in Figure 3-3. Therefore, the wave-corrected flow properties may be found using

$$\lambda^* = \lambda_0^* + \Delta t_{eff} \left(\frac{\partial \lambda}{\partial t}\right)_0 \tag{3.4}$$

where λ is a flow property. Here, λ_0^* is the solution to the Riemann problem at the mid-point, and the time derivative is evaluated at the beginning of the time step at ζ. After obtaining the wave-corrected values for the flow properties λ^*, they must be modified for the background quasi-steady flow changes from the random position to the mid-node position as was done initially. Again, there exists the possibility that the local flow will choke, so an isentropic wave is placed such that the local flow is sonic at the mid-point.

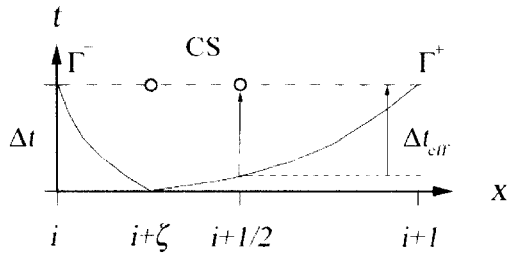


Figure 3-3 – Influence of time derivatives

This process is repeated for a second half-time-step. The second half time step will have the effect of repositioning the nodes to their original places (recall after the first half time step the nodes are shifted by $\Delta x/2$). In between half time steps, the FE portion of the simulation is run, providing the required parameters as output for the internal ballistic calculations to be run. Amongst this solution procedure, the pyrolysis rate calculations must be run as they are integral to the flow solution; they are described in the following section.

3.4 Pyrolysis Rate Calculations

The pyrolysis rate or burning rate of the propellant is an implicit part of the internal ballistic calculations. The burning rate is a quasi-steady, rapid-kinetic-rate model, which neglects transient effects. It consists of three components: pressure-induced burning, velocity-induced burning and acceleration-induced burning.

The pressure-induced burning rate follows the empirical law of de St. Robert:[24]

$$r_p = Cp^n \tag{3.5}$$

where r_p is the pressure-induced burning rate, n is the pressure exponent, p is the local static pressure and C is the burning rate coefficient. The burning rate coefficient is a function of the initial bulk temperature of the propellant and is found using[24]

$$C = C_0 \exp\left(\sigma_p \left(T_i - T_{io}\right)\right) \tag{3.6}$$

where σ_p is the pressure dependant burning rate temperature sensitivity, and T_i is the initial temperature of the propellant. T_{i0} and C_0 are the reference temperature and reference burning rate coefficient. The pressure exponent and reference burning rate coefficient are experimentally derived values (taken at the reference temperature) usually from a strand burner or small-scale motor test.

The velocity-induced augmentation of the burning rate is more commonly called erosive burning. According to the phenomenological model employed in this study, it is governed by the local convective heat transfer at the burning surface. This increases the heat flux back into the propellant, therefore increasing the local burning rate. The erosive burning model used in this thesis is shown below:[25,26]

$$r_b = r_0 + \frac{h(T_f - T_s)}{\rho_s[C_s(T_s - T_i) - \Delta H_s]} \tag{3.7}$$

where r_0 is the base burning rate component which includes pressure and acceleration effects, C_s is the propellant specific heat capacity, T_s is the surface temperature and ΔH_s is the surface heat of reaction of the propellant. The convective heat transfer coefficient (h) is a function of the core flow, and is defined as

$$h = \frac{\rho_s r_b C_p}{\exp\left(\frac{\rho_s r_b C_p}{h^*}\right) - 1} \tag{3.8}$$

where,

$$h^* = \frac{k}{d} \mathrm{Re}_d \, \mathrm{Pr}^{1/3} \frac{f}{8} \tag{3.9}$$

and

$$f^{-1/2} = -2\log_{10}\left(\frac{2.51}{\sqrt{f}\,\mathrm{Re}_d} - \frac{\varepsilon/d}{3.7}\right) \tag{3.10}$$

k is the thermal conductivity of the gas, f is the Darcy-Weisbach friction factor inside the port, Re_d is the local Reynolds number based on the hydraulic diameter d and the core flow velocity, Pr is the Prandtl number and ε is the surface roughness of the

propellant. This erosive burning model is suitable for larger length-to-diameter ratio (typically over 6 L/D) motors where the internal quasi-steady velocity (and erosive burning) becomes significant towards the aft end of the motor.[26]

Acceleration-based effects are of a prime interest in this thesis. Augmentation of the burning rate due to an acceleration field is based on a phenomenological model, which combines normal, lateral and longitudinal acceleration effects.[5,27] The acceleration augmented burning rate is defined as

$$r_b = \frac{\beta\left(r_b + G_a/\rho_s\right)}{\exp\left[\frac{C_p\delta_0(\rho_s r_b + G_a)}{k}\right]-1} \tag{3.11}$$

where β is the heat flux coefficient, δ_0 is the reference energy film thickness and G_a is the accelerative mass flux which are all defined by Equations (3.12), (3.13) and (3.14) respectively:

$$\beta = \frac{C_p(T_f - T_s)}{C_s(T_s - T_i) - \Delta H_s} \tag{3.12}$$

$$\delta_0 = \frac{k}{\rho_s r_0 C_p}\ln(1+\beta) \tag{3.13}$$

$$G_a = \frac{a_n p}{r_b}\frac{\delta_0}{RT_f}\frac{r_0}{r_b} \tag{3.14}$$

where a_n is the acceleration normal to the surface and r_0 is the base burning rate which in this instance is due to pressure and core flow effects. The accelerative mass flux defined in Equation (3.14) accounts only for accelerations normal to the burning

surface. To account for lateral and longitudinal accelerations, a term that considers the orientation of the acceleration vector is required, and is defined by[5]

$$G_a = G_{a0} \cos^2 \phi_d \tag{3.15}$$

where G_{a0} is the peak accelerative mass flux defined by Equation (3.14), and ϕ_d is the displacement angle and is defined as

$$\phi_d = \tan^{-1}\left[K\left(\frac{r_0}{r_b}\right)^3 \frac{a_l}{a_n} \right] \tag{3.16}$$

where K is an overall orientation correction factor (experimentally derived) and a_l is the vector sum of the lateral and longitudinal accelerations. K for this thesis is set at 8.[5] The displacement angle is treated as an augmented orientation angle; the normal orientation angle is ascertained by

$$\phi = \tan^{-1}\left(\frac{a_l}{a_n}\right) \tag{3.17}$$

The orientation angle (ϕ) is the angle off normal at which the overall acceleration vector is oriented towards. This is illustrated in Figure 3-4.

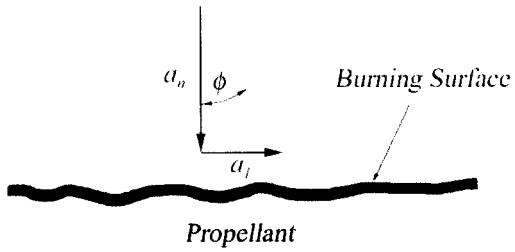


Figure 3-4 – Acceleration vector orientation

Note the direction of the normal component of acceleration is directed into the propellant surface. There is only acceleration-based augmentation when the normal acceleration is negative, such that the combustion layer is being compressed.[5] If the normal acceleration is positive, acceleration effects are assumed to be negligible. Lateral and longitudinal accelerations are sign independent and tend to reduce the amount of augmentation produced by the normal acceleration.

It is evident from Equation (3.7) to Equation (3.16) that there is a lot of nonlinear co-dependence amongst these equations. This is apparent specifically in the base-burning rate (r_0), which is in both the erosive burning equations and the acceleration augmentation equations. As mentioned before, the base burning rate in either the core flow or acceleration equations represents the burning rate effects from the other two burning rate components. This requires a unique method of solution.

The pressure-induced burning rate (Equation (3.5) and Equation (3.6)) is calculated first, as it is dependent only on the local chamber pressure. Using the pressure dependent burning rate only as the base burning rate (no acceleration component), the erosive burning calculations are in turn performed. Since the erosive burning equations (Equation (3.7) to Equation (3.10)) are nonlinear, a Newton-Raphson numerical method is used to solve for the burning rate. The Newton-Raphson method is a standard method of solving nonlinear equations and is shown below:[18]

$$r_b^i = r_b^{i-1} - \frac{f\left(r_b^{i-1}\right)}{f'\left(r_b^{i-1}\right)} \quad , \text{for } i \geq 1 \tag{3.18}$$

where f and f' is Equation (3.7) and its derivative with respect to r_b. Equation (3.18) is iterated until the difference between r_b^i and r_b^{i-1} is less than some specified value.

Once the velocity-induced component of the burning rate is found, the normal acceleration component of burning rate may be solved. Recall this is done only if the normal acceleration component is negative. The method of solution is the same as that

for erosive burning (using a Newton-Raphson method), as Equation (3.11) is also nonlinear. The base burning rate now includes pressure and velocity components, f is Equation (3.11) and f' is its derivative with respect to r_b. The solution is iterated again until convergence. However, it is possible that the solution will not converge as the original base burning rate used to solve for the erosive burning did not include acceleration effects. This would be expected when the acceleration augmentation is high. If convergence is not reached, the erosive burning loop is once again entered using the sum of the current value of the burning rate due to acceleration and the pressure burning rate as the base-burning rate. This whole process is repeated until convergence is reached.

Once the burning rate due to pressure, velocity and normal acceleration are solved for, the lateral and longitudinal components of acceleration are included. It is tempting to include Equation (3.15) in the original acceleration augmentation calculations; however, this can cause convergence difficulties. Regardless, the lateral and longitudinal acceleration components are included as geometric corrections so they are most easily incorporated afterwards. Again, the equations are nonlinear and a Newton-Raphson method is employed. Of course, this time Equation (3.15) is substituted into Equation (3.11) and iterated until convergence is reached. This whole process is illustrated below in Figure 3-5.

This algorithm is easily suited to one section of the SRM where there is one value for pressure, velocity and longitudinal acceleration, but there are many values of lateral and normal acceleration along the Control Boundary (CB) segments. The algorithm flow charted in Figure 3-5 is simply applied at each individual segment. This will provide a unique burning rate to every segment around the periphery.

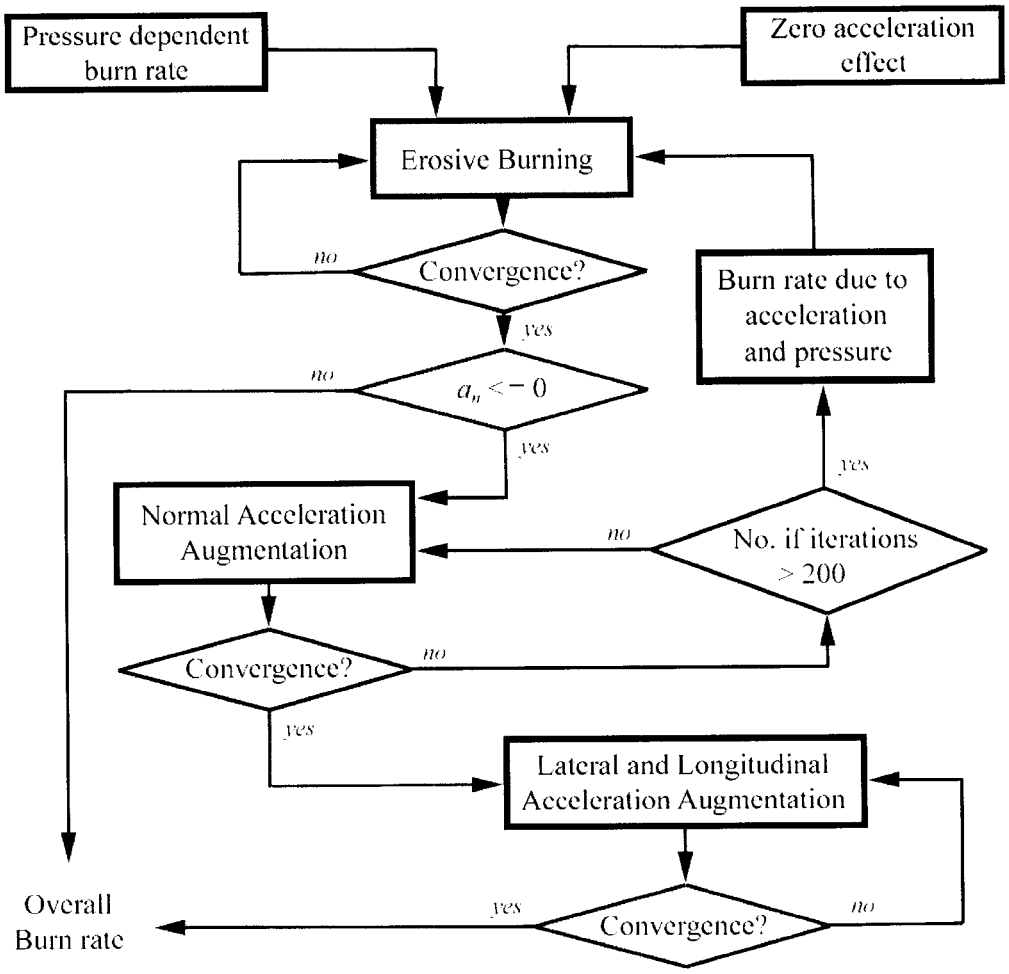


Figure 3-5 – Pyrolysis rate algorithm flow chart

4 Component Integration

4.1 Introduction

This chapter deals with the details of integrating the Finite Element (FE) components and the Internal Ballistic Flow (IBF) components. The various boundary conditions that are required to be passed between FE and IBF components and their significance are discussed. In addition, an averaging method is employed to smooth out spurious noise in the pressure boundary condition, details of which are explained and some results are presented. The evaluation of the time step of the simulation and details on its employment, are discussed. Finally, the solution procedure that is used in this simulation is explained in detail and illustrated in a flow chart.

4.2 Passing Boundary Conditions

In this simulation, there are a few boundary conditions being passed between both the FE components and the IBF components. These boundary conditions must be passed from component to component in the correct format ensuring continuity between modules. The FE components require the chamber pressure and burn-rate from the IBF component, and in turn, the IBF component requires the section periphery length, segment length and number along the periphery, port area and the normal and lateral accelerations of the segments, from the FE component.

Some of the IBF module boundary conditions do require a bit of care when passing from the FE component. The length of each segment around the Control Boundary (CB) is easily found using,[15]

$$l_{segment} = \sqrt{[(x_2 + u_2) - (x_1 + u_1)]^2 + [(y_2 + v_2) - (y_1 + v_1)]^2} \qquad (4.1)$$

where $l_{segment}$ is the segment length, x_n and y_n are the x and y coordinates of node n and u_n and v_n are the x and y displacements of node n. (n = 1, 2) The overall periphery length of the section is found by adding up the segment lengths. In doing this, the number of segments per section along the CB may also be counted. Note that if symmetry is exploited, the periphery length must be divided by the fraction of which the grid was reduced with, to get the proper answer.

The port area is found by finding the area of a triangle bounded by the two points in each segment and the center of the system (the origin) as the third, and adding up these areas. This calculation simply involves finding half of the magnitude of the cross product between two vectors representing only two sides of the triangle,[15]

$$A_i = \frac{1}{2}\left|(x_1 + u_1)(y_2 + v_2) - (y_1 + v_1)(x_2 + u_2)\right| \tag{4.2}$$

then adding up all the contributions from all the segments to find the port area. This calculation is illustrated in Figure 4-1. Again, if symmetry is exploited, the resulting port area must be divided by the fraction of which the grid was reduced. Note that both

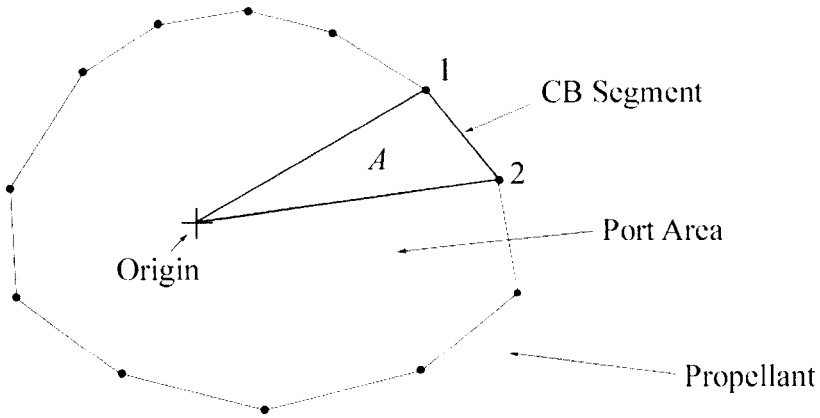


Figure 4-1 – Calculation of port area

Equations (4.1) and (4.2) include the nodal displacements in their calculations. This will account for changes in port area and periphery length due to structural deformation that will affect mass-flow and other internal ballistic calculations in the IBF solver.

The most important boundary conditions in this simulation model however are the local normal and lateral accelerations along the CB segments. For the normal acceleration, it is important that the sign convention, as was illustrated in Figure 3-4, is followed. Since only negative normal accelerations affect the burning rate, the component of the acceleration vector (for any one CB segment) that lies along the unit burn vector (see Section 2.7) is used in the acceleration augmented burn-rate calculations. Thus, a simple dot product is taken between the directional unit vector (recall from Section 2.7: $\{u\}_i = \begin{Bmatrix} u_x & u_y \end{Bmatrix}_i$) and the acceleration vector:

$$a_n = a_x u_x + a_y u_y \tag{4.3}$$

Note that by this method the magnitude of the normal acceleration will always be negative, so long as some component of the acceleration vector can be projected along the unit burn vector. The direction of the lateral acceleration component is unimportant and is found by finding the component of the acceleration vector that lies perpendicular to the unit burn vector. Similar to the normal acceleration, the lateral acceleration is found by taking the magnitude of the cross product between the acceleration vector and the directional unit vector:

$$a_{lat} = a_x u_y - a_y u_x \tag{4.4}$$

In the acceleration-based burn-rate calculations, the lateral and longitudinal acceleration components are first vectorially combined, then used in Equation (3.16) as the variable a_l. The longitudinal acceleration is currently calculated as a part of the IBF solution.

4.3 Time Step

The time step of the simulation must be carefully considered. Both the transient FE and IBF calculations are explicit and therefore are only conditionally stable. The maximum FE time step is defined by Equation (2.67) and restated here:[11]

$$\Delta t_{\max_{FE}} \leq C_{n_{FE}} \frac{\Delta l_{\min}}{\sqrt{\frac{E}{\rho(1-\upsilon^2)}}} \quad \text{, where } C_{n_{FE}} \leq 1 \tag{2.67}$$

Equation (2.67) however does not define the maximum time step allowed for the IBF solution. Time steps in computational fluid dynamic solutions do however follow a similar ideology to the origin of the FE maximum time step. They are also based on the maximum speed at which information may travel through the domain (the CFL condition). In fluid mechanics, the speed information travels through the medium is a function of the wave speed (c) defined as,

$$c = a + U$$

where a is the sound speed of the fluid and U is the velocity of the fluid. Thus, using methodology similar to that used for Equation (2.67), the maximum time step for the IBF calculations may be defined as:[28]

$$\Delta t_{\max_{IBF}} \leq C_{n_{IBF}} \frac{\Delta l_{\min}}{c} \quad \text{, where } C_{n_{IBF}} \leq 1 \tag{4.5}$$

In Equation (4.5), $\Delta t_{\max_{IBF}}$ is the time it takes a wave of speed c to travel a distance $\Delta l_{\min}$. The grid for the IBF calculations is linear (one-dimensional), so $\Delta l_{\min}$ is simply the distance between nodes. $C_{n_{IBF}}$ is the Courant number for the CFD calculations and has the same purpose as it does in Equation (2.67) and discussed in Section 2.4.4. $C_{n_{IBF}}$

is set to 0.5 for this simulation. For the FE time step, $C_{n_{FE}}$ is set for each specific grid such that performance and accuracy are maintained (see Section 2.4.4).

Having the maximum time step defined for both the FE and IBF solvers, the simulation time step must be chosen that satisfies both Equation (2.67) and Equation (4.5). Since the FE time step is generally smaller (from experience) than the IBF half time step, the FE time step could be used as the simulation time step. However, there are other considerations; time synchronization between the FE and IBF components must be factored in as well.

Time synchronization between the FE and IBF modules is maintained by finding a positive integer (R) that best approximates the ratio between the IBF half time step and the FE time step. Knowing this integer, the FE time step may be rescaled such that it equals the IBF half time step divided by the integer. This integer represents the number of FE time steps that must be run for every IBF half time step. A simple algorithm to find this integer is shown below:

$$R = trunc\left(\frac{\Delta t_{\max_{IBF}}}{2\Delta t_{\max_{FE}}} + 1\right) \tag{4.6}$$

For example, if the ratio between the IBF half time step and FE time step is 8.246..., R is effectively rounded up to 9 using Equation (4.6). Knowing R, the FE time step may be rescaled as:

$$\Delta t_{FE} = \frac{\Delta t_{\max_{IBF}}}{R} \tag{4.7}$$

Equation (4.7) ensures synchronicity between the FE and IBF components even if the FE time step is larger than the IBF half time step.

This algorithm generates optimal time steps for the simulation while maintaining synchronicity between the FE and IBF components when $\Delta t_{\max_{FE}} \leq \Delta t_{\max_{IBF}}$, and ensures that Equation (2.67) and Equation (4.5) are both satisfied. If $\Delta t_{\max_{FE}} > \Delta t_{\max_{IBF}}$, then the above algorithm will yield $\Delta t_{FE} = \Delta t_{\max_{IBF}}$. Thus, there will be more FE time steps than necessary; however, there must be a current (synchronized) FE solution for every IBF half time step, and so this algorithm suits the situation well. Besides, the FE time step is smaller that the IBF half time step in most of the cases used in this thesis.

4.4 Pressure Control

Mentioned in the previous section, an issue must be accounted for before the pressure values may be input as a boundary condition into the FE calculations. The structure of this simulation is such that the FE portion of the simulation is run in-between the half-time-steps of the IBF portion, such that before the next half-time-step is run, the FE and IBF solutions are synchronized. This structure has a problem that is likely inherent to the RCM in the IBF component. Every half time step, the RCM shifts its nodes by $\Delta x/2$; thus, the value of pressure for any one section will change slightly from one half time step to the next. This could create oscillations in the pressure at any one section along the length of the motor (as the FE sections along the axis of the motor do not shift by $\Delta x/2$). This is undesirable, as it will generate numerical noise in the FE solution. The plot in Figure 4-2 shows the pressure values at one section in the motor over a short time span as output from the IBF solver. It is evident from the magnitude of the pressure oscillations in this graph that control of these oscillations is required.

One way to reduce these oscillations is to average the current value and the value from the previous time step such that:

$$p^{t}_{FE_{Ave}} = \frac{p^{t}_{IBF} + p^{t-\Delta t}_{IBF}}{2} \tag{4.8}$$

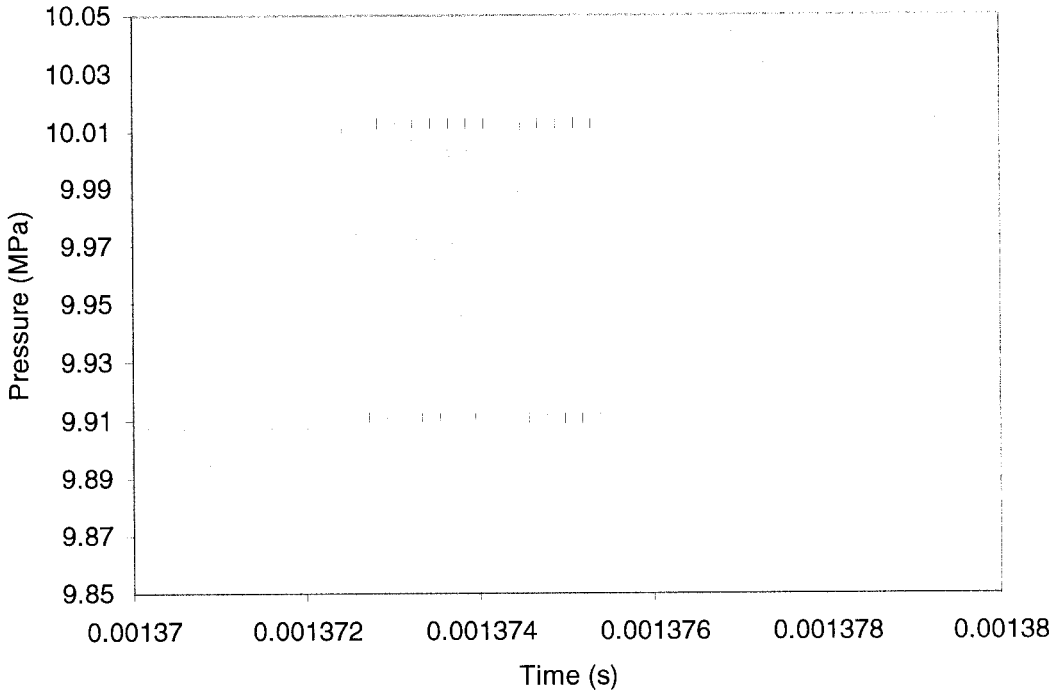


Figure 4-2 – Pressure oscillations from the RCM

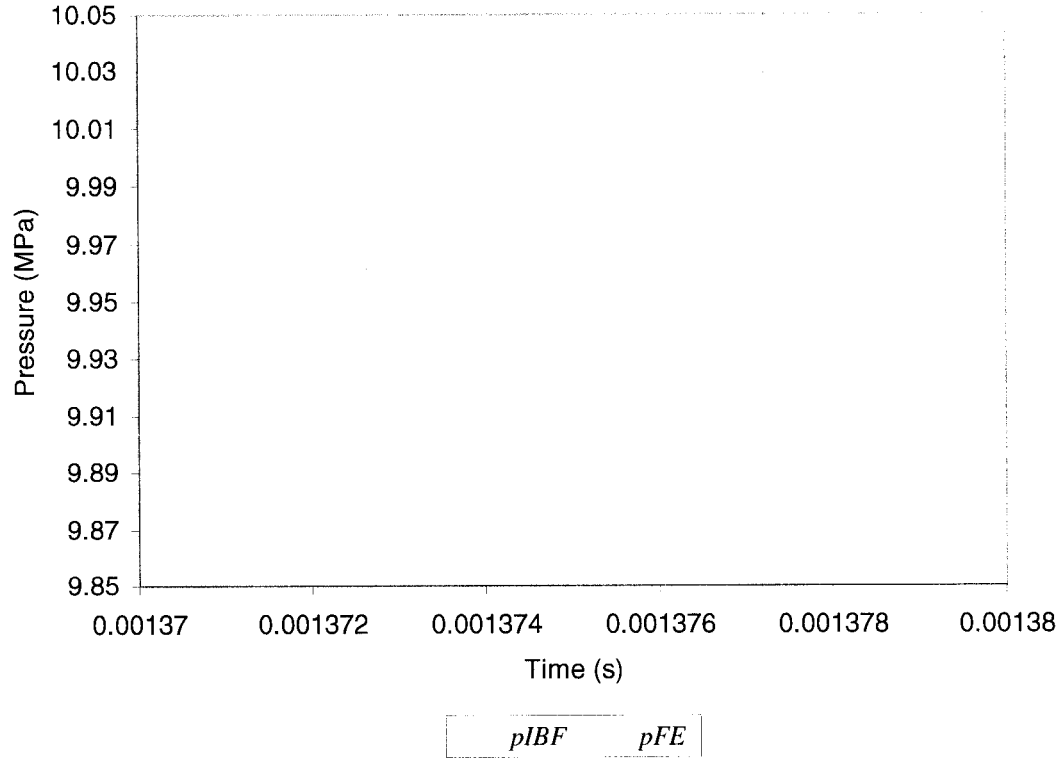


Figure 4-3 – Averaged pressure values

In Equation (4.8), the subscript *IBF* denotes the values output from the IBF solver and the subscript FE_{Ave} denotes the averaged value to be used as the boundary condition FE solver. Appling Equation (4.8) to the pressure curve in Figure 4-2 yields the results shown in Figure 4-3. Here the oscillations are effectively damped out from the original pressure values (*pIBF*) and a smooth pressure value for the section remains (*pFE*) (Note: $pFE = p_{FE_{Ave}}$ and $pIBF = p_{IBF}$). This will help minimize the noise associated in the FE solution due to this boundary condition.

Looking at Figure 4-3, there is a stepping of the averaged pressure values every so often. The reason for this is unclear and it doesn't seem to affect the IBF calculations to any degree. However, this stepping will affect the FE module generating noise in the solution, as the pressure doesn't increase at a steady rate from one IBF half time step to the next. An attempt is made to smooth out the averaged pressure values in order to keep the changes in pressure as smooth as possible (as seen by the FE module) without impairing accuracy.

An algorithm is created to provide a lag between the initial and final value at a step in pressure. It is based on a linear interpolation between the beginning and end of an IBF half time step and the assumption that $p^{t}_{FE_{Ave}}$ remains relatively constant for several half time steps as observed in Figure 4-3. This algorithm is shown below in Equation (4.9).

$$p^{t}_{FE} = p^{t-\Delta t}_{FE} + \frac{1}{D_p}\left(p^{t}_{FE_{Ave}} - p^{t-\Delta t}_{FE}\right)\frac{r}{R} \tag{4.9}$$

The amount of lag is controlled by the constant D_p termed a "lag parameter". Note that R in Equation (4.9) is the same as that in Equation (4.6). Having r / R in the second term in Equation (4.9) interpolates the pressure value (linearly) for the FE component in-between the IBF half time steps (again to not have sudden stepped pressure increases); r is an integer counter that counts the number of FE time steps that have elapsed since the last IBF half time step such that $r = 1, 2, \ldots, R$.

The effects of Equation (4.9) are plotted in Figure 4-4 using various values of D_p. The effects of the lag parameter are clear. Note that when D_p = 1, the resulting curve is identical to the *pFE* values in Figure 4-3. Although there is a discontinuity in the lagged pressure at the step, this has little effect on the FE solution in comparison to the larger original step in pressure. With this lag, the changes in pressure are less severe for each FE time step, yet the true value of the averaged IBF pressure values (D_p = 1) is realized. There is however a danger that the averaged IBF pressures may change before the lagged pressures reach the averaged IBF values – especially if the value of the lag parameter is greater than about 6. Therefore, the lag parameter should not be made too large in order to maintain consistency with the averaged IBF pressure values.

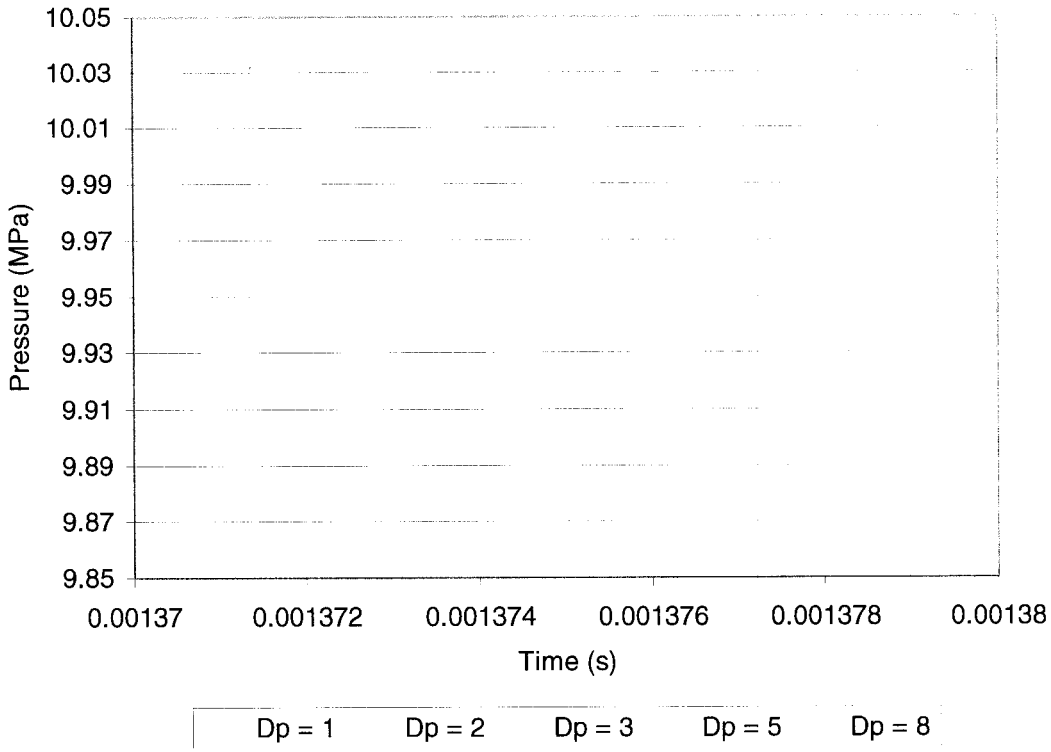


Figure 4-4 – Lagged pressure values

With this in mind, the lag parameter for all the simulations run in this thesis, is set to 6. As mentioned earlier in this section, it is unclear as to why the pressure output

from the IBF portion of the simulation steps in this manner. No attempts were made to find the source of the problem in the IBF solver – if it even is a problem and not simply a characteristic of the RCM solution of the flow with time.

4.5 Procedure

The purpose of this section is to show how all the components discussed thus far come together to complete the simulation. This is most easily done through a flow chart, which is shown in Figure 4-5. As with any computer program, there are initialization routines that must first be run. The first are the grid input and initialization routines. These routines input the grid from its formatted text file and generate a node list and element list from it. These lists include connectivity information that links the specific nodes to an element. If any symmetry in the mesh is exploited, it is also done at this time. Grid initialization is followed by the FE and IBF initialization routines. These routines allocate memory, initialize arrays, generate the CB properties for the grid, copies the FE grid to all the required sections, input any data files that would list constants, control parameters, etc., and establishes the proper links between the FE and IBF components. Having the simulation initialized, the program may enter the main simulation loop. Note however, at this point, any harmonic testing that needs to be done, is done at this point. Although not a part of the actual simulation, harmonic testing is typically done on a new geometry. Once done, the resonant frequency is set into a data file which is read upon program initialization, and no further resonant testing is needed.

Each cycle of the simulation is identical to the next, so there is no difference in the first time step from the next. Therefore, this procedure applies to all time steps. First, the structural properties for each section along the longitudinal axis of the motor are evaluated (see Section 4.2). These are in turn passed to the IBF solver and a half time step is taken. Note that the IBF solver has the capability of solving both unsteady and quasi-steady internal ballistic flows (the quasi-steady differential equations are the same as Equations (3.1) through (3.3), but with the time derivatives set to zero). Quasi-

steady calculations are used to speed up the simulation at points where there is no need for the more computationally expensive unsteady calculations (i.e., when there is no wave motion inside the motor). Once the IBF solver has completed its solution for the flow properties and the burning rate for all the segments in each section at a new half-time-step, the FE time step is evaluated as explained in Section 4.3. Knowing the number of FE time steps to be taken before the next IBF half time step, the Pressure Control routines can be run on the pressure output from the IBF component of the simulation.

Everything in the simulation is now ready for the FE solver to be run. A finite element solution must be made for every section along the motor at any single time step. So, the FE solver is put into a loop until a solution is created for all the sections. The FE solver consists of routines that assemble the external and internal force vectors and the mass matrix (as discussed in the sub-sections of Section 2.4), solve for the displacement, velocity and acceleration vectors using Newmark's Method (Section 2.4.4), apply the appropriate boundary conditions (Section 2.4.5) and regress the propellant (Section 2.7).

Once FE solutions are generated for all the sections, time is incremented and the graphics routines are run. The graphics routines are written using OpenGL libraries. These libraries relieve the central processor from handling the graphics routines, which are instead handled by the graphics processor. This greatly increases the available CPU resources for the simulation. Data output routines are also typically run alongside the Graphics output routines.

At this point, the FE solutions are completed for that time step. A Boolean statement is posed to see whether more FE time steps must be run in order to bring the FE solution time in synchronization with the IBF solution time. If $r \leq R$ (r and R are the same as those in Equation (4.9)), then more FE time steps must be run and the program increments r, and loops back to the Pressure Control routines for another FE time step. Otherwise, r is reset to 1 and the next cycle of the simulation is entered where the next

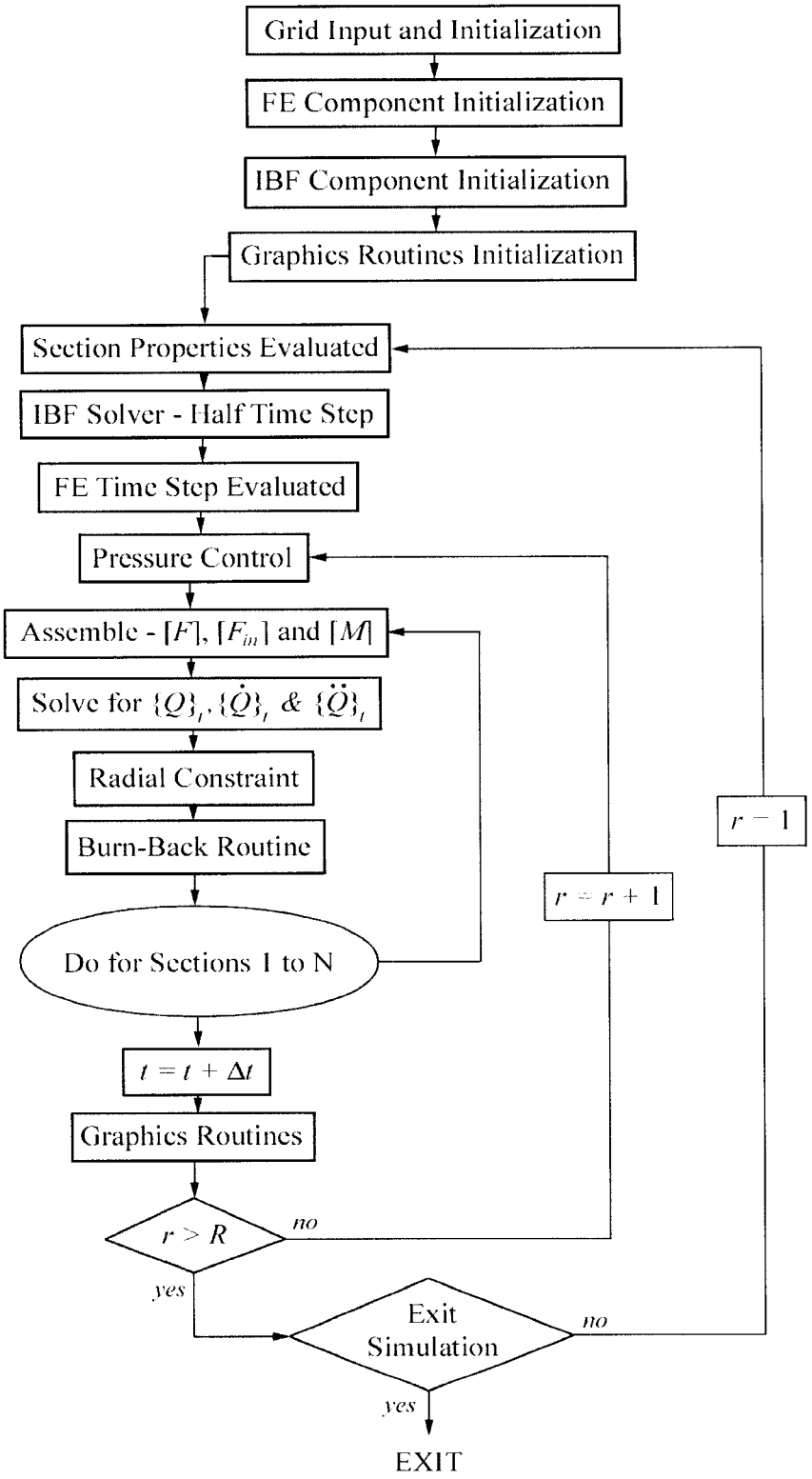


Figure 4-5 – Simulation flow chart

IBF half time step is run. This process is repeated until a manual or scheduled interrupt is entered to stop the simulation.

A complete simulation would typically consist of quasi-steady IBF calculations along with propellant regression and no FE calculations, up to a predetermined point when unsteady and FE calculations are commenced and allowed to stabilize shortly before a pressure pulse is introduced into the SRM. Doing this reduces computer time significantly as unnecessary calculations are avoided. Note that although the burn-back routines are still run, the grid modification routines are halted. As mentioned in Section 2.7, the elements used in this simulation cannot account for the transient changes due to grid modifications, and so the mass and stiffness properties are frozen for the duration of the simulation once the FE calculations are started.

Once a pressure pulse is delivered, the simulation is allowed to run until it has progressed typically 20 – 40 ms in simulated firing time. Data collected from the simulation would include displacement and acceleration values at key points in the structure, head-end pressure, and burn-rate information from various sections. The data is stored in a text file and may be plotted in any graphing program.

5 Cylindrical-Grain SRM Results

5.1 Introduction

The purpose of this chapter is to compare the results from this simulation to the experimental and numerical results found by Greatrix and Harris for a cylindrical-grain SRM.[2] This comparison will help prove the validity of the simulation as a whole. Additionally, it will allow a comparison of the differences that do appear between the simulation and the results found in Reference 2 – as far as the structural model is concerned (the internal ballistic simulation model in Reference 2 is identical to the one used in this thesis – barring modifications to integrate it with the FE components). Other output will include a burning rate augmentation graph, which will allow an observation on the effect of acceleration augmentation on burning rate with regards to the normal (radial) acceleration.

The motor is pulse-triggered into instability using a 1.21 MPa traveling pressure wave. The pulse has a compression wave front and an expansion tail. The pulse is introduced at a time identical to that in Reference 2 (90.5 ms into the simulated firing), after the unsteady calculations have been run (to settle out) for 10 ms (recall in Section 4.5, that to save on computer time, quasi-steady calculations are done in the first part of the simulation). The simulation is left to run for another 25 ms after the pulse is triggered.

The motor has the same geometry and material properties as the thin-sleeved motors tested in Reference 2. The dimensions of the motor and the characteristics of their materials are presented in Table 2-1. The grid used in the FE components of the simulation is a $1/20^{th}$ section of grid CG3 shown in Figure 2-5 (labelled CG3-20 to differentiate it from the whole grid CG3). There are 260 nodes in the IBF solution, 120 of which represent the propellant grain, so there will be 120 copies of grid CG3-20. For the purposes of identification, the numerical output presented from Reference 2 is

labelled "UTROC", and the numerical output from the simulation created for this thesis is labelled "Q3DSRM".

5.2 A Comparison of Results

In addition to the cylindrical-grain motor data displayed in Table 2-1, a few more motor characteristics are listed in Table 5-1.

Sectional resonant frequency (f_n)	14927	Hz
FE sectional damping ratio (ζ)	0.35	-
Propellant grain length (L_p)	518	mm
Nozzle throat diameter (d_t)	16	mm
Grain/nozzle conv. length ratio (L_p/L_c)	16	-
Pressure-dependent burning rate (r_p)	$0.0005[p(kPa)]^{0.35}$	m/s
Propellant specific heat (C_s)	1500	J/(kg-K)
Propellant flame temperature (T_f)	3000	K
Propellant surface temperature (T_s)	1000	K
Initial propellant temperature (T_i)	294	K
Propellant surface roughness (ε)	400	μm
Gas specific heat (C_p)	2000	J/(kg-K)
Gas Prandtl number (Pr)	0.8	-
Specific gas constant (R)	320	J/(kg-K)
Gas thermal conductivity (k)	0.2	W/(m-K)
Gas absolute viscosity (μ)	8.07×10^{-5}	Pa-s
Gas specific heat ratio (γ)	1.2	-
Particle mass fraction (α_p)	0 %	-
Casing/Prop. long. damping ratio (ξ_L)	0.1	-
Net surface heat of reaction (ΔH_s)	0	J/kg

Table 5-1 – Cylindrical-grain SRM characteristics

These characteristics are typical for a nonaluminiumized ammonium perchlorate/hydroxyl-terminated polybutadiene (AP/HTPB) propellant. Note that the sectional resonant frequency is higher than the value for grid CG3 shown in Table 2-1. This is because in these simulated firings, some of the propellant will have burned away which will increase the resonant frequency of the motor structure. Also, the resonant frequency is evaluated at the mid-point along the propellant grain; because the propellant, mainly due to erosive burning, will regress at various rates along the long axis of the grain, each section will have a slightly different resonant frequency. Thus, measuring the resonant frequency at the mid-point of the grain will provide a sort of averaged value for all the sections. The burning rate coefficient and pressure exponent seen in Equation (3.5) are combined in Table 5-1 into the pressure-dependent burning rate (r_p).

Figure 5-1 and Figure 5-2 show the head-end chamber pressure for the Q3DSRM and UTROC simulations respectively. Both the Q3DSRM and UTROC output have a similar overall dc pressure shift. The limit-cycle wave amplitude for the Q3DSRM simulation varies between approximately 3.5 – 4.0 MPa as opposed to 4.5 MPa in the UTROC simulation. It should be noted that the approximated resonant radial frequency employed in Reference 2 was approximately 25% higher, which may account for at least some of this difference. The lower limiting wave amplitude of the Q3DSRM simulation is in closer correspondence to the experimental results (3 MPa) seen in the head-end chamber pressure presented in Reference 2.

Figure 5-3 and Figure 5-4 plot the mid-point acceleration at and normal to the propellant surface (radial direction) for both the Q3DSRM and UTROC simulation respectively. Recall that only the negative values augment the burning rate as was discussed in Section 3.4. However, both positive and negative values are plotted for completeness. The Q3DSRM acceleration cycles have more low-level noise in between peaks. This would be expected from having many nodes along the burning surface in the FE section, which could all affect each other as opposed to the one-dimensional approach used in UTROC. This noise is also observed in Figure 5-1, where there is also

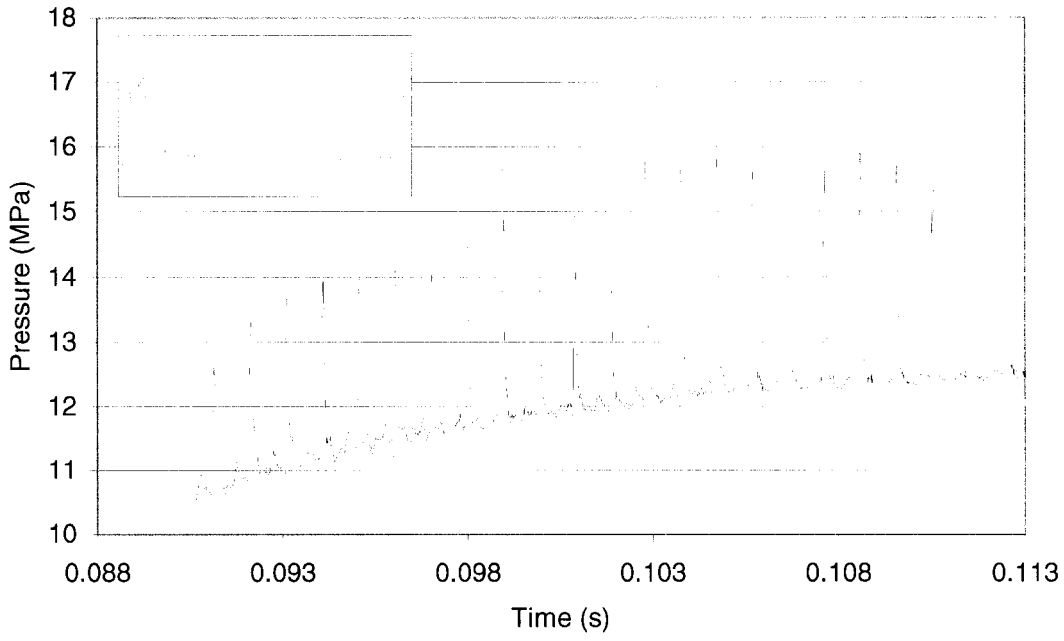


Figure 5-1 – Head-end chamber pressure (Q3DSRM)

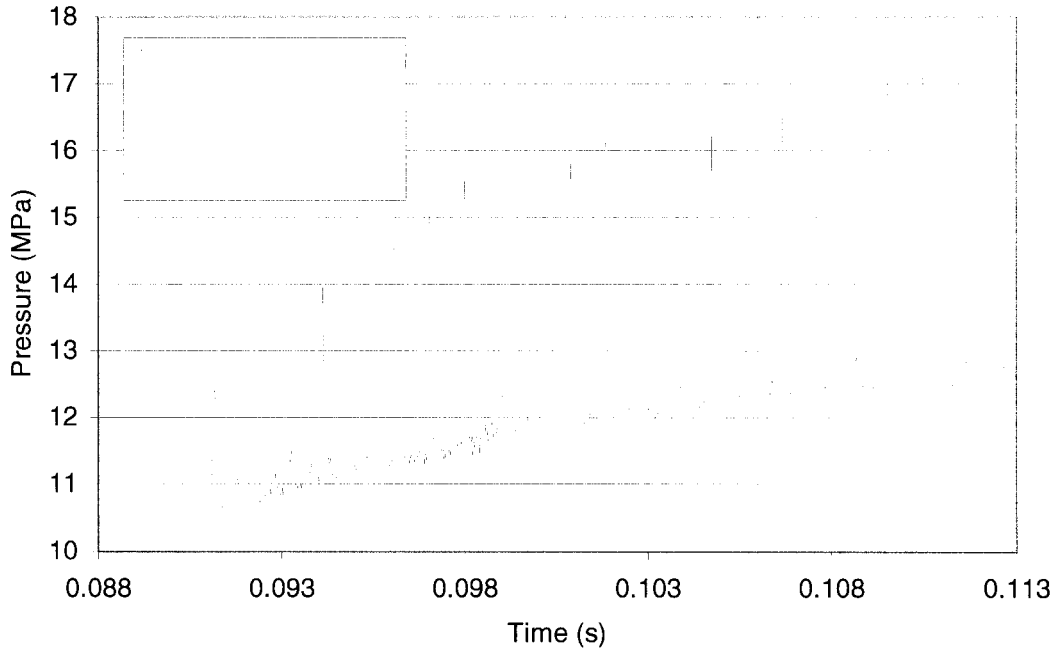


Figure 5-2 – Head-end chamber pressure (UTROC)

more low-level noise in between cycles compared to the UTROC head-end pressure output in Figure 5-2. The positive acceleration peaks in the Q3DSRM simulation are higher than in the UTROC simulation. It is unclear physically why this would happen; it may be likely that this is algorithmic in nature, or due to the damping model used. Nonetheless, it is most likely due to the different structural modelling technique used in the Q3DSRM simulation model with respect to the UTROC model. The negative acceleration peaks in Q3DSRM are only slightly lower than the values observed in the UTROC simulation. Thus, the burning rate augmentation in the Q3DSRM simulation will be similar to the UTROC simulation. No experimental correlation can be made with the propellant surface accelerations from Reference 2, as it was not possible to measure this property.

The mid-point external sleeve accelerations (normal to the surface; i.e., radial) are presented in Figure 5-5 and Figure 5-6. In Figure 5-5, the positive acceleration peaks are still higher than the negative peaks, however not to such a large degree as seen in Figure 5-3. There is better correlation between the Q3DSRM and UTROC external accelerations; however, the Q3DSRM accelerations still exhibit more low-level noise in between peaks. Nevertheless, the Q3DSRM predicted peak external accelerations, which are in the range of +1500/-1000 g, are similar to the experimental values seen in Reference 2 and provide a better overall correlation than the UTROC values.

Finally, Figure 5-7 shows the burning rate of the propellant at the mid-point in the SRM. The pressure-dependent burning rate is visible from the outline at the bottom of the curve; note how it increases momentarily as a wave passes. The velocity- and acceleration-dependent burning rate components are visible through the peaks in the curve in Figure 5-7. However, most of the burning rate increase comes from acceleration augmentation, as the core velocities are relatively low at this section of the SRM (u ~ 100 m/s). The peak burning rate values in Figure 5-7 can be correlated to the (negative) peak radial acceleration values in Figure 5-3. This gives an indication of the magnitude of acceleration augmentation of the burning rate.

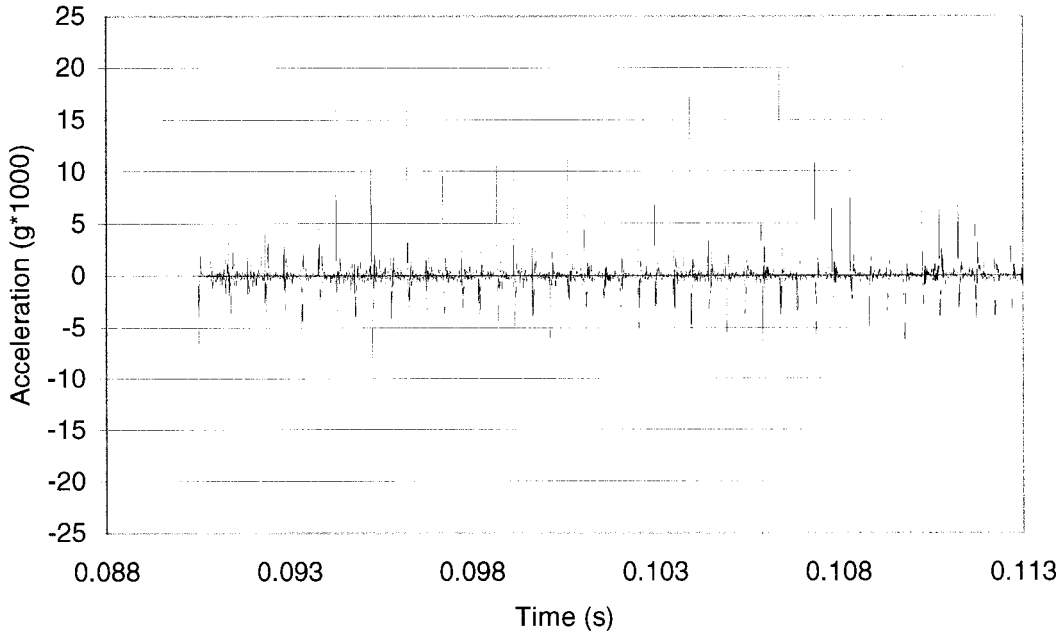


Figure 5-3 – Mid-point propellant normal acceleration (Q3DSRM)

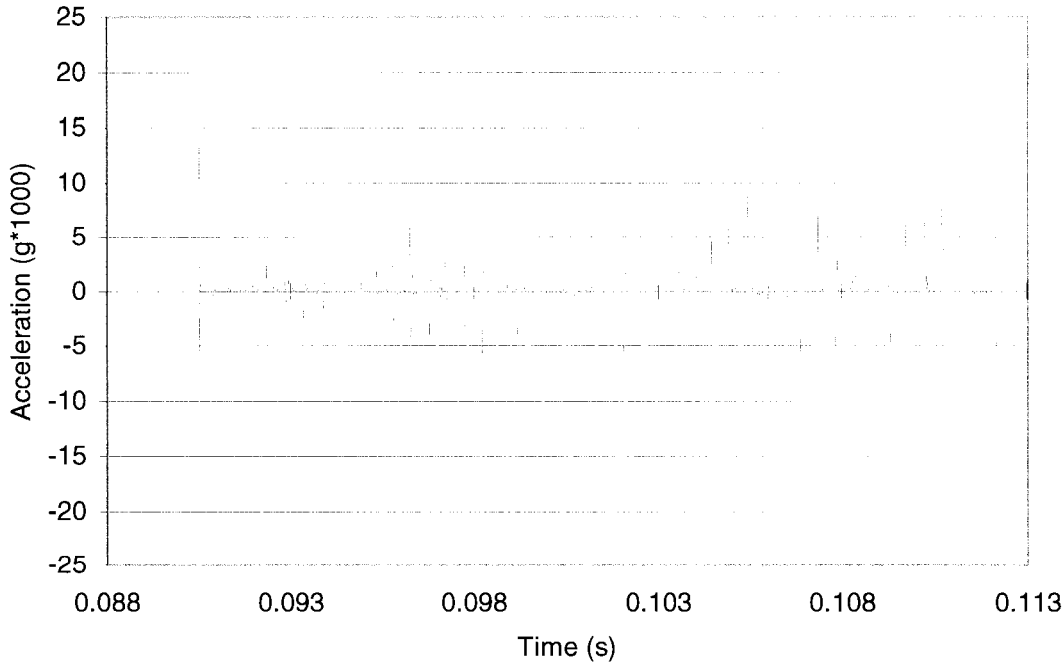


Figure 5-4 – Mid-point propellant normal acceleration (UTROC)

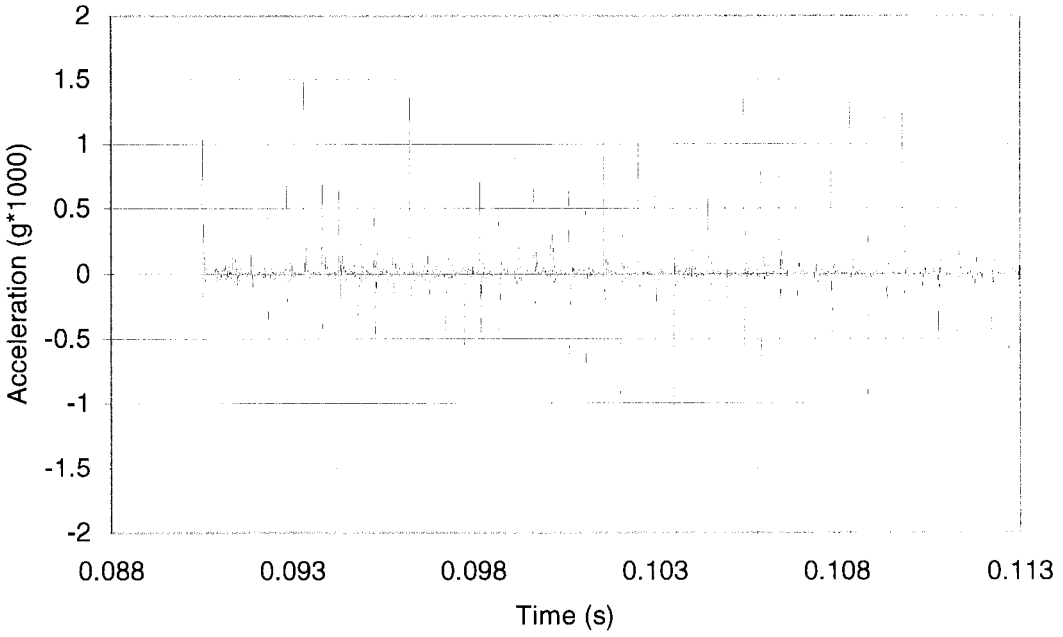


Figure 5-5 – Mid-point exterior sleeve acceleration (Q3DSRM)

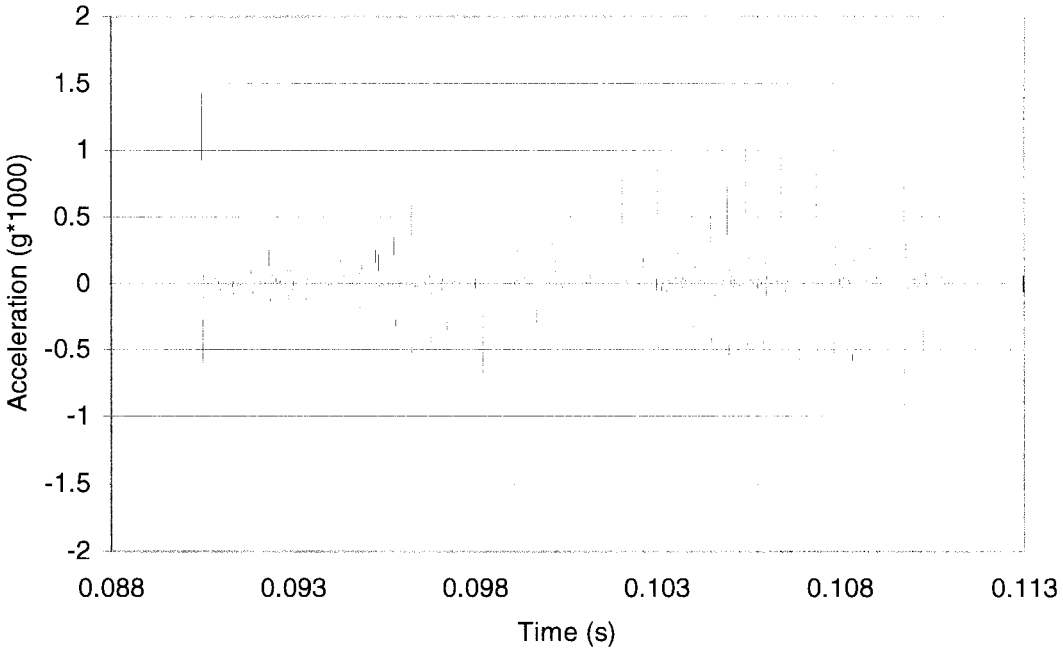


Figure 5-6 – Mid-point exterior sleeve acceleration (UTROC)

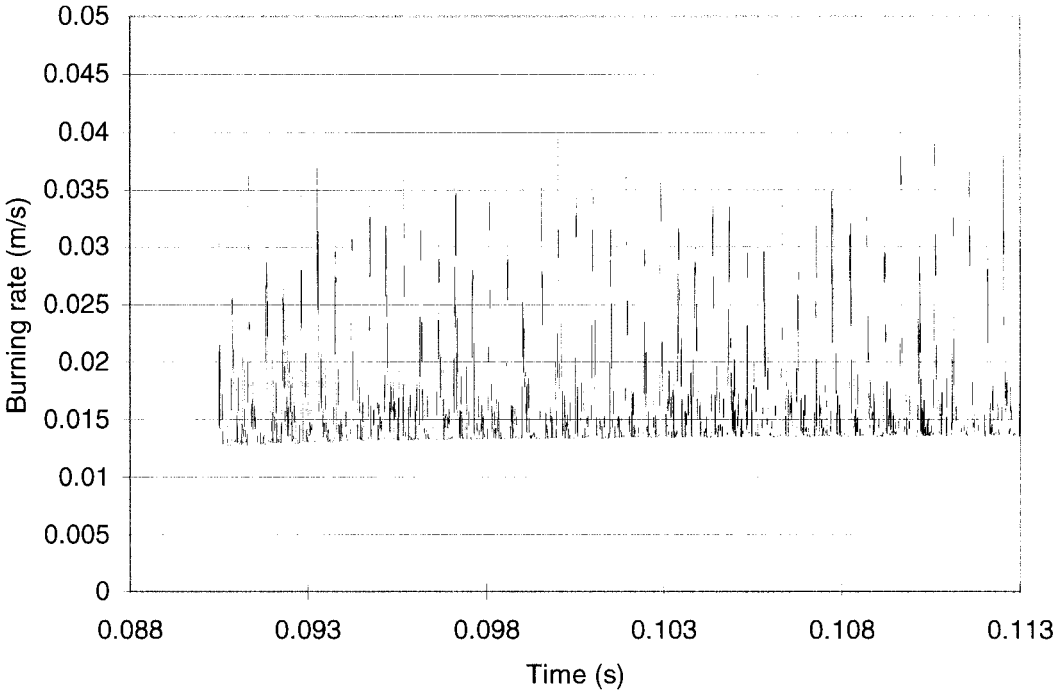


Figure 5-7 – Mid-point burning rate (Q3DSRM)

5.3 Remarks

In the preceding section, a comparison of output for a cylindrical-grain SRM was made comparing two simulation models (Q3DSRM and UTROC), each using different structural models. There is good overall correlation between the Q3DSRM and UTROC predictions. In some cases, there is even better correlation of Q3DSRM output with the experimental data found in Reference 2 as seen in Figure 5-1 and Figure 5-5. However, there is also a little more low-level noise in the Q3DSRM output. As mentioned, this is probably due to the nature of the structural model where in the case of the Q3DSRM simulation, there are many nodes in the solution matrix leading to possible grid noise or related artificial activity.

The disturbance routine in the Q3DSRM simulation was modified to attempt to reduce spurious minor wave activity as observed in the first several cycles in Figure 5-1 and Figure 5-2. The algorithm was updated to have an interpolated pressure (from the local chamber pressure to the local chamber pressure plus pulse strength) in the

expansion tail with characteristic velocity to match. A constant velocity wave center with characteristic pressure; and again an interpolated pressure (back to the local chamber pressure) with a matching characteristic velocity in the compression wave front (see Figure 5-8). The magnitude of the pulse is retained at 1.21 MPa above the local chamber pressure. The pulse shape was altered slightly to extend the expansion tail so the structure would be less affected by the sudden introduction of the pulse, which would cause structural vibration that was not initiated by the passing wave itself. The disturbance wave front could be made as either a gradual isentropic compression wave or a shock-fronted wave. The pulse shape used in Section 5.2 is also displayed as a reference.

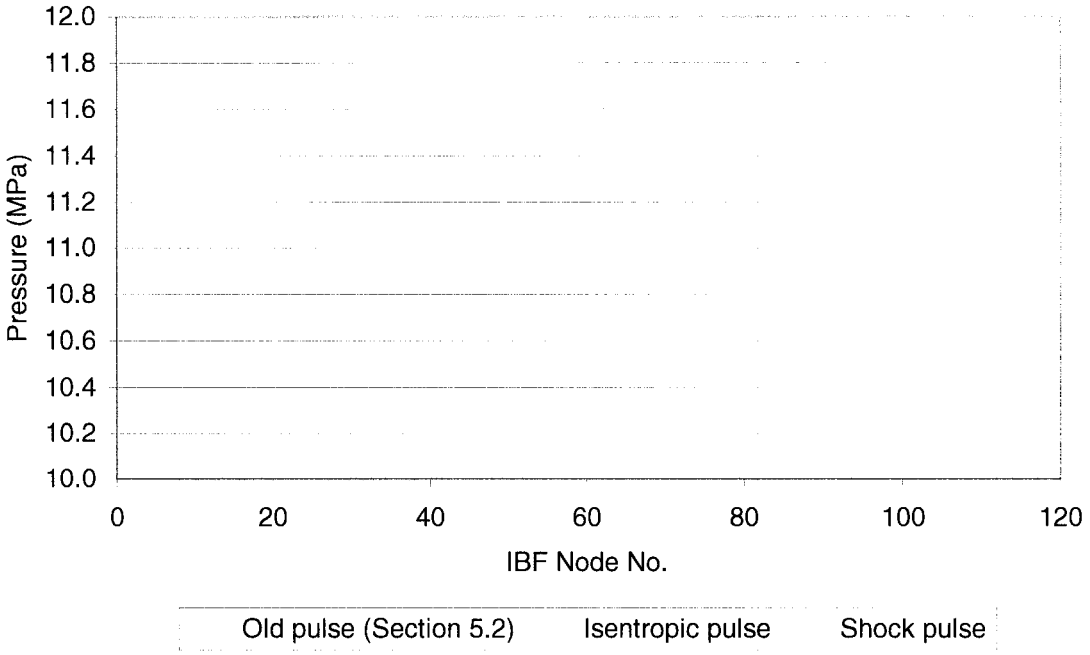


Figure 5-8 – Disturbance routine modification

Another two Q3DSRM simulations were run using both the new isentropic pulse and a shock-fronted pulse (see Figure 5-8), to see what affect the new disturbance routine would have on minor spurious wave activity. The head-end chamber pressure for the shock-fronted pulse test is displayed in Figure 5-9, while the head-end chamber

pressure for the isentropic pulse is displayed in Figure 5-10. It is immediately evident that the low amplitude waves seen in the first several cycles in Figure 5-1 and Figure 5-2 are no longer present when either of the new pulses is used. The dc shift is nearly the same as in Figure 5-1 as is the limited-cycle wave amplitude. There is also less low-level noise in between cycles. One notable difference observed in Figure 5-9 when using the shock-fronted pulse is that the wave amplitude in the first several cycles is larger by approximately 1 MPa. This is due to the stronger structural response from the passing of a shock-fronted pulse causing a greater increase in burning rate locally. A shock-fronted pulse is generally shaped like a developed, limited amplitude wave, and so it will behave like one from the onset of pulsing.

From the above discussion, it is plain that the Q3DSRM simulator does in fact produce results that are comparable to previous numerical and experimental work for a cylindrical-grain SRM by Greatrix and Harris.[2] A new disturbance routine was also introduced that reduced spurious wave activity due to the introduction of the pulse. The Q3DSRM simulation model may now be used to simulate the coupled internal ballistics and structural response, of a star-grain SRM.

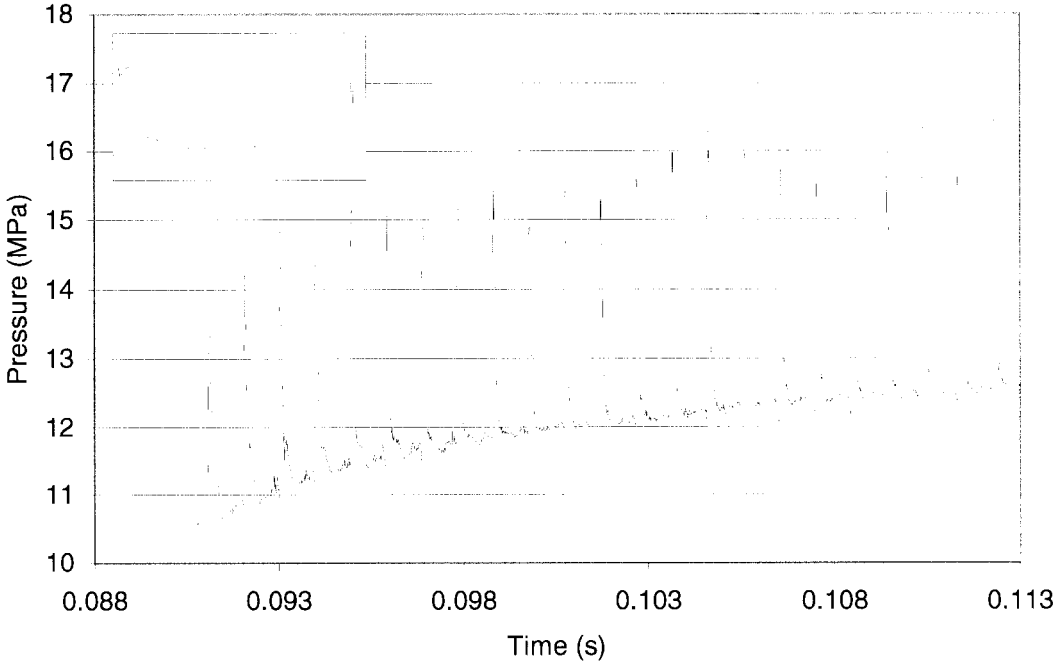


Figure 5-9 – Head-end chamber pressure using new shock-fronted pulse

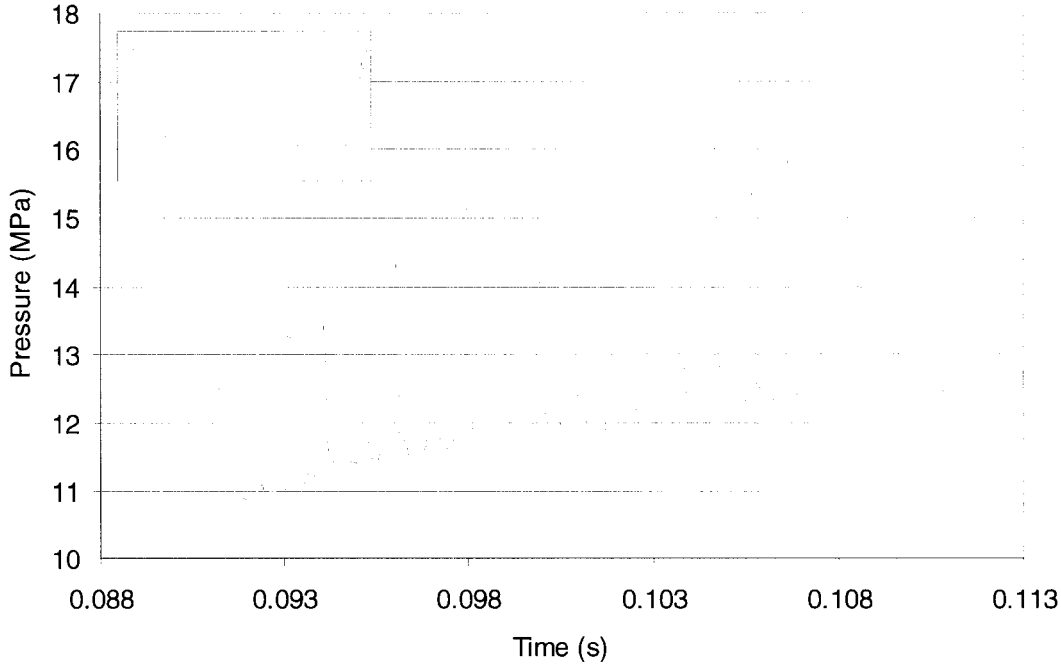


Figure 5-10 – Head-end chamber pressure using new isentropic pulse

6 Star-Grain SRM Analysis

6.1 Introduction

Having a numerical simulation model that is proven against a variety of cases, the remainder of this thesis can now be shifted to the primary focus of this thesis – star-grain SRMs. This chapter contains results and discussion on axial wave development inside star-grain SRMs, and details on how the structure couples to this wave development. The reader must be reminded that although many physical terms and properties of structural mechanics and internal ballistics of SRM's are discussed and referred to in the following chapter(s), they are in this thesis simulated numerically. These are numerical results, not experimental results. However, for the purposes of discussion, any simulated physical terms and properties may be thought of in regards to the actual physical terms and properties.

The motor is pulse-triggered into axial instability using a traveling pressure wave similar to the newer modified pulses seen in Chapter 5. The pulse is introduced at a time when approximately 28% propellant by mass has been consumed; this is typically around 154 – 166 ms into the simulated firing depending on the grid used. This time is not arbitrary; a similar pulsing time is used in the experimental work done by Harris, Wong and de Champlain.[3] In order to have some correlation with existing data, it was decided to pulse the motors in this simulation at a comparable time in the motor firing. As done with the cylindrical-grain SRMs, quasi-steady calculations are run initially to save on computer time, then unsteady calculations are run for 6 ms to settle out. A shorter settling time is used in the star-grain tests, as it was found to be unnecessary to have a longer settling time. The simulation is then left to run for another 25 ms after the pulse is triggered.

The grids used in the FE components of the simulation are $1/10^{th}$ pie sections of the complete star-grain grids seen in Figure 2-8 through Figure 2-11 (recall that grids that exploit symmetry have a suffix attached that denotes the fraction used. i.e. SG2-10

seen in Figure 2-14. From this point on, the grids will be labelled by the SG# prefix instead of their figure numbers). Using complete grids is only done to compare the FE response between whole grids and symmetric grids, to ensure that symmetric grids produce similar results. Typically, simulations are run using first lower density grids such as SG1-10, and then once basic trends are established, simulations that are more detailed are run using grids SG2-10 or SG3-10 to resolve these trends. Finally, grid SG4-10 could be used once specific program settings are finalized and/or a specific test requires more detail. Grid SG4-10 is also used as reference grid to establish grid independence and to verify the response of coarser grids. It was found that grid SG3-10 achieved grid independence; however, this grid was found to perform poorly when run with the burn-back routines and so will not be used in the rest of this section. As before in Chapter 5, there are 260 nodes in the IBF solution, 120 of which represent the propellant grain, so there will be 120 copies of the star-grain grid.

The fundamental resonant frequency for the FE damping model is found using the techniques discussed in Section 2.5 using grid SG4-10. However, as the propellant regresses, it will regress at various rates along the motor, the difference being mainly due to erosive burning; so, each section along the motor would realistically have a slightly different resonant frequency. In order to not have to evaluate the resonant frequency for every section, the resonant frequency found at the mid-point of the motor, when the motor's total propellant mass fraction is 28% (not to be confused with a section's 28% propellant mass fraction as seen in Figure 2-37), is applied to all the sections. The discrepancy between the true value and the applied value of the resonant frequency from one section to another is not great in all the cases, and will not cause any significant problems in the simulation. Further discussion on the damping model is found in Section 6.2.2.

Mentioned in Section 2.2, the star-grain motors that are simulated here are similar to those used in Reference 3; for the initial propellant dimensions refer to Figure 2-7 and for the casing and steel sleeve dimensions and all material properties, refer to Table 2-1. The steel sleeve in the motor setup is not used in flight-worthy SRMs; it is

used in static tests for safety reasons. In order to have the results in this thesis comparable to other experimental static SRM firings (at any time in the future), the steel sleeve is modelled. The SRM characteristics that will remain unchanged are listed in Table 6-1. The terms that aren't listed such as structural damping ratio, pulse strength, numerical dissipation constant and time of pulsing, are presented as variables for each simulation firing.

FE resonant frequency – 28% (f_n)	4199	Hz
Propellant grain length (L_p)	518	mm
Nozzle throat diameter (d_t)	23	mm
Grain/nozzle conv. length ratio (L_p/L_c)	16	-
Pressure-dependent burning rate (r_p)	$0.0007[p(kPa)]^{0.35}$	m/s
Propellant specific heat (C_s)	1500	J/(kg-K)
Propellant flame temperature (T_f)	3000	K
Propellant surface temperature (T_s)	1000	K
Initial propellant temperature (T_i)	294	K
Propellant surface roughness (ε)	10	μm
Gas specific heat (C_p)	1920	J/(kg-K)
Gas Prandtl number (Pr)	0.72	-
Specific gas constant (R)	320	J/(kg-K)
Gas thermal conductivity (k)	0.2	W/(m-K)
Gas absolute viscosity (μ)	8.07×10^{-5}	Pa-s
Gas specific heat ratio (γ)	1.2	-
Particle mass fraction (α_p)	0 %	-
Casing/Prop. long. damping ratio (ξ_L)	0.1	-
Net surface heat of reaction (ΔH_s)	0	J/kg

Table 6-1 – Star-grain SRM characteristics

The general outline for this chapter first presents simulation output using a low-density grid (SG1-10) in order to evaluate general trends of certain variables in the

simulation. Without having experimental data to compare to, this parametric study is a suitable method to try to understand the characteristics of the star-grain SRM's instability behaviour and structural coupling to that instability. Then, detailed simulated firings are run on key points of interest. These simulations will be accompanied by an analysis of the structural response to the passing of a shock wave. Together, a description of how the structure influences wave development can be presented.

6.2 Preliminary Results

This section is devoted to identifying basic trends in the star-grain simulations using a lower density grid, namely the effect of changes in the numerical dissipation constant, damping ratio and pulse strength. Because the damping properties of the visco-elastic propellant are not clearly known, simulated firings are made using a variety of damping ratios – all else being constant. In addition, structural tests are run in order to clarify the damping properties of the star-grain structure. In the cylindrical-grain SRM tests in the previous chapter, the pulsing strength was set at ~11% of the local base chamber pressure. This is somewhat higher than the experimental pulse strength used in References 2 and 3 (~5%). The higher pulse strengths were used in the numerical simulations in Reference 2 because there was trouble producing comparable fundamental-mode wave systems at lower pulse strengths. To see if similar difficulties would arise with the current simulation model, a variety of pulse strengths are tested to see if they produce the expected wave system – again, all else being constant. The numerical dissipation constant, though not a SRM characteristic, was found to affect axial wave development as it controls the level of noise in the FE solution, and affects the response of the structure through the damping model (damping out higher frequency modes). Ideally, one would hope for no dissipation to be required ($\gamma = 0.5$) as numerical dissipation tends to smear results over time; also, having $\gamma = 0.5$ assures second-order accuracy in time. However, as it will be shown, the employed grids typically do require some numerical dissipation, so a series of simulated firings will be run to observe these trends as well. Additionally, the result of a run using a higher

density grid is presented, to show the difference a higher quality grid makes on noise reduction.

6.2.1 Numerical Dissipation Constant and Grid Density

The effects of the numerical dissipation constant are discussed first. Even though it is not a true SRM characteristic, it does significantly affect the simulation results as will be presented shortly. As discussed in Section 2.4.4, the numerical dissipation constant γ must be ≥ 0.5. However, when early simulations were run at settings close to 0.5, there was too much high-frequency algorithmic noise in the finite element (FE) solution (which eventually passes to the internal ballistic flow (IBF) solution) to produce reasonable results. Therefore, a series of runs was done to evaluate the effect of the numerical dissipation constant. These first series of simulated firings will establish the trends for changes in the numerical dissipation constant, using a damping ratio (ζ) of 0.35 and a pulse strength of 11% of the base pressure (16 atm) using grid SG1-10.

Figure 6-1 through Figure 6-4 display the head-end chamber pressure for this series of simulated firings using numerical damping constant values of 0.55, 0.60, 0.74 and 0.98. The inset in these figures displays a detailed view of one arbitrary cycle. It is clear that setting $\gamma = 0.55$ as done in Figure 6-1, produced far too much structural noise to maintain a sustained shockwave. However, using higher values of γ does produce a stable sustained shock wave. The key difference between using the various values of γ in Figure 6-2 through Figure 6-4 is the dc shift noticeably becomes smaller as γ is increased; however, the limiting wave amplitude remains the same at ~4 MPa. This reduction in dc rise is a result of less artificial high-frequency algorithmic noise, which is somewhat visible in the inset plots in Figure 6-1 through Figure 6-4.

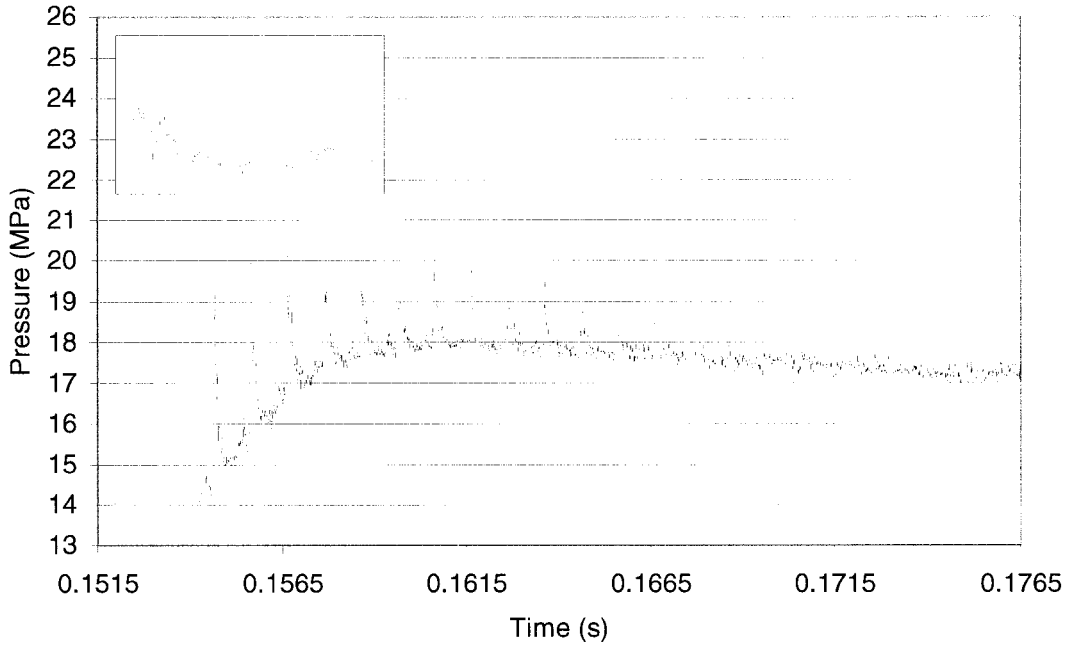


Figure 6-1 – Head-end chamber pressure, γ= 0.55, ζ = 0.35, 16 atm pulse, SG1-10

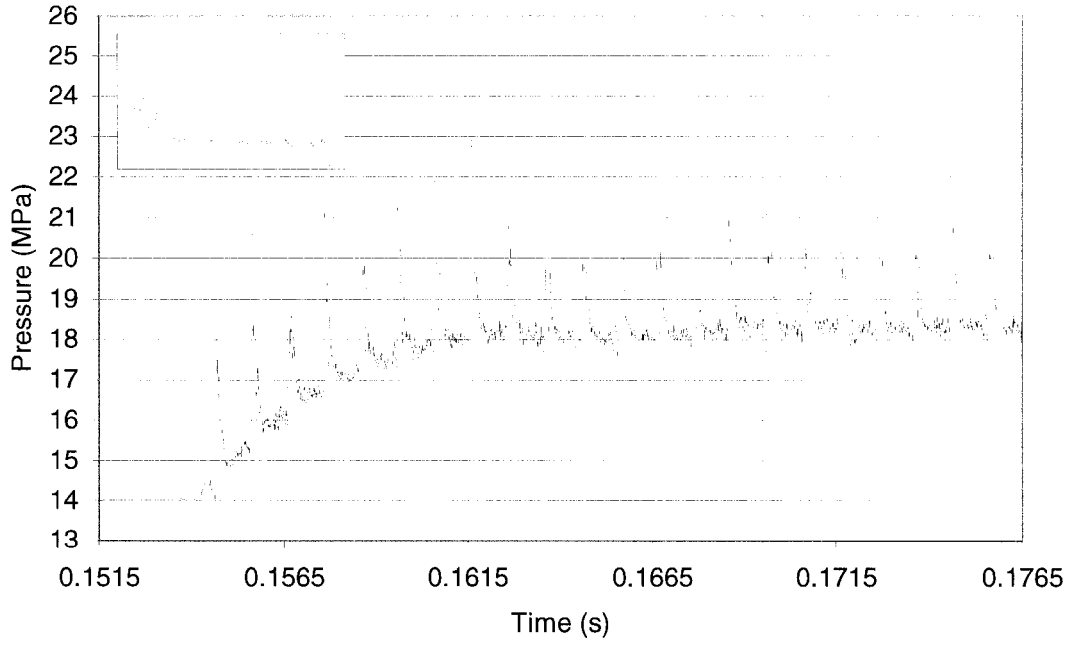


Figure 6-2 – Head-end chamber pressure, γ= 0.60, ζ = 0.35, 16 atm pulse, SG1-10

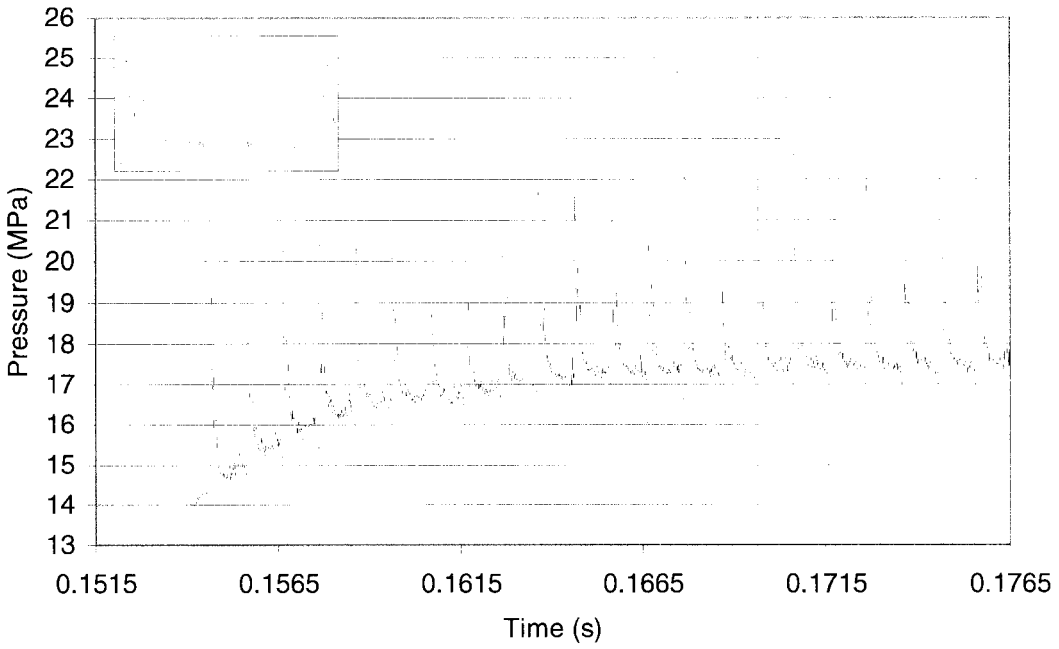


Figure 6-3 – Head-end chamber pressure, γ= 0.74, ζ = 0.35, 16 atm pulse, SG1-10

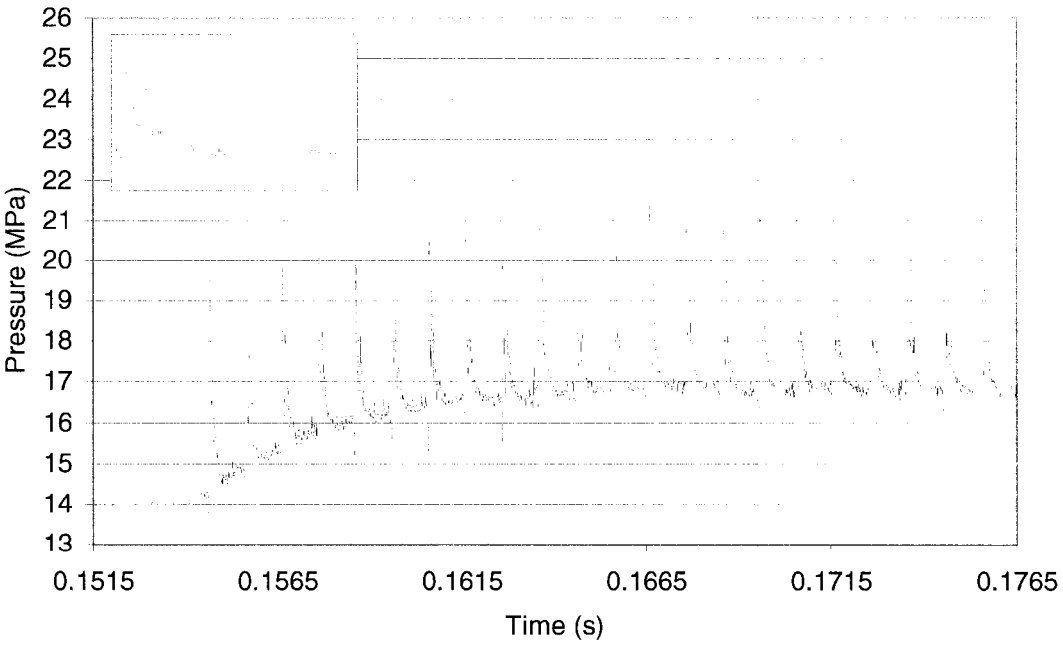


Figure 6-4 – Head-end chamber pressure, γ= 0.98, ζ = 0.35, 16 atm pulse, SG1-10

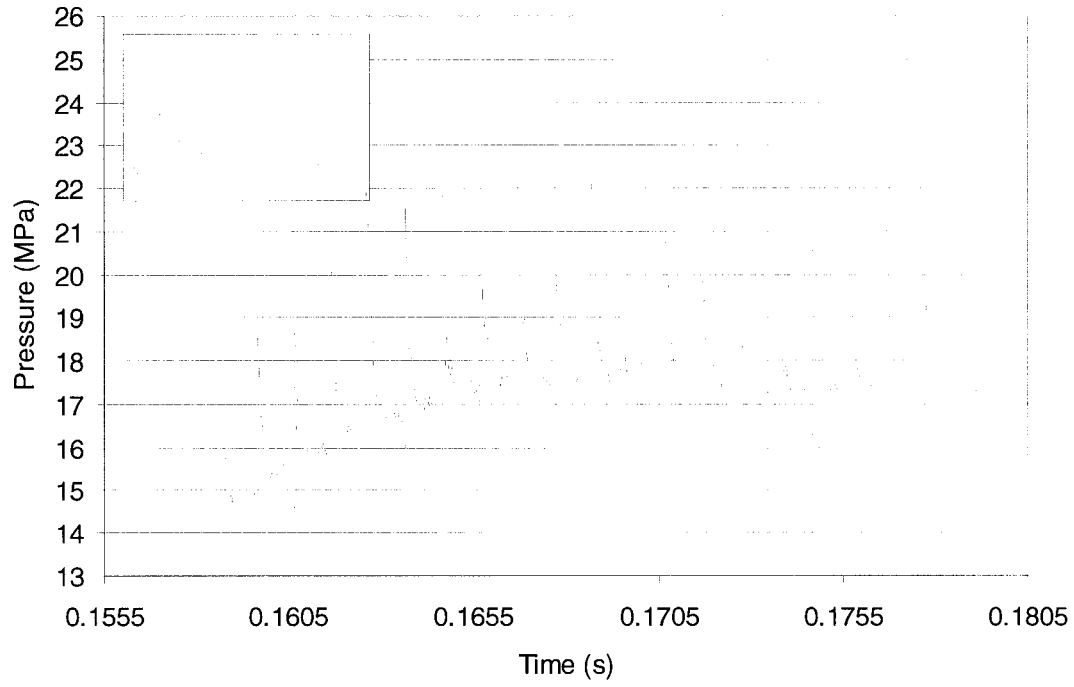


Figure 6-5 – Head-end chamber pressure, γ= 0.55, ζ = 0.35, 16 atm pulse, SG2-10

It is clear that the numerical dissipation constant can have a significant effect on the employed wave development mechanisms. Reducing the strength of the higher frequency vibrations (both artificial and physical) will reduce the strength in the local acceleration field and thus augment the burning rate to a lesser degree. However, as the numerical dissipation constant damps out higher frequency oscillations, care must be taken not to over-damp the system, such that the simulated physical response of the system is not compromised. The use of higher density grids will in general improve the quality of the results by reducing artificial noise in the solution, through providing more degrees of freedom in the solution. Figure 6-5 graphs the head-end chamber pressure for the case tested in Figure 6-1 using grid SG2-10, where γ= 0.55, ζ = 0.35 and a 16 atm pulse is used. The difference is clear, in that the simulation in Figure 6-1 did not produce stable sustained wave motion, where in Figure 6-5 there is sustained wave activity. Notice that the dc rise however is not comparable being ~1 MPa lower than the dc rise seen in the sustained wave activity in Figure 6-2 (which has a slightly higher value of γ). Of course, this is likely due to the lower amplitude, high frequency noise in

the solution, the effect of which is evident in the inset graph of Figure 6-5. In effect, a finer grid provides a form of damping in that it helps reduce high frequency noise in the solution.

So, in retrospect, the level of higher frequency oscillations, both artificial and physical, influence the wave development mechanisms used in this simulation. The numerical dissipation constant attempts to control the level of noise in the solution by providing additional damping into the system. Although this does damp the physical modes as well, careful application of the numerical dissipation constant will provide satisfactory results while maintaining solution accuracy. What is interesting, is that in the next section, it will be shown that in order to simulate a proper structural response, the numerical dissipation constant must be increased somewhat, such that the structure responds to what would be expected with a given damping ratio.

6.2.2 Structural Damping and the Damping Ratio

The viscous damping model used in this simulation depends primarily on the product of the fundamental resonant frequency of the structure, the damping ratio, and the mass matrix, as presented in Equation 2.62. The mass matrix is a property derived from the FE discretization process. The damping ratio is a parameter that controls/describes the damping characteristics of a system. As mentioned previously, a damping ratio of 0.1 should reduce the displacement amplitude in each cycle by about half. This parameter is not exactly known for propellants; it is usually estimated based on empirical observation and trial and error. The resonant frequency is found through harmonic testing as described in Section 2.5. Typically, the resonant frequency that produces the greatest amplitude increase per cycle is the fundamental resonant frequency.

However, in the case of a star-grain motor, the peak (thickest web) and trough (thinnest web) of the star grain will generally oscillate at different frequencies, as will

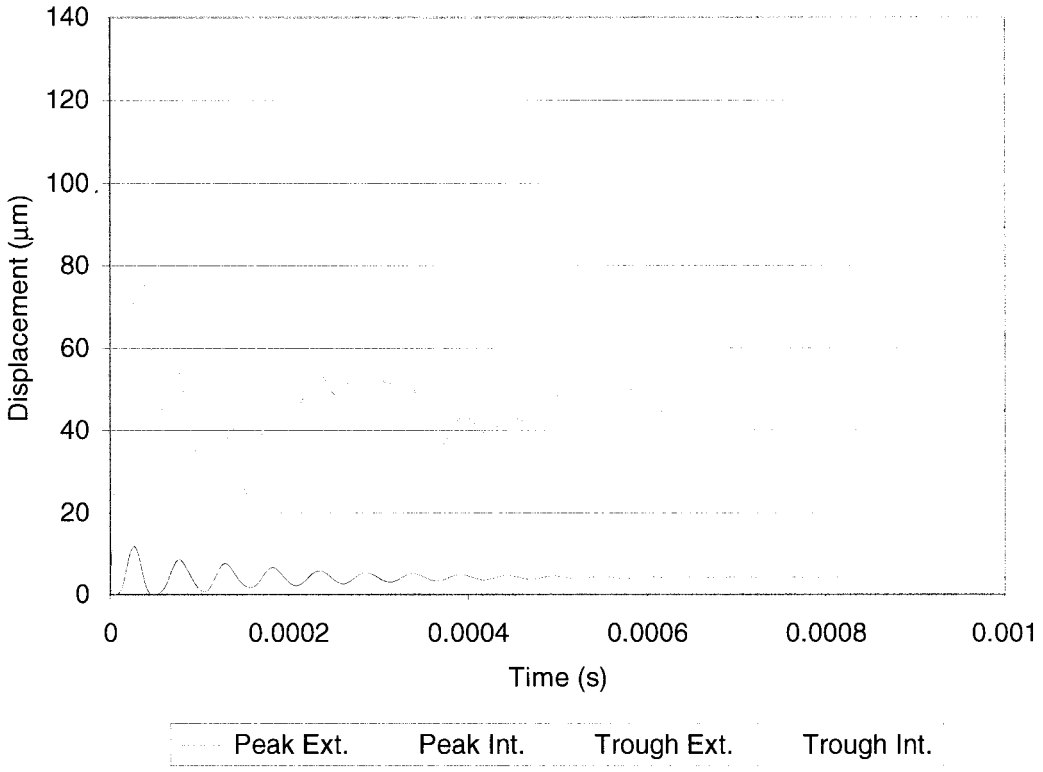


Figure 6-6 – Displacement-time plot of 10.5 MPa step impulse (γ= 0.6, ζ = 0.1)

the steel sleeve. Figure 6-6 displays a displacement-time graph for a 10.5 MPa step pressure increase using a damping ratio of 0.1 and a numerical dissipation constant setting of 0.6 at 28% propellant mass fraction using grid SG4-10. The displacements are measured radially from the center of the motor at both the interior propellant surface and exterior steel sleeve at the peak and trough of the star grain. Two principal resonant frequencies are evident. The lower frequency is the fundamental resonant frequency (~4.2 kHz) exhibited primarily by the interior propellant surface at the peak and to a lesser degree the trough. The higher frequency (~14 kHz) is clearly seen in the response of the exterior steel sleeve at both over the peak and trough, but is also seen superimposed onto the lower frequency at the interior propellant surface points. The damping ratio can be measured from Figure 6-6 using:[16]

$$\zeta = \frac{\delta}{\sqrt{(2\pi)^2 + \delta^2}} \tag{6.1}$$

where,

$$\delta = \frac{1}{j}\ln\left(\frac{u_1}{u_j}\right) \tag{6.2}$$

In Equations (6.1) and (6.2), ζ is the damping ratio, u_1 is the displacement amplitude (off the static equilibrium value) of the first cycle and u_j is the displacement amplitude (off the static equilibrium value) of the j^{th} cycle (greater measurement accuracy is assured by using a larger value of j). Using the above method, the damping ratios are extracted from the dynamic response of the system in Figure 6-6. The lower frequency, measured at the interior propellant surface at the peak of the star grain, has a damping ratio of ~0.991. While the higher frequency oscillations, measured at the exterior steel sleeve over the peak, exhibit a damping ratio of ~0.053 (Note that these damping ratio measurements are approximate as it is difficult to measure superimposed modes from a graph). The higher frequency vibrations must be more effectively damped such that the whole system would respond according to a given damping ratio.

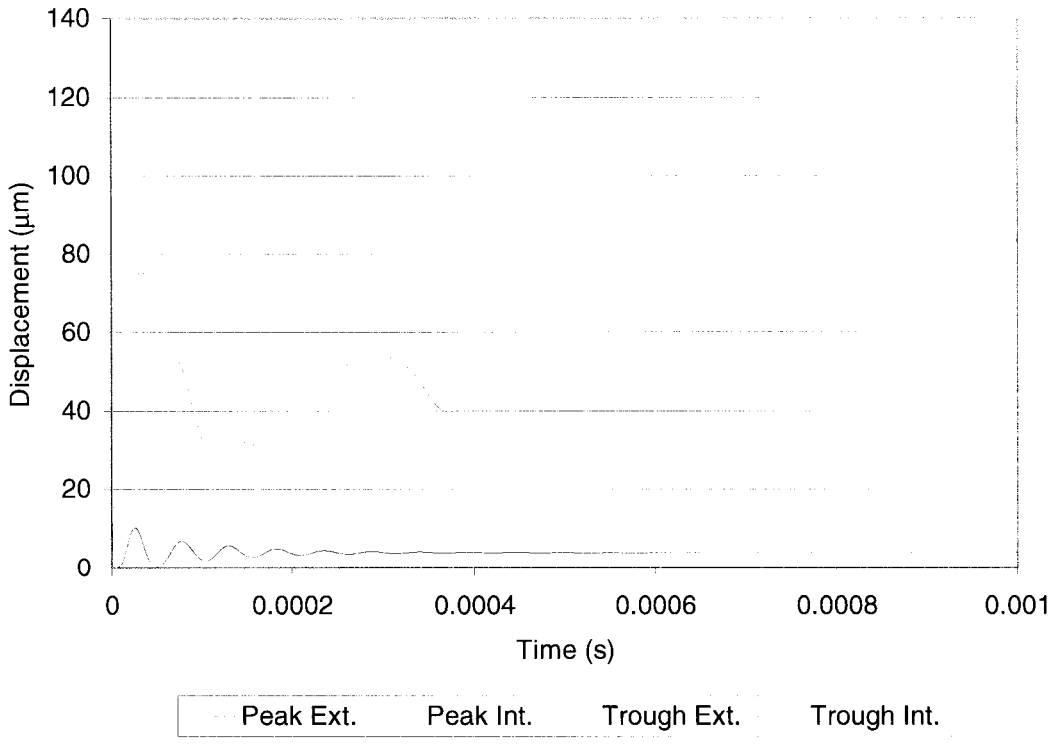


Figure 6-7 – Displacement-time plot of 10.5 MPa step impulse (γ= 0.98, ζ = 0.1)

An effective way of doing this is by increasing the numerical dissipation constant. Discussed in the previous section, the numerical dissipation constant damps higher frequency oscillations, both artificial and physical (the higher the frequency, the greater the damping – the effect is linear)[11]. Therefore, increasing this parameter will increase the effective damping ratio for the principal higher frequency mode. To maintain a consistent damping ratio throughout the system for both the high and low principal resonant frequencies, the numerical dissipation constant is set to 0.98. Again, a 10.5 MPa step pressure increase is applied to the system with the damping ratio set to 0.1, at a propellant mass fraction of 28%. The response is displayed in Figure 6-7. Here, the measured damping ratio for the lower principal frequency is ~0.104, and the higher principal frequency is ~0.099. These two values are close enough (considering the accuracy of the damping ratio measurement method used), to assume that the whole system responds, as one would expect having a specified damping ratio of 0.1. This test was repeated using the same value of γ and different values of damping ratio with similar results.

Having a numerical dissipation constant that helps provide an accurate response must be treated with care, since as discussed in the previous section, the numerical dissipation constant can significantly affect the wave development mechanisms in this simulation. Since a larger value of γ is used, there will be less response, and thus a weaker acceleration field, from the structural vibrations to promote wave growth. In order to assure that a sustained wave activity is developed in the SRM simulations, and since the damping ratio of the SRM is not clearly defined through experiment, a series of simulations are run that vary the damping ratio in order to define the related trends.

Five simulations were run using damping ratios of 0.15, 0.25, 0.35, 0.45 and 0.55, using a numerical dissipation constant setting of 0.98 and a pulse strength of 16 atm on grid SG1-10. The test run using a damping ratio of 0.15 produced poor results due to excessive noise in the solution, and so it will not be displayed here. The remaining four simulations are displayed in Figure 6-8 through Figure 6-10, with the

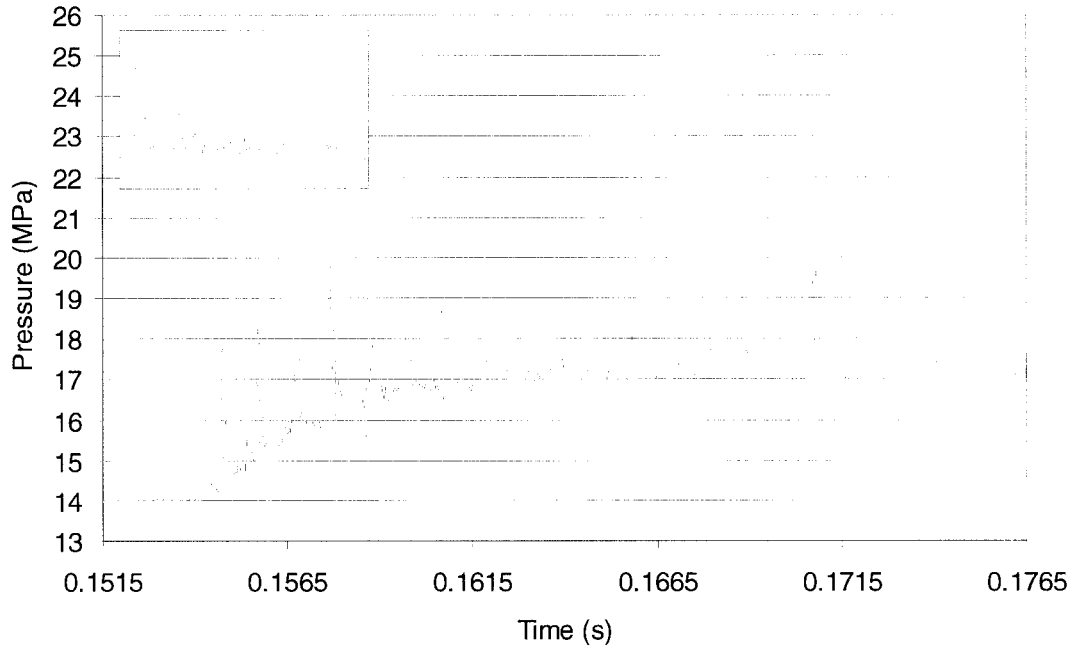


Figure 6-8 – Head-end chamber pressure, γ= 0.98, ζ = 0.25, 16 atm pulse, SG1-10

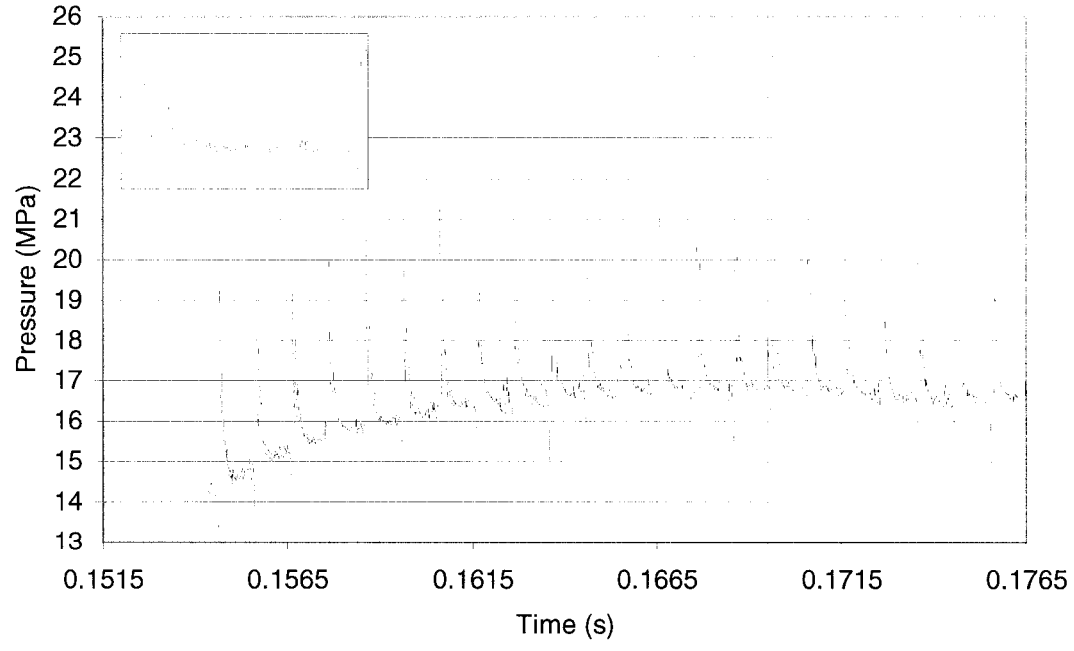


Figure 6-9 – Head-end chamber pressure, γ= 0.98, ζ = 0.45, 16 atm pulse, SG1-10

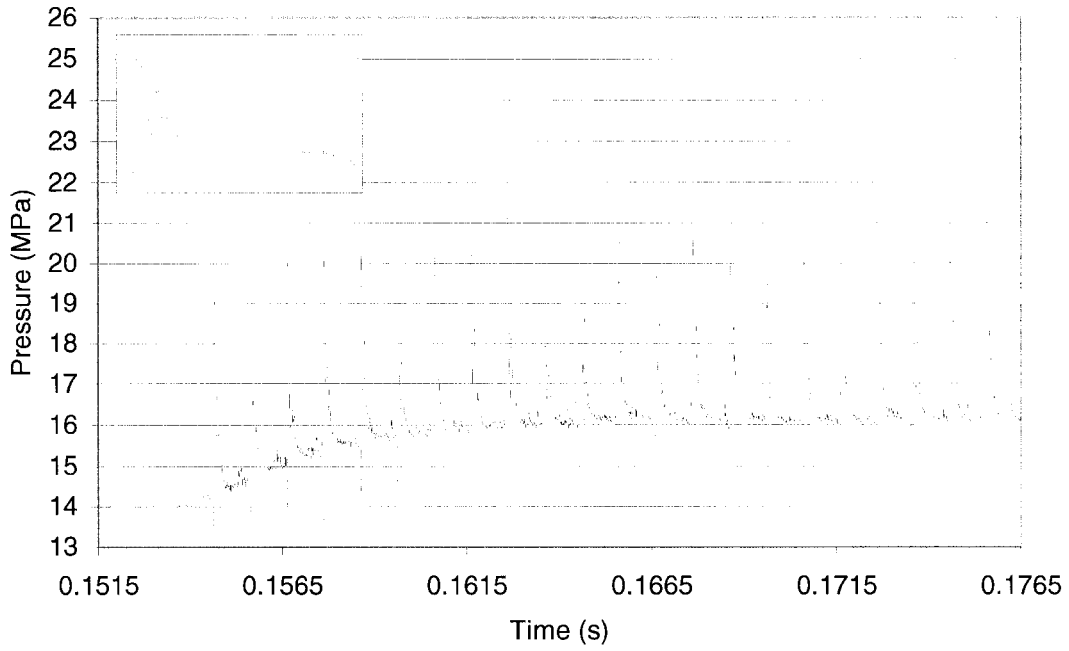


Figure 6-10 – Head-end chamber pressure, γ= 0.98, ζ = 0.55, 16 atm pulse, SG1-10

simulation that uses a damping ratio of 0.35 displayed in Figure 6-4 in Section 6.2.1. As one would expect, the greater the damping ratio, the lower the dc chamber pressure rise, as is observed in Figure 6-4 and Figure 6-8 through Figure 6-10. Looking at the inset plots in these figures, a secondary wave following the primary shock front is clearly seen. This is a result of the structural vibrations coupling with the burning rate mechanisms to promote wave development. As the damping ratio is increased, this secondary wave decreases in amplitude. This would be expected, as greater damping in the system would reduce post-shock vibration levels at a section, thus reducing the ability of the wave development mechanisms to strengthen the passing wave(s).

From this discussion, it is clear that the wave activity level in the SRM is dependent on the amount of damping in the system. Although it is not made clear on the exact amount of damping required in the system to model a real star-grain SRM structure, an estimate can be made by examining the head-end chamber pressures in this section and observing the resulting dc rise. The dc rise should be somewhat comparable to the shift seen in the time-averaged results for a star-grain SRM in

Reference 3 (≥ 3 MPa). This means that the damping ratio should be set for further detailed tests at less than or equal to 0.35, using the above results as an approximate benchmark. Furthermore, the numerical dissipation constant is set to 0.98 in order to have the FE system respond properly according to a given damping ratio.

6.2.3 Pulse Strength

The pulse strength was set to ~11% of the base chamber pressure (16 atm) for the majority of the simulations done up until now. The value of 11% was carried over from the cylindrical-grain SRM simulations run in Chapter 5. Originally, Greatrix and Harris had trouble numerically establishing a fundamental-mode wave system in a cylindrical-grain SRM using experimental pulse strengths (2% – 5% base chamber pressure), so they increased the pulse strength to 11% to finally get a suitable wave system developed.[2] In an attempt to see if similar problems with low pulse strengths occurred in this star-grain simulation model, a series of runs were done that gradually lowered the pulse strength down to ~2% of the base pressure.

These series of simulations all use a damping ratio of 0.35 and a numerical dissipation constant of 0.98 to keep noise reduction to a minimum in grid SG1-10. The runs begin with a higher pulse strength of ~13% base pressure (18 atm), to see if any changes in dc rise or limit wave amplitude result, and also to place an arbitrary upper limit on the pulse strength. Then the disturbance pressure is gradually reduced in steps of ~2% base pressure until a pulse strength of 5% base pressure is reached (7 atm); then the last run uses a pulse strength of ~2% base pressure (3 atm). Figure 6-11 through Figure 6-16, display the head-end chamber pressure for pulse strengths of 18, 14, 12, 10, 7 and 3 atm respectively. The simulation that uses a pulse strength of 16 atm, is displayed in Figure 6-4, in Section 6.2.1.

It is clear from Figure 6-11 through Figure 6-16, that sustained wave activity is possible in this simulation model employing star-grain geometry, which uses pulsing

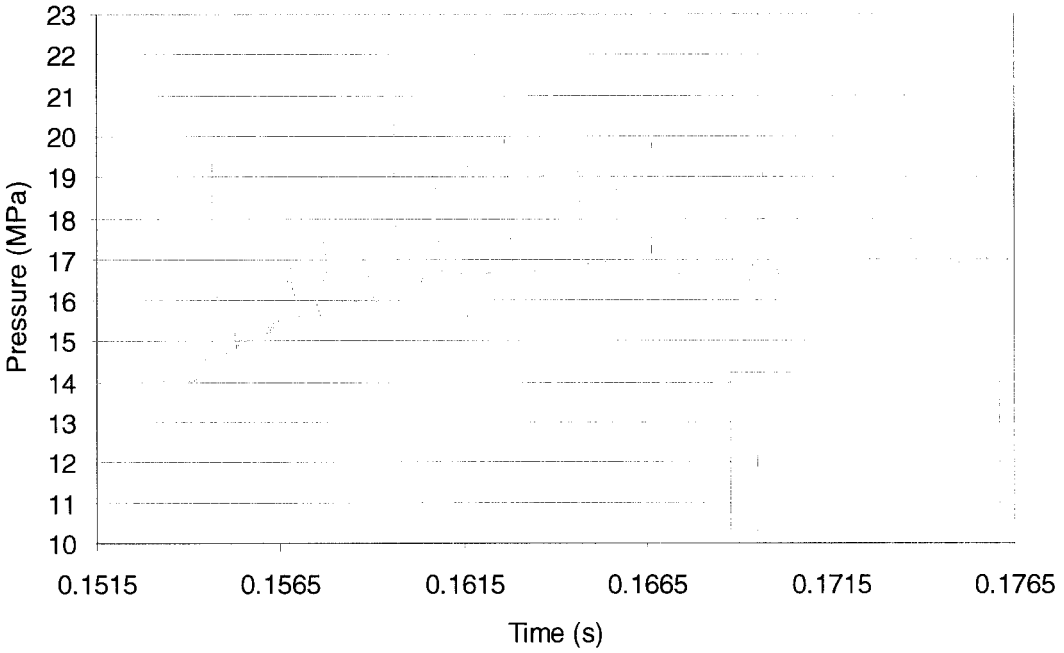


Figure 6-11 – Head-end press., $\gamma = 0.98$, $\zeta = 0.35$, 18 atm pulse (~13% base press.)

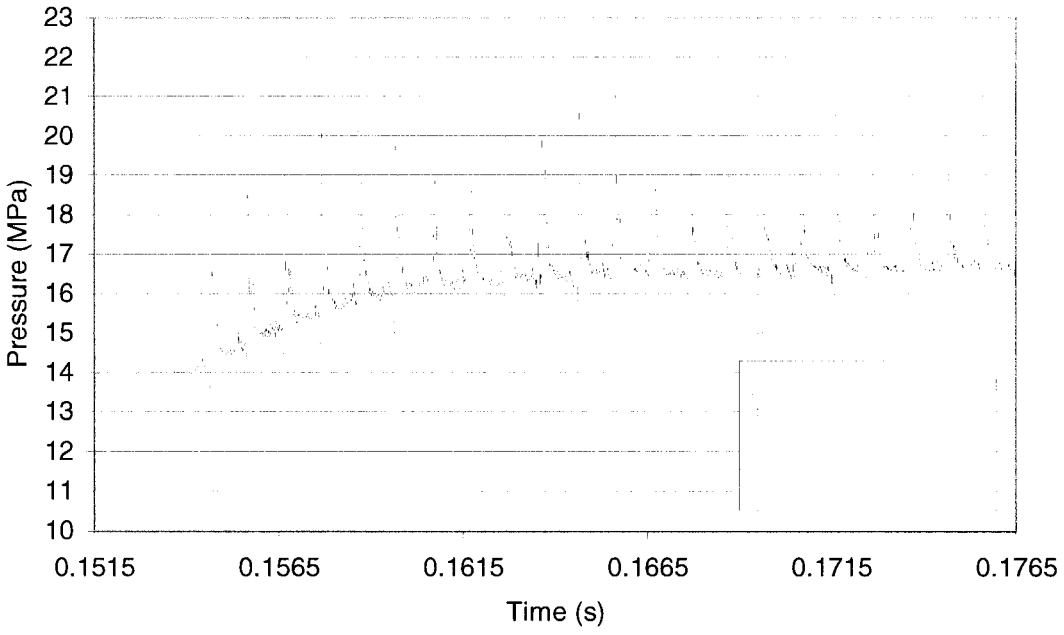


Figure 6-12 – Head-end press., $\gamma = 0.98$, $\zeta = 0.35$, 14 atm pulse (~11% base press.)

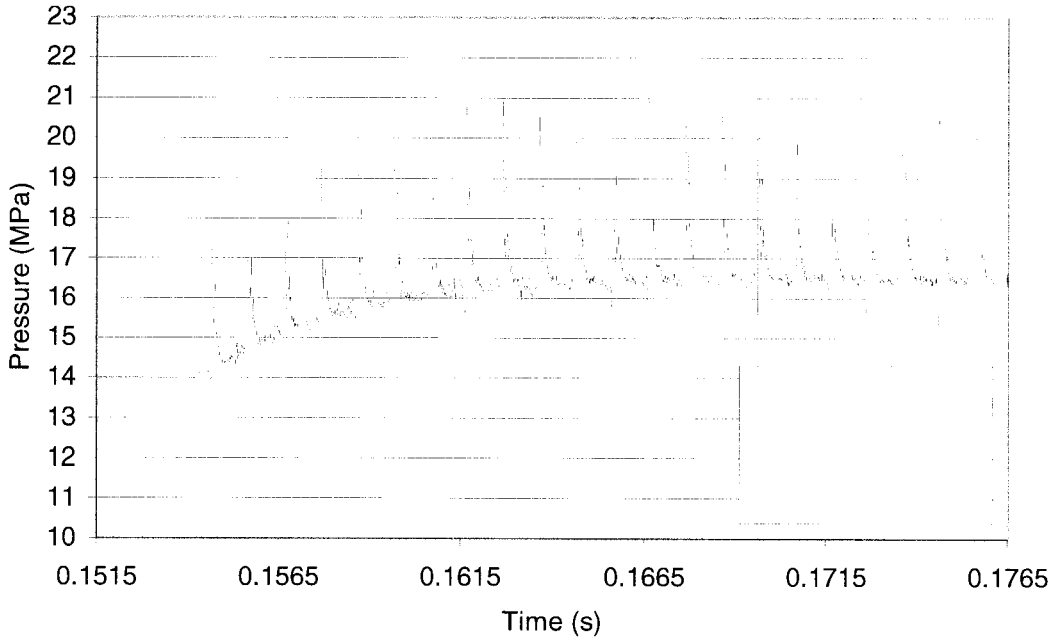


Figure 6-13 – Head-end press., γ= 0.98, ζ = 0.35, 12 atm pulse (~9% base press.)

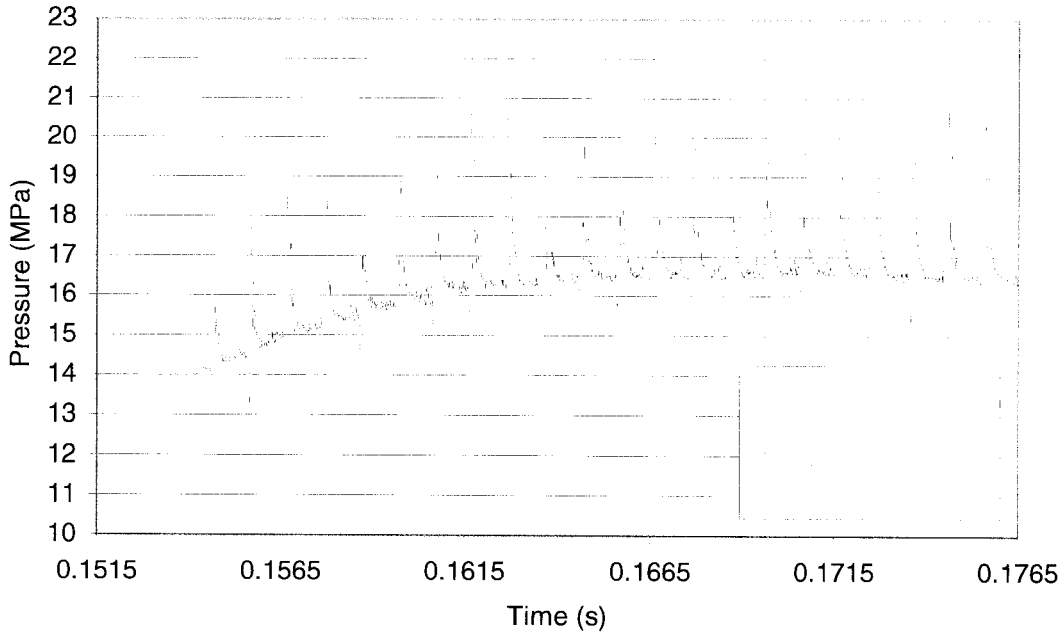


Figure 6-14 – Head-end press., γ= 0.98, ζ = 0.35, 10 atm pulse (~7% base press.)

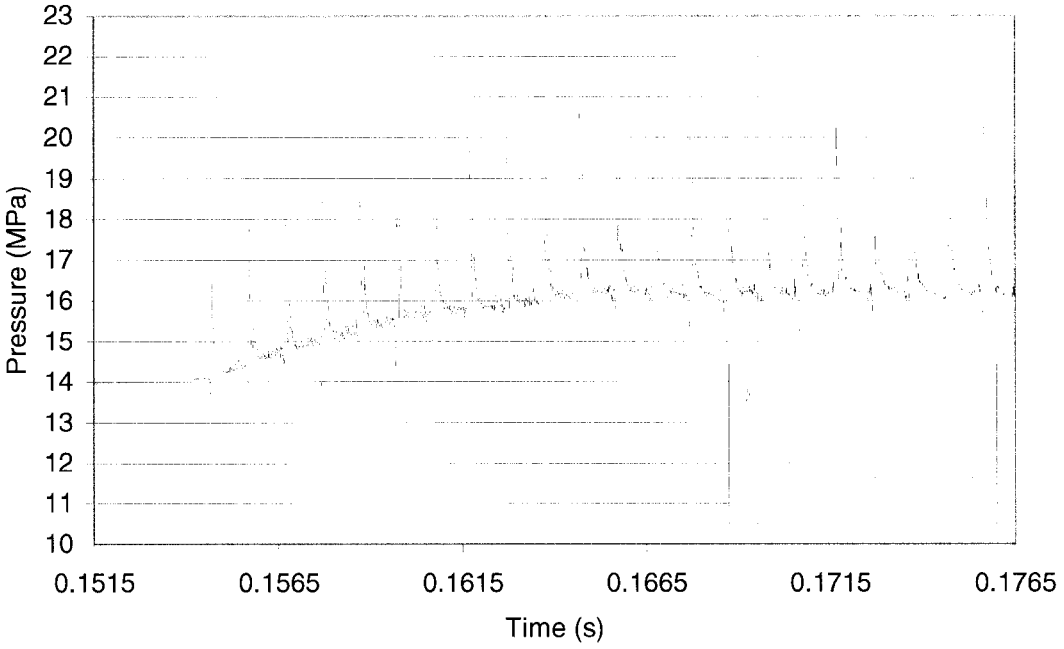


Figure 6-15 – Head-end press., γ= 0.98, ζ = 0.35, 7 atm pulse (~5% base press.)

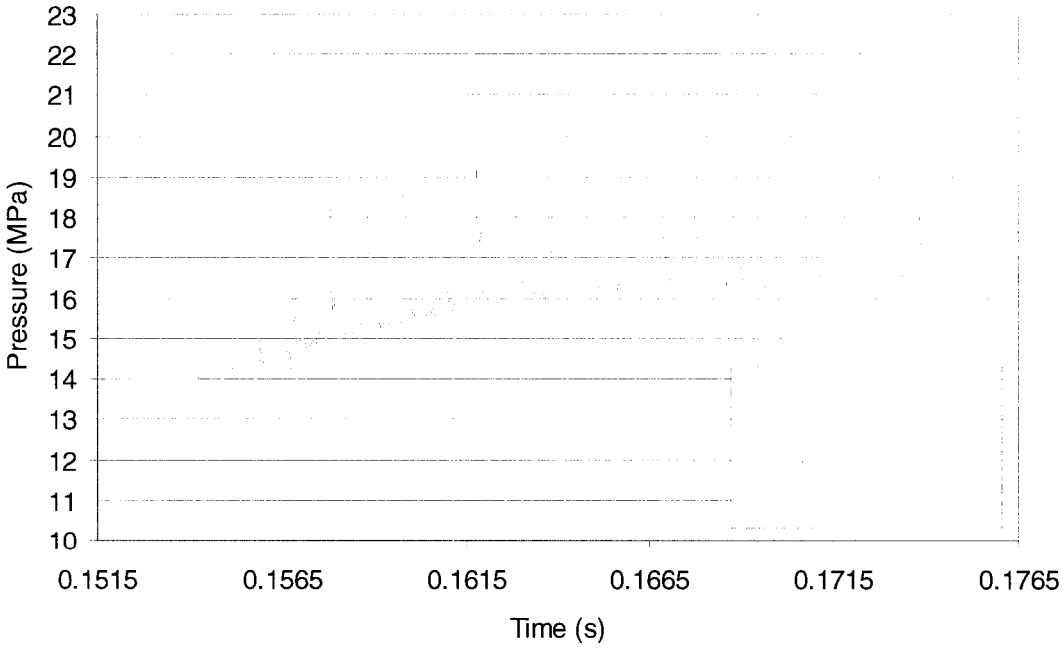


Figure 6-16 – Head-end press., γ= 0.98, ζ = 0.35, 3 atm pulse (~2% base press.)

strengths similar those used in experimental tests. The most evident trend observed in Figure 6-11 through Figure 6-16, is the more gradual dc rise seen using lower pulsing strengths, along with the more gradual increase in wave amplitude. The dc shift decreases only slightly (< 250 kPa) with decreasing pulse strength. This decrease is likely due to the more gradual increase in wave strength, which could create somewhat less noise in the solution. At the higher pulsing strengths, the limit-cycle wave amplitude is reached quickly, in some cases in the first cycle. This is physically valid, as the stronger the disturbance is, the greater the energy it possesses, and thus less wave cycles are required, which would normally strengthen the wave as it passes due to the burning rate mechanisms employed, to reach the limiting wave amplitude.

Therefore, it is possible to use lower pulse strengths in this simulation with star-grain SRMs. The problems dealt with by Greatrix and Harris for their numerical simulations of a cylindrical-grain SRM are not present in this simulation model. So, for the remainder of the simulations in this thesis, a pulse strength of 5% of base pressure (typically about 7 atm) is used to trigger the SRMs into instability, as this is a typical value used in experimental tests.

6.3 Detailed Results and Analysis

Simulations were run, in lieu of the discussion presented in the previous sub-sections, in order to get a dc pressure rise in a simulated firing of an SRM that best suited the time-averaged results presented in Reference 3 by only varying the damping ratio (the pulse strength is 7 atm and $\gamma = 0.98$). Doing so, one can assume that the damping properties are similar to those in the real SRM structure. The simulations were run using grids SG2-10 and SG4-10, and the damping ratio was changed from 0.1 to 0.35. The simulated firing that best compared to the time-averaged star-grain SRM experimental results in Reference 3 used a damping ratio of 0.25. The head-end chamber pressure is displayed in Figure 6-17.

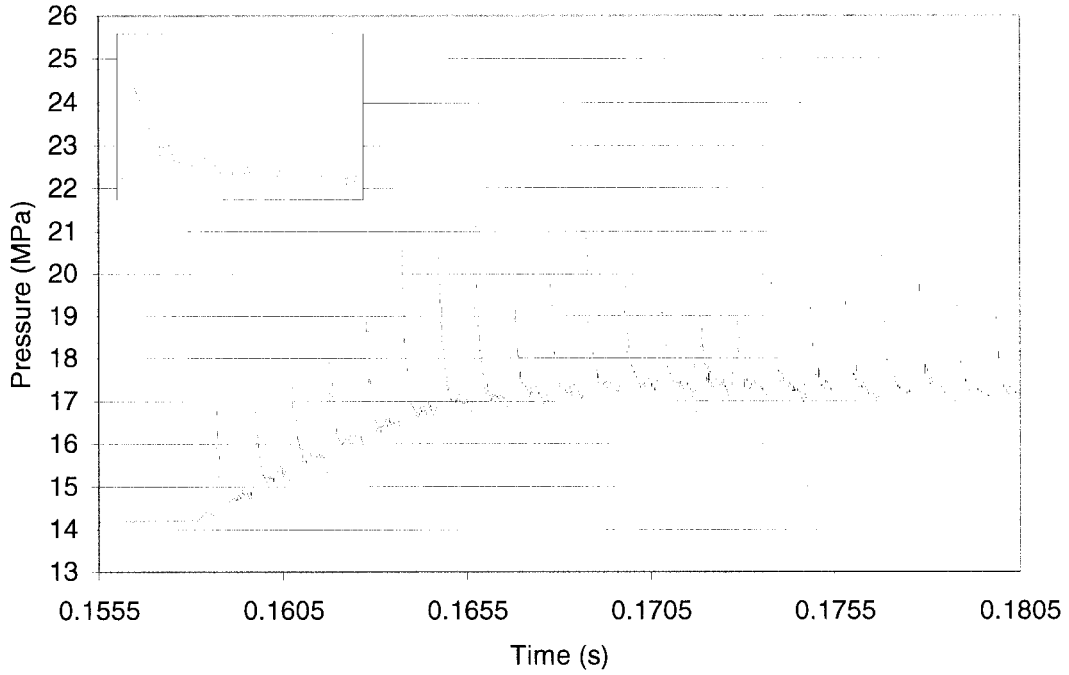


Figure 6-17 – Head-end chamber pressure, γ= 0.98, ζ = 0.25, SG2-10

The limiting wave amplitude reaches a value of approximately 4 MPa several cycles after the pulse is triggered, and the base pressure increases to a value of ~17.5 MPa, a dc rise of ~3.5 MPa. The star-grain propellant structure will generally have a greater number of significant vibration modes than cylindrical-grain motors. Of particular interest are the regions of the trough and the peak of the inner propellant surface. These two regions have differing natural frequencies as Harris, Wong and de Champlain also noted in their FE structural analysis, and so will affect the coupling of the structural vibrations to the burning rate.[3] Looking at the inset of Figure 6-17, one cycle of a wave is displayed. In comparison to the numerical cylindrical-grain SRM results seen in Chapter 5, there is more post-shock activity present in Figure 6-17. The higher frequency oscillations in the trough of the grain section would appear, through the acceleration-augmented burning rate mechanism, to especially reinforce the secondary waves after the initial shock front has passed, in addition to burning input from the peak section of the grain.

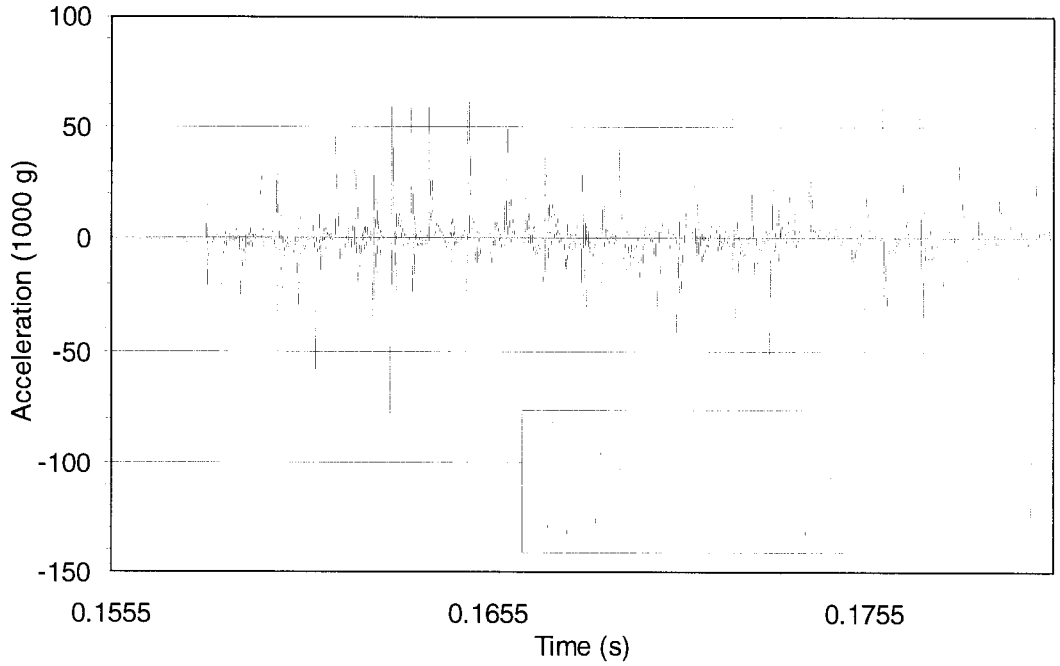


Figure 6-18 – Mid-point radial acceleration at the trough of the propellant surface

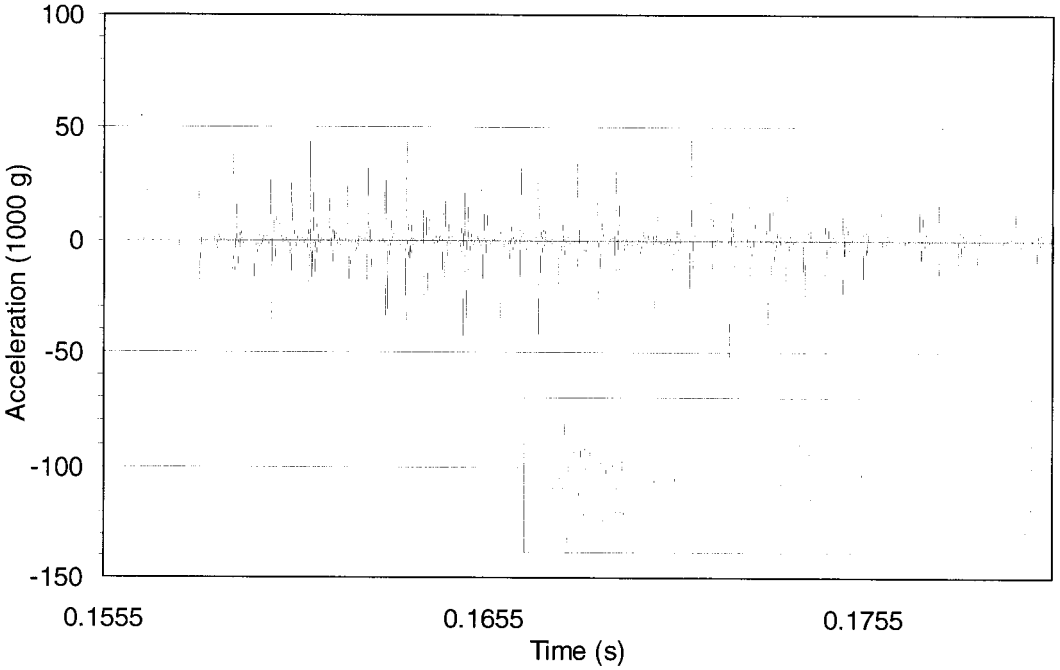


Figure 6-19 – Mid-point radial acceleration at the peak of the propellant surface

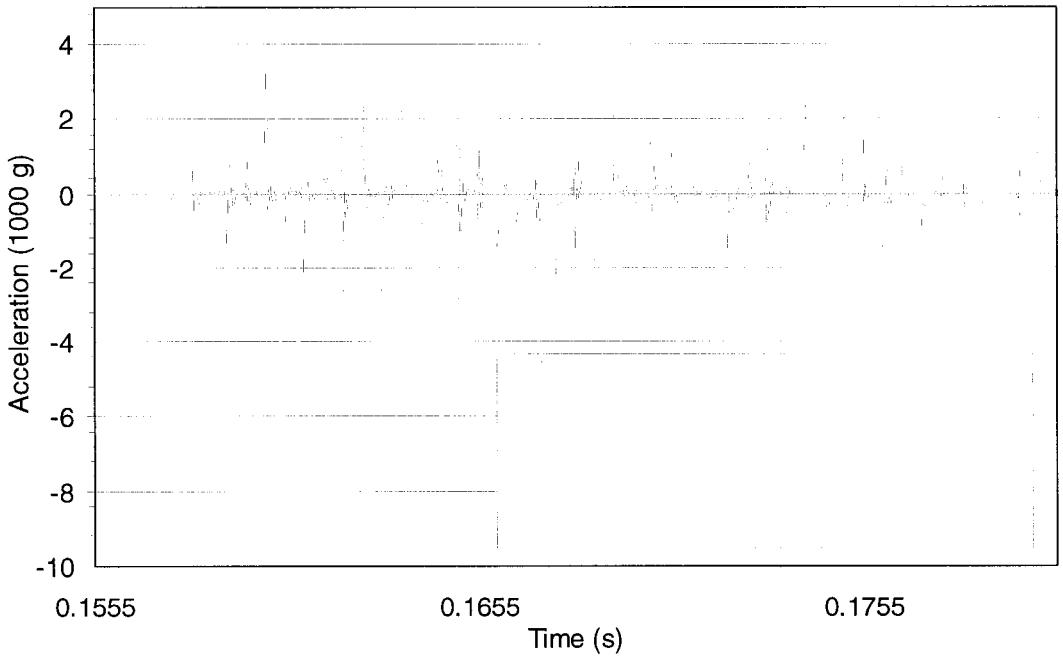


Figure 6-20 – Mid-point radial acceleration at the ext. steel sleeve over the trough

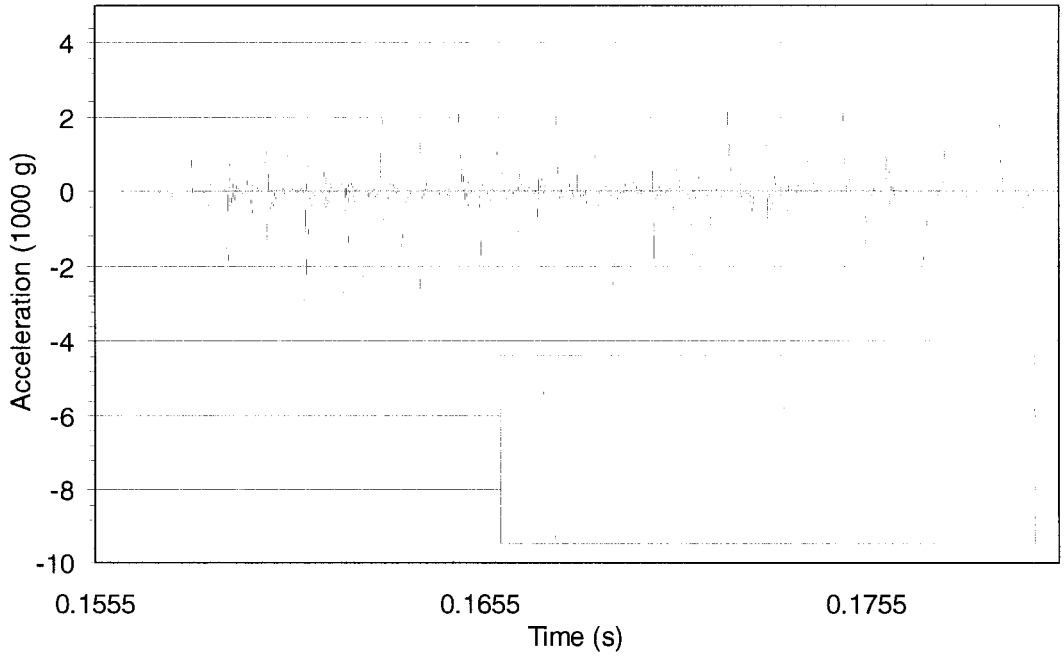


Figure 6-21 – Mid-point radial acceleration at the ext. steel sleeve over the peak

Figure 6-18 and Figure 6-19 display the radial acceleration-time graphs for the trough and peak respectively, of the inner propellant surface at the mid-point of the SRM. Here, it is evident that there is more activity in the trough of the section than the peak. The peak acceleration levels approach ±55000 g in Figure 6-18 while they reach ±45000 g in Figure 6-19. Also, looking at the insets in Figure 6-18 and Figure 6-19, it can be seen that there is more activity in the post-shock oscillations in the trough. This suggests that there is more burning rate augmentation in the troughs than at the peaks as a result. The mid-length sleeve external wall radial accelerations for positions directly over the trough and peak are displayed in Figure 6-20 and Figure 6-21 respectively. Again, variations in the peak acceleration levels are evident; Figure 6-20 having peak levels of ±3000 g and Figure 6-21 having peak levels of ±2500 g.

In order to examine the coupling of the structural vibrations with wave development, a displacement-time plot of one cycle at the mid-length is displayed in Figure 6-22 along with the corresponding mid-length chamber pressure in Figure 6-23. The higher frequency oscillations in the trough are evident in Figure 6-22. As discussed in Section 6.2.2, the principal frequency of the peak region is the fundamental resonant frequency of the motor section (~4 kHz), while the principal frequency of the trough region of the star grain is higher at ~14 kHz. This will correlate to larger local accelerations in the trough, thus leading to a higher burning rate augmentation via the mechanisms presented in Section 3.4.

The grain section peak has a lesser role in the burning rate with a lower vibration frequency and comparable deflection. The lower vibration frequency produces a longer period where the local accelerations augment the burning rate to a lesser degree than the trough, due to the lower mean amplitude. The net effect of the overall burning rate input is evident in the inset of Figure 6-17 and in Figure 6-23, where the post-shock pressure does not decay immediately to the base pressure level, as observed by Greatrix and Harris for the numerical results for a cylindrical-grain SRM.[2]

Figure 6-22 – Mid-point propellant surface displacements

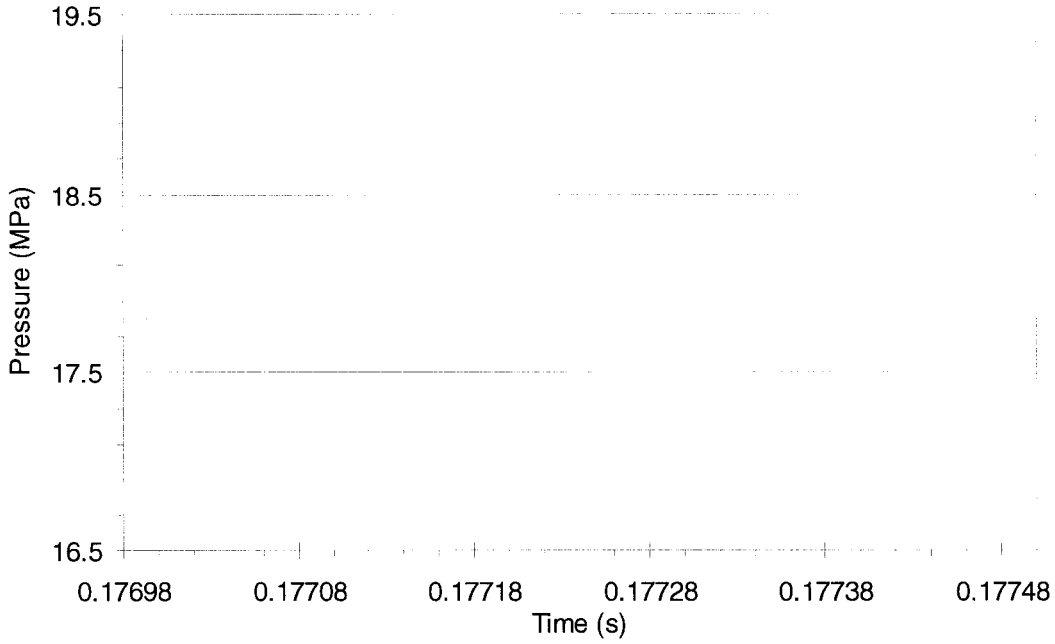


Figure 6-23 – Mid-point chamber pressure

7 Concluding Remarks

7.1 Conclusions

The predictive capability of this numerical simulation to model star-grain SRM nonsteady behaviour has been demonstrated. Although there was no high-resolution experimental data to compare to, the numerical output does correlate to the time-averaged results.[3] One of the key factors affecting wave development in star-grain motors is the vibration of the peak and trough of the star geometry. Each region oscillates at a different frequency; therefore, each region will affect pressure wave development to a level that depends on the dynamic response of that region. Since the trough generally oscillates at a higher frequency, the local acceleration field will be stronger, reinforcing the passing shock wave and generating secondary pressure waves behind the shock. The peak of the star geometry oscillates at a lower frequency and thus augments the burning rate to a lesser degree over a longer period. This helps to reduce the post-shock pressure decay in the pressure wave.

This numerical model predicts SRM combustion instability symptoms based on acceleration-augmented burning rate mechanisms. Axial vibration is not a strong factor in this study due to the comparatively small axial acceleration levels, given the heavyweight motor system. Other sources of acceleration fields lie in structural vibrations affected by transverse waves. Although not modelled in this simulation, transverse waves could play an important role in star-grain internal ballistic behaviour. Especially early on in the firing where transverse wave vibration frequencies may lie closer to the natural frequency of the trough region; so, even if the trough region has a higher damping, transverse wave activity may augment existing structural vibrations to negate the effect of greater damping, and further enhance wave development.

Basic trends in the simulation model are also established through a parametric study. The effect of the numerical dissipation constant and grid density were analyzed,

as were the damping properties of the structure and the triggering pulse strength. The numerical dissipation constant has an effect on the amplitude of high frequency oscillations both artificial and physical. Increasing it has the effect of damping these higher frequencies out. An increase in grid density helps reduce only artificial high frequency noise. This feature of the numerical dissipation constant is used to maintain a consistent damping ratio in the system.

Since various regions of the star grain oscillated at different frequencies, the effective damping ratio was different. To keep the damping model consistent with the physical damping ratio, the numerical dissipation constant is increased somewhat to help damp the higher frequency oscillations. Once a consistent damping model was assured, a series of simulated firings was run to observe the trends of damping ratio on the wave development mechanisms. Decreasing the damping ratio, as one would expect, increases the wave activity level in the SRM.

Finally, tests were run to observe the effect of the pulsing strength to see if axial instability could be initiated using pulse strengths comparable to those used in previous experiments. It was found that lower pulsing strengths similar to those seen experimentally in Reference 2 and 3, do initiate axial instability. Stronger pulse strengths allow the limit-cycle wave amplitude to be reached faster, as well as letting the dc pressure rise more rapidly.

7.2 Future Recommendations

In order to more accurately model a real SRM, a full three-dimensional structural and internal flow model should be utilized. This will allow the modelling of transverse and other waves present in the SRM, that would effect the wave development mechanisms. In addition, other mechanisms that could potentially enhance wave growth could also be effectively modeled in a three-dimensional numerical model. However, using the current simulation model, the immediate improvements would be towards addressing some of the simulation assumptions. For example using

elements that have transient mass and stiffness properties; or increasing the order of the shape function for the elements, to provide a better approximation of the acceleration field. There are always new and better solution methods that are developed which could improve the predictive capability of this simulation model. Aside from developing a new computer model, future work could focus on alternate grain designs, triggering the pulse at times other than at 28% burn-back, removing the steel sleeve, applying spin to the SRM and then pulse-triggering the motor, using multiple port configurations and even using different propellants. There are always many possibilities to investigate.

References

1. F. S. Blomshield and C. A. Beiter, "Nonlinear Stability Testing and Pulsing of Full Scale Tactical Motors", 27th AIAA/ASME/SAE/ASEE Joint Propulsion Conference, AIAA Paper No. 91-1953, June 22-26, 1991.

2. D. R. Greatrix and P. G. Harris, "Structural Vibration Considerations for Solid Rocket Internal Ballistics Modeling", 36th AIAA/ASME/SAE/ASEE Joint Propulsion Conference, AIAA Paper No. 2000-3804, July 17-19, 2000.

3. P. G. Harris, F. C. Wong and A. de Champlain, "The Influence of Structural Vibrations on Pulse-Triggered Nonlinear Instability in Solid Rocket Motors: An Experimental Study", 32th AIAA/ASME/SAE/ASEE Joint Propulsion Conference, AIAA Paper No. 96-3250, July 1-3, 1996.

4. M. Golafshani, M. Farshchi and H. Ghassemi, "Effects of Grain Geometry on Pulse-Triggered Combustion Instability in Rocket Motors", Journal of Propulsion and Power, Vol. 18, No. 1, Jan. – Feb. 2002, pp. 123-130.

5. D. R. Greatrix, "Parametric Analysis of Combined Acceleration Effects on Solid-Propellant Combustion", Canadian Aeronautics and Space Journal, Vol. 40, No. 2, June 1994, pp. 68-73.

6. J. J. Gottlieb and D. R. Greatrix, "Numerical Study of the Effects of Longitudinal Acceleration on Solid Rocket Motor Internal Ballistics", Journal of Fluids Engineering, Vol. 114, September 1992, pp. 404-410.

7. D. R. Greatrix, "Acceleration-Based Combustion Augmentation for Non-Cylindrical Grain Solid Rocket Motors", 31st AIAA/ASME/SAE/ASEE Joint Propulsion Conference, AIAA Paper No. 95-2876, July 10-12, 1995.

8. D. R. Greatrix, "Structural Vibration and Solid Rocket Combustion Instability", Canadian Aeronautics and Space Journal Vol. 44, No. 1, March 1998, pp. 9-24.

9. A. O. Galyukov and P. A. Voinovich, "Two-Dimensional Unstructured Triangular Grid Generator", Advanced Technology Center, St. Petersburg, Russia (unpublished), 1993.

10. R. D. Cook, W. C. Young, Advanced Mechanics of Materials, Macmillan, New York, 1985.

11. R. D. Cook, D. S. Malkus, M. E. Plesha and R. J. Witt, Concepts and Applications of Finite Element Analysis, 4th ed., John Wiley & Sons Inc., New York, 2002.

12. T. R. Chandrupatla and A. D. Belegundu, Introduction to Finite Elements in Engineering, 2nd ed., Prentice Hall, Upper Saddle River, NJ, 1997.

13. J. Stewart, Calculus, 3rd ed., Brooks/Cole Pub. Co., Pacific Grove, CA, 1995.

14. F. L. Stasa, Applied Finite Element Analysis for Engineers, Holt, Rinehart and Winston, New York, 1984.

15. H. Anton, Elementary Linear Algebra, 7th ed., John Wiley and Sons Inc., New York, 1994.

16. L. Meirovitch, Fundamentals of Vibration, McGraw-Hill, New York, 2001.

17. R. C. Hibbler, Engineering Mechanics: Statics and Dynamics, 7th ed., Prentice Hall, Englewood Cliffs, NJ, 1995.

18. R. L. Burden and J. D. Faires, Numerical Analysis, 6th ed., Brooks/Cole Pub. Co., Pacific Grove, CA, 1997.

19. W. H. Press, S. A. Teukolsky, W. T. Vetterling and B. P. Flannery, Numerical Recipes in C, 2nd ed., Cambridge Univ. Press, Cambridge, Ch. 11, 1997.

20. D. R. Greatrix and V. Kudriavtsev, "Modeling of Structural Vibration for Motor Chamber Internal Flow Studies", ASME Pressure Vessels and Piping Conference, Aug. 4-8, 2002.

21. D. R. Greatrix and J. J. Gottlieb, "Higher-Order Random-Choice Method for Internal Ballistic Flows", 7th Annual Conference of the CFD Society of Canada (*CFD99*), May 30-June 1, 1999.

22. J. Glimm, "Solutions in the Large for Nonlinear Hyperbolic Systems of Equations", Communications in Pure and Applied Mathematics Vol. 18, 1965, pp. 697-715.

23. M. Ben-Artzi and J. Falcovitz, "An Upwind Second Order Scheme for Compressible Duct Flows", SIAM Journal on Scientific & Statistical Computing, Vol. 7, No. 3, July 1986, pp. 774-768.

24. G. P. Sutton and O. Biblarz, Rocket Propulsion Elements, 7th ed., John Wiley and Sons Inc., New York, 2001.

25. D. R. Greatrix, "Parametric Analysis of Solid-Propellant Erosive Burning" CSME Forum, June 27-29, 1994, pp. 425-435.

26. D. R. Greatrix and J. J. Gottlieb, "Erosive Burning Model for Composite-Propellant Rocket Motors with Large Length-to-Diameter Ratios", Canadian Aeronautics and Space Journal Vol. 33, No. 3, Sept. 1987, pp. 133-142.

27. D. R. Greatrix and J. J. Gottlieb, "Normal Acceleration Model for Composite-Propellant Combustion", CSME Transactions Vol. 12, No. 4, 1988, pp. 205-211.

28. C. B. Laney, Computational Gasdynamics, Cambridge Univ. Press, Cambridge, 1998.

Appendix

The simulation consists of two primary modules: one for the finite element (FE) solution and the other for the internal ballistic flow (IBF) solution. The FE module is written in a mixture of C and some C++, while the IBF module is written in FORTRAN. Both modules are far too long to be presented in their entirety in this thesis, so a couple of pertinent routines are extracted and presented here. From the FE module, only the main source file is printed. This file includes a listing of global variables, the arrangement of the FE and graphics initialization routines, as well as the main program loop of the simulation. Details of the program loop are flow-charted in Figure 4-5. From the IBF module, the pyrolysis rate routine is printed, the details of which are flow-charted in Figure 3-5.

Main Source File (see Figure 4-5)

```
////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
//
//                                                  Ryerson University
//
//                                  QUASI-3D SOLID ROCKET MOTOR SIMULATOR
//
//                                    by: Sonny Loncaric          Dec. 2000 – Dec. 2002
//
//                                                  Propulsion Research Facility
//
////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////

#include <glut.h>
#include <stdio.h>
#include <math.h>
#include "Graphics.h"
#include "IO.h"
#include "Properties Struct.h"
#include "Node Struct.h"
#include "Node Functions.h"
#include "Element Struct.h"
#include "Element Functions.h"
#include "BCs and Init Functions.h"
#include "FE Functions.h"
#include "Templates.h"
#include "Matrix Functions.h"
#include "Burn Line.h"
#include "Prop Burn.h"
#include "Newmark.h"
#include "Section.h"
```

```
//Constant/Setting Variables
int order;                                          //Order of FE solution (Structural)
int Torder;                                         //Order of Transient Thermal solution (in time)
float Tatm;                                         //Outside air temp. [K]
double h;                                           //Exterior Convection Coefficient [W/K]
double Patm;                                        //Outside air press. [Pa] - gauge
GLfloat w2scale;                                    //Time between divisions in window #2 [ms]
GLfloat w2sscale;                                   //Time between subdivisions in window #2 [ms]
GLfloat vscale;                                     //Scaling factor for vertical size of Window #2
GLdouble scale;                                     //Deformation Scaling factor (for display only)
int DataOutNum;                                     //Data Output every ## iterations (make even #)
int RunGraphs;                                      //Graphics Output every ## iterations (make even #)
int pa, pb;                                         //nodes to display info on in window #2 & vibout.dat
int pc, pd;                                         //nodes to display info on in vibout.dat (set to 0 if unused)
float w3min, w3max;                                 //UTRoc Output window range [* 10 MPa]
double s1max, s1min, s2max, s2min;                  //Max & Min values for stress display
double dispmax, velmax, accelmax, Tempmax;          //Max values for Q, V, A, T display
float scla, sclb;                                   //Scaling factors for Window #2 displacements
double Penalty;                                     //Stiffness Multiplier for Penalty Constraint
float CFLs, CFLf, CFLt;                             //Courant # for CFL cond. - s Structural, f IBF, t Thermal
double ND;                                          //Numerical Dissipation const. - must be >= 0.5
float PD;                                           //Lag Parameter for press. BC - must be >= 1.0
float Asmth;                                        //Smoothing const. for Accel. BC - 0 <= Asmth <= 1
double te;                                          //te - section thickness - (ROLEN in utinp1.dat / Ns)
int TotNn;          //Total # of nodes to display from UTRoc in Window 3 - (can't be > 'KB' in utinp1.dat)
double w, Pin, Tin; //w - angular Velocity [RPM], Pin - chamber press. [Pa], Tin - chamber temp. [K]
int symmdiv;        //Into how many pie sections to cut the grid (to exploit symmetry) -  MUST BE >= 1
double arbcutoff;   //Acceleration cutoff value [g] for burn rate calcs. (should be -ve)
double dtQS;        //Time step to use during QS calcs (and not doing any FE calcs) [s]
float Tpb;                                          //Time to begin/end burn back calcs. [s]
float Tstr;                                         //Time to include/begin structural FE calcs. [s]
float Pfreq;                                        //Harmonic Testing Frequency
float PAmp;                                         //Harmonic Testing Pressure Amplitude [Pa]
char WaveType;                                      //Harmonic Testing Wave Type - 1 = sine, 2 = N wave

//Switches (use either - true/false or 1/0)
//Run-time Switches(used/changed in program)
bool pause = true, mat = false, grid = true, stressp1 = false, stressp2 = false;
bool disp = false, vel = false, accel = true, vr = false, vt = false, temp = false;
//Compile-time Switches (need to set at compile time)
bool Window2, Window3;
        //Window2 = disp./vel./accel. vs. time for FE section OR Pc at head end.  Window3 = UTRoc Output
bool shwpts, FullScr, shwCB;                        //Show display point (a,b,c,d), Full Screen, Show CB
bool Plainstrain, Propburn;                         //If Plainstrain = false, then Plainstress = true
bool IBFok, Thermal, Structural, Hartst;            //Modules to run
bool AccAve;                                        //Use Spatial Acceleration Averaging
bool CrvApprox;     //Type of CB Approximation - true = Circular, false = Weighted Average
bool NonLin;                                        //Use large deformation analysis

double bb = 0.02;                                   //For debugging burn-back calcs. [m/s]

//Input/Output Files
char Settings[25] = {"settings.ini"};               //Constants and Settings Input file
char GridData[25] = {"data.ini"};                   //Grid Input File
char OscillationOutput[25] = {"vibout.dat"};        //FE Output File
char Tout[25] = {"Tout.dat"};                       //Thermal FE Output File
```

```
char RBOutput[25] = {"RBout.dat"};                  //Burn Rate/Accel. Output File
char PressOutput[25] = {"Pout.dat"};                //Chamber Pressure History File
char CBOutput[25] = {"CBout.dat"};                  //CB Position Output File

//Global Variables
int scrX, scrY;                                                    //x & y resolution of screen
int Ne, Nn, N2n, Ns, ntime = 0;                                    //# of Elements, # of Nodes, # of sections
int Matnum = 0;                                                    //# of materials
int sectnum = 0;                                                   //Section position (0 = head end)
float MaxX, MinX, MaxY, MinY;                                      //Graphics Parameters
float XX, YY, DispVal = 0;                                         //Mouse Parameters
double t = 0, tp = 0, ts[4], dt = 0, Udt = 0, Ut = 0, gtime, CN;   //Time Variables [s]
double OrgPropMass = 0;                                            //Original Propellant Mass
double MassRatio;                                                  //Mass Ratio
int Tchk = 0;                                                      //Time Sync. Variable
int f = 0;                                                         //Harmonic Testing Frequency [Hz]
bool graph = false, Datastrt = true, Cyclegraphics = true;         //More Switches
bool loadchk = false, CBModified = false, TSchk = true, FEPropUpdt = true;
char Win1title[25]  = {" - Accel. = "}, Win1titleA[9] = {" g"}, W1type = 6;      //Window title vars.

//Data Variables
Sectiontype *Xsect;
Nodetype *Nodes;
Node2type *Nodes2;
BLinetype *BLSides, *OutrSides, *RCSides;
ElementType *Elements;
Proptype MatProp[4];

//Solution Variables
int isect, idt = 1, IBFrun = 1, IBFcntr = 0, odt = 1, gdt = 1;     //Counters
double qe[12], re[12];                                             //Internal Force Assembly Matrices
double *M, *Fext, *Fint, *ktemp1, *At1;                            //Solution Matrices
double *Qt, *Vt, *At;                                  //Reference Section Matrices at t = ntime * dt
double *T;                                                         //Solution matrix for thermal FE

//Transfer arrays (to UTRoc)
#define GN 500  //Also set in UTRoc in SUBROUTINE IBF def'n... make sure it's the same in both codes!
int NCB[GN];
double Press[GN], Aport[GN], CBl[GN], Temp[GN], Vel[GN];
double CBrbi[GN][GN], an[GN][GN], at[GN][GN], L[GN][GN];

//Prototypes
void idle(void);
void display1(void);
void display2(void);
void display3(void);
void keyboard1(unsigned char, int, int);
void keyboard2(unsigned char, int, int);
void keyboard3(unsigned char, int, int);
void mouse1(int, int);
void SetWindowTitles();
void ExitFunction(int, Sectiontype []);
inline void PrintData();
extern "C" { void __stdcall IBF (int ntime, float CFLf, double *Udt, double *Ut, double CBrbi[GN][500], double
             Press[], double Temp[], double Vel[], double Aport[], double CBl[], int NCB[], double an[GN][500],
             double at[GN][500], double L[GN][500], int *Tchk); }
```

```
//Shortcut Definitions
#define NList Xsect[isect].sNodes
#define NList2 &Xsect[isect].sNodes
#define N2List Xsect[isect].sNodes2
#define EList Xsect[isect].sElements
#define EList2 &Xsect[isect].sElements
#define SList Xsect[isect].sCB
#define OList Xsect[isect].sOut
#define RList Xsect[isect].sRC
#define TEle Xsect[isect].ste

//MAIN
////////////////////////////////////////////////////////////////////////////////////////////////////////////
void idle(void)
{
//Event Decisions
//      if ((MassRatio <= 0.72) && (ntime > 0)) pause = true;                   //Pause at specified mass fraction
        if ((t >= Tstr) && (Structural == false)) Structural = true;            //Begin Structural Calculations
        if (t >= Tpb) Propburn = ((Propburn) ? false : true);                   //Begin Propellant Regression
        if (t >= Tstr) FEPropUpdt = false;                                      //Freeze FE properties.
        if ((t >= Tstr) && (t < Tstr + 0.000005)) { graph = false; gtime = t; } //Clear Window #2
        if ((t >= Tstr+0.00) && (Hartst)) f = Pfreq;                    //Time to begin pulses for resonant

        if (pause == false)
        {
// IBF (UTRoc)
                if (idt == IBFrun)                                              //Recalc time step every 'idt' iterations
                {
                        SectionProp(Ns, Xsect, Matnum, symmdiv, t, ts, Asmth, AccAve);
                        if (IBFok)
                        {
                                CBoutput(Ns, Xsect, Aport, CBl, NCB, an, at, L, w, symmdiv, arbcutoff);
//Skip TimeSyncCheck when transferring from Q.S. to U.S. calcs
                                if ((TSchk == false) && (idt == 1)) TSchk = true;
                                if (TSchk) if (TimeSyncCheck(&t, Ut, loadchk) == false) ExitFunction(Ns,
                                        Xsect);
                                IBF(IBFcntr, CFLf, &Udt, &Ut, CBrbi, Press, Temp, Vel, Aport, CBl, NCB,
                                        an, at, L, &Tchk);
                                if ((Tchk == 3) || (Tchk == 0)) IBFcntr++;
                                BCInput(Ns, Press, CBrbi, Xsect, &loadchk);
                        }
                        dt = FETimeStep(CN, Ns, Xsect, Structural, Thermal, dtQS);
                        IBFrun = TimeScaling(&dt, Udt, 1);//Last # - Calc dt this often if Udt = 0... 1 - 100
                        idt = (Tchk == 1 ? IBFrun : 1);
// Data Output
                        if ((odt == DataOutNum))// || (Tchk == 0))
                        {
                                DataOut(Datastrt, Ns, pa, pb, pc, pd, Ns/2-1, t, Xsect, OscillationOutput,
                                        RBOutput, PressOutput, CBOutput, symmdiv);
                                if (Thermal) DataOutT(Datastrt, t, Xsect[Ns/2-1].sNodes, Tout);
                                Datastrt = false;
                                odt = 1;
                        }
                        else odt++;
                }
                else idt++;
```

```
// FE Structural / Thermal
                if (Tchk == 1) { Tchk = 2; TSchk = false; }
                else
                {
                        if (Structural || Thermal)
                        {
                                if (IBFok) PressureInterp(Ns, idt, IBFrun, Tchk, Xsect, PD);
                                for (isect = 0; isect < Ns; isect++)
                                {
                                        if (Hartst) Xsect[isect].sPin = HarmonicPress(t, &tp, f, PAmp, Pin,
                                                WaveType);
                                        ReConnectSteppingmatricies(Qt, Vt, At, T, NList, N2List, order);
                                        if (Thermal) SolveTempraturevector(Xsect[isect].sNn, Torder, T,
                                                At1, ktemp1, EList, OList, MatProp, qe, re, dt, h, Tatm,
                                                TEle);
                                        if (Structural)
                                        {
                                                FillIntForcevector(Nn, Fint, EList, MatProp, qe, re, TEle);
                                                FillExtForcevector(Nn, Fext, EList, MatProp, TEle, w,
                                                        Xsect[isect].sPin, Patm, Matnum);
                                                FillLmpMassmatrix(Nn, N2n, M, EList, MatProp, TEle,
                                                        order, symmdiv, 0.0);
                                                ENewmarkL(Nn, N2n, dt, ND, M, Fext, Fint, Vt, At,
                                                        ktemp1, At1, MatProp, order);
                                                Qupdate(Nn, dt, Qt, Vt, At1);
                                        }
                                        ConnectSteppingmatricies(Qt, Vt, At1, T, NList);
                                        if (Structural) RadialConstraint(RList);
                                        if (Thermal) ThermalConstraint(SList, Tin);
                                }
                        }
                        ntime++;
                        TimeShift(t, ts);
                        t += dt;
//Propellant Regression
                        if (Propburn)
                        {
                                for (isect = 0; isect < Ns; isect++)
                                {
                                        BurnBack(dt, SList, Matnum, MatProp, TEle, order, bb, symmdiv,
                                                CrvApprox);
                                        if (FEPropUpdt) CBModified = CBMod(&Xsect[isect].sNe,
                                                &Xsect[isect].sNn, EList2, NList2, SList, Matnum,
                                                MatProp, TEle, symmdiv, CrvApprox);
                                        if (CBModified) Xsect[isect].sRC = FindRCCB(Xsect[isect].sNodes,
                                                Xsect[isect].sOut, symmdiv);
                                }
                                MassRatio = RemainingMassFraction(Ns, Matnum, Xsect, OrgPropMass);
                        }
                        if ((NonLin) || (FEPropUpdt)) SetFEProp(Ns, Xsect, Matnum, MatProp, order,
                                symmdiv, 0.0, NonLin);
                }
        }
// Graphics
        if (gdt == RunGraphs)
        {
```

```
                if (stressp1 || stressp2) StressCalc(Xsect[sectnum].sElements, MatProp);
                if (vr || vt) Cylvelocity(Xsect[sectnum].sNodes);
                if (Window3 || Window2)
                {
                        glutSetWindow(1);
                        glutPostRedisplay();
                        glutSetWindow(2);
                        if (Window3 && Window2)
                        {
                                glutPostRedisplay();
                                glutSetWindow(3);
                        }
                }
                if (Cyclegraphics) glutPostRedisplay();
                SetWindowTitles();
                if (pause == false) PrintData();
                gdt = 1;
        }
        else gdt++;
}

void main(int argc, char** argv)
{
//Initialization
        InitializeConstantsandSwitches(Settings, &order, &Torder, &scrsize, &Tatm, &h, &Patm, &w2scale,
                &w2sscale, &vscale, &scale, &DataOutNum, &RunGraphs, &pa, &pb, &pc, &pd, &w3min,
                &w3max, &s1max, &s1min, &s2max, &s2min, &dispmax, &velmax, &accelmax, &Tempmax,
                &scla, &sclb, &Penalty, &CFLs, &CFLf, &CFLt, &ND, &PD, &Asmth, &te, &TotNn, &w,
                &Pin, &Tin, &symmdiv, &arbcutoff, &dtQS, &Tpb, &Tstr, &Pfreq, &PAmp, &WaveType,
                &Window2, &Window3, &shwpts, &FullScr, &shwCB, &Plainstrain, &Propburn, &IBFok,
                &Thermal, &Structural, &Hartst, &AccAve, &CrvApprox, &NonLin);
        InitializeGrid(GridData, &Ns, &Nn, &N2n, &Ne, &Nodes, &Nodes2, &Elements, &BLSides,
                &OutrSides, &RCSides, MatProp, &te, &Matnum, Patm, h, order, Plainstrain, Thermal,
                symmdiv, CrvApprox, 0.0);
        InitializeMatricies(Nn, N2n, &M, &Fext, &Fint, &Qt, &Vt, &At, &At1, &ktemp1, &T, order);
        if (Thermal) ThermalConstraint(BLSides, Tin);
        CN = FECnum(CFLs, CFLt, MatProp, Matnum, Structural, Thermal);
        Xsect = new Sectiontype [Ns];
//Setup grid along length of SRM
        PositionGrids(Xsect, Ns, Nn, N2n, Ne, Nodes, Nodes2, Elements, BLSides, OutrSides, RCSides, te, Pin);
//Calculate the initial Propellant Mass
        OrgPropMass = RemainingMassFraction(Ns, Matnum, Xsect, OrgPropMass);
        printf("Ns = %d Ne = %d  DoF = %d Total DoF = %d                \nBeginning Iteration (Press ESC to
                exit.)...               \nPAUSED...! Press 'p' to begin.\r", Ns, Ne, 2*(Nn+N2n), 2*(Nn+N2n)*Ns);
        ts[0] = ts[1] = ts[2] = ts[3] = 0;
        if (IBFok) bb = 0.0;

// Open GL Init.
        MaxVP(&MinX, &MaxX, &MinY, &MaxY, Nn, Nodes);
        glutInit(&argc, argv);
        glutInitDisplayMode(GLUT_DOUBLE | GLUT_RGB);
        scrX = glutGet(GLUT_SCREEN_WIDTH);
        scrY = glutGet(GLUT_SCREEN_HEIGHT);
        if (IBFok == false) Window3 = false;
        if ((Structural == false) && (IBFok == false) && (Propburn == false)) Window2 = false;
        if ((IBFok) && (Propburn)) Structural = false;
```

```
//Create Window #2
        if (Window2)
        {
                glutInitWindowSize(0.485*scrX, 0.465*scrY);
                glutInitWindowPosition(0.00625*scrX, 0.12*scrY);
                glutCreateWindow("Graphs");
                glClearColor(1.0, 1.0, 1.0, 0.0);
                glMatrixMode(GL_PROJECTION);
                glLoadIdentity();
//Make sure window title matches - set time in sec.
                glOrtho(0, w2size, -vscale, vscale, 0, 1);
                glutDisplayFunc(display2);
                glutKeyboardFunc(keyboard2);
        }
//Create Window #3
        if (Window3)
        {
                glutInitWindowSize(scrX, 0.3333*scrY);
                glutInitWindowPosition(0.0, 0.625*scrY);
                glutCreateWindow("UTRoc Output");
                glClearColor(0.0, 0.0, 0.3, 0.0);
                glMatrixMode(GL_PROJECTION);
                glLoadIdentity();
                glOrtho(-1, TotNn, w3min, w3max, 0, 1);
                glutDisplayFunc(display3);
                glutKeyboardFunc(keyboard3);
        }
//Create Window #1
        glutInitWindowSize(0.5*scrX, 0.6666*scrY* (MaxY-MinY)/(MaxX-MinX));
        glutInitWindowPosition(0.498*scrX, 0.09*scrY);
        glutCreateWindow("2D SRM Section");
        if (FullScr) glutFullScreen();
        glClearColor(0.0, 0.0, 0.0, 0.0);
        glutSetCursor(GLUT_CURSOR_FULL_CROSSHAIR);
        glMatrixMode(GL_PROJECTION);
        glLoadIdentity();
        glOrtho(MinX,MaxX,MinY,MaxY,-1,1);
        glutDisplayFunc(display1);
        glutKeyboardFunc(keyboard1);
        glutPassiveMotionFunc (mouse1);
//Run Main Program Loop
        glutIdleFunc(idle);
        glutMainLoop();
        return;
}
```

Pyrolysis Rate Algorithm (see Figure 3-5)

```
C LOOP ON BURN RATE CALC.
 542 CONTINUE
C EROSIVE BURNING
C     GOTO 17965
C         IF(TIME.GE.TIMERD)PRINT *,'1st RB Calc. in ',I,ICHEK1,IBLAH
          RB=RBUR(I)
C     IF(ICHEK1.EQ.2)RB=RB+0.00005
```

```
      IF(ICHEK1.EQ.2)RB=RB+DELR
      IF(REI.LT.1.D0)GO TO 60544
C     FEXPF=DABS(8.*RP*RB/(RO*UO)/SNGL(F))
      FEXPF=DABS(8.*RP*RB/(RO*UO)/(F))
C     IF(FEXPF.GE.100.)FEXPF=100.
      IF(FEXPF.GE.10.)FEXPF=10.

      FBL=8.*RP*RB/(DEXP(FEXPF)-1.)/RO/DABS(UO)
      HBL=0.0
      REXPF=DABS(RP*RB*CP/HST)
      IF(REXPF.GT.100.)REXPF=100.
C     IF(REXPF.GT.10.)REXPF=10.
      HC=RP*RB*CP/(DEXP(REXPF)-1.)
      TCORE=TF
      IF(TO.GT.TF)TCORE=TO
      RVC(I)=HC*(TCORE-TS+TEX)/(RP*CS*(TS+TEX-TI))
      GO TO 60543
60544 RVC(I)=0.
      HBL=0.
      FBL=0.
C 60543 IF(IGA.EQ.1.OR.PO.LT.102000.)R0=C*(PO/1.E3)**AN
60543 CONTINUE
      PEXN(I)=PO
C     GAIN8=1.00
C     PEXOO=PEXO(I)
C     IF(IBLI.EQ.0)PEXO(I)=PEXO(I)+GAIN8*(PEXN(I)-PEXO(I))
      PEXO(I)=PEXN(I)
C     IF(TIME.GT.TIMBEG.AND.TIME.LT.TIMEND)THEN
C     IF(I.GT.110.AND.I.LE.J)THEN
C     DELPST=300.D6
C     DELPST=0.D6
C     PEXO(I)=PO+DELPST
C     END IF
C     END IF
C     GAIN9=0.9999
C     PCDOT=PEXO(I)-PEXOO
C     IF(PCDOT.GT.0.1)PCDOT=0.1
C     PEXO(I)=PEXO(I)-GAIN9*PCDOT
      PES=PEXO(I)
      IBLI=1
C     PES=PO
      IF(IGA.EQ.1.OR.PO.LT.102000.)R0=C*(PES/1.E3)**AN
      IF(IGA.EQ.2.AND.PO.GE.102000.)R0=C*(PES/1.E3-101.)**AN
      IF(PO.LE.PLIM)R0=PO/1.E3/(ALOW+BLOW*(PO/1.E3)**(2./3.))
      IF(TIME.LT.TSP)RAC(I)=0.
C     IF(TIME.LT.TSP)GO TO 207
          GOTO 207
C
C FE MODULE MOD
C RMT IS BASELINE BURN RATE BEFORE ACCELERATION EFFECTS
C RB AND RBUR(I) ARE OVERALL BURN RATE IN PROGRESS OF ITERATION
C IF TIME LT. TSP, MEAN BURN RATE ASSIGNED TO ALL ELEMENTAL SEGMENTS
      RMT=R0+RVC(I)
      IF(IFE.EQ.1)GO TO 17965

17965 CONTINUE
```

```
C
      IF(TIME.LT.TSP)GO TO 94000
C Mod to original algorithm - SL***
C     SUM=0.
C     SUMS=0.
C     DO 68877 IA=1,ITOT(I)
            ICHEK1=1
      ACC=ANS(I,IA)
      IF(TIME.LT.TSP) ACC=-ACCMIN
C           ACC=0
      PHI=0.0
C     ACCXLIM=500000.
C     IF(ACC.LT.-ACCXLIM)THEN
C     ACC=-ACCXLIM
C     RNEW(I)=ACC*(2.D0*DTEH)**2+2.*ROX(I)-ROX2(I)
C     IF(I.EQ.1)RNEW(I)=ACC*(2.D0*DTEH)**2+2.D0*RIX(I)-ROX(I)
C     END IF
C     IF(ACC.GT.ACCXLIM)THEN
C     ACC=ACCXLIM
C     RNEW(I)=ACC*(2.D0*DTEH)**2+2.*ROX(I)-ROX2(I)
C     IF(I.EQ.1)RNEW(I)=ACC*(2.D0*DTEH)**2+2.D0*RIX(I)-ROX(I)
C     END IF
C     APROP(I)=ACC/9.81
C FE MODULE MOD, ORIGINAL OUTSIDE WALL ACCELERATION FOR CYLINDRICAL
C GRAIN MOTOR CASE WAS ACCW(I), M/S^2
C     ACASE(I)=ACCW(I)/9.81
C TEMP
C     IF(I.GT.40)THEN
C     ACASE(I)=ACASE(40)
C     ACC=ACASE(I)
C     ETAT(1,I)=ETAT(1,40)
C     END IF
C90254 CONTINUE
C THRESHOLD MOD ON EFFECTIVE NORMAL ACCELERATION
C SPECIFY THRESHOLD ACCEL. (G)
C     ACCTHR=100.
C     ACCTHR=0.
C     ACC=ACC+ACCTHR*9.81
      IBLO=1
C     IF(ACC.GE.-0.01)ACC=-0.01
      IF(ACC.GE.-ACCMIN)ACC=-ACCMIN
      IF(ACC.LT.-ACCMAX1.AND.ACC.GE.-ACCMAX2)ACC=-ACCMAX1
      IF(ACC.LT.-ACCMAX1)ACC=-ACCMAX1
C           DHS=0.
      IF(TIME.LT.TIMERD) DHS=-3451000.
            IF(TIME.GE.TIMERD) DHS=0.
C     IF(ACC.LT.-ACCMAX2)DHS=DHS1
C     IBLO=1
C     END IF
C TEMP, MULTIPLIER ON ACCEL. SENSITIVITY
      AMULT=1.0D0
C     AMULT=0.575D0
      IF(ACC.LT.-ACCMAX2)AMULT=0.53D0
      DEL0=AK/(RP*RMT*CP)*DLOG(1.+
     * AMULT*CP/CS*(TF-TS+TEX)/(TS+TEX-TI-DHS/CS))
63509    CONTINUE
```

```
C         IF(TIME.GE.TIMERD)PRINT *,'63509 ',I,IA,ICHEK1,RB
      RA=ACC*PO/RB*DEL0/(RGAS*TF*RP)*RMT/RB
C TEMP, MULTIPLIER ON ACCEL. SENSITIVITY
C     RA=AMULT*RA
      TERMC=RP*CP*DEL0/AK*(RB+RA)
C     IF(TERMC.GT.20..OR.TERMC.LT.-20.)GO TO 63510              **SL
          IF(TERMC.EQ.0.)TERMC=1.E-6
      GOTO 63505
63510 CONTINUE
      PRINT *,'RB RA RMT ',I,RB,RA,RMT,TERMC
      IBLAH=IBLAH+1
      RBUR(I)=0.003+DELR1
C TEMP
C     RBUR(I)=C*(PRE(I)/1.E3)**AN+DELR1
C
      IF(IBLAH.GT.1)DELR1=DELR1+DELR10
      ICHEK1=1
      IF(IBLAH.GT.20000)THEN
      WRITE(6,*)' GRP 63510 CYCLE ON IBLAH TOO MUCH, I = ',I,' IA = ',IA
C     WRITE(6,*)' RBUR(I-1) RBUR(I+1) ',RBUR(I-1),RBUR(I+1)
C     WRITE(6,*)' ACASE(I-1) ACASE(I+1) ',ACASE(I-1),ACASE(I+1)
C     WRITE(6,*)' ACASE(I) I ',ACASE(I),I
      WRITE(6,*)' ACC RA RB RBUR REXPF REI HST HC TERMC ',ACC,RA,RB
          * ,RBUR(I),REXPF,REI,HST,HC,TERMC
      WRITE(6,*)' R0 RVC RMT ',R0,RVC(I),RMT
      WRITE(6,*)' PO RO UO DEL0 ',PO,RO,UO,DEL0
      WRITE(6,*)' TIME = ',TIME
      PRINT*, 'IBLAH > 20000 3168
          STOP
C         GOTO 67056
C     RBUR(I)=RBUR(I+1)
C     GO TO 550
      END IF
C RETURN TO EROSIVE BURNING
      GOTO 542
63505 RACC=CP/CS*(TF-TS+TEX)/(TS+TEX-TI-DHS/CS)*
     * AMULT*(RB+RA)/(DEXP(TERMC)-1.)
C
      RAC(I)=RACC-RMT
63207 FR=R0+RVC(I)+RAC(I)-RB
      IF(ICHEK1.EQ.1)GO TO 66604
          IF(RB-RBO.EQ.0.)PRINT *,'63207 RB-RBO = 0'
      TERMX=(FR-FRO)/(RB-RBO)
          IF(TERMX.EQ.0.)PRINT *,'63207 TERMX = 0'
      RBUR(I)=DBLE(RB-FR/(TERMX))
C     IF(DABS(RBUR(I)).GT.1.D0)THEN
      IF(DABS(RBUR(I)).GT.2.D0)THEN
          PRINT *,'RBUR(I) > 2.0 - 63207'
      IBLAH=IBLAH+1
      RBUR(I)=0.003+DELR1
          IF (RAC(I).GT.0.02) RAC(I)=0.02
C TEMP
C     RBUR(I)=C*(PRE(I)/1.E3)**AN+DELR1
      IF(IBLAH.GT.1)DELR1=DELR1+DELR10
      ICHEK1=1
      IF(IBLAH.GT.20000)THEN
```

```
      WRITE(6,*)' GRP 207 CYCLE ON IBLAH TOO MUCH, I = ',I
C     WRITE(6,*)' RBUR(I-1) RBUR(I+1) ',RBUR(I-1),RBUR(I+1)
C     WRITE(6,*)' ACASE(I-1) ACASE(I+1) ',ACASE(I-1),ACASE(I+1)
C     WRITE(6,*)' ACASE(I) RB RAC(I) I ',ACASE(I),RB,RAC(I),I
      WRITE(6,*)' R0 RVC(I) I ',R0,RVC(I),I
      WRITE(6,*)' ACC RA RMT REI REXPF HST HC ',ACC,RA,RMT,REI,REXPF,
     * HST,HC
      WRITE(6,*)' PO RO UO ',PO,RO,UO
      WRITE(6,*)' TIME = ',TIME
            RBUR(I)=0.04
            GOTO 66550
C     STOP
C           GOTO 67056
C     RBUR(I)=RBUR(I+1)
C     GO TO 550
      END IF
C RETURN TO EROSIVE BURNING
      GOTO 542
      END IF
      DELCON=1.0E-6
      IF(ICHEK1.GT.20)DELCON=1.0E-4
      IF(DABS((RB-RBUR(I))/RBUR(I)).LE.DELCON)GO TO 66550
66604 FRO=FR
      RBO=RB
      ICHEK1=ICHEK1+1
C MOD, BASED ON AN PROBLEMS ON RB
      IF(ICHEK1.GT.40)THEN
      IF(I.GT.1)THEN
      RBUR(I)=0.003+DELR1
C TEMP
C     RBUR(I)=C*(PRE(I)/1.E3)**AN+DELR1
      END IF
      IF(I.EQ.1)RBUR(I)=0.003+DELR1
            IF (RAC(I).GT.0.045) RAC(I)=0.045
C TEMP
C     IF(I.EQ.1)RBUR(I)=C*(PRE(I)/1.E3)**AN+DELR1
      IBLAH=IBLAH+1
      IF(IBLAH.GT.1)DELR1=DELR1+DELR10
      ICHEK1=1
      IF(IBLAH.GT.20000)THEN
            PRINT *,'ICHEK1 > 40 - 66604'
      WRITE(6,*)' 604 GRP, TOO MUCH IBLAH ON OUTER ITER LIMIT '
      WRITE(6,*)' RBUR(I) TIME I ',RBUR(I),TIME,I
C     WRITE(6,*)' ACASE(I) TIME I ',ACASE(I),TIME,I
C     STOP
            PRINT*, 'IBLAH > 20000'
            RBUR(I)=0.04
            GOTO 66550
C           STOP
C           GOTO 67056
C     RBUR(I)=RBUR(I+1)
C     GO TO 550
      END IF
C RETURN TO EROSIVE BURNING
      GOTO 542
C     GOTO 63509
```

```
      END IF
      IF(TIME.LT.TSP)GOTO 542
            RB=RBUR(I)
            IF(ICHEK1.EQ.2)RB=RB+DELR
            GOTO 63509
C     WRITE(6,*)' 66550 I RB RHO UX ',I,RBUR(I),RHO(I),UX(I)
66550 CONTINUE
            IF(TIME.LT.TSP)GOTO 94000
C NEW, ALLOW FOR LATERAL AND LONGITUDINAL ACCEL. ON NORMAL ACCEL.
      IF(ACC.GE.-ACCMIN)GO TO 93666
C
C TEMP
C     ALM(I)=50.*9.81
C
      PHI=DATAN(DABS(DSQRT(ALM(I)**2+ALATS(I,IA)**2)/ACC))
            IF(DABS(PHI).GT.PHIMAX)PHI=PHIMAX
      GAX0=RA*RP
      BX=AMULT*CP/CS*(TF-TS+TEX)/(TS+TEX-TI-DHS/CS)
      RMT=R0+RVC(I)
      RB0=RBUR(I)
      RB=RB0+0.0001
      ICHK1=1
C           IF(TIME.GE.TIMERD)PRINT *,'ALAT1 in ',I,IA,RB
91666 CONTINUE
      RBT=RB
            IF(ICHK1.GT.20)DELCON=1.0E-4
            IF(ICHK1.GT.2000) GOTO 93666
      IF(ICHK1.EQ.2)RBT=RBT+0.0001
      GAX=GAX0*(DCOS(DATAN(8.0*(RMT/RBT)**3*DTAN(PHI))))**2
      TERMX=CP*DEL0/AK*(RP*RBT+GAX)
      FRB=RBT-BX*(RBT+GAX/RP)/(DEXP(TERMX)-1.D0)
      IF(ICHK1.EQ.1)GO TO 92666
      RB=RBT-FRB/((FRB-FRBO)/(RBT-RBO))
      IF(DABS((RB-RBT)/RBT).LT.DELCON)GO TO 93666
92666 ICHK1=ICHK1+1
      RBO=RBT
      FRBO=FRB
      GO TO 91666
93666 CONTINUE
C     IF(ACC.GE.-ACCMIN)RB=RMT
C           IF(TIME.GE.TIMERD)PRINT *,'ALAT1 out ',I,IA,RB
            RBS(I,IA)=RB
      SUM=SUM+RBS(I,IA)*DSA(I,IA)
      SUMS=SUMS+DSA(I,IA)
68877 CONTINUE
      RBAVE=SUM/SUMS
      RB=RBAVE
C
C RETURN TO ORIGINAL PROGRAM WITH MEAN BURN RATE RB FOR SECTION
C
93000 CONTINUE
      RBUR(I)=RB
94000 CONTINUE
```