An adaptive multi-material Arbitrary Lagrangian Eulerian algorithm for computational shock hydrodynamics.

Barlow, Andrew

UNIVERSITY OF WALES SWANSEA

An Adaptive Multi-Material Arbitrary Lagrangian Eulerian Algorithm for Computational Shock Hydrodynamics

By
Andrew Barlow

Department of Civil Engineering

Supervisor
Professor Nigel Weatherill

29th January 2002

This thesis is submitted for the degree of Doctor of Philosophy

ProQuest LLC.
789 East Eisenhower Parkway
P.O. Box 1346
Ann Arbor, MI 48106 – 1346

Abstract

The Arbitrary Lagrangian Eulerian (ALE) method was first proposed 25-30 years ago as a hybrid technique that attempts to exploit the strengths of traditional Lagrangian and Eulerian methods for shock hydrodynamic problems without suffering any of their deficiencies. However, most published ALE algorithms are Simple ALE (SALE) schemes, that is they restrict all material interfaces to remain Lagrangian. This is a great simplification and it also severely limits the range of applications to which an ALE hydrocode can be applied.

This thesis describes a robust and accurate adaptive multi-material ALE algorithm. The algorithm offers two types of material interface treatment; a Lagrangian slide method and a Volume of Fluid (VOF) method, which employs a modified SLIC interface reconstruction technique. The slide treatment is more appropriate for low deformation and provides a natural way to include interface physics, whilst the VOF approach is a very robust technique which is well suited to high interface deformation. The research that has been carried out to develop the multi-material ALE algorithm can be divided into four main areas; Lagrangian interface methods, multi-material interface treatments, mesh adaption algorithms and multi-material advection methods for non-orthogonal unstructured grids.

The traditional treatment of multi-material cell components during the Lagrangian phase, that is used in most multi-material Eulerian and ALE hydrocodes has been re-examined and an alternative approach proposed. The new approach attempts to introduce sub-zonal Lagrangian physics within multi-material cells. This approach conserves total energy and produces more realistic state variables for the partial components of multi-material cells.

This thesis will describe the complete multi-material ALE algorithm, the supporting research that has been carried out in these four main areas, and finally, present results from some of the calculations that have been performed to verify and validate the algorithm.

Acknowledgements

I would like to thank Professor N Weatherill (Swansea University, Civil Engineering Department) and Mr D Pilkington (formerly the Group Leader for Computational Physics at the Atomic Weapons Establishment (AWE), Aldermaston, UK and now working at the Lawrence Livermore National Laboratory (LLNL), USA) for their help and guidance throughout the course of this work.

I am also grateful to other members of the Computational Physics Group at AWE for many helpful discussions and would especially like to thank John Whittle who developed the pure Lagrangian hydrocode that formed the starting point for this research.

I dedicate this thesis to my wife Rachel, our daughter Abigail, my mum and my dad for all their support and encouragement.

I also acknowledge the financial support of AWE.

Nomenclature

R	Region number
L_R	First logical mesh dimension for region number R
K_R	Second logical mesh dimension for region number R
$\mathbf{x}$	Vector of nodal coordinates
z, r	Cartesian coordinates for axisymmetric geometry
x, y	Cartesian coordinates for plane geometry
t	Time
ρ	Density
p	Pressure
V	Volume
ε	Specific internal energy
$\mathbf{u}$	Velocity vector
u, v	Velocity components
q	Artificial viscosity
M_e	Element Mass
M_{nodal}	Nodal Mass
c	Sound speed
C_L, C_Q	Linear and quadratic coefficients of artificial viscosity
C	Courant number
N_i	Shape function
W_i	Weight function
S_{xx}, S_{xy}, S_{yy}	Stress deviators
Y	Yield strength
μ	Shear modulus
$\varepsilon_{xx}, \varepsilon_{xy}, \varepsilon_{yy}$	Strain rates
e_{xx}, e_{xy}, e_{yy}	Strain rate deviators
ω_{xy}	Spin tensor
W^p	Plastic work
W^e	Elastic work
ε^p	Equivalent plastic strain
γ	Ratio of specific heats
ξ, η	Isoparametric coordinates

ϕ, ψ	Winslow's mesh line potentials
Δs	Minimum distance across Element
f	Volume fraction

Subscripts

i	Element number
j	Local node number
k	Material number

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Hydrocodes are large computer programs used to simulate shock hydrodynamics problems [1]. They differ from Computational Fluid Dynamics (CFD) codes in that they must be applicable to solid materials as well as to liquids and gases. This demands that the numerical schemes used are compatible with any general equation of state, include additional physics such material strength, and provide realistic material interface treatments.

1.1 Lagrangian and Eulerian methods

There are two main types of hydrocode in common use, that is Lagrangian and Eulerian codes. Lagrangian codes solve the Euler equations assuming a Lagrangian frame of reference, the computational mesh moving with the material. Eulerian codes solve the Euler equations in the Eulerian frame of reference, the mesh remaining fixed with the material flowing through it. In practice Eulerian codes are normally implemented using split methods, where one or more Lagrangian step is performed each time step, followed by an advection step which remaps the solution back on to the initial fixed grid. Both the Lagrangian and Eulerian methods have strengths and weaknesses: in practice this makes neither suitable for all applications, the best method being chosen on a problem by problem basis. The advantages and disadvantages of the Lagrangian and Eulerian code descriptions will now be discussed, before moving on to consider the potential benefits of ALE methods.

The initial mesh for a Lagrangian simulation is mapped onto the materials of interest and can conform well to the initial geometry of each material. This means the initial Lagrangian mesh should be more ideal, or have a higher quality in some respects, than an Eulerian grid. However, since the Lagrangian mesh follows the material flow, if the flow becomes highly distorted, then so does the mesh. This means that the Lagrangian mesh will be lower in quality than the fixed Eulerian mesh for highly distorted flows and may cause the Lagrangian calculation to fail. Essentially,

while the flow is well behaved the Lagrangian mesh has resolution where it is required, does not imprint on the solution and has an effective numerical stencil which is generally aligned with the principle flow directions. Once the flow becomes distorted the Lagrangian mesh leads to a poorly defined numerical stencil with which to calculate derivatives, and may be prone to artificial stiffness introduced by elements locking under certain modes of distortions.

However, the Lagrangian mesh framework allows mesh resolution to be focused where it is required, and computational cells are only required where there is material in the problem, as opposed to an Eulerian code which requires cells throughout all the spatial domain potentially of interest. Given that fewer cells are required for comparable resolution and there is no advection phase, Lagrangian codes are usually computationally less expensive in run time and memory. However, it is worth noting that if the Lagrangian mesh becomes severely distorted anywhere in the domain, the time step can drop severely, which will not happen in a Eulerian computation; in such a case Eulerian calculations can be less expensive.

In a lot of the applications where hydrocodes are applied, material interfaces are very important. If the material interfaces are not too distorted then the Lagrangian approach again has significant advantages, since with nodes placed on material boundaries, the interface can be tracked accurately. It is also possible to decouple the tangential forces from the materials to either side of the interface, allowing slip or slide to occur. Slide lines also provide a natural way to include interface physics such as friction, void opening and void closure. However, if the interface becomes distorted, for example if the interface is unstable, then this will soon lead to mesh tangling and a Lagrangian calculation will fail. An Eulerian code will cope with severe interface distortion. However, since Eulerian interfaces are constructed on top of the mesh, and also remembering that there is advection across the interface, they will not represent the interface as well as a Lagrangian approach. The Lagrangian description also has benefits for external boundaries. The fixed nature of the Eulerian mesh means that there will be a point where material expands out of the domain and is no longer included in the calculation, whereas the Lagrangian description follows the expansion of the material.

The presence of advection in the Eulerian description has the disadvantage that it introduces a level of smoothing or numerical diffusion on the solution. This can reduce the calculational accuracy to some extent, but can also be beneficial in acting to further reduce numerical oscillations in the solution. In many hydrocodes the advection step is operator split rather than truly multi-dimensional which may also compromise the solution. However, it is fair to say that a distorted Lagrangian grid can also compromise the 2D or 3D nature of a solution.

The final distinction between the two descriptions is the problem of adding other physics packages to the two types of code. In general it is simpler to implement additional physics in an orthogonal grid Eulerian code. However, where the Lagrangian grid has a higher mesh quality it potentially offers a more accurate solution

for the other physics packages as well as the hydrodynamics.

In practical applications as neither technique is ideal for the entire problem, it is not uncommon to start a calculation on one type of code then link the solution across to the other to complete the problem.

1.2 Arbitrary Lagrangian Eulerian (ALE) methods

An ALE hydrocode can be viewed as an attempt to maintain an optimum mesh, to exploit the advantages of both the Lagrangian and Eulerian descriptions by using the most appropriate description for different parts of the problem, or as an essentially Lagrangian code with an intelligent automatic rezone capability to keep the calculation running. The latter description illustrates that ALE can also be viewed as an adaptive mesh scheme. Alternatively, ALE can be described as solving the Euler equations in a reference frame that is arbitrary and varies with space and time.

These different descriptions all give an impression of the strengths of the ALE method and how it can be applied. It has the potential to provide a single code that can tackle a wide range of applications without the need to transfer the solution across to another code to complete the problem. It also has the potential to offer better overall accuracy than either the Lagrangian or Eulerian approach. It is clear that better mesh quality can be established for the Lagrangian phase, as it is possible to start with the normal Lagrangian mesh, but as the flow starts to become distorted the mesh can be relaxed to improve its quality, but still retain a reasonable representation of the bulk flow. The calculation can also retain further significant Lagrangian benefits such as slide where interfaces are well behaved, and materials that do not undergo severe distortion can be treated as pure Lagrangian, minimising the numerical diffusion that advection methods inevitably introduce. The mesh movement can also be weighted to improve the resolution of shocks, material interfaces, materials or regions. This provides an inexpensive, albeit if penalty based, mesh refinement capability.

The preservation of symmetry can be very important for some applications, such as Inertial Contained Fusion (ICF). These problems require very high flow convergence and shock focusing to achieve ignition. It is important to be able to assess the impact of symmetry perturbations on such problems. This can only be done if the unperturbed problem remains symmetric calculationally. ALE mesh movement techniques can be used to keep the mesh isotropic with respect to the symmetry of interest. This is very important for 2D axisymmetric calculations, which normally rely on an area weighted scheme to preserve spherical symmetry. ALE mesh movement algorithms can then be used to preserve equal angular zoning, which is required for such an area weighted scheme to be effective, and will minimise further symmetry errors introduced by the advection step. In contrast, orthogonal grid

Eulerian codes may introduce symmetry errors in both Lagrangian and advection steps.

The computational cost to obtain a given level of solution accuracy should also be lower for a multi-material ALE code than either a Lagrangian or Eulerian codes. This is because ALE calculations generally run with higher time steps than Lagrangian codes, introduce less advection than Eulerian methods and can focus mesh resolution more effectively throughout the simulation than either Lagrangian or Eulerian codes.

Although ALE schemes have been in existence for 25-30 years [2, 3, 4, 5, 6, 7, 8, 9] most of the published work has ignored the complexities of material interfaces. The methods appearing in the literature are described as SALE or Simple ALE schemes as they impose the restriction that the interfaces must remain Lagrangian; an approach which is adequate for many CFD applications, but not for most hydrocodes applications. If ALE codes are to be successful in challenging both traditional Lagrangian and Eulerian hydrocodes then they must provide robust and accurate interface treatments that fully exploit the accuracy benefits of Lagrangian schemes for low deformation and the robustness of Eulerian schemes for high material deformation.

The present study has focused on the development of an adaptive 2D multi-material Arbitrary Lagrangian Eulerian hydrocode. This has required the development of robust and accurate multi-material advection methods and mesh movement techniques for unstructured non-orthogonal grids. A strength of the resulting ALE scheme is that it has two alternative treatments for material interfaces, Lagrangian slide lines or a Volume of Fluid (VOF) scheme based on the Simple Line Interface Construction (SLIC) scheme [10]. This flexibility enables the more accurate Lagrangian slide approach to be used when ever an interface is not too distorted. It also provides a natural frame work for the inclusion of friction and other interface physics such as void opening and void closure. The Lagrangian approach can also be used at the start of a problem and the interface then merged and treated as a VOF multi-material interface once the interface distortion becomes too severe. The traditional treatment of multi-material cell components during the Lagrangian phase, also used in most multi-material Eulerian hydrocodes, has also been re-examined and an alternative approach proposed. The new approach attempts to introduce sub-zonal Lagrangian physics within multi-material cells. This approach conserves total energy and produces more realistic state variable for the partial components of multi-material cells.

1.3 Staggered versus Unstaggered schemes

The first successful numerical method for treating shocks was suggested by von Neumann [11] in the 1950's. Von Neumanns's idea was simply to add additional

dissipation or artificial viscosity to smear shock discontinuities over a number of zones and so avoid oscillations near shocks. Since then the two communities developing CFD codes and hydrocodes have been divided. The CFD community has focussed on improving the resolution or capture of shocks through the development of high resolution shock capturing methods, which will resolve shock fronts typically over 2-3 zones, but require all variables to be collocated either at cell vertices or cell centres. In contrast, most hydrocodes are still based on staggered mesh schemes, where positions, velocities and accelerations are stored at the nodes, and all other variables are at cell centres combined with artificial viscosity methods. Considerable work has however been put into improving the artificial viscosity methods used [12, 13, 14, 15, 16, 17, 18, 19] particularly by the defence hydrocode community, so that shock capturing with these methods is now almost comparable with what can be achieved with high resolution schemes.

The high resolution shock capturing techniques used by the CFD community are of two main types: linear-hybridized schemes and Godunov methods. The former are hybrid schemes which combine high and low order methods, a notable example being Boris and Books Flux-Corrected transport (FCT) algorithm [20]. The problem with high order schemes is that they produce oscillations near shock discontinuities. This is avoided for the linear-hybridized schemes by applying a high order scheme in the smooth parts of the flow, smoothly blended into lower order method near discontinuities. This produces a method which produces a monotone solution throughout the domain. The monotonicity is preserved due to the large dissipative truncation error of the lower order scheme. Godunov methods [21] are based on the idea of treating each zone as containing a piecewise constant state, then solving the non-linear interaction of these states exactly as a series of 1D Riemann problems at each cell boundary. The results from the separate Riemann problems are then averaged to find the updated flow solution. This approach is an accurate and well behaved way to treat shocks, but it is only first order accurate, which makes it very diffusive. However, higher order Riemann based Godunov solvers have been developed, which retain the benefits of the original approach for shocks. These schemes are normally based on the use of higher order interpolation methods within a cell.

Godunov methods which solve the Riemann problem have further deficiencies. These include the need for accurate sound speeds, or a good approximation to them, for all states that a material can reach during a simulation. This is not always possible and can be computationally very expensive for the more complicated equations of state used in hydrocode simulations. It is also difficult to define wave speeds for the other types of physics such as material strength [22] required in hydrocode simulations. An operator split approach could however be taken to overcome this problem [22]. However, this is likely to reduce the benefits of employing a high resolution method.

Unstaggered methods are not favoured by the hydrocode community for a number of reasons. In introducing material strength, for example, the strain rate tensor

must be calculated, and it is difficult to do this accurately on an unstaggered mesh. It is also difficult to produce accurate material interface algorithms on unstaggered meshes. Unstaggered schemes may not preserve symmetry as well as staggered mesh schemes due to the spatial operator splitting required. In contrast, staggered mesh schemes are naturally multi-dimensional for Lagrangian methods and for the Lagrangian step of the split methods. Some effort has however been put into developing multi-dimensional high resolution schemes over the last few years by a number of researchers [23]. However, these methods have yet to fully mature. Given these concerns it has also been decided to employ a staggered grid approach in this work.

1.4 Mesh Adaption

Adaptive mesh schemes dynamically modify the computational grid, as the problem evolves either to improve the accuracy obtained for a given computational cost, or to reduce the cost of obtaining a given accuracy. This can be achieved either by improving mesh quality or focusing the mesh resolution where and when it is most required.

Mesh quality is hard to quantify. However, certain desirable properties can be defined. A high quality mesh will maximise the computational time step and not imprint upon the solution. The former suggests the need for a fixed uniform orthogonal grid as normally employed in Eulerian hydrocodes. However, it contradicts the latter requirement as an orthogonal mesh will inevitably imprint on the solution when the flow is not aligned with the mesh. A mesh that is aligned with material interfaces, principle flow directions and any underlying symmetry in the problem will tend to minimise mesh imprinting on the solution and so has a high quality.

If an unstructured mesh of triangular elements is used, it is possible to improve mesh quality and introduce mesh refinement by reconstructing part, or all, of the mesh by using Delauney triangulation with source points to focus refinement [24]. However, hydrocodes normally employ quadrilateral elements, which do not lend themselves to totally unstructured mesh adaption. The Eulerian Adaptive Mesh Refinement (AMR) scheme developed by Berger [25, 26, 27] and Quirk [28] provides a natural way to introduce mesh refinement/derefinement, but cannot be used to improve mesh quality. The AMR technique essentially defines a course background mesh upon which hierarchical mesh refinement can be built dynamically as required. Error estimators are used to determine where refinement and derefinement is required during the calculation. Some examples of the error estimators used include the solution gradient, estimates of the local truncation error and physics followers. In pure hydrodynamics problems the physics followers may simply be used to force discontinuities in the flow such as shocks and material interfaces to be held at the finest mesh resolution. This overcomes many of the numerical difficulties

of treating refinement boundaries. Lagrangian hydrocodes require mesh adaption mainly to improve mesh quality. The Lagrangian method is already computationally more efficient than an Eulerian treatment and naturally refines the mesh in the vicinity of shocks. But it is unable to robustly handle severe deformation where mesh tangling occurs. The ALE method discussed above provides a natural way to improve mesh quality through introducing arbitrary mesh movement at the end of each time step. This technique can also be used to improve mesh resolution in some parts of the domain by pulling zones into a feature of interest. This is known as a penalty based adaption method, as the mesh topology is fixed, and one region is refined at the expense of another. It is a computationally more efficient approach as it does not have the data structure overheads of the AMR technique.

However, only limited mesh refinement can be achieved compared to AMR methods. It is believed by the author that the best approach may be a combined ALE and AMR code, the ALE techniques used to extract as many of the Lagrangian benefits as possible while retaining mesh quality and robustness. The AMR technique only providing the additional mesh refinement that can not be achieved with the less expensive mesh movement technique. This should offer gains in efficiency and accuracy and it should keep the data structure overhead of AMR method down to a minimum.

1.5 An introduction to the adaptive multi-material ALE algorithm

The adaptive multi-material ALE algorithm has been developed principally for highly convergent hydrodynamics problems such as Inertial Contained Fusion (ICF) capsule implosions, which require accurate resolution of peak compressions and the preservation of spherical symmetry. The ALE algorithm also provides a natural framework for introducing interface physics and is well suited to modeling material interfaces under severe deformation. This is important for assessing the influence of drive perturbations on ICF, and also makes the technique well suited to shaped charges, explosively formed projectiles (EFPs), projectile impact problems and other applications where material interfaces are important.

A Lagrangian approach has significant advantages for these problems, but is limited in its use by its lack of robustness. A number of palliatives can be employed in pure Lagrangian calculations to increase the robustness of such codes, but these are generally problem dependent, and often compromise the fidelity of the solution. The ALE method developed here retains robustness by adapting the mesh by moving nodes to improve mesh quality at the end of each time step, but strives to retain as much Lagrangian character to the mesh as possible.

Given this philosophy, the algorithm employs a split scheme with separate Lagrangian and advection steps. The starting point for the research was a well validated 2D axisymmetric finite element Lagrangian hydrocode developed by Whittle at AWE [29]. A predictor corrector time discretization is used for the Lagrangian step giving second order accuracy in time. A staggered mesh is employed in order to overcome the concerns already expressed over unstaggered schemes for hydrocode applications. The spatial discretization employs explicitly integrated bilinear quadrilateral finite elements. A scalar monotonic artificial viscosity is used to introduce irreversible shock heating and stabilise the scheme for shock discontinuities. This viscosity is calculated using monotonically limited velocity gradients for each cell edge. This offers significant advantages over non-monotonic viscosity treatments in that it produces sharper shock fronts (typically smearing the discontinuity over 2-3 zones compared to 4-5 for non-monotonic artificial viscosity forms) and introduces less artificial shock heating. It produces comparable results to Riemann based Godunov codes.

The computational mesh used by the code is indirectly addressed using connectivity arrays typical of most unstructured finite element codes [30, 31]. However, the algorithms that have been developed do assume some structure. The mesh for each problem is constructed using a separate generator program, where the code user builds up problems from a series of logically rectangular mesh blocks or regions. The regions represent different materials and can be arbitrarily connected. The interfaces between regions can be treated as merged interfaces or slide lines. The elements on either side of a merged interface are restricted to either line up exactly 1:1, or be in ratios of 1:2, 1:3 or 1:4. There is no restriction on the mesh either side of a slide line. It is this flexibility in how the logical rectangular blocks can be connected that leads to the need for an unstructured data layout.

The code may be used to perform pure Lagrangian or pure Eulerian calculations. However, although these options may be useful for some problems they really exist more for validation purposes. In order to realise the strengths of the code the ALE mesh movement algorithms must be exploited. This requires the user first to decide which regions and interfaces, if any will, remain Lagrangian. It may be desirable to treat some interfaces as Lagrangian to enable slide or frictional slide to be used or simply to retain some Lagrangian character to the mesh adaption. An interface or region can also be defined to be initially Lagrangian until some time is reached and it is allowed to adapt.

The user currently selects which mesh movement algorithms are to be applied to each ALEing region. This could be automated by introducing error estimators to determine whether the mesh movement algorithm can move a node and possibly even to determine locally how the node should be moved. However, although some research has been performed on this topic, it is considered preferable to leave the control with the user. This choice was made as it is difficult to define a mesh movement algorithm that suits all applications and the best solutions are generally

obtained when the code user invests some effort in thinking about the best way to move the mesh, both to achieve a robust calculation and to focus mesh resolution to capture the physics of interest.

The mesh movement strategy defined by the user for each region is expressed in terms of three main ingredients: interface movement, internal node movement, and constraints. It is important to keep some interfaces Lagrangian to both facilitate the inclusion of interface physics such as friction and to introduce some Lagrangian character to the internal mesh movement. However, nodes can require redistribution along these Lagrangian interfaces to maximise mesh quality and robustness. A number of techniques have also been developed for constraining the mesh movement either to further enhance the Lagrangian character of the mesh movement, or allow the user to focus mesh resolution on features of interest.

1.6 Thesis Outline

This thesis attempts to describe the adaptive multi-material ALE algorithm in sufficient detail to enable it to be implemented in a Lagrangian code. The research that has been performed to develop the current algorithm and investigate potential improvements to the scheme is then presented, along with calculations, to verify and validate the method.

A description is first given in chapter 2 of the mesh and data structure used in CORVUS to implement the adaptive multi-material ALE algorithm. In chapter 3 the 2D axisymmetric finite element pure Lagrangian hydro algorithm that formed the starting point for this research is described, along with some improvements that have been made to the scheme during the course of this research.

In chapter 4, the Lagrangian slide algorithms in CORVUS are described, along with the extension of the scheme by the author to include a model for dynamic friction. Three forms for the frictional force are introduced which can be applied in isolation or as a linear combination.

The mesh adaption algorithms that have been developed for improving grid quality and introducing local refinement are described in chapter 5. In chapter 6 the single and multi-material advection methods that have been developed for non-orthogonal grids are presented.

In chapter 7 the treatment of multi-material cells during the Lagrangian phase is revisited, its deficiencies illucidated and demonstrated in sample calculations. A novel method developed by the author is then presented, which attempts to build in subzonal Lagrangian physics for multi-material cells. Numerical solutions are then presented which demonstrate the benefits of this new approach. The extension of the method to introduce other interface physics, such as void closure, is also discussed.

The adaptive multi-material algorithm has been applied to many test problems

and real applications during its development. A sample of these test problems are presented throughout the text of the thesis where relevant. However, it is also important to demonstrate the power of this technique on real applications. Chapter 8 includes a selection of validation and application problems that have been performed.

The main conclusions and recommendations for possible further work are suggested in chapter 9.

Chapter 2

Mesh and Data Structure

2.1 Overview

In order to aid the reader's understanding of what is to follow, a brief overview is given in this section of the adaptive multi-material ALE algorithm, and the steps that are performed during a single time step. It is assumed that the ALE code user has successfully defined the initial geometry, computational mesh, material data and supplied a control file.

CORVUS performs a series of initialisation steps, which includes initialising all the element and node based variables, constructing connectivity arrays and the data structure required to support the slide algorithms.

A stable time step is then calculated at the start of each step, which also attempts to avoid cell volume collapse, element bow-ties and boomerangs. The sequence of operations for the rest of the time step can then be divided into three main phases: the Lagrangian step, mesh adaption step and the advection step.

The Lagrangian step can be further subdivided into (i) a pressure predictor phase, (ii) acceleration calculation and (iii) a corrector phase. The former estimates element pressures half way through the time step for use in the acceleration phase. The acceleration phase applies the conservation of momentum to determine the acceleration of all nodes, and this includes the influence of slide and other interface physics. The corrector step then updates all the element properties to allow for the cell volume change calculated during the momentum step. If multi-material cells are present, then the Lagrangian phase also updates the partial material components within the cells.

Once the Lagrangian step is completed, then the mesh adaption phase starts by inserting additional mesh lines in one logical mesh direction, if required, in each region. The mesh insertion is controlled by the application of a user defined aspect ratio error estimator. This option was added to CORVUS for high convergence problems typically involving the implosion of thin layers and as a precursor to the planned development of a combined ALE and AMR capability.

However, the main means of introducing mesh adaption is through arbitrary mesh movement. This process proceeds by first applying a series of constraints to determine which nodes are allowed to move, then calculates material or region weights for each node, applies boundary or slide line movement algorithms, and finally determines new internal node positions by applying these nodal weights and internal mesh movement algorithms on a region by region basis.

The advection step is then performed first by calculating exact overlap volume for all faces of all ALEing elements. New multi-material cells are then dynamically created whenever these overlap volumes indicate that a cell is accepting material either from a single material cell of a different material, or from a multi-material cell. The advection equation is then solved using a second order method for the single material cells, and a first order scheme for the multi-material cells to rezone all the state variables from the mesh at the end of the Lagrangian step to the new adapted mesh. The final step is then to remove any multi-material cells that are no longer required, as they only contain a single material.

2.2 Mesh Structure

CORVUS has been developed to run on unstructured multi-block meshes constructed from logically rectangular mesh blocks or regions, which can be arbitrarily connected. This mesh structure is very flexible, allowing users to mesh complicated geometries, and potentially offers greater computational efficiency than totally unstructured grids by enabling the use of algorithms which exploit the local logical structure of individual mesh blocks. However, the data structure has been kept as general and unstructured as possible to avoid any restrictions on unstructured mesh adaption in the future if this is shown to be required. This approach differs from most of the other published ALE algorithms which have tended to employ a logical mesh structure.

A staggered mesh data layout is used, as discussed in the introduction, with positions, velocities and accelerations at cell vertices or nodes, while all other variables are element centred. Both the node and element centred variables are stored as 1D arrays [32] or vectors in order of element or node number. The elements and nodes in each region being numbered sequentially on a row by row basis. The logical mesh dimensions (L_R, K_R) are stored along with pointers to the first and last element in each region or mesh block. This approach was initially taken to enable the code to be efficiently vectorised on the CRAY C98D at AWE. However, the code has been successfully ported to an IBM SP2 a Massively Parallel Processor (MPP) super computer and to Sun work stations at AWE. Although the released code is only serial running on one processor of the MPP at the moment, a parallel version of the code is currently under development.

2.3 Connectivity Arrays

The unstructured nature of the code requires a series of connectivity arrays to be defined at the start of the calculation and redefined if the mesh topology changes. Three types of 2D connectivity arrays are used providing, element to node, element to element, and node to node [30, 31]. The element to node connectivity defines the global node numbers for the four local nodes that form each quadrilateral element and is initially calculated during the mesh generation. It is then used in CORVUS to calculate the other two connectivity arrays. The calculation of the element to element connectivity arrays can be expensive if done naively.

A fast method has been developed for CORVUS by exploiting the multi-block mesh structure. The connectivity arrays are first initialised to a negative values to denote that they are unset. The algorithm then loops over the four element faces of each element. A test is then made to see if its value is already set (positive value), if it is not set then the six most likely element numbers are tested to see if they share nodes with the element in question across this cell face. These six element numbers are chosen assuming the neighbouring element is in the same mesh block in which case if the connectivity is being calculated for element i the six element numbers tested as potential neighbours are; i+1, i-1, $i+L_R$, $i-L_R$, $i+K_R$, $i-K_R$. If none of these is successful then a rigorous search is performed. Once the neighbouring element is found, the connectivity information is updated in both directions for the pair of elements.

The node to node connectivity can be generated directly from the element to node connectivity without a search. Both the element to element and node to node connectivities are set to zero if they point out of a free or reflecting boundary condition, or across a non-ALEing material interface.

2.4 Slide lines

The slide line algorithms in CORVUS treat material interfaces as two separate surfaces, one for each material, which can slide over each other. These algorithms will also treat both void opening and void closure. In order to implement these algorithms, a list is stored of all the global node numbers on each slide surface. This approach improves efficiency, as slide nodes can be addressed indirectly. Storage requirements are also reduced, as slide specific quantities are only stored at slide nodes. The slide surfaces are stored in pairs: the master surface first then the slave in each case. Pointers are also stored which point to the index of the first and last slide node in these lists for each of the slide surfaces. This enables operations to performed upon various sets of nodes relevant to the slide calculation such as: all slide nodes, master surfaces, slave surfaces or one user defined slide surface.

2.5 Disjoint nodes

It is possible to define merged Lagrangian interfaces with 1:1, 1:2, 1:3 and 1:4 ratios for the mesh spacing across the interface. In the latter three cases this introduces disjoint or hanging nodes. A special treatment is applied for these disjoint nodes, which requires a dedicated data structure. This is created during mesh generation and again takes the form of an unstructured list of the disjoint nodes and their immediate non-disjoint node neighbours. A further array is also created which gives the relative positions of the disjoint node between these two non-disjoint node neighbours.

2.6 Multi-material Cells

The volume fractions which define the relative volume of the ALEing materials in each multi-material cells are simply stored as mesh wide 2D arrays, as this simplifies the implementation of the multi-material advection. There are two ways to address multi-material cells in CORVUS. The first is simply through a list held of the current multi-material cells. The second is by the region number stored for all elements, which is positive for single material cells and negative for multi-material cells. A further mesh wide integer array is also available that points back from multi-material cell to their index in the multi-material cell list. A list is also maintained of single material cells in ALEing regions so that it is possible to loop separately over either multi-material or single material cells.

The state variables for each of the partial material components present in each multi-material cell are packed into a series of vectors, one for each state variable. The individual material components are addressed through a special type of connectivity array, which points from the list of multi-material cells to the position in each vector where the states of that material component are to be found, as illustrated in Fig. 2.1. This data structure was chosen to minimise storage and to link in naturally with the vector equation of state package CORVUS uses.

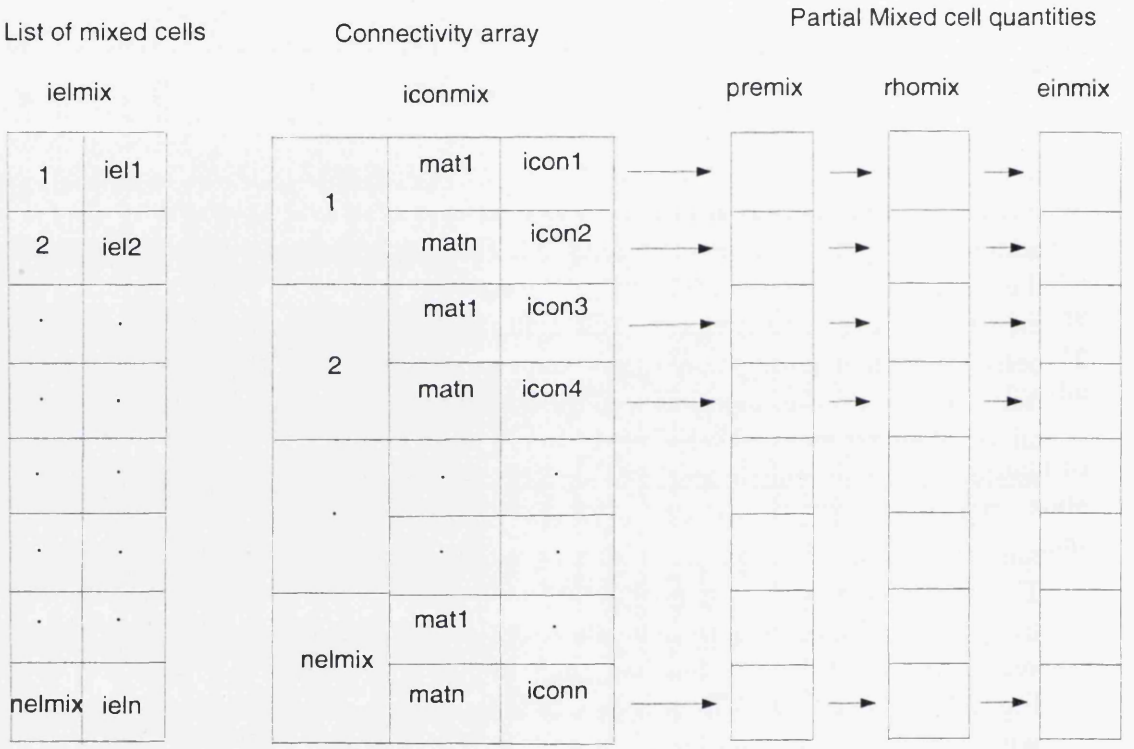


Figure 2.1: Data structure for multi-material cell components.

Chapter 3

Lagrangian Hydro Scheme

The adaptive multi-material ALE algorithm has been implemented within CORVUS, an existing 2D finite element Lagrangian hydrocode. The pure Lagrangian hydro scheme in CORVUS was developed at AWE by Whittle [29]. An overview will first be given of this Lagrangian scheme before moving on to consider the adaptive multi-material ALE algorithm and how it has been implemented in CORVUS.

3.1 Lagrangian Equations

In order to simplify the description of the Lagrangian scheme, material strength will initially be ignored and deferred to a later section. The system of partial differential equations to be solved are the conservation laws of mass, momentum and energy for inviscid fluid flow with a Lagrangian frame of reference, which can be written:

$$\frac{D\rho}{Dt} = 0 \tag{3.1}$$

$$\frac{D\rho\mathbf{u}}{Dt} = -\nabla p \tag{3.2}$$

$$\frac{D\rho\varepsilon}{Dt} = -p\,\nabla\cdot\mathbf{u} \tag{3.3}$$

where D/Dt is the Lagrangian derivative,

$$\frac{D\Phi}{Dt} = \frac{\partial\Phi}{\partial t} + \nabla\cdot(\Phi\mathbf{u}) \tag{3.4}$$

The momentum and internal energy equations may be expanded by chain differentiation, and using the mass equation rewritten:

$$\rho\frac{D\mathbf{u}}{Dt} = -\nabla p \tag{3.5}$$

$$\rho \frac{D\varepsilon}{Dt} = -p \nabla \cdot \mathbf{u} \tag{3.6}$$

These equations are only valid for differentiable compressible inviscid fluid flow, not shock hydrodynamics problems, which contain discontinuities. This difficulty is overcome by introducing an artificial viscosity term, as suggested by Von Neumann in the 1950's [11]. The momentum and energy equations then become,

$$\rho \frac{D\mathbf{u}}{Dt} = -\nabla(p+q) \tag{3.7}$$

$$\rho \frac{D\varepsilon}{Dt} = -(p+q) \nabla \cdot \mathbf{u} \tag{3.8}$$

The artificial viscosity term has two functions; it provides the irreversible shock heating that shock physics demands, and it stabilises second order numerical schemes, such as that used in CORVUS, for shock discontinuities. In practical terms this means the shock discontinuity will actually be smeared over 4-5 computational zones.

The system of coupled partial differential equations is closed, by introducing an equation of state for each material in the problem of the form,

$$p = p(\rho, \varepsilon) \tag{3.9}$$

3.2 Time step control

The numerical solution of any time-dependent problem requires the specification of a time increment, Δt, in order to march the solution forward in time. For explicit methods this value cannot be arbitrary, rather it must be less than some maximum value allowable for stability. For a Lagrangian formulation, Δt is limited by the Courant-Friedrichs-Lewy (CFL) criterion,

$$\Delta t = C\left(\frac{\Delta s}{c}\right) \quad ; \quad C \leq 1 \tag{3.10}$$

where Δs represents a measure of the minimum distance a signal has to propagate to cross a computational cell, c is the sound speed, and C is the Courant number. The CFL criterion says physically that the explicit timestep must be no greater than the time required for a sound wave to cross any cell in the mesh. This requires that we know the local sound speed in each cell, as well as the minimum distance across the cell. In 1D the latter distance is simply the cell width. However, in 2D and 3D Lagrangian codes it is too computationally expensive to rigorously obtain the true minimum distance, so some approximation must be made. In many 2D codes such

as DYNA2D [33] the square root of the area of each element is used. In CORVUS, the minimum of four perpendicular projections from the halfway along each side to the intersection with another face, is taken.

This is seen as a good compromise between rigorous stability and computational efficiency. However, it is worth noting that simpler distance measures can often lead to greater robustness in practice. Although stability may be violated, the time step will not drop to the same extent for a given mesh distortion. If this stability violation occurs outside the domain of real interest it probably will not degrade the solution. However, this is a risky strategy. The approach with CORVUS has been to go with the more rigorous and computationally expensive distance, described above, and rely upon the improved mesh quality that can be obtained with the ALE technique to maintain a high time step.

The local sound speeds are calculated within the equation of state routines. The Courant number (sometimes also known as the safety factor) is chosen to ensure the stability of the numerical scheme. The maximum allowable value for C for stability in explicit time-dependent finite difference calculations usually varies from 0.5 to 1.0 depending upon the actual application. The default safety factor, or Courant number in CORVUS, is taken to be $C = 0.5$, given that the actual minimum distance for a signal to propagate across a cell may be less than that calculated using the half side perpendicular projections.

The introduction of a shock viscosity, Q, requires that the above CFL criterion be modified [34]. An empirical correction is applied to the square of the sound speed from the equation of state,

$$c^2 = c_{\text{EOS}}^2 + \frac{2Q}{\rho} \tag{3.11}$$

where c_{EOS} is the bulk sound speed calculated from the equation of state routines.

The time step will vary from cycle to cycle or step to step, and hence is calculated at the start of each time step or cycle. Also, the sound speed will vary from cell to cell in any given cycle, thus the time step is taken over the entire spatial grid. A time step is therefore calculated for each cell of the mesh according to (3.10), where the value of the sound speed, c, is taken from (3.11). The minimum time step over the mesh is then compared with the previous time step multiplied by a growth factor, and a specified maximum allowable time step, and the minimum of these three values chosen as the stable time step. The growth factor is used to limit the amount by which the time step can increase from step to step.

In ALE calculations the time step is also constrained by an advection time step limit for all the advecting elements. This advection time step limit takes the same form as (3.10) but replaces the sound speed with the maximum velocity of the elements four nodes.

3.3 Artificial Viscosity Treatment

Von Neumann and Richtmeyer [11] first introduced the concept of an artificial viscosity, q, to enable the numerical calculation of fluid flow problems containing shocks. Since then, many papers have been written on the subject, and many test problems defined to investigate and compare different models. Basically, artificial viscosity allows the representation of a shock discontinuity, by smearing the shock front over several cells, so that the shock front can be propagated as a travelling wave. In other words, the discontinuous pressure jump across a shock is replaced by a rapid but continuous change. The artificial viscosity is added consistently to the pressure in the governing fluid flow equations and is designed to be non-zero only in those cells where the local velocity gradient is interpreted as representing the presence of a shock.

A linear combination of quadratic and linear viscosities is used in CORVUS, as recommended by Landshoff [12]. Landshoff was first to note that, whilst the original quadratic artificial viscosity of von Neumann works reasonably well, small oscillations still occurred after the shock, and linear viscosities used in isolation are overly diffusive. It is possible to obtain superior results to both techniques in isolation by employing a linear combination of both. Since then many papers have been presented on artificial viscosity. However, possibly the most significant improvement in the performance of these methods came with the introduction of monotonic artificial viscosity terms. One such method is the monotonic scalar artificial viscosity [35, 18, 17], devised by Randy Christensen [16], and extended for 2D meshes by Tipton [19], has been adapted for the unstructured grids used in CORVUS by the author.

In practical terms, a monotonic artificial viscosity of this type leads to greater accuracy, as it introduces the minimum of dissipation, like Riemann solvers: sharpening shock fronts, without introducing numerical ringing. It also introduces less artificial shock heating in regions of isotropic convergence, and where shocks initially form at material interfaces, rigid and reflecting boundary conditions. The first origins of the method can be traced back to a 1980 paper where Wilkins [14] noted the formal similarity between von Neumann's artificial viscosity and the Hugoniot relation for a steady state, planar shock. Wilkins showed that the linear Q coefficient of the von Neumann viscosity should be of order unity based on an analogy with the Reimann solvers used in Godunov methods. However, in practice a linear Q coefficient of 1.0 produces an unacceptable amount of numerical dissipation. In contrast, high order Godunov codes, based upon a more sophisticated use of the Hugoniot relations (Reimann solvers) do not exhibit this level of dissipation. In fact they introduce minimum dissipation. In 1986, Christensen [16] explained this apparent discrepancy. It is insufficient to simply set the linear viscosity coefficient to some value near unity to obtain a complete correspondence between artificial viscosity and Reimann solvers. It is also necessary to limit the velocity jump Δu^n

used to calculate the artificial viscosity according to a monotonic principle. Christensen developed a limiting principle for Δu^n which is trivial to add to the normal von Neumann artificial viscosity.

Figure 3.1 illustrates how the velocity difference Δu used in monotonic Q for smooth and steep velocity gradients. In contrast a standard von Neumann artificial viscosity takes Δu to be the simple velocity difference between the right and left edge of the zone.

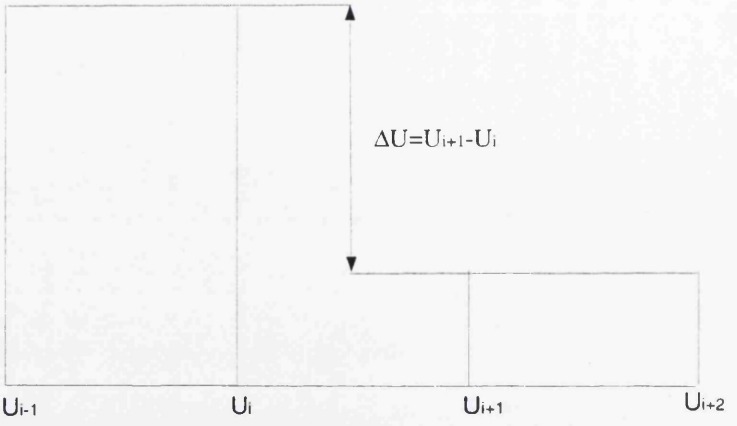


Figure 3.1: Δu used by monotonic artificial viscosity for smooth and steep velocity gradients.

In contrast, in the monotonic q formulation, the nodal velocities are fitted with a piecewise linear distribution, whose slopes are adjusted according to the monotonic principal. The monotonic principle will tend to select the flatter of the two velocity slopes touching each node. This velocity distribution is then used to extrapolate velocities from the right and left edge to the zone centre. The difference between the right and left extrapolated values is then taken as Δu and used to calculate the artificial viscosity. It is also clear from Figure 3.1 that the monotonic Δu has the desirable property that it reduces to the von Neumann Δu for steep gradients and tends to zero for smooth velocity fields. In fact if the velocity field is exactly linear, then the monotonic q will vanish, whilst the conventional von Neumann q could be quite large. In this sense, then, the monotonic q is a discontinuity detector that will only turn on when there is a discontinuity in the velocity field.

It is the latter character that allows the use of order unity linear q coefficients without introducing excessive damping in smooth flow regions. This is in turn important, as it is the order unity linear q coefficient that enables sharp narrow shock profiles of the order of one to two zones thick to be obtained without suffering numerical ringing. The results obtained using the monotonic artificial viscosity are in fact quite comparable with those obtained with Reimann solvers.

The monotonic artificial viscosity calculation is best explained by considering 1D first, then moving on to the extension to 2D in CORVUS. The monotonic Δu is calculated by considering the velocity gradient in the zone of interest and in zones on either side of it.

$$\begin{aligned}
\left(\frac{\partial u}{\partial x}\right)_L &= \frac{u_i - u_{i-1}}{x_i - x_{i-1}} \\
\left(\frac{\partial u}{\partial x}\right)_C &= \frac{u_{i+1} - u_i}{x_{i+1} - x_i} \\
\left(\frac{\partial u}{\partial x}\right)_R &= \frac{u_{i+2} - u_{i+1}}{x_{i+2} - x_{i+1}}
\end{aligned} \tag{3.12}$$

Next we define the left and right ratios of the velocity gradient.

$$\begin{aligned}
R_L &= \frac{\left(\frac{\partial u}{\partial x}\right)_L}{\left(\frac{\partial u}{\partial x}\right)_C} \\
R_R &= \frac{\left(\frac{\partial u}{\partial x}\right)_R}{\left(\frac{\partial u}{\partial x}\right)_C}
\end{aligned} \tag{3.13}$$

Then define ϕ as:

$$\phi = max(0, min(\frac{1}{2}(R_L + R_R), 2R_L, 2R_R, 1)) \tag{3.14}$$

Finally the monotonic q for the central zone zone C is given by:

$$q = C_Q \rho_C |\Delta u|^2 (1 - \phi^2) + C_L \rho_C C_{SC} |\Delta u| (1 - \phi) \tag{3.15}$$

where $\Delta u = u_{i+1} - u_i$ is the usual zone centred velocity difference, C_Q is the quadratic viscosity coefficient, C_L the linear viscosity coefficient and C_{SC} the sound speed of element C.

C_L should always be set to $\frac{1}{2}$, which is its default value. C_Q depends on the Hugoniot for the material. Christensen [16] showed that it should be equal to half the slope of the Hugoniot curve for shock velocity plotted against particle velocity. Since for most metals this is approximately 1.5, the default value in CORVUS for C_Q is taken as $\frac{3}{4}$.

It is not obvious what the best approach is for extending the scalar monotonic from 1D to 2D. In fact in general there are a number of difficulties associated with extending artificial viscosity methods to multi-dimensions, in addition to the concerns already expressed for 1D. The magnitude of the viscosity should depend on the strain rate perpendicular to the direction of shock propagation, and should only act in that direction [14] for example. The optimum approach may be to employ a tensor rather than a scalar artificial viscosity [13], and make this monotonic in a 2D sense. However, an operator splitting approach suggested by Tipton [19] has been found to be effective provided the cell aspect ratio is not too far from one to one. Although Tipton's original idea was intended for a logical grid code, it was not difficult to recast it for the indirectly addressed unstructured grids used in CORVUS.

The 2D scalar monotonic artificial viscosity in CORVUS defines four separate 1D-like q's for each element; q_t (top), q_b (bottom), q_l (left) and q_r (right), as shown in Fig. 3.2; the first two being associated with compressions in one logical mesh direction, the last two with that orthogonal to the other two. In order to calculate the four q's, appropriate directions or mesh legs must be defined. Following the notation given in Fig. 3.2 these mesh legs are defined by,

$$\begin{aligned} Lhor_r &= -(z_2 + z_3 - z_4 - z_1) \\ Lhor_z &= (r_2 + r_3 - r_4 - r_1) \\ Lver_r &= (z_1 + z_2 - z_3 - z_4) \\ Lver_z &= -(r_1 + r_2 - r_3 - r_4) \end{aligned} \tag{3.16}$$

Distance measures are also required for each of these directions in each cell and are given by,

$$\begin{aligned} \Delta x_{tb} &= \frac{area}{|\bar{Lhor}|} \\ \Delta x_{lr} &= \frac{area}{|\bar{Lver}|} \end{aligned} \tag{3.17}$$

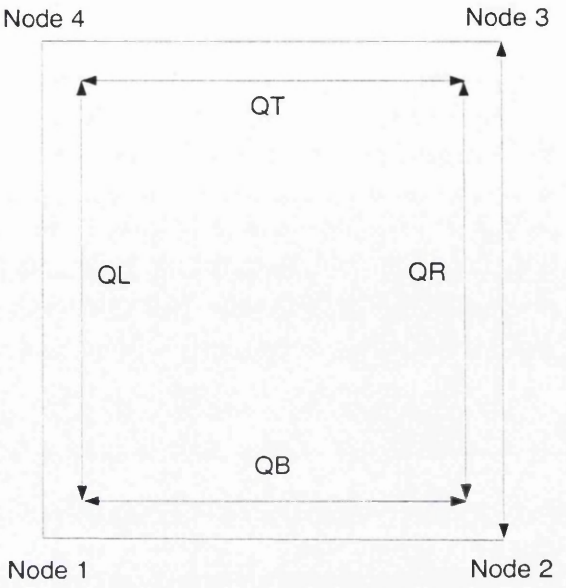


Figure 3.2: Notation for monotonic artificial viscosity.

where area is the area of the zone in question. The projected velocity gradient for each of the four sides of each element is then given by,

$$\frac{\Delta u_b}{\Delta x_b} = \frac{\bar{Lhor} \cdot (\bar{u_1} - \bar{u_4})}{area}$$

$$\frac{\Delta u_l}{\Delta x_l} = \frac{\bar{Lver} \cdot (\bar{u_3} - \bar{u_4})}{area} \tag{3.18}$$

$$\frac{\Delta u_t}{\Delta x_t} = \frac{\bar{Lhor} \cdot (\bar{u_2} - \bar{u_3})}{area} \tag{3.19}$$

$$\frac{\Delta u_r}{\Delta x_r} = \frac{\bar{Lver} \cdot (\bar{u_2} - \bar{u_1})}{area} \tag{3.20}$$

Christensen's monotonic limit is then applied to each of the four sides of each element, exactly as outlined above for the strictly 1D case.

The method is implemented in CORVUS through two main loops. The first loops over all elements and uses the element to node connectivity to calculate the two distance measures for each element, unit vectors for the two mesh legs and the unlimited velocity gradients for each of the four edges of each element. A second loop then employs the element to element connectivity to calculate the left and right velocity slope ratios (3.13), which are then used to calculate limiter functions (3.14)

for each edge. The velocity slope ratio is set to zero across free boundaries and material interfaces, but set to one at reflecting boundaries. Although Tipton and Christensen's ideas were originally intended for logical mesh based codes, the above procedure extends the method naturally to an unstructured grid of quadrilaterals: the only requirements being element to node and element to element connectivity information. This allows the two logical mesh directions to be identified locally for each element. The velocity slopes can then be limited, provided neighbours can be identified to establish the velocity slope ratios. In CORVUS each mesh block is locally logically rectangular, so within blocks this will always be the case. If a suitable neighbouring velocity slope cannot be identified, then that edge remains unlimited as at material interfaces.

Once the velocity slope ratios have been calculated, then any edge velocity slope which indicates expansion is set to zero,

$$If \ \frac{\Delta u}{\Delta x} > 0 \ \ then \ \ \frac{\Delta u}{\Delta x} = 0 \tag{3.21}$$

The four edge q's can then each be calculated from (3.15). An average element centred q is then obtained by averaging the two edge q's for each logical mesh direction and then summing the averages obtained for the two logical mesh directions,

$$q_{scalar} = \frac{1}{2}(q_b + q_t + q_l + q_r) \tag{3.22}$$

In addition to setting the edge velocity slopes to zero if they are greater than zero, it has also been found to be important to set the final scalar q to zero if the element volume is expanding. This scalar monotonic element centred q or artificial viscosity is then used consistently throughout the time step in solving both the momentum and internal energy equations.

3.4 Hourglass Filters

In two dimensions, each fluid parcel has exactly six hydrodynamic degrees of freedom. This is seen by dividing its motion into two categories: translational modes and modes leading to deformation. There are clearly two translational degrees of freedom, and the number of modes leading to deformation is equal to the number of degrees of freedom of the total strain rate tensor. By the symmetry of the tensor and assuming axial symmetry, then the total strain rate tensor has exactly four independent degrees of freedom. These include a volume changing mode, rotation, shear and a final mode, where the volume change of the fluid parcel in one direction due to compression is exactly matched by the volume change for expansion in the other, leading to no overall volume change.

However, although the fluid parcel should only have six degrees of freedom, a quadrilateral element or zone has a total of eight degrees of freedom (2 velocity

components per node x 4 nodes). This leads to the problem that the computational zones or elements in CORVUS are capable of reproducing the six physical degrees of freedom plus two non-physical degrees of freedom. These modes are called hourglass, or zero energy modes. The former term comes from the shape of adjacent zones when hourglass instabilities are present and latter term from the fact that no internal energy change occurs because the element volume is unchanged.

If the numerical schemes used are completely faithful to the partial differential equations they are used to solve, then unphysical modes would never be activated. However, in practice, small errors are present, which are sufficient to seed these instabilities. If nothing is done to suppress these errors in Lagrangian calculations, then the instabilities will grow, to the point where they can destroy the solution or stop calculations from running to completion.

Hourglass modes are less of a concern for Eulerian simulations where the mesh is continually remapped and the cells remain orthogonal with a 1:1 aspect ratio. However, hourglassing can still sometimes be seen in these Eulerian simulations imprinting on the velocity field. In an ALE code these modes must be suppressed, as parts of a problem may be Lagrangian or very close to Lagrangian.

There have been many techniques published for suppressing or filtering hourglass modes out of the solution. One of the first successful techniques was the anti-rotational artificial viscosity developed by Seymour Sack and George Maenchen for their code TENSOR [13]. Since then, most of the published techniques have been hourglass filters [33, 36, 37, 38]. These methods have two common ingredients: the first part defining the mode shape, the second defining the resisting force. The potential benefit of filters over modified viscosities is that they should act only to suppress the undesirable modes, leaving the rest of the solution unchanged.

Three hourglass filters have been implemented in CORVUS, including Hallquist's DYNA2D filter [33], Flanagan and Belytscko [36] and Hancocks's filter used in PISCES [39]. In practice the author has found little to distinguish between the former two well published forms. The PISCES form uses the same mode shape as the DYNA filter, but a different lengthscale, which appears to be more effective, as it is free from a dependence on the material's sound speed.

3.5 Time Discretization

The time differencing is performed using a predictor-corrector solution method. The predictor corrector method performs a forward Euler half time step with first order accuracy to calculate a half time step pressure (predictor step). The half time step pressure is then used to advance the solution with second order accuracy to the end of the time step (corrector step). It is this centering of the pressure at the half step that makes the final values second order accurate in time.

At the start of each time step, a stable time step is determined and an artificial

viscosity calculated for each cell. The time step control and the form of artificial viscosity used in the scheme has already been discussed. The predictor step then uses the velocities at the start of each time step to estimate the position of cell vertices or nodes at the half time step.

$$\mathbf{x}^{n+\frac{1}{2}} = \mathbf{x}^n + \frac{1}{2}\Delta t\ \mathbf{u}^n \qquad \forall \text{nodes} \tag{3.23}$$

A new volume is then calculated for each element and used to obtain half step densities and internal energies.

$$V^{n+\frac{1}{2}} = V(\mathbf{x}^{n+\frac{1}{2}}) \qquad \forall \text{cells} \tag{3.24}$$

$$\rho^{n+\frac{1}{2}} = \frac{M^e}{V^{n+\frac{1}{2}}} \tag{3.25}$$

$$\varepsilon^{n+\frac{1}{2}} = \varepsilon^n + \frac{1}{2}\Delta t \frac{(p^n + q^n)}{M^e} \nabla \cdot \mathbf{u}^n \tag{3.26}$$

The half step pressures are then obtained from the equation of state,

$$p^{n+\frac{1}{2}} = p(\varepsilon^{n+\frac{1}{2}}, \rho^{n+\frac{1}{2}}) \tag{3.27}$$

The velocities at the end of the time step can then be obtained with second order accuracy from the momentum equation,

$$\mathbf{u}^{n+\frac{1}{2}} = \mathbf{u}^n - \frac{\Delta t}{M_{nodel}} \nabla (p^{n+\frac{1}{2}} + q^n) \qquad \forall \text{nodes} \tag{3.28}$$

The average velocity over the time step is then used to calculate the positions of the nodes at the end of the time step,

$$\mathbf{x}^{n+1} = \mathbf{x}^n + \Delta t\ \bar{\mathbf{u}} \qquad \forall \text{nodes} \tag{3.29}$$

where,

$$\bar{\mathbf{u}} = \frac{1}{2}(\mathbf{u}^n + \mathbf{u}^{n+1}) \tag{3.30}$$

This enables the volume of the elements at the end of the time step to be calculated and used to update the densities, internal energies and pressures with second order accuracy.

$$V^{n+1} = V(\mathbf{x}^{n+1}) \qquad \forall \text{cells} \tag{3.31}$$

$$\rho^{n+1} = \frac{M^e}{V^{n+1}} \tag{3.32}$$

$$\varepsilon^{n+1} = \varepsilon^n + \Delta t \frac{(p^{n+\frac{1}{2}} + q^n)}{M^e} \nabla \cdot \bar{\mathbf{u}} \tag{3.33}$$

$$p^{n+1} = p(\varepsilon^{n+1}, \rho^{n+1}) \tag{3.34}$$

3.6 Spatial Discretization

The spatial discretization in CORVUS employs explicitly integrated bilinear isoparametric finite elements. This approach was selected in preference to alternative discretization methods, such as finite volume and contour integration methods, due to the arbitrary choices required by such methods. There are, for example, many different differencing strategies that can be employed for finite volume methods, and many different ways to define the contours used for contour integration methods. However, once the order of the shape functions have been defined then a fairly unique finite element discretization follows. Bilinear elements are used for all the node centred variables such as acceleration, velocity and position, whilst a piecewise constant representation is used for all the element centred quantities.

3.6.1 Isoparametric Mapping

The isoparametric mapping is constructed by adopting a natural coordinate system (ξ,η) for an element, such that the element has sides $\xi = \pm 1$ and $\eta = \pm 1$.

The four bilinear shape functions can then be defined as,

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

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

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

$$N_4 = \frac{1}{4}(1-\xi)(1+\eta) \tag{3.35}$$

3.6.2 The Equation of Conservation of Momentum

The acceleration at any point in an element e, can now be expressed in terms of the accelerations at its nodes $\dot{u}_k$ and their assosciated shape functions N_k,

$$\dot{\mathbf{u}} = \sum_k \dot{u}_k N_k \tag{3.36}$$

The momentum equation in axisymmetric geometry (3.2) can then be rewritten in weak form with the weighting function W_j as,

$$\rho_e \int_\Omega \sum_k \dot{u}_k N_k W_j d\Omega = -\int_\Omega \frac{\partial p_e}{\partial r_i} W_j d\Omega \tag{3.37}$$

Note that in axisymmetric geometry

$$d\Omega = rdrdz \tag{3.38}$$

and Petrov-Galerkin weighting must be used,

$$W_j = \frac{1}{r}N_j \tag{3.39}$$

Note that ρ_e and p_e in (3.37) are constant across each element, which leads to a problem in obtaining partial derivatives of p. However, appealing to Green's theorem, the right hand side can be rewritten as,

$$\int_\Omega \frac{\partial p_e}{\partial r_i} W_j d\Omega = p_e \int_{-1}^{1}\int \frac{\partial N_j}{\partial r_i} detJ d\xi d\eta - \left[\int_\Gamma \bar{p} N_j n d\Gamma\right] \tag{3.40}$$

where n is a unit vector which is normal to Γ the boundary of the element. The term in square brackets can be ignored except along an applied pressure boundary condition, since when evaluated along the common sides of two neighbouring elements, the terms will be equal in magnitude, but opposite sign, and so will cancel.

The left hand side of (3.37) is simplified by "mass-lumping";

$$\int_\Omega \sum_k \dot{u}_k N_k W_j d\Omega = \dot{u}_j \int_{-1}^{1}\int N_j detJ d\xi d\eta \tag{3.41}$$

Hence (3.37) can be rewritten,

$$\rho_e \dot{u}_j \int_{-1}^{1}\int N_j detJ d\xi d\eta = p_e \int_{-1}^{1}\int \frac{\partial N_j}{\partial r_i} detJ d\xi d\eta \tag{3.42}$$

This is exactly the same as for the Cartesian case. It is not obvious why Petrov-Galerkin weighting must be used, but Galerkin weighting $W_j = N_j$ leads to unacceptable axial behaviour, whilst Petrov-Galerkin weighting forces a spherical problem to remain spherical. This approach is analogous to the use of area weighted finite difference methods. If a Galerkin weighting was used then the factor r in the differential volume places too little a weight (it is actually zero in the limit) on the nodes along the z axis [18].

The solutions to the integrals on the left and right of 3.42 are presented in Appendix A. However, it is also worth considering what these terms mean. The right hand term,

$$\rho_e \int\int N_j detJ d\xi d\eta \tag{3.43}$$

is the mass contribution from the element e to its local node j. The best way to think of this is that each node has some volume associated with it which intersects each

of its adjacent elements; the mass contribution from each element being defined in terms of the element density and donated volume.

The term,

$$p_e \int \int \frac{\partial N_j}{\partial r_i} detJ d\xi d\eta \tag{3.44}$$

is the force in the r_i direction acting on local node j due to the pressure in element e.

The numerical method in CORVUS is implemented as two steps: a scatter operation which consists of an outer loop over the four local nodes associated with each element, and an inner loop over all the elements in the problem. This scatters each element's force and mass contributions out to its four local nodes. The second step, the gather operation, then sums the force and mass contributions for each node and then calculates a new nodal acceleration for the time step.

3.6.3 The Equation of Conservation of Specific Internal Energy

Ignoring source terms, the energy equation can be written,

$$\rho \dot{\varepsilon} = -p \nabla \cdot \mathbf{u} \tag{3.45}$$

A "weak" form of this equation can be created as before, with an arbitrary weight function W, and expanding the $\nabla \cdot \mathbf{u}$ term into axisymmetric components,

$$\int_\Omega \rho_e \dot{\varepsilon} d\Omega = \int_\Omega -p_e \left[\left(\frac{\partial u}{\partial z} + \frac{\partial v}{\partial r} \right) + \left(u \frac{\partial r}{\partial z} + v \frac{\partial r}{\partial r} \right) \right] W d\Omega \tag{3.46}$$

Choosing W=1, noting ρ_e and p_e are constant within each element, and putting $u = \sum_k u_k N_k$ and $v = \sum_k v_k N_k$,

$$\rho_e \dot{\varepsilon} \int_\Omega d\Omega = -p_e \int_\Omega \left(u_j \frac{\partial N_j}{\partial z} + v_j \frac{\partial N_j}{\partial r} \right) + \frac{v_j N_j}{r} d\Omega \tag{3.47}$$

Changing to the natural coordinates, and observing that

$$\rho_e \int_\Omega d\Omega = M_e \tag{3.48}$$

$$\dot{\varepsilon}_e = \frac{-p_e}{M_e} \left[\int \int \left(u_j \frac{\partial N_j}{\partial z} + v_j \frac{\partial N_j}{\partial r} \right) r_k N_k detJ d\xi d\eta + \int \int v_k N_k detJ d\xi d\eta \right] \tag{3.49}$$

The first integral on the left hand side is extremely complex, involving some 32 terms, (note the implied summation), but this can be reduced by a similar process to mass lumping,

$$\sum_k \int u_j \frac{\partial N_j}{\partial z} r_k N_k d\phi \approx \sum_j \int u_j r_j \frac{\partial N_j}{\partial z} d\phi \tag{3.50}$$

$$...etc$$

reducing 3.49 to a simple explicit expression for the rate of change of specific internal energy for the element e:

$$\dot{\varepsilon}_e = \frac{-p_e}{M_e} \sum_{j=1}^{4} \left[u_j r_j \left(\int \frac{\partial N_j}{\partial z} \right) - v_j r_j \left(\int \frac{\partial N_j}{\partial r} \right) + v_j \left(\int N_j \right) \right] \tag{3.51}$$

where

$$\left(\int \Psi \right) = \int_{-1}^{1} \int_{-1}^{1} \Psi det J d\xi d\eta \tag{3.52}$$

This differencing was used with reasonable success for several years until it was determined by the author that the differencing of $\nabla \cdot \mathbf{u}$ was the cause of small axial symmetry errors observed in some axisymmetric problems. A typical example of this problem is given in Fig. 3.3 which shows a blow up of the centre of a high convergence spherically symmetric flow problem. The fundamental problem with (3.51) is that the nodal velocities on the axis appear in isolation, with a radial weighting, so they have no influence on $\nabla \cdot \mathbf{u}$. The cell volumes are however accurately calculated, and since all we are really trying to do is calculate PdV, the cell volume change can be used instead to update the cell internal energy. This significantly improves the axial symmetry, as shown in Fig. 3.4.

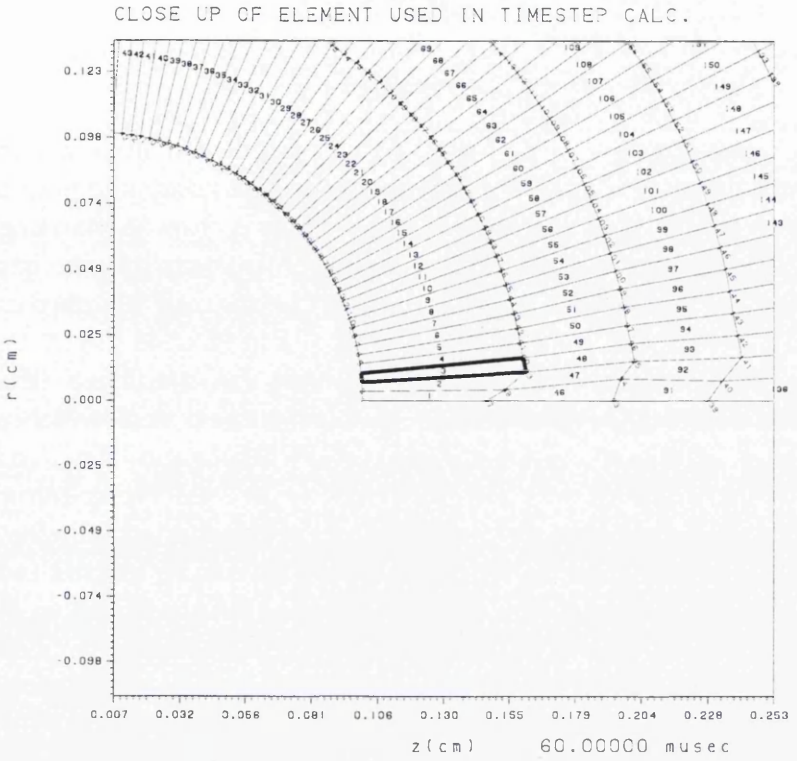


Figure 3.3: Axial symmetry error.

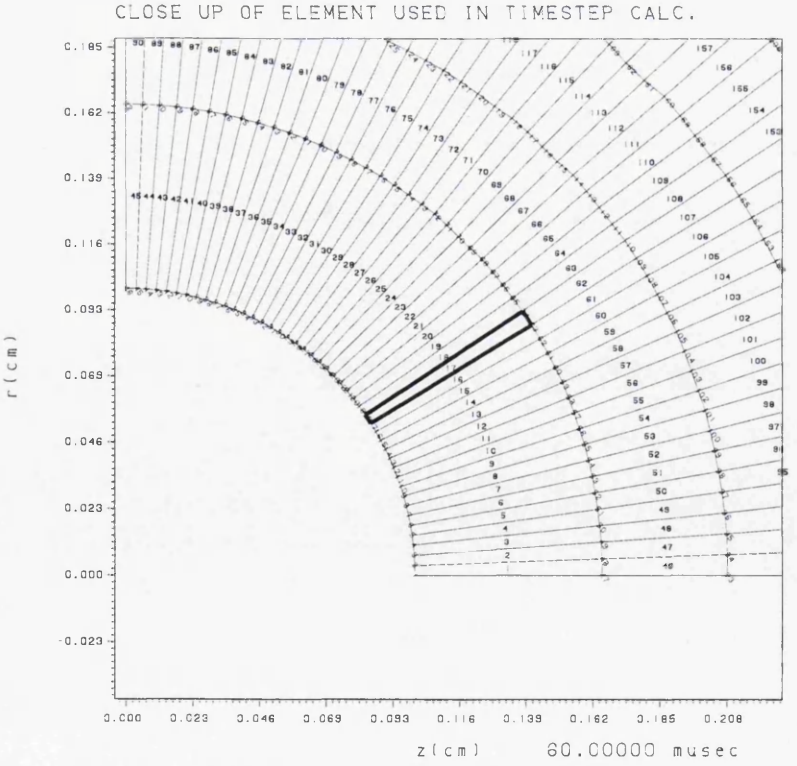


Figure 3.4: Improved axial symmetry obtained when cell volume change is used for the internal energy update.

3.7 Lagrangian test problems

3.7.1 Sod's shock tube problem

Sod's shock tube problem [40], although simple in definition, addresses the fundamental ability of a code to model shocks, contact discontinuities and rarefaction waves. It has an analytical solution [41], and many different numerical solutions have been published for comparison. The problem consists of a rigid walled shock tube containing two gases which are initially at rest, and separated by a diaphragm. The initial conditions for these gases are given in Fig. 3.5. The lower pressure gas is termed the test gas, and the higher pressure gas, the driver. The diaphragm is burst at the start of the problem and is assumed to have no further influence on the simulation. A shock wave then forms as the diaphragm bursts and moves to the right, whilst a rarefaction wave propagates to the left. The solution prior to either of these waves reaching the end walls can be divided into five regions, as illustrated in Fig. 3.6. Regions 1 and 5 are the remains of the driver and test sections that have not as yet been reached by the shock and rarefaction waves; region 2 corresponds to the states within the rarefaction fan; region 3 is a constant state, bounded by the tail of the rarefaction wave and the contact discontinuity; and region 4 is the constant state bounded by the shock front and the contact discontinuity.

A 1 by 100 uniformly spaced computational mesh has been used for consistency with the calculations in [40]. Normal reflecting boundary conditions were used for all the external boundaries, and ideal gas equations of state were used for both gases with the ratio of specific heats $\gamma = 1.4$ in both. Two calculations were performed, the first with bulk artificial viscosity with a quadratic coefficient of 1.0 and a linear coefficient of 0.1, and the second with monotonic artificial viscosity with a quadratic coefficient of 0.75 and a linear coefficient of 0.5.

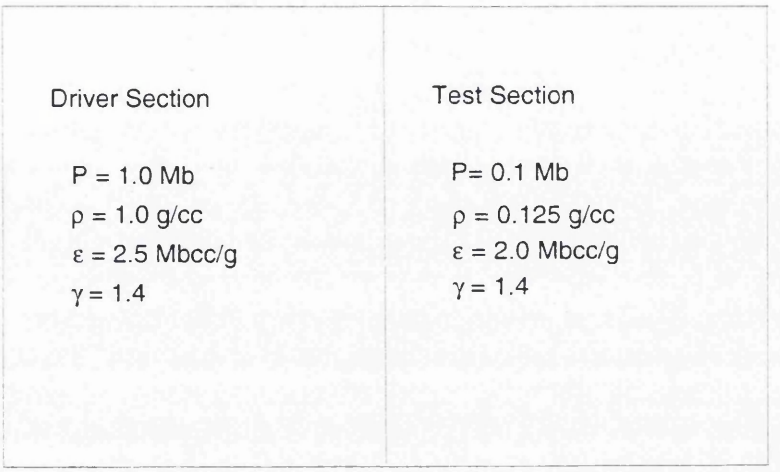


Figure 3.5: Initial conditions for Sod's shock tube problem.

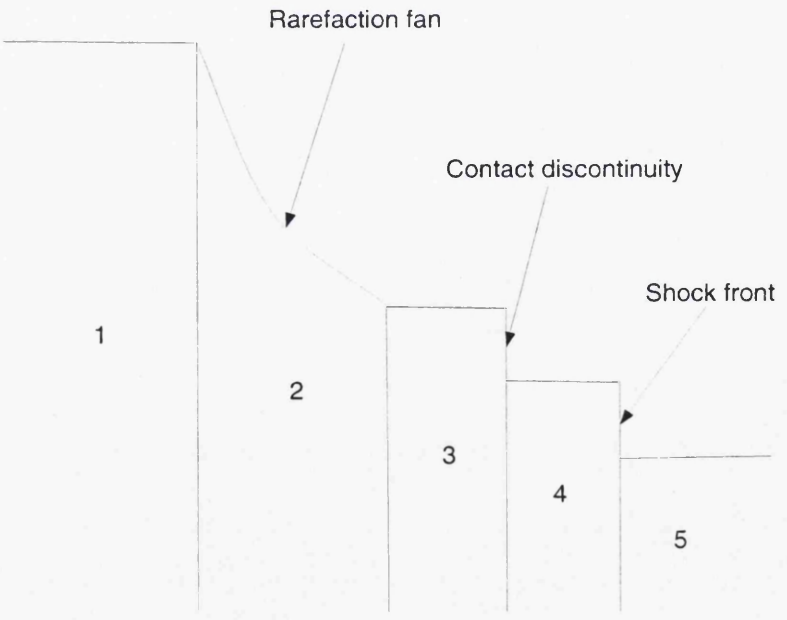


Figure 3.6: Sod's shock tube problem.

Numerical Results

The solutions obtained with the CORVUS Lagrangian hydro scheme are given for the two artificial viscosities in Fig. 3.7 and 3.8. In both cases the numerical solution is plotted as a dotted line, and the analytical solution as a solid line. The bulk artificial viscosity solution stands up quite well in comparison, with the results in [40]. The degree of rounding at the head and tail of the rarefaction fan is certainly comparable to results presented for the other second order accurate schemes. The shock front is captured over 4-5 zones and the contact discontinuity is perfectly captured as the mesh is moving Lagrangian. However, there is some ringing behind the shock front and a large disturbance is observed at the initial site of the diaphragm. The former is related to the artificial viscosity used, and if the coefficients are increased for both the linear and quadratic viscosity terms, then the oscillations will be reduced, at the expense of increasing the shock smearing. However, the larger disturbance at the diaphragm site is a wall heating error, which manifests itself here as a start up error. Essentially, too much artificial shock heating is introduced at the start of the calculation because it takes a number of time steps for the shock front to be smeared out over 4-5 zones. This error could probably be reduced by suppressing the time step until the shock front has spread out to this degree.

The solution presented in Fig. 3.8 clearly show the benefits of adopting a monotonic artificial viscosity. The oscillations behind the shock have been completely removed without increasing the shock smearing: in fact the shock front is actually a little sharper, now rising over 3-4 zones. The rarefaction is, as expected, comparable with the bulk viscosity solution, as artificial viscosity is not applied in expansion. The start up error at the diaphragm site has also been reduced a little, probably because the shock front is sharper.

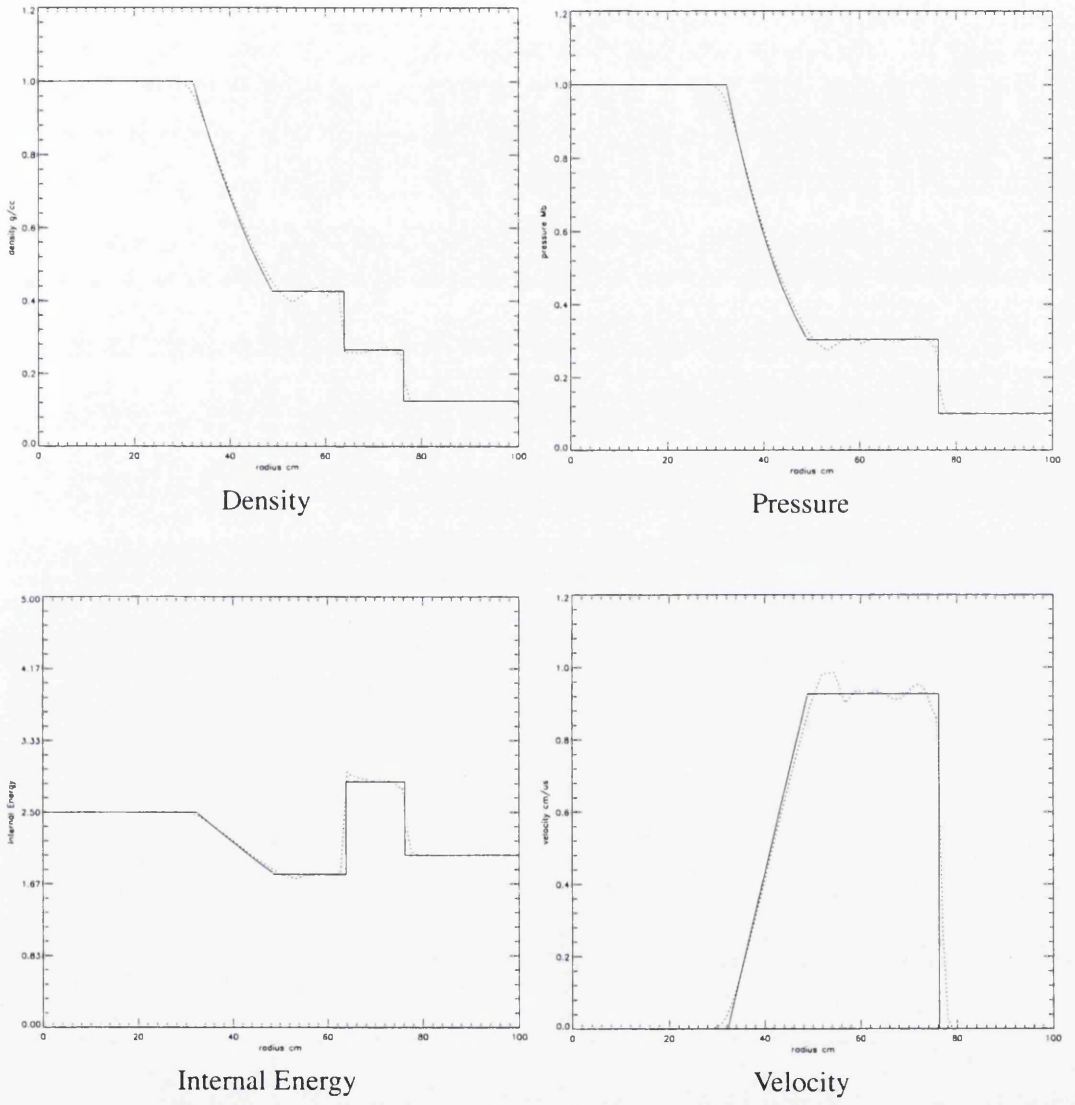


Figure 3.7: Sod's shock tube problem calculated with bulk artificial viscosity at 15.0 μs.

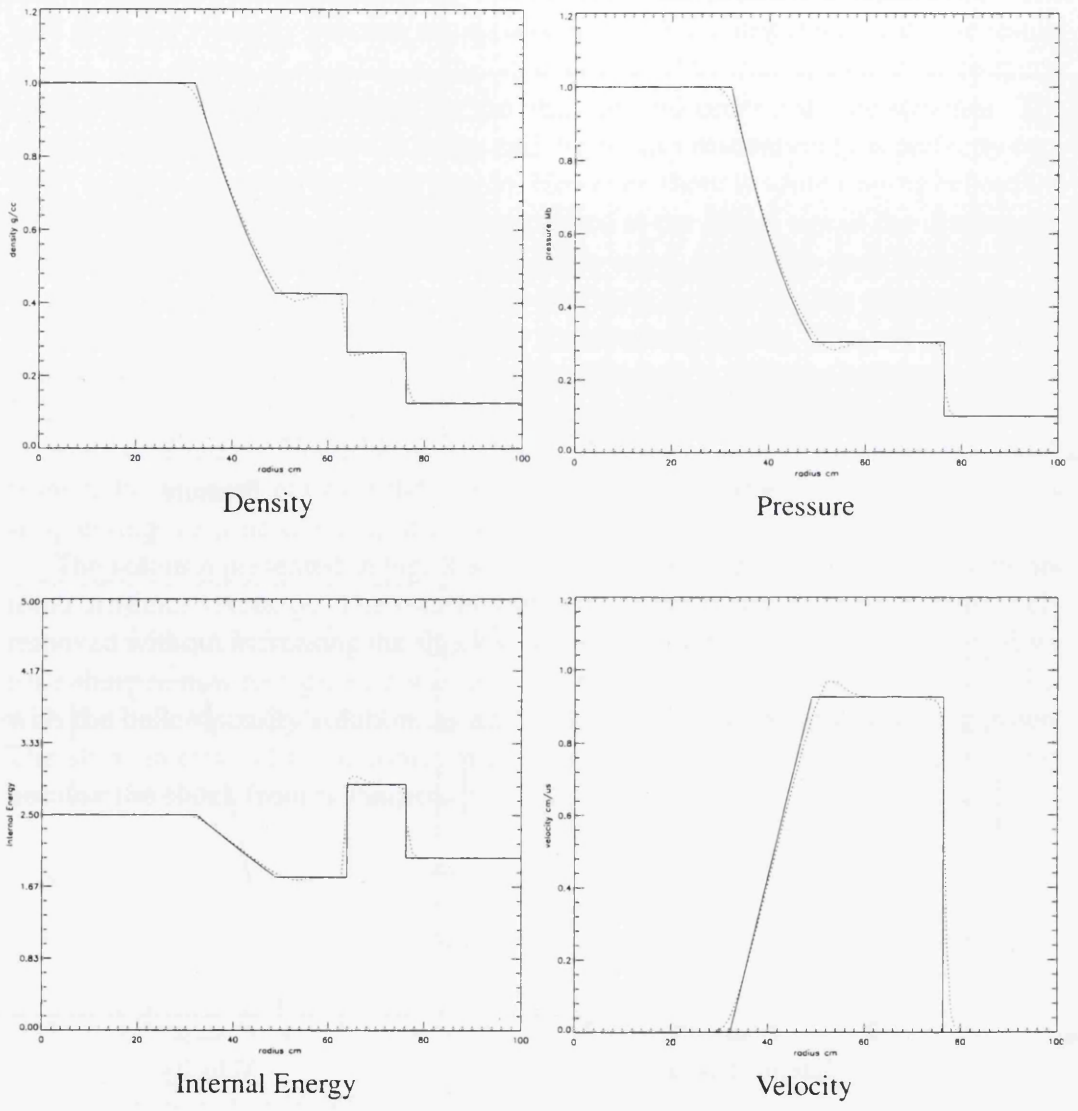


Figure 3.8: Sod's shock tube problem calculated with monotonic artificial viscosity at 15.0 μs.

3.7.2 Noh's problem

In 1987 Noh proposed three test problems in a paper on artificial viscosity errors [15]. The three problems all employed an ideal gas with $\gamma = \frac{5}{3}$ and the same initial conditions for all three problems: the internal energy is 0.0 Mb cc, the density $1.0gcm^{-3}$ and the gas has a uniform velocity of -1.0 $cm\mu s^{-1}$ in the radial direction. The only difference between the problems is that the first is planar (n=1), the second is cylindrical (n=2) and the third is spherical (n=3). A shock wave is generated at the origin in all the problems. The analytical solution for the three problems then defines the shock velocity to be $\frac{1}{3}cm\mu s^{-1}$, the internal energy behind the shock $\frac{1}{2}Mbcc$ and the density behind the shock $4^n gcm^{-3}$.

The three problems provide a test for two types of artificial viscosity error, which both manifest themselves as artificial shock heating. The first type of error is the wall heating error discussed above and is observed as the shock forms at the centre for all three variants of the problem. The second type of overheating is only relevant to the cylindrical and spherical problems, and occurs where an artificial viscosity is unable to distinguish between compression, due to a shock and isotropic compression, due to flow convergence. Only the spherical case will be presented, as it represents the worse case for both types of error. It also provides a test of how well spherical symmetry is maintained.

The initial mesh used for Noh's problem contains 100 radial elements and 3 degree angular zoning as shown in Fig. 3.9. The bulk artificial viscosity calculation was again performed with a quadratic coefficient of 1.0 and a linear coefficient of 0.1. The monotonic artificial viscosity calculation again used a quadratic coefficient of 0.75 and a linear coefficient of 0.5.

Numerical Results

Spherical symmetry is well preserved for both forms of artificial viscosity. Typical mesh and density contour plots are given at $0.6\mu s$ in Fig. 3.10 and Fig. 3.11 for the monotonic artificial viscosity form, whilst Fig. 3.12 shows density profiles as a function of radius for both of artificial viscosities. The solid line denotes the monotonic artificial viscosity form and the dotted line the bulk artificial viscosity. The analytical solution for the density behind the shock is $64.0gcm^{-3}$, which is not reached by either calculation at this mesh resolution. The calculated density in both cases is not flat, as demanded by the analytical solution, but falls towards the origin.

This drop in density and the corresponding increase in internal energy are the result of the spurious shock heating, discussed above. Both forms of artificial viscosity are introducing more shock heating than required where the shock is initially formed, and are introducing shock heating in some parts of the problem where shocks are not even present. The latter occurs because neither type of artificial viscosity can truly distinguish between isotropic compression, due to flow convergence

and, compression, due to the presence of a shock. The monotonic form of artificial viscosity does reduce this type of error, which explains the higher peak density obtained. However, both types of artificial viscosity introduce wall heating errors near the origin. Once the problem is established, the shock front will be smeared over a constant number of zones. However the impulsive problem start means that the shock front will initially rise over a single zone, resulting in excessive shock heating. A similar error is always introduced using artificial viscosity methods when ever a shock crosses a material interface. Overall the results obtained are superior with the monotonic artificial viscosity. The shock front is sharper and there is no sign of ringing behind the shock front. The peak density behind the shock is also significantly closer to the analytical solution. However, the density is lower close to the origin suggesting the wall heating error is larger. This may be because the flux limiter used for the monotonic form is less effective next to the central boundary condition, and the linear viscosity coefficient used is higher than for the bulk form.

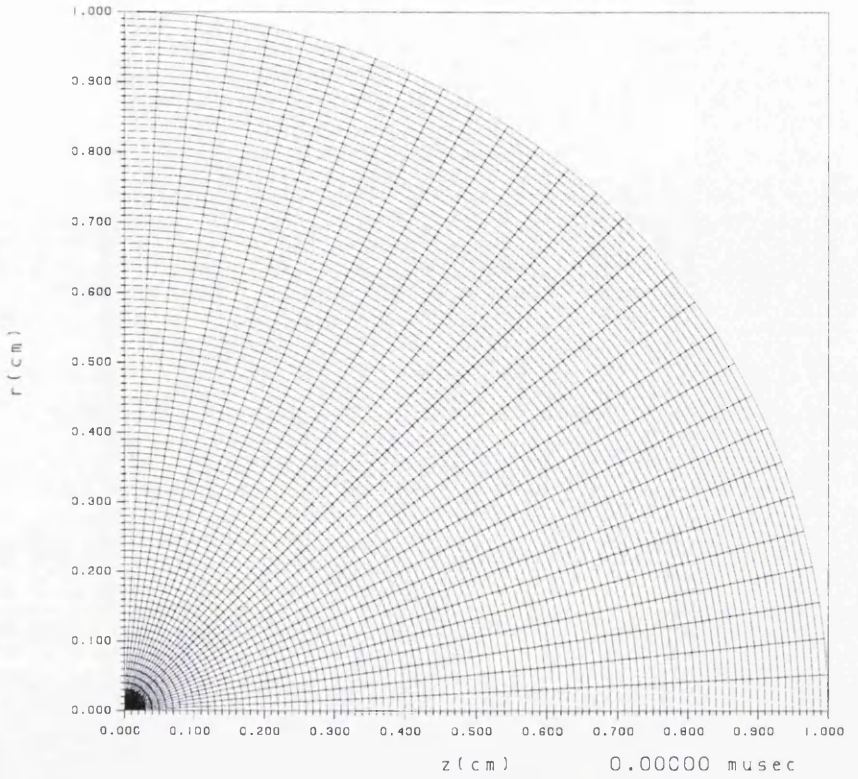


Figure 3.9: Initial Mesh for Noh's problem.

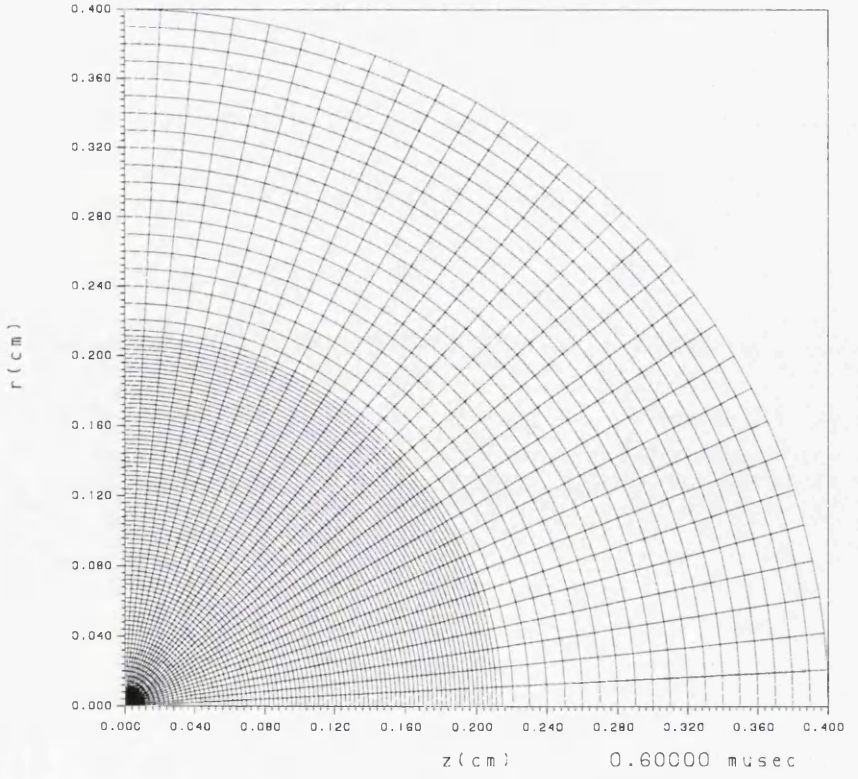


Figure 3.10: Mesh for Noh's problem calculated with monotonic artificial viscosity at 0.6 μs.

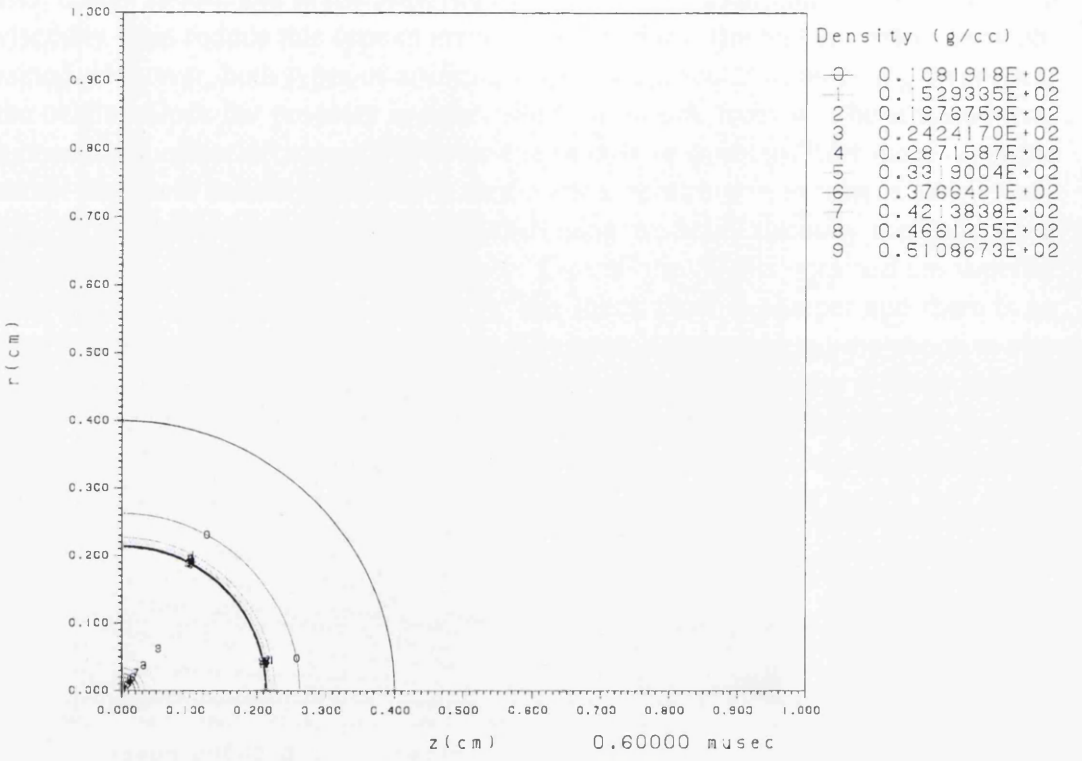


Figure 3.11: Density contour plot for Noh's problem calculated with monotonic artificial viscosity at 0.6 μs.

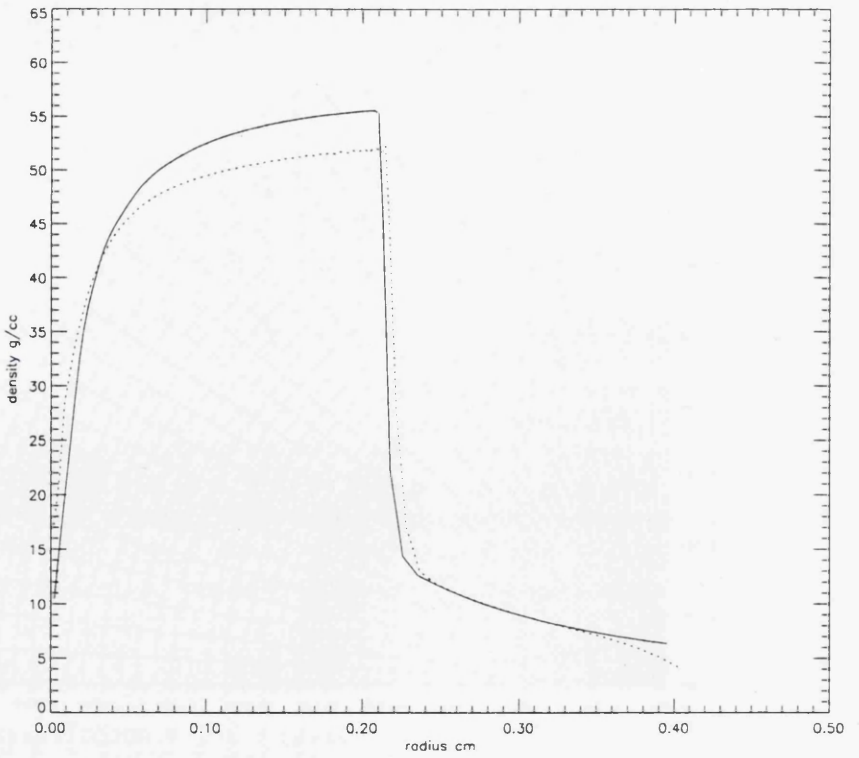


Figure 3.12: Density profile plot for Noh's problem calculated with bulk and monotonic artificial viscosity at 0.6 μs.

3.8 Material Strength

The treatment of material strength in CORVUS closely follows the method developed by Wilkins [38], but discretized with a finite element methodology, consistent with that used for the hydrodynamics, as described above. The treatment is primarily aimed at the ductile elasto-plastic response of metals. The method decouples the stress into an isotropic pressure and stress deviator components. The isotropic response is described by an equation of state for each material, which determines the pressure as a function of material density and specific internal energy (3.9). The constitutive relations for the deviator part of the material response are in incremental form, and follow the Prandtl-Reuss treatment of elasto-plastic flow, which deals with the elastic and plastic strains simultaneously. The elastic response is described by the hypo-elastic abstraction of linear elasticity. The functions for the yield strength and shear modulus are those given by Steinberg, Cochrran and Guinan [42], which allows for strain and pressure hardening and thermal softening, but does not include strain rate dependence; the stress deviators being forced to lie on this yield surface by employing Wilkin's radial return algorithm [38].

The equations for the conservation of momentum and internal energy (3.7,3.8) are now rewritten to include the influence of material strength.

$$\rho \frac{Du}{Dt} = -\frac{\partial(p+q)}{\partial x} + \frac{\partial S_{xx}}{\partial x} + \frac{\partial S_{xy}}{\partial y} + \frac{S_{xy}}{y} \tag{3.53}$$

$$\rho \frac{Dv}{Dt} = -\frac{\partial(p+q)}{\partial y} + \frac{\partial S_{yy}}{\partial y} + \frac{\partial S_{xy}}{\partial x} + \frac{S_{yy} - S_{zz}}{y} \tag{3.54}$$

$$\rho \frac{D\varepsilon}{Dt} = -(p+q)\,\nabla \cdot \mathbf{u} + \rho \frac{DW^p}{Dt} \tag{3.55}$$

Clearly if the stress deviator terms and rate of plastic work are all zero, then the original hydrodynamic equations (3.7,3.8) are recovered.

3.8.1 Time discretization for material strength model

The following time descritization should be contrasted with that for the pure hydrodynamic equations given above. Essentially, the stress deviators and plastic strain rate must be evaluated before the modified conservation equations (3.53,3.54,3.55) can be solved.

- **a) Pressure Predictor Step (n to $n+\frac{1}{2}$)**

  First the elastic prediction of the Stress Deviators, S^*_{ij}, neglecting plastic strain rate components is obtained from,

$$\left(\frac{DS^*_{xx}}{Dt}\right)^{n+\frac{1}{2}} = \left(2\mu e_{xx} + 2\omega_{xy}S^*_{xy}\right)^n \tag{3.56}$$

$$\left(\frac{DS^*_{xy}}{Dt}\right)^{n+\frac{1}{2}} = \left(2\mu e_{xy} + \omega_{xy}(S^*_{yy} - S^*_{xx})\right)^n \tag{3.57}$$

$$\left(\frac{DS^*_{yy}}{Dt}\right)^{n+\frac{1}{2}} = \left(2\mu e_{yy} + 2\omega_{xy}S^*_{xy}\right)^n \tag{3.58}$$

  where the Strain Rate deviators are given by,

$$e_{xx} = \varepsilon_{xx} - \frac{1}{3}(\nabla \cdot \mathbf{u}) \tag{3.59}$$

$$e_{xy} = \varepsilon_{xy} \tag{3.60}$$

$$e_{yy} = \varepsilon_{yy} - \frac{1}{3}(\nabla \cdot \mathbf{u}) \tag{3.61}$$

  and the strain rates and spin tensor are,

$$\varepsilon_{xx} = \frac{\partial u}{\partial x} \tag{3.62}$$

$$\varepsilon_{xy} = \frac{1}{2}\left(\frac{\partial v}{\partial x} + \frac{\partial u}{\partial y}\right) \tag{3.63}$$

$$\varepsilon_{yy} = \frac{\partial v}{\partial y} \tag{3.64}$$

$$\varepsilon_{zz} = \frac{v}{y} \tag{3.65}$$

$$\omega_{xy} = \frac{1}{2}\left(\frac{\partial u}{\partial y} - \frac{\partial v}{\partial x}\right) \tag{3.66}$$

  Applying the yield limit to S^*_{ij} to obtain S_{ij},

$$|S^*|^2 = 2({S^*_{xx}}^2 + {S^*_{xy}}^2 + {S^*_{yy}}^2 + S^*_{xx} \cdot S^*_{yy})^{n+\frac{1}{2}} \tag{3.67}$$

$$S^{n+\frac{1}{2}}_{ij} = \begin{cases} S^*_{ij} & : \quad |S^*| \le \sqrt{\frac{2}{3}}Y^n \\ \sqrt{\frac{2}{3}}\frac{Y^n}{|S^*|}S^*_{ij} & : \quad |S^*| > \sqrt{\frac{2}{3}}Y^n \end{cases} \tag{3.68}$$

The plastic Strain Rate Deviators, ε^p, are obtained by writing the strain rate deviators e as the sum of elastic and plastic components,

$$e_{ij} = e^e_{ij} + \varepsilon^p_{ij} \tag{3.69}$$

then

$$(\varepsilon^p_{xx})^n = e^n_{xx} - \frac{1}{2\mu}\left(\frac{DS_{xx}}{Dt} - 2\omega^n_{xy}S^n_{xy}\right) \tag{3.70}$$

$$(\varepsilon^p_{xy})^n = e^n_{xy} - \frac{1}{2\mu}\left(\frac{DS_{xy}}{Dt} - \omega^n_{xy}(S^n_{yy} - S^n_{xx})\right) \tag{3.71}$$

$$(\varepsilon^p_{yy})^n = e^n_{yy} - \frac{1}{2\mu}\left(\frac{DS_{yy}}{Dt} + 2\omega^n_{xy}S^n_{xy}\right) \tag{3.72}$$

where

$$\frac{DS_{ij}}{Dt} = \frac{S^{n+\frac{1}{2}}_{ij} - S^n_{ij}}{\frac{1}{2}\Delta t} \tag{3.73}$$

and the internal energy is given by,

$$\rho^n \frac{D\varepsilon^{n+\frac{1}{2}}}{Dt} = (-p\nabla \cdot u)^n + \left(\rho\frac{DW^p}{Dt}\right)^n \tag{3.74}$$

where

$$\left(\rho\frac{DW^p}{Dt}\right)^n = 2(S_{xx}\varepsilon^p_{xx} + S_{yy}\varepsilon^p_{yy} + S_{xy}\varepsilon^p_{xy})^n + (S_{xx}\varepsilon^p_{yy} + S_{yy}\varepsilon^p_{xx})^n \tag{3.75}$$

The density update is performed as for the hydrodynamic case, coordinates being advanced to the $n+\frac{1}{2}$ time level; the cell volume recaculated and hence the new density obtained. The pressure is then obtained from (3.9), but using the internal energy obtained from (3.74).

- **b) Velocity Update**

  The element contributions to the nodal accelerations are then calculated, noting that the u and v components have different forms as given by (3.54,3.55), and that all variables are at the $n+\frac{1}{2}$ time level. The nodal positions are updated as for the hydrodynamic case,

$$\mathbf{x}^{n+1} = \mathbf{x}^n + \Delta t\ \bar{\mathbf{u}} \qquad \forall \text{nodes} \tag{3.76}$$

where

$$\bar{\mathbf{u}} = \frac{1}{2}(\mathbf{u}^n + \mathbf{u}^{n+1}) \tag{3.77}$$

- **c) Volume, Internal Energy and Pressure Update (n to n+1)**

  The elastic prediction of the stress deviators follows as for step a), but note the time levels of the variables,

$$\left(\frac{DS^*_{xx}}{Dt}\right)^{n+1} = 2\mu e^{n+\frac{1}{2}}_{xx} + 2\omega^{n+\frac{1}{2}}_{xy}\bar{S}^*_{xy} \tag{3.78}$$

  where

$$\bar{S}^*_{xy} = \frac{1}{2}(S^{*\,n+\frac{1}{2}}_{xy} + S^*_{xy}n)$$

$$...etc$$

  Note that this requires the solution of simultaneous equations to obtain $S^{n+\frac{1}{2}}_{ij}$. The yield limit is applied as in step a). Evaluate the elastic strain rate deviators (at $n+\frac{1}{2}$),

$$(e^e_{xx})^{n+\frac{1}{2}} = \frac{1}{2\mu}\left(\frac{D\bar{S}_{xx}}{Dt} - 2\omega^{n+\frac{1}{2}}_{xy}S^{n+\frac{1}{2}}_{xy}\right) \tag{3.79}$$

$$...etc$$

  where,

$$\frac{D\bar{S}_{xx}}{Dt} = \frac{S^{n+1}_{xx} - S^n_{xx}}{\Delta t}$$

  Evaluate rate of change of elastic work,

$$\left(\rho\frac{DW^e}{Dt}\right) = 2(e^e_{xx}\bar{S}_{xx} + e^e_{xy}\bar{S}_{xy} + e^e_{yy}\bar{S}_{yy}) + e^e_{xx}\bar{S}_{yy} + e^e_{yy}\bar{S}_{xx} \tag{3.80}$$

$$W^{e^{n+1}} = W^{e^n} + \frac{1}{\rho^{n+\frac{1}{2}}}\left(\rho\frac{DW^e}{Dt}\right)\Delta t \tag{3.81}$$

  Update plastic strain rate deviators

$$(\varepsilon^p_{xx})^{n+\frac{1}{2}} = e^{n+\frac{1}{2}}_{xx} - (e^e_{xx})^{n+\frac{1}{2}} \tag{3.82}$$

$$...etc$$

  Update Equivalent Plastic Strain

$$\varepsilon^{p^{n+1}} = \varepsilon^{p^n} + \Delta t\sqrt{\frac{4}{3}(\varepsilon^p_{xx})^2 + (\varepsilon^p_{xy})^2 + (\varepsilon^p_{yy})^2 + \varepsilon^p_{xx}\varepsilon^p_{yy}} \tag{3.83}$$

Evaluate rate of change of plastic work as in (a), with all variables at $n+\frac{1}{2}$:

$$\rho\frac{DW^p}{Dt} = 2(S_{xx}\varepsilon^p_{xx} + S_{yy}\varepsilon^p_{yy} + S_{xy}\varepsilon^p_{xy}) + S_{xx}\varepsilon^p_{yy} + S^p_{yy} \tag{3.84}$$

$$W^{p^{n+1}} = W^{p^n} + \frac{1}{\rho^{n+\frac{1}{2}}}\left(\rho\frac{DW^p}{Dt}\right)\Delta t$$

Then update internal energy as per (a) from n to n+1, and update pressure.

3.8.2 Finite Element analysis for strength model

As the isotropic (pressure) contributions are decoupled from the deviatoric components of the material response, the finite element analysis given in section 3.6 is valid in its entirety and represents the behaviour of materials without strength. Consequently only the additional terms introduced by the strength model will be considered here, namely the elastic prediction of the stress deviators, the strain rate deviators, and the rate of plastic work.

- **i) Stress Prediction**

  Consider a typical term for the rate of change of the (unlimited) stress deviator:

$$\frac{DS^*_{xx}}{Dt} = 2\mu e_{xx} + 2\omega_{xy}S^*_{xy} \tag{3.85}$$

  We need to calculate this value for each element. Define a weak form, and consider the integral over a typical element, noting that in the currently notation y=r and x=z, and thus $d\Omega = ydxdy$:

$$\int_\Omega \frac{DS^*_{xx}}{Dt}W_j y dxdy = 2\mu\int_\Omega e_{xx}W_j y dxdy + 2\int_\Omega \omega_{xy}S^*_{xy}W_j y dxdy \tag{3.86}$$

  The strain rate deviators can be expanded in terms of strain rates and $\nabla\cdot\mathbf{u}$,

$$e_{xx} = \frac{2}{3}\frac{\partial u}{\partial x} - \frac{1}{3}\left(\frac{\partial v}{\partial y} + \frac{v}{y}\right) \tag{3.87}$$

  Using an approach consistent with that used for the momentum equation in section 3.6.2, but noting that here we only require one value, e_{xx}, for the element. In the momentum equation we require the contributions to the four nodes, and so used the four shape functions that span the element; here we could simply put W=1, but for consistency we choose:

$$W = \sum_{j=1}^{4}\frac{1}{y}N_j = \frac{1}{y} \tag{3.88}$$

Consider the first of the right hand side integrals:

$$\int_\Omega e_{xx} W y dx dy = \int_\Omega \sum_k \left(\tfrac{2}{3} u_k \tfrac{\partial N_k}{\partial x} - \tfrac{1}{3} \left(v_k \tfrac{\partial N_k}{\partial y} + \tfrac{v_k N_k}{y} \right) \right) \tfrac{1}{y} y dx dy$$

$$= \sum_k \tfrac{2}{3} u_k \int\int \tfrac{\partial N_k}{\partial x} det J d\xi d\eta$$

$$- \sum_k \tfrac{1}{3} v_k \int\int \tfrac{\partial N_k}{\partial y} det J d\xi d\eta$$

$$- \sum_k \tfrac{1}{3} \tfrac{v_k}{y} \int\int N_k det J d\xi d\eta \qquad (3.89)$$

The final term still has a $\frac{1}{y}$ dependence: previously the radial dependence has always cancelled out. A number of more sophisticated treatments of this term were tried, but the simple mean value was found to be the most successful and robust approximation. The other terms can be expanded similarly, giving an explicit expression for $\{\int \psi\} = \int_{-1}^{1}\int_{-1}^{1} \psi det J d\xi d\eta$ ss deviators. Using the shortened notation

$$\{\int \psi\} = \int_{-1}^{1}\int_{-1}^{1} \psi det J d\xi d\eta \qquad (3.90)$$

the expression can be written:

$$\frac{DS^*_{xx}}{Dt} = \left(\tfrac{1}{\{J\}}\right) \left[2\mu \sum_k \left(\tfrac{2}{3} u_k \{\int \tfrac{\partial N_k}{\partial x}\} - \tfrac{1}{3} v_k \{\int \tfrac{\partial N_k}{\partial y}\} - \tfrac{1}{3} \tfrac{v_k}{y} \{\int N_k\} \right) \right.$$

$$\left. + 2S^*_{xy} \sum_k \left(\tfrac{1}{2} u_k \{\int \tfrac{\partial N_k}{\partial y} - \tfrac{1}{2} v_k \{\partial N_k \partial x\} \right) \right] \qquad (3.91)$$

All the other stress deviator rates can be calculated in a similar manner. The plastic strain rate deviators can also be analysed following the same method, taking care to use the limited stress deviators where appropriate.

- **ii) Momentum Equation**

The strength model adds extra terms to the momentum equation involving the stress deviators. In section i) above expressions were derived for the unlimited stress deviators, which, when limited, will contribute to the acceleration of the nodes. This derivation as discussed above is consistent with the Petrov-Galerkin discretization of the isotropic components of the material response.

The two components of momentum can be written:

$$\rho \frac{Du}{Dt} = -\frac{\partial p}{\partial x} + \frac{\partial S_{xx}}{\partial x} + \frac{\partial S_{xy}}{\partial y} + \frac{S_{xy}}{y} \qquad (3.92)$$

$$\rho \frac{Dv}{Dt} = -\frac{\partial p}{\partial y} + \frac{\partial S_{yy}}{\partial y} + \frac{\partial S_{xy}}{\partial x} + \frac{S_{yy} - S_{zz}}{y} \qquad (3.93)$$

On the right hand side, the first term is exactly as in the pure hydrodynamic case, and the next two terms can be expanded by exactly the same method, as, like pressure, the S_{ij} terms are constant across the element. The final term expands like the left hand side, taking $\frac{1}{y}$ as a mean value, elementwise constant, as in the stress deviator analysis. The contribution to the acceleration of the local node j of a typical element can thus be written:

$$\{\int \rho^e\}\frac{Du_j}{Dt} = p^e\{\int \frac{\partial N_j}{\partial x}\} - S^e_{xx}\{\int \frac{\partial N_j}{\partial x}\}$$

$$-S^e_{xy}\{\int \frac{\partial N_j}{\partial y}\} + S^e_{xy}\frac{1}{y}\{\int N_j\} \qquad (3.94)$$

and

$$\{\int \rho^e\}\frac{Dv_j}{Dt} = p^e\{\int \frac{\partial N_j}{\partial y}\} - S^e_{yy}\{\int \frac{\partial N_j}{\partial y}\}$$

$$- S^e_{xy}\{\int \frac{\partial N_j}{\partial x}\} + (S^e_{yy} - S^e_{zz})\frac{1}{y}\{\int N_j\} \qquad (3.95)$$

Note $S_{xx} + S_{yy} + S_{zz} = 0$.

- **iii) Internal Energy Equation**

  For the specific internal energy calculation, the strength model adds an additional term, the rate of plastic work. This has already been defined in terms of a products of the stress deviators and the plastic strain rates. All that has to be done is to evaluate the contribution that the plastic work makes to the internal energy, in a manner consistent with the evaluation of the $-p\nabla \cdot \mathbf{u}$ term. All the terms are known and piecewise constant, and thus the discrete form of the internal energy equation can be written:

$$\{\int \rho^e\}\frac{D\varepsilon^e}{Dt} = -p^e \sum_j \left(u_j y_j \{\int \frac{\partial N_j}{\partial x}\} + v_j y_j \{\int \frac{\partial N_j}{\partial y}\} + v_j \{\int N_j\} \right)$$

$$+ [2(S_{xx}\varepsilon^p_{xx} + S_{yy}\varepsilon^p_{yy} + S_{xy}\varepsilon^p_{xy}) + (S_{xx}\varepsilon^p_{yy} + S_{yy}\varepsilon^p_{xx})] \sum_j y_j \{\int N_j\} \qquad (3.96)$$

3.9 Strength Test problems

3.9.1 Berylium stopping shell

The berylium stopping shell problem is used to check the symmetry and energy conservation of the material strength treatment. The problem consists of a 2 cm thick spherical berylium shell with an inner radius of 8 cm. The equation of state

used was developed at LANL by R. K. Osbourne [43], and takes the form of a quadratic fit to experimental data,

$$p = \frac{a_1 + \bar{a}\eta^2 + e(b_0 + b_1 + b_2\eta^2) + e^2(c_0 + c_1\eta)}{e + e_0} \tag{3.97}$$

where $a_1, a_2, a_2^*, b_0, b_1, b_2, c_0, c_1, e_0$ are material dependent parameters, $\eta = \rho/\rho_0 - 1$ and,

$$\bar{a} = \begin{cases} a_2 & : \quad \eta \geq 0 \\ a_2^* & : \quad \eta < 0 \end{cases} \tag{3.98}$$

A constant yield strength of 3.3 kb and a constant shear modulus of 3.3 Mb is also assumed.

An initial radial velocity distribution is applied to keep the shell incompressible while it implodes. The shells kinetic energy is then gradually converted into plastic work during the implosion until the shell stops. The initial radial velocity distribution assumed is of the form,

$$u = \left(\frac{r_0}{r}\right)^2 u_0 \tag{3.99}$$

where r_0 is the initial radius and u_0 the initial radial velocity of the inner surface. It remains then to define u_0, this is achieved by specifying a final inner radius and theoretically determining the initial velocity required to stop at this radius [44]. In this case the berylium shell was required to stop with an inner radius of 3 cm, which occurs theoretically at $100.0\mu s$ with $u_0 = 0.067504 cms^{-1}$ [45].

Numerical Results

The problem was calculated on a 2° angular mesh with 45 radial zones in the berylium shell as shown in Fig. 3.13. The solution obtained at $100.0\mu s$ is given in Fig. 3.14. The calculation stopped at 3.0 cm and the total energy only dropped by 0.01 % during the calculation. The Lagrangian scheme used in CORVUS does not conserve energy perfectly because the nodal masses are allowed to vary with time, due to the finite element nature of the scheme, and the *PdV* formulation does not explicitly guarantee that identical work is done in the momentum and corrector internal energy steps.

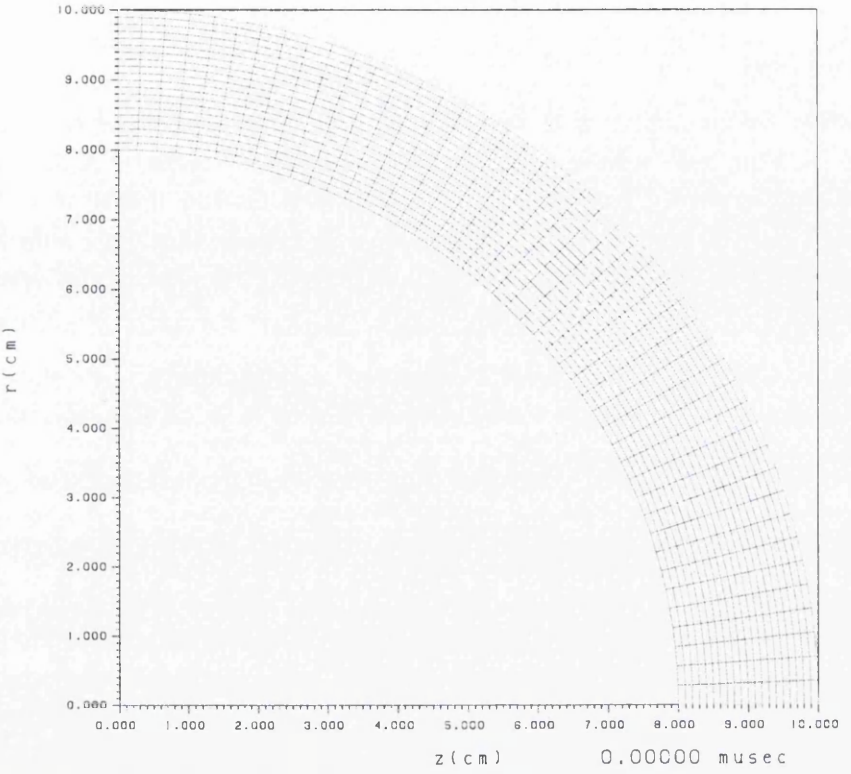


Figure 3.13: Initial mesh for berylium stopping shell problem.

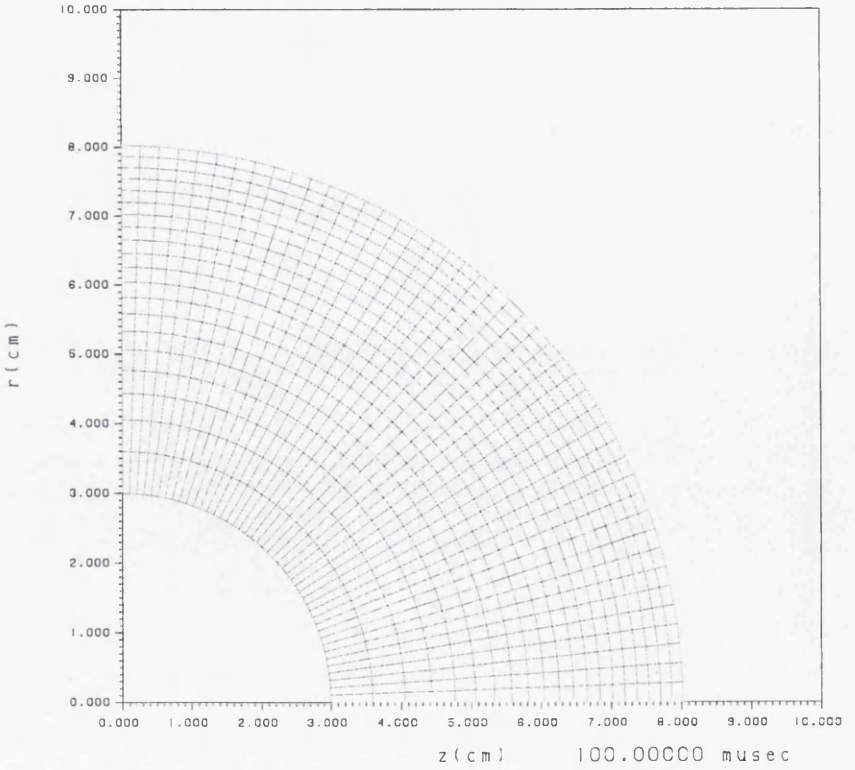


Figure 3.14: Mesh for berylium stopping shell problem at 100.0 μs.

3.9.2 Taylor rod impact test

Definition

The Taylor impact test was selected to test the modelling of strain hardening. The problem involves the normal impact of a copper cylinder with an axial velocity on a rigid target. The cylinder is 10 cm long, 1.5 cm in diameter and it has an impact velocity of $0.019 cms^{-1}$. The copper was modelled with an Osbourne equation of state (3.97) and a constant shear modulus of 0.477 Mb. The yield strength was given by,

$$Y = 0.003(1 + 1.2934\varepsilon^p) \tag{3.100}$$

where ε^p is the plastic strain.

Numerical Results

Fig. 3.15 shows the mesh and Fig. 3.16 the plastic work contours obtained for the Taylor impact problem at $250.0\mu s$. The plastic wave has travelled just over 5 cm down the rod from the impact site, which is in good agreement with the 1D plastic wave velocity, which suggests the plastic wave should be 5.1 cm from the impact site [45].

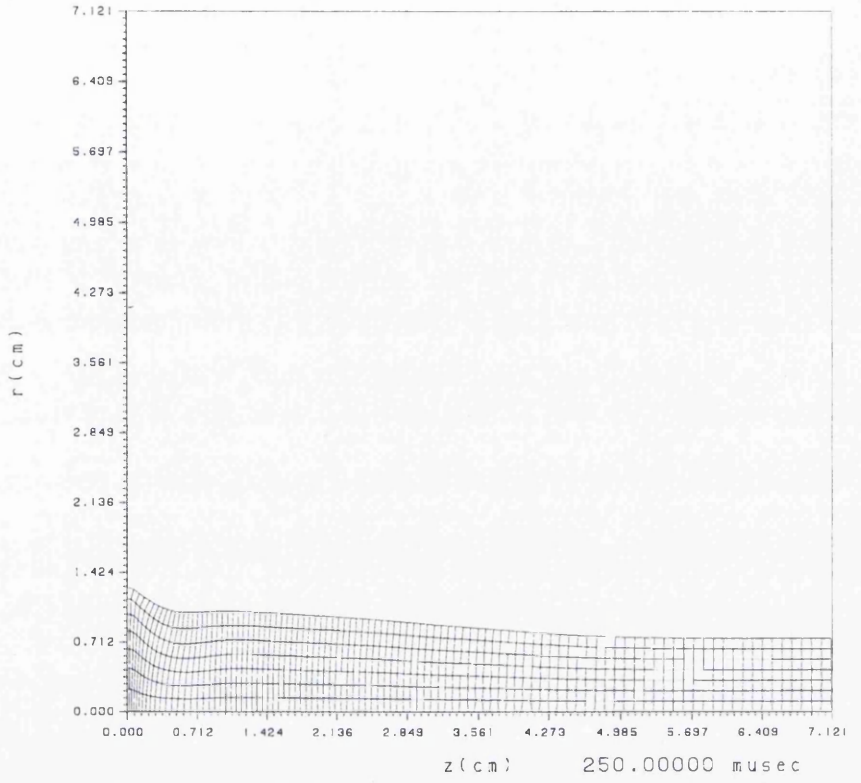


Figure 3.15: Mesh for Taylor copper rod impact test at 250.0 μs.

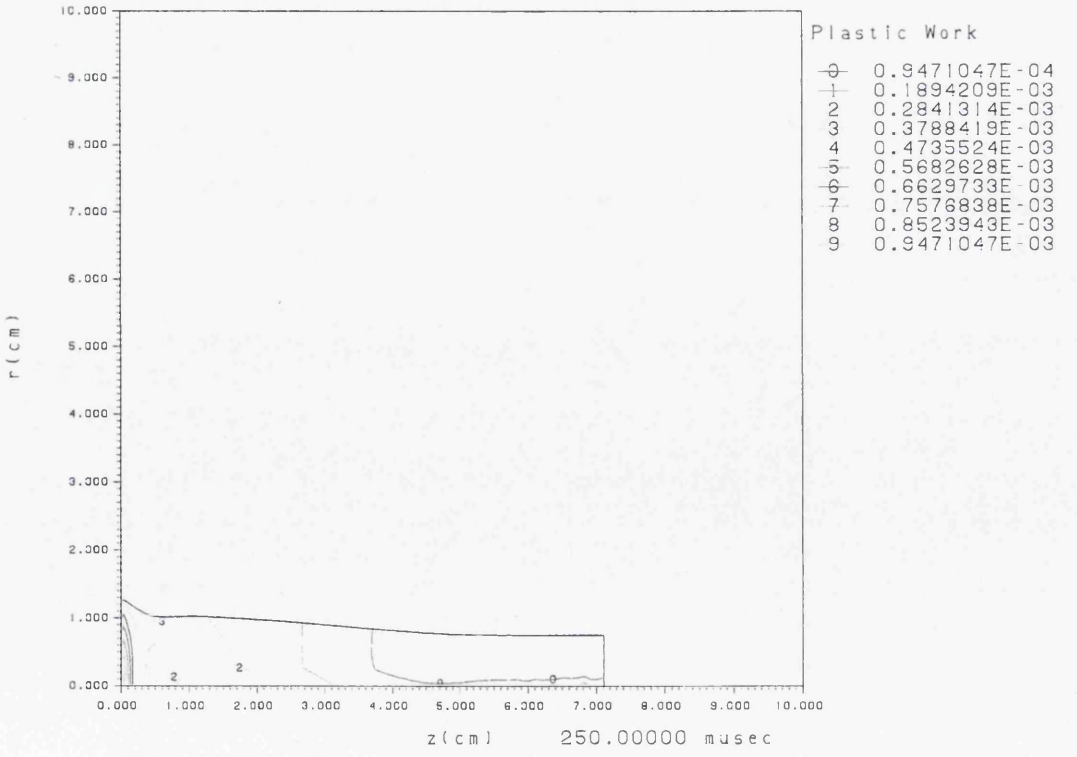


Figure 3.16: Plastic work contour plot for Taylor copper rod impact test at 250.0 μs.

3.9.3 Berylium vibrating plate test problem

Definition

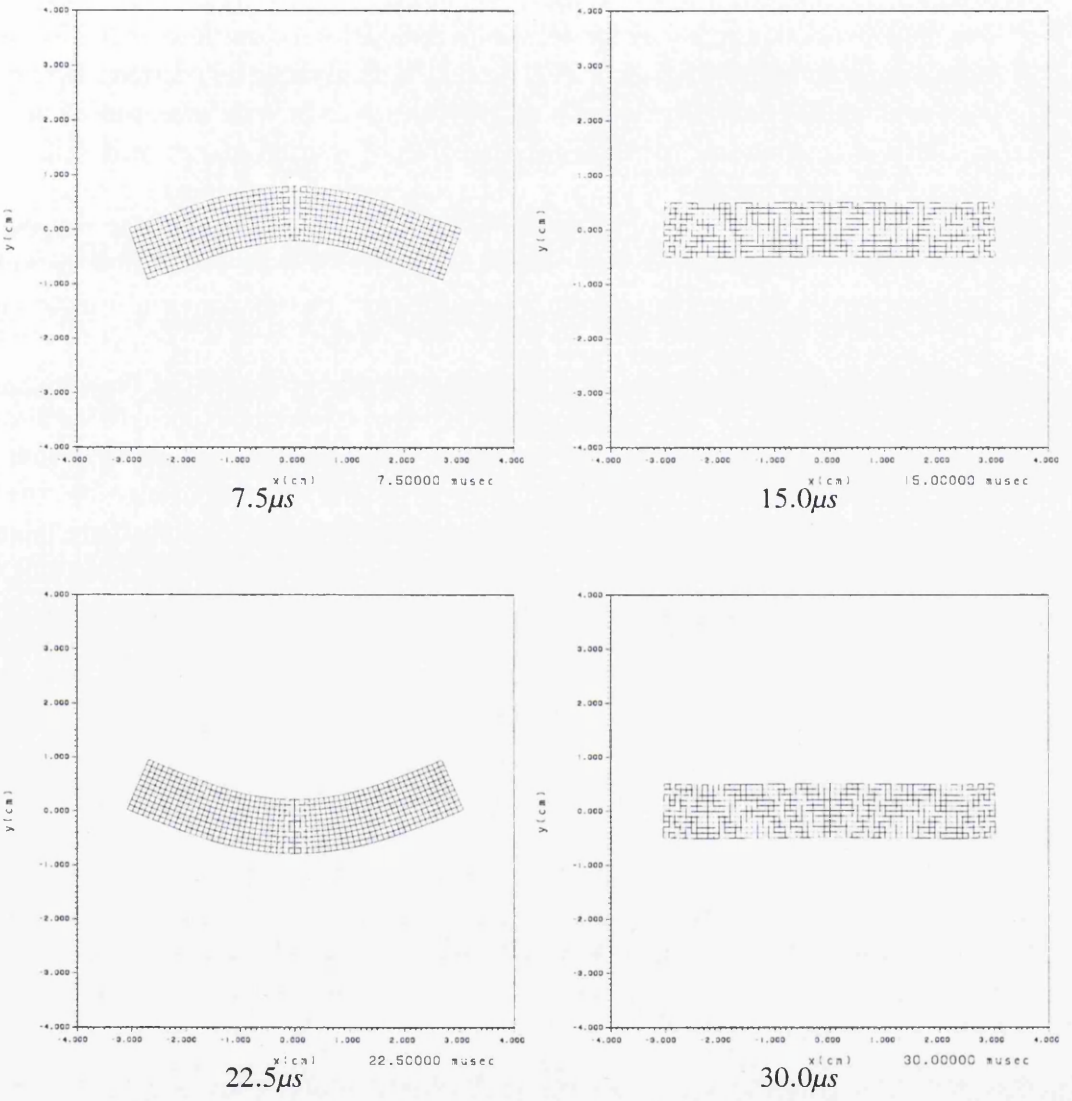
The berylium vibrating plate is a purely elastic test problem. It consists of a rectangular plate of infinite extent in the z-direction with no supports or constraints. The x- and y- dimensions are 6 cm and 1 cm respectively. The berylium was again modelled with an Osbourne equation of state (3.97). A simple constant yield strength of 1 Mb and shear modulus of 1.51 Mb constitutive model is used. The first flexural mode is selected by defining the initial velocity distribution in the y- direction as:

$$\begin{aligned} u_y &= \tilde{\omega}_1 A[C_1(sinh(\Omega_1(x+3)) + sin(\Omega_1(x+3))) \\ &- S_1(cosh(x+3)) + cos(\Omega_1(x+3)))] \end{aligned} \tag{3.101}$$

where $\tilde{\omega}_1 = 0.235974$, A=0.004337, $C_1 = 56.6369$, $S_1 = 57.6355$ and $\Omega_1 = 0.7883902$. This should give a period of $26.63\mu s$ and an amplitude of 0.5 cm [45].

Numerical Results

The problem was calculated with 10 cells in the y-direction and 60 in the x-direction. A $30.0\mu s$ period with an amplitude of about 0.5 cm was obtained numerically as illustrated in Fig. 3.17. It is not surprising that the calculated period is longer than the analytical prediction, as the theory assumes long thin plates, while the numerical simulation is applied to a 6:1 aspect ratio plate. The energy conservation obtained is reasonable with 0.15 % of the total energy lost during the calculation.



7.5μs 15.0μs

22.5μs 30.0μs

Figure 3.17: Berylium vibrating plate test problem

3.10 Lagrangian multi-material cell treatment

The proceeding sections have detailed the Lagrangian step as it is applied to single material cells. In this section the modifications that have been made to the Lagrangian scheme to allow for the inclusion of multi-material cells will be discussed. The reader should already be familiar with the data structure introduced to support multi-materials cells which was also discussed in a previous section. However, a brief reminder of the salient details will first be given here before proceeding.

Volume fractions are defined for each ALEing materials in each cell. These volume fractions define the fraction of the cell volume occupied by each of these materials. In a single material cell within an ALEing region, for example, the volume fraction component for the cells own material will be 1.0, whilst the volume fractions for all the other material components will be 0.0. In a multi-material cell more than one of the volume fraction components will be non-zero, indicating that more than one material is present. The state variables for each of the components in a multi-material cell are stored in separate arrays, which are indirectly addressed. Volume fraction weighted averages of these components are stored for each state variable of each multi-material cell. In order to simplify the discussion, the pure hydrodynamics without strength is considered first. In this case there are three key areas that require modification: equation of state calls, the internal energy or work updates and the acceleration calculation.

CORVUS employs a vector equation of state (EOS) package. This simply means that one call to the package returns pressures and sound speeds for as many elements as required. This is a natural approach for unstructured codes such as CORVUS and allows the EOS package to be vectorised on vector architectures. The existing calls to the EOS package are essentially unmodified, but the values obtained for the multi-material cells are discarded. Additional EOS calls are added which pass vectors containing the state variables for all the multi-material cells material components and return pressures and sound speeds for all these partial components. Volume fraction averages are then assembled for the cell pressures and mass fraction weighted averages for the sound speeds of the multi-material cells,

$$P_i = \sum_{k=1}^{m} f_{k,i} p_{k,i} \tag{3.102}$$

$$c_i = \frac{\sum_{k=1}^{m} m_{k,i} c_{k,i}}{\sum_{k=1}^{m} m_{k,i}} \tag{3.103}$$

The predictor and corrector internal energy or work updates for the single material cells simply calculate the work done on or by the element in terms of an increment in $(p+q)\nabla \cdot \bar{u}$ (3.3). This is straightforward and simply requires the pressure and volume change of the element to be known. The latter is simply calculated

from the divergence of the velocity field for the element. However, when it comes to updating the partial internal energies for separate material within a multi-material cell then this is less well defined. Separate pressures are certainly available for the individual components, and the overall cell or element volume change is known, as discussed above. But the fraction of the cell volume change that should be apportion to each individual material component is undefined.

The current default option for CORVUS follows the assumption of equal compressibility made in most multi-material Eulerian and ALE codes [46]. This is clearly unphysical for most material combination and will not satisfy the pressure continuity at the material interface that shock physics demands. These concerns will be revisited later in this thesis and an alternative treatment proposed. However, the equal compressibility assumption does provide a relatively simple and robust internal energy update procedure, the volume fractions remaining unchanged during the Lagrangian phase, and component volume changes simply being taken as the product of the material volume fraction and cells volume change. The internal energy updates for a material component at the half and full step are,

$$\varepsilon_{k,i}^{n+\frac{1}{2}} = \varepsilon_{k,i}^{n} + \frac{1}{2}\Delta t \frac{(p_{k,i}^{n} + q_i^{n})}{m_{k,i}} f_{k,i} \nabla \cdot \mathbf{u}^{n} \tag{3.104}$$

$$\varepsilon_{k,i}^{n+1} = \varepsilon_{k,i}^{n} + \Delta t \frac{(p_{k,i}^{n+\frac{1}{2}} + q_i^{n})}{m_{k,i}} f_{k,i} \nabla \cdot \bar{\mathbf{u}} \tag{3.105}$$

It should be noted that while separate material component pressures are used to update the internal energy of each material component, the artificial viscosity or q term used is the same for all material components.

The acceleration calculation is effectively unchanged with cell average values used for multi-material cells. The artificial viscosity is calculated at the start of the time step and essentially does not currently distinguish between the single and multi-material cells; the lengthscale for the viscosity simply being calculated from cell average values for the multi-material cells. However, the velocity slope ratios, used to define the monotonic slope limiters, across multi-material cell boundaries, are set to zero.

Chapter 4

Sliding Interfaces

CORVUS provides two different interface treatments, a volume of fluid (VOF) based interface reconstruction method, which will be discussed later, and a Lagrangian slide algorithm. The former is a very robust technique and is the method of choice for interfaces which undergo high deformation. However, if an interface is well behaved the Lagrangian slide treatment is preferred, as it should be more accurate and is more amenable to the addition of interface physics. This chapter will focus on the Lagrangian slide algorithm that has been developed and how it has been modified to introduce void closure, void opening and friction.

A slide algorithm defines a material interface as two discrete Lagrangian surfaces or slide lines. This allows the nodal acceleration calculation to be decomposed, and different procedures used to obtain the normal and tangential accelerations. In calculating the normal acceleration the normal force is assumed to be continuous at the interface, reflecting the reaction of one material against the other. However, the details of the tangential acceleration calculation depend on what is assumed about the tangential force or friction acting between the two materials. Most codes simply assume zero friction, but in principle any frictional force up to the limit of a locked interface can be introduced. The main difficulty here is that the physics of dynamic friction for the regime of interest for most hydrocode applications, that is pressures in the $50 - 200 Mbar$ range and relative slip velocities of about $2 mm\mu s^{-1}$, is not well understood [47]. However, the facility to introduce a frictional force through a slide algorithm [48] is valuable for dynamic friction research, since it allows friction models to be assessed by code comparison against experimental data. It is also valuable in making engineering assessments of the potential importance of dynamic friction.

Slide algorithms can be subdivided into two main types. The first stems from structural analysis, and is mainly concerned with enforcing the correct contact constraints at the interface. The approach taken for achieving this is to calculate the forces across the interface and then apply them as a force boundary conditions. Typical examples of this approach are the Lagrange multiplier method, discussed in

[49] and the penalty method given in [50].

In contrast the hydrocode community is more concerned with propagating the correct stress waves across material interfaces. Most hydrocodes, including HEMP [38] and TENSOR [13], designate one of the surfaces in each pair of slide lines as a master, and the other as the slave. This choice is somewhat arbitrary, but the rules of thumb are that the slave should be the material most likely to deform to the other material. These algorithms then treat the interfaces as merged in calculating the normal acceleration of the interface, by mapping the mass and stress distribution from the slave surface onto the master. The CORVUS scheme was strongly influenced by these schemes, but offers improvements in the symmetry of the scheme, and so reduces the sensitivity of the solution to the choice of which surface is designated as the master, and which the slave. The basic CORVUS slide algorithm was developed principally by Whittle [51] and later extended by the author to include a friction model [48].

4.1 The CORVUS slide algorithm

Two approaches were considered for the development of slide in CORVUS. The original slide algorithm was written when the code was purely hydrodynamic. The algorithm was designed to be as consistent as possible with the underlying hydrodynamic scheme, and worked by using the pressures in elements on one side of the interface (the slave side) to define a piecewise constant pressure force on the other (master) side, which was then integrated and added to the other forces acting on nodes of the master side. With the assumption that the master interface was now accelerated "correctly", the slave surface was forced to conform to the master in a "put-back-on" step (PBO) that conserved the slave side tangential velocity. During these calculations, master and slave elements are assumed to remain quadrilateral, even though they may have penetrated and become arbitrary polygons.

This algorithm had several failings. It is a classic "master/slave" treatment, and so is not symmetric, and choosing the master/slave relationship at an interface is not always straightforward. The basic rule appears to be to make the stronger material the master, or the denser if they are both weak, or, if there is still no clear choice, the master should be the material being driven, rather than that driving the motion. Secondly, the assumption that the free surface tangential velocity of the slave is a reasonable approximation to the constrained tangential velocity is questionable. Thirdly, the PBO does not enforce a continuous normal velocity. Finally, and crucially, the use of the pressure in the slave elements ignores material strength terms, and this means that the algorithm is invalid for slave side materials with any strength and it is not clear how to include the strength terms.

For all its limitations, the algorithm was quite effective. However, it was recognised that the slide model would have to be adapted to include material strength

terms. Attempts to adapt the existing algorithm were unsuccessful, resulting in instability, long wavelength perturbations, and inaccuracy.

As a result of this experience a new slide algorithm was devised which is now the recommended option in CORVUS. The philosophy in this algorithm was to compromise a little on accuracy in favour of a robust symmetric treatment. Forcing the algorithm to be more symmetric makes it much easier to incorporate strength terms in a physically sensible manner and also makes the algorithm more robust. Applying the method to real problems has shown that it is less accurate than the old slide algorithm for coarse mesh calculations on problems with no strength on the slave side, but both converge to similar results on fine meshes.

The new slide algorithm breaks down into several steps, and these are detailed below. The slide lines are recorded as a single list of node numbers, with a pointer to the vector location of the start and end nodes. Slide lines are considered in pairs, master then slave. Associated with the list of slide nodes there are parallel lists of the elements that border the slide line, unit tangent vectors at each node and so forth.

The first four steps are performed after node Lagrangian accelerations have been calculated, but before they have been applied to move any nodes.

- **Step 1** Calculate normals and tangents for all nodes and element faces on the slidelines.
- **Step 2** Calculate free surface tangential velocities, and effective pressure force at nodes.
- **Step 3** Calculate "Lower" values, interpolated forces and masses at pseudo node point on opposite surface.
- **Step 4** Modify the accelerations of the nodes.

  After all nodes have been moved, but before internal energy or pressure are updated:

- **Step 5** Slave side nodes are "put-back-on" to master side line segments, and their normal velocity modified to match the interpolated master side velocity. Step 5 is thus the only non-symmetric part of the algorithm.

4.1.1 Step 1

The tangents and normals are calculated using a "long" measure, the nodes either side defining the tangent and normal, rather than the half side points, to suppress instability. Unit vectors for normals and tangents, based on half time step node positions, are stored for all slide nodes, and slide element faces (see Figure 4.1). This is a profligate use of memory, as either tangent or normal can be derived easily from

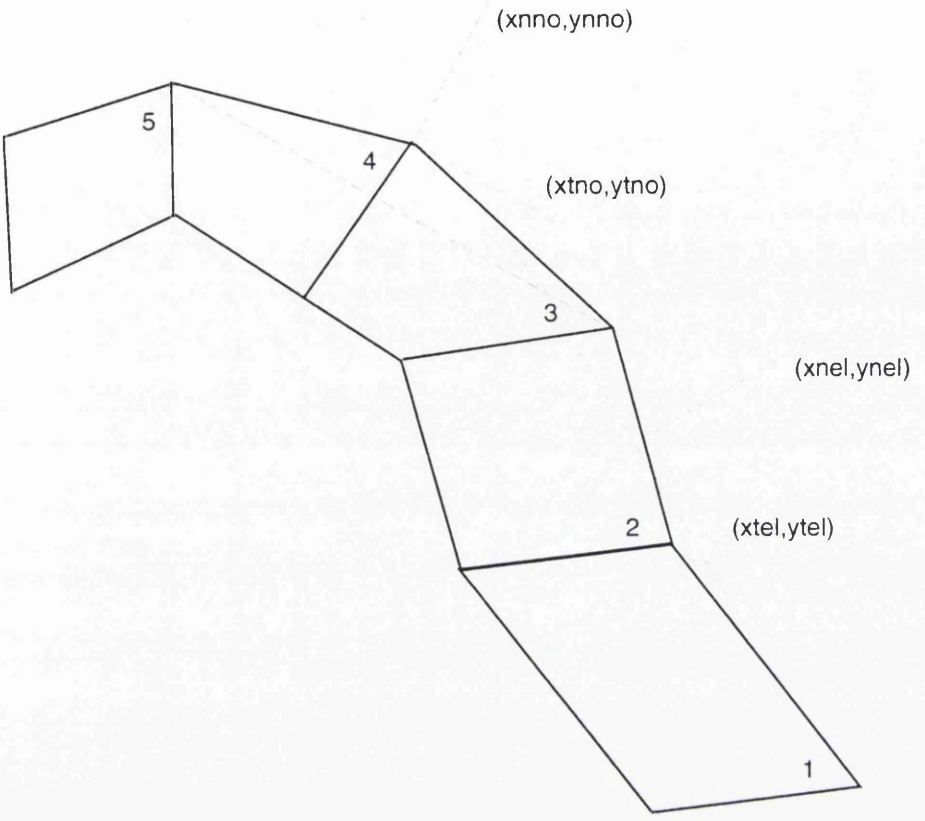


Figure 4.1: Tangent and normal construction.

the other, but it is justified because the numbers of slide nodes is always small compared with the total number of nodes and it makes the coding much easier to read. If the algorithm was to be extended to 3D, memory would become more of an issue. At this point a length scale representing the effective area over which the pressure at each node acts normal to the free surface is calculated. Later the pressure force from the opposite surface will be applied to the node as if to an element side of this length.

4.1.2 Step 2

The accelerations for all nodes for the current timestep have been calculated from the element contributions, and boundary conditions applied, so that a free surface acceleration is known for each node on the slide lines. The tangential component of this acceleration is used to calculate a half-time step free surface tangential velocity component. For the normal component, the forces above and below the surface are required so that they can be summed to form the total force (and hence acceleration) of the node in the normal direction. This is equivalent to producing

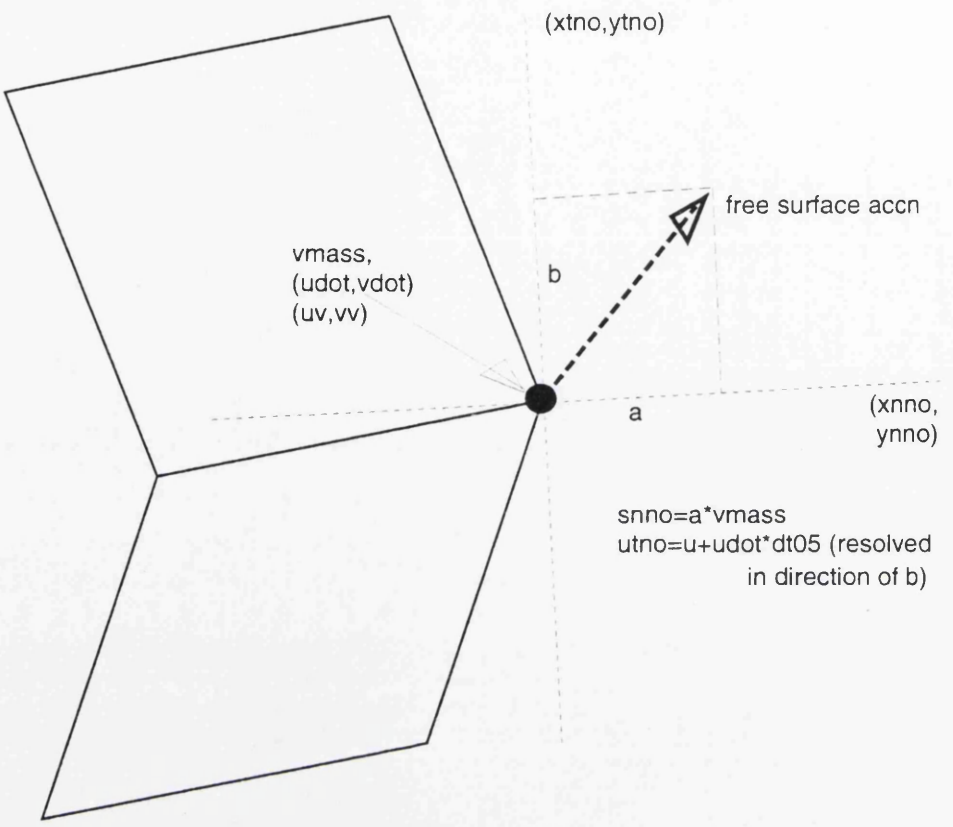


Figure 4.2: Free surface tangential velocity.

pseudo cell contributions to a nodal acceleration. The contribution could be calculated by resolving the element pressure and stress contributions, but a simple and robust way of achieving the required result is to resolve the free surface acceleration in the direction of the normal and divide by the nodal mass, to produce the equivalent normal pressure (see Figure 4.2).

4.1.3 Step 3

Having derived the "upper" nodal quantities, that is, those that can be calculated from the elements containing the node, the "lower" quantities must be interpolated to allow the opposite surface to influence the normal motion (but not the tangential). The terminology "upper" and "lower" is used as the same calculations are performed, whether it is a master side affected by its slave, or vice versa. For each node on the upper surface, the lower surface is searched for the pair of nodes on the lower side which bracket it. This search is speeded up by the knowledge that the lists are of contiguous nodes, and by "remembering" which node-pair were closest last time, and only searching a small part of the lower list. The lower quantities,

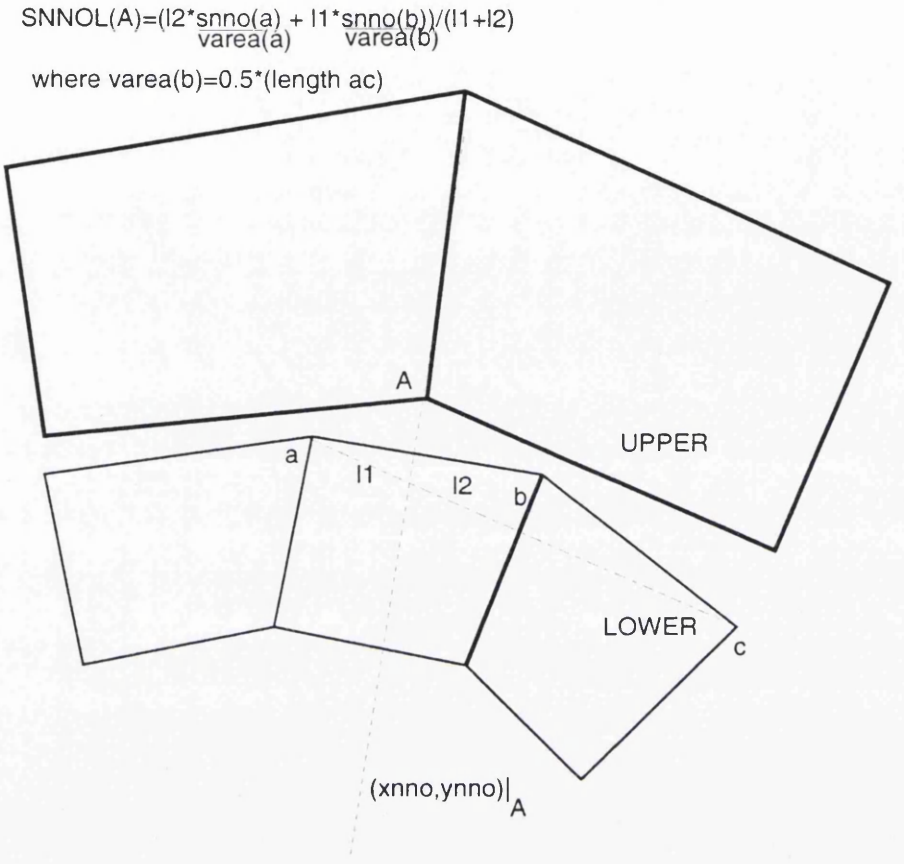


Figure 4.3: Lower interpolation.

effective normal pressure and effective nodal mass are interpolated linearly from the intersection of the upper normal with the lower node pair as shown in Figure 4.3. Note that these are forces and mass per unit lengthscale. The routine is called twice for each slide line pair, master-slave and slave-master.

4.1.4 Step 4

For each node, the new normal acceleration is calculated from the upper and lower effective pressures and the effective nodal mass. This is added to the tangential component of the free surface acceleration, modified by averaging with the acceleration of the inner neighbour nodes. If the actual free surface tangential acceleration is used, the elements at the interface tend to skew. CORVUS uses an average of the nodes acceleration with that of the inner neighbour nodes to overcome this as shown in Figure 4.4.

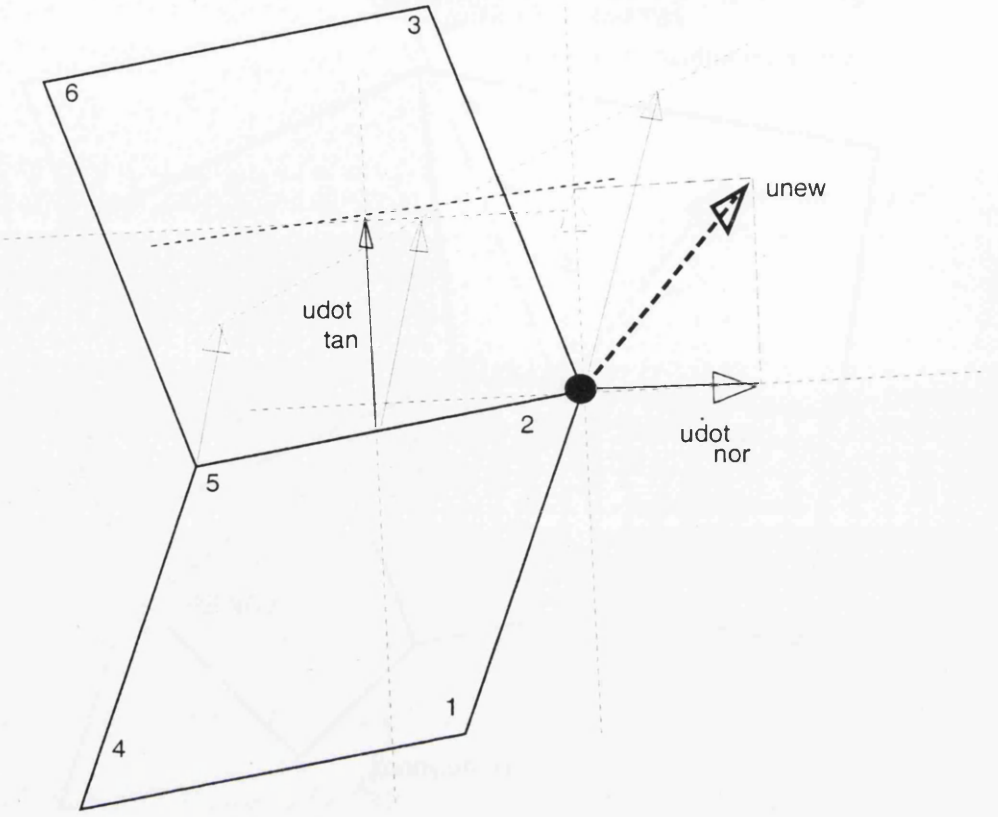


Figure 4.4: Normal acceleration calculation.

4.1.5 Step 5

The put-back-on step is the most unsatisfactory phase of master/slave slide line algorithms. It is asymmetric and arbitrary, and experience has shown that it to tends to introduce zero energy modes (hourglass effects) in bilinear element solutions. In the CORVUS slide put back, slave nodes are made to lie on the master slide line segments by moving them along the slave side normal, and their normal velocity is altered so that it matches the interpolated master node velocity, resolved in the slave normal direction (see Figure 4.5). There remains a small discrepancy between the normal velocity of the master and slave surfaces, due to the difference between the normals - but there is no "right" way to measure the normals for the piece-wise linear line segments sets that form the two surfaces.

In some ways the approximations used in the old slide are better than those used in the new algorithm. The evaluation of the pressure field imposed on the master surface by the slave material is more accurately treated in old slide - each slave element imparts a contribution derived from a linear shape function in a manner completely consistent with the element contributions to internal nodes. In comparison, the method in new slide, of taking the closest element as being representative

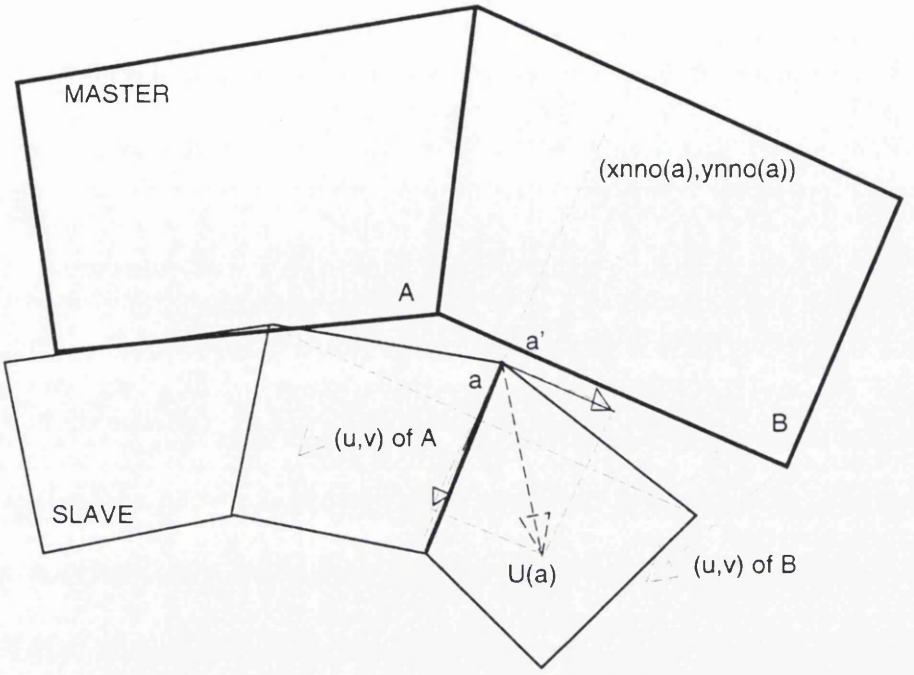


Figure 4.5: Slave node put-back-on.

of the pressure field is crude, particularly where there is a significant disparity in element sides. However, it is this circumstance that makes the old slide method unworkable for a symmetric treatment, as the summation of multiple contributions requires the slave side to be meshed finer than the master, and the treatment cannot be reversed to allow the "coarse" master to influence the "fine" slave.

The accuracy of new slide algorithm has been assessed on a range of problems and it has been concluded that whilst both methods converge to the correct solution, new slide converges slower and is thus less accurate on a coarser mesh.

4.2 Void Closure

Most published void closure algorithms such as those in HEMP and DYNA betray their origins as subtle variations on slide algorithms where the sliding surfaces are treated as free surfaces until some geometric closure criterion is satisfied. At that point, various adjustments are made to the velocities and positions of nodes to satisfy continuity conditions, particularly momentum. Void re-opening is similarly ad hoc, HEMP for example allows a node to reopen if it separates from the master surface by 10 % of its cell size - a measure that is dependent on mesh resolution and time step. With the knowledge that all most codes did was so approximate, it made sense to spend some time considering the problem of void closure from a physical

viewpoint. One aspect of this was to consider the Rankine-Hugoniot conditions at a planar impact. The other was to consider what variable should be conserved at impact.

The instantaneous conservation of momentum on impact, implied by the immediate adjustment of node velocities to transfer momentum to the impacted material, actually implies that all void closure is inelastic. Whilst this assumption may be reasonable for modelling the impact of a car with a wall, whether it can be justified for problems involving significantly higher velocities is more questionable. In fact, on impact, some of the kinetic energy is converted to internal energy through the compression of material, and one should be wary of adjusting velocities to conserve momentum without also checking that the energy balance of the system has not been changed. The question that arises then is "what is the correct velocity to impose at a closed boundary?", and the answer comes from the Rankine-Hugoniot conditions.

A simple plate impact problem was set up, and some calculations performed with a variety of simple models for the boundary velocity at closure. Particular attention was paid to the velocity profile which established itself over $\approx 10\mu s$, the profile of internal energy, and the conservation of total energy in the system. The problem definition and analytic solution is given below in Figure 4.6.

The velocity of the impact surface was hard-coded to represent various options - bringing the surface instantaneously to rest, accelerating it to the impact velocity, adjusting it to conserve momentum, and adjusting it to satisfy the analytic solution. The results from all these models were broadly in agreement with the analytic solution, and with the same meshing, total energy was conserved to similar accuracy in all cases. In fact, the key to obtaining a reasonable solution was to use a sufficiently fine mesh, and this result is easy to understand if you consider Figure 4.7.

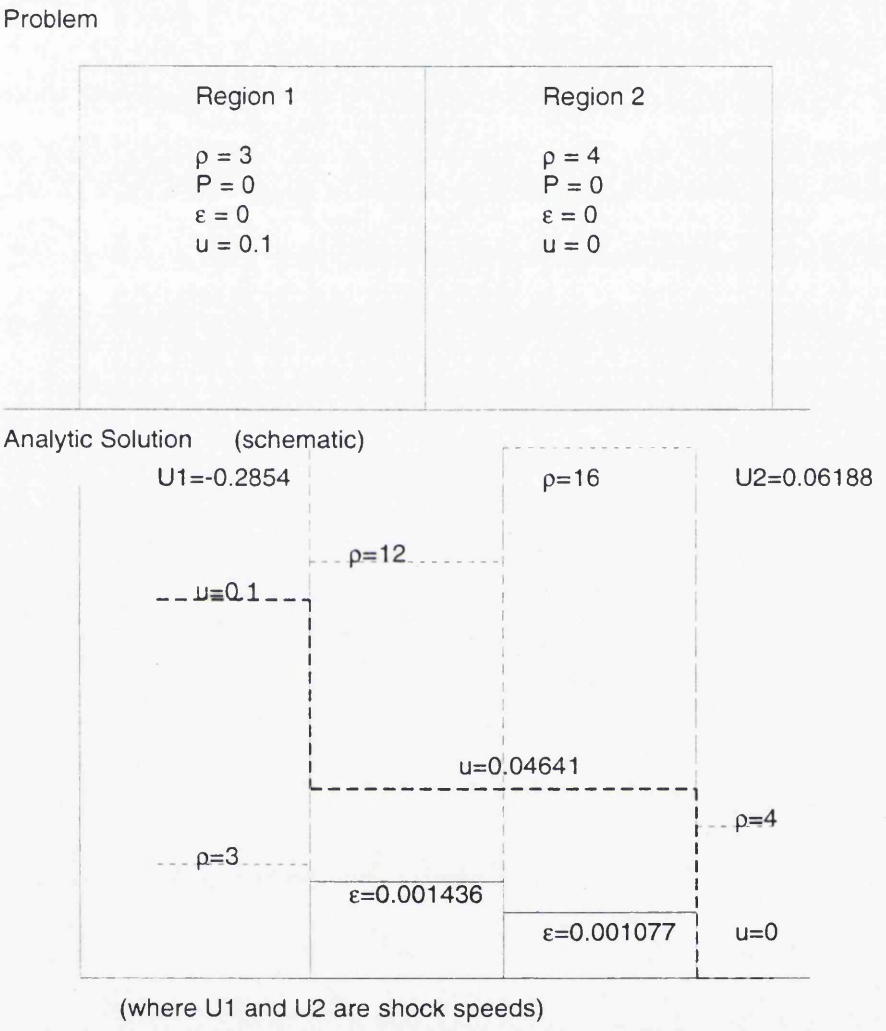


Figure 4.6: Simple impact problem and analytic solution.

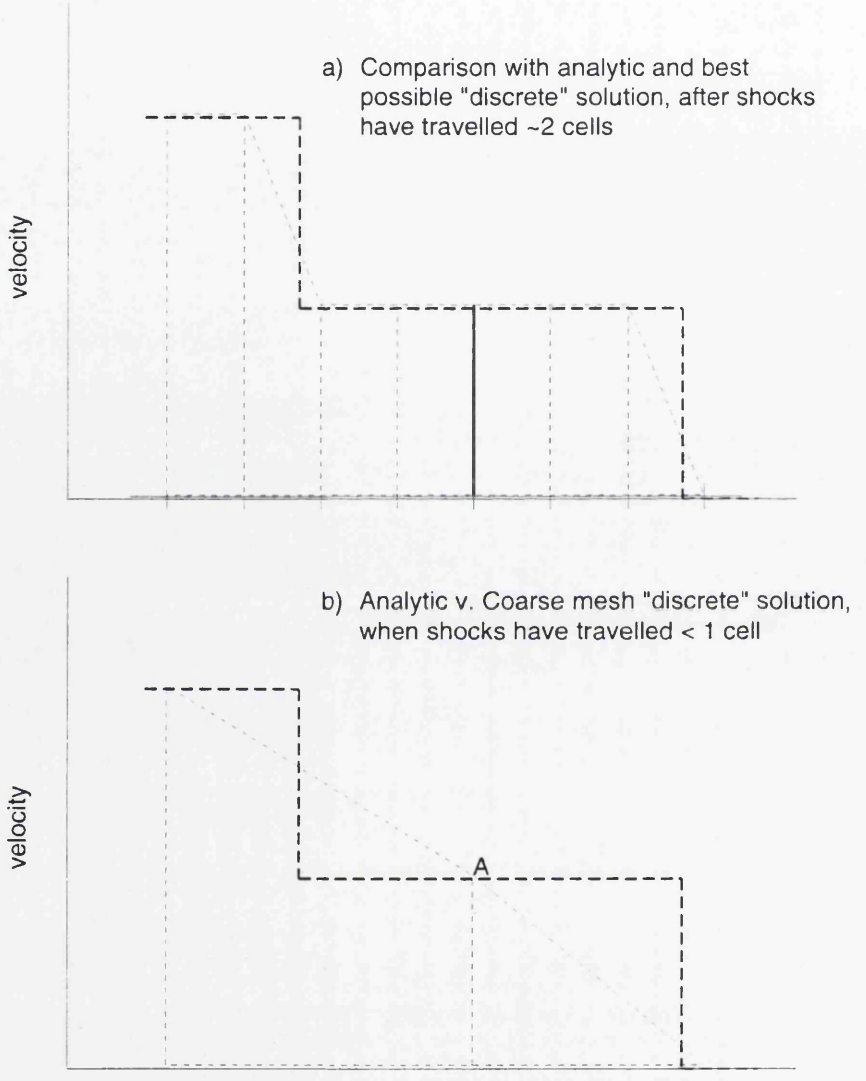


Figure 4.7: Effect of mesh resolution on momentum summation.

Figure 4.7a) shows how once the profile is established a discrete solution can, potentially, provide a reasonable approximation of the total momentum in the system, except that the representation of a shock is a ramp of one cell length, and not a step function. In practise, the shock is smeared over two or three cells, which means that the effect is exaggerated further. The shaded area is proportional to the total momentum in the system, as summed for an output print, or for instantaneous momentum conservation (assuming the elements are mass-matched). This looks to be a reasonable approximation to the area under the analytic line for an established profile, but one needs to consider the situation at impact, before the shocks have travelled a full cell length in either direction Figure 4.7b). At impact, the "best" solution, where every node has the analytic velocity, may not conserve total momentum as summed, depending on the shock speeds, the time step and the lengths of the elements. In fact, to conserve momentum the interface velocity A could be anywhere between the two unshocked velocities, depending on these other factors.

If momentum conservation is not a practicable or meaningful measure for void closure, then the alternative of kinetic energy conservation is similarly flawed, as the step function is now approximated by a quadratic curve, which may provide a close approximation to the analytic solution or not depending on the same criteria as for momentum.

4.2.1 Void Closure in CORVUS

Having analysed void closure algorithms and concluded that existing methods had no especial justification in physics, some different approaches were tried. The first idea was to calculate the normal velocity at impact from the Rankine-Hugoniot equations and apply this to master and slave surfaces - the idea being that the interface would move with the correct velocity from impact onwards. The two difficulties with this are the expense of the calculation, and the problem of enforcing a changed velocity on master nodes. The master nodes do not necessarily align with slave nodes, and adjusting their velocity affects the closer of neighbouring nodes, so it is very difficult to define a scheme which does not involve iteration. The conclusion drawn from these experiments was that a method which only adjusted slave node velocities and positions at closure time would be easiest to implement, and also that some effort should be made to conserve the kinetic energy "lost" in the adjustment of nodal velocities. It is also clear that void closure can only be modelled with reasonable accuracy if there is sufficient mesh resolution to model the shocks caused at impact - this implies a fine mesh has to be used. Given that a fine mesh will be used, most ad hoc methods will result in the correct velocity profile being established in a few (short) time steps, i.e. a simple and cheap method will be accurate enough, provided it is robust.

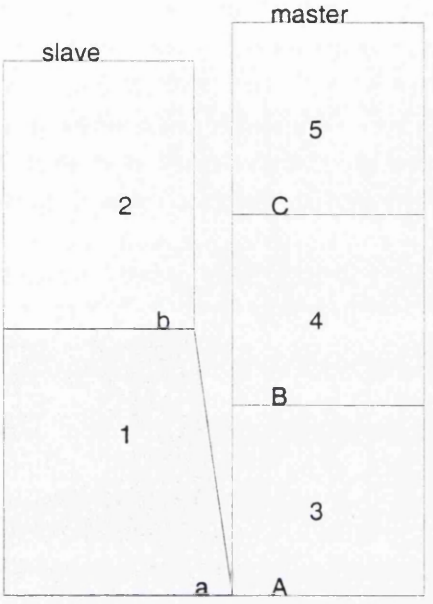
The simplest method of coding void closure is to allow a free slave surface to

impact a master surface, and move each node onto the master surface as is penetrates the master material. The normal velocity of the slave node has to be adjusted to match the interpolated normal velocity of the master line segment (consistently using the slave side normal to avoid couples, as with standard slide put-back-on). There is then the question of whether to correct for the reduction in kinetic energy resulting from changing the slave node velocity. The natural assumption is that something must be done, but it is not obvious that this is the case. First, the summation of kinetic energy has been shown to be strongly affected by the discretization, and second, since fine meshes are required, the amount of energy lost from the system is very small. Despite these considerations, CORVUS void closure has been coded with the "lost" kinetic energy thrown back into the slave material internal energy, although test problems show negligible difference between this approach and losing the energy.

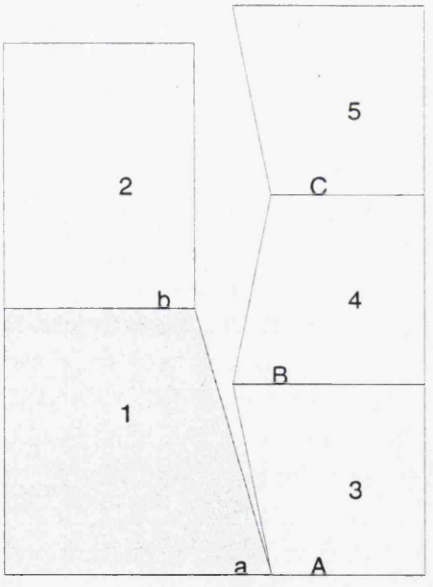
This method has now been in use for a number of years and in general works pretty well unless the "lost" kinetic energy becomes too large. This suggests it may be better either to allow the energy to be lost and simply keep track of the error introduced, or apply a different method for calculating normal velocity after the impact which minimises the kinetic energy error. A further robustness problem has been encountered with oblique impacts where a jet forms ahead of the closure. In such cases the unit vector normal which is used to detect the time and point of closure can predict a spurious early impact away from the true impact point. The latter problem could probably be overcome by using the intersection of the free surface trajectory vector with the target surface rather than the unit vector normal to detect closure.

4.3 Void Opening

Void opening was initially coded as the antithesis of the void closure criteria - the position of the (closed) slave node was calculated with normal contributions from the master surface, and if, at the end of the time step, the node had separated from the master surface, the node was treated as "open" and behaved as a free surface from then on. In practise, nodes would close, reopen and reclose in consecutive time steps, and this "bouncing" would continue for many time steps. The problem was made worse by the zero-energy mode (hourglass) problem caused by the gradual closure of a void by a tangential pressure wave (Figure 4.8).



Step 1: +ve pressure in 1 drives node a toward element 3



Step 2: +ve pressure in 3 drives node B toward element 1, pulling 4 into tension. Node C pulls away from slave surface, and "hourglass" propagates ahead of closure.

Figure 4.8: Zero energy tangential closing of void.

The criteria for reopening was clearly too lax, and a variety of palliatives were tried. Nodes were prevented from reopening until several time steps after they had first closed, but it was found that the number of time steps required was problem dependent. Instead, nodes were "put-back-on" unless they had separated by a percentage of the element size - again the percentage required was found to be problem dependent, and if a very short time step was being enforced from elsewhere in the problem the nodes might never reopen. The conclusion drawn was that methods based on highly localised measures such as slave node velocity are flawed.

What was needed was a measure of whether the interface was in tension, but there is no true measure of interface pressure available. Instead, the state of the material either side of the interface is considered, and a node is only allowed to open if both sides are in tension, or at least at zero pressure. This is a relatively crude approximation to a physical reality, but is easier to justify than the recipes based on the movement of individual nodes. It has been tested on a variety of problems and is quite robust. It is not sensitive to the hourglass modes that void closure can introduce, and in fact tends to oppose them, acting as a damping influence. As currently coded, slide surfaces in CORVUS can either close and open, or remain closed for all time.

4.4 Bouncing ball test problem

In order to test the void closure and void opening algorithms the following test problem was calculated. A stainless steel ball bearing is dropped from a height of 2m onto a thick steel plate, it first deforms, then rebounds. Although the low stresses and small deformations put this problem into a different regime than the high strain rate, high distortion problems normally of interest at AWE, it does present a challenging problem for the void closure algorithm.

The problem is modelled by defining a 1cm radius sphere with an initial velocity of $0.00626 cm/\mu s$, equivalent to dropping from 2m, toward a steel plate 4cm thick with a rigid rear boundary. The sphere is initially 0.01cm from the plate, and so flies nearly $16\mu s$ before the first impact. By $200\mu s$ the ball has rebounded clear of the plate, and is travelling with a near uniform velocity, from which the height of the bounce can be calculated, and hence the coefficient of restitution.

Figure 4.9 shows four frames from the calculation, colour coded such that black represents the negative initial velocity (- $0.00626 cm/\mu s$), and white a positive velocity slightly higher than the rebound speed of the ball ($0.004 cm/\mu s$). The times shown are 0, 30, 55 and $125\mu s$. The small amount of distortion is evident.

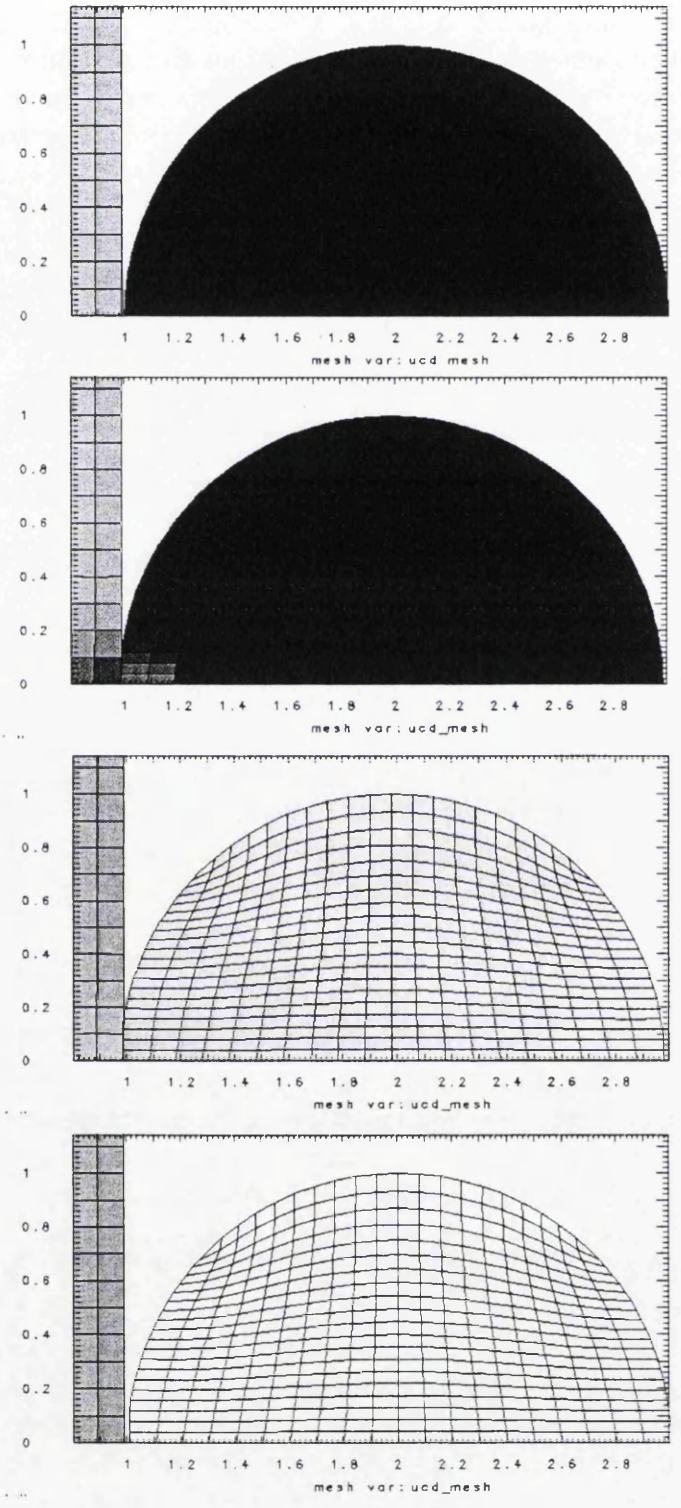


Figure 4.9: Bouncing ball test problem at 0, 30, 55 and 125 μs.

The calculated bounce speed of $0.00365 cm/\mu s$ would imply a coefficient of restitution of 0.34. Intuitively, this seemed low, and the numerical output from the calculation implied significant amounts of internal energy had been created, leading to the fear that energy was being dissipated by a numerical process. However, closer examination of the graphical output showed that there was very little internal energy in the system, and the high total in the numerical output is therefore attributable to elastic distortional energy, which is thrown into the regional summation. To satisfy the unanswered question as to whether 0.34 is a reasonable value for the coefficient of restitution, a simple experiment was carried out which gave values of between 0.25 and 0.30 for a similar system.

4.5 Friction

The CORVUS slide algorithm discussed above assumes that there is no tangential force acting across material interfaces (zero friction). Maximum friction can also be represented by a merged or locked interface, where the nodes on the interface are treated just like internal nodes. If one of the interface materials has no strength then the frictionless slide treatment is probably the best treatment. However, some intermediate level of friction is probably the best description when both materials are strong, but this physics is not well understood [47]. The actual physical processes responsible to dynamic friction are probably very different to those of classical friction. Dynamic friction may for example result from the formation of a boundary layer in the weaker material, where the plastic work is focussed reducing the yield strength within the boundary layer. However, there are many physical processes including the movement of dislocations, adiabatic shear, material mixing and shock welding, which may be important [47].

In order to try to represent the effect of such processes, the CORVUS slide algorithm has been modified to introduce a simple friction model [48]. This can be used to make engineering assessments of the importance of dynamic friction in applications and is being used to design dynamic friction experiments [52]. Once the data from the latter experiments is available it will be used to normalise the simple model, and ultimately contribute to the development of a predictive physics-based dynamic friction model.

In developing the current simple friction model it was considered important that the friction model should always produce frictional forces which are bounded by the limiting cases of frictionless slide and a locked or merged interface. The model should also be as free as possible of mesh dependence and introduce the minimum of new free parameters. There should also be sufficient sensitivity in the model to suggest that the fit to experimental data can be improved. The model should however also be based on physics and not just be a fit to a series of experimental data points. The simple model that is described here achieves some of these objectives

and is certainly an improvement over a merged interface, but is really just a first step. It still shows some mesh dependence and is not in anyway built on physics.

The simple model contains three simple empirical forms for the frictional force which can be applied in isolation or in a linear combination. They are however known to fit observation within certain regimes. The first term is believed to be most applicable for metals at low velocities, a simple example being a metal block initially at rest on top of a metal plate, which is gradually inclined. At some critical angle the block starts to slide. In such a situation the block is initially held in place by friction. The frictional force F for such a situation has been observed to be proportional to the normal force at the interface. The other two terms express the observation that friction at high velocities has been observed to increase as the velocity difference between the surfaces increases. The two forms express linear and square law dependence on the difference in tangential velocities of the two surfaces.

$$F = \alpha_1 a_n m + \alpha_2 (u_{tr} - u_{tp})A + \alpha_3 (u_{tr} - u_{tp})^2 A \tag{4.1}$$

where α_i are the friction coefficients, a_n the normal acceleration of the node, m the mass of the node, u_{tr} is the tangential velocity component for the real node, u_{tp} is the tangential velocity component for its pseudo node on the opposite surface and A is the effective area associated with the real node.

In order to introduce a frictional force, the slide package has been modified first to calculate two tangential accelerations for each slide node assuming a true slip or zero friction boundary condition and a merged or locked boundary condition. The latter is achieved by calculating tangential forces required to match the free surface tangential accelerations for each real node, then interpolating tangential forces for each pseudo node, using the frictionless slide procedure for interpolating normal forces. A tangential acceleration for the merged boundary condition can then be calculated by summing the tangential force from the real node and its partner pseudo node, and applying it to the combined mass. This is again analogous to the procedure described above for calculating the modified normal accelerations in the original frictionless slide algorithm.

An effective frictional force for the merged boundary condition is then calculated from the difference in tangential accelerations for the two boundary conditions. The frictional force is then calculated for the nodes on one of the pair of slide surfaces using (4.1), but it is limited to not exceed the merged frictional force and its direction is set to act in the same direction as the merged force would. In order to ensure the algorithm is conservative, the forces are calculated directly for one side, then mapped across onto the opposite surface, and applied as a reaction.

The frictional forces obtained on both sides are then used to modify the tangential accelerations calculated with a slip or zero friction boundary condition. These tangential accelerations are then simply recombined with the normal accelerations calculated from the frictionless slide procedure.

Chapter 5

Mesh Adaption

5.1 Mesh movement

Mesh movement for the adaptive multi-material ALE algorithm has four key ingredients boundary node movement, node movement constraints, nodal weights and internal node movement. How these options are combined and controlled is currently under user control. This puts some emphasis on users learning how best to use these mesh movement algorithms. These choices could be automated, thus removing this level of control from the user. However, the author believes that it is better for the users to retain control, both to enable them to devise the best adaption strategy for a given problem, and also to encourage intelligent use of the code.

There are however three clear types of problem that can be identified where different styles of mesh movement are more appropriate. It is worth discussing these briefly before giving details of the mesh movement algorithms that have been developed. It is hoped this will enable the reader to obtain an impression of the flexibility of the adaptive multi-material ALE algorithm and the different ways it can be applied. This should also give an appreciation of why the author does not favour fully automating the mesh movement options.

Probably the most common use of the adaptive multi-material ALE agorithm is for problems containing a limited number of materials undergoing severe deformation. This deformation prevents such problems from running to completion with a pure Lagrangian treatment. However, there are clear benefits to be obtained from applying Lagrangian mesh motion to as many materials as possible. In such problems the best approach is to apply ALE mesh relaxation only to the materials undergoing severe distortion. Equipotential mesh relaxation is well suited to this type of task, provided the boundary nodes are well placed, as they will have a strong influence over the quality of the internal mesh generated. A number of mesh movement algorithms have been developed by the author for boundary nodes, in order to ensure that these nodes are well placed before the mesh relaxation procedure is applied to the internal nodes. Node movement limits are also useful for this type

of problem, as they can be used to delay mesh relaxation in these ALEing regions and so retain some Lagrangian character to the mesh.

There are also a number of ways to use ALE mesh adaption to introduce a limited degree of local mesh refinement for features of particular interest in a simulation. These features could be a particular material, material interfaces or shock fronts for example. The refinement is achieved by applying mesh movement weights to the nodes of elements that are contained within the feature of interest at the start of each advection step. This weighted mesh movement leads to mesh lines being pulled from the lower to higher weight regions, with the resolution difference being proportional to the weight difference applied. This approach is very useful for problems with a limited number of features of interest.

The final class are problems that are almost suitable to run pure Lagrangian. In this case local ALE mesh adaption can be beneficial. Problems of this type may well run through to completion pure Lagrangian, but be slow to complete due to low time steps. The Lagrangian mesh in this case is usually important for the whole domain. In this case the best approach may be to employ a series of local element quality measures. These quality measures are simply calculated at the end of each Lagrangian step and if they are outside of user defined tolerances the nodes of that element are flagged for mesh relaxation.

The user applies these algorithms by first defining a series of super regions. These super regions consist of a number of adjacent regions or mesh blocks. Mesh relaxation is only allowed across material interfaces between blocks that are in the same super region. The super region concept also helps the code user to determine the best mesh movement strategy to use by breaking the problem down into constructing a mesh movement strategy suitable for each super region. Different mesh movement algorithms can be applied to the individual regions within a super region provided they are compatible.

5.1.1 Interface and boundary node movement algorithms

The quality of an ALE mesh does not just depend on the mesh movement algorithm applied to internal nodes, but is also strongly dependent on the boundary conditions applied. In the adaptive multi-material ALE algorithm, the boundary conditions for mesh movement algorithms simply consist of the positions of nodes on the boundary of each super ALE region, and significant improvements in mesh quality have been obtained by redistributing the boundary nodes before applying a global mesh movement algorithm to the internal nodes.

Two options are available for distributing nodes along reflecting boundaries. The first simply positions each boundary node to lie halfway between its two immediate neighbours on the boundary. This is useful for a limited number of problems, where spherical symmetry is not an issue. The preferred and most general treatment

which is suitable for problems where symmetry is important, is to generate symmetry nodes and also apply the internal mesh movement algorithm to the boundary nodes.

If a boundary is a Lagrangian slide line then the nodes on one of the pair of slide surfaces can be redistributed along the other surface. This is a very useful technique as it improves robustness and mesh quality, helps to retain some Lagrangian character to the mesh, and allows sliding interfaces to be retained to much later problem times. Many different ways of redistributing the slide nodes have been considered. Three simple options were coded first, redistributing the nodes to positions of equal angular spacing about some centre of convergence, equal x and equal y from their two immediate neighbours. The former has been very useful for highly convergent problems, such as ICF capsules where ignition often depends on the subtle focussing of interacting shock waves. The equi-distribution of nodes in x or y coordinate directions has also been found to be useful for plane geometry experiments. Such experiments often involve the use of finite explosive charges driving finite metal plates. These metal plates often become highly distorted during such experiments, due to the influence of edge effects reducing the drive near the charge boundaries.

Although these simple methods have been very useful and are still in frequent use they are not sufficiently general. It is undesirable for users to have to sort through a vast array of options if a single option can be devised that will handle all or most of these situations. Also it is not uncommon to encounter more complicated engineering and physics problems where one of these options may be the best choice for one section of a material interface, but a different redistribution method may be more suitable for the rest of the interface. In order to address this problem an option to redistribute nodes to positions of equal arc length from their two immediated neighbours has been developed. This has been found to be the most generally applicable interface redistribution method. However, it should be noted that if small wavelength (relative to the cell size) jets form and continue to grow then choosing the best of the three simple methods to match the dominant symmetry in the problem will still offer advantages in terms of robustness.

The momentum advection of slide nodes is not as accurate as that of internal nodes. It is forced to drop to first order accuracy normal to the interface, as a velocity slopes cannot easily be formed across a sliding interface. The node movement along the piecewise linear slide surface also lead to some inconsistencies in how the nodal control volumes used for the momentum advection. Given these considerations, and the freedom to choose between re-distributing master or slave nodes, it is always best to redistribute the slave nodes if possible. This is because the master/slave approach employed by the CORVUS slide algorithm and most other published slide algorithms puts the emphasis on calculating the master surface as accurately as possible, with the slave normal velocity components being interpolated from the master after the put-back step. Therefore any errors introduced by

advecting slave nodes should have little influence on the accuracy of the interface treatment.

A further option has also been added to further minimise these errors by minimising the number of time steps when nodes are redistributed in this way. This option provides a simple error estimator to control whether the nodes should be redistributed or not. At each time step the mean arc length is first evaluated for the slide line in question. The maximum and minimum arc length are then determined. If the maximum exceeds 1.5 times the mean arc length, or the minimum, is less than 0.5 times the mean arc length, then all the nodes on the slide surface are redistributed to positions of equal arc length from their two immediate neighbours.

Another very useful slide node movement option which is included in CORVUS, and is an important part of the adaptive multi-material ALE algorithm, repositions each slide node on a slide surface to be coincident with its corresponding slide node on its partnering slide surface. This option is currently limited to pairs of slide surfaces which both have the same number of slide nodes, hence forcing the same mesh resolution to be employed for the materials on either side of the surface. However, the strength of this simple approach is that it allows the slide surface to be merged at end stage during a calculation. This means that calculations can be performed allowing slip between two surfaces, until later stages in the problem, when the interface is undergoing too much deformation for the calculation to be continued with a Lagrangian interface. The sliding interface is then merged and multi-material ALE mesh relaxation is allowed across the interface to enable the calculation to continue robustly to completion.

Currently it is not possible to redistribute nodes on free boundaries. However, this may be a useful future extension and could be implemented by creating a slide like data structure for the free surface nodes. The free nodes could then be repositioned along their original profile, with their normal velocity interpolated from the values of the pair of original nodes bracketing their new position. It may also be advantageous to reposition nodes along merged Lagrangian interfaces. In principal this can be done by redefining the interface as a slide line and then using the friction model discussed later in this thesis to represent a merged interface. Any of the slide line movement options could then be applied.

5.1.2 Mesh movement constraints

A number of mesh movement constraints have been found to be useful to increase the Lagrangian character of the mesh, which should in turn lead to improved solution accuracy. These include an optional ALE on time for each region and a series of cell based criteria, which must be satisfied before the nodes of an element can be relaxed by the mesh movement algorithms.

These criteria include restricting the nodes of high explosive (HE) cells from being relaxed, until they have fully detonated. In addition to improving grid quality,

this also avoids the need to advect burn times and burn intervals for programmed burn. It also offers greater accuracy, in that it improves the resolution of the detonation front, and avoids numerical diffusion across this front.

A similar approach has also been taken for porous materials, which are normally treated with a simple locking or snow plough model. The code stores the initial density of the porous material and the solid density of the matrix material. The equation of state is then fixed to return zero pressure until the solid density is reached. The nodes of these porous elements are constrained from relaxing until solid density is reached. This also removes the problem of how to advect the history variable that denotes whether a cell is still porous or not.

A further constraint that has been found useful is to restrict nodes in ALEing regions from being relaxed until they have a non-zero Lagrangian velocity, denoting that a physical signal such as a shock wave has reached this point.

The amount of mesh movement or degree of mesh relaxation once a node is allowed to relax, can also be controlled. There are two controls for this. The first is simply the number of iterations that mesh movement algorithm is allowed to perform for each time step. This is applicable to both Winslow's scheme, and the simple averaging of immediate neighbours positions. In most problems one iteration is sufficient and retains a little of the Lagrangian character to the mesh. However, some problems require more iterations for parts of the problem to run robustly. There is also a region dependent acceleration parameter for both these mesh movement options. This is set to 1.0 by default, which leaves the mesh movement unchanged. If the acceleration parameter is reduced to 0.0, then pure Lagrangian mesh motion is obtained. However, most problems benefit from a fairly low value such as 0.25. This increases the Lagrangian influence on the mesh movement significantly and produces what can best be described as a smooth Lagrangian mesh.

It is also possible to introduce adaptivity in time as well as space, by performing an advection step, only if the current time step is lower than that of the previous step. This strategy constantly reintroduces Lagrangian character into the mesh whenever an advection step is not required, and so generally produces robust meshes with more Lagrangian character than would be produced by simply using a low value for the acceleration parameter. Overall this also improves computational efficiency, as the advection step is about three times as expensive as the Lagrangian step, but clearly the time steps required will in general be lower, so there is some trade off.

5.1.3 Mesh movement algorithms for internal nodes

Equipotential zoning was first developed by Crowley [53] for the automatic construction of two dimensional meshes. Equipotential mesh generation is best described by considering a structured mesh, constructed from two sets of nonorthogonal but continuous mesh lines, one for each of the logical mesh coordinates. These

mesh lines are assigned two sets of potentials ϕ and ψ, one for each mesh coordinate. A smooth mesh is then obtained if these potentials satisfy Laplace's equation in x y coordinates:

$$\begin{aligned} \nabla^2\phi &= 0 \\ \nabla^2\psi &= 0 \end{aligned} \tag{5.1}$$

along with suitable boundary conditions.

Equations (5.1) had been suggested before as a basis for mesh generation. But Crowley made the contribution of inverting the equations to obtain differential equations for x and y as functions of ϕ and ψ. Winslow [54, 55] later showed that the inverse differential equations produced by Crowley were not the simplest form and that the correct inverse equations are

$$\alpha\frac{\partial^2 x}{\partial\phi^2} - 2\beta\frac{\partial^2 x}{\partial\phi\psi} + \gamma\frac{\partial^2 x}{\partial\psi^2} = 0 \tag{5.2}$$

$$\alpha\frac{\partial^2 y}{\partial\phi^2} - 2\beta\frac{\partial^2 y}{\partial\phi\psi} + \gamma\frac{\partial^2 y}{\partial\psi^2} = 0 \tag{5.3}$$

where

$$\alpha = \left(\frac{\partial x}{\partial\psi}\right)^2 + \left(\frac{\partial y}{\partial\psi}\right)^2 \tag{5.4}$$

$$\beta = \frac{\partial x}{\partial\psi}\frac{\partial x}{\partial\phi} + \frac{\partial y}{\partial\psi}\frac{\partial y}{\partial\phi} \tag{5.5}$$

$$\gamma = \left(\frac{\partial x}{\partial\phi}\right)^2 + \left(\frac{\partial y}{\partial\phi}\right)^2 \tag{5.6}$$

and the Jacobian of the transformation $J = \frac{\partial x}{\partial\phi}\frac{\partial y}{\partial\psi} - \frac{\partial x}{\partial\psi}\frac{\partial y}{\partial\phi} \neq 0$.

These equations can be discretized by making the following finite difference approximations

$$\frac{\partial^2 x}{\partial\phi^2} = x_{\phi-1,\psi} - 2x_{\phi,\psi} + x_{\phi+1,\psi} \tag{5.7}$$

$$\frac{\partial^2 y}{\partial\phi^2} = y_{\phi-1,\psi} - 2y_{\phi,\psi} + y_{\phi+1,\psi} \tag{5.8}$$

$$\frac{\partial^2 x}{\partial \psi^2} = x_{\phi,\psi-1} - 2x_{\phi,\psi} + x_{\phi,\psi+1} \tag{5.9}$$

$$\frac{\partial^2 y}{\partial \psi^2} = y_{\phi,\psi-1} - 2y_{\phi,\psi} + y_{\phi,\psi+1} \tag{5.10}$$

$$\frac{\partial^2 x}{\partial \phi \partial \psi} = \frac{1}{4}\left(-x_{\phi+1,\psi-1} + x_{\phi+1,\psi+1} - x_{\phi-1,\psi+1} + x_{\phi-1,\psi-1}\right) \tag{5.11}$$

$$\frac{\partial^2 y}{\phi \partial \psi} = \frac{1}{4}\left(-y_{\phi+1,\psi-1} + y_{\phi+1,\psi+1} - y_{\phi-1,\psi+1} + y_{\phi-1,\psi-1}\right) \tag{5.12}$$

and defining α, β and γ to be:

$$\begin{aligned}
\alpha_{\phi,\psi} &= \left(x_{\phi,\psi+1} - x_{\phi,\psi-1}\right)^2 + \left(y_{\phi,\psi+1} - y_{\phi,\psi-1}\right)^2 \\
\beta_{\phi,\psi} &= \frac{1}{2}\left(\left(x_{\phi,\psi+1} - x_{\phi,\psi-1}\right)\left(x_{\phi+1,\psi} - x_{\phi-1,\psi}\right)\right. \\
&+ \left.\left(y_{\phi,\psi+1} - y_{\phi,\psi-1}\right)\left(y_{\phi+1,\psi} - y_{\phi-1,\psi}\right)\right) \\
\gamma_{\phi,\psi} &= \left(x_{\phi+1,\psi} - x_{\phi-1,\psi}\right)^2 + \left(y_{\phi+1,\psi} - y_{\phi-1,\psi}\right)^2
\end{aligned} \tag{5.13}$$

Then substituting these definitions and approximations into (5.2,5.3) and solving the resulting equations for $x_{\phi,\psi}$ and $y_{\phi,\psi}$ gives:

$$\begin{aligned}
x_{\phi,\psi} &= \frac{1}{2(\alpha_{\phi,\psi} + \gamma_{\phi,\psi})}\left[\alpha_{\phi,\psi}\left(x_{\phi+1,\psi} + x_{\phi-1,\psi}\right)\right. \\
&+ \gamma_{\phi,\psi}\left(x_{\phi,\psi+1} + x_{\phi,\psi-1}\right) \\
&+ \left.\beta_{\phi,\psi}\left(x_{\phi+1,\psi-1} - x_{\phi+1,\psi+1} + x_{\phi-1,\psi+1} - x_{\phi-1,\psi-1}\right)\right]
\end{aligned} \tag{5.14}$$

$$\begin{aligned}
y_{\phi,\psi} &= \frac{1}{2(\alpha_{\phi,\psi} + \gamma_{\phi,\psi})}\left[\alpha_{\phi,\psi}\left(y_{\phi+1,\psi} + y_{\phi-1,\psi}\right)\right. \\
&+ \gamma_{\phi,\psi}\left(y_{\phi,\psi+1} + y_{\phi,\psi-1}\right) \\
&+ \left.\beta_{\phi,\psi}\left(y_{\phi+1,\psi-1} - y_{\phi+1,\psi+1} + y_{\phi-1,\psi+1} - y_{\phi-1,\psi-1}\right)\right]
\end{aligned} \tag{5.15}$$

These formulae are applied to the x and y coordinates of the grid iteratively. If $x^m_{\phi,\psi}$ and $y^m_{\phi,\psi}$ are the grid points after m iterations, then a smoother grid, represented by

$x^{m+1}_{\phi,\psi}$ and $y^{m+1}_{\phi,\psi}$, can be obtained from:

$$\begin{aligned} x^{m+1}_{\phi,\psi} &= \frac{1}{2(\alpha_{\phi,\psi}+\gamma_{\phi,\psi})}\Big[\alpha_{\phi,\psi}(x^m_{\phi+1,\psi}+x^m_{\phi-1,\psi}) \\ &+ \gamma_{\phi,\psi}(x^m_{\phi,\psi+1}+x^m_{\phi,\psi-1}) \\ &+ \beta_{\phi,\psi}(x^m_{\phi+1,\psi-1}-x^m_{\phi+1,\psi+1}+x^m_{\phi-1,\psi+1}-x_{\phi-1,\psi-1})^m\Big] \end{aligned} \tag{5.16}$$

$$\begin{aligned} y^{m+1}_{\phi,\psi} &= \frac{1}{2(\alpha_{\phi,\psi}+\gamma_{\phi,\psi})}\Big[\alpha_{\phi,\psi}(y^m_{\phi+1,\psi}+y^m_{\phi-1,\psi}) \\ &+ \gamma_{\phi,\psi}(y^m_{\phi,\psi+1}+y^m_{\phi,\psi-1}) \\ &+ \beta_{\phi,\psi}(y^m_{\phi+1,\psi-1}-y^m_{\phi+1,\psi+1}+y^m_{\phi-1,\psi+1}-y^m_{\phi-1,\psi-1})\Big] \end{aligned} \tag{5.17}$$

The Winslow formula is a weighted average of the coordinate values of a nodes nearest neighbours. However, in practice it behaves far better than a simple average, which must be the result of the weights used. Notice how the values at the grid points $(\phi+1,\psi)$ and $(\phi-1,\psi)$ are weighted by the lengthscale between the grid points $(\phi,\psi+1)$ and $(\phi,\psi-1)$. Similarly, the values of the grid points $(\phi,\psi+1)$ and $(\phi,\psi-1)$ are weighted by the lengthscale between the grid points $(\phi+1,\psi)$ and $(\phi-1,\psi)$. This cross weighting means that if the grid contains large aspect ratio cells, the Winslow formula will act to connect the grid points that are close together with a straightline, rather than those that are far apart. This type of mesh movement is ideal for large aspect ratio zones as it acts to avoid "boomerang" cells and cell invertion.

It should also be noted that β is proportional to the dot product of ϕ and ψ mesh legs. This means that the β term will vanish if the grid is orthogonal. If β is small then further iterations will act to reduce it further and make the mesh more orthogonal.

The Winslow formula also implicitly assumes a weight function which is proportional to the Jacobian of the transformation, which happens to equal the area of the quadrilateral cells. This contributes to why Winslow's formula is so effective. It means that at each iteration large cells will attract grid points more strongly than small ones and tend to even out cell areas. Some of these observations are clear from direct inspection of the formula, however, Tipton [56] derives Winslow's formula from a variational principle, which leads to further insights.

Winslow's equipotential mesh relaxation algorithm has been implemented in CORVUS as part of the adaptive multi-material ALE algorithm by exploiting the block structured nature of the unstructured grid that the code uses. The node to

node connectivity enables a nine node stencil to be constructed for all internal nodes within each region or mesh block. The only complication is the treatment of block boundaries, where the number of immediate neighbours will depend on the initial mesh generation for the problem performed by the code user. Reflecting boundary conditions have been implemented by simply reflecting the nodal coordinates and then applying the formula directly. Merged material interfaces are either treated as Lagrangian, or if a nine cell stencil cannot be constructed to allow direct application of Winslow's scheme, then a simple average of the coordinates of the immediate neighbour nodes is used. The latter is considered adequate as this typical only amounts to a very small fraction of the nodes in the problem.

5.1.4 Weighted mesh movement

In the unmodified form, discussed above, Winslow's mesh movement algorithm is unbiased in the sense that it strives for equal mesh quality throughout the domain. However, in some problems the resolution of certain materials or flow features is of far greater importance than the rest of the problem. In order to calculate efficiently this class of problems it is important to be able to focus resolution in these key areas. This can be achieved, to a degree, by modifying Winslow's original scheme to include nodal weights or weight functions [57, 58], the weights simply being higher in areas of greater importance. Mesh lines are then attracted towards these features.

Brackbill and Saltzman's [57] approach was to recast Winslow's scheme in terms of a variational principle comprising four optional integral terms. The first of these is a measure of global mesh smoothness, and if applied in isolation is equivalent to Winslow's original scheme. The next two terms provide alternative measures of mesh orthogonality, whilst the final term is a volume integral of a user defined weight function. It is this last term which can be used to adapt the mesh to resolve some feature of interest.

Winslow, in contrast [58] generalizes his original scheme by introducing a diffusion coefficient into (5.1) by writing them in the form

$$\begin{aligned} \nabla \cdot (D \nabla \phi) &= 0 \\ \nabla \cdot (D \nabla \psi) &= 0 \end{aligned} \tag{5.18}$$

These equations are then inverted in a similar manner to Winslow's original method. It was suggested in [58] that the diffusion coefficient for the modified scheme would be a function of the local gradient of one of the physical variables.

A simpler approach has been taken by the author, principally due to concerns over the use of the weight functions used by the two schemes discussed above. These concerns stem from the penalty nature of ALE mesh adaption; that is the number of zones in the problem is fixed, so if an area is refined then a compensating

derefinement must also occur somewhere else in the problem. This means that the unbounded weight functions could produce severe changes in mesh resolution, which could actually degrade the quality of the solution rather than improve it.

Given these concerns, a simpler scheme has been developed by the author. The user simply defines a series of constant weight values for the features or material to be refined in each region. A background weight value of unity is assumed. This enables the user to ensure that the mesh resolution does not change to rapidly. This typically requires the user defined weights to not exceed 3-4, although there is an element of problem dependence. The weights are assigned at each time step just prior to the ALE mesh movement algorithms being applied. The weights are assigned first by looping over all the elements and testing to see if each element meets some refinement criterion. If it does, then appropriate user defined weight value is applied to the four nodes of the element. Two further operations are then required. The first acts to propagate the high weight values outwards to create a buffer zones around the features or materials to be refined. This is done by performing a series of sweeps over all the nodes, redefining the nodal weight to be the maximum weight value assigned to each node and it's immediate neighbours. The more sweeps that are performed, the larger the buffer zone that is created. A final series of sweeps over the nodes is then performed but now applying the average weight that has been assigned to the node and it's immediate neighbours. This last step is required to ensure a smooth variation in the weights applied. This again attempts to reduce the errors that can be introduced when shocks propagate across sudden changes in mesh resolution. The number of sweeps performed for both these latter operations is again under user control, but five passes for both operations is recommended from the numerical experiments that have been performed by the author.

It now remains to discuss how to modify Winslow's mesh movement algorithm to take account of these nodal weights. This has been achieved simply by taking (5.16, 5.17) and scaling the α, β and γ terms by their normalized nodal weight. Equations (5.16, 5.17) then become,

$$\begin{aligned} W_{\phi,\psi} &= \alpha_{\phi,\psi}(w_{\phi+1,\psi} + w_{\phi-1,\psi}) + \gamma_{\phi,\psi}(w_{\phi,\psi+1} + w_{\phi,\psi-1}) \\ &+ \beta_{\phi,\psi}(w_{\phi+1,\psi-1} - w_{\phi+1,\psi+1} + w_{\phi-1,\psi+1} - w_{\phi-1,\psi-1}) \end{aligned} \tag{5.19}$$

$$\begin{aligned} x^{m+1}_{\phi,\psi} &= \frac{1}{W_{\phi,\psi}}\Big[\alpha_{\phi,\psi}(w_{\phi+1,\psi}x^m_{\phi+1,\psi} + w_{\phi-1,\psi}x^m_{\phi-1,\psi}) \\ &+ \gamma_{\phi,\psi}(w_{\phi,\psi+1}x^m_{\phi,\psi+1} + w_{\phi,\psi-1}x^m_{\phi,\psi-1}) \\ &+ \beta_{\phi,\psi}(w_{\phi+1,\psi-1}x^m_{\phi+1,\psi-1} - w_{\phi+1,\psi+1}x^m_{\phi+1,\psi+1} \\ &+ w_{\phi-1,\psi+1}x^m_{,\psi+1} - w_{\phi-1,\psi-1}x^m_{\phi-1,\psi-1})\Big] \end{aligned} \tag{5.20}$$

$$
\begin{aligned}
y^{m+1}_{\phi,\psi} &= \frac{1}{W_{\phi,\psi}}\Big[\alpha_{\phi,\psi}(w_{\phi+1,\psi}y^{m}_{\phi+1,\psi} + w_{\phi-1,\psi}y^{m}_{\phi-1,\psi}) \\
&+ \gamma_{\phi,\psi}(w_{\phi,\psi+1}y^{m}_{\phi,\psi+1} + w_{\phi,\psi-1}y^{m}_{\phi,\psi-1}) \cdot \\
&+ \beta_{\phi,\psi}(w_{\phi+1,\psi-1}y^{m}_{\phi+1,\psi-1} - w_{\phi+1,\psi+1}y^{m}_{\phi+1,\psi+1} \cdot \\
&+ w_{\phi-1,\psi+1}y^{m}_{\phi-1,\psi+1} - w_{\phi-1,\psi-1}y^{m}_{\phi-1,\psi-1})\Big] \qquad (5.21)
\end{aligned}
$$

In order to preserve symmetry where reflecting boundary conditions are applied, in addition to reflecting the coordinate values, as discussed above, the weights are also reflected in the same way. The modified iteration equations (5.20, 5.21) are then applied to generate new positions for the boundary nodes in exactly the same way as for the internal nodes, using a mixture of internal, boundary and reflected nodal coordinates and their weights.

This nodal weighting scheme has also been implemented for use in conjunction with the simpler mesh movement technique, where nodes are repositioned to take the average coordinates of their immediate neighbours. The nodal weights are then used to produce a weighted average of the coordinates of the nodes immediate neighbours. Global weights can also be applied to this scheme to bias the mesh movement in favour of a particular logical mesh direction throughout a region. This can be useful in some problems, for example if a material interface is undergoing severe deformation, but it is important to treat the interface as a Lagrangian slide line, possibly to allow slip or void opening at the interface. Then the region weights can be chosen to force the mesh lines to follow the profile of deforming interface, the mesh adaption being focussed completely on conforming to this boundary. This works well, provided the rest of the problem is well behaved, and the mesh is not required to conform to other boundaries or flow features as well.

5.2 Mesh Insertion

In many applications, particularly those involving convergent flows, it is impossible to maintain adequate mesh resolution for all flow features of interest using node movement alone. A hybrid method, combining ALE with Adaptive Mesh Refinement (AMR), could solve this problem. In such a scheme, mesh movement would only be used to maintain mesh quality and robustness, while AMR would be used to add and remove zones where and when required to maintain adequate resolution of all features of interest. In order to provide some relief from this problem, and start to assess the possibility of developing a hybrid capability, an Automatic Mesh Insertion (AMI) facility was developed by the author. AMI dynamically inserts mesh lines in one logical mesh direction, as required, to ensure a user defined aspect ratio is not exceeded [59].

In order to use the AMI option, the user must define the regions where AMI is active, select the logical mesh direction in which mesh lines will be inserted, and a time interval between error checks. The maximum number of mesh lines that can inserted during the problem and the maximum tolerable aspect ratio must also be defined for each region where AMI is active. An optional minimum tolerable aspect ratio can also be defined to avoid over refinement.

The maximum storage requirement is determined at the start of the problem from the maximum number of mesh lines that can be inserted in each region and the initial meshing. The aspect ratio error check finds the maximum aspect ratio for each logical line of elements for each region where AMI is active. If the tolerable aspect ratio is exceeded, then the line of elements is flagged for refinement. Adjacent lines of elements that have all been flagged for refinement are grouped into blocks and passed to an integral rezone package.

The cells flagged for refinement are subdivided exactly to simplify this rezone process and improve computational efficiency. The mesh insertion step is performed before the ALE advection step, which can then be used to improve the quality of the refined grid. In order to introduce new elements, the current element and node indices are shifted upwards at the point where the elements are to be added, to maintain a continuous list.

The solution must then be interpolated or rezoned from the old mesh onto the new refined grid. This is a similar process to the continuous rezoning required to support the ALE mesh movement. However, the latter problem is greatly simplified by the mesh topology not changing. This restriction allows efficient second order advection methods to be used. However, inserting meshes does change the mesh topology and so requires a more expensive integral rezone method to be used.

The two different types of rezone also have different requirements. Both must be strictly conservative, but advection or continuous rezone methods are applied every time step, so a first order method would be too diffusive. In contrast, mesh insertions are less frequent, so a first order integral rezone is considered acceptable. The cost of the integral rezone has been minimised by exactly subdividing zones: this fixes the topology of the superimposed old and new grids, significantly reducing the logic that would be required to support mapping between two totally arbitrary grids.

The element centred quantities such as density and internal energy are mapped first. If strength is present, then the stress deviators, equivalent plastic strain, plastic work and elastic distortional energy are also mapped. The two velocity components are then mapped as momentum components using the same rezone procedure as for the element centred quantities, but using a dual node centred mesh. The dual mesh must however first be constructed for both the old and new meshes. This is done by taking the centroids of each of the old and new cells as vertices for the dual mesh. As a finite element Lagrangian hydro scheme is used, the nodal control volumes can only be approximated by the dual mesh, except where the mesh is orthogonal.

However, this approximation appears perfectly adequate in practice.

The integral rezone simply calculates new values for the state variables of each of the new elements, by integrating the values from the old mesh contained within the volume of each new element, and then dividing by the new elements volume for each new element.

Computationally efficient conservative first and second order rezone methods have been developed by Dukowitz and Ramshaw [60, 61, 62, 63]. The first order scheme described in [62] is used here, although it is envisaged that a second order rezone option for the AMI package may be added at a later date following the approach described in [63].

A first order rezone assumes the conserved quantity to uniform within each old cell of the original mesh. This reduces the rezone problem to the calculation of overlap volumes for the superimposed old and new meshes. In [62] an efficient and straightforward computational procedure is outlined for carrying out this procedure for arbitrary quadrilateral elements, which is applicable to cartesian and cylindrical meshes.

The basic building block for Ramshaw's rezone method for 2D cartesian meshes is the formula for the area of an arbitrary polygon P. If the sides of the polygon are labelled with an index s, the coordinates of the end points of a side s will be denoted by (x_1^s, y_1^s) and (x_2^s, y_2^s). Then the area of the polygon P is given by,

$$A_p = \frac{1}{2} \sum_s \varepsilon_s^p (x_1^s y_2^s - x_2^s y_1^s) \tag{5.22}$$

where the summation is over all the sides of P, and ε_s^p is either +1 or -1 according to whether P lies to the left or right, respectively, of side s. It is important to note that each line segment must be treated as a separate entity, not as part of a particular polygon. The segment is best visualised as a directed line segment. In this way left and right can be defined uniquely for each line segment, by taking an observer who is facing from end point 1 to end point 2. Equation (5.22) can be derived by integrating the identity $\nabla \cdot \mathbf{r} = \mathbf{2}$ (where $\mathbf{r} = \mathbf{xi} + \mathbf{yj}$ is the position vector) over the area of P and then applying the divergence theorem.

Now consider the superposition of two arbitrary 2D quadrilateral meshes, the original mesh and the new mesh. This superposition creates a network of overlap areas such as those shown in Fig. 5.1, each of these overlap areas being contained within a single cell of the old mesh and a single cell of the new mesh. The overlap areas are all polygons whose sides are line segments. Each segment is common to two overlap areas, which may both be considered to be associated with the side.

The objective of the rezone is now to apportion a conserved quantity Q, whose volume density q is uniform within each cell of the old mesh, into the cells of the new mesh. Consider a particular overlap area that belongs to an old cell of density q. The overlap area A is given by (5.22), and the quantity $\nabla Q = qA$ is the contribution

Figure 5.1: Superposition of old and new meshes.

of this overlap to the new cell in which it lies. (Note unit depth is assumed for cartesian problems so areas and volumes have the same numerical values). The final value of Q for the new cell is the sum of all the ∇Q contributions from all the overlap areas that make up the new cell area, where division by the cell area gives the density q for the new cell. It is clear that this procedure leads to no creation or destruction of Q at any stage, so the rezoning method is conservative.

Ramshaw's efficient implementation of this rezoning method for arbitrary 2D meshes exploits the observation that according to (5.22) each of the overlap increments ∇Q are themselves a sum of contributions associated with the individual sides that make up the overlap area. Thus instead of calculating each individual overlap area directly in turn, the efficiency of the computation can be improved, if the same operations are performed in a different order. A sweep is made over all the line segments s, which are each common to two overlap areas, contributing ∇Q_L to the left overlap and ∇Q_R to the right overlap area. Given that both of these contributions involve the common factor $\frac{1}{2}(x_1^s y_2^s - x_2^s y_1^s)$, it is more efficient to calculate them at the same time. However, the way these contributions are calculated, and how they are used depends on whether the segment is part of the old or new mesh.

If the segment is part of the old mesh, and is common to two old cells, L and R, then the L overlap is completely contained within the L cell, which has density q_L

and the R overlap is completely contained within the R cell, which has a density q_R. If both these overlap areas are completely contained within the same new cell, then the quantity Q in this new cell containing s is therefore incremented by an amount

$$\nabla_s^O = \frac{1}{2}(q_L - q_R)(x_1^s y_2^s - x_2^s y_1^s) \tag{5.23}$$

If side s is a segment of the new mesh and so is common to two new mesh cells, L and R, then the L overlap area is entirely contained within the cell L and the R overlap area is entirely contained within the R cell. If both these overlap areas are entirely contained within the same old cell of density q_o, then the quantity Q in new cell L, to the left of segment s, should be incremented by an amount

$$\nabla_s^N = \frac{1}{2} q_o (x_1^s y_2^s - x_2^s y_1^s) \tag{5.24}$$

and the quantity Q, in the new cell R by just $-\nabla_s^N$. Therefore Q_L is to be incremented by ∇_s^N, while Q_R is to be decremented by the same amount.

Clearly, there are ambiguities for the above procedure when an old mesh segment coincides with a new mesh segment. Ramshaw overcomes these problems by adding additional rules that handle the problem in a symmetric manner. The increment ∇_s^o for the old mesh segment is evaluated in the usual manner, but each of the common new cells to the segment s are incremented by $\frac{1}{2}\nabla_s^o$. The density q_o, used for the coincident new mesh segment, is taken as the average of the density values for the adjacent old cells. The ∇_s^N contributions are then evaluated in the usual way using (5.24).

The extension of the method to cylindrical (r, z) coordinates is straightforward, and is based on the use of a formula for the volume of revolution of a polygon P about the z coordinate axis, and is given in terms of the side endpoint coordinates (z_1^s, r_1^s) and (z_2^s, r_2^s) by

$$V_p = \frac{\pi}{3} \sum_s \varepsilon_s^p (r_1^s + r_2^s)(z_1^s r_2^s - z_2^s r_1^s) \tag{5.25}$$

The cartesian method can then be applied to the cylindrical case, simply by replacing the $\frac{1}{2}(x_1^s y_2^s - x_2^s y_1^s)$ term by $\frac{\pi}{3}(r_1^s + r_2^s)(z_1^s r_2^s - z_2^s r_1^s)$.

In order to implement the rezone procedure for arbitrary old and new quadrilateral meshes, fairly complicated and expensive logic must be put in place to determine all the intersection points for the old and new meshes. A data structure must then be created, which describes the resulting network of line segments with all their end points. However, the topology of the superimposed old and new mesh is always fixed for the rezones that must be performed in support of the AMI package, as the zones refined in this case are always exactly subdivided. This has made it possible to introduce significant simplifications to Ramshaw's procedure and further efficiency improvements.

Adjacent mesh lines flagged for refinement are first grouped into blocks and passed to the rezone package. A number of 2D arrays are then created. These include the x and y coordinates for the element vertices for the old and new meshes, for mapping the element centred quantities, and the x and y coordinates of the vertices of the dual mesh, used to map the nodal quantities. The element centred quantities are mapped first. For each element centred variable that must be remapped two sweeps are performed in the two logical mesh coordinates, first for the old mesh, and then the new mesh. In each case, the contributions to the quantity being rezoned are calculated using (5.23) and summed for each new element. Similar sweeps are then performed over the new mesh data structure to sum the contributions from the new mesh segments using (5.24). The new element volumes are also calculated in the same way simply by taking the old mesh density to be unity. The integrated quantities are then divided by the new cell volume to obtain the volumetric density for the variable to be mapped. The rezone procedure for node centred quantities such as velocity is analogous, except that it is performed on the dual mesh.

This rezone method is attractive not only for efficiency, but also because there is a clear means of extending it to second order in the future. The extension of the above method to second order [61, 63] relies upon reducing the number of dimensions for the rezone, the divergence theorem being used to reduce volume integrals to surface integrals for 3D or surface to line integrals for 2D i.e.

$$\int\int\int_{V_k} \nabla \cdot \bar{F} dV = \int\int_{S_k} \bar{n} \cdot \bar{F} dS \tag{5.26}$$

where $\bar{F}$ is a flux vector, V_k is the volume of integration, S_k is its surface and $\bar{n}$ is the outward unit normal vector to the surface. In order to use this relationship however the flux vector must be found, whose divergence is equal to the quantity to be mapped.

5.3 Mesh insertion Test problems

5.3.1 1D Spherical implosion

The 1D implosion problem is used to test spherical symmetry and energy conservation. The problem consists of a spherical Tantalum shell meshed with 2° angular zoning and 2 radial zones. The shell is given an initial uniform radial velocity of $1.0 cm\mu s^{-1}$. A Osbourne equation of state (3.97), constant yield strength and shear modulous is used to model the Tantalum. The problem was calculated both with and without automatic mesh insertion. The AMI calculation was performed with a maximum of 40 mesh insertions allowed, with aspect ratio error checks made every $0.1\mu s$ and the aspect ratio limit of 1.5.

Numerical Results

The mesh at $4.0\mu s$ from the calculation without mesh insertion is given in Fig. 5.2 and with mesh insertion in Fig. 5.3 and the corresponding density contour plots are given in Fig. 5.4 and Fig. 5.5. These results show that spherical symmetry is maintained, but the implosion is a little faster with mesh insertion. The steeper gradients revealed in the density contour plot from the calculation with AMI on, suggest the faster implosion is simply a result of increased resolution. The total energy conservation is comparable for the two calculations which also supports this conclusion.

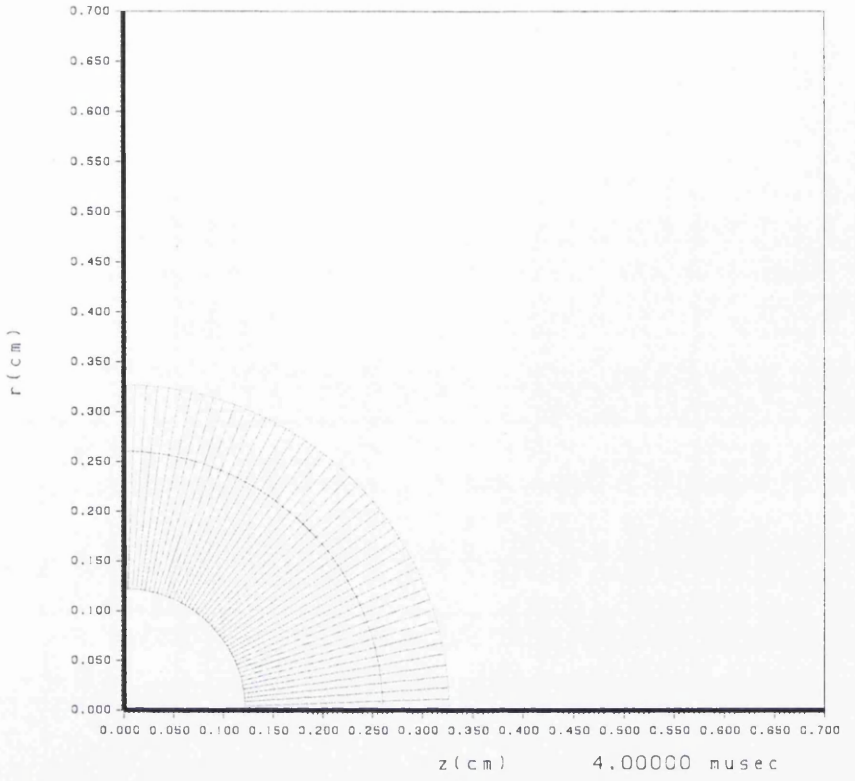


Figure 5.2: Mesh for 1D Spherical implosion problem calculated without AMI at 4.0 μs.

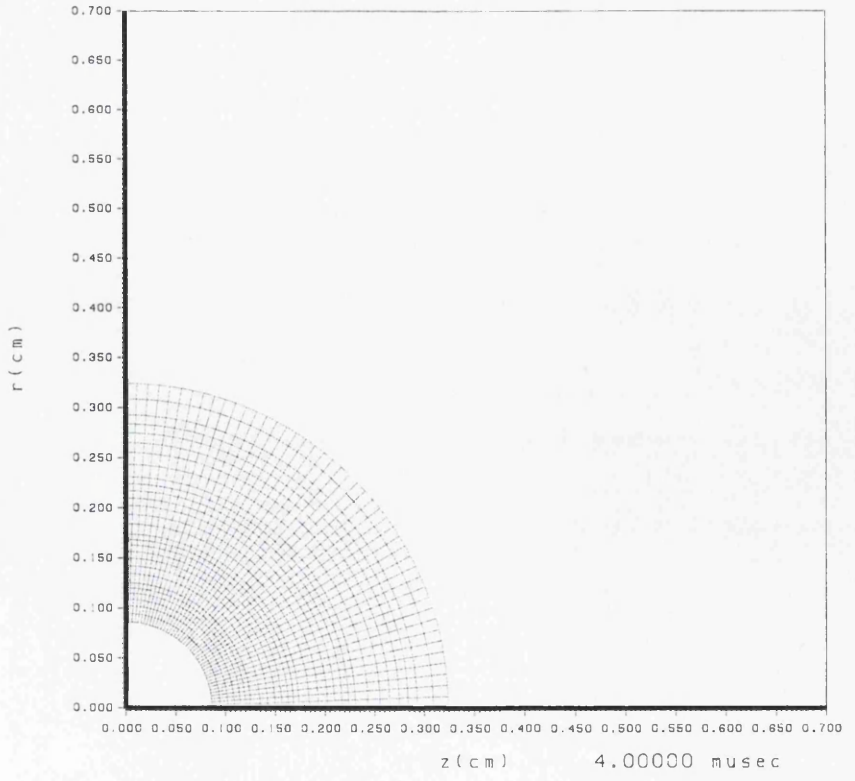


Figure 5.3: Mesh for 1D Spherical implosion problem calculated with AMI at 4.0 μs.

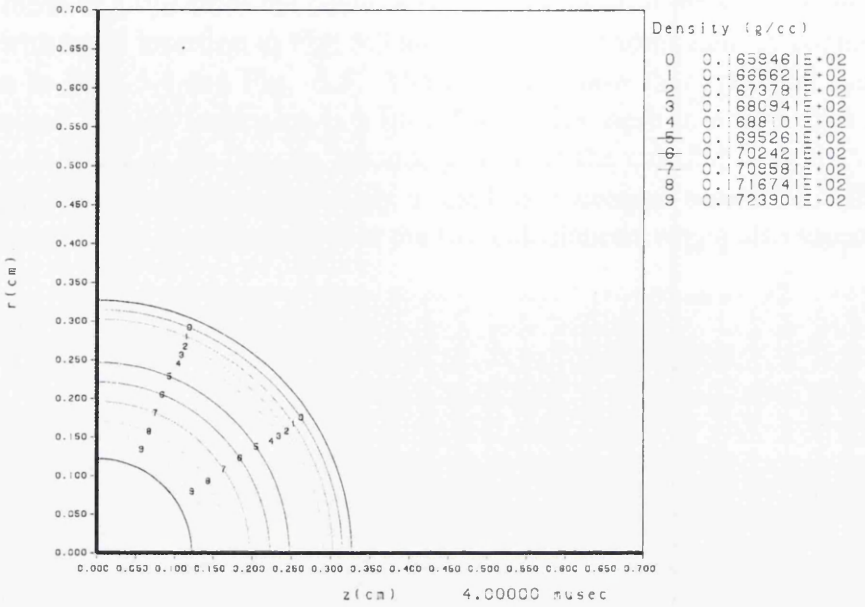


Figure 5.4: Density contour plot for 1D Spherical implosion problem calculated without AMI at 4.0 μs.

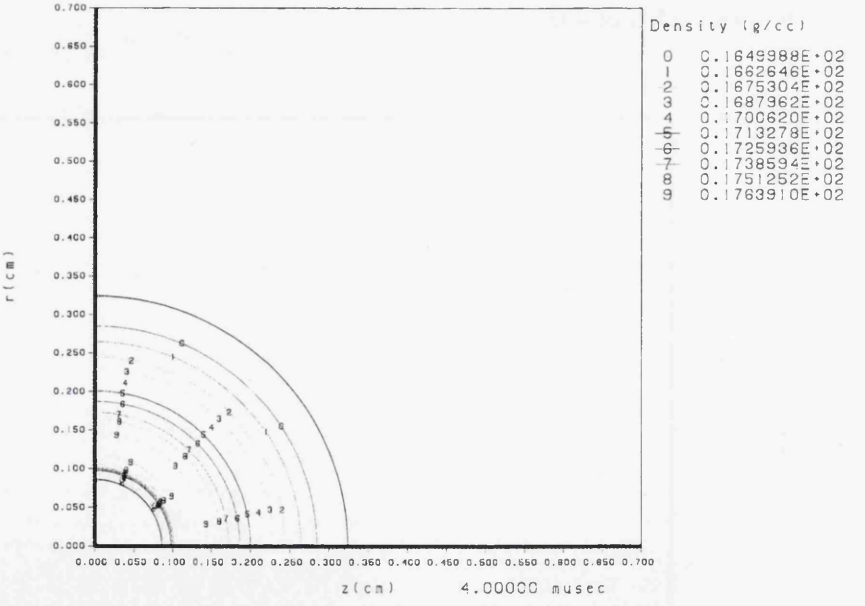


Figure 5.5: Density contour plot for 1D Spherical implosion problem calculated with AMI at 4.0 μs.

5.3.2 2D Spherical implosion

A 2D implosion test problem has also been calculated, where a Tantalum shell is imploded using an asymmetric drive that could, for example, have been generated by the application of explosives or high power lasers. In this case, a blunt toroidal jet forms around the equator, which presents the main challenge in terms of accuracy and robustness for the problem. The problem was again calculated with, and without, Automatic Mesh Insertion. The maximum number of additional mesh lines that can be inserted in the AMI calculation was again limited to 40; aspect ratio error checks were again made every $0.1\mu s$; and the aspect ratio limit was set to 1.5.

Numerical Results

The mesh at $5.2\mu s$ from the calculation without mesh insertion is given in Fig. 5.6 and with mesh insertion in Fig. 5.7, and pressure contour plots are also given for the two cases in Fig. 5.8 and Fig. 5.9. On comparing the interfaces obtained from the two calculations, the main differences, as expected, are in the jet profile. The jet obtained in the AMI calculation is blunter with more localised curvature. Away from the jet, the calculations show very similar interface profiles with the non-AMI calculation again a little slower. The pressure contours again show steeper gradients in the AMI calculation, but also show some 2D differences in the jetting region. The energy conservation is again comparable with and without AMI. This combination of a similar solution being obtained in the undeformed part of the shell, and local differences around the jetting site with and without AMI, suggest that the AMI facility is improving resolution without introducing any significant errors.

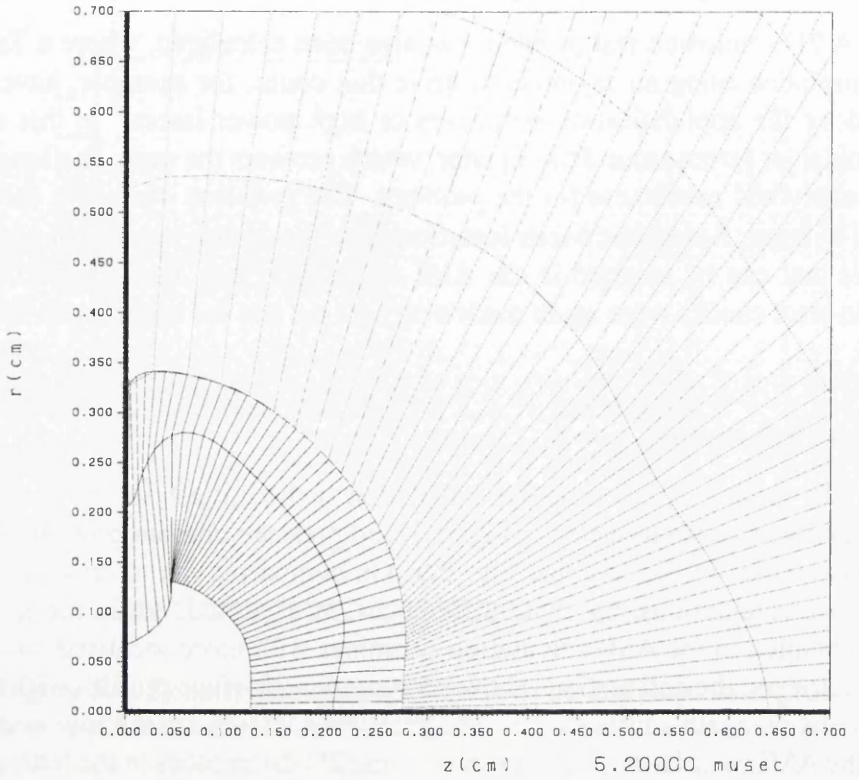


Figure 5.6: Mesh for 2D implosion of a spherical metal shell calculated without AMI at 5.2 μs.

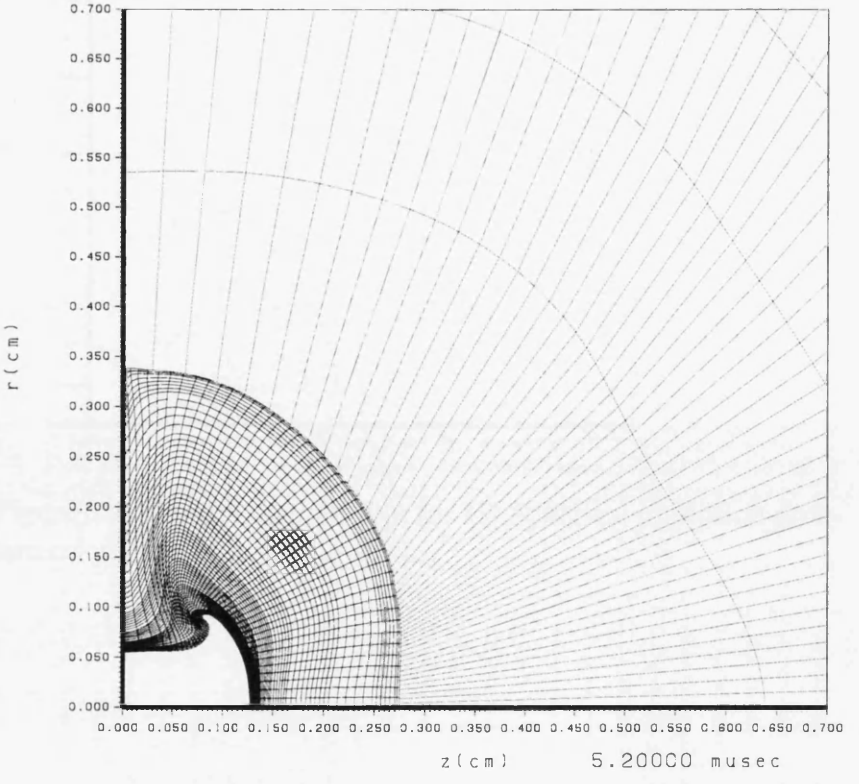


Figure 5.7: Mesh for 2D implosion of a spherical metal shell calculated with AMI at 5.2 μs.

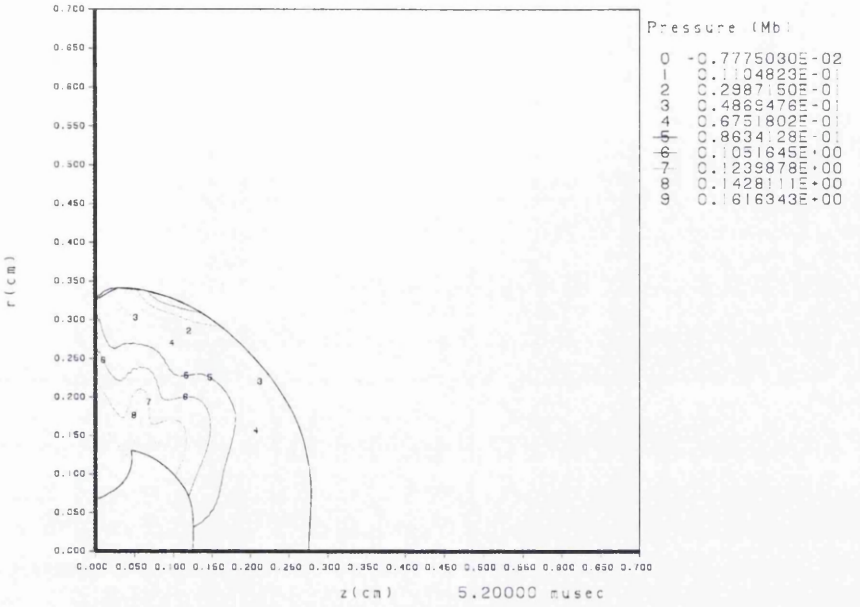


Figure 5.8: Pressure contour plot for 2D implosion of a spherical metal shell calculated without AMI at 5.2 μs.

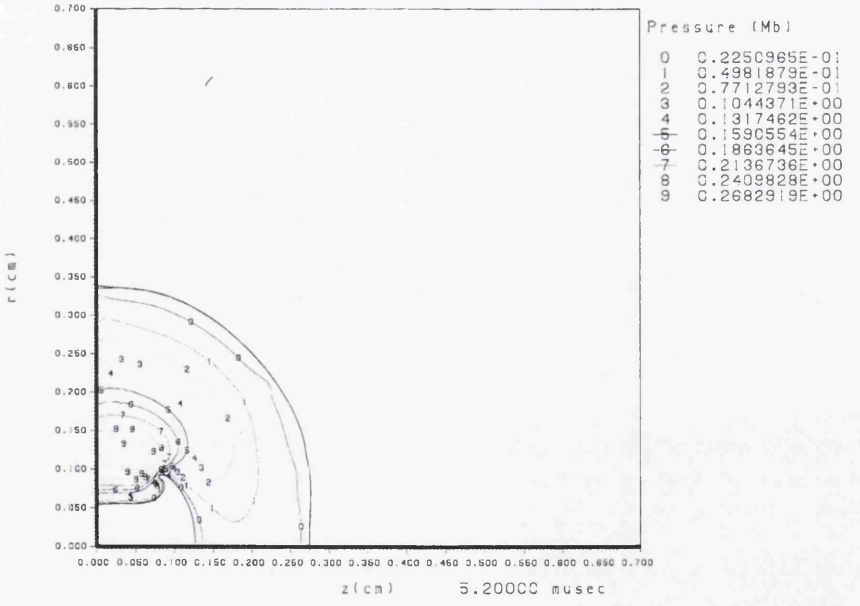


Figure 5.9: Pressure contour plot for 2D implosion of a spherical metal shell calculated with AMI at 5.2 μs.

Chapter 6

Advection Methods

The process of mapping from one computational mesh to another is termed a rezone. There are two main types of rezone, integral and continuous methods. The integral methods overlay the two meshes to form sub volumes, then integrate each variable to be mapped over each of these subvolumes. The conserved quantity obtained is then summed for all the subvolumes within each new element, and then divided by the new element volume to obtain the new quantity for each element. The advantage of integral methods is that the new mesh is not constrained to have the same topology or resolution as the old mesh, and there are no limits on how far a node can move relative to it's immediate neighbours. The main disadvantage of integral methods is their computational expense, which in practice precludes their use at every time step.

Continuous rezones are less expensive and can in practice be used at every time step, so this is the approach used here. Continuous rezones are derived from the solution of the linear advection equation, and so are usually termed advection methods.

$$\frac{\partial a}{\partial t} + \lambda(x,t)\frac{\partial a}{\partial x} = 0 \tag{6.1}$$

$$a(x,t_0) = a_0(x)$$

where a is the variable to be rezoned and $\lambda(x,t)$ the mesh velocity. If λ is taken to be constant then the exact solution is simply

$$a(x,t) = a_0(x - \lambda t) \tag{6.2}$$

which corresponds to the initial data profile being translated at a speed of λ over a distance λt without changing profile.

6.1 Single material advection

6.1.1 van Leer Advection

van Leer developed a series of second and third order 1D advection methods as part of his seminal work to develop a high order Godunov method [64, 65, 66, 67, 68]. These advection methods can be extended to 2D using the well known Strang operator splitting [69] technique. This alternates the order with which 1D advection steps are performed at each time step, in order to obtain second order accuracy on 2D orthogonal grids. van Leer's advection methods were the starting point for the single material advection scheme used in the adaptive multi-material ALE algorithm. Although van Leer's advection methods are fully described in [67], a summary will be given here to aid the reader's understanding of how the method was modified to make it compatible with the non-orthogonal unstructured grids used in CORVUS.

In [67] van Leer discusses three second order and three third order advection schemes. These can all be summarised in four steps as follows:

1. Given the initial function $a(x,t_0)$, determine cell average values

$$\bar{a}_{i+\frac{1}{2}} = \frac{1}{\Delta x_{i+\frac{1}{2}}} \int_{x_i}^{x_{i+1}} a(x,t_0)dx \tag{6.3}$$

2. Replace the original distribution with either a piecewise constant, linear or parabolic function depending on which of the schemes is to be used.

3. Integrate over a finite time step Δt subject to the normal CFL condition

$$|\sigma| \leq 1 \tag{6.4}$$

where $\sigma = \frac{\lambda \Delta t}{\Delta x}$ is the Courant or CFL number.

4. Determine the new mesh averages and then repeat the sequence from step 2 onwards.

The order of the scheme is defined by the level of approximation used for the initial data in step 2. A 1st order scheme is obtained if a piecewise constant distribution is assumed, a linear distribution gives 2nd order and a parabolic distribution a 3rd order scheme.

A 2nd order advection method has been selected for CORVUS, as this is consistent with the accuracy of the Lagrangian step and is believed to offer best balance between accuracy and computational cost. However, the simple 1st order case, as in Godunov's original method, will be described first for completeness and to aid understanding of what follows.

The initial distribution is replaced by a piecewise constant distribution $A(x,t_0)$ given by

$$A(x,t_0) = \bar{a}_{i+\frac{1}{2}}, \qquad x_i < x < x_{i+\frac{1}{2}} \tag{6.5}$$

This is then integrated over a timestep to obtain new values,

$$a(x,t_1) = A(x-\lambda\Delta t, t_0) \tag{6.6}$$

New mesh averages are then obtained for $a(x,t_1)$. The numerical scheme for updating the cell average value is then given by the conservative formula

$$a^{n+1}_{i+\frac{1}{2}} = a^n_{i+\frac{1}{2}} + \frac{\Delta t}{\Delta x_{i+\frac{1}{2}}}(f_i - f_{i+1}) \tag{6.7}$$

where the intercell fluxes f_i are given by

$$f_i = \begin{cases} \lambda a_{i-\frac{1}{2}} & if \quad \lambda \geq 0 \\ \lambda a_{i+\frac{1}{2}} & if \quad \lambda < 0 \end{cases} \tag{6.8}$$

The 1st order upwind scheme for the linear advection equation obtained is the CIR scheme of Courant, Isaacson and Rees [70].

The piecewise linear method developed by van Leer assumes a piecewise linear distribution of the form

$$A(x,t_0) = \overline{a}_{i+\frac{1}{2}} + \frac{\Delta a_{i+\frac{1}{2}}}{\Delta x_{i+\frac{1}{2}}}(x - x_{i+\frac{1}{2}}), \qquad x_i < x < x_{i+\frac{1}{2}} \tag{6.9}$$

As for the 1st order case, (6.7) is used to update the cell average values, but the form of the fluxes is clearly different. In the 1st order scheme Δf_i defined a rectangular area crossing the cell boundary. In this case the areas will be trapezoids and the fluxes given by,

$$f_i = \begin{cases} \lambda\left[a_{i-\frac{1}{2}} + \frac{\Delta a_{i-\frac{1}{2}}}{2\Delta x_{i-\frac{1}{2}}}(\Delta x_{i-\frac{1}{2}} - \lambda\Delta t)\right] & if \quad \lambda \geq 0 \\ \lambda\left[a_{i+\frac{1}{2}} - \frac{\Delta a_{i+\frac{1}{2}}}{2\Delta x_{i+\frac{1}{2}}}(\Delta x_{i+\frac{1}{2}} + \lambda\Delta t)\right] & if \quad \lambda < 0 \end{cases} \tag{6.10}$$

It now remains to define how to calculate the slopes used. Three different slope definitions are proposed by van Leer in [67] which lead to three different 2nd order schemes.

Definition I simply obtains Δa from a central finite difference of $\overline{a}$, i.e.

$$\Delta a_{i+\frac{1}{2}} = \frac{1}{2}(\overline{a}_{i+\frac{3}{2}} - \overline{a}_{i-\frac{1}{2}}) \tag{6.11}$$

Definition II differences the original initial value distribution $a(x,t_0)$

$$\Delta a_{i+\frac{1}{2}} = a(x_{i+1},t_0) - a(x_i,t_0) \tag{6.12}$$

Δa is then taken to be independent from $\bar{a}$ and separate methods used to update Δa and a as described in [67].

Definition III requires Δa for each cell $A(x,t_0)$ to have the same first moment as $a(x,t_0)$, i.e.

$$\Delta a_{i+\frac{1}{2}} = \frac{\int_{x_i}^{x_{i+1}} a(x,t_0)\cdot\left[\frac{x-x_{i+\frac{1}{2}}}{\Delta x}\right]dx}{\int_{x_i}^{x_{i+1}}\left[\frac{x-x_{i+\frac{1}{2}}}{\Delta x}\right]^2 dx}$$

$$= \frac{12}{\Delta x^2}\int_{x_i}^{x_{i+1}} a(x,t_0)\cdot(x-x_{i+\frac{1}{2}})dx \qquad (6.13)$$

Δa is again independent from $\bar{a}$ and as with definition II separate methods are used for updating Δa and a as described in [67]. The three schemes resulting from these three definitions are all 2nd order accurate but can produce spurious oscillations. In order to overcome this problem van Leer applies a monotonic limiter [67] to the slopes calculated by the above definitions. This approach will completely remove the spurious oscillations for linear advection problems. The slopes, as modified by the monotonicity algorithm, are given by

$$(\Delta a_{i+\frac{1}{2}})_{mono} = \begin{cases} \alpha(a_{i+\frac{3}{2}} - a_{i-\frac{1}{2}}) & if \quad \beta > 0 \\ 0 & otherwise \end{cases} \qquad (6.14)$$

where

$$\alpha = min(\frac{1}{2}|\Delta a_{i+\frac{1}{2}}|, 2|a_{i+\frac{3}{2}} - a_{i+\frac{1}{2}}|, 2|a_{i+\frac{1}{2}} - a_{i-\frac{1}{2}}|)$$

$$\beta = (a_{i+\frac{3}{2}} - a_{i+\frac{1}{2}})(a_{i+\frac{1}{2}} - a_{i-\frac{1}{2}})$$

Comparable third order advection schemes are also developed by van Leer in [67]. These methods are simply obtained by approximating the initial value distribution by a quadratic polynomial of the form,

$$A(x,t_0) = \bar{a}_{i+\frac{1}{2}} + \Delta a_{i+\frac{1}{2}}\frac{x-x_{i+\frac{1}{2}}}{\Delta x_{i+\frac{1}{2}}} + \frac{1}{2}\Delta^2 a_{i+\frac{1}{2}}\left[\left(\frac{x-x_{i+\frac{1}{2}}}{\Delta x_{i+\frac{1}{2}}}\right)^2 - \frac{1}{12}\right] \qquad (6.15)$$

Again, three different schemes are proposed which correspond to third order extensions of the three second order schemes given above. However, these will not be discussed further as a second order method is used in CORVUS.

6.1.2 Advection on 2D unstructured non-orthogonal grids

In developing the 2D Eulerian code MUSCL (Monotonic Upstream-centred Scheme for Conservation Laws) [67] Woodward used the operator splitting approach of Strang [69] to extend van Leers's 1D advection methods to 2D. However, although this is a natural approach for logical orthogonal meshes, it is not obvious how to apply Strang splitting to unstructured non-orthogonal grids, where there are no natural sweep directions. In CORVUS an isotropic method is used to extend van Leer's method into 2D. The isotropic method simply calculates the four quasi 1D advection fluxes to be exchanged between each cell and its four immediate neighbours, and then performs a single advection update for each cell using these fluxes. This treatment will clearly lead to a corner coupling error, since it will take two time steps for information to propagate from a cell to its immediate diagonal neighbours. However, these corner coupling errors should be small, since the mesh movement algorithms used in CORVUS will tend to keep the mesh aligned with the underlying flow. If problems are encountered later then a solution to the corner coupling problem may be to adopt some variant of Collela's Corner Transport Upwind (CTU) [71] scheme, which is compatible with the multi-block unstructured grids used by CORVUS.

The slope and distance measures used by van Leer also require modifications to allow for the non-orthogonal ALE meshes used. In CORVUS [59] the volume coordinate approach developed by Benson [9] is used to overcome this problem. Benson uses van Leer's second order scheme, which assumes a linear initial distribution of the variable to be mapped in each cell. However, given the potentially large variations that can occur in mesh size for ALE computations, the slope for each isoparametric direction is obtained by fitting a parabola to the variable ϕ to be mapped in each cell. For a cell α and with two immediate neighbours, $\alpha - 1$ and $\alpha + 1$ the slope of the parabola at α is given by:

$$\begin{aligned} \frac{\partial \phi_\alpha}{\partial x} &= \frac{(\phi_{\alpha+1} - \phi_\alpha)\Delta x_\alpha^2 + (\phi_\alpha - \phi_{\alpha-1})\Delta x_{\alpha+1}^2}{\Delta x_\alpha \Delta x_{\alpha+1}(\Delta x_\alpha + \Delta x_{\alpha+1})} \\ \Delta x_\alpha &= x_\alpha - x_{\alpha-1} \end{aligned} \tag{6.16}$$

This slope gives second order accuracy, but is unstable. Stability is achieved as with van Leer's original scheme by applying a monotonic limiter to this slope. Benson's

limiter [9] is essentially the same as that used by van Leer.

$$\phi'_\alpha = \frac{1}{2}(sgn(\Delta\phi_\alpha) + sgn(\Delta\phi_{\alpha+1}))min\left(\left|\frac{\partial\phi_\alpha}{\partial x}\right|, |\Delta\phi_\alpha|, |\Delta\phi_{\alpha+1}|\right)$$

$$\Delta\phi_\alpha = \frac{\phi_\alpha - \phi_{\alpha-1}}{x_\alpha - x_{\alpha-1}}$$

$$\Delta\phi_{\alpha+1} = \frac{\phi_{\alpha+1} - \phi_\alpha}{x_{\alpha+1} - x_\alpha} \tag{6.17}$$

The volume swept out along the isoparametric coordinate directions then provides a natural and unique distance measure for non-orthogonal grids. Physical intuition also suggests this approach if the element boundaries are considered to define a variable cross section channel in one of the two isoparametric coordinate directions. The volume between integration points is then the mean path length for particles flowing down the channel scaled by the average cross sectional area.

The slope in volume coordinates is then obtained by substituting the following definitions into (6.17), where the notation used is explained in Fig. 6.1 and Benson calculates the partial volumes exactly. In CORVUS these volumes are replaced by volume integrals of the four shape functions for each element, which are already available as they are calculated during the Lagrangian step and used to apportion or scatter element masses and forces out to the nodes.

$$x_\alpha = V_{12}^- + V_{13}^- + V_{21}^- + V_{24}^- \tag{6.18}$$

$$x_{\alpha+1} = V_{22}^- + V_{23}^- + V_{31}^- + V_{34}^- \tag{6.19}$$

Advection fluxes are then calculated across each of the four edges of the ALE-ing elements, each flux being defined in terms of an overlap volume, $\Delta V_{\alpha i}$, and the average value of the variable to be mapped within this volume. The overlap volumes are constructed from the superposition of the old and new meshes as shown in Fig. 6.2. The overlap volumes are always quadrilaterals, so their volume can be calculated exactly using the same procedure as used to calculate element volumes. This requires four bi-linear shape functions to be defined for each overlap volume. The four vertices of these quadrilaterals are given by the coordinates of the two nodes defining the edge where the flux is to be calculated, at the end of the Lagrangian step and after mesh relaxation. The volume can then be obtained from the sum of the volume integrals of these shape functions in isoparametric coordinates,

$$\sum_{j=1,4}\int_{-1}^{1}\int_{-1}^{1} N_j r_i detJ d\xi d\eta$$

$$\sum_{j=1,4}\int_{-1}^{1}\int_{-1}^{1} \ldots d\xi d\eta \tag{6.20}$$

These overlap volumes will be calculated twice as they are common to two elements. In order to simplify the implementation of the advection scheme, the overlap

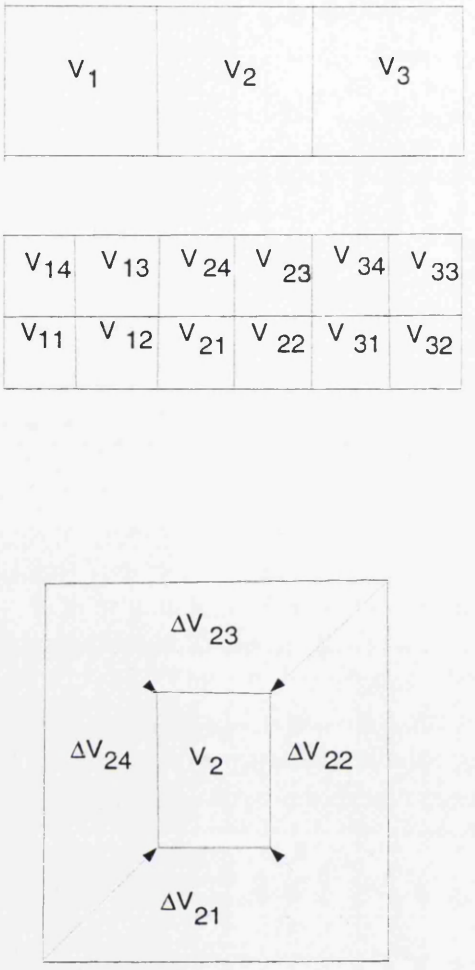


Figure 6.1: Notation for volume coordinates and overlap volumes.

volumes are set to zero, if they represent an incoming flux, and set positive, if they represent a donated or outgoing flux.

The fluxes, $\Delta\phi_{\alpha i}$, are evaluated using one-point integration, where α refers to the element number, and the second subscript is the side through which the flux flows. Following the notation given in Fig. 6.1 the element sides are numbered one through four. The derivatives $\phi'_{2,\xi}$ and $\phi'_{2,\eta}$ are the monotonic slopes, in the η and ξ-directions respectively, evaluated using the volume coordinates from (6.17) and

Figure 6.2: Overlap volumes are formed from the superposition of the old and new meshes.

(6.19). Then the fluxes become:

$$\begin{aligned}
\Delta\phi_{21} &= \Delta V_{21}(\phi_2^- - \phi'_{2,\xi}(V_{21} + V_{22} + \frac{1}{2}\Delta V_{21})) \\
\Delta\phi_{22} &= \Delta V_{22}(\phi_2^- - \phi'_{2,\eta}(V_{22} + V_{23} + \frac{1}{2}\Delta V_{22})) \\
\Delta\phi_{23} &= \Delta V_{23}(\phi_2^- - \phi'_{2,\xi}(V_{23} + V_{24} + \frac{1}{2}\Delta V_{23})) \\
\Delta\phi_{24} &= \Delta V_{24}(\phi_2^- - \phi'_{2,\eta}(V_{24} + V_{21} + \frac{1}{2}\Delta V_{24}))
\end{aligned} \tag{6.21}$$

The new value of ϕ is then evaluated using:

$$\phi_2^+ = \frac{1}{V_2^+}\left(\phi_2^- V_2^- - \sum_{i=1}^{4}\Delta\phi_{2i} + \Delta\phi_{53} + \Delta\phi_{34} + \Delta\phi_{41} + \Delta\phi_{12}\right) \tag{6.22}$$

It should also be noted that the advection scheme described above is expressed in what is termed a volume weighted form: that is, that the slopes and distance

measures are expressed in terms of volume. A mass weighted form is simply obtained by replacing the overlap volumes with mass fluxes and the slopes used as gradients with respect to change in mass, rather than volume. The author is not aware of any theoretical arguments that support the use of one of these weightings in preference to the other, but in practice the results from some problems do show an improvement, when internal energy is advected with a mass rather than a volume weighting.

In CORVUS mass is advected with a volume weighting, but the mass fluxes are stored for later use as overlap masses. The nodal velocities are then advected with a mass weighting, which seems natural as it is the momentum that is conserved. Internal energy is then advected with a mass weighting. The remaining variables are currently advected using volume weighting as no sensitivity to mass or volume weighted has been noted for these variables. These include the three stress deviators, equivalent plastic strain, elastic distortional energy and plastic work. A number of other variables are also advected when other optional physics packages are active.

6.1.3 Momentum advection

The advection of node centred variables, such as velocity, will now be discussed. The principle difficulty in advecting node centred variables is that there is no direct analogue of the element overlap masses for the nodes. Advection of nodal quantities on unstructured meshes adds additional complications, because a node can be connected to an arbitrary number of elements. This means that, in general, it is not possible to simplify the advection algorithm by only considering the exchange of advection fluxes with four main neighbours, as has been done for the element centred variables. However, given that CORVUS meshes are constructed from locally logically rectangular mesh blocks, it can be assumed that most nodes are only connected to four neighbours, and a special treatment applied to the block boundaries.

There are two main strategies in use by most hydrocodes for advecting nodal quantities [72]. Additional element centred variables can be introduced, advected using an element centred advection scheme, and the post advection values of these variable quantities used to determine the required nodal velocities. Alternatively the nodal mass fluxes can be estimated from the mass fluxes for the elements surrounding the node. The latter approach is usually referred to as a staggered mesh algorithm, as it creates a new mesh for the nodes, and then uses the new staggered mesh to perform the advection of the nodal quantities. This staggered mesh advection strategy was first used in a code call YAQUI developed at Los Alamos [2].

Staggered mesh schemes are attractive because they inherit the same dispersion and monotonicity characteristics as the underlying cell centred advection scheme used. They also only require one variable to be advected for each velocity component, whereas cell centred advection algorithms require several variables (two

for one dimension, four for two dimensions, and eight for three dimensions) in order to avoid adding dispersion errors to the solution [72]. However, the more expensive element centred advection algorithms do have the advantage that they are directly amenable to unstructured grids. However, as the grids currently in use with CORVUS are locally logically rectangular the more computationally efficient staggered mesh algorithm has been exploited within each mesh block, and a more general unstructured approach is applied to the relatively small number of nodes on block boundaries where more than four immediate neighbours must be considered. In order to simplify the presentation, the staggered mesh scheme used within blocks is described first, before moving on to discuss the more general treatment which is applied to the nodes on block boundaries.

The staggered method used in CORVUS differs a little from that in YAQUI [2], in that it does not explicitly define momentum control volumes. The nodal mass is simply obtained by dividing the mass of each element evenly among all its nodes. The mass fluxes are then derived by invoking the consistency condition of DeBar [73], which states that if a body has a uniform velocity and a variable density before advection, then the body must have the same uniform velocity after advection. This consistency condition is satisfied if the nodal mass fluxes are defined as the average of the mass fluxes for the four elements surrounding the node. This can also be further justified, if the momentum control volume of each node is taken to be constructed from $\frac{1}{4}$ of the volumes of its four adjacent elements. The boundary of this control volume is then a polygon, whose vertices are the centroids of the adjacent elements and the halfside points along the edges of the elements which are directly connected to the node, as shown in Fig. 6.3. Mass fluxes are then exchanged between these momentum control volumes, along the cell edges or mesh legs. Each of these mass fluxes can then be subdivided into the two sub-fluxes across the two sides of the momentum control volume adjacent to each mesh leg. The mass flux along each side of the control volume can then be obtained as $\frac{1}{2}$ the average of the upstream and downstream element edge mass fluxes, where the factor $\frac{1}{2}$ is required, as each side of the control volume only extends from the edge of each element to its centre.

The method is illustrated in Fig. 6.3 where a mass flux dmn_4 is exchanged between two regular nodes n_0 and n_4. In this example node n_0 is connected to 4 other nodes; n_1, n_2, n_3 and n_4, is surrounded by the 4 elements e_1, e_2, e_3 and e_4 and so its momentum control volume is defined as a 8 sided polygon. The mass flux required dmn_4 is subdivided into the two sub-fluxes dmn_{44} and dmn_{41}, which can be obtained, as discussed above, from the element edge mass fluxes; dme_{44}, dme_{34}, dme_{14} and dme_{12} as,

$$dmn_4 = \frac{1}{4}(dme_{14} + dme_{12} + dme_{44} + dme_{34}) \tag{6.23}$$

Once all the nodal mass fluxes have been defined, then the rest of the advection

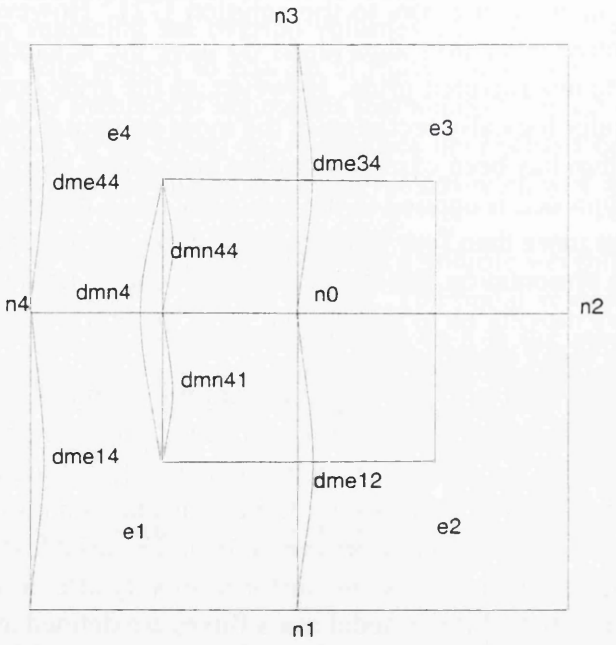


Figure 6.3: Nodal mass flux calculation for regular node.

procedure simply mirrors the description given in the previous section for the advection of element centred quantities. In this case the two velocity components are advected as two separate variables using mass weighting, the four advection fluxes required are calculated as for the element centred advection from (6.21), where ΔV_{2j} (for j=1,4) is replaced by the corresponding nodal mass fluxes, and V_{2j} (for j=1,4) is redefined in mass coordinates centred on the node. The momentum advection fluxes are then used to obtain new nodal velocity components from (6.22), where V_2^+ and V_2^- are now the pre and post advection nodal masses. The pre-advection nodal mass is defined as $\frac{1}{4}$ of the sum of the mass of the node's four adjacent elements. The post advection mass can then be obtained by allowing for the nodal mass fluxes entering and leaving.

It now remains to describe the more general momentum advection scheme that is required when a node is connected to an arbitrary number of immediate neighbours. This requires more general expressions for the nodal mass, mass fluxes and the slopes that are not tied to four immediate neighbours. However, having discussed the regular case in some detail, this now follows quite naturally. The nodal control volume is again defined as a polygon enclosing the $\frac{1}{4}$ volumes of all the elements adjacent to the node. The control volume can however now have an arbitrary number of sides as shown in Fig. 6.4. The mass flux for each mesh leg can then again be obtained from the sum of mass fluxes through the two sides of the polygon

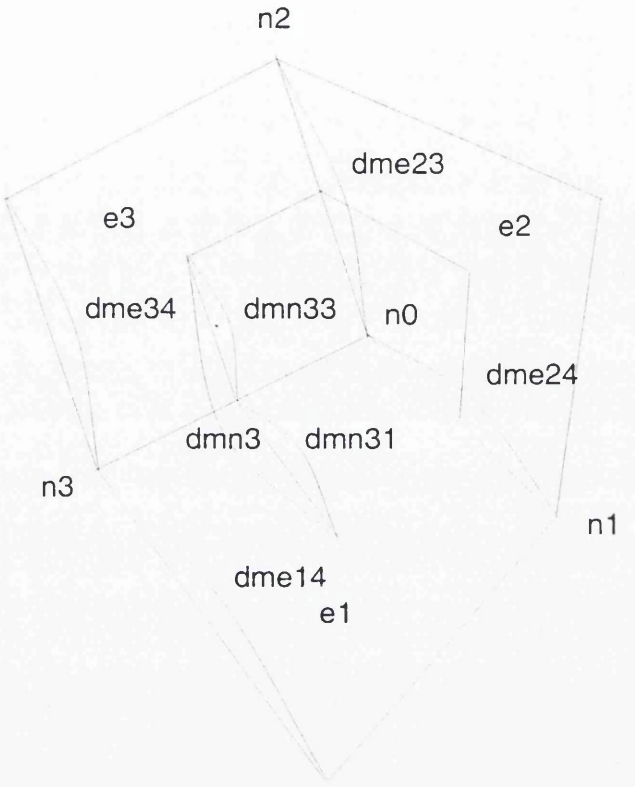


Figure 6.4: Nodal mass flux calculation for a typical irregular node with 3 mesh legs.

that are adjacent to that mesh leg. The mass flux through each side of the polygon is obtained from the element mass fluxes exactly as explained above for the regular nodes. For example in Fig. 6.4 the mass flux exchanged between nodes n_0 and n_3 is given by,

$$dmn_3 = \frac{1}{4}(dme_{14} + dme_{24} + dme_{23} + dme_{34}) \tag{6.24}$$

The only detail that remains is the evaluation of the slope for each mesh leg. A monotonically limited slope is again required, which implies that three nodal velocity values are needed. Two of these are the node to be advected and the node with which it is exchanging momentum. The third node is selected from the remaining direct neighbours. This choice is made by defining a series of mesh leg unit vectors, which point from the node being advected to its the immediate neighbours, and from the node with which momentum is being exchanged to the node being advected. The latter defines the preferred direction for the third node, that must be identified. The third node is then selected, as the node whose mesh leg unit

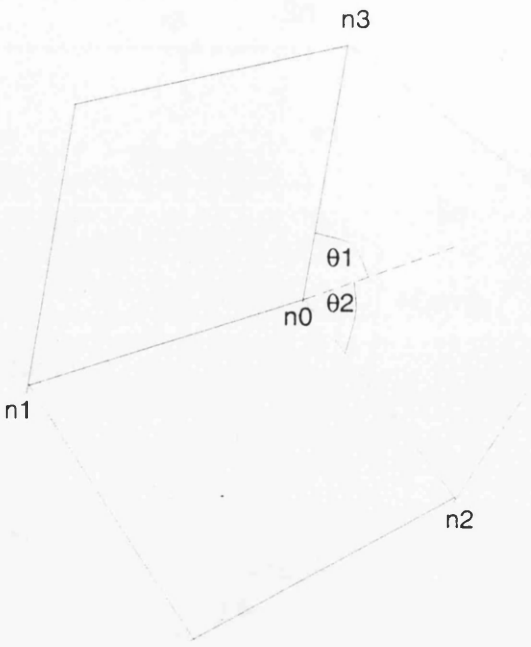


Figure 6.5: Selection of 3rd node required for monotonic slope limiting.

vector lies most in line with this preferred direction. This is illustrated in Fig. 6.5 for the calculation of an advection flux between nodes n_0 and n_1, where n_0 is the node being updated during the advection step. In this case if $\theta_1 < \theta_2$ then node n_3 is selected as the third node or if $\theta_1 >= \theta_2$ node n_2 is selected.

This completes the description of the momentum advection used in CORVUS for both single and multi-material cells. It should however be noted that whilst the current scheme conserves all the variables that are advected, it does not conserve total energy. This simply reflects the fact that if we choose to advect mass, momentum and internal energy it is not possible to also directly conserve kinetic energy and, so conservation of total energy is not achieved. However, some hydrocodes overcome the problem by also advecting kinetic energy. The difference between the advected kinetic energy and kinetic energy calculated using velocities obtained from the momentum advection, is then added to the internal energy to conserve total energy [72]. This has not been done in CORVUS as the author feels this is just trading errors in total energy for errors in internal energy. However, the issue requires further investigation.

6.2 Multi-material advection

Multi-material advection is a well established technique for Eulerian hydrocodes, but is only in use in a limited number of ALE hydrocodes. This is probably due to the increased complexity introduced when non-orthogonal grids are used. Advection schemes of this type require the relative volume of each material in each multi-material cell to be known. A set of rules is then used to determine how much of each material is advected. These rules are based either on explicitly or implicitly tracking the interfaces between the materials. Two reviews which cover most of the published interface tracking methods are given in [74, 75]. Most of these methods track the surfaces with marker particles, or reconstruct the surfaces using the volume fractions of the different materials in each cell, although a number of codes are now using level set methods.

The interface tracking problem is largely geometrical in nature as there is only one velocity field available for all the materials within each multi-material cell. This means that its impossible to calculate slip explicitly at multi-material cell interfaces. In problems containing a gas next to a solid, for example, a boundary layer will be formed in the solid at the interface with a non-zero shear stress, and the velocity profile of the gas close to the wall will be altered by the artificial ahhesion of these two material that is introduced when a multi-material cell representation is used. This is one of the reasons why a Lagrangian slide treatment has also been developed for CORVUS. However, it should be noted that methods have been developed for Eulerian codes which approximate slide for multi-material cells. However, although they have yet to be able to match the accuracy of traditional Lagrangian slide algorithms for low deformation problems, they are robust and can be applied to high deformation problems where relative slip may be important [76, 77].

6.2.1 Donor cell methods

The most popular methods used in production Eulerian hydrocodes are volume of fluid (VOF) based interface reconstruction methods, the volume fraction for each material being defined as the fraction of the cells volume occupied by each material, where the sum of all the volume fractions for each cell must always equal 1 [46]. Volume fraction based methods are attractive as the volume fraction provides a simple and economical means of describing the geometry of material interfaces. A more accurate representation could potentially be achieved using marker particles. However, the marker particle approach is both expensive and very complicated to work with.

The simplest of the volume fraction based methods do not explicitly construct material interfaces, the fraction of each material in the transport volume typically being determined by a simple averaging procedure. Although such methods may appear efficient, they will in general smear the interface significantly more than the

more advanced schemes. The best schemes should be able to resolve the width of the material layer with a volume fraction less than 1 that separate two materials to within a single computational cell. Whilst this layer may extend over several multi-material cells for the simpler schemes, this not only reduces the accuracy of the interface treatment, but also has efficiency implications, as multi-material cells are significantly more expensive computationally than single material cells. Hence, although a given multi-material cell may be less expensive to treat with a simpler multi-material advection scheme, the overall cost may be greater if the smearing it introduces creates significantly more multi-material cells in the problem.

Typical examples of the simple volume fraction based schemes include those used in PISCES [39], TOIL [78] and the original versions of CSQ [79] and JOY [80]. The method used in PISCES is a generalisation of that used in TOIL and will be described here as a typical example of these simple methods.

If we denote upstream and downstream volume fractions respectively by V_i^u and V_i^d, the upstream and downstream cell volumes by V_u and V_d, the volume transported between adjacent elements as f_i, and the volume transported for each of the k material components that is to be determined by f_i^k. Then the first step only considers materials that are common to both the upstream and downstream elements. Each of these materials is then advected in proportion to the downstream volume fraction, unless that exceeds the available material in the upstream zone. If the summation is limited to the common materials, this step can be expressed as,

$$f_i^1 = min(V_i^u V^u, \frac{f_i V_i^u}{\sum_j V_j^d}) \tag{6.25}$$

If some of the materials are exhausted during this first stage then the sum of f_i^k will be less than the total available transport volume f_i. In order to rectify this a second step is required, where the summation is further restricted to the remaining materials and $\tilde{f}$ is the remainder of the available transport volume.

$$f_i^2 = min(V_i^u V^u, \frac{\tilde{f}_i V_i^u}{\sum_j V_j^d}) \tag{6.26}$$

This step is repeated until all the transport volume has been used or there is know longer any available material which is common to both the upstream and downstream cell remaining in the upstream cell, all the increments from each of these stages being summed for each material.

A final stage is then performed where the remaining materials in the upstream cell are used to fill the transport volume.

$$f_i = \frac{\tilde{f}_i V_i^u}{\sum_j V_j^d} \tag{6.27}$$

The original version of JOY [80] used a similar procedure, but took the average value of the volume fractions to define the flux for the common materials.

$$F_i = min(V_i^u V^u, \frac{1}{2}(V_i^u + V_i^d) f_i) \tag{6.28}$$

When materials were depleted the original JOY algorithm then selected materials based upon how close they were to the original common materials to make up the remainder of the transport volume. That is, if material 2 is depleted it will then look to the material next to it in the cell. If this is already depleted it searches back through the materials in the cell in order, until it finds enough unused material to fill the transport volume.

The next level of increased complexity involved the explicit reconstruction of material interfaces in an element from the common materials in adjacent elements. Methods of this type typically only consider two materials at a time. That is the material of interest and all the other materials combined. One example of this approach is the method used in the BBC code [81], which was the precursor to the SLIC [10] algorithm. Although the basic concepts of the BBC algorithm are quite simple, its actual implementation is quite complicated.

The four adjacent elements are searched for the presence of the material of interest. This leads to a 4 bit code for each material, with either a 0 for "no" or a 1 for "yes" being used to record the presence or absence of the material in each of the 4 elements. The code has 30 possible configurations, which are reduced to eight types, three of which have subcases, which are finally divided into three classes. The first class contains material in horizontal layers, the second vertical layers and the third contains the exceptions. These are mostly where material is concentrated either at the corners or mid sides of the elements, with the thickness of the layers being taken to be proportional to their volume fractions. The material concentrated either at the corners or mid way along the sides is taken to have the same aspect ratio as the element in which it resides.

The amount of each material transported is next calculated from the geometry of the reconstructed interface and the geometry of the transport volume (i.e. volume and which edge). Since each material is considered independently, they can overlap and there is no guarantee that the sum of the individual material volume fluxes will sum to equal the transport volume. In order to overcome this problem the volume transported for the last material is chosen to be equal to the difference between the transport volume and the sum of the volumes calculated for the other materials. However, this is constrained to be positive and to not exceed the amount of the last material in the upstream element. If this constraint is not satisfied then further steps are required. These steps essentially strive to uniformly scale back the transport volumes of the other material until their sum matches the total transport volume. Adjustments are applied, as required, to ensure the scaling does not exceed the volume available for any of the materials. If this scaling does not work the actual transport volume is reduced.

6.2.2 The SLIC algorithm

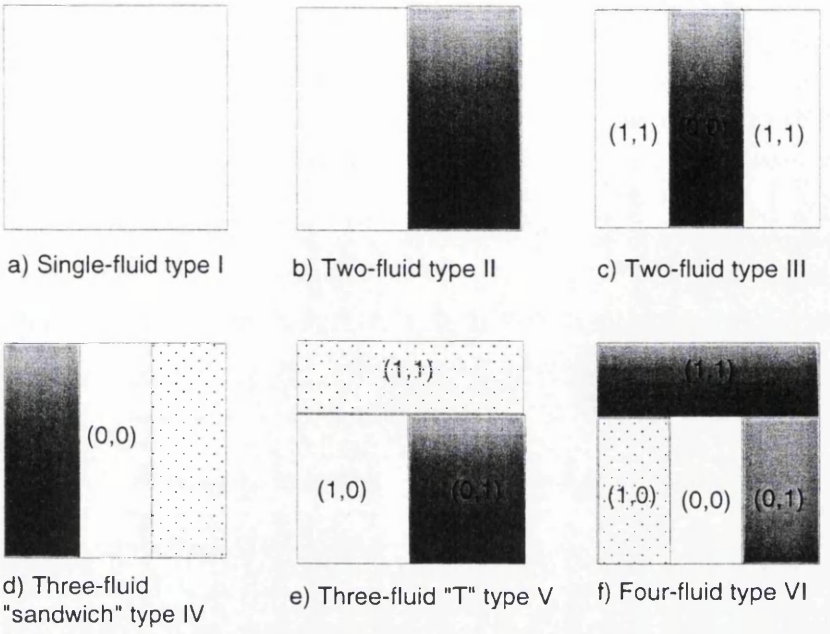
The Simple Line Interface Calculation or SLIC algorithm developed by Noh and Woodward [10, 82] is probably the best of the first order interface tracking methods. It is also the closest of the published interface tracking methods to the scheme that has been developed by the author for CORVUS. Given these two important considerations it warrants special consideration. SLIC is an alternating direction method which constructs the interface out of straight lines which are either perpendicular or parallel to the advection direction, the choice of interface topology being completely defined by testing whether or not the four adjacent elements contain the various materials. This is a similar approach to that of the BBC algorithm [81] discussed above. However, the excessive complication of the latter method has been removed, a further major advantage of the SLIC scheme being that volume transport for each material can be defined in a single pass.

Fluid occupation numbers are defined for each material i, IL_i and IR_i. These values are 0 if material i is absent and 1 if it is present in the elements to the left and right of the current element, respectively. This leads to four possible combinations or fluid groups, and so materials that have the same (IL_i, IR_i) values are treated as belonging to the same fluid group and are treated equally. Six possible configurations are allowed for these four fluid groups as shown in Fig. 6.6. The relative sizes of the rectangles used to construct these are simply obtained from the known volume fractions of the materials present. It should be noted that the interfaces are constructed independently for each advection direction, and so there is no requirement for consistency between the two directions. A modified version of the SLIC algorithm has also been developed by Chorin [83] to introduce the facility to explicitly construct interface corners.

6.2.3 Higher order VOF methods

The higher order interface reconstruction schemes in today's Volume of Fluid (VOF) based Eulerian hydrocodes employ arbitrarily oriented straight line segments to represent material interfaces. These interfaces will however almost certainly be discontinuous at element boundaries. Codes which employ high order interface reconstruction schemes of this type include CAVEAT [8], KRAKEN [73], SOLA-VOF [46], PELE [84], MESA [85], CTH [86] and AWEs codes PETRA [87] and NUTMEG . The main differences between the higher order VOF methods stem from how the interface slope is calculated and how the materials within a given multi-material cell are ordered.

The first and simplest technique for calculating the slope was proposed by Hirt and Nichols and implemented in the SOLA-VOF code [46]. Given a rectangular mesh, the height of a each material above the bottom edge of the row of elements Z is first defined, in terms of the elements dimensions, Δy and Δz, and the volume



a) Single-fluid type I
b) Two-fluid type II
c) Two-fluid type III
d) Three-fluid "sandwich" type IV
e) Three-fluid "T" type V
f) Four-fluid type VI

Figure 6.6: The six fluid-configuration types used in the SLIC algorithm.

fractions f, where the indices reflect a 2D logical mesh as,

$$Z_{i+\frac{1}{2}} = f_{(i+\frac{1}{2},j-\frac{1}{2})}\Delta z_{(i+\frac{1}{2},j-\frac{1}{2})} + f_{(i+\frac{1}{2},j+\frac{1}{2})}\Delta z_{(i+\frac{1}{2},j+\frac{1}{2})} + f_{(i+\frac{1}{2},j+\frac{3}{2})}\Delta z_{(i+\frac{1}{2},j+\frac{3}{2})} \tag{6.29}$$

Given these material heights, the slopes can then be defined as,

$$\left.\frac{\partial Z}{\partial y}\right|_{(i+\frac{1}{2},j+\frac{1}{2})} = \frac{2(Z_{(i+\frac{3}{2})} - Z_{(i-\frac{1}{2})})}{\Delta y_{(i-\frac{1}{2},j+\frac{1}{2})} + 2\Delta y_{(i+\frac{1}{2},j+\frac{1}{2})} + \Delta y_{(i+\frac{3}{2},j+\frac{1}{2})}} \tag{6.30}$$

If the calculated slope is nearly vertical then the slope $\frac{\partial Y}{\partial z}$ is calculated following a similar procedure. The interface position is then adjusted to match the cell's volume fractions.

The next increment in complexity was introduced in the KRAKEN and PELE codes, which generate a set of interfaces for each cell edge, the transport of material across each edge was then calculated using the appropriate set of interfaces. Each set of interfaces are constructed by fitting straight lines through the two cells adjacent to the edge in question, the interface slope and intercept with the edge being determined by the requirement that the area below the line in each cell should match the volume occupied by the appropriate material in that cell, as defined by the material's volume fraction. There are four different configurations that can occur (for further details see [18]).

A different strategy for calculating the interface slope was proposed for 2D problems by Youngs [87] and then later extended to provide a 3D interface reconstruction method [88]. A similar approach has now also been implemented in CTH [86]. The KRAKEN/PELE approach is used as a starting point with straight line being fitted to match the volume fractions for each pair of elements adjacent to each of the four edges, exactly as discussed above. The intercepts of these lines with each of the edges are then used to define edge fractions, $b_{(i,j+\frac{1}{2})}$. These edge fractions correspond to the distance along each edge to the intercept where the element is treated as a unit square. The edge fractions are then differenced to obtain the interface slope.

$$S_{(i+\frac{1}{2},j+\frac{1}{2})} = \frac{b_{(i+1,j+\frac{1}{2})} - b_{(i,j+\frac{1}{2})}}{b_{(i+\frac{1}{2},j+1)} - b_{(i+\frac{1}{2},j)}} \tag{6.31}$$

The position of the interface must then be adjusted while maintaining this slope, until the volume occupied by the two materials is matched. The advantages of Youngs' method over the KRAKEN/PELE approach are that it uniquely defines both the slope and position of the straight line interface for the cell, which can then be consistently used in calculating the advection fluxes across the four cell edges. Further, for straight line material interfaces the interface construction is exact.

Since there is no natural extension of concept of side fraction to three dimensions, Youngs has taken a slightly different approach for 3D. In this case Youngs calculates the normal to the interface by differencing the volume fractions of the surrounding elements. The position of the interface is then adjusted by translating it in the direction of the normal, as in 2D. In order to illustrate Youngs' method for the 3D slope calculation we consider the normal calculation in 2D. A uniform orthogonal structured 2D mesh is again assumed, with the volume fractions of all eight of the elements neighbours having some influence over the interface slope. Effective volume fractions are then expressed for four principal directions, N, E, S and W as,

$$\begin{aligned}
f_E &= \frac{f_{(i+\frac{3}{2},j+\frac{3}{2})} + \alpha f_{(i+\frac{3}{2},j+\frac{1}{2})} + f_{(i+\frac{3}{2},j-\frac{1}{2})}}{2+\alpha} \\
f_W &= \frac{f_{(i-\frac{1}{2},j+\frac{3}{2})} + \alpha f_{(i-\frac{1}{2},j+\frac{1}{2})} + f_{(i-\frac{1}{2},j-\frac{1}{2})}}{2+\alpha} \\
f_N &= \frac{f_{(i-\frac{1}{2},j+\frac{3}{2})} + \alpha f_{(i+\frac{1}{2},j+\frac{3}{2})} + f_{(i+\frac{3}{2},j+\frac{3}{2})}}{2+\alpha} \\
f_S &= \frac{f_{(i-\frac{1}{2},j-\frac{1}{2})} + \alpha f_{(i+\frac{1}{2},j-\frac{1}{2})} + f_{(i+\frac{3}{2},j-\frac{1}{2})}}{2+\alpha}
\end{aligned} \tag{6.32}$$

The components of the normal vector for the interface are then obtained by differencing these quantities as follows,

$$\begin{aligned} \frac{\partial f}{\partial x_1} &= \frac{f_E - f_W}{2\Delta x_1} \\ \frac{\partial f}{\partial x_2} &= \frac{f_N - f_S}{2\Delta x_2} \end{aligned} \tag{6.33}$$

Youngs' method has also been extended to non-orthogonal meshes and axisymmetric geometry for use in CAVEAT by Johnson [89]. Johnson's extension of Youngs' method evaluated (6.33) in logical mesh coordinates, L_i, and then transformed the normal vector back into real coordinates by inverting,

$$\begin{bmatrix} \frac{\partial f}{\partial x_1} \\ \frac{\partial f}{\partial x_2} \end{bmatrix} = \begin{bmatrix} \frac{\partial L_1}{\partial x_1} & \frac{\partial L_2}{\partial x_1} \\ \frac{\partial L_1}{\partial x_2} & \frac{\partial L_2}{\partial x_2} \end{bmatrix} \begin{bmatrix} \frac{\partial f}{\partial L_1} \\ \frac{\partial f}{\partial L_2} \end{bmatrix} \tag{6.34}$$

Once the normal vector has been determined it then remains to adjust the position of the interface again to match the known volume fractions for the cell in question. However, while for Youngs' original scheme as applied to orthogonal grids, the position could be calculated explicitly. In axisymmetric geometry with a non-orthogonal grid, the location of the interface is given by a cubic equation which Johnson solves using Newton iteration.

McGlaun and Thompson [86] use a contouring algorithm to reconstruct interfaces in CTH. An average volume fraction is assembled at element corners from the adjacent element values. Then a set of straight line volume fraction contours are constructed for the element. The contour which matches the known volume fractions for the element is then selected for the material interface.

Once the interface slope and position has been defined the advection step can be performed. First order or donor cell differencing is used in all the codes that the author has encountered. This means that problem becomes a geometric one of determining how the fluxing volume is divided by the interfaces that have been constructed. This determines the individual volume fraction fluxes for each material across each element face. This is clearly well defined but involves reasonably complicated logic which increases rapidly in moving from structured orthogonal grids to unstructured non-orthogonal grids with the computational cost increased even more rapidly when the method is applied to 3D unstructured non-orthogonal meshes. Although increasing the accuracy of the multi-material cell advection above first order is clearly attractive, however, the problem is far from trivial: the natural extension that is compatible with the single material advection in CORVUS, for example, would require monotonic slopes to be defined for the flow variables in the two isoparamatric directions across the multi-material cell. However, while the pressure and particle velocity are continuous across a material interface, other variables

such as density and internal energy are not guaranteed to be. Hence separate slopes would be required for each material component, and there is insufficient information to define these.

6.2.4 Handling more than two materials

The high order VOF based algorithms for explicitly calculating interfaces have all been described in terms of two materials. There are two natural ways to extend these methods to an arbitrary number of materials. However, none of these explicitly handle 'T' junctions and the materials are always assumed to lie in continuous layers of material that can overlap.

The simplest approach is to calculate the interface between each material in turn and a mixture of all the others. The second approach requires the ordering of the materials through the cell to be defined. This could be a user defined priority list. An interface is then constructed between the first material and a mixture of all the others. The next interface is then calculated between a mixture of the first and second materials, and a mixture of all the remaining materials, and so on, until all the interfaces have been constructed. The second approach does not eliminate the problem of materials intersecting, but it does minimise this problem and enables isolated thin layers which are less than an element thick to be tracked accurately.

6.3 Multi-material advection in CORVUS

Higher order VOF schemes such as Youngs' method can be modified to make them compatible with non-orthogonal grids, as discussed above, but this makes these methods computationally expensive, especially for 3D. Given that it is planned that the adaptive multi-material ALE algorithm will be extended to 3D in the future a new computationally less expensive interface scheme has been developed for CORVUS. The aim with the new scheme is to resolve material interfaces with at least equivalent, if not greater, accuracy on an ALE grid than can be achieved with a high order VOF scheme on an orthogonal Eulerian mesh. It is accepted that the new scheme will not however perform as well as the higher order VOF schemes when both schemes are applied on orthogonal Eulerian grids.

The starting point for this new scheme was the CALE interface scheme developed at LLNL by Tipton [19]. Since it has been so influential on the development of the CORVUS interface scheme, a brief description will first be given of the CALE method before proceeding to describe the CORVUS scheme in detail. The problem to be solved for any multi-material advection scheme is how to divide the fluxing volumes amongst the materials present in each donor cell. The CALE scheme breaks down this advection problem into different flow topologies like the SLIC algorithm, but only allows two flow topologies: serial and parallel. It

is this simplification which vastly reduces the complexity of the advection method for non-orthogonal grids compared to the high order VOF methods. The choice of flow topology is made separately for each material and for both mesh directions. An initial selection is made from the slope of the material interface. However, this may then be over ruled depending on the results of a series of heuristic tests which attempt to detect corners. If the serial flow topology is selected for a material then it is advected in preference to all other serial materials behind it, while all the materials with a parallel flow topology are given equal preference, although some account is given to the inclination of its interface. A final normalisation step is then performed to ensure that the total volume of material advected at each face matches the overlap volume.

The CALE algorithm was initially implemented in CORVUS, essentially as described above, except that it had to be modified to make it work with the isotropic single material advection method used in CORVUS. CALE is a logical mesh code, so it is able to use Strang operator splitting [69] to reduce advection corner coupling errors. This breaks the advection step down into two separate sweeps in the x and y directions, so there is no risk that more material in total can be advected out of a cell than is initially present. However, this can occur with isotropic multi-material advection, if measures are not taken to avoid it. This problem was solved in CORVUS by simply storing the initial volume fraction for each material at the start of the multi-material advection step as a temporary variable. The volume fraction fluxes were then calculated first for one direction, where the volume fraction flux out of the cell was then limited to not exceed the value held of this temporary variable. The temporary volume fraction variable is then updated to hold any remaining volume fraction. The volume fraction fluxes are then calculated for the remaining direction, again limiting the flux out of the cell to not exceed the value stored in the temporary volume fraction variable. At each time step the mesh direction whose advection flux is calculated first is alternated to avoid any directional bias.

This approach was found to work well in most applications, and gives solutions almost as good as Youngs' scheme for orthogonal grids. However, noisy step interfaces were obtained for convergent flow problems involving long thin aspect ratio zones. In addition, expanding isolated thin layers where observed to break up artificially. These problems were eventually traced to a number of deficiencies with the CALE algorithm. Noisy interfaces on long thin zones were traced to be due to a combination of the heuristics, which can override the choice of flow topology and lead to the serial flow topology being applied to the same material on adjacent faces: that is, a corner is detected, which is not real. However, this noise was also increased by the choice of volume fraction slope used for the parallel advection scheme in CALE. The poor treatment of thin layers was traced to how the multi-dimensional slope was calculated in zones containing more than two materials.

The new CORVUS multi-material advection algorithm which has been developed to address these issues will now be described in detail. The two material cases

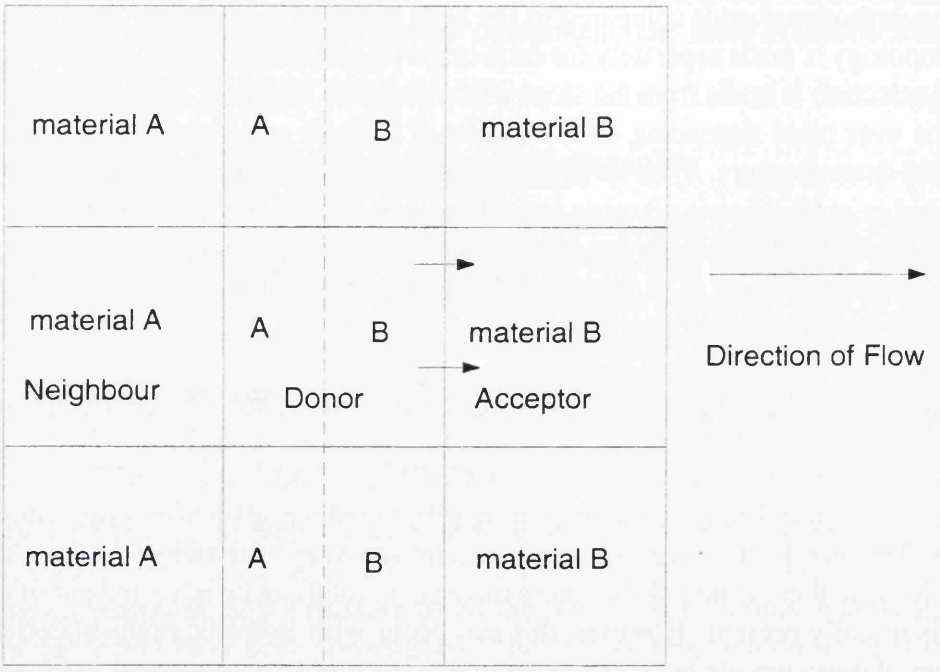


Figure 6.7: Serial flow topology.

will be assumed initially, then the scheme will be generalised to an arbitrary number of materials.

The serial and parallel flow topologies are illustrated in Figures 6.7 and 6.8. These illustrate the main ideas that for the parallel topology, the two materials are advected symmetrically, while for the serial case, all of material A must be advected before any of material B can be advected.

In CORVUS the choice of flow topology is made purely on the basis of the interface slope relative to the mesh. This enforces adjacent edges to always have opposite flow topologies. If only two materials are present the CORVUS scheme calculates the interface slope S using the volume fractions of the cell's four immediate neighbours; FN_m, FA_m, FU_m and FL_m as defined in Fig. 6.9.

$$S = \frac{|FN_m - FA_m|}{|FU_m - FL_m|} \tag{6.35}$$

If this slope S is greater than 1 then serial flow is assumed, otherwise the parallel flow topology is used.

If more than two materials are present then slopes are calculated for all the interfaces using an onion skin model. The code user must define a priority list for the materials in the problem. This simply defines the order that the materials should

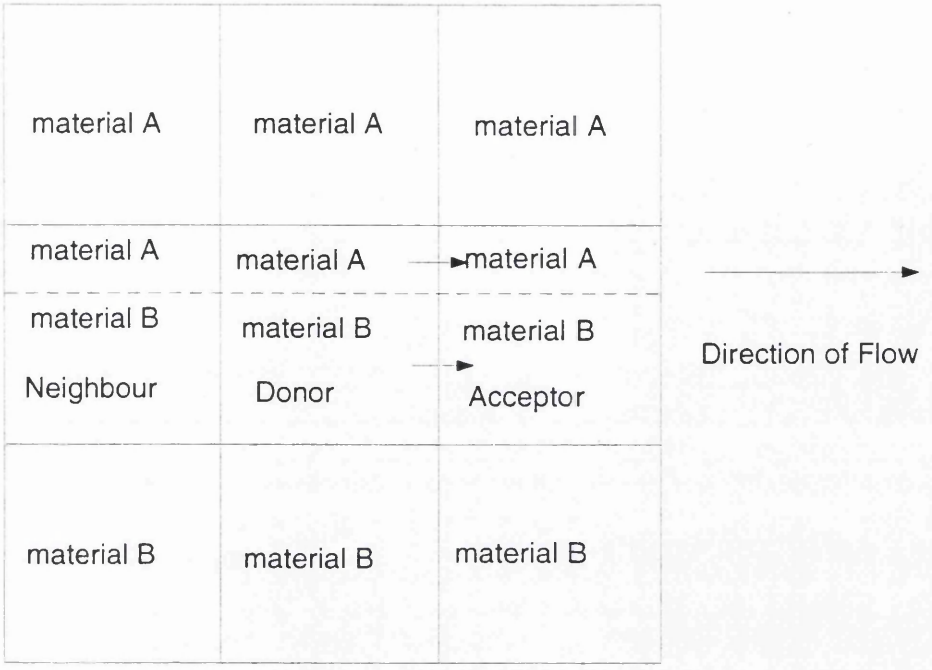


Figure 6.8: Parallel flow topology.

be encountered and allows an integrated volume fraction variable, $\bar{F}_m$, to be defined for each material m in terms of a sum of the first m volume fraction, F_m, that is,

$$\bar{F}_m = \sum_{i=1,m} F_i \tag{6.36}$$

The multi-dimensional slope is then calculated from (6.35) for each material interface in each multi-material cell using the integrated volume fraction, which always provides a well defined slope for all material interfaces, even those associated with isolated thin layers. Given that the flow topology of isolated layers should depend on the topology of the two interfaces that enclose it, the average of the two interface slopes on either side of isolated materials is then used to determine their flow topology, whilst for the first and last materials in the cell, the flow topology is uniquely determined from the slope of a single material interface.

Once the flow topology has been defined for the element, the fraction of the transfer volume ΔV corresponding to each material, η_m, can be determined for each of the four cell edges. If the material is to the front of the cell and a serial flow topology has been selected then as much of the material is advected as the transport volume requires. This however is limited to not exceed the remaining material present in the donor cell as discussed above. Materials to the back of the cell are

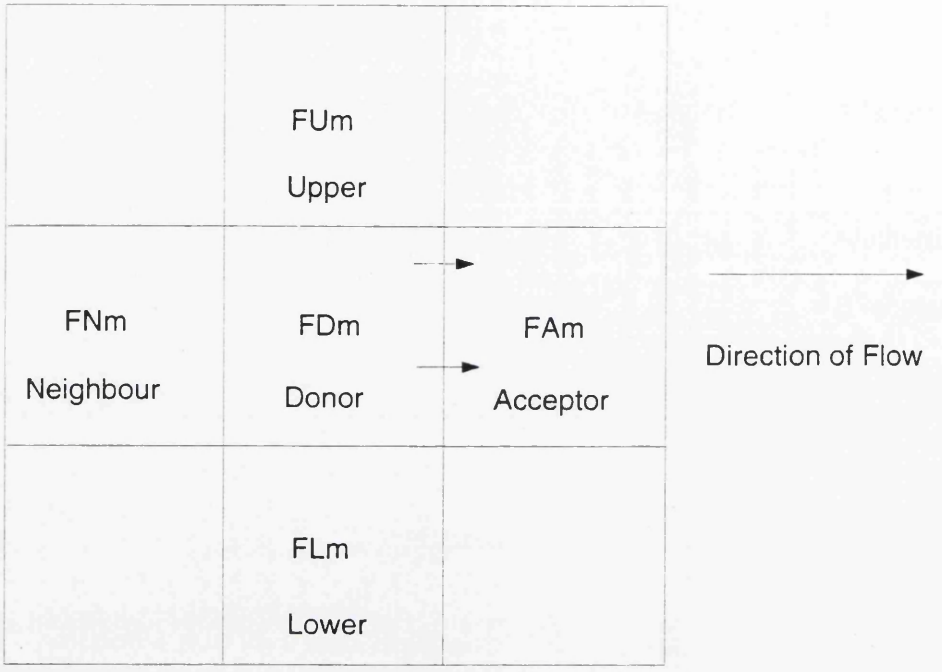


Figure 6.9: Notation for interface slope calculation.

not advected until the materials at the front have been exhausted.

$$\begin{aligned} &if \quad (FA_m > FN_m) \quad \eta_i = \frac{FD_m V_d}{\Delta V} \\ &else \quad \eta_m = 0 \end{aligned} \tag{6.37}$$

If there is more of the material in the neighbour than in the donor cell, and only a small amount of the material in the acceptor, then it is not advected preferentially, even though the material may be towards the front of the cell.

$$\begin{aligned} &if \quad ((FN_m >= FD_m) \\ &and \quad (FA_m < 1.0e^{-6})) \quad \eta_m = 0 \end{aligned} \tag{6.38}$$

If the parallel flow case is selected, then the materials on either side of the interface are advected with equal preference. However, in contrast to the SLIC scheme, some allowance is made for the inclination of the interface in an attempt to raise the accuracy of the interface treatment towards second order for this flow topology. Three volume fraction slopes are defined in volume coordinates following a similar approach to that used for the single material advection in CORVUS: the first slope being taken across the acceptor face, the second across the neighbour cell face and

third obtained by taking the slope at the centre of the donor cell of a parabolic fit taken through the neighbour, donor and acceptor volume fractions. If V_a is the acceptors cell volume, V_n the neighbours and V_d the donor cells volume, then these three slopes are given by,

$$\frac{\partial F}{\partial V_a} = \frac{FA_m - FD_m}{\frac{1}{2}(V_d + V_a)} \tag{6.39}$$

$$\frac{\partial F}{\partial V_n} = \frac{FD_m - FN_m}{\frac{1}{2}(V_n + V_d)} \tag{6.40}$$

$$\begin{aligned} \frac{\partial F_d}{\partial V} &= \frac{(F_n - F_d)\Delta V_a^2 + (F_d - F_a)\Delta V_n^2}{\Delta V_a \Delta V_n (\Delta V_a + \Delta V_n)} \\ \Delta V_a &= \frac{1}{2}(V_d + V_a) \\ \Delta V_n &= \frac{1}{2}(V_n + V_d) \end{aligned} \tag{6.41}$$

The slope from the parabolic fit is then made monotonic by applying the van Leer's limiter (6.17) used for the single material advection, which becomes,

$$\frac{\partial F^{'}}{\partial V} = \frac{1}{2}(sgn(\frac{\partial F}{\partial V_a}) + sgn(\frac{\partial F}{\partial V_n}))min\left(\left|\frac{\partial F}{\partial V_d}\right|, \left|\frac{\partial F}{\partial V_a}\right|, \left|\frac{\partial F}{\partial V_n}\right|\right)$$

Once the volume fraction slope has been obtained, a volume V_x is defined as a measure, in volume coordinates, of the distance from the centre of the donor cell to the centre of the transfer volume ΔV,

$$V_x = \frac{1}{2}(V_d - \Delta V) \tag{6.42}$$

A second order estimate of the volume fraction at the centre of the transfer volume is then calculated, and used to define the fraction of the transfer volume that the material of interest occupies, η_i. This is obtained in terms of this distance measure and the volume fraction slope as,

$$\eta_m = FD_m + \frac{\partial F^{'}}{\partial V} V_x \tag{6.43}$$

As discussed for the serial flow case, care must be taken to ensure that more material is not advected than there is remaining in the donor cell. In CORVUS this is again achieved through the use of a temporary variable which keeps track of the

volume fraction of each material, reducing it by the volume fraction flux leaving the donor cell through each face as they are calculated.

The η_m's for each cell face must sum to unity. This is clearly not guaranteed since the total flux through each face may contain some materials exhibiting serial flow and other materials experiencing parallel flow. A normalisation phase is therefore required for the η_m's corresponding to each donor cell face. The normalisation procedure is divided into three stages. The first simply defines a normalisation factor for the n materials in the cell, η_{norm}, where

$$\eta_{norm} = \frac{1}{\sum_{m=1,n} \eta_i} \tag{6.44}$$

If the original η_m is significant ($\eta_m > 1.0e^{-80}$) then it is multiplied by η_{norm}. This is of course still subject to the limit that more material cannot be advected out of the donor cell than there is remaining.

A second pass is then made over the middle materials: that is, all the materials which exhibit serial flow and are not either at the front or back of the donor cell. This means that the materials of interest must not have been involved in the first pass of the normalisation procedure, so $\eta_m < 1.0e^{-80}$. The definition used to define a middle material is that it exhibits serial flow ($S > 1$), the donor cannot be almost clean ($FD_m < (1 - 1.0e^{-10})$), there must be more of the material in the donor than in either the neighbour ($FD_m > FN_m$) or acceptor ($FA_m < 0.01$) and there must be less material present in the donor than in either the neighbour ($FN_m < 0.01$) or the acceptor ($FA_m < 0.01$).

If material m is determined to be a middle material from this definition, then, as much of the remaining transfer volume as remains after the first pass, is given to the material as is possible, without exceeding the volume of the material that remains in the donor cell and ensuring that $\eta_m >= 0$.

The third and final pass then loops over the all the materials present in the donor cell in the priority order defined by the user, and gives away the remaining transfer volume. The only constraint on the final pass is that the remaining volume of each material in the donor cell still cannot be exceeded. The sum over all materials of η's for each face should always be unity. The volume fluxes, $\Delta v_{i,j,m}$, for material m across face j of the i th element are now uniquely defined in terms of $\eta_{i,j,m}$ and transfer volume across face j $\Delta V_{i,j}$ as,

$$\Delta v_{i,j,m} = \eta_{i,j,m} \Delta V_{i,j} \tag{6.45}$$

6.4 ALE Test problems

Many of the test problems that have been used to verify the ALE capability in CORVUS have already been presented as pure Lagrangian test problems. This is

deliberate as it also enables a direct comparison of such results, further increasing confidence in the ALE results and illustrating the benefits of the ALE approach.

6.4.1 Sod's shock tube

Definition

Sod's shock tube problem was fully described in section 3.7.1, so it just remains to discuss the differences in how the ALE calculations were performed. The ALE calculations were all performed using monotonic artificial viscosity and Eulerian mesh motion. Eulerian grid motion was used to allow direct comparison with published results from Eulerian schemes. The first calculation was performed without interface tracking on a 1x100 mesh. The remaining two were both performed with interface tracking, one on a 1x100 mesh and the other on a 1x500 mesh.

Numerical Results

The solution obtained without interface tracking (dotted line) on a 1x100 mesh is given in Fig. 6.10. It compares favourably with the analytical solution (solid line) also given in Fig. 6.11 except close to the contact discontinuity. The constant states are all in good agreement with the analytical solution and the rarefaction is well matched except for some rounding at the head and tail. There are no signs of post shock oscillations, just a small disturbance at the site of the burst diaphragm, which is due to a wall heating, or start up error when the shock is initially formed. The contact discontinuity is however smeared over 2-3 zones.

The solution with interface tracking (dotted line) on a 1x100 mesh is compared with the analytical solution (solid line) in Fig. 6.11. This is a very similar result to that in Fig. 6.10 as expected, but in this case the interface tracking captures the contact discontinuity over 1 to 2 zones.

The solution on a 1x500 mesh (dotted line) is compared with the analytical solution (solid line) in Fig. 6.12. It is now very difficult to distinguish between the numerical and analytical solutions, although the main deficiency is still present at the burst diaphragm site it is very small. In practice the solution is essentially unchanged, with further increases in mesh resolution, and is considered to be mesh converged at this resolution.

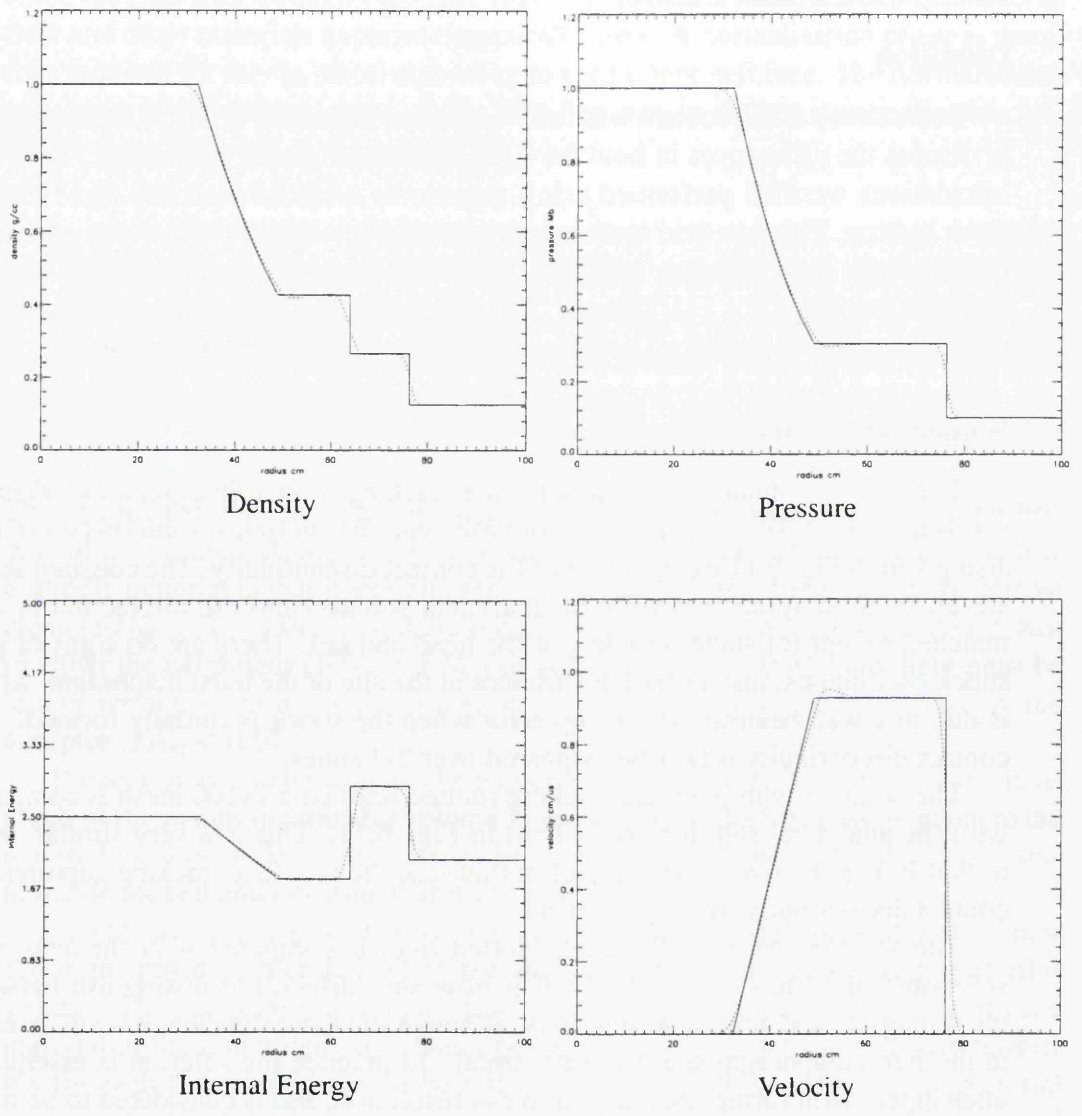


Figure 6.10: Sod's shock tube problem calculated with Eulerian mesh motion and monotonic artificial viscosity on 1x100 cell mesh at 15.0 μs without interface tracking.

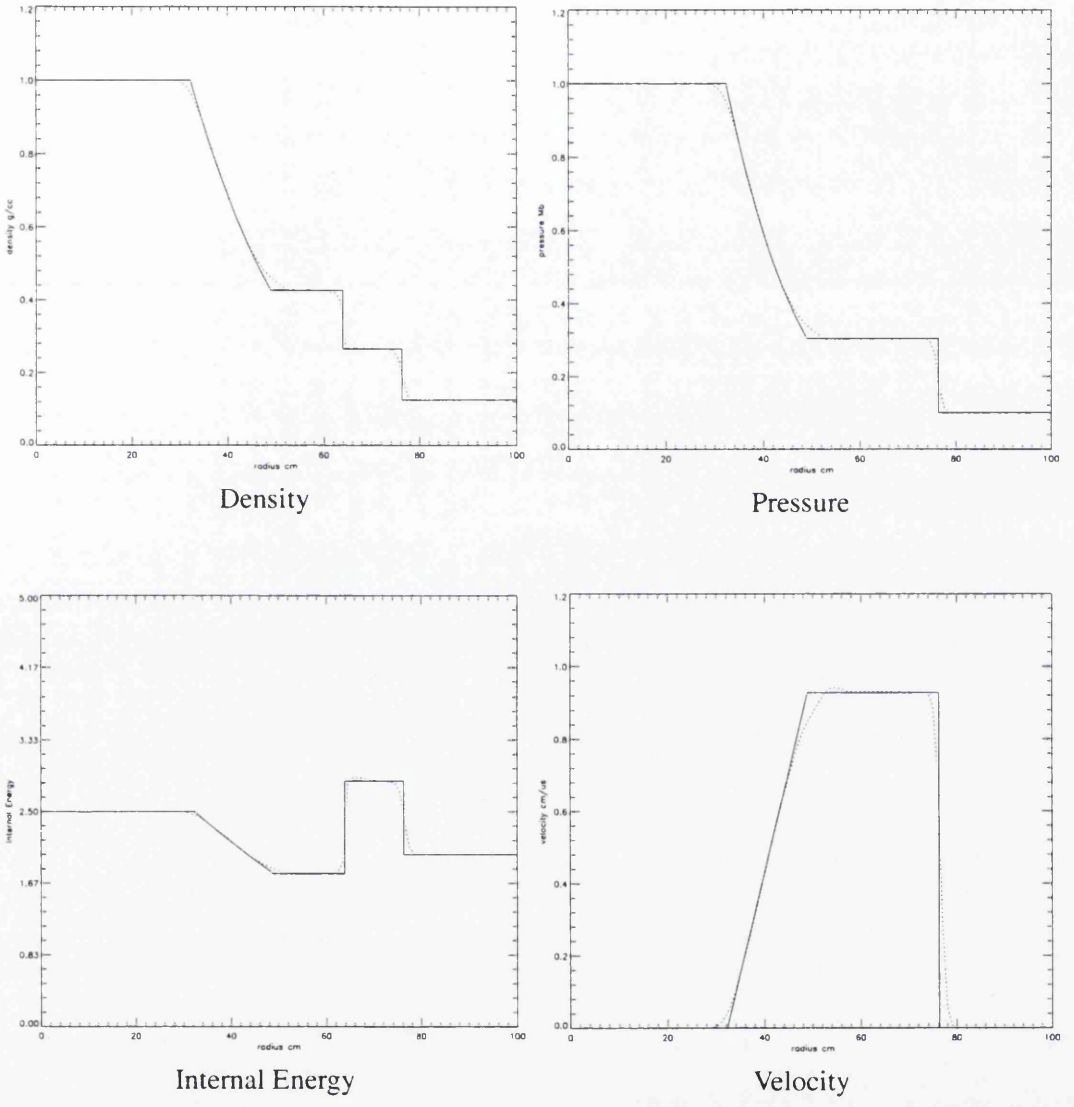


Figure 6.11: Sod's shock tube problem calculated with Eulerian mesh motion and monotonic artificial viscosity on 1x100 cell mesh at 15.0 μs.

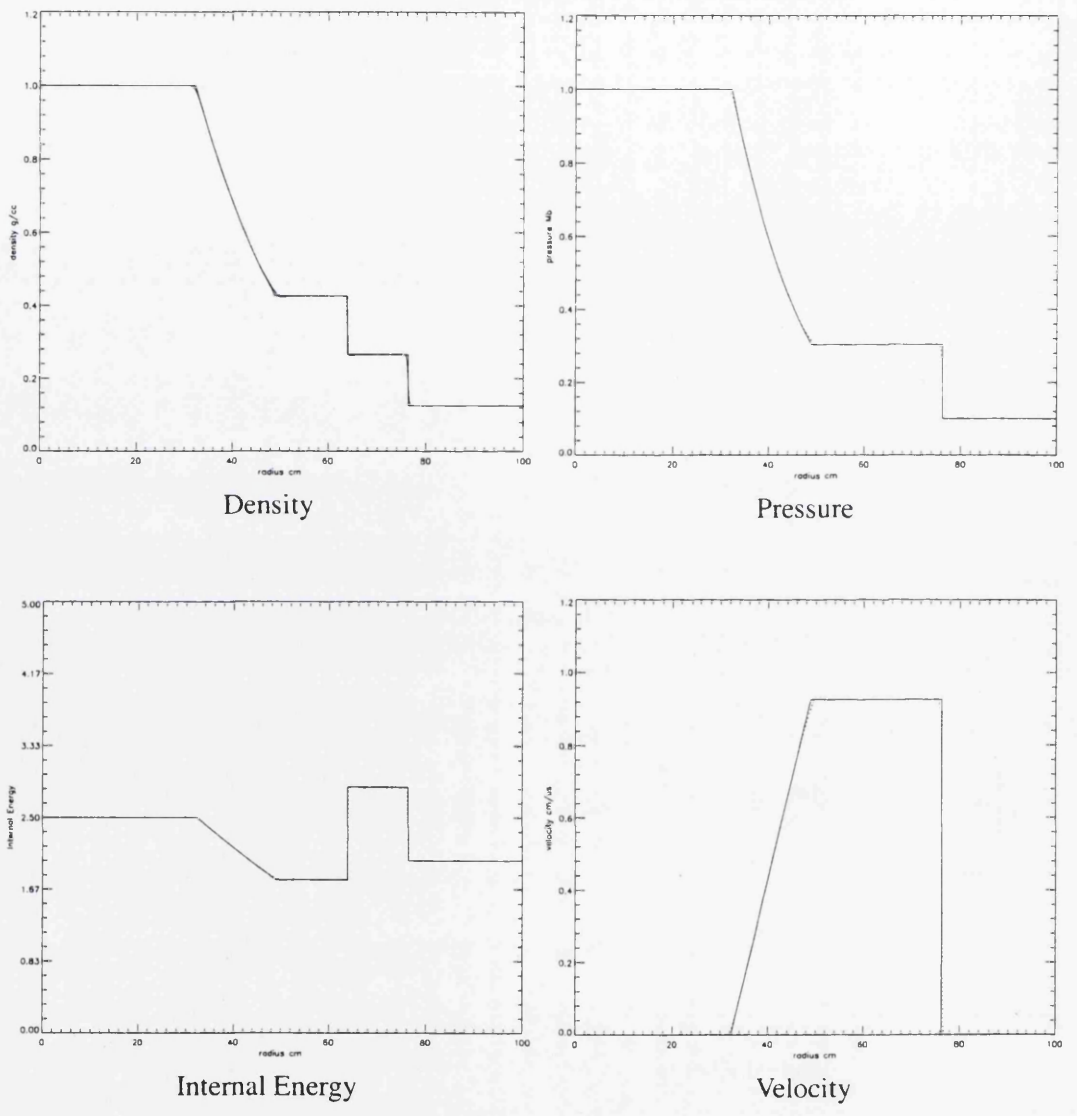


Figure 6.12: Sod's shock tube problem calculated with Eulerian mesh motion and monotonic artificial viscosity on 1x500 cell mesh at 15.0 μs.

6.4.2 Berylium stopping shell

Definition

The Berylium stopping shell problem was described in section 3.9.1, so it just remains to describe any differences in how the ALE calculations were performed. Two versions of the test problem have been calculated both with 2° angular zoning. The first retains a Lagrangian outer interface, but with the central void zoned with 100 radial zones, so that multi-material advection could be used across the inner interface. The second applies multi-material advection to both the inner and outer interfaces. In this case a second 3 cm thick void region was added outside the Berylium shell. In both calculations Winslow's equipotential mesh movement algorithm was used, with weights applied to maintain a mesh with equal spacing in radius and angle. The latter problem then becomes a non-orthogonal Eulerian calculation.

Numerical Results

The initial mesh for the first calculation is given in Fig. 6.13 and the corresponding mesh at $100.0\mu s$ in Fig. 6.14. An initial mesh for the second ALE calculational methodology is given in Fig. 6.15 and its mesh at $100.0\mu s$ in Fig. 6.16. The first ALE route stops at about 2.99 cm and it's total energy increases by 0.02 % during the calculation. The second ALE route also stops at about 2.99 cm and the total energy in the problem increases by 0.015 %. The advection scheme does not conserve kinetic energy exactly, although it will conserve mass, momentum and internal energy. This results in the ALE solutions not conserving total energy as well as the Lagrangian scheme. However, the errors in total energy for both approaches are very small and considered insignificant for practical calculations. The stopping radius obtained in both these ALE calculations was also closer to the analytical value, and the energy conservation was improved over the previous Eulerian results obtained at AWE for orthogonal and spherical geometry calculations [45].

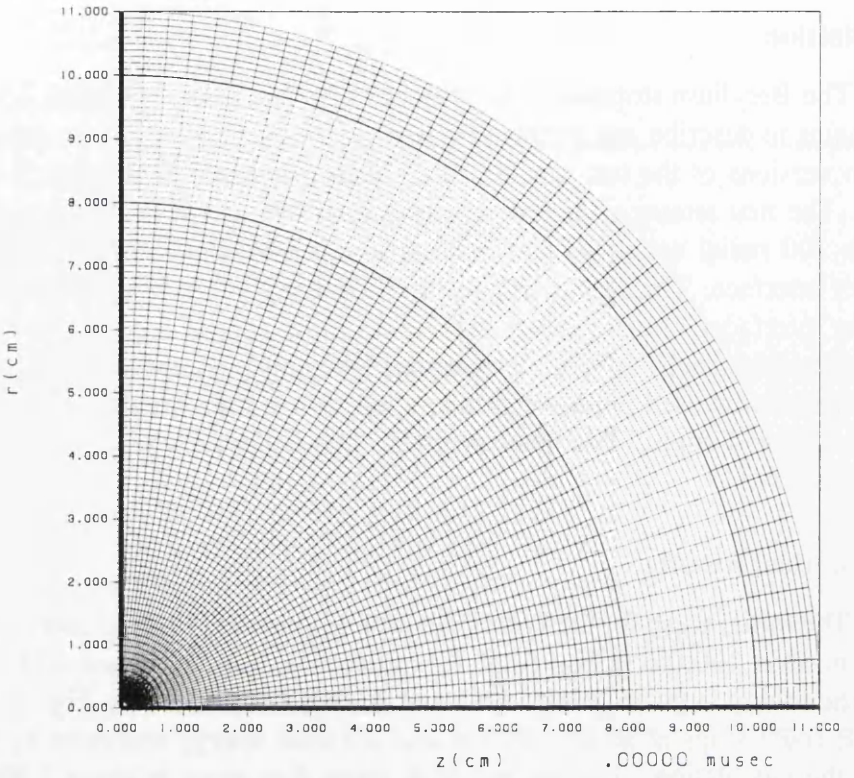


Figure 6.13: Initial mesh for Berylium stopping shell problem for Eulerian mesh motion.

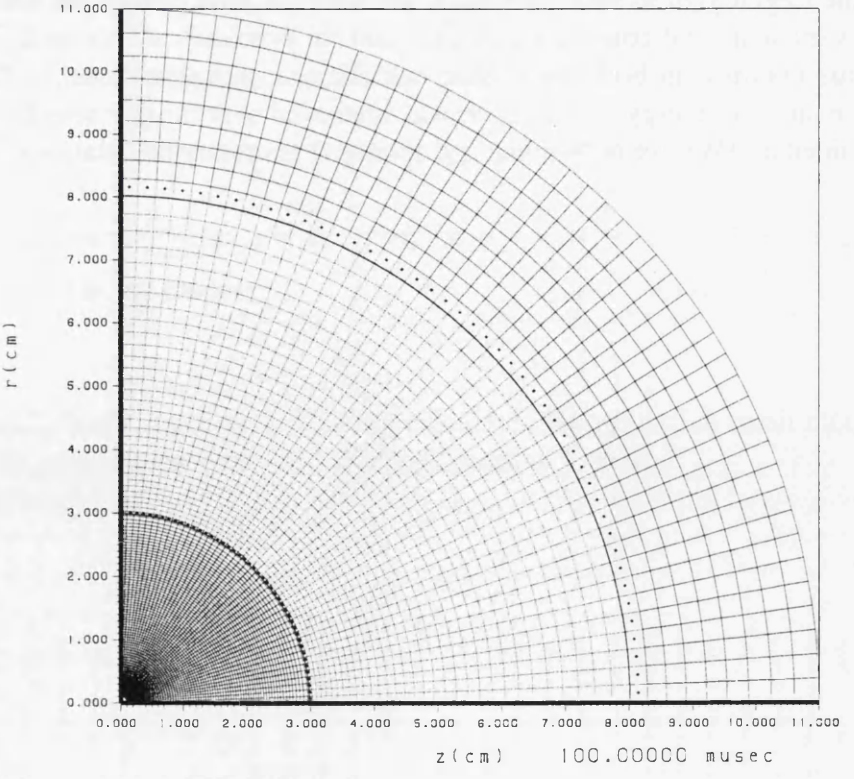


Figure 6.14: Mesh for Berylium stopping shell problem at 100.0 μs with Eulerian mesh motion.

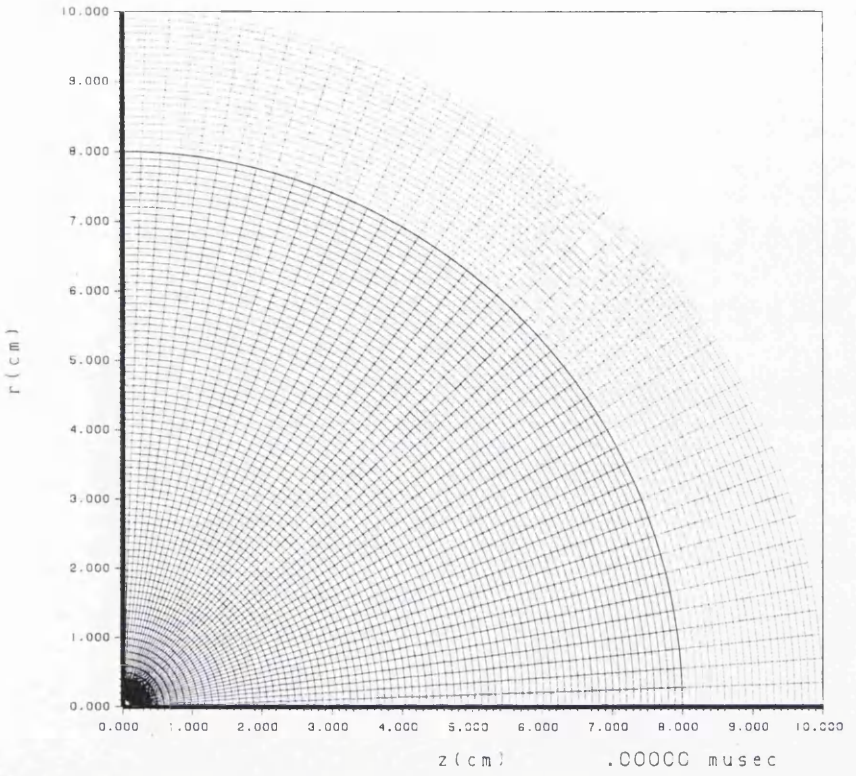


Figure 6.15: Initial mesh for Berylium stopping shell problem for Equipotential mesh motion.

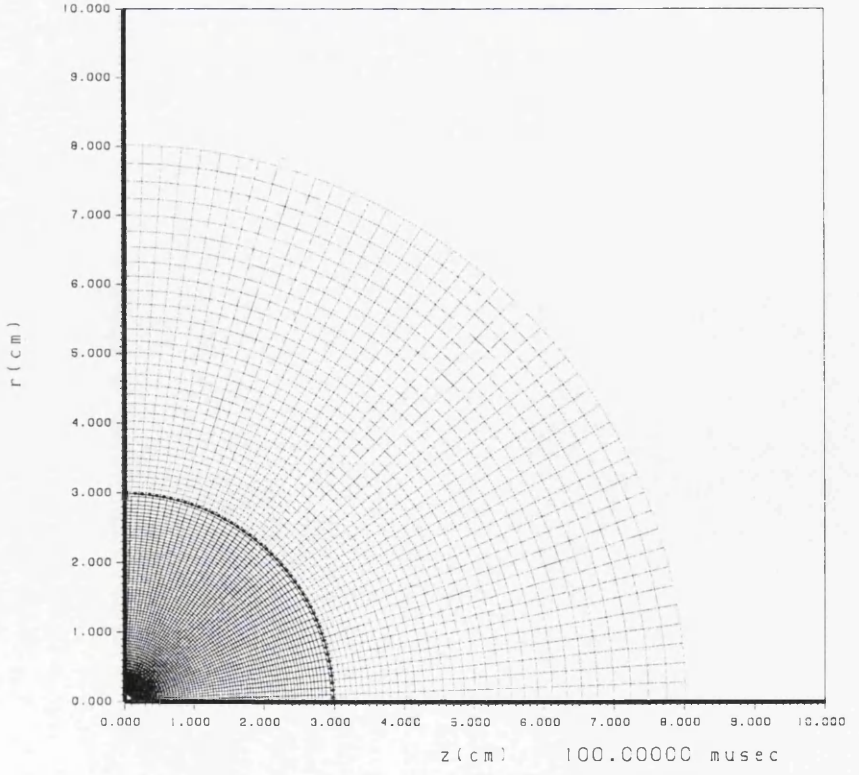


Figure 6.16: Mesh for Berylium stopping shell problem at 100.0 μs with Equipotential mesh motion.

6.4.3 Taylor rod impact test

Definition

The Taylor impact test was fully defined in section 3.9.2. It has also been calculated in ALE mode using Eulerian grid motion.

Numerical Results

Fig. 6.17 and Fig. 6.18 show the ALE solution calculated with Eulerian grid motion on CORVUS. The interface profiles and plastic work contours are in good agreement both with the Lagrangian solution presented in Fig. 3.15 and Fig. 3.16 and with previous results obtained using Eulerian codes at AWE [45]. The reflex profile of the rod after impact and the position of the plastic wave both also agree well with 1D theory.

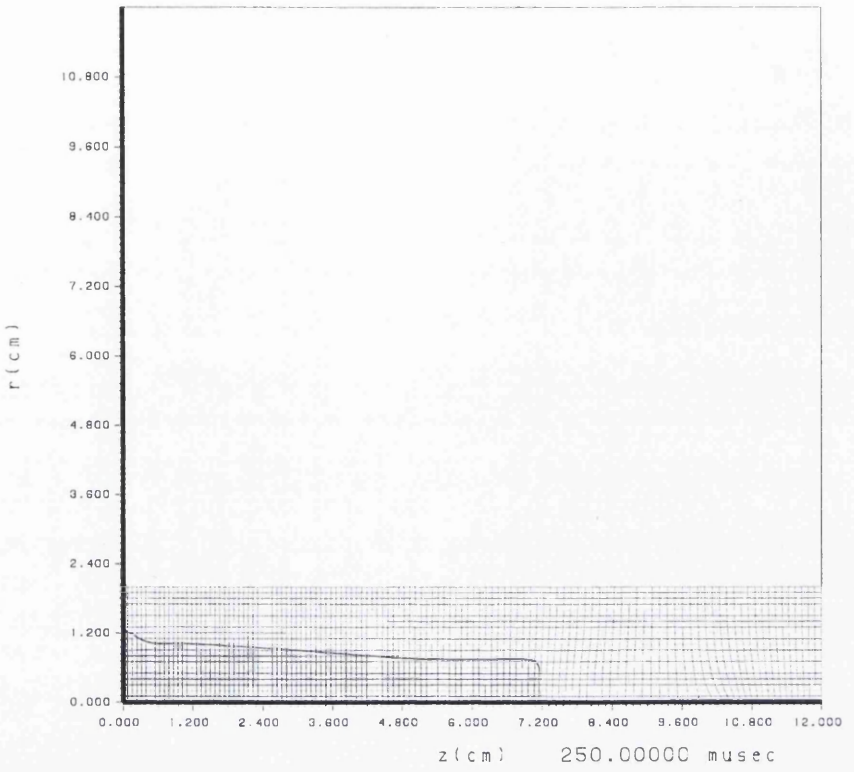


Figure 6.17: Mesh for Taylor copper rod impact test at 250.0 μs calculated with Eulerian mesh motion.

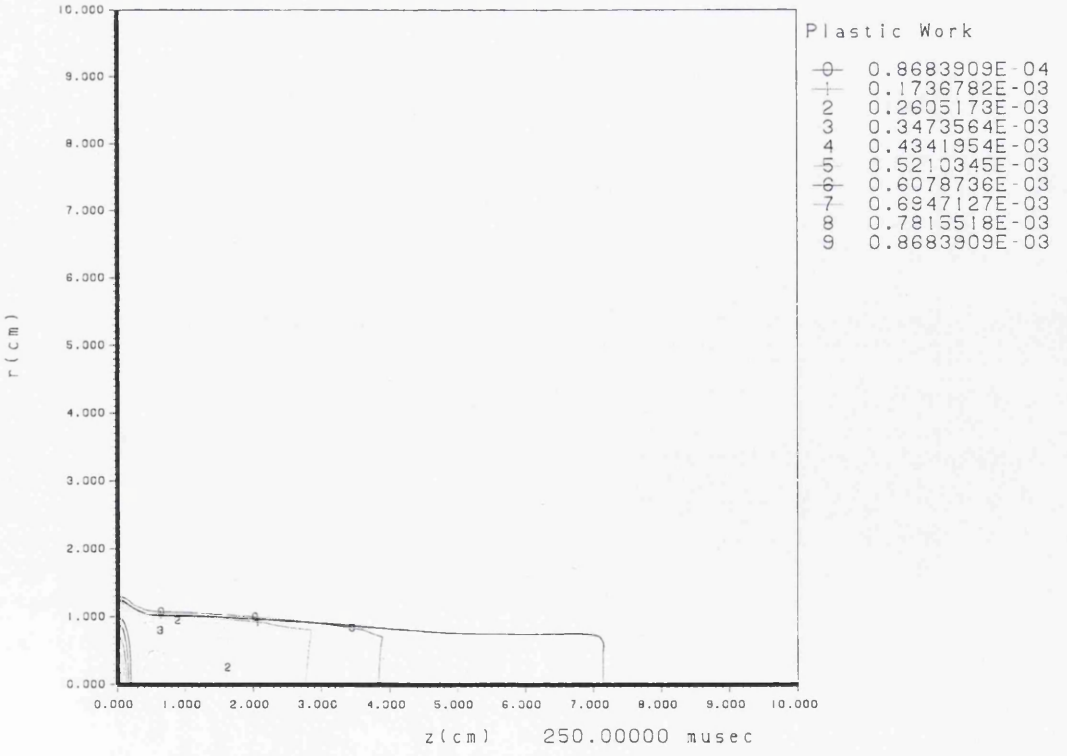


Figure 6.18: Plastic work contour plot for Taylor copper rod impact test at 250.0 μs calculated with Eulerian mesh motion.

6.4.4 Berylium vibrating plate test problem

Definition

The Berylium vibrating plate test problem was fully defined in section 3.9.3. It has also been calculated in ALE mode with Eulerian mesh motion on a 0.1 cm resolution mesh to give comparable resolution to the Lagrangian calculations of the problem.

Numerical Results

A 30.0μs period and an amplitude of about 0.5 cm was again obtained as illustrated in Fig. 6.19. The solutions are in good agreement with previous Eulerian and Lagrangian code results. It is not surprising that the calculated period is longer than the analytical prediction, as the theory assumes long thin plates, while the numerical simulation is applied to a 6:1 aspect ratio plate. The energy conservation obtained is reasonable, with the ALE solution having lost 2.5 % of its total energy by 60.0μs compared to the 0.15 % observed for the pure Lagrangian simulations presented in chapter 3. It also agrees closely with the quoted energy losses for AWE's Eulerian codes [45]. If a more Lagrangian mesh movement strategy was used in the ALE calculation, this error would probably be reduced.

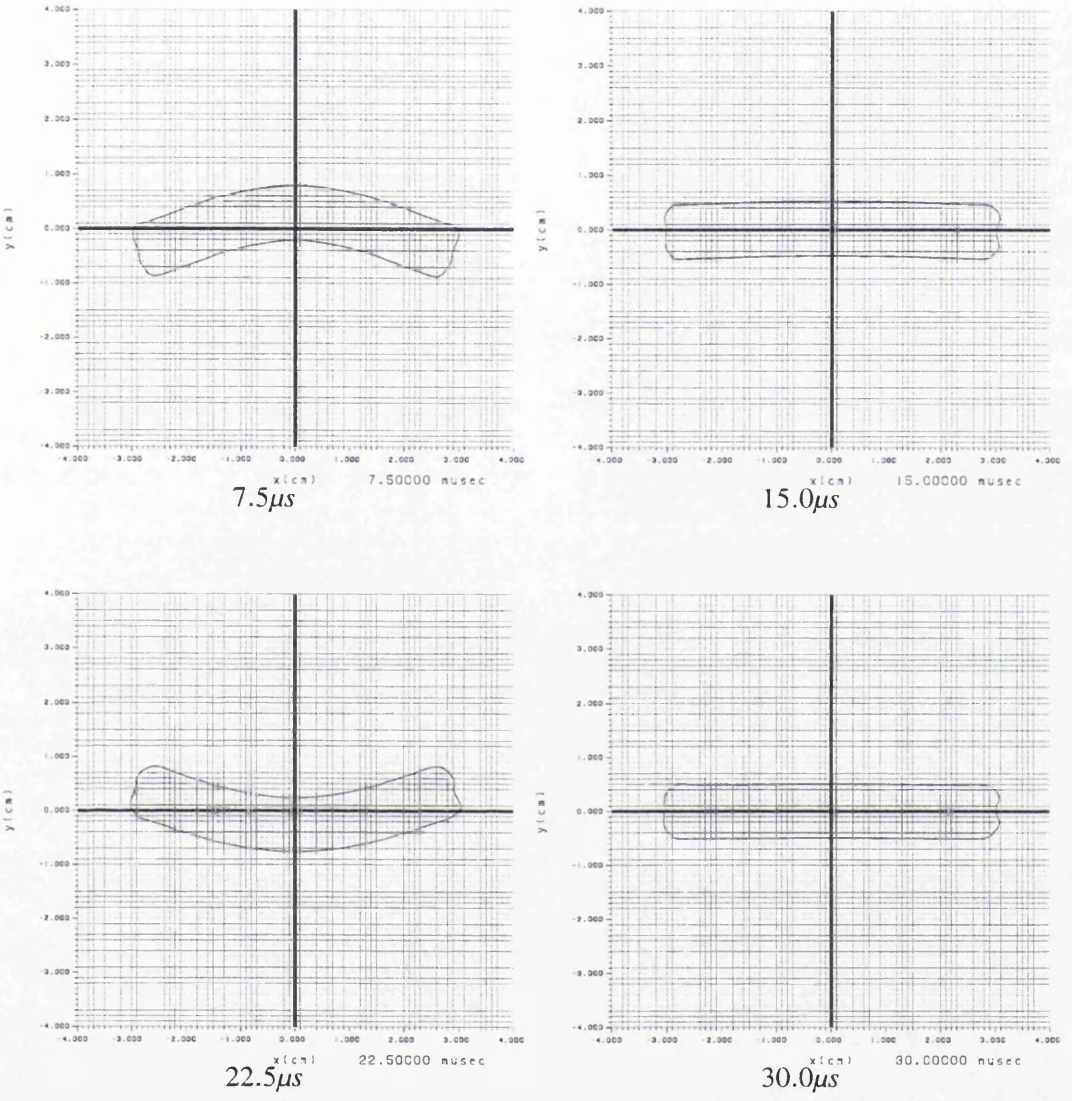


Figure 6.19: Berylium vibrating plate test problem at 7.5, 15.0, 22.5 and 30.0 μs calculated with Eulerian mesh motion.

6.4.5 1D Spherical implosion

Definition

The 1D Spherical implosion problem was fully defined in section 4.2.1. The problem is used to demonstrate that spherical symmetry is still maintained when the problem is calculated using the multi-material ALE package. The spherical Tantalum shell is again initially meshed with 2° angular zoning and 2 radial zones as shown in Fig. 6.20. In this case the void inside the shell is also meshed to allow multi-material ALE to be used to maintain the mesh resolution in the shell as it implodes.

Numerical Results

The initial mesh is given in mesh at $4.0\mu s$ for the ALE calculation is given in Fig. 6.21 and can be compared with mesh obtained when the calculation was performed in pure Lagrangian mode with Automatic Mesh Insertion in Fig. 5.3. The results clearly demonstrate that spherical symmetry is retained for the ALE calculation and show good qualitative agreement in terms of the inner shell radius at $4.0\mu s$.

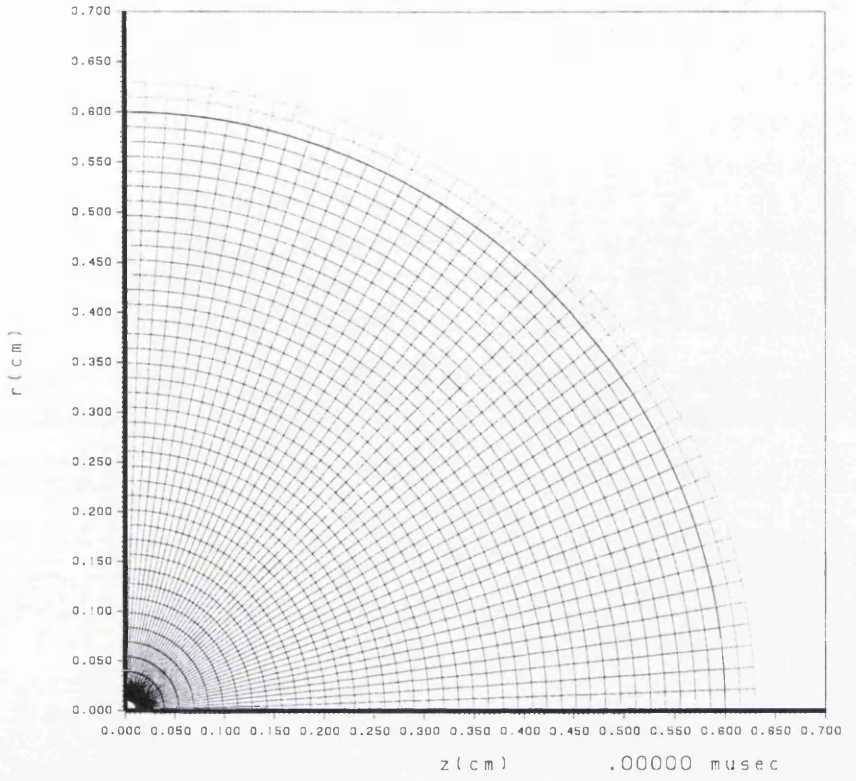


Figure 6.20: Initial Mesh for 1D Spherical implosion problem calculated using ALE with Equipotential mesh motion.

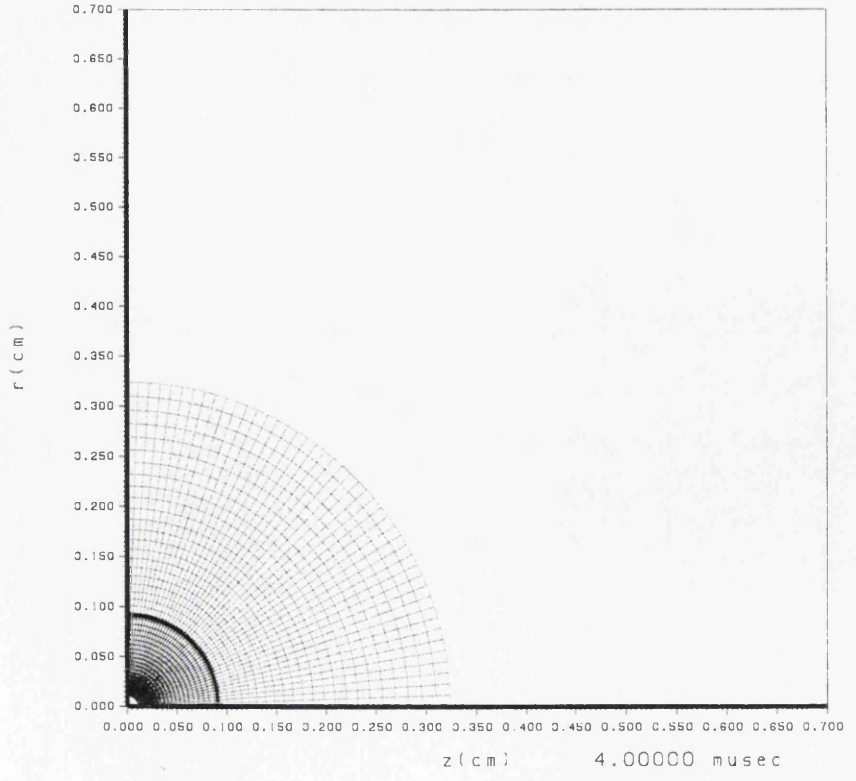


Figure 6.21: Mesh for 1D Spherical implosion problem calculated using ALE with Equipotential mesh motion at 4.0 μs.

6.4.6 2D Implosion of spherical metal shell

Definition

The 2D Spherical implosion problem which was fully defined in section 4.2.2 has also been calculated using the multi-material ALE package. In this case the central void is again meshed to allow Equipotential mesh relaxation to be used to maintain the mesh resolution of the Tantalum shell. This also demonstrates the robustness of the ALE approach in treating the high deformation of the inner boundary of the Tantalum shell.

Numerical Results

The mesh at $5.2\mu s$ from the ALE calculation is given in Fig. 6.23 and can be compared with the pure Lagrangian calculation with Automatic Mesh Insertion given in Fig. 5.7. Qualitative agreement is obtained in terms of both the inner and outer interfaces of the Tantalum shell. The small differences observed are probably mainly due to greater resolution in the AMI calculation.

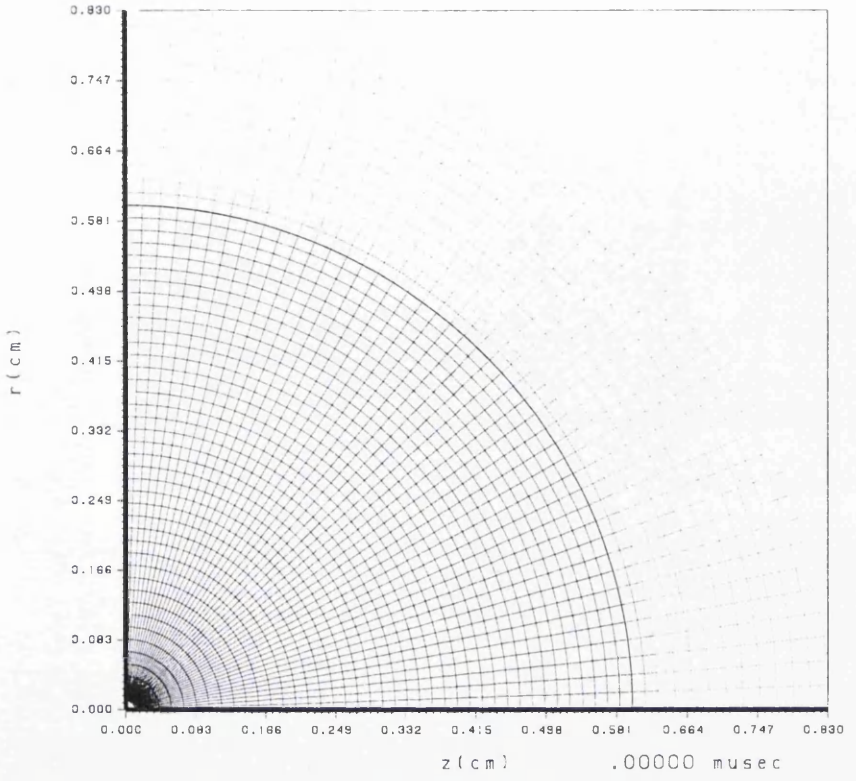


Figure 6.22: Initial mesh for 2D implosion of a spherical metal shell calculated with ALE and Equipotential mesh motion.

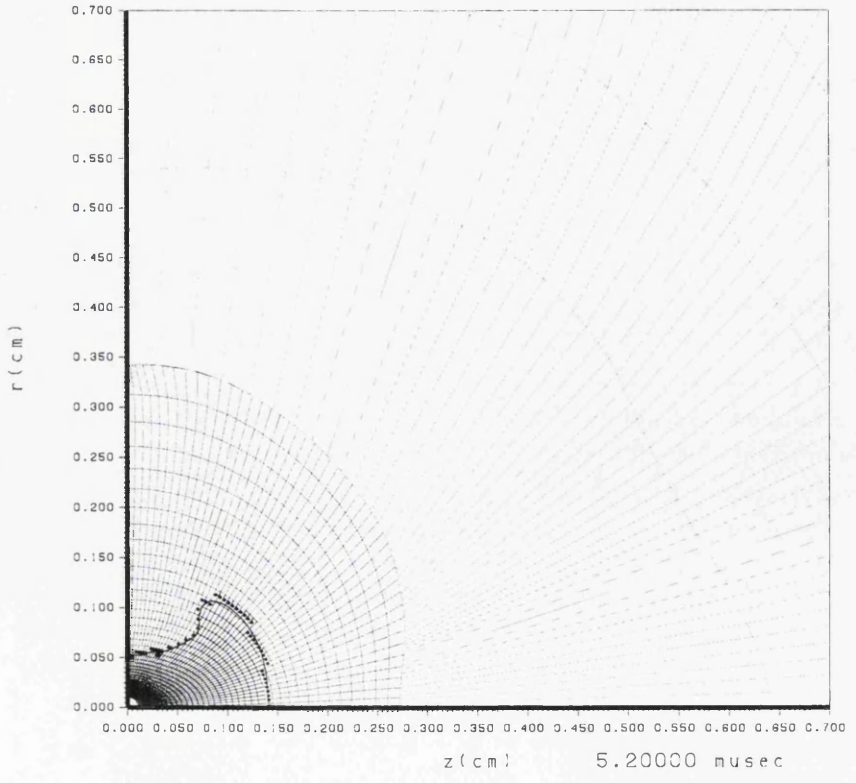


Figure 6.23: Mesh for 2D implosion of a spherical metal shell calculated with ALE and Equipotential mesh motion at 5.2 μs.

Chapter 7

The Lagrangian treatment of multi-material cells

7.1 Introduction

The extension of any staggered grid Lagrangian hydrodynamic scheme to the treatment of multi-material cells requires two fundamental issues to be addressed: how to determine the thermodynamic states of the individual materials within each multi-material cell, and the forces generated at its nodes. These two issues are clearly related, since the forces will depend on the states within each zone. The main difficulty is the lack of information about the velocity distribution within multi-material cells.

A separate set of material properties is normally maintained for all the materials in each multi-material zone, along with the volume fractions which define the fraction of the cell's volume occupied by each of it's materials. A sub-element model is then required to define how the volume fractions and states of the individual materials evolve during the Lagrangian step. Three different types of sub-element model are in routine use in hydrocodes. The simplest of these assumes that the volumetric strain is equal in all the materials within each multi-material cell and so leaves the volume fractions unchanged during the Lagrangian phase. This is the default treatment in CORVUS. The main deficiency of this approach is clearly illustrated by considering a cell containing a mixture of gas and metal. The gas is clearly more compressible than the metal and so should take up most of the volume change, but the simple equal strain model forces the metal to undergo the same volumetric strain as the gas. Although the equal strain treatment is physically unrealistic in practice it does a surprisingly good job.

The other two types of sub-element model either enforce pressure, or pressure and temperature equilibrium, for all the material components within a multi-material cell. The pressure at a material interface should be continuous, however,

as the pressure within a computational zone represents an average pressure integrated over the cell volume, there is no physics requirement for absolute pressure equilibrium within a multi-material cell. In fact, for the entire computational zone to come to pressure equilibrium, a shock wave would have to cross the element many times, while the CFL stability condition restricts a shock wave from crossing any cell in a single time step. However, pressure continuity at material interfaces does suggest that the pressure within a multi-material cell should move towards pressure equilibrium, not diverge away from it. If pressure separation is allowed to continue unabated, as can occur with the constant strain treatment, non-physical thermodynamic states are likely to be produced, which will inevitably lead to code robustness problems. The assumption of thermal equilibrium is more questionable, as it implies the need for infinite thermal conductivity within each multi-material cell.

Once the thermodynamic states for the individual materials have been determined, by one of these sub-element models, then the definition of the nodal forces follows naturally, proceeding exactly as for a single material cell, but using a volume fraction weighted average pressure for the equal strain model, and the equilibrium pressure for the other two sub-element models. Both of these will be consistent and conserve energy, since the work done in the momentum step will be equal to the total change in internal energy for the zone calculated in the corrector internal energy step.

At AWE a further type of sub-element model has been developed and is used in the 2D multi-material Eulerian code PETRA. This method attempts to move towards pressure equilibrium in a controlled manner, which is more representative of the true interface physics, and is hence termed a pressure relaxation scheme. The pressure is however only relaxed at the end of the Lagrangian after the equal volumetric strain method has been applied.

In this chapter a new method which has been developed for CORVUS by the author will be presented. The new scheme attempts to emulate the behaviour of a series of Lagrangian subzones, rather than a single mixed element. A series of test problems have also been developed for benchmarking Lagrangian multi-material cell schemes. The results from these test problems are presented for the new scheme, the equal volumetric strain treatment and the pressure relaxation method discussed above.

7.1.1 PETRA pressure relaxation scheme

The method in PETRA was derived by applying a weak shock wave approximation. In order to explain the method and its derivation only two materials are considered as the procedure is applied to each interface in isolation, with the resulting volume and internal energy exchanges then being combined to obtain the final material states.

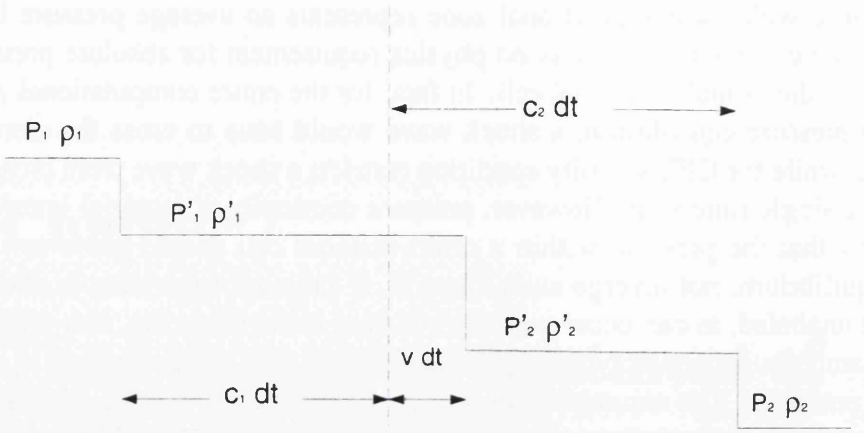


Figure 7.1: Notation for PETRA pressure relaxation scheme.

Consider two materials with pressures and densities of (p_1, ρ_1) and (p_2, ρ_2), where $p_1 > p_2$ and allow a weak shock to be generated. After a time, dt, the situation will be as shown in Fig. 7.1, the disturbances will both have travelled a distance, cdt, where c is the local sound speed.

From the definition of the sound speed it can be written

$$c_1^2 = \frac{p - p_1}{\rho_1' - \rho_1} \tag{7.1}$$

$$c_2^2 = \frac{p - p_2}{\rho_2' - \rho_2} \tag{7.2}$$

where $p = p_1' = p_2'$ is the pressure in the disturbed region. Then by simply applying the conservation of mass we can obtain,

$$\rho_1 c_1 dt = \rho_1'(c_1 + v)dt \tag{7.3}$$

$$\rho_2 c_2 dt = \rho_2'(c_2 - v)dt \tag{7.4}$$

where v is the material velocity in this disturbed region relative to the frame of reference of the cell, which can be eliminated to obtain the pressure p at the material interface,

$$\begin{aligned} p^2(c_1 + c_2) \quad &+ \quad p[c_2(c_1^2\rho_1 - p_1 - p_2) - c_1(c_2^2\rho_2 + p_1 + p_2)] \\ &+ \quad c_1 p_1(c_2^2\rho_2 + p_2) + c_2 p_2(c_1^2\rho_1 + p_1) = 0 \end{aligned} \tag{7.5}$$

The weak shock approximation simplifies this further by assuming that for a small disturbance $v \ll c_1$ and $v \ll c_2$, so $\rho_1' \approx \rho_1$ and $\rho_2' \approx \rho_2$. This allows (7.3) and (7.4) to be rewritten as,

$$\rho_1 v = -c_1(\rho_1' - \rho_1) \tag{7.6}$$

$$\rho_2 v = c_2(\rho_2' - \rho_2) \tag{7.7}$$

ρ_1' and ρ_2' can then be eliminated using (7.1) and (7.2) which then allows the following expressions to be obtained for the interface pressure p and the relative velocity of the interface with respect to the frame of reference of the cell v,

$$p = \frac{p_1\rho_2 c_2 + p_2\rho_1 c_1}{\rho_1 c_1 + \rho_2 c_2} \tag{7.8}$$

$$v = \frac{p_1 - p_2}{\rho_1 c_1 + \rho_2 c_2} \tag{7.9}$$

The volume exchange between the two materials is then defined as,

$$\delta V = v\delta t A \tag{7.10}$$

where δt is the time step and A the area across which the volume is exchanged. This volume exchange is however also limited from changing the volume fraction of either of the materials by more than 25 %, which reflects the approximate nature of the procedure. The complete set of volume exchanges is then used to define a new set of volume fractions and densities for all the materials in the multi-material cell. The product of the interface pressures p and the volume exchange is then used to define the internal energy exchange between the two materials.

7.2 The new Lagrangian multi-material cell algorithm

In view of the deficiencies of the four sub-element models discussed above a new algorithm has been developed. The new scheme is formulated as an integral part of the Lagrangian step, with pressure equilibrium being approached in a controlled manner which attempts to mimic the behaviour of individual Lagrangian subzones. The aim is to improve the accuracy of both the individual thermodynamic states of the material components and the average response of the cells in terms of the forces generated at its nodes.

Only two materials will be considered in the following description. However, the algorithm does naturally extend to m materials in a similar manner to the PETRA pressure relaxation algorithm discussed above, by operating on each pair of

materials associated with each interface in isolation. This enables a volume exchange to be determined between each material pair. The total volume change is then accumulated for each material component. The densities, internal energies and pressures are then updated for each material.

The first problem that had to be solved in developing the new method was how to allow volume fractions to vary during the Lagrangian step and still conserve energy. This was achieved by noting that to conserve energy the total work done on or by a multi-material cell during the momentum step must equal the total internal energy added or removed from this zone during the corrector internal energy update. This is achieved for the equal volumetric strain treatment. The average cell pressure used in the momentum step is the volume fraction weighted average of the individual half step material pressures (7.11). The change in internal energy applied to each individual material component during corrector internal energy update is simply the product of the material components, pressure, volume fraction and the cells volume change (7.12).

$$P_i = \sum_{k=1}^{m} f_{k,i} p_{k,i} \tag{7.11}$$

$$\varepsilon_{k,i}^{n+1} = \varepsilon_{k,i}^{n} + \Delta t \frac{(p_{k,i}^{n+\frac{1}{2}} + q_i^n)}{m_{k,i}} f_{k,i} \nabla \cdot \bar{\mathbf{u}} \tag{7.12}$$

In order to retain this consistency when volume fractions are allowed to vary during the Lagrangian step, a new, more general, definition is required for the average cell pressure used for multi-material cells in the momentum step. This can be achieved by defining the relative compressibility $\beta_{k,i}$ for material k in element i as the ratio of the materials volume change $\Delta v_{k,i}$ to the total cell volume change ΔV_i,

$$\beta_{k,i} = \left| \frac{\Delta v_{k,i}}{\Delta V_i} \right| \tag{7.13}$$

The average cell pressure can then be defined as a weighted average of the individual material pressures using the new compressibility factors $\beta_{k,i}$ as the weights.

$$P_i = \sum_{k=1}^{m} \beta_{k,i} p_{k,i} \tag{7.14}$$

This means that the sum of the products of material component pressures and volume changes will always be equal to the product of the average cell pressure used in the momentum step and the total cell volume change, so the work done on or by the multi-material cell in both the momentum step and corrector internal energy update must again be the same.

The use of this new definition for the average cell pressure of a multi-material cell has a number of consequences. The value of $\beta_{k,i}$ must be determined at the predictor internal energy update, so that it can be consistently used in the momentum step and the corrector internal energy update. In practice, individual material volume changes must also have the same sign, or non-physical average pressures are calculated. However, the volume change of a components can be zero, so significant differences in relative compressibility or expansivity are still allowed.

It remains to determine how to calculate the relative volume changes for the material components. The interface velocity could be calculated by solving the Riemann problem exactly. However, such schemes are expensive especially for general equations of state. An approximate non-iterative Reimann solver such as that due to Dukowicz [90] could be used. However, Duckowicz's method requires a strong shock limit for the maximum density change of each material. This may not always be available, so a simpler method is proposed here which approximates one of the Rankine Hugoniot relations [91],

$$v = \frac{p_1 - p_2}{\rho_2 c_2} \tag{7.15}$$

where the subscript 1 corresponds to the higher pressure material and subscript 2 to the lower pressure component. The interface velocity v relative to the frame of reference of the computational cell is assumed to equal the change in particle velocity, the difference in pressure between the two materials is taken as an approximation to the shock pressure and the sound speed of the lower pressure component as an estimate of the shock speed.

The interface velocities are then converted into volume fraction fluxes. An area weighted form is used to preserve symmetry in axisymmetric geometry. The orientation of the interface is calculated from the multi-dimensional slope. The half side perpendicular length $l_{\frac{1}{2}}$ across the cell is then calculated in the logical mesh direction that lies parallel to the interface. This lengthscale relates to the area over which the volume fraction flux is exchanged, as illustrated in Fig. 7.2. For a time step δt and cell area A then the area weighted volume fraction flux is given by,

$$\delta f_i = \frac{v \delta t L_{\frac{1}{2}}}{A} \tag{7.16}$$

In practice it has been found important to limit these volume fractions changes to not exceed 25 % of there original values. This is justified by the approximate nature of (7.15) and the need to minimise any overshoot past pressure equilibrium.

The preferred volume fraction changes are now defined. However, the sum of the volume changes applied to all material components must not exceed the total cell volume change, and all the material volume changes must be applied in the same direction. In order to ensure the former, the volume fraction changes for all

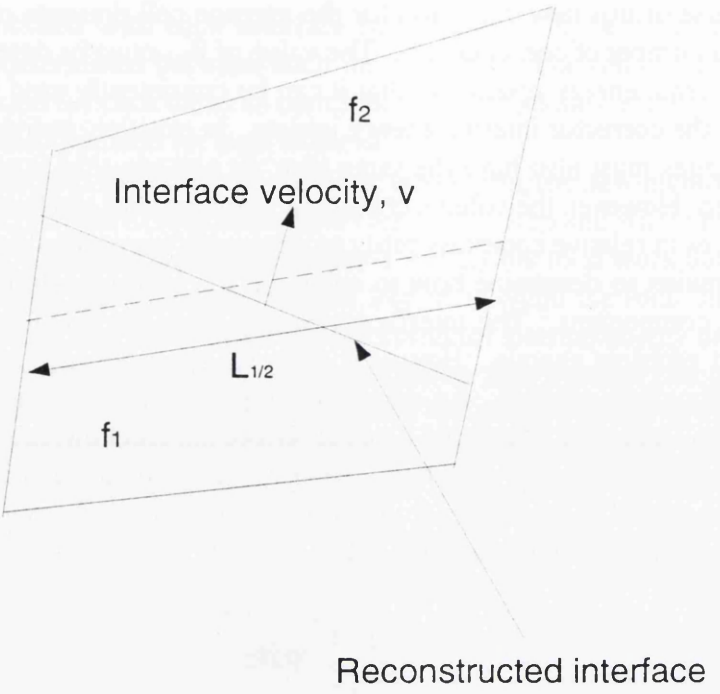


Figure 7.2: Area weighted volume fraction flux.

the materials are converted into volume changes and summed to give a total volume change. If this is larger than the cell volume change, then all these individual volume changes are simply scaled back uniformly to match the total available cell volume change.

The sense or direction of these volume changes then depends on whether the cell is in compression or expansion. If the cell is in compression then the lower pressure component in each pair of materials associated with each interface will be reduced in volume. If the cell is in expansion then the higher pressure component in each pair will be increased in volume. If the total volume change associated with these changes in volume fraction is less than the total volume change for the zone, then new volume fractions are calculated and the remaining volume distributed in proportion to the new volume fractions.

The volume changes for all the materials have now been defined for the predictor internal energy step. The volume fractions and half step densities are then recalculated using these half step volume changes. The half step internal energies are then calculated from,

$$\varepsilon_{k,i}^{n+\frac{1}{2}} = \varepsilon_{k,i}^{n} + \frac{(p_{k,i}^{n} + q_{k,i}^{n})}{m_{k,i}} \Delta v_{k,i}^{n+\frac{1}{2}} \tag{7.17}$$

This is equivalent to the equal volumetric strain update, but now uses the new volume changes and a multi-phase artificial viscosity for each material component.

The multi-phase form is calculated in the same manner as the single phase artificial viscosities but now uses the appropriate density and sound speed for each material component instead of cell average values. A cell average artificial viscosity is however still required for the momentum step. This is defined as a compressibility factor $\beta_{k,i}$ weighted average of the multi-phase artificial viscosities, which ensures the multi-phase artificial viscosities also conserves energy.

In the momentum step, the compressibility factors are calculated and then used to evaluate the average pressures and artificial viscosities for the multi-material cells. The latter are then used to calculate the nodal accelerations exactly, as with the equal volumetric strain method.

The compressibility factors β_k, i are then used to define the fraction of the total cell volume change given to each material during the corrector internal energy update. These volume changes are also used to determine the final volume fractions and densities at the end of the Lagrangian step for each material component. The internal energy for each of the material components at the end of the step is obtained from the volume change, half step pressure and multi-phase artificial viscosity for each material component (7.18).

$$\varepsilon_{k,i}^{n+1} = \varepsilon_{k,i}^{n} + \frac{(p_{k,i}^{n+\frac{1}{2}} + q_{k,i}^{n})}{m_{k,i}} \Delta v_{k,i}^{n+1} \tag{7.18}$$

7.2.1 Compatibility with Material Strength

If the algorithm is to be fully compatible with material strength, a number of issues need to be resolved. The first issue is how to modify the interface velocity definition (7.15) to reflect the stress difference across the material interface. In principle this could be achieved by simply replacing the pressure difference in (7.15) with the difference in the normal stress at the interface. The compressibility factors and changes in volume fraction can then be evaluated as above. The compressibility factors can then be used to redistribute the element's strain rate deviator components amongst the materials. The modified strain rate deviators obtained can then be used to calculate stress deviators, and yield limiting applied. Plastic strain rate deviators can then be calculated and used to calculate the plastic work done on each of the material components. The element average stress deviators required for the momentum step can then be defined as a compressibility weighted average of the stress deviators for the individual material components. The arguments already presented for energy conservation without material strength should then also apply with material strength present.

7.3 Test Problems

In order to assess the potential benefits of the new Lagrangian multi-material cell scheme, a series of interface test problems have been developed. The problems were designed to run to completion with Lagrangian mesh motion, using either a conventional Lagrangian or multi-material cell based interface treatment. Multi-material cells have been introduced into the Lagrangian calculations by merging the pairs of cells on either side of the material interface to form larger multi-material cells. The problems were then calculated with the following Lagrangian multi-material cell schemes, equal volumetric strain, the PETRA pressure relaxation and the new CORVUS scheme.

This approach allowed a direct comparison of the three schemes with a conventional Lagrangian interface treatment on the same computational mesh. The absence of advection in these calculations increases the sensitivity of the solutions to the details of the Lagrangian multi-material cell schemes. Three test problem will be presented here; Sod's shock tube [40], a modified version of Sod's shock tube and a 1D spherical implosion.

7.3.1 Sod's Shock Tube

Shock tube problems are well suited to the testing of Lagrangian multi-material cell schemes, as they allow an arbitrary initial pressure step to be created across the material interface at the start of a problem, by defining appropriate initial conditions for the driver and test gases. They are also simple to interpret and analytical solutions are easily obtained, which allows a detailed comparison to be made of all the thermodynamic state variables for all material components in the multi-material cells with analytical values.

Sod's problem has been calculated on a 1 by 100 cell computational mesh with the two cells either side of the material interface merged to form a single multi-material cell. The solutions obtained at $0.15\mu s$ for the three different Lagrangian multi-material cell schemes considered; equal volumetric strain, PETRA pressure relaxation and the new CORVUS scheme are given in tables 7.2 to 7.4 respectively. The tables include state variables and sound speeds for the individual material components and cell averages. The sound speeds are included for two reasons: high sound speeds are indicative of code robustness problems, and provide a single variable that is sensitive to errors in all the state variables. This makes the sound speed quite a good figure of merit of the quality of the interface scheme. An analytical solution for the jump conditions across the material interface is also given for comparison in table 7.1.

Material no.	Density gcm^{-3}	Internal Energy $Mbcm^3g^{-1}$	Pressure Mb	Sound Speed $cm\mu s^{-1}$
1	0.4263	1.7776	0.30313	0.9977
2	0.2656	2.8535	0.30313	1.2641

Table 7.1: Analytical solution to Sod's shock tube problem.

Material no.	Density gcm^{-3}	Internal Energy $Mbcm^3g^{-1}$	Pressure Mb	Sound Speed $cm\mu s^{-1}$
cell average	0.3675	2.0624	0.30314	1.0747
1	0.6533	2.1089	0.55107	1.0867
2	0.0817	1.6903	0.05521	0.9729

Table 7.2: Solution to Sod's shock tube problem with equal volumetric strain treatment.

Material no.	Density gcm^{-3}	Internal Energy $Mbcm^3g^{-1}$	Pressure Mb	Sound Speed $cm\mu s^{-1}$
cell average	0.3675	2.0624	0.30314	1.0747
1	0.3932	1.9274	0.30314	1.0389
2	0.2411	3.1428	0.30314	1.3266

Table 7.3: Solution to Sod's shock tube problem with the pressure relaxation scheme.

The equal volumetric strain treatment produces by far the worst solution of the three schemes. The average multi-material cell pressure is in reasonable agreement with the analytical solution, but the individual pressures for the two material components are well separated, and both differ significantly from the analytical value. The other state variables for the two materials also differ significantly from the analytical solution by between 19 and 83 % as shown in table 7.5. The test material expands, when it should contract, while the driver material expands, but not as much as is required to match the analytical solution. The sound speeds are also higher for the driver material, and lower for the test material than their corresponding analytical values, which leads to a cell average which is too high.

Material no.	Density gcm^{-3}	Internal Energy $Mbcm^3g^{-1}$	Pressure Mb	Sound Speed $cm\mu s^{-1}$
cell average	0.3947	1.9202	0.30318	1.0369
1	0.4212	1.7994	0.30318	1.0038
2	0.2626	2.8862	0.30318	1.2713

Table 7.4: Solution to Sod's shock tube problem with new CORVUS scheme.

Material	Density	Internal Energy	Pressure	Sound Speed
1	+53%	+19%	+83%	+9%
2	−69%	−41%	−82%	−23%

Table 7.5: Percentage errors for Sod problem calculated with equal volumetric strain treatment.

The pressure relaxation scheme offers a considerable improvement over the equal volumetric strain procedure. The pressures for the two materials are in equilibrium and both within 0.003 % of the analytical solution, and the other state variables for both materials are all within 10 % of their analytical values, as shown in table 7.6. The densities are in this case too low and the internal energies too high for both materials. The sound speeds are, as a result, too high for both materials, which again leads to a cell average sound speed which is too high and comparable in value to that obtained with the equal volumetric strain method.

Material	Density	Internal Energy	Pressure	Sound Speed
1	−8%	+8%	+0.003%	+4%
2	−9%	+10%	+0.003%	−5%

Table 7.6: Percentage errors for Sod problem calculated using the pressure relaxation treatment.

The new Lagrangian multi-material cell scheme performs the best of the three methods considered. It achieves pressure equilibrium matching the analytical pressures to within 0.02 %, and the analytical values for all the other state variables to within 1 % as shown in table 7.7. The new scheme also produces lower sound speeds for both the material components than either of the other two methods considered. This also gives it the lowest cell averaged sound speed of the three methods,

suggesting it should lead to greater robustness and computational efficiency as well as accuracy.

Material	Density	Internal Energy	Pressure	Sound Speed
1	-1%	$+1\%$	$+0.02\%$	$+0.6\%$
2	-1%	$+1\%$	$+0.02\%$	$+0.6\%$

Table 7.7: Percentage errors for Sod problem calculated using new CORVUS scheme.

7.3.2 Sod's Problem with an Artificial Interface

In order to investigate the behaviour of the Lagrangian multi-material cell schemes at an interface between identical materials, Sod's shock tube has also been calculated with an artificial material interface inside the driver section. The driver section in this case is subdivided into two materials, both with identical equations of state. Solutions obtained with the three different Lagrangian multi-material cell schemes were then compared with the solution obtained without the artificial interface.

Density plots for a section of the rarefaction fan centred on the position of the artificial interface are given in Figures 7.4 to 7.6 for the three different Lagrangian multi-material cell schemes. The solution obtained when the artificial interface is not present is given Fig. 7.3 for comparison. The equal volumetric strain and new CORVUS scheme solutions both show a similar perturbations created by the presence of the artificial material interface. A similar perturbation was also obtained by Sim's [92]. In contrast the PETRA pressure relaxation scheme Fig. 7.6 produces a smooth solution comparable to that obtained without the artificial material interface Fig. 7.3.

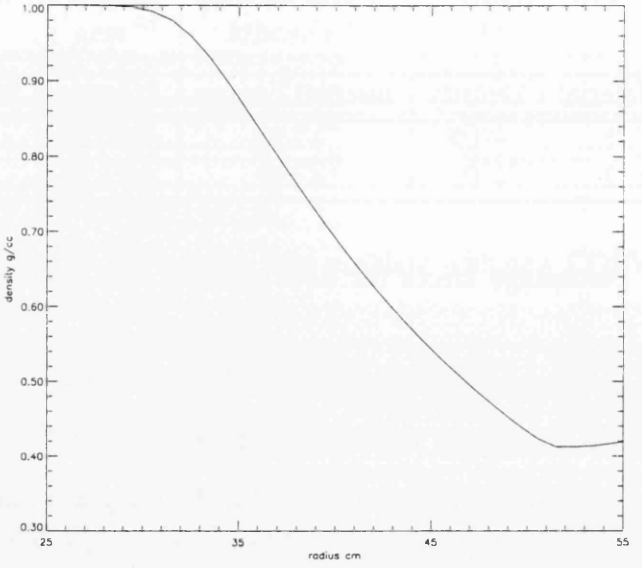


Figure 7.3: Sod's shock tube calculated without the artificial material interface at $0.15\mu s$.

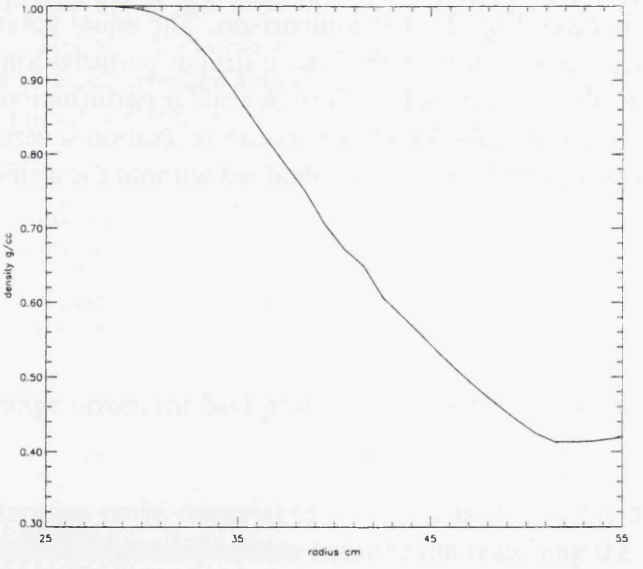


Figure 7.4: Sod's shock tube calculated using equal volumetric strain with an artificial material interface in the driver section at $0.15\mu s$.

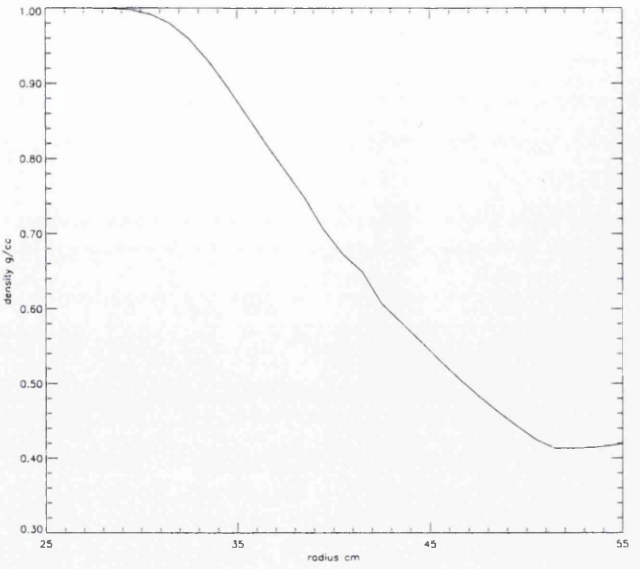


Figure 7.5: Sod's shock tube calculated using new CORVUS Lagrangian multi-material cell scheme with an artificial material interface in the driver section at $0.15\mu s$.

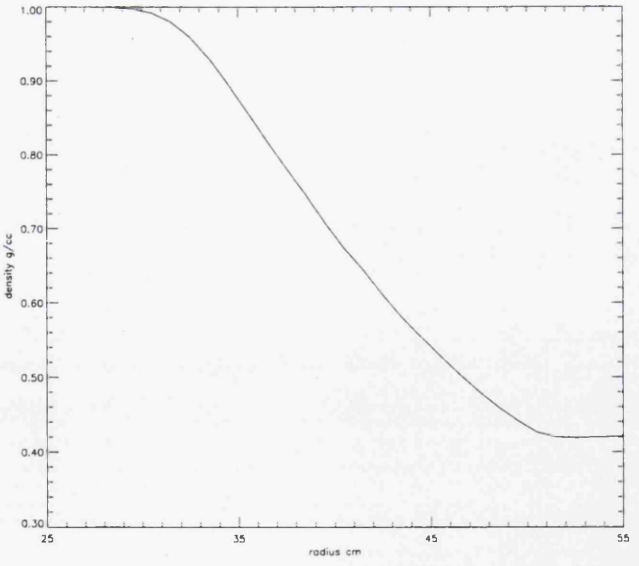


Figure 7.6: Sod's shock tube calculated using the PETRA pressure relaxation scheme with an artificial material interface in the driver section at $0.15\mu s$.

The origin of the perturbations is probably due to the differences in the advection fluxes entering and leaving the multi-material cell that contains the artificial interface. Clearly within the rarefaction fan this will lead to pressure separation between the two material components. As the equal volumetric strain method does not change the volume fractions during the Lagrangian phase, it will not act to ameliorate this problem. The new CORVUS scheme does attempt to modify the volume fractions. However, the sum of the material volume changes is not allowed to exceed the total cell volume change for the cell. The cell volume change will be small inside the rarefaction, and insufficient to allow the new method to reach pressure equilibrium. In contrast, the PETRA pressure relaxation scheme is not limited by the cell's volume change, and so can achieve pressure equilibrium. A multi-material cell, once in pressure equilibrium, will behave essentially like a single material cell, which explains the good solution obtained with the PETRA pressure relaxation scheme for this problem. It also suggests that the new CORVUS scheme may be further enhanced by employing the PETRA pressure relaxation scheme in multi-material cells when the cell volume change over a time step is small.

7.3.3 Modified Sod's Shock Tube

Sod's problem is not particularly challenging, so a modified version of the problem has been defined. The new problem was defined by increasing the ratio of the initial pressures from 10:1 to 25:1, which was achieved by raising the ratio of specific heats γ for the driver gas from 1.4 to 2. This problem also demonstrates the conclusions drawn from Sod's problem are generally valid.

An analytical solution to the modified Sod problem is given in table 7.8 and the results obtained with the three different Lagrangian multi-material cell schemes in tables 7.9 to 7.11. The opportunity was also taken to calculate the problem with the new CORVUS scheme combined with a standard single phase rather than the recommended multi-phase artificial viscosity to demonstrate it benefits. The results obtained are given in table 7.12.

Material no.	Density gcm^{-3}	Internal Energy $Mbcm^3g^{-1}$	Pressure Mb	Sound Speed $cm\mu s^{-1}$
1	0.45	1.13	0.5122	1.503
2	0.36	3.6	0.5122	1.414

Table 7.8: Analytical solution to modified Sod's shock tube problem.

The equal volumetric strain method produces an average cell pressure which is within 1 % of the analytical interface pressure. However, it produces material component pressures that are in error by as much as 90 %. The errors obtained for the

Material no.	Density gcm^{-3}	Internal Energy $Mbcm^3g^{-1}$	Pressure Mb	Sound Speed $cm\mu s^{-1}$
cell average	0.3506	1.5713	0.51203	1.6959
1	0.6234	1.5596	0.97219	1.7661
2	0.0779	1.6647	0.05188	0.9655

Table 7.9: Solution to modified Sod's shock tube problem with equal volumetric strain treatment.

other state variables are given in table 7.13 and vary from 18 to 90 %. The general behaviour is similar to that obtained with Sod's problem. The driver material again fails to expand sufficiently to reach the analytical value, and the test material expands when it should be compressed. The sound speeds are again higher for the driver material, lower for the test material than the analytical values, and the cell average value is too high.

Material no.	Density gcm^{-3}	Internal Energy $Mbcm^3g^{-1}$	Pressure Mb	Sound Speed $cm\mu s^{-1}$
cell average	0.3670	1.6654	0.51209	1.6379
1	0.3745	1.3674	0.51209	1.6537
2	0.3161	4.0497	0.51208	1.5059

Table 7.10: Solution to modified Sod's shock tube problem with pressure relaxation scheme.

The PETRA pressure relaxation scheme improves the solution significantly over equal volumetric strain. The pressures for the two materials are in equilibrium and both within 0.002 % of the analytical solution. However in this case all the state variables are only matched to within 21 % of the analytical values, as shown in table 7.14. The densities are too low and the internal energies too high for both materials. The sound speeds are too high for both materials, and the cell average sound speeds are again no better than the values obtained using the equal volumetric strain method.

The new CORVUS Lagrangian multi-material cell scheme combined with the multi-phase viscosity again clearly performs the best of the three methods considered. It achieves pressure equilibrium, matches the analytical pressures to within 0.002 % and the analytical values for all the other state variables to within 3.5 %, as shown in table 7.15. The new scheme also produces significantly lower sound speeds for both the material components than either of the other two methods and

Material no.	Density gcm^{-3}	Internal Energy $Mbcm^3g^{-1}$	Pressure Mb	Sound Speed $cm\mu s^{-1}$
cell average	0.4278	1.4339	0.51221	1.5165
1	0.4381	1.1692	0.51221	1.5292
2	0.3606	3.5512	0.51221	1.4102

Table 7.11: Solution to modified Sod's shock tube problem with new CORVUS scheme.

Material no.	Density gcm^{-3}	Internal Energy $Mbcm^3g^{-1}$	Pressure Mb	Sound Speed $cm\mu s^{-1}$
cell average	0.4278	1.4536	0.5123	1.5140
1	0.4436	1.1548	0.51224	1.5198
2	0.3332	3.8435	0.51223	1.4671

Table 7.12: Solution to modified Sod's shock tube problem with new CORVUS scheme using single phase artificial viscosity.

has the lowest cell average sound speed of the three methods. This also supports the conclusion that it should provide overall greater robustness, computational efficiency and accuracy than the other two schemes.

The errors for the new CORVUS scheme with the single phase viscosity are also given for comparison in table 7.16. These results are significantly less accurate overall than those with the multi-phase artificial viscosity supporting the need to use the multi-phase form. The errors for the test material increase from less than 1 % with the multi-phase artificial viscosity, to up to 8 % with the single phase form, but with a decrease of about 1 % in the state errors for the driver material.

Material	Density	Internal Energy	Pressure	Sound Speed
1	+39%	+38%	+90%	+18%
2	−78%	−54%	−90%	−32%

Table 7.13: Percentage errors for modified Sod problem calculated with equal volumetric strain treatment.

Material	Density	Internal Energy	Pressure	Sound Speed
1	−17%	+21%	+0.002%	+10%
2	−12%	+12%	+0.002%	−6%

Table 7.14: Percentage errors for modified Sod problem calculated using pressure relaxation treatment.

Material	Density	Internal Energy	Pressure	Sound Speed
1	−2.6%	+3.5%	+0.002%	+1.7%
2	+0.2%	+1%	+0.002%	+0.3%

Table 7.15: Percentage errors for modified Sod problem calculated using new CORVUS scheme with multi-phase artificial viscosity.

The latter may be explained by considering the process by which pressure equilibrium is established at the contact discontinuity with the new Lagrangian multi-material cell scheme, and contrasting this process with the physics that actually occurs. Shock and rarefaction waves should form with the driver material expanding and the test material compressing simultaneously to maintain pressure equilibrium at the interface. However, numerically the new Lagrangian multi-material cell scheme divides this process into separate periods of expansion and compression, the volume changes applied to the materials within each multi-material cell being only allowed to occur in the one direction or sense during each time step. If the cell is in expansion, the volume change will be given away preferentially to the higher pressure component. If the zone is in compression then the volume change will preferentially go to the low pressure component. Pressure equilibrium is then established numerically by initially expanding one material component for a number of steps, then compressing the other component, and so on, until pressure equilibrium is established over a number of time steps.

This approach allows for variable compressibility of the different materials and also attempts to apply the right entropy increase to the shocked material, without changing the entropy of the unshocked material, with the test material component undergoing shock heating while the driver material is cooled adiabatically. However, the approximate nature of the procedure for calculating the interface velocities inevitably leads to overshoots, which in turn lead to the need to recompress the driver material to achieve pressure equilibrium. The use of a multi-phase artificial viscosity produces more realistic shock heating in the test material, but also leads

Material	Density	Internal Energy	Pressure	Sound Speed
1	−1.4%	+2.2%	+0.008%	+1.1%
2	+7.4%	+6.8%	+0.006%	+3.8%

Table 7.16: Percentage errors for modified Sod problem calculated using new CORVUS scheme with single-phase artificial viscosity.

to too much shock heating in the driver material if it expands past pressure equilibrium and must be recompressed. Hence, while the states obtained for the test material are significantly more accurate, the errors introduced in the driver material states increase.

7.3.4 1D Spherical Implosion

The third test problem consists of a spherical axisymmetric metal shell containing a high pressure gas. An initial uniform radial velocity of 1.0 mm μs^{-1} is applied to the metal shell. The metal shell then implodes inwards compressing the gas until the gas pressure is high enough to turn the shell around. The problem tests the symmetry and the accuracy of a Lagrangian multi-material cell scheme for a range of interface conditions.

A series of calculations were performed, restarting at 3.5 μs and then introducing multi-material cells, by merging the line of zones either side of the interface. These calculations were then continued to 6.0 μs using each of the three Lagrangian multi-material cell schemes. The initial mesh at 0.0 μs is given in Fig. 7.7 and at 3.5 μs in Fig. 7.8. Typical mesh and interface plots are given at 4.2 μs for equal volumetric strain in Fig. 7.9 and for the new CORVUS Lagrangian multi-material cell scheme in Fig. 7.10. The volume fractions are of course unchanged for the equal strain treatment, while a similar compression is observed for the gas components of the multi-material cells to that in the single material gas zones with the new CORVUS scheme.

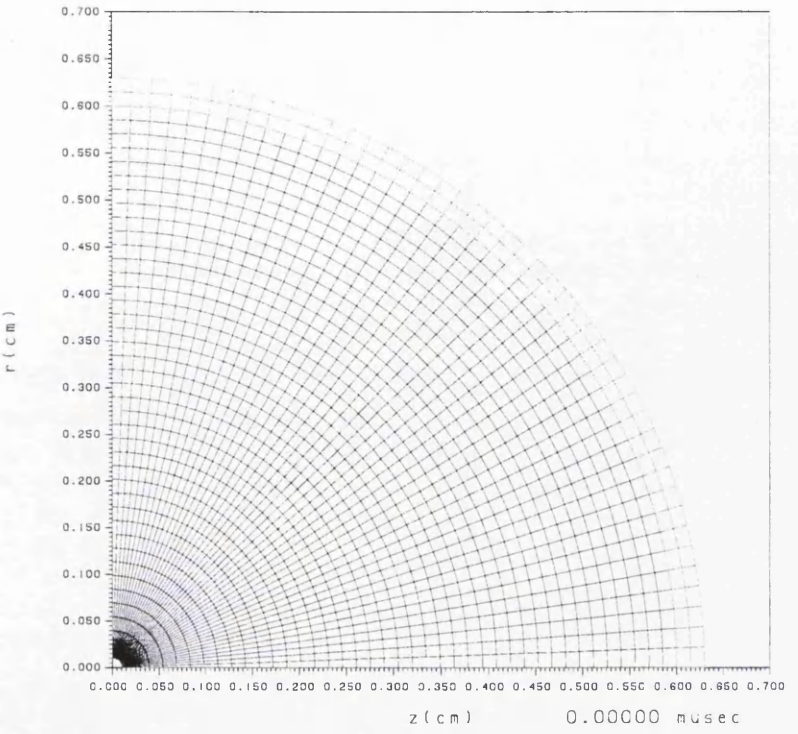


Figure 7.7: Initial mesh for spherical implosion problem.

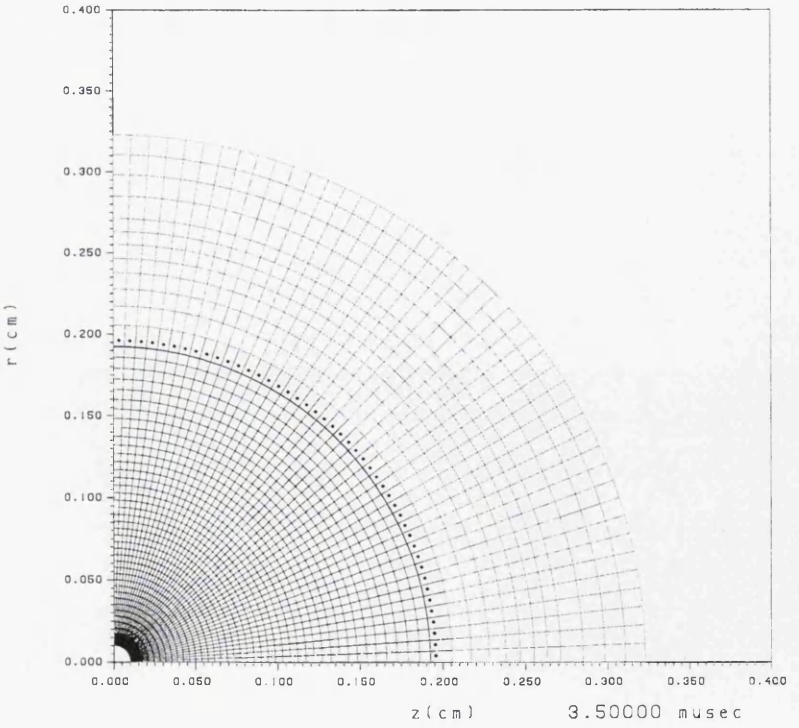


Figure 7.8: Lagrangian mesh at 3.5 μs for spherical implosion problem.

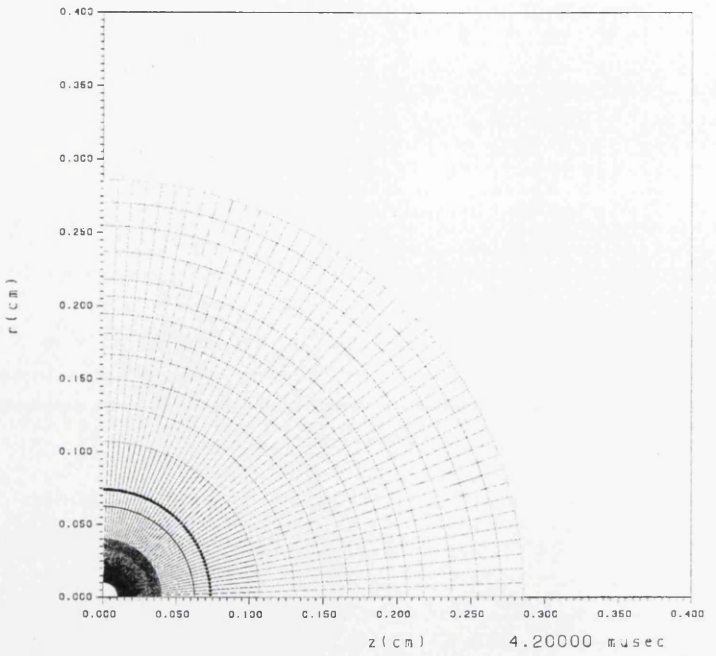


Figure 7.9: Spherical implosion at 4.2 μs calculated with equal volumetric strain.

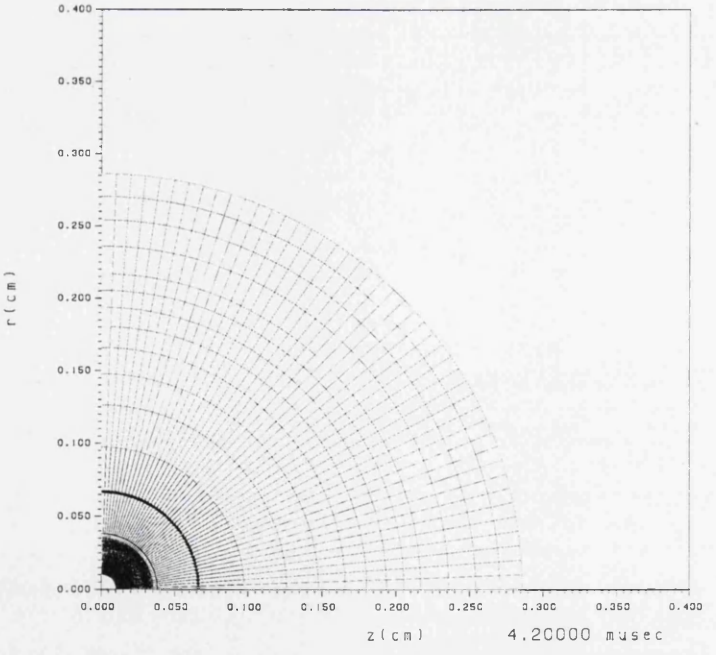


Figure 7.10: Spherical implosion at 4.2 μs calculated with new Lagrangian multi-material cell scheme.

A more quantitative comparison is given in Fig. 7.11 where the gas cavity volume is plotted against time for all three of the Lagrangian multi-material cell schemes, and for a conventional Lagrangian interface treatment. These results clearly show that the equal volumetric strain method provides by far the worst match to the conventional Lagrangian interface treatment. It also fails to match the maximum compression, turns the implosion around too early and expands too rapidly compared to the conventional Lagrangian interface calculation. In contrast the PETRA pressure relaxation scheme and the new CORVUS scheme both provide a good match to the Lagrangian treatment, matching the time and value of the minimum volume for the gas cavity. However, the new CORVUS scheme then follows the Lagrangian gas cavity volume more closely on expansion than either of the other multi-material cell schemes.

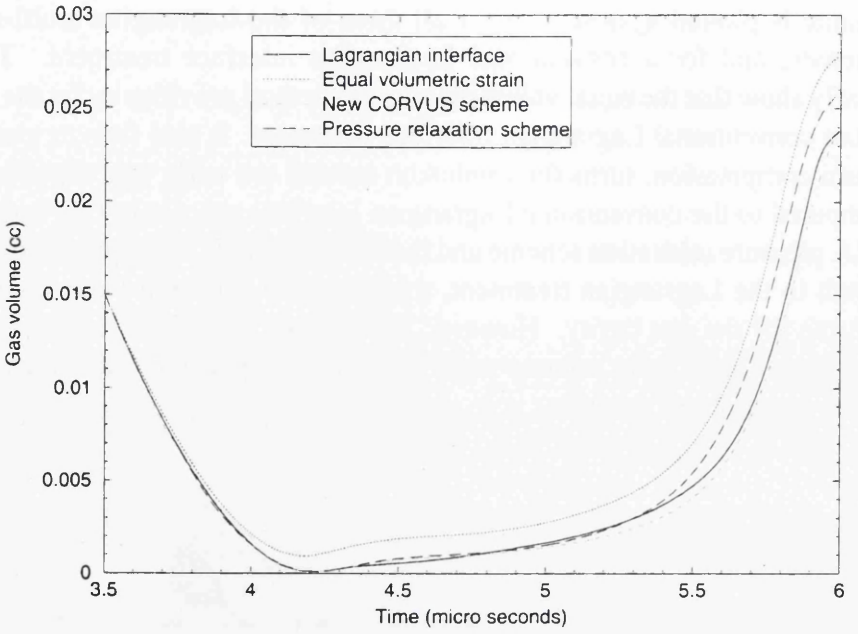


Figure 7.11: Gas cavity volume v's time for spherical implosion calculated with Lagrangian mesh motion using three different Lagrangian multi-material cell schemes.

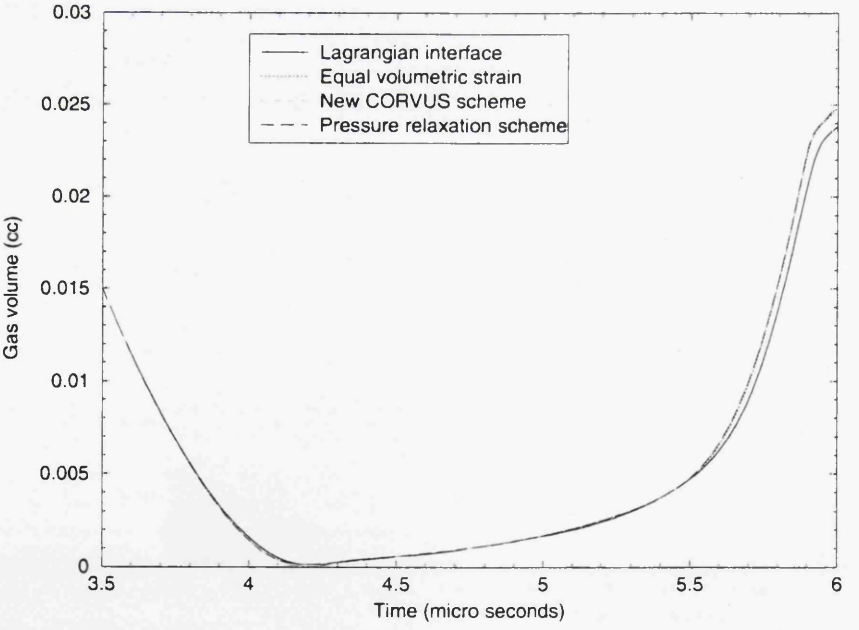


Figure 7.12: Gas cavity volume v's time for spherical implosion calculated with equipotential mesh relaxation using three different Lagrangian multi-material cell schemes.

Fig. 7.12 shows the results obtained when the three calculations were repeated with advection present, and the mesh moved using Winslow's equipotential mesh movement algorithm. In this case all three interface schemes produce solutions in good agreement with conventional Lagrangian interface treatment. These results demonstrate how the presence of significant advection acts to reduce the errors that can be introduced by Lagrangian multi-material cell schemes. It also demonstrates that accurate solutions are obtained with the new CORVUS scheme, whatever mesh motion is used.

7.4 Multi-material Void Closure

The background gas in most impact problems has little influence over the physics of the problem, but can introduce numerical difficulties if it is modelled explicitly. In Lagrangian simulations the time step will collapse in the gas as impact is approached, while multi-material ALE and Eulerian simulations tend to trap some gas between the impactors, leaving it in a non-physical (P,V) state. However, these difficulties can be overcome by replacing the background gas with an idealised void and using a dedicated void closure scheme. In Lagrangian codes this is usually a modified slide line algorithm like the one described in chapter 4. However, although these schemes are accurate, they are not robust for all applications, so a multi-material void closure scheme has also been developed for CORVUS.

7.4.1 Void Closure Algorithm

The only part of the CORVUS Lagrangian multi-material cell scheme which is invalid when voids are present, is the interface velocity definition (7.15), which assumes a shock will form in the lower pressure material, which clearly cannot occur in a void. The scheme also introduces a limit on the changes in volume fractions that are allowed, which will stop the void closing completely. A new constraint is also required to ensure that when impact or void closure does occur it closes the void exactly.

These issues could be overcome by always giving away the cell volume change preferentially to void materials. However, when a shock wave arrives at a free surface the non-void material should be compressed and then expand as the rarefaction wave forms, leaving the material with more entropy than it had before the shocks arrival. If this simple void closure extension is used, the non-void material will not go through this cycle, and its entropy will remain unchanged.

Given this concern, a more conservative approach has been taken in modifying the new CORVUS Lagrangian multi-material cell scheme to include void closure. If a multi-material zone contains a void material and has adjacent single material void cells or is expanding, then the equal volumetric strain procedure is applied to all

the void material interfaces in that cell. All other non-void interfaces in the cell are calculated using the algorithm described above. If no adjacent single material void zones are detected and the cell is under compression, then the voids are considered to be closing and all the cell's volume change is given to the void materials in the cell, in preference to all other materials. This procedure is then followed until the voids are all closed exactly, and then the remaining cell volume change is distributed following the original Lagrangian multi-material cell algorithm, as described above. The algorithm then proceeds as before.

The definitions used for the multi-phase viscosity and the compressibility weighted averages for the average cell pressure and artificial viscosity also ensure that energy conservation is maintained with voids present. It is also encouraging to note that these definitions also ensure that there is no cell pressure in a multi-material cell containing two materials if they are separated by a void until the void has closed.

7.5 1D Plate Impact Problem

The problem consists of two identical 8 mm thick metal plates, each travelling with a velocity of $0.5mm\mu s^{-1}$ towards each other. Symmetry allows the problem to be reduced calculationally to one plate colliding with a reflecting boundary condition. The pure Lagrangian interface calculations were performed by starting with the plate in contact with the boundary and applying a timing offset, while the multi-material cell calculations required the void to be explicitly meshed.

A range of mesh resolutions were used in both cases from 4 mm down to $\frac{1}{16}mm$ to assess the rate of mesh convergence. In the Lagrangian multi-material cell calculations the void stand off was scaled so that there was always only one cell across the void, and a timing offset applied. The void cell and first zone in the metal plate were then merged, as in previous test problems, to create a large multi-material cell. All the calculations were run to $30.0\mu s$. In order to compare the solution obtained using the new Lagrangian multi-material void closure scheme against the Lagrangian interface treatment, the maximum density obtained in the metal plate was recorded at each time step for all of the calculations, and compared against the analytical solution for the density post impact of 35.0 gcm^{-3}.

7.5.1 Numerical Results

The maximum density plots with Lagrangian mesh motion for a conventional Lagrangian interface and the new multi-material cell based void closure treatment are compared in Fig. 7.13. At each mesh resolution the density plots overlay exactly for the conventional Lagrangian interface and the new multi-material void closure scheme. This suggests the void closure scheme is doing as good a job as it can do

and any discrepancies from the analytical solution are due to lack of mesh resolution, and the limitations of the underlying Lagrangian hydro scheme, not the void closure treatment. The plot also demonstrates that for an impactor of this size, a minimum mesh resolution of 2 mm is required to reach the analytical peak density before release occurs, and that the solution is only just mesh converged at $\frac{1}{16}mm$.

Density plots for Eulerian calculations are also given in Fig. 7.14. These results are comparable to those with Lagrangian grid motion except for a few subtle differences in the rise of the profiles, and a similar rate of mesh convergence is obtained.

In order to assess the potential benefits of the new scheme when combined with more optimum ALE grid motion, a further CORVUS calculation was performed using Winslow's Equipotential mesh movement algorithm. The results obtained are given in Fig. 7.15. This strategy results in finer mesh resolution around the impact site than the Eulerian calculations, and significantly accelerates the rate of mesh convergence. Even the 4 mm calculation almost reaches the analytical density, and the 1 mm initial mesh is effectively mesh converged.

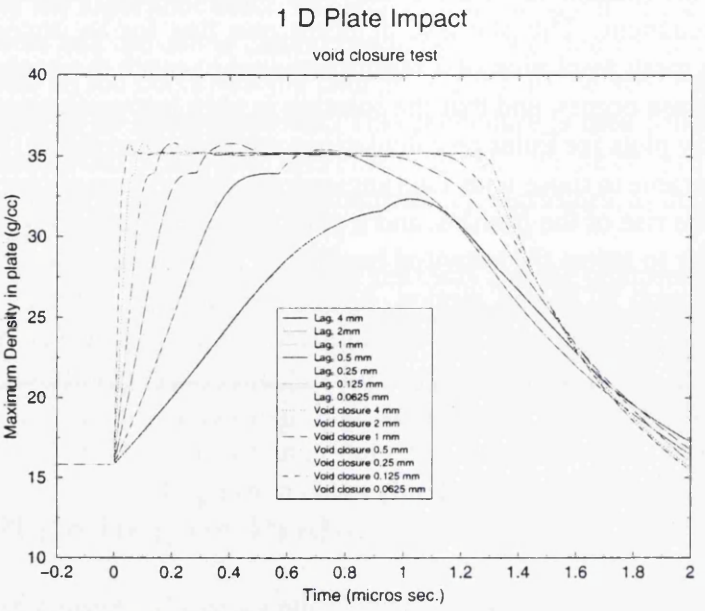


Figure 7.13: Maximum Density for CORVUS Lagrangian 1D Plate impact calculations.

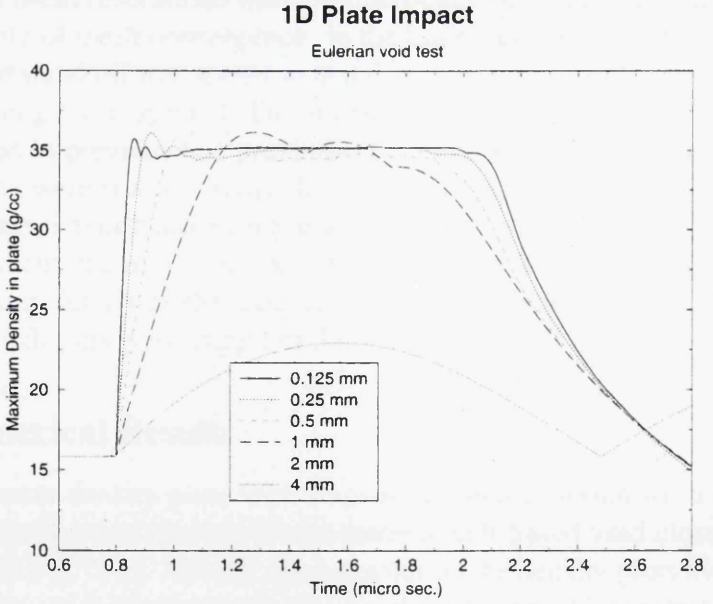


Figure 7.14: Maximum Density for CORVUS Eulerian 1D Plate impact calculations

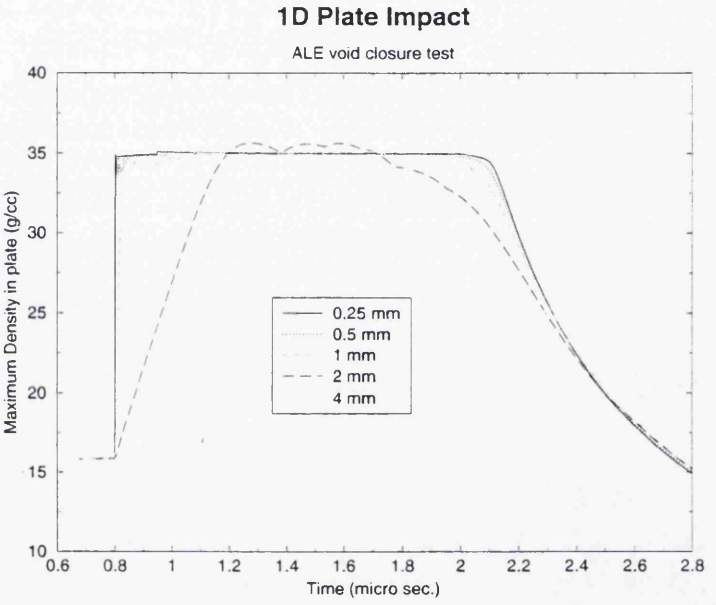


Figure 7.15: Maximum Density for CORVUS ALE 1D Plate impact calculations.

Chapter 8

Validation and Application

The aim of this chapter is to increase the reader's confidence in the adaptive multi-material ALE algorithm and demonstrate how the method can be applied to real physics and engineering problems. Three problems are presented to achieve this. The first is an example of the validation of CORVUS against experiment and another code, while the others are modelling problems where CORVUS has been used. The calculations presented were all performed by the author using CORVUS either at AWE Aldermaston or at the Los Alamos National Laboratory (LANL).

8.1 Shock bubble interaction

In the previous chapters of this thesis a series of test problems have been presented to demonstrate and verify the adaptive multi-material algorithm as implemented in CORVUS. These problems were simple in nature, allowing direct comparison to be made against analytic solutions and desirable properties such as symmetry. They also allowed the performance of individual parts of the algorithm to be assessed in relative isolation. However, it is also important to validate codes. This tests the codes fitness for purpose or its ability to model its target applications. This can usually only be achieved through code comparison against relevant experiments.

One experiment which is well suited to hydrocode validation was performed by Haas and Sturtevant [93]. The experiment consisted of a cylindrical bubble of R22 refrigerant gas ($CHClF_2$, $\gamma = 1.249$, $R = 91$ J/kg K, $C_v = 365$ J/kg K) surrounded by air in a shock tube. A planar shock wave ($M_s = 1.22$) was then generated which travelled down the tube and collided with the bubble, the shock fronts and the bubble interface locations being recorded as a series of Schlieren images as the problem evolved.

In this section, calculations will be presented of the Haas and Sturvevant experiment performed on CORVUS by the author. These results are then compared both

against the available experimental data and another numerical solution obtained using a Free Lagrange (FL) code, VUCALM [94]. The VUCALM simulations were performed using comparable mesh resolution by the developer of VUCALM Graham Ball from Southampton University while on Sabatical to AWE [95]. VUCALM is an interesting code to compare against since it is based on a radically different numerical scheme, but has been developed with similar target applications in mind. It is a non-staggered finite volume hydrocode which employs a high resolution Godunov-type shock capturing method. Comparison against VUCALM is also interesting since it is a pure Lagrangian technique which is amenable to high deformation problems.

8.1.1 CORVUS Calculational Methodology

A single rectangular mesh of 0.5 mm square zones was defined for the entire domain. Air was then defined as the background material and the R22 bubble was painted on top. This introduces multi-material cells at the start of the problem as shown in Fig. 8.1. The strength of this approach is that it separates the geometry definition from the mesh generation, so the mesh quality is not compromised by the need to conform to the problem geometry. A constant velocity boundary condition of $0.011358 cm\mu s^{-1}$ was then applied to the right hand boundary in order to generate an equivalent shock to that found in the experiment. Normal reflecting boundary conditions were applied to the remaining edges of the domain, to represent the rigid walls of the shock tube and the symmetry of the problem.

The first attempt to calculate the experiment then applied Winslow's equipotential mesh movement algorithm at every time step, but with each node constrained to only be relaxed once it had recorded a non-zero velocity. The mesh relaxation as required by the Winslow's algorithm was also reduced by a factor of four to retain some Lagrangian character to the mesh. However, these initial results showed the calculated bubble to be slow relative to the experiment. This loss in kinetic energy was attributed to result from explicitly advecting momentum but not kinetic energy. The error is probably more significant in this problem because of the long time scales involved $\approx 1000\mu s$.

In order to improve the match to the experiment a new mesh movement strategy was devised, where the mesh motion constraints were turned off and the advection step was only called if the new time step was lower than that of the previous step. This significantly reduced the loss in kinetic energy of the bubble bringing its position into very good agreement with experiment throughout the problem. The meshes generated by this approach also retained more Lagrangian character than the constrained Winslow method, and the computational cost of the calculation was also significantly reduced.

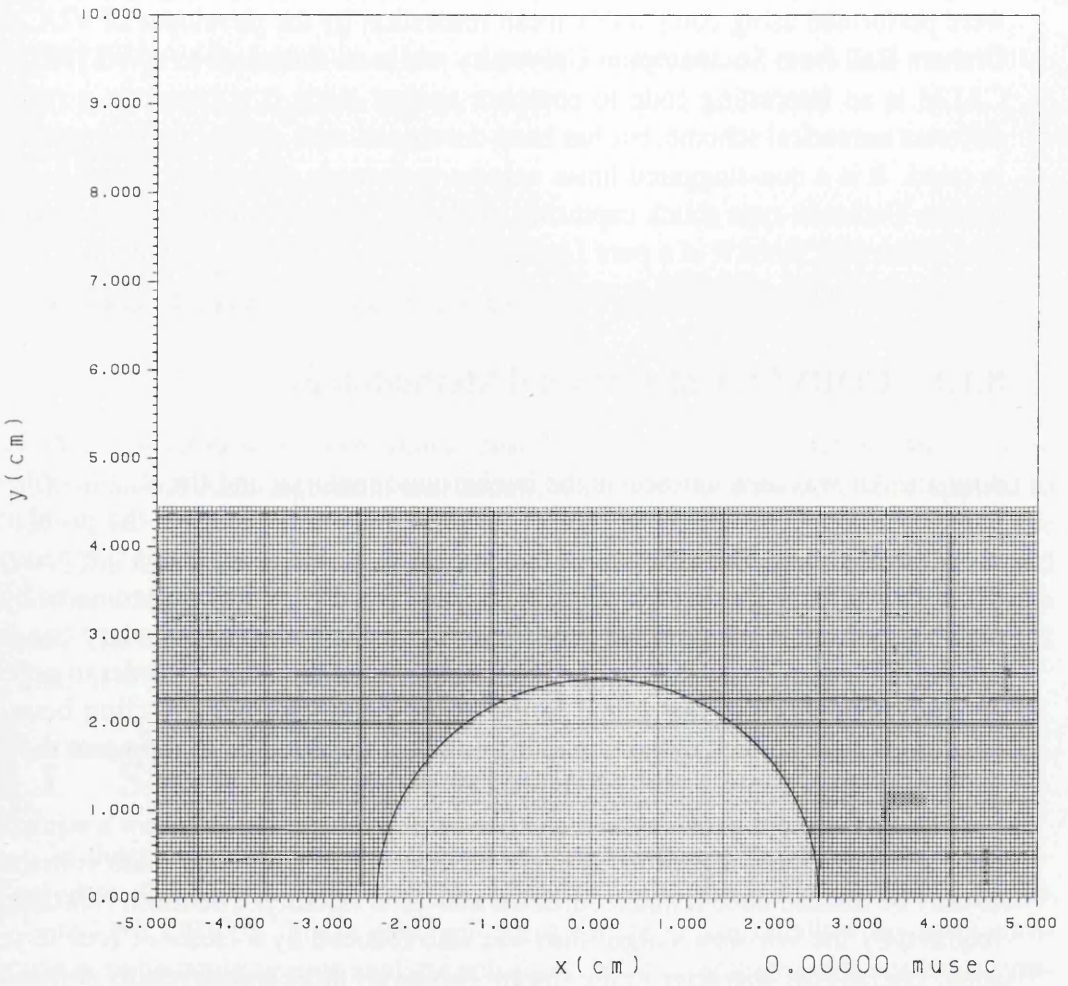


Figure 8.1: Section of initial mesh for CORVUS shock/bubble interaction calculation showing how the bubble was painted on top of the background mesh.

8.1.2 Numerical Results

An early stage in the development of the R22 shock/bubble interaction, at $t = 135\mu$s from first shock impact on the bubble, is shown in Fig. 8.2. The transmitted shock inside the bubble lags behind the air shock, due to the lower sound speed in R22, while significant displacement of the right bubble boundary has occurred. The details of the pressure field in the ALE and FL solutions, including the shock thickness, are almost identical.

Figure 8.3 shows an intermediate stage of the interaction, at $t = 417\mu$s. The dominant flow feature is the roll-up of the top of the bubble, which is associated with a strong counter-clockwise vortex. Here some differences are apparent – the ALE solution shows a lower pressure at the vortex core, and a more extensive roll-up. The reason for this difference is obscure. One other difference of note is the R22 spike to the left of the bubble, on the symmetry axis, which is narrower and longer in the ALE solution. In the experimental image the spike is present, and is intermediate in shape between the FL and ALE solutions, while the amplitude of the roll-up is probably less than seen in the FL case, but the experiment is subject to mixing and viscous effects which are absent from the simulations. In all other respects the solutions are again very similar, and in good agreement with experiment.

Figure 8.4 corresponds to the final experimental image, at $t = 1020\mu$s. The macroscopic features of the three solutions continue to agree acceptably with each other, and with the experimental data in shape and amplitude. The variation in the horizontal position of the right bubble boundary, on axis, between the two simulations is only 1.4 mm. The movement of this boundary is approximately 5% more than in the experiment, as estimated from the published image. The scatter in the left boundary position, on axis and ignoring the spike, is about 1.1mm, and both results agree with the experiment to better than the measurement uncertainty. Finally, scatter in the maximum leftward extent of the bubble tip is within 2 mm. The experimental image is too indistinct in this region for comparison.

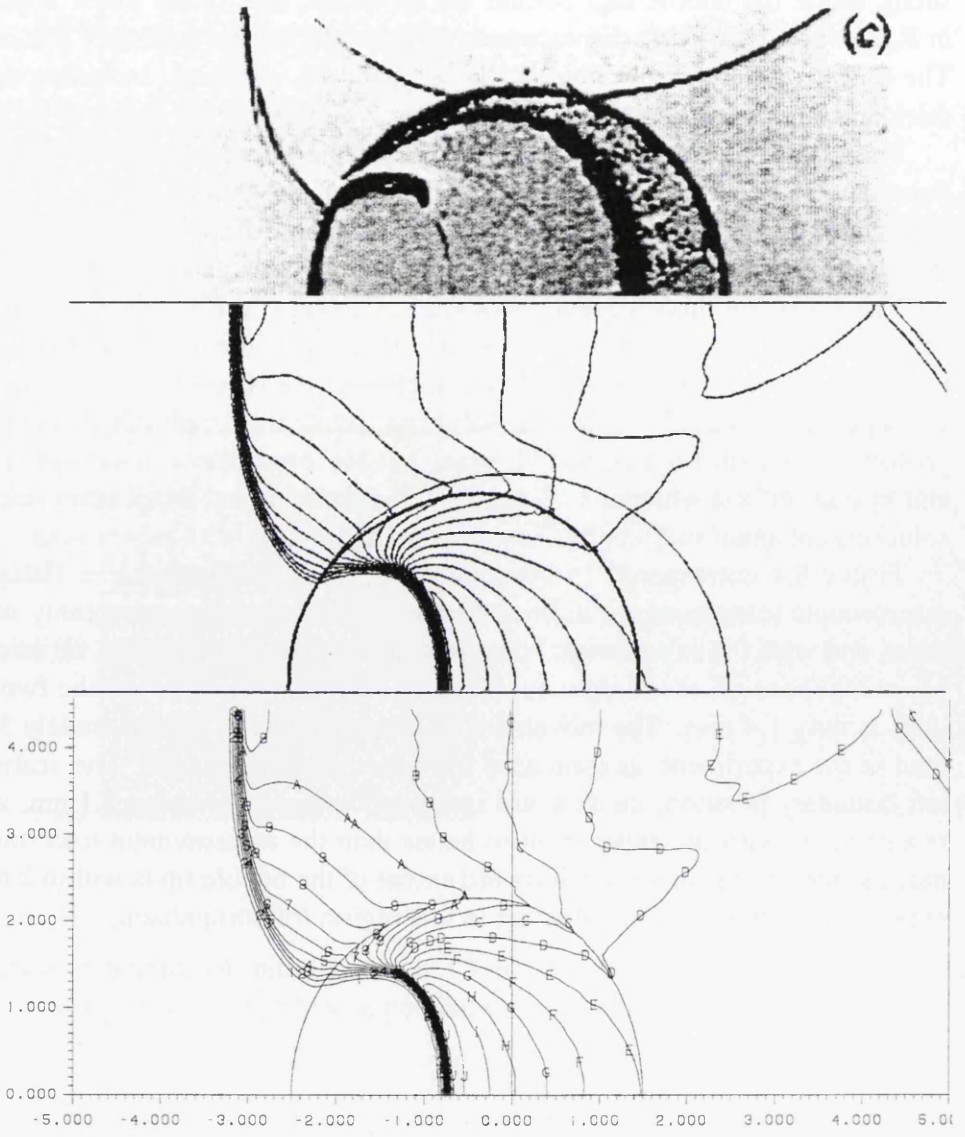


Figure 8.2: Shock/bubble interaction at $t = 135\mu$s, Schlieren image from experiment (top), VUCALM solution (middle) and CORVUS solution (bottom). Contours are pressure, Δp=0.05bar. The scale is in centimetres and the origin is at the initial centre of the bubble.

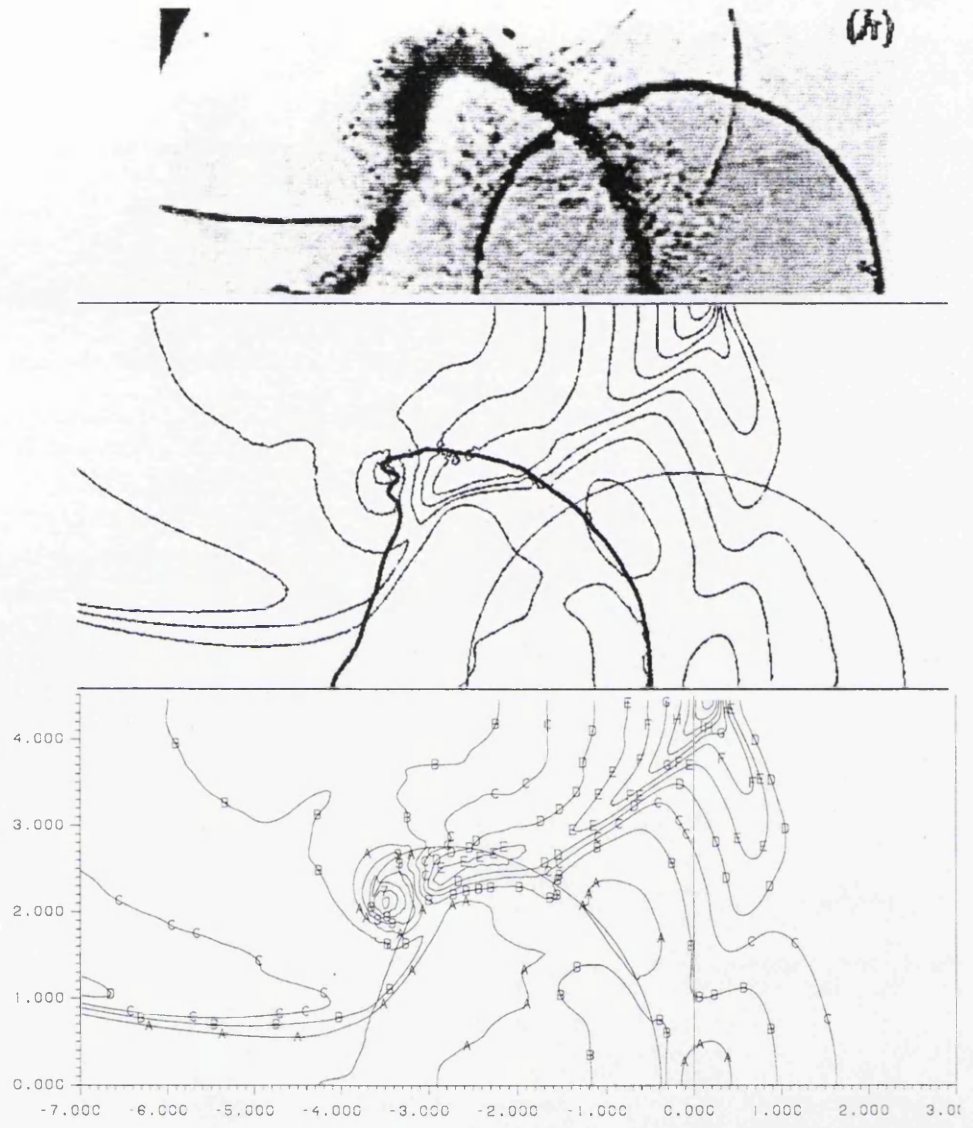


Figure 8.3: Shock/bubble interaction at $t = 417\mu$s, Schlieren image from experiment (top), VUCALM solution (middle) and CORVUS solution (bottom). Contours are pressure, Δp=0.05bar. The scale is in centimetres and the origin is at the initial centre of the bubble.

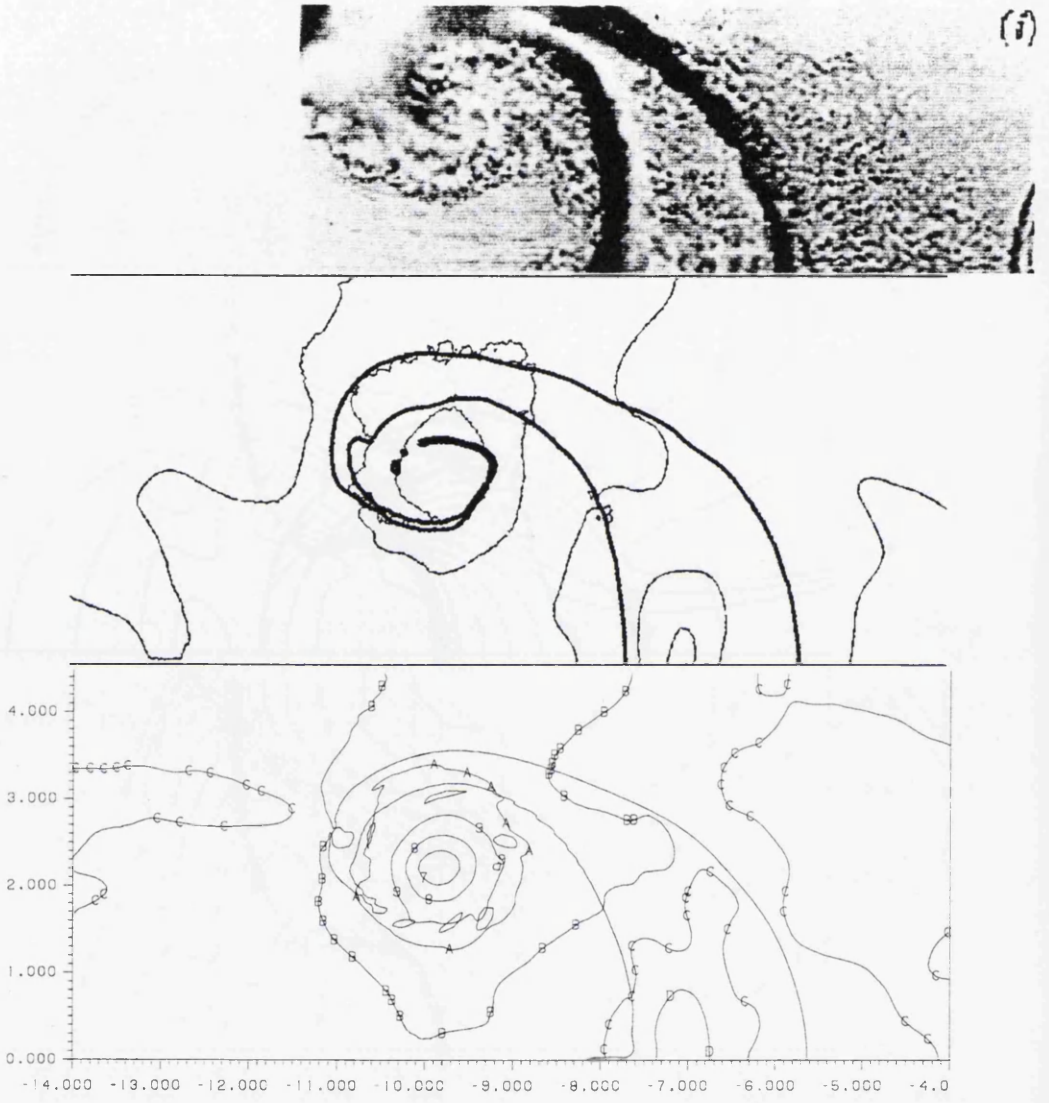


Figure 8.4: Shock/bubble interaction at $t = 1020\mu$s, Schlieren image from experiment (top), VUCALM solution (middle) and CORVUS solution (bottom). Contours are pressure, Δp=0.05bar. The scale is in centimetres and the origin is at the initial centre of the bubble.

The detailed structure of the roll-up does however show significant differences. Figure 8.5 shows details of the local mesh structure in the roll-up. The FL mesh is approximately uniform in resolution, with typical cell aspect ratios near unity, whereas the ALE mesh exhibits significant spatial variations in resolution, with cell aspect ratios of order ten in some regions. Consequently, the ALE solution yields a very smooth R22/air interface on the upper bubble surface, where the local mesh pitch is very fine in the normal direction and the mesh is aligned closely to the interface. On the underside of the roll-up, where the ALE mesh is less well aligned, steps occur on the interface. Towards the vortex core, the FL calculation shows an extended, and almost continuous, filament of R22, while in the ALE solution this feature fragments into isolated islands. The process can be seen at an early stage at the tip of the roll-up where the ALE solution shows two consecutive "necks" in the R22. This is thought to be an artefact of the multi-material advection algorithm, aggravated by the relatively coarse mesh resolution in this region.

Both ALE and FL simulations are in good agreement with the experimental data. For the early stages of the shock/bubble interaction, when shock propagation is dominant, the ALE and FL results are nearly identical, demonstrating the monotonic artificial viscosity approach used in CORVUS provides comparable resolution for shock fronts to the Reimann based approach used in VUCALM. Later, when interface deformation is the dominant process, differences occur in details, primarily due to spatial variation in the quality of the ALE mesh. The FL scheme appears to be better able to capture thin material Filaments. While the ALE simulation is resolving a far steeper vortex in this area which may also have an influence on the filament break up. Turbulent mixing is also clearly present at the interfaces in the experiment and not included in either simulation, which makes it difficult to say definitively which code is providing the best overall simulation. However, accepting the omission of turbulence, the level of agreement between both codes and the experiment is considered very encouraging.

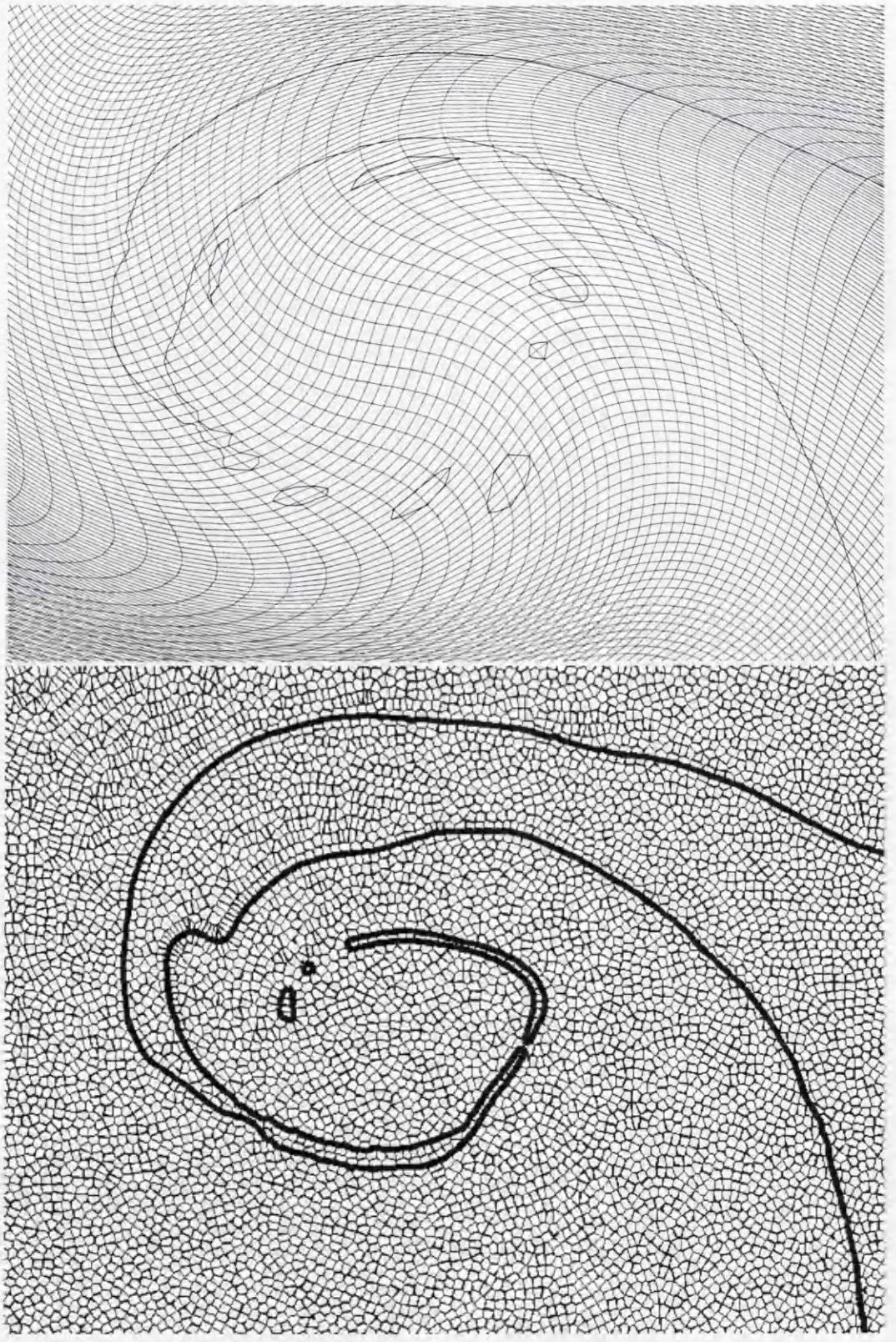


Figure 8.5: Mesh details in roll-up region at $t = 1020$s, ALE (top) and FL (bottom)

8.2 Projectile impact problem

In this section, a typical projectile impact problem calculated on CORVUS is presented. The problem involves the normal impact of a 12 mm diameter spherical steel projectile travelling at $0.2cm\mu s^{-1}$ with a water tank. The tank is cylindrical with two 3.2 mm thick aluminium walls 250 mm apart at its front and back. The top and bottom walls are assumed to be rigid. CORVUS is used to simulate the trajectory of the projectile from its impact with the front wall to its final exit out of the tank through the back wall. The influence of mesh resolution on the trajectory is considered and a simple but effective mesh adaption strategy presented. But the main aim is to demonstrate the strengths of the multi-material ALE algorithm for projectile impact and penetration simulations.

8.2.1 Calculational Methodology

The problem was generated with the centre of the projectile positioned at the origin, but offset from the near tank wall by 0.6cm, leading to an initial impact with the near aluminium wall at $3.0\mu s$. An initial velocity of $0.2cm\mu s^{-1}$ was applied to the steel projectile. A tabular equation of state was used for the water. Both Steel and Aluminium were treated as elasto-plastic materials with a linear shock particle velocity equation of state, and Steinberg, Cochran and Guinan Constitutive model [42] used in both cases.

Simulations were performed for two different mesh resolutions, the coarser calculation employing 212x105=22260 elements with 6 elements initially across the aluminium walls and 10 across the diameter of the projectile. The mesh for the finer calculation was simply constructed by doubling up the coarse mesh, giving a total of 424x210=89040 elements. The projectile was explicitly meshed, as shown in Figure 8.6. The void regions were also meshed in front of, and behind, the tank to allow the mesh to be relaxed across the front and back interfaces of the tank walls. Winslow's equipotential mesh movement algorithm was applied to all regions in the problem, although nodes in the rear wall and in the void region behind it were constrained to not be allowed to relax until they had obtained a non-zero Lagrangian velocity.

However, initial calculations showed the motion of the projectile to be very sensitive to mesh resolution. So, in order to maximise the mesh resolution of the projectile for both the coarse, and fine mesh calculations, material weighted mesh movement was used with a material weight of 2 assigned to the projectile material and a weight of 1 to all other materials. A 5 cell deep buffer zone was also introduced around the projectile in order to improve the orthogonality and aspect ratio of the zones in projectile. This improved mesh quality also improves the accuracy of the momentum advection by reducing corner coupling errors and ensures

that the mesh *ahead* of the projectile is sufficiently well resolved to minimise diffusion errors. It should also naturally resolve gradients within the tank walls at impact. The effective resolutions obtained with this mesh movement strategy in the vicinity of the projectile was 0.25-0.5mm (fine calculation) and 0.5-1.0mm (coarse calculation), the resolution in the rest of the problem typically being 1.0mm (fine calculation) and 2.0mm (coarse calculation).

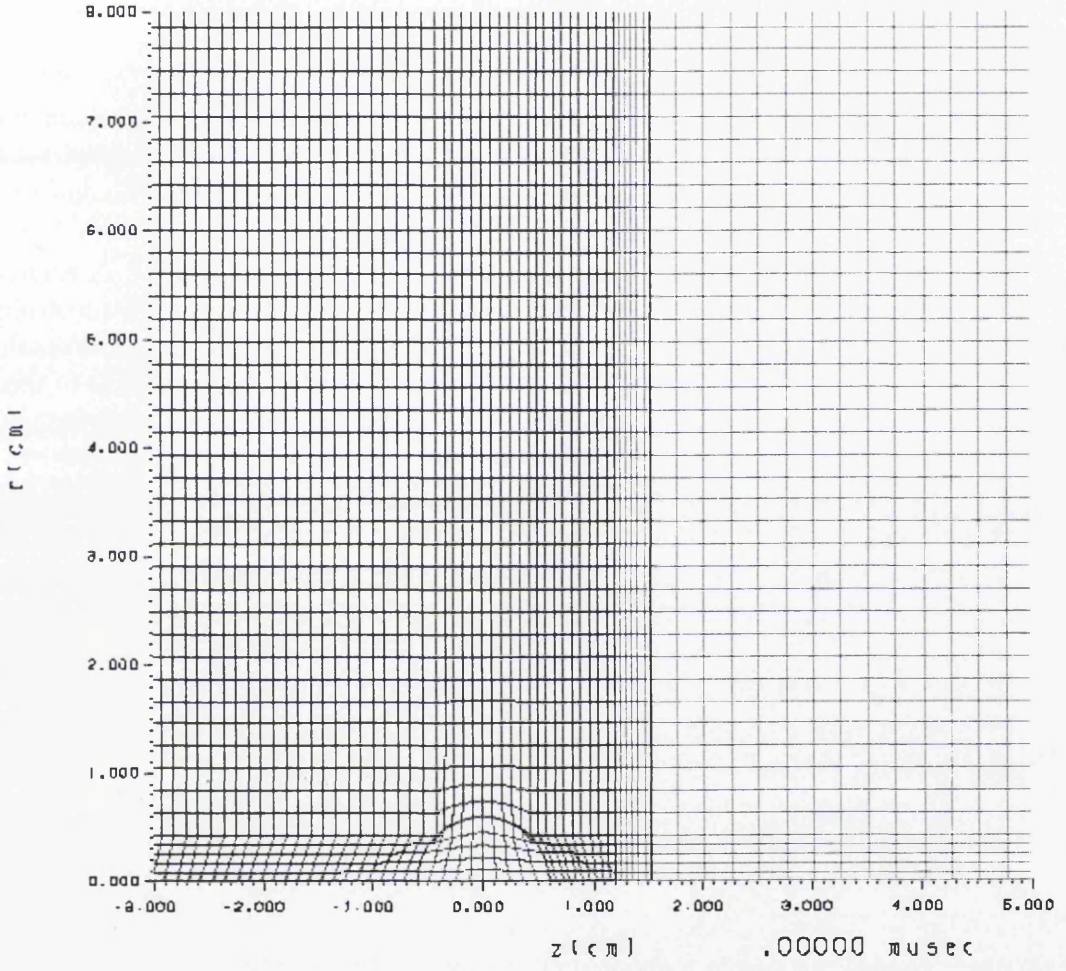


Figure 8.6: Initial Meshing of the projectile.

8.2.2 Numerical Results

Mesh and interface plots are given at 10, 200, 400 and 550 μs for the coarse mesh and the fine calculation in Figures 8.7 to 8.20. These mesh plots clearly confirm both the resolution estimates quoted in the previous section, and give a good feel for the quality of the mesh retained around the projectile throughout the run for both calculations. Figure 8.8 in particular illustrates how well the mesh refines in front of the projectile and conforms to the deformation of the target material during penetration.

The interface plots show the projectile to undergo very little deformation from penetrating either wall, with just a little flattening to its front. The interface plots of the complete problem at 550 μs for the coarse mesh calculation in Figure 8.17 and at 450 μs for the fine mesh calculation in Figure 8.19 clearly show the deformation predicted for the front and rear walls.

Distance against time plots are given for the projectile in Figures 8.21 for the two resolutions. The coarse mesh calculation predicts a significantly lower projectile velocity, with the projectile just reaching the rear wall by 550 μs. The fine resolution calculation impacts at about 325 μs and has broken out of the other side of the tank by 410 μs. Further finer mesh calculations, indicate that this fine mesh calculation is close to mesh converged in terms of the projectile break out time.

The projectile velocity against time is plotted for both resolutions in Figure 8.22. Again this emphasises the significantly lower projectile velocities in the coarse mesh calculation. However, it also shows oscillations, which are believed to be related to the reverberations of elastic waves within the projectile generated by the initial impact. The amplitude of these oscillations generally falls during the problem for both resolutions. However, the fine mesh calculation also shows an increase in amplitude at about the time that the reflected shock off the back wall reaches the projectile.

These results clearly demonstrate that adaptive multi-material ALE algorithm offers a robust and computationally efficient technique for the simulation of impact and penetration problems.

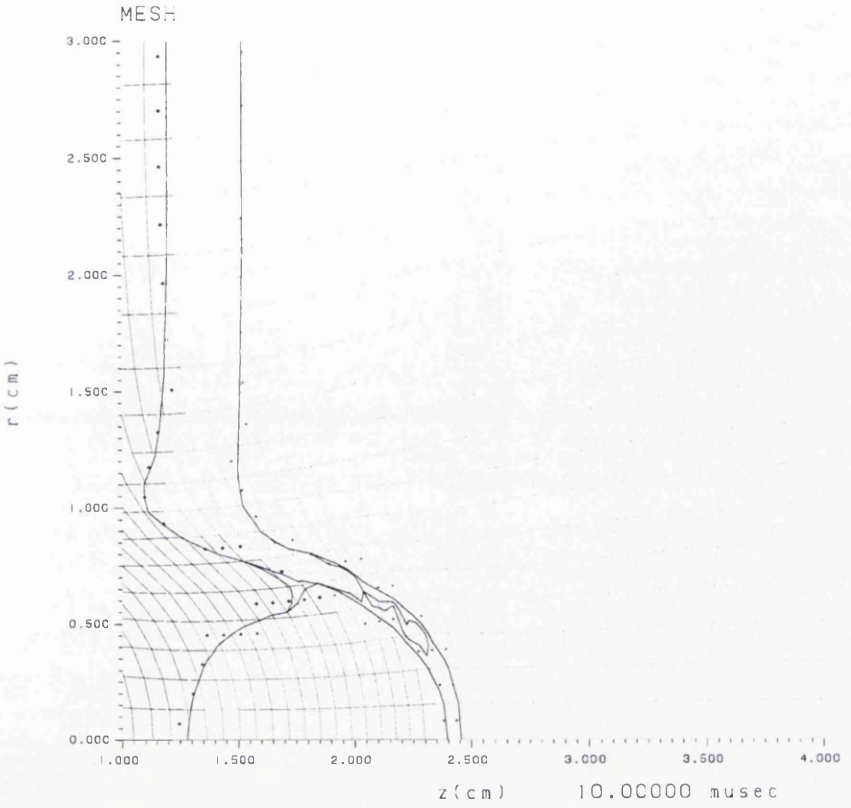


Figure 8.7: Mesh and interface plot at 10 μs for coarse mesh projectile impact problem.

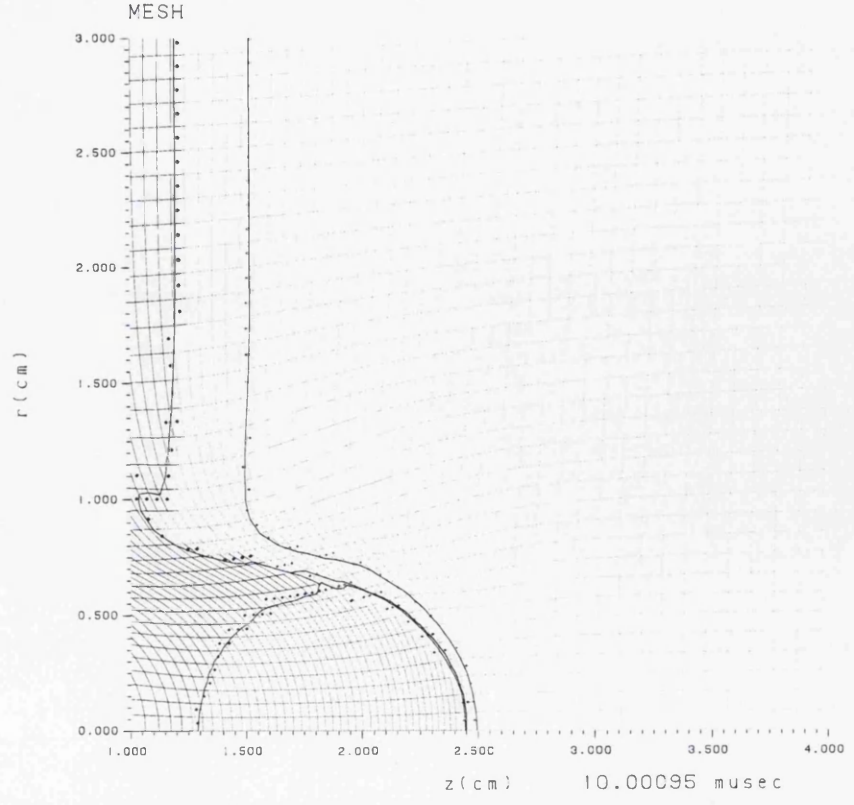


Figure 8.8: Mesh and interface plot at 10 μs for fine mesh projectile impact problem.

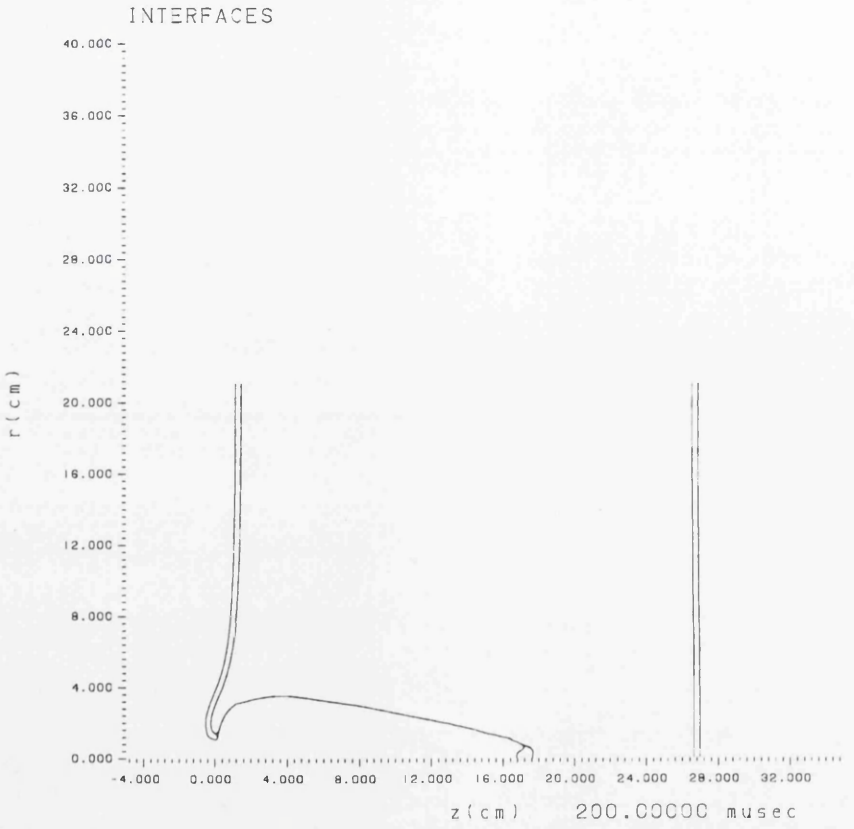


Figure 8.9: Interface plot at 200 μs for coarse mesh projectile impact problem.

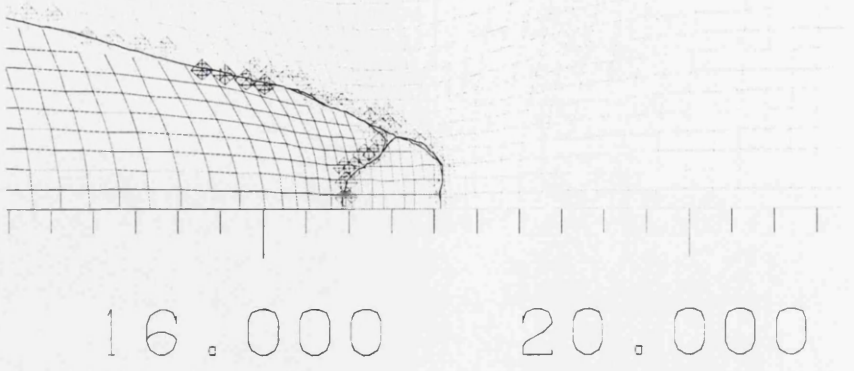


Figure 8.10: Mesh plot at 200 μs for coarse mesh projectile impact problem.

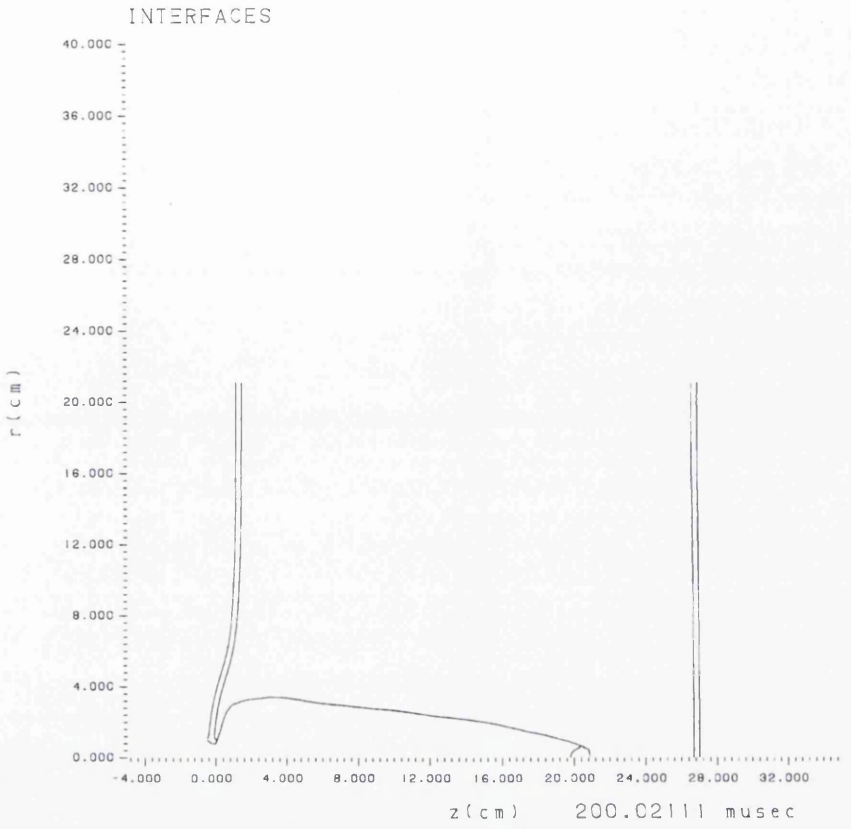


Figure 8.11: Interface plot at 200 μs for fine mesh projectile impact problem.

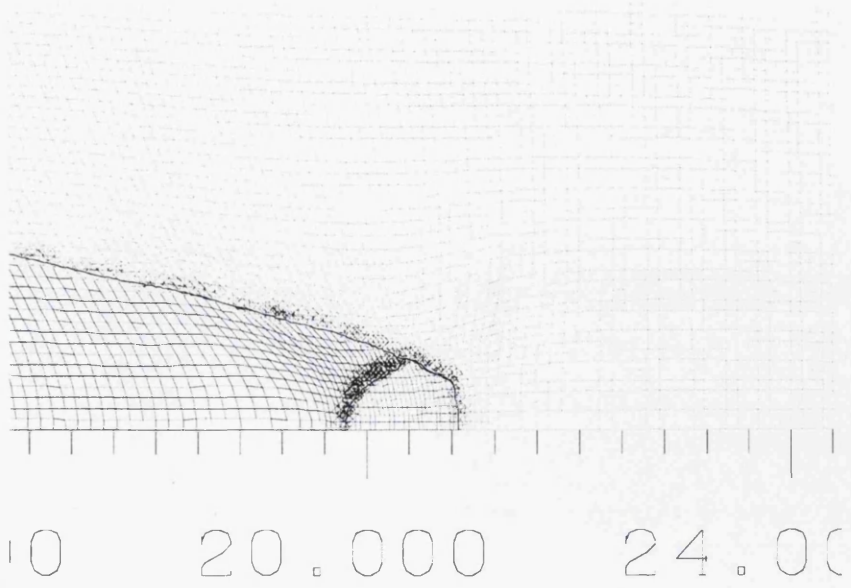


Figure 8.12: Mesh plot at 200 μs for fine mesh projectile impact problem.

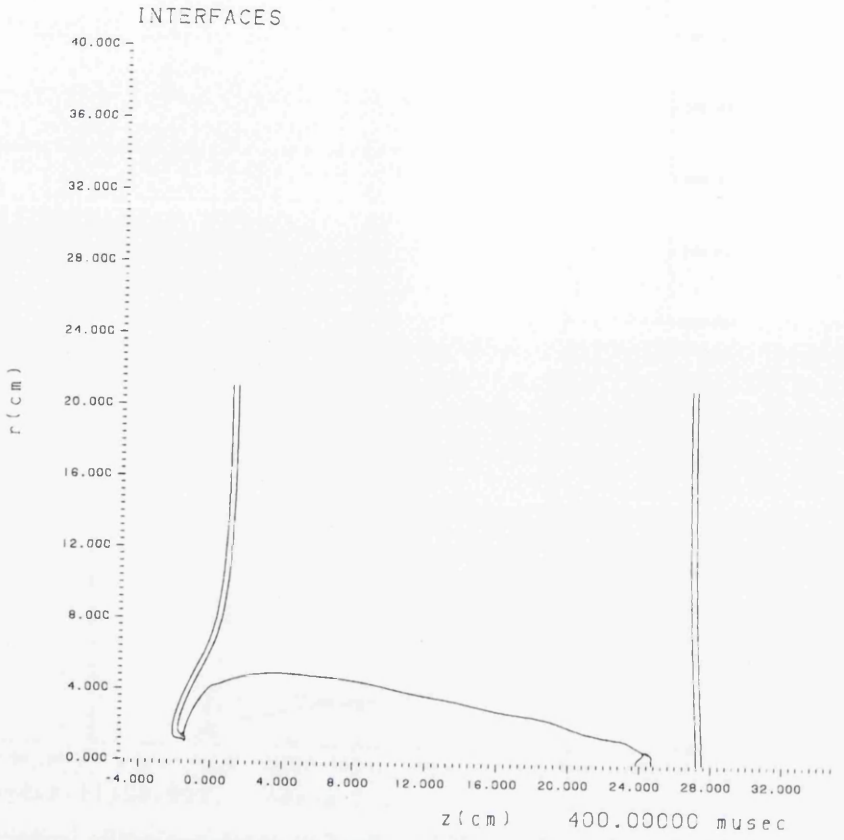


Figure 8.13: Interface plot at 400 μs for coarse mesh projectile impact problem.

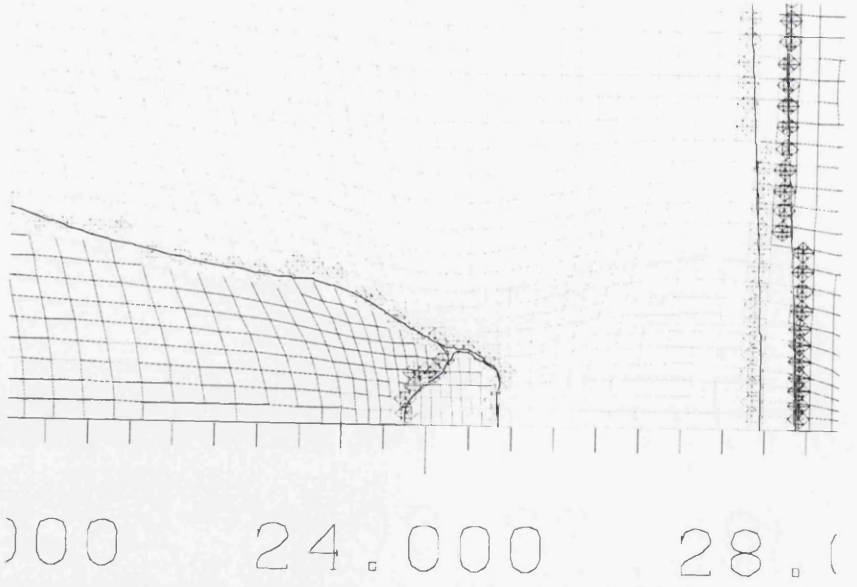


Figure 8.14: Mesh plot at 400 μs for coarse mesh projectile impact problem.

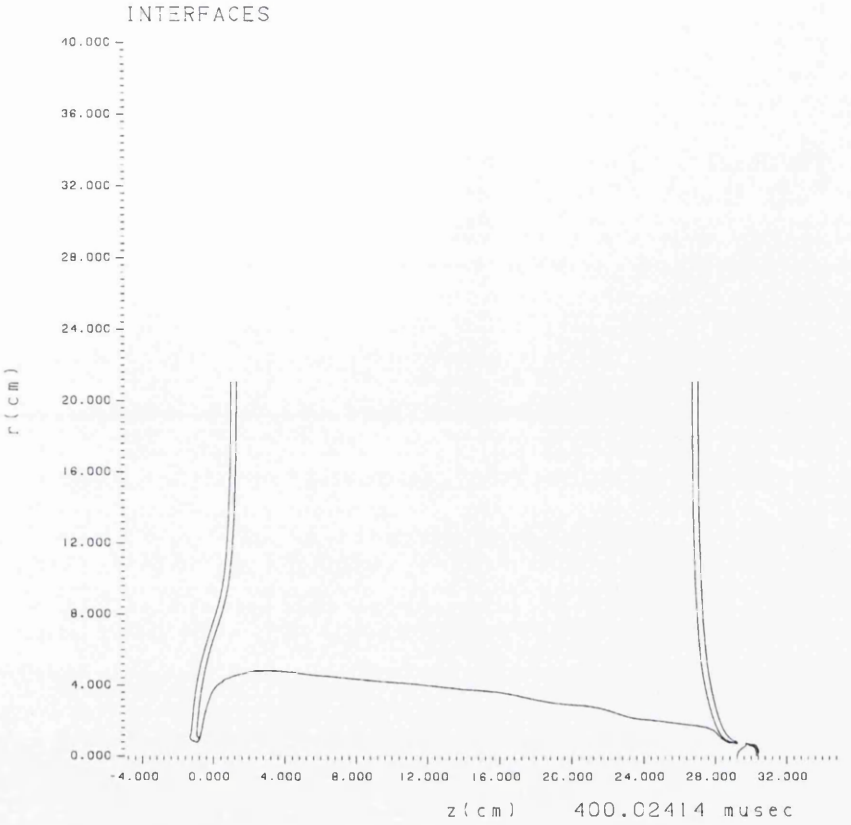


Figure 8.15: Interface plot at 400 μs for fine mesh projectile impact problem.

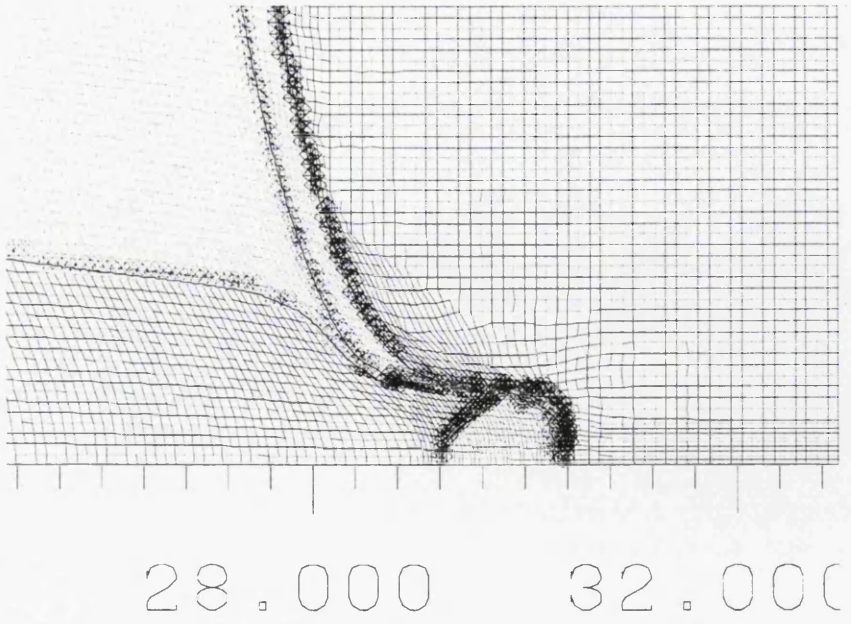


Figure 8.16: Mesh plot at 400 μs for fine mesh projectile impact problem.

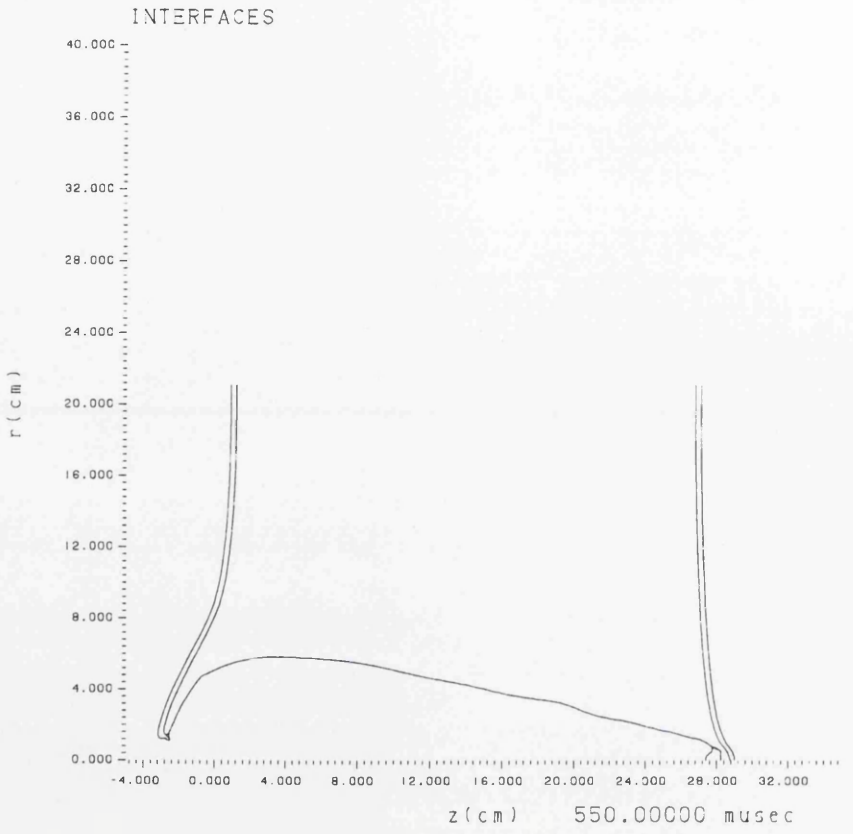


Figure 8.17: Interface plot at 550 μs for coarse mesh projectile impact problem.

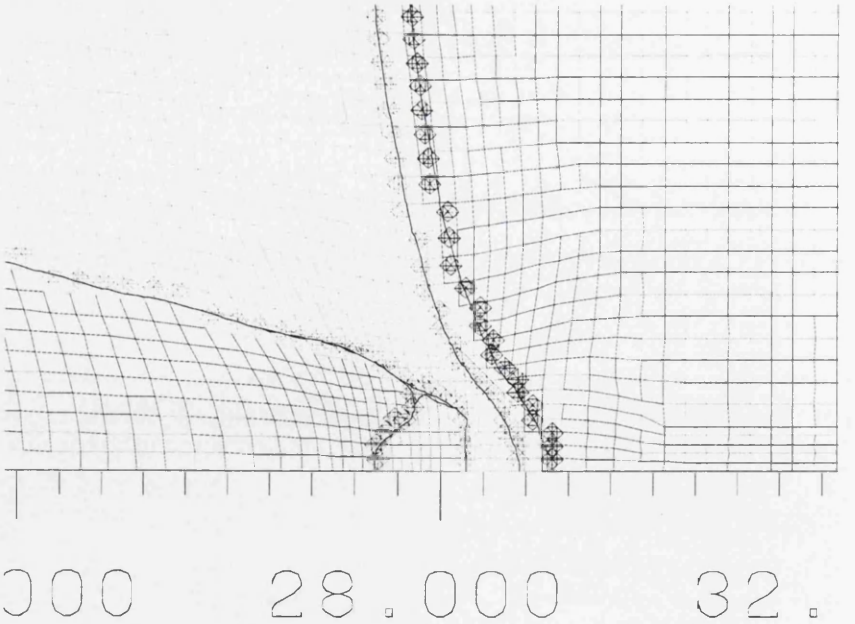


Figure 8.18: Mesh plot at 550 μs for coarse mesh projectile impact problem.

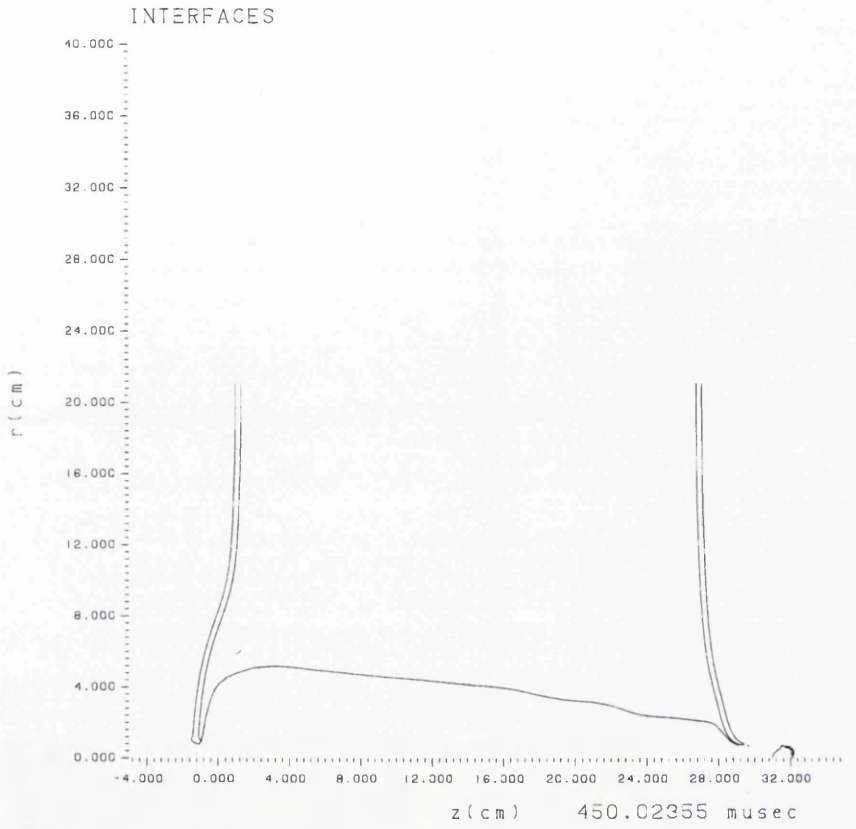


Figure 8.19: Interface plot at 450 μs for fine mesh projectile impact problem.

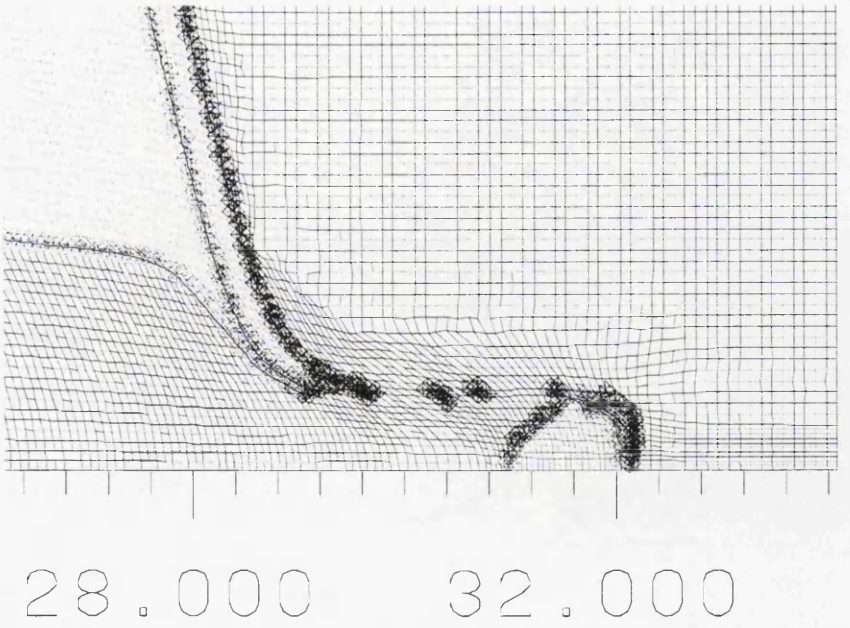


Figure 8.20: Mesh plot at 450 μs for fine mesh projectile impact problem.

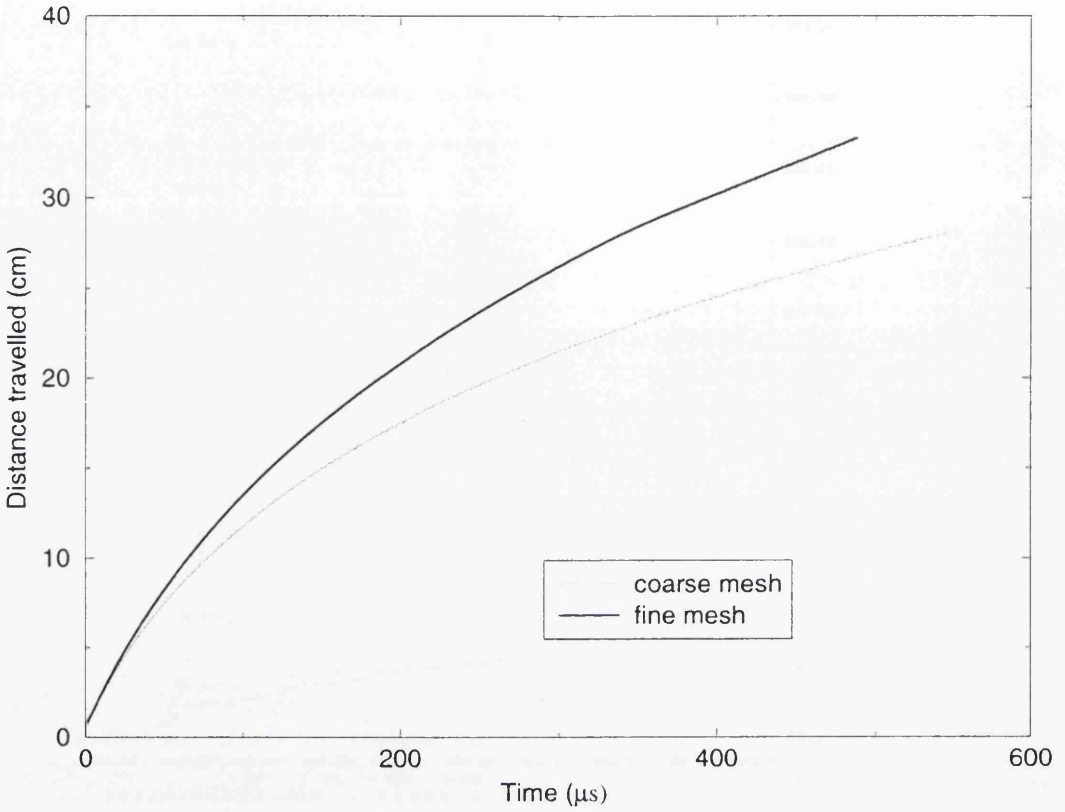


Figure 8.21: Calculated projectile position.

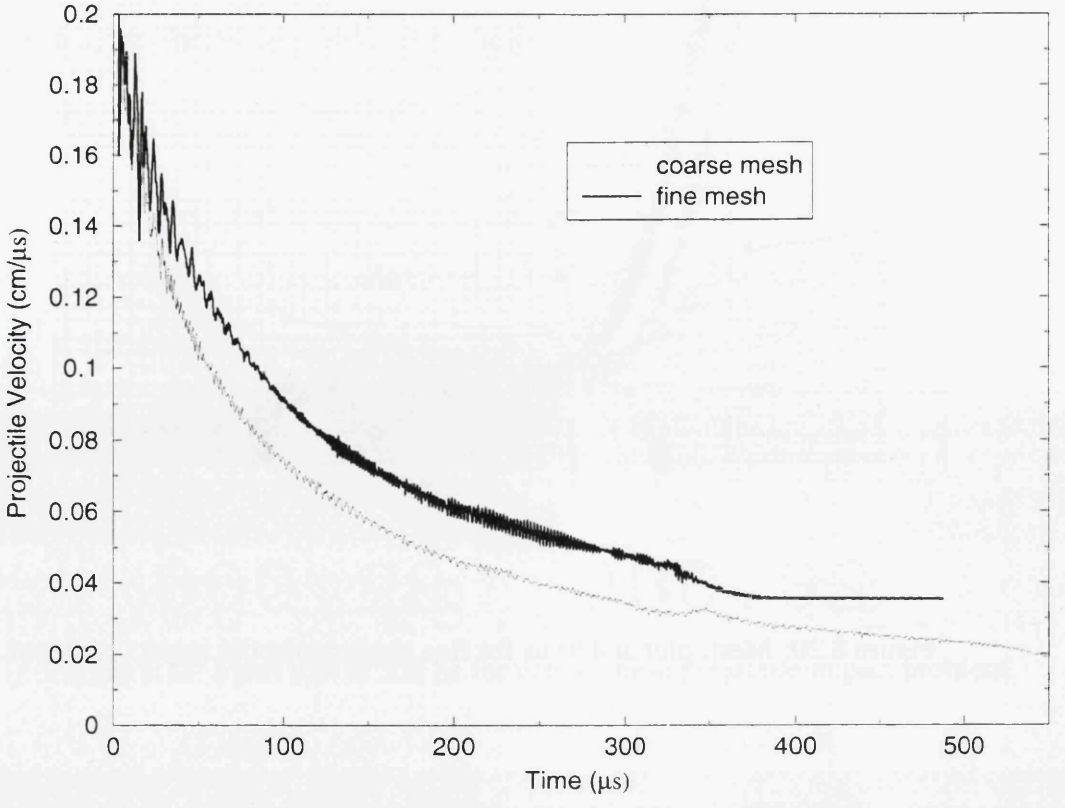


Figure 8.22: Calculated velocity history for projectile.

8.3 Dynamic friction experiments

Dynamic friction experiments are currently being performed at AWE to validate the friction model described in section 4.4. CORVUS is being used to model these experiments because it allows material interfaces either to be represented using an interface reconstruction technique or Lagrangian slide lines. The former offers robustness for the interfaces that undergo severe deformation, whilst the slide treatment potentially offers greater accuracy and allows the influence of friction to be assessed.

8.3.1 FN1

A schematic of AWEs first dynamic friction experiment FN1 is given in Figure 8.23 [52]. The experiment consisted of aluminium and steel target plates with a high explosive drive. The explosive drive was used to provide a strong shock in each of the target plates. Aluminium and steel were chosen for the target materials as they have significantly different shock speeds. A single fiducial marker was placed normal to the interface between the target plates 4.0 cm from the drive end. The fiducial was made of tantalum inside the aluminium and lead inside the steel to improve radiographic contrast. A single radiograph of the fiducial was taken at $40.0\mu s$.

In the absence of friction after the leading shock has passed, the fiducial should remain fairly straight in the aluminium with a break at the interface, due to the difference in shock speed between the two materials. However, if friction is present the fiducial will bend back, lagging towards the interface in the aluminium and bending forwards in the steel. The higher the friction, the more bending should occur. The aim of the FN1 experiment was to measure the bending of the fiducial marker and compare it against CORVUS calculations with different friction treatments. In addition to inferring the amount of friction present, the variation in curvature of the fiducial with distance away from the interface should also provide an indication of the distribution of plastic work near the interface.

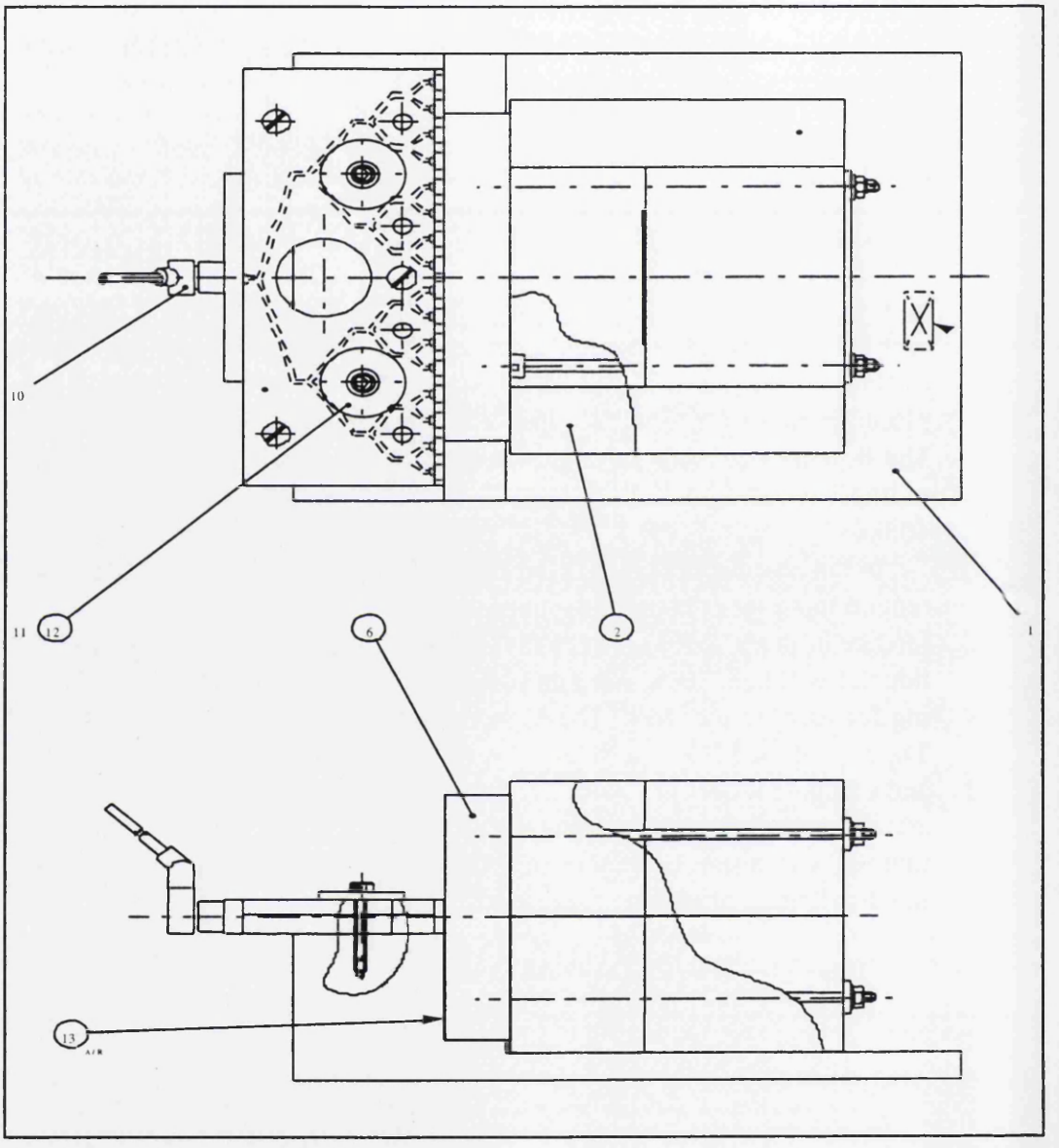


Figure 8.23: Schematic of FN1 experiment.

8.3.2 Calculational Methodology

Winslow's equipotential mesh movement algorithm was applied to all materials except the target plates. However, the nodes were also constrained to not be allowed to relax until they had achieved a non-zero velocity. The mesh relaxation, as required by Winslow's algorithm, was also reduced by a factor of $\frac{1}{4}$ to retain more Lagrangian character to the mesh. Lagrangian mesh motion was enforced throughout most of the target plates to enable the Lagrangian mesh lines to be compared directly with the experimental fiducial markers.

The principal difficulty in performing these calculations was that a Lagrangian slide line was required between the target plates to allow friction to be included, whilst for robustness the multi-material cell treatment was required for all the other interfaces. This required the slide line to be terminated at the T-junction in a robust and accurate manner: this was achieved by tying the last node on each of the two slide lines together at the T-junction, and then allowing mesh relaxation for the first few zones in the target plates, and across the interface between the HE drive and the target plates. It was also necessary to employ advective slide for the first few zones along the slide line adjacent to the T-junction.

8.3.3 Numerical Results from the FN1 Calculations

Preshot calculations were required, both to optimise the design of the experiment, and to determine the best time to radiograph. These calculations were performed for the two limiting cases of zero friction and a locked or merged interface treatment, in order to enable an assessment to be made by the experimentalists as to whether the two cases could be distinguished radiographically. The radiograph time of 40.0μs was chosen as it corresponded to maximum slip for the zero friction calculation. The initial mesh used for the pre-shot calculations is given in Figure 8.24. A mesh plot at 40.0μs is given in Figure 8.25 for a locked or merged interface calculation. The calculated fiducial bending from this calculation is plotted in Figure 8.26 providing a prediction of the maximum bending that could have been observed experimentally.

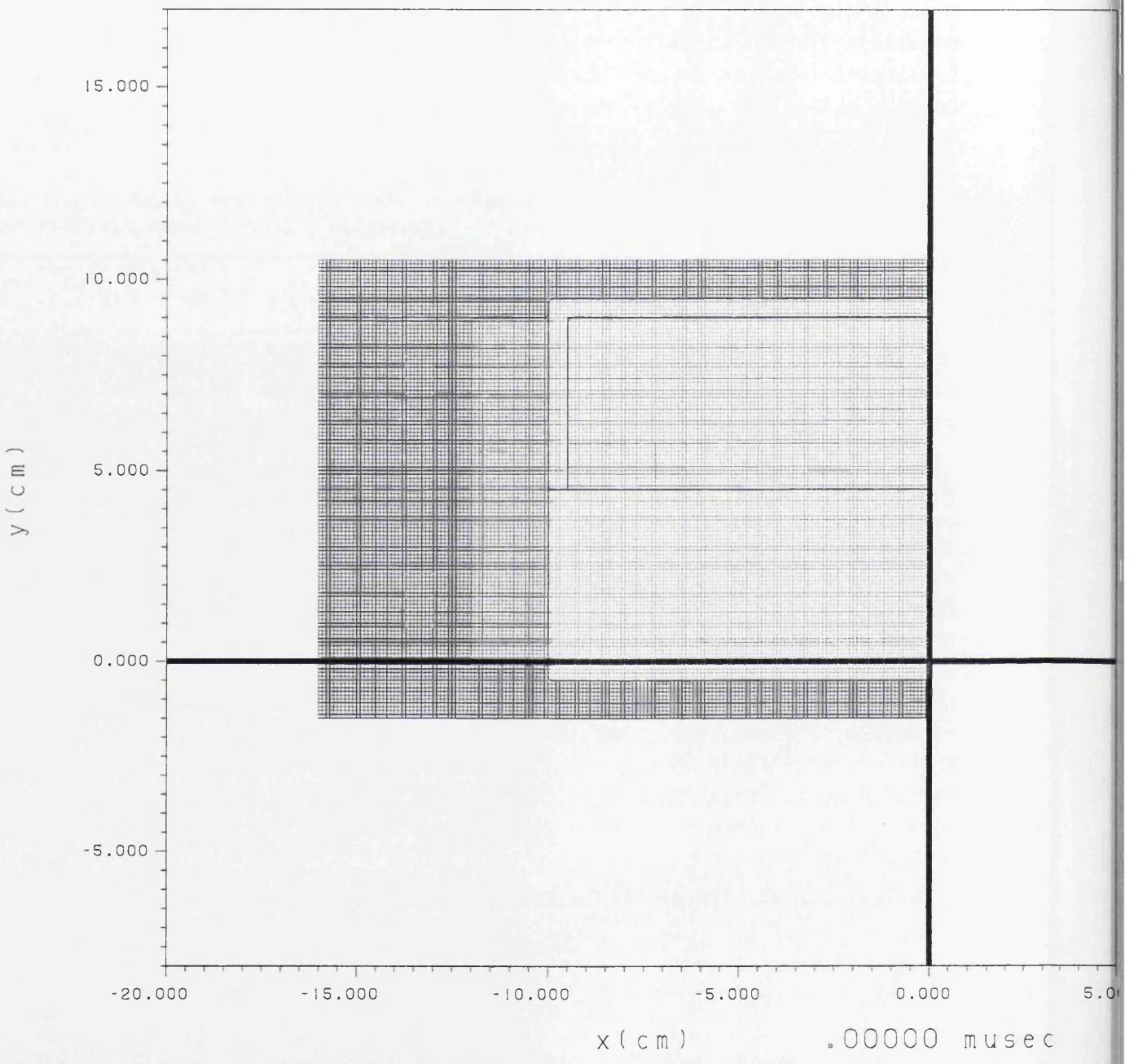


Figure 8.24: Initial mesh for FN1 calculations.

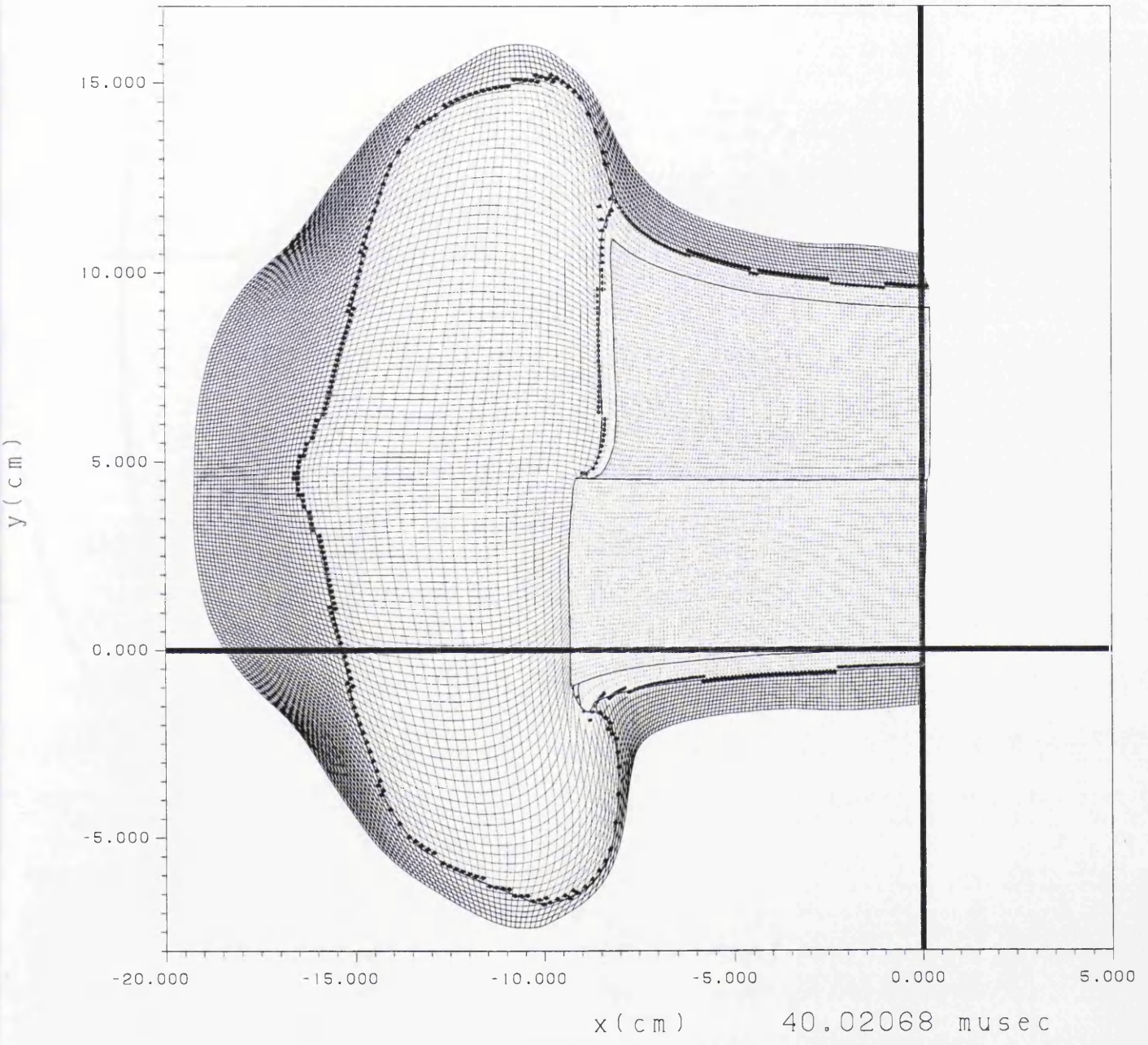


Figure 8.25: Preshot FN1 calculation with a merged interface treatment at 40 μs.

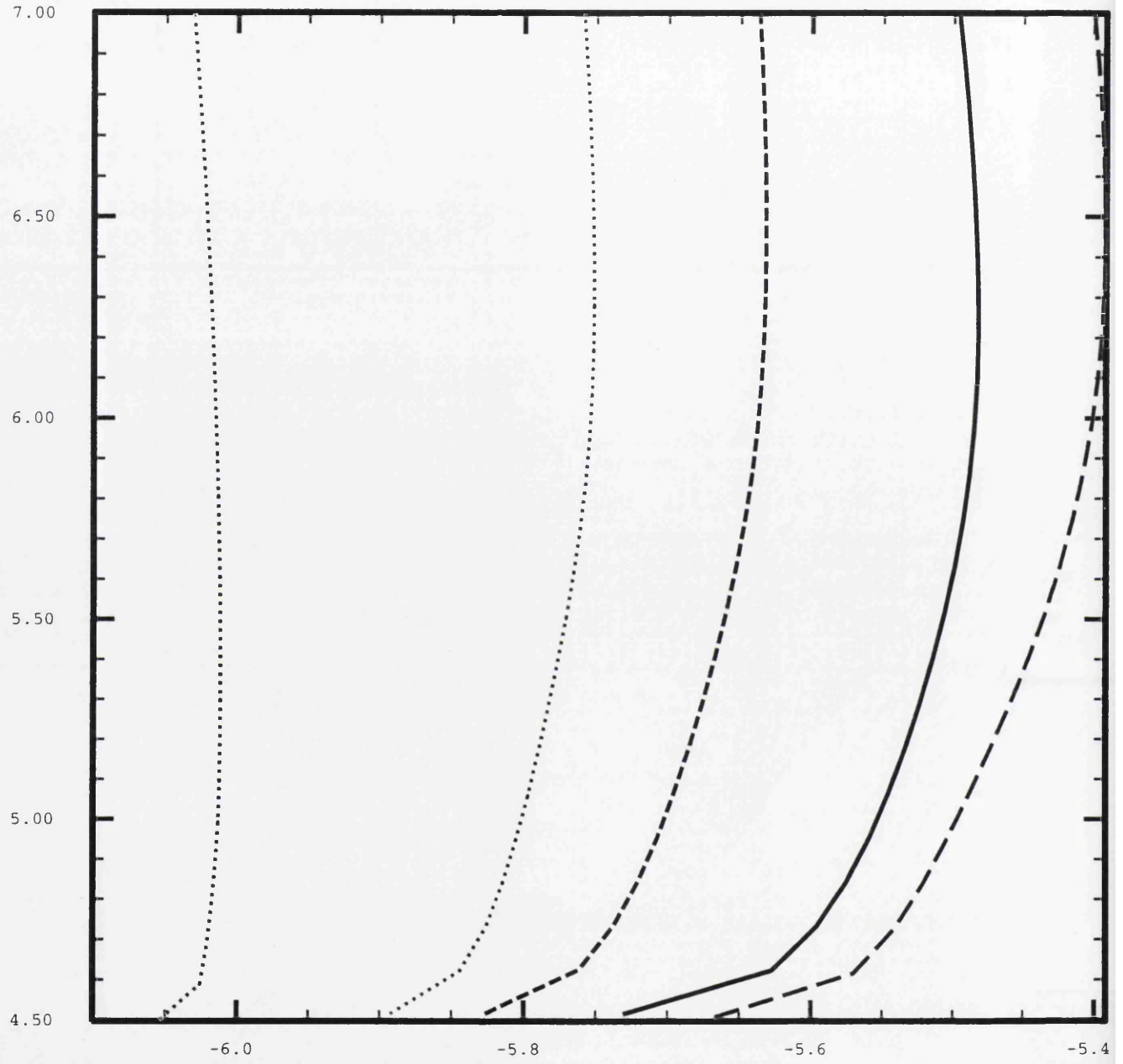


Figure 8.26: Calculated fiducial bending for a locked interface at 0, 10, 20, 25, 30, 35 and 40 μs.

FN1 was fired in April of 2001 with both a static and a dynamic radiograph at $40.0\mu s$ being successfully obtained, as shown in Figures 8.27 and 8.28. Unfortunately the experiment was only partially successful. The primary objective had been to prove the diagnostic approach, i.e. check that the fiducial markers could be seen radiographically. This first objective was achieved, as the markers can be seen clearly on the original static and dynamic radiographs, although they are less clear on the images reproduced here. The other objective was to collect some initial data that could be used for friction model validation. This was not achieved because the plates separated. This possibility had not been considered, and so void opening had not been allowed in the preshot calculations. Unless void opening is expected, this option is not normally enabled, as it can lead to robustness problems for some applications. The radiograph also showed no appreciable curvature to the fiducial.

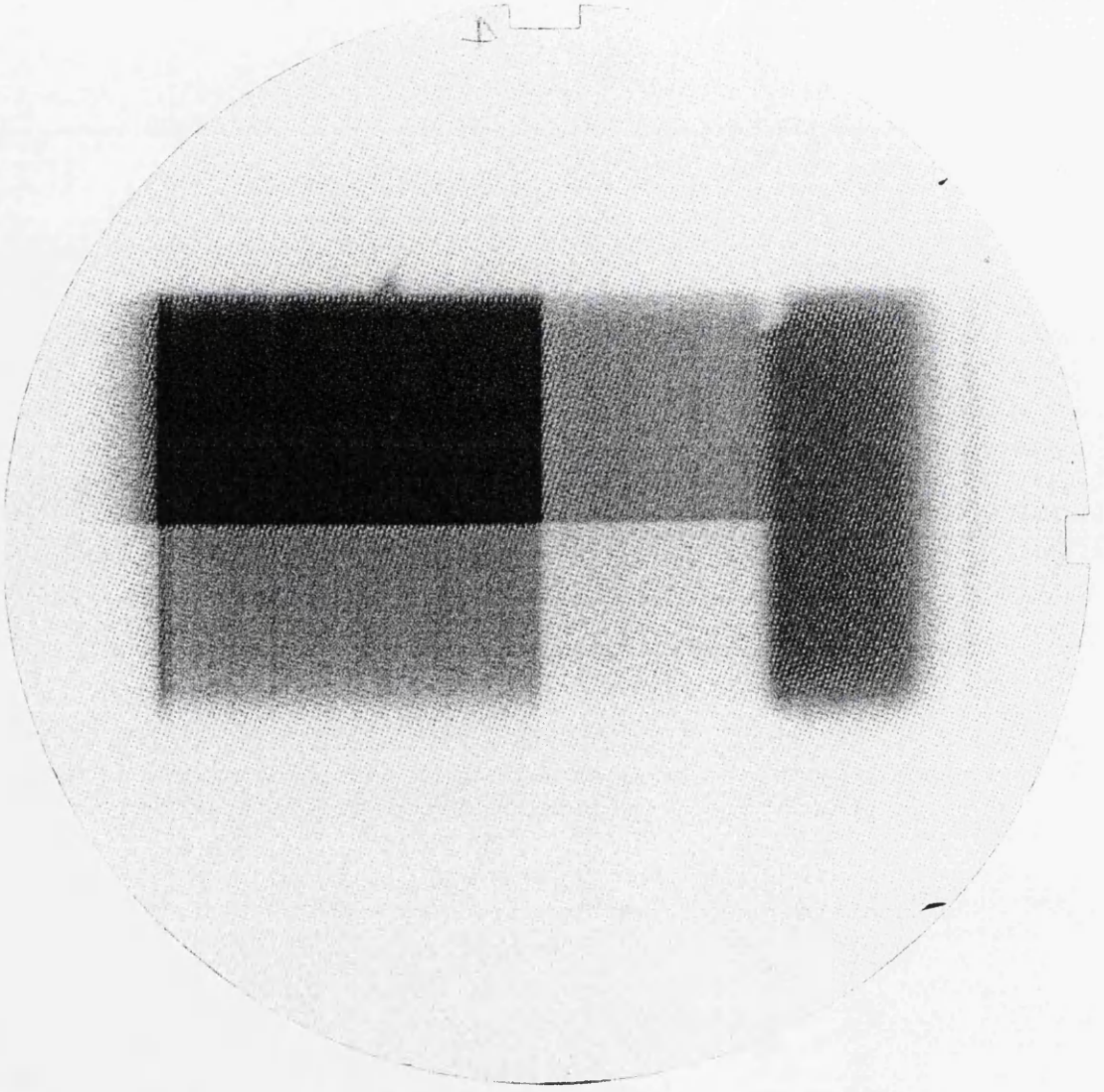


Figure 8.27: FN1 Static Radiograph.

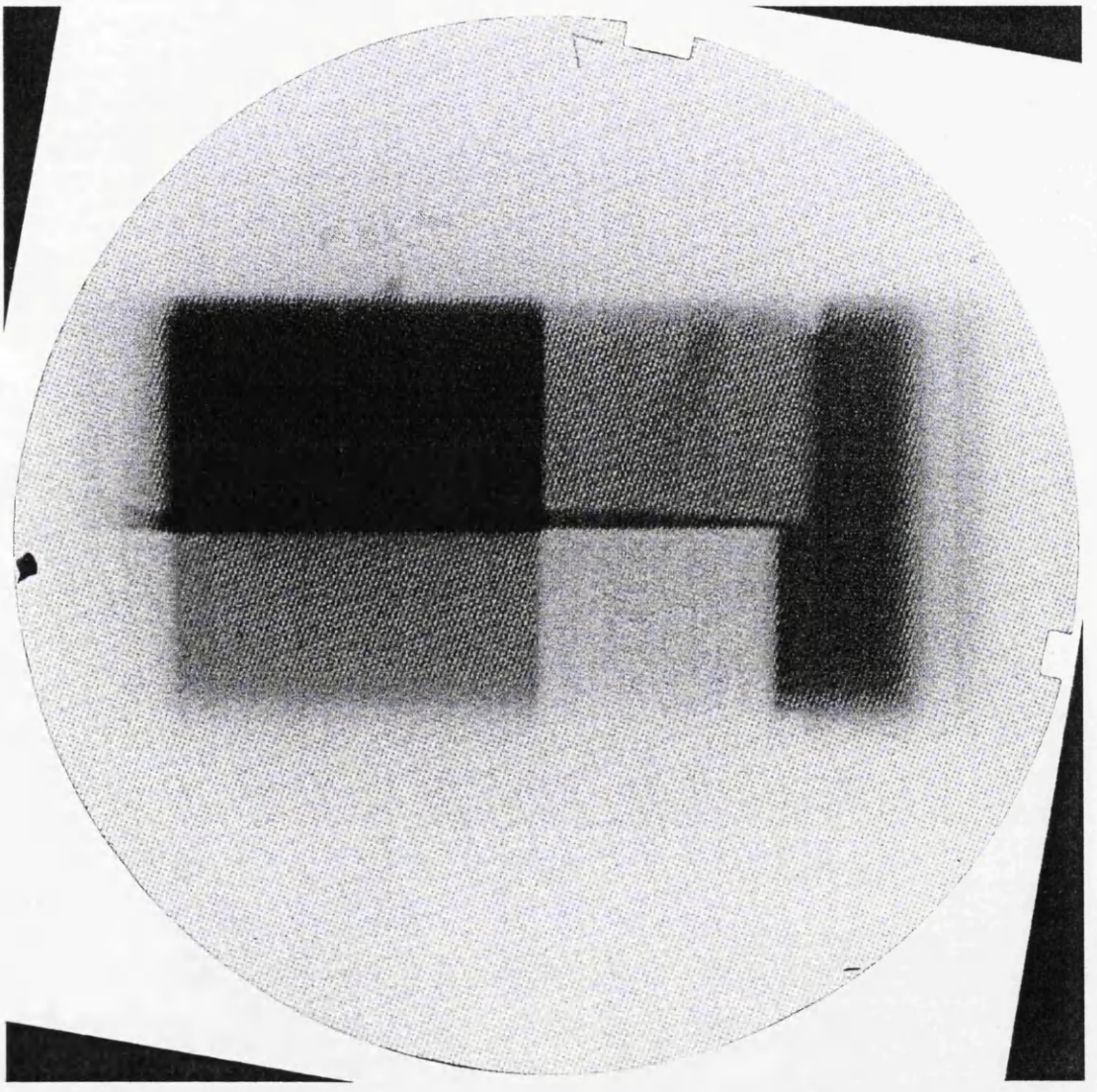


Figure 8.28: FN1 Dynamic Radiograph at 40 μs.

Post shot calculations were then performed with the void opening option enabled. A mesh plot from this calculation at $40.0\mu s$ is given in Figure 8.29. These calculations predicted void opening to have occurred between $10.0 - 15.0\mu s$. Given this prediction, and assuming that once the plates separated any fiducial bending would be frozen in, Figure fn1fid was re-examined to determine whether the experiment could still offer any data to discriminate between friction models. Unfortunately there is no appreciable difference at this time between the two interface treatments of zero friction and a merged interface treatment. Given these conclusions, further analysis of the data from FN1 was abandoned and effort was focussed on redesigning the experiment in order to delay void opening long enough to obtain useful data for friction model validation. However, it should be noted that the post shot calculation did provide a fairly good match in terms of the size of the gap that opened, and the relative displacement of the end of the two target plates at the free surface end. There was some suggestion that void opening at the target end may be under estimated, possibly due to the technique employed for terminating the slide line. However, the overall level of agreement was qualitatively very good, suggesting that it would be possible to determine by calculation if, when and where, the target plates would separate in future target designs.

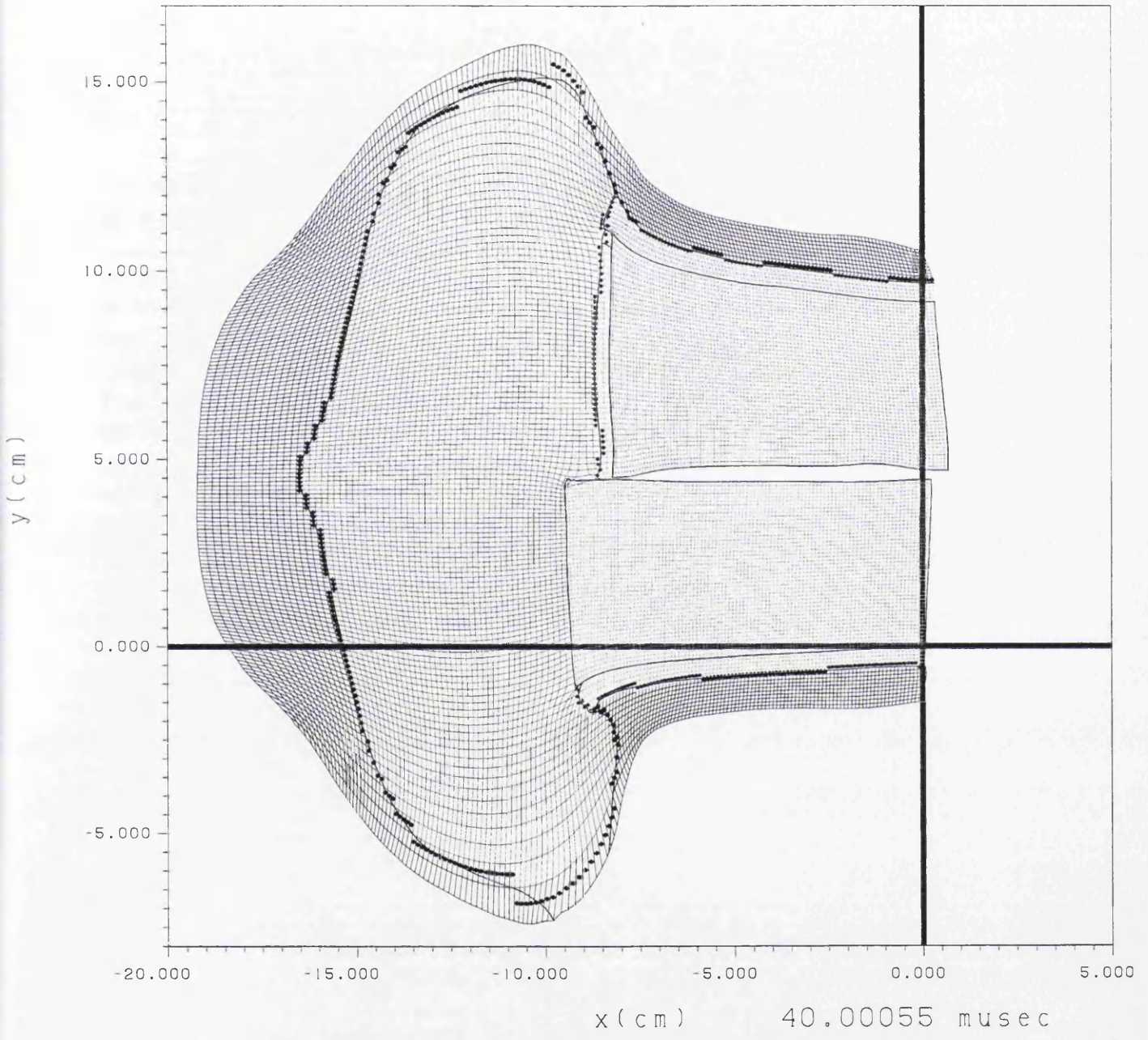


Figure 8.29: Post shot FN1 calculation with slide and void opening at 40 μs.

8.3.4 FN1/2 Calculations

FN1/2 was designed by the author using CORVUS in order to delay the void opening that occurred in FN1 [52]. The aim was to allow sufficient fiducial bending to occur in the merged interface calculations to suggest that it would be possible to discriminate experimentally between different friction models. The new design was also required to retain as many of the original components from FN1 as possible, to enable the follow on experiment to be fired without delay. This was achieved simply by adding an additional steel plate, identical to that used in FN1, onto the other side of the aluminium to form a three plate target. The HE dimensions were also unchanged from FN1, but the charge was moved up and re-centred on the aluminium plate.

The initial meshing and geometry for a preshot calculation of FN1/2 is given in Figure 8.30. FN1/2 was calculated with zero friction and void opening enabled, to establish how much time was available for the fiducial bending. Mesh plots from this calculation are given for $25.0\mu s$ and $30.0\mu s$ in Figures 8.31 and 8.32 respectively. These results show the plates start to separate at $\approx 25.0\mu s$ which is much later than for FN1. The plates also separated on release from the far end of the target away from the HE drive and so remained in contact at the fiducial site until about $30.0\mu s$, as can be seen in Figure 8.33. Therefore the best radiograph time should be somewhere between 25.0 and $30.0\mu s$. In order to determine whether such a radiograph time would offer sufficient model discrimination, a calculation was then performed with a merged interface. The calculated fiducial shapes for both zero friction and a merged interface, and a range of possible radiograph times, are presented in Figure 8.34. This shows that, at a conservative radiograph time of $25.0\mu s$, there is maximum difference of $\approx 1.4mm$ in fiducial position at the interface between the two treatments. The curvature for the merged interface calculation spreads over $\approx 1.5cm$. These differences should be resolvable using flash X-ray radiography.

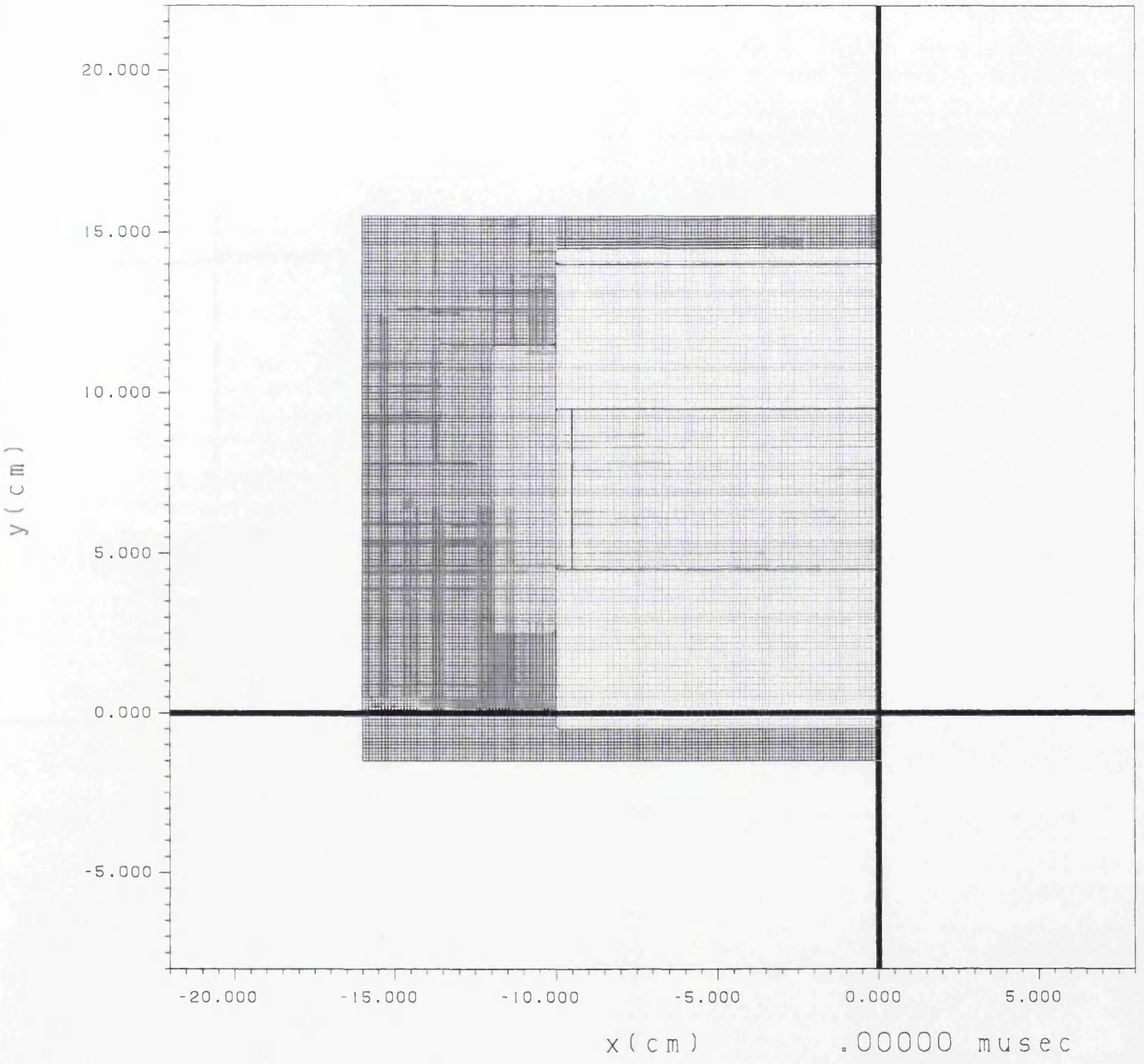


Figure 8.30: Initial mesh for FN1/2 calculations.

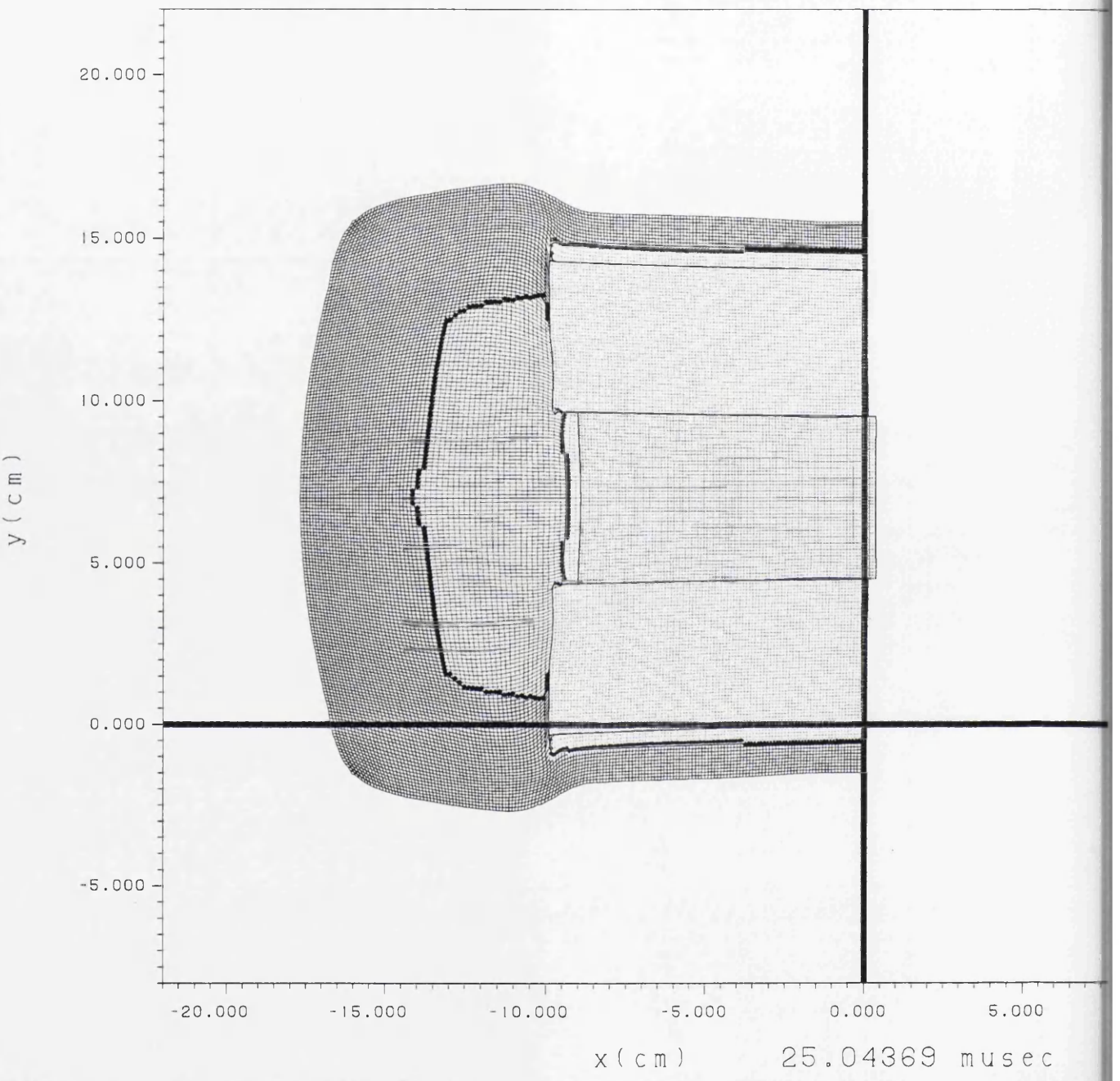


Figure 8.31: Preshot FN1/2 calculation with slip and void opening at 25 μs.

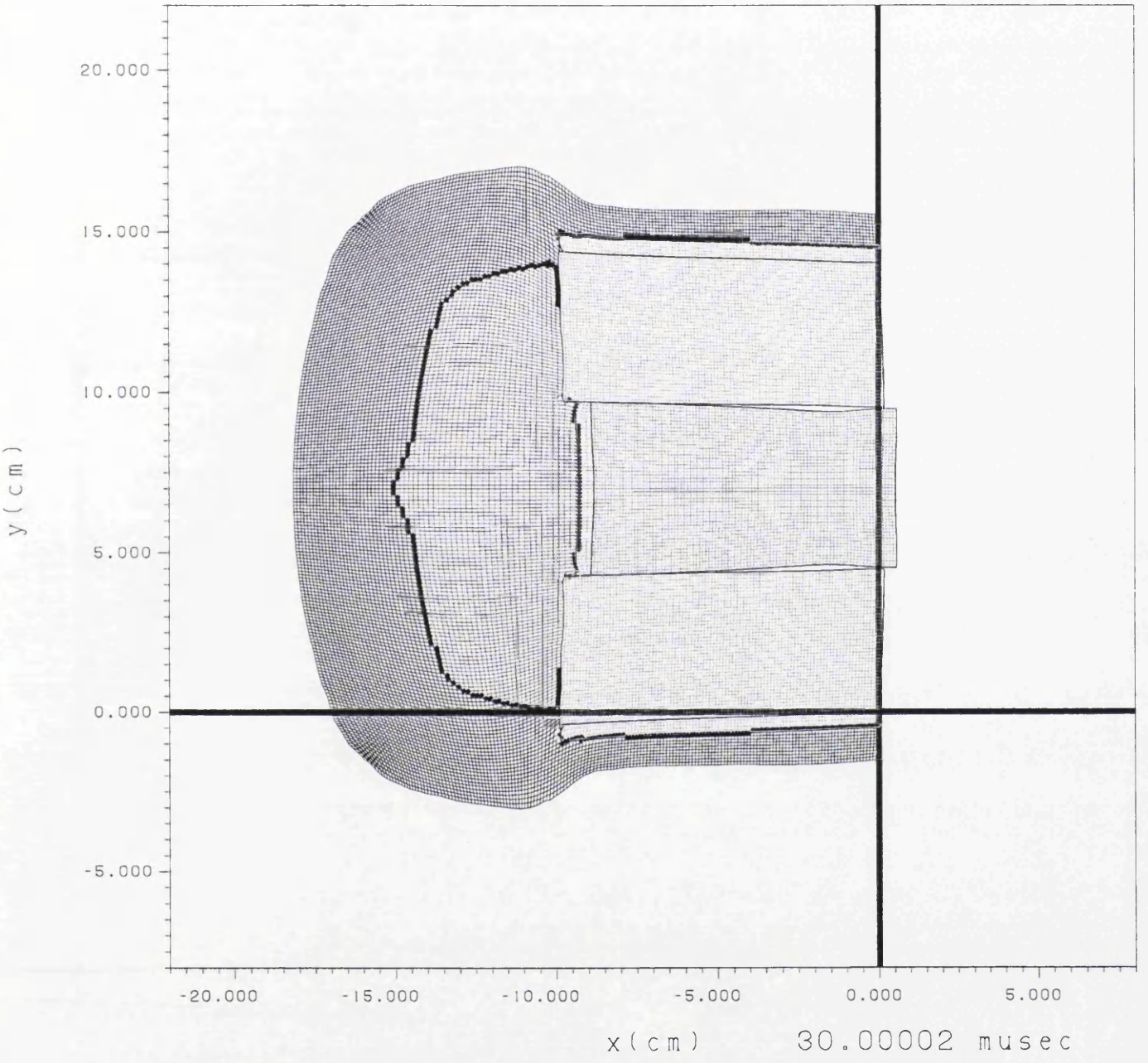


Figure 8.32: Preshot FN1/2 calculation with slip and void opening at 30 μs.

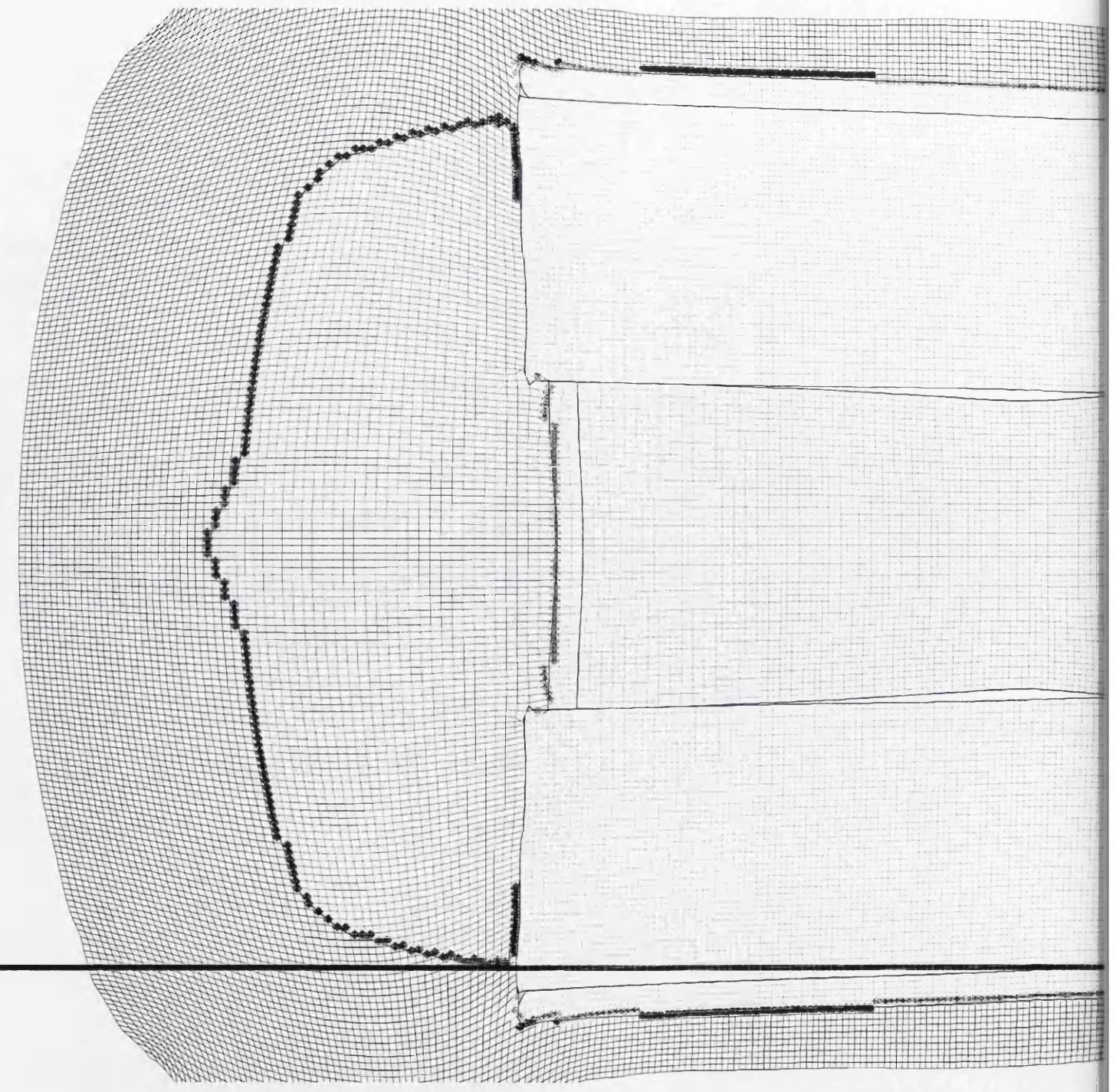


Figure 8.33: Blow up of FN1/2 calculation with slip and void opening at 30 μs showing the extent of the void opening.

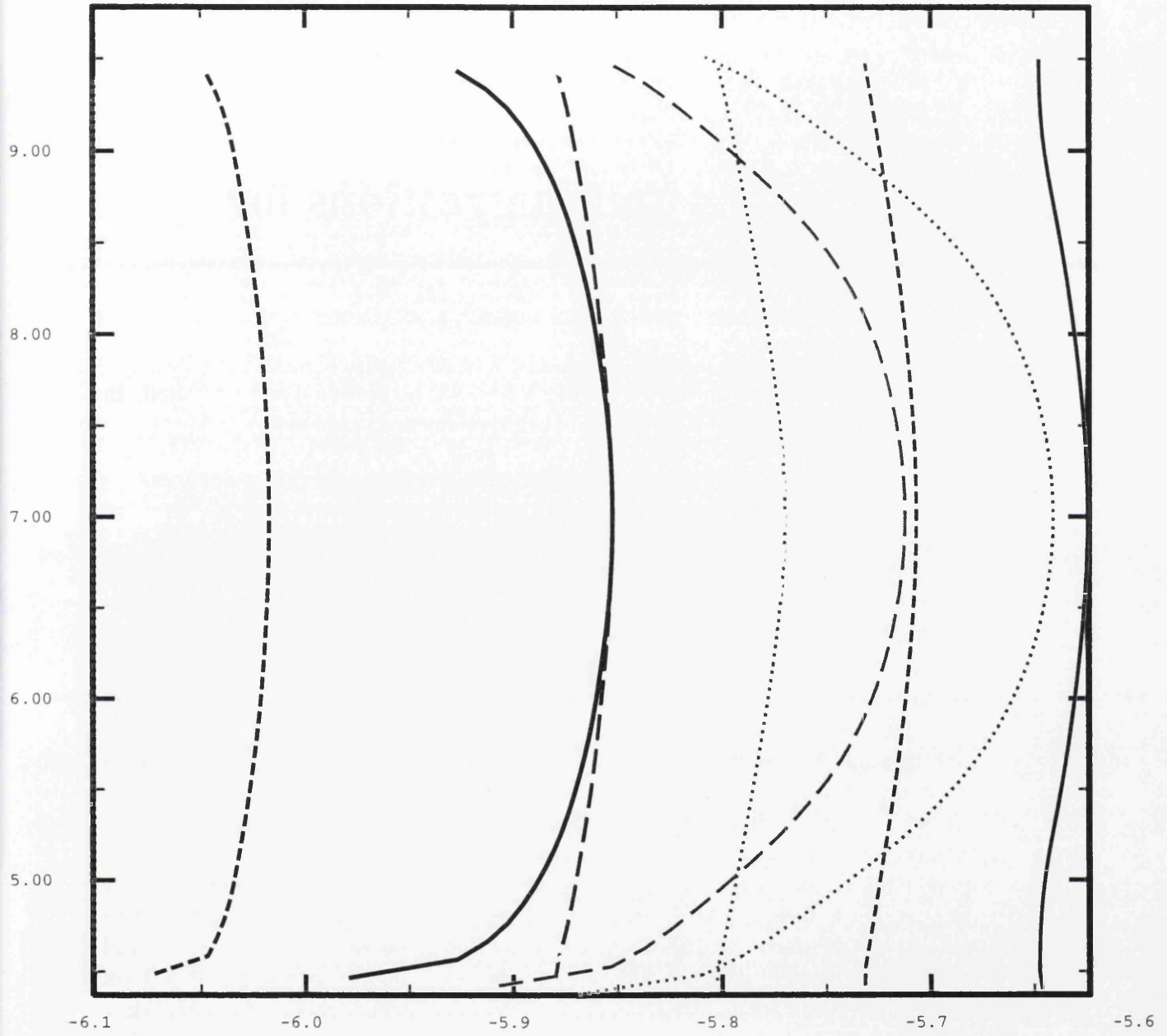


Figure 8.34: Calculated fiducial bending for true slip and a locked interface at 0, 10, 15, 20, 25 and 30 μs.

Chapter 9

Conclusions and Suggestions for Future Work

In this chapter, the findings from this study will be collected and reiterated. In addition, recommendations for possible further work are suggested.

9.1 Conclusions

Hydrocodes are used to simulate unsteady compressible flow problems that may involve solid materials which can undergo severe deformation. This requires numerical methods that can capture shock waves and material interfaces accurately. It also requires the inclusion of interface physics such as slip, void closure, void opening and friction. The numerical methods used in hydrocodes must also be compatible with general equations of state and allow the inclusion of the influence of material strength. Although Lagrangian and Eulerian schemes are the most widely used techniques for hydrocode applications, there is also third type of method, the Arbitrary Lagrangian Eulerian (ALE) technique, which potentially offers significant advantages over the other two techniques for many hydrocode applications. However, most of the ALE methods that appear in the literature are Simple Arbitrary Lagrangian Eulerian (SALE) schemes, where material interfaces are restricted to remain Lagrangian. This limits the accuracy and robustness of SALE codes and the range of applications to which they can be applied. The purpose of this work was to develop an adaptive multi-material Arbitrary Lagrangian Eulerian (ALE) hydrocode, where material interfaces are not constrained to remain Lagrangian. The most novel aspects of this work are concerned with the numerical treatment and modelling of material interfaces in an ALE code.

The adaptive multi-material ALE algorithm has been implemented in an existing 2D staggered grid Lagrangian hydrocode CORVUS, which was developed at AWE

by Whittle [29]. This was possible because the adaptive multi-material ALE algorithm splits each timestep into separate Lagrangian and advection phases. A split approach was preferred, as opposed to solving the ALE equations directly in conservative form. This simplifies the treatment of material interfaces and the inclusion of additional physics. The CORVUS Lagrangian step is fully described in chapter 3. In summary however, the Euler equations are solved in Lagrangian non-conservative form, using a predictor corrector time discretization and explicitly integrated bilinear isoprametric quadrilateral finite elements for the spatial discretization.

The only change required to the original Lagrangian step to implement the adaptive multi-material ALE algorithm, was to make it compatible with the presence of multi-material cells. However, during the course of this study some improvements were also made to the Lagrangian hydro scheme. The most significant of these was the introduction of a monotonic artificial viscosity. This significantly improves the shock capture of the scheme, making it comparable with second order Godunov methods, as illustrated by the bubble/shock interaction problem in chapter 8. The original DYNA-like anti-hourglass filter used by Whittle was also modified in a dimensionally consistent manner, to remove its dependence on the material sound speed. This has significantly improved the performance of the scheme for long thin aspect ratio zones. Finally, spherical symmetry for axisymmetric problems was found to be improved by the author, if the divergence of the velocity field used in the internal energy update was replaced by the rate of change of the cell's actual volume. A number of idealised test problems are also presented to demonstrate the quality of the solutions that can be obtained with the underlying Lagrangian scheme.

In the author's opinion, one of the great strengths of the adaptive multi-material ALE algorithm is that it provides two different schemes for modelling material interfaces. In chapter 4 the Lagrangian slide algorithms are described. These methods are potentially more accurate than multi-material cell based methods, because they restrict nodes to always lie on the material interface. This enables force and mass contributions to be calculated directly at the interface and the tangential force contributions to be decoupled. Inevitably however, these methods will suffer robustness problems when a material interface undergoes severe deformation. Slide algorithms also provide a more natural framework for the inclusion of interface physics than multi-material cell based interface schemes. The slide algorithm used in the multi-material ALE algorithm was developed by Whittle [29] and is fully described in chapter 4.

The author's first real involvement in the development of CORVUS came during the extension of the slide algorithm to include void closure and void opening. The author made two main contributions here, the first of which was the suggestion that an oblique impact at close to normal incidence between two metal plates would provide a 2D test of the accuracy and robustness of the void closure algorithm. The solution to this problem should approximate that of a normal impact: a

problem which can be solved analytically. This 1D analytical solution could then be used to assess the 2D numerical solution. In practice, the main difficulty encountered by Whittle [29] in extending the slide algorithm to provide void opening and void closure, was a tendency for a slide line to reverberate opening and closing for numerical reasons, rather than as a result of the physics being simulated. The author proposed the solution to this problem, whereby if the slide surface acceleration indicates that the slide line should open, the pressures in adjacent elements are examined to determine whether or not the interface is in tension. If these elements are all in tension, then the slide line is allowed to open, otherwise it remains closed.

However, the author's main contribution to the slide line algorithms used in CORVUS has been in the field of dynamic friction. Dynamic friction refers to the physics that govern the relative tangential motion of two strong materials in contact when a shock wave propagates across the interface. This physics is not well understood and is an area of active research. It has been speculated by the author that a boundary layer is formed in the weaker material, where the plastic work is focussed. This could be as a result of changes in microstructure, dislocation dynamics, shock welding or adiabatic shear. However, the focussing of plastic work into this boundary layer may then lead to a local reduction in yield strength, producing more local shear or slip than would otherwise be predicted for a locked interface.

A conventional slide algorithm assumes there is no friction or tangential force coupling the two materials. If one or both the materials have no appreciable material strength then this is a good approximation. In addition, a Lagrangian interface can also be treated as merged, which is equivalent to the maximum amount of friction that could be present. However, it is speculated that reality lies somewhere between these two limits, and what is really required in a hydrocode is an interface constitutive law, which adequately represents the physics that actually occurs. In order to start to investigate dynamic friction, the CORVUS slide line algorithm has been modified by the author to allow a frictional force to be introduced. The current model is very simple, including three terms that can be applied in isolation or as a linear combination. The first term is simply a function of the normal force acting at the interface, while the second two terms are functions of the tangential velocity difference across the interface. Given user defined friction coefficients, the frictional force is then evaluated for the nodes on one slide surface. The magnitude of the force is then limited so that it cannot exceed the frictional force required to lock the interface. An equal opposite reaction is then mapped to the opposite surface to make the scheme conservative and preserve symmetry. Experiments are currently being performed at AWE to provide data to validate, and if necessary, to refine this model. The calculations used to design and analyse these experiments have all been performed on CORVUS by the author, some of which were presented in chapter 8.

In chapter 5, the mesh adaption algorithms that have been developed by the author for CORVUS, were presented. Two different mesh adaption strategies are available: mesh movement and mesh insertion. The former is principally used to

improve mesh quality and robustness: the latter is used to maintain resolution in one logical mesh direction for convergent flow problems. The mesh movement algorithms that have been developed are very flexible and can be used in a number of different ways. However, in general the author has found that best approach is to constrain the mesh movement to retain as much Lagrangian character to the mesh as possible. The mesh movement options that have been developed can be subdivided into algorithms for interface and boundary node movement, node movement constraints, internal node movement and weighted node movement. Winslow's equipotential algorithm is used to moved the internal nodes. However, the quality of the meshes generated by this algorithm have been found to depend strongly on the positions of the nodes on the boundaries of the region to which Winslow's method is applied.

In order to improve the mesh quality obtained using Winslow's scheme, a number of methods have been developed by the author for repositioning nodes along slide lines before Winslow's method is applied. The best overall method of this type that has been developed is one which adaptively repositions slide nodes in arc length along slide surfaces, whenever the minimum arc length between the slide nodes is smaller than some user defined fraction of the mean node spacing. This approach has been shown both to improve the quality of the meshes generated, and to allow slide lines to be retained in calculations for longer, under more severe interface deformation. This also increases the range of interface physics options that can be applied and introduces more Lagrangian character into the mesh. The retention of more Lagrangian character in the meshes generated by the mesh movement algorithms is beneficial, as it naturally refines and aligns the mesh with flow features of importance, such as shock fronts. It should also reduce advection errors and numerical diffusion. The Lagrangian character of the mesh has also been improved by introducing a number of mesh movement constraints to limit when and by how much nodes can be moved by Winslow's algorithm. Winslow's mesh movement algorithm has also been modified to allow a user defined material or region weights to be applied. This provides a penalty based mesh refinement capability, which will pull zones into materials or regions of interest to the user, a technique which was demonstrated to be very effective on the bullet impact problem presented in chapter 8.

In convergent flow problems the computational domain of interest generally changes aspect ratio as the problem evolves. This leads to a reduction in resolution in one direction, which cannot be improved by mesh movement. In order to address this issue an automatic mesh insertion option has been added to CORVUS by the author. This option simply inserts additional mesh lines in one logical mesh direction, as required by a aspect ratio error estimator. An efficient integral rezoner method has been used to implement this scheme, based on the work of Dukowitcz and Ramshaw [60, 61, 62, 63]. The method has been demonstrated on 1D and 2D spherical implosion test problems.

The single and multi-material advection methods developed for the adaptive multi-material ALE algorithm were described in chapter 6. The single material advection employs van Leer's second order advection method, implemented using an isotropic approach on 2D unstructured non-orthogonal grids, using volume co-ordinates as proposed by Benson [9]. The main novel aspect of the single material advection method here is the generalisation of the YAQUI [2] for momentum advection method, to make it applicable to arbitrarily connected nodes, which may have any number of immediate neighbours. A new multi-material advection method has also been developed by the author for non-orthogonal grids. This scheme is based on the improved SLIC method developed by Tipton at LLNL for CALE. The author had to modify the CALE scheme to make it compatible with the isotropic advection used in CORVUS. Further improvements were then made to the scheme to improve it's tracking of isolated thin layers and to remove the noise the scheme generated on long thin aspect ratio zones in convergent flow problems. The Lagrangian test problems which were presented in chapters 3 and 5 have also been repeated using the ALE capability, and are presented in chapter 6 for comparison. These ALE calculations are in good agreement with the Lagrangian results, giving confidence in the accuracy and robustness of the new method.

The main difficulty in the Lagrangian treatment of multi-material cells is the lack of information about the velocity distribution within the cell. This means a subcell model must be assumed in order to determine the thermodynamic states of the material components and the forces such zones generate at the nodes. An equal volumetric strain treatment was initially applied to all the material components of each multi-material cell in CORVUS, and the average cell pressure was taken to be a volume fraction weighted average of the individual material pressures, as described in chapter 3. However, this is clearly unphysical, as at a gas metal interface the metal must undergo the same compression as the gas in a multi-material cell. This treatment can lead to code robustness problems, material components entering non-physical (P,V) states and does not provide a suitable platform for building in additional interface physics such as slip and void closure. In chapter 7 the Lagrangian treatment of multi-material cells was revisited, and a novel new scheme was developed, which attempts to emulate the behaviour of separate Lagrangian subzones, allowing the volume fractions of a multi-material cells to vary during the Lagrangian step in order to move the material components towards pressure equilibrium.

In the predictor phase of the Lagrangian step, the new scheme applies a simple wave interaction model at each material interface within each multi-material cell. This is analogous to applying an approximate Riemann solver, and is used to calculate an interface velocity relative to the frame of reference of the cell from an approximation to one of the Rankine Hugoniot relations. This velocity is used in turn to calculate a volume fraction flux between the two materials. The volume fraction fluxes are then applied as volume changes. But the sum of these volume

changes for each multi-material cell is constrained to not exceed the total predicted cell volume change, and the volume changes are only allowed in one sense or direction at each time step. Compressibility factors are then evaluated from the fraction of the predicted cell volume change that has been given to each material. These new volume changes are then used to update the half step density and internal energy of each material component. It was also shown that more accurate states are obtained if a new multi-phase artificial viscosity is used. This is simply calculated as a separate artificial viscosity for each material component, using the components individual density and sound speed. New average cell pressures are then defined as compressibility weighted averages of the values for the individual material components to ensure the work performed in the momentum step is consistent with that in the corrector internal energy update. The final cell volume changes are then apportioned to the materials, according to the compressibility factors for each material.

A novel approach was also used to demonstrate the accuracy of the new method, where multi-material cells were introduced at the start of a calculation, which was then performed with pure Lagrangian mesh motion. The states of the individual material components were then compared with an analytical solution and other numerical solutions, employing the equal volumetric strain scheme and a pressure relaxation algorithm. The new method was in close agreement with the analytical solution and showed significant improvement over the other two numerical techniques. The method has also been extended to provide a consistent void closure treatment which has been benchmarked in a similar manner. This suggests that the new scheme should form a suitable framework as desired for the inclusion of additional interface physics.

In chapter 8, three examples of the validation and application of CORVUS were presented. In the first example, CORVUS is used to calculate a shock bubble interaction problem. The CORVUS calculations were compared against both the experimental data, and another hydrocode, VUCALM. This experiment is well suited to hydrocode validation in that it offers a series of images for a range of problem times which clearly show shock fronts and material interfaces. VUCALM is an interesting rival numerical technique to compare against as it is unstaggered, and employs a Riemann solver rather than an artificial viscosity method, as used in CORVUS. So it is interesting to compare the ability of the two codes to capture shocks. VUCALM is also a free Lagrange code, so it also represents a pure Lagrangian technique which can be applied to high deformation problems. As CORVUS must reconstruct material interface it is also interesting to compare the ability of the two codes to model material interfaces which under high deformation.

In modelling the experiment with CORVUS, a new mesh movement strategy was also devised. This introduced a simple procedure for temporal adaption, where the advection step is only called if the time step is dropping. This was found to be computationally very efficient for this problem as the cost of the Lagrangian step is about a third of that of the advection step. This approach also acts to constantly

build in Lagrangian character to the mesh. The solution obtained with this method showed good agreement with all the experiment images throughout the problem in terms of the position of the shock fronts and bubble interfaces. CORVUS and VUCALM both provided an equally good match to experiment. The two codes also provided comparable shock capture. In addition to providing validation of the adaptive multi-material ALE algorithm, these results also demonstrate that a staggered grid hydro method, combined with a monotonic artificial viscosity, can capture shocks with a similar level of accuracy to that achievable with second order Godunov methods.

The adaptive multi-material ALE algorithm should be well suited to projectile impact and penetration problems. A typical example of such a problem was also given in chapter 8, which involved the impact of a steel projectile with a tank of water with aluminium walls. This problem demonstrated how the weighted Winslow's mesh movement algorithm can be used to focus mesh resolution into a feature of interest, in this case the projectile. The calculations presented followed the trajectory of the projectile as it penetrated the outside wall, travelled through the tank, and exited through the back wall, demonstrating the robustness of the adaptive multi-material ALE algorithm for problems of this type.

One of the many uses of CORVUS at AWE is in the design and analysis of dynamic friction experiments. These experiments are required to validate the author's friction model, described in chapter 4, and to increase our understanding of the physics of dynamic friction. In chapter 8 some of the calculations that have been performed by the author to model these experiments were presented. These calculations clearly demonstrate the importance of the two interface treatments that the adaptive multi-material ALE algorithm provides. The experiments to date involve two metal plates under shock loading from a high explosive charge placed at one end. In order to allow different interface physics models to be assessed the CORVUS slide line treatment had to be used between the two metal plates. However, such calculations will not run to completion with a Lagrangian interface treatment for the explosive material interfaces, since the relatively untamped explosive products expand rapidly around the metal plates, producing severe interface deformation. The explosive interfaces can however be modelled robustly using the multi-material cell based interface reconstruction technique, available in CORVUS.

9.2 Possible Future Work

A brief review will now be given of possible future work on CORVUS. This will include priorities, and a brief update on the status of any work that has already commenced, and will be subdivided into further improvements to the adaptive multi-material ALE algorithm, and increased functionality.

It may be necessary for some future applications of CORVUS to develop a more

local or minimum intervention mesh adaption strategy. This may be important, for example, in radiation hydrodynamics problems. In such applications users typically employ meshes with large aspect ratio cells ($\approx 100:1$) to reduce run times. This will put far greater emphasis on the underlying Lagrangian step. A series of recent papers by Caramana at LANL [96, 97, 98, 99] describe a compatible, energy and symmetry preserving Lagrangian hydrodynamics algorithm which appears to offer some significant improvements over the current Lagrangian step used in CORVUS. Caramana's finite volume scheme could be used as a replacement for the current finite element Lagrangian step. Alternatively, it may be possible to modify the current fine element scheme to exploit some of Caramana's ideas.

Caramana's scheme is termed "compatible" since instead of updating the internal energy of an element using a PdV formulation, a compatible work update is performed. This updates the internal energy, using the nodal forces from the acceleration step and the distance moved by each node during the time step. This has two benefits it ensures that energy is conserved to round off for the Lagrangian step, which is not achieved with a PdV formulation, and it enables edge based artificial viscosities to be used. Both of these are obvious improvements, but the latter potentially offers greater benefits. The current scalar monotonic artificial viscosity used in CORVUS is very effective, but does not behave as well as desired for large aspect ratio zones. One route forward is to develop a tensor monotonic artificial viscosity. However, tensor viscosities often introduce additional problems, especially at centres of convergence on non-orthogonal grids. The monotonic edge viscosity proposed by Caramana should avoid these difficulties, while still improving the behaviour of large aspect ratio zones.

The current slide line algorithm in CORVUS could also be improved in a number of ways. An equal voracity treatment which does not distinguish between master and slave surface may offer improved accuracy. A simple way to achieve this may be to alternate which surface is taken as the master at each time step. However, the author has not noticed any significant problems which stem from the current approach. The most significant deficiencies of the current slide treatment in practice appear to lie in the void closure treatment. The current approach is not robust when two surfaces gradually close, and a jet forms between the two surfaces. This generally leads to poorly defined unit vector normals for the slave surface, and errors in the positions of slave nodes, as they are put back onto the master surface, this in turn may result in mesh tangling and time step collapse. This problem may be solved by using the free surface trajectory of the node to calculate the point of impact, rather than its unit vector normal. A more significant problem that has been noted in some applications is a strong sensitivity to the choice of master and slave and their respective mesh resolutions in void closure problems. This is believed to be due to the current method for correcting a node's velocity after impact, and the internal energy fix-up introduced to conserve total energy after the impact, described in chapter 4.

In the author's opinion, something closer to an equal voracity treatment is required for void closure. The current method simply puts the slave nodes back in contact with the master surface, and then interpolates a new velocity for the slave node from adjacent master nodes. If kinetic energy has been lost, this is then added back in as internal energy. A better approach may be to conserve momentum in the normal direction between each pair of real and pseudo nodes, which collide, and so define new normal velocities for each real node after it's collision. This should minimise the loss or gain in kinetic energy. Since pressure continuity should be enforced at an impact, the lost or gained internal energy could be distributed amongst the two materials in such a way as to move towards pressure equilibrium.

The remaining deficiency with the current slide algorithms in CORVUS is that the void closure algorithm is currently incompatible with the friction model. This is important as real engineering problem inevitably have tolerance gaps, and gaps may open during an experiment, as discussed in chapter 8. This may cloud the interpretation of the influence of friction, both in applications and dynamic friction experiments, so it is of relatively high priority that this deficiency is corrected.

The new Lagrangian multi-material cell scheme that has been developed for CORVUS was a significant improvement. However, this work could also be taken further. The next step should be to extend the method to make it fully compatible with material strength. One possible way to achieve this has already been discussed in chapter 7. If this is achieved then the great challenge is to extend the method further to provide multi-material slide. A multi-material cell scheme that includes all the interface physics that can be introduced through slide algorithms would be a very robust technique, and would allow the importance of slide and friction to be assessed for the first time for interfaces undergoing high deformation. In addition, it may also be possible to extend the multi-material void closure scheme, to also treat impact problems where a gas initially separating the impactor and the target is important, and must be modelled explicitly, rather than treated as an idealised void.

As discussed at the beginning of this section, it may be possible to further improve the accuracy of the adaptive multi-material ALE algorithm by developing more local mesh movement algorithms. However, the risk of such an approach is that while gains may be achieved for some problems, it may make the code less robust. This is why a constrained global mesh movement strategy is currently used. However, it may be possible to further refine the adaptive in time mesh movement algorithm, that was demonstrated for the shock bubble interaction problem in chapter 8. This approach employs a minimum intervention strategy, and attempts to run with as close to Lagrangian mesh motion as possible, but when it does adapt it applies a global mesh movent algorithm. Possibly the simple time step based criterian could be improved and additional criteria devised to define when a Lagrangian interface should be relaxed. It has also been noted by the author that Winslow's algorithm does sometimes move a node in the wrong direction, particularly in the vicinity of Lagrangian boundaries with high local curvature. It may be possible to

modify Winslow's algorithm to detect these situations and apply a different strategy in such circumstances.

There are also a number of improvements that could be made both to the single and multi-material advection algorithms. Corner coupling errors introduced by the isotropic advection method used can be seen in idealised test problems, but do not appear to have any significant effect on real applications. These errors could be reduced significantly however, by either using an advection method based on Collela's corner transport upwind (CTU) algorithm, or simply calculating how much of each overlap volume actually lies in the donor cells diagonal neighbours, rather than their main neighbours, and subdividing the advection fluxes appropriately. However, possibly the most important deficiency of the advection methods used in the adaptive multi-material ALE algorithm is the loss in kinetic energy, which is incurred when momentum and internal energy are advected. This loss in kinetic energy could be calculated and added back in as additional internal energy, as is done in a number of codes. However, in the author's opinion this is just trading one error for another. The author has also found a significant sensitivity in some applications where this approach is taken. A better approach may be to try to minimise the energy error, rather than fix it up after the error has been introduced. This may be achieved by comparing the mass weighted van Leer advection of velocity (momentum) and the square of the velocity (kinetic energy). This may imply that the error would be reduced if a higher order initial value distribution was assumed for the momentum advection, such as in the piecewise parabolic method.

The multi-material advection method used again performs well in real applications, but could be further improved. An explicit treatment of T-junctions would certainly be useful for some problems. In addition, it would also be interesting to implement Youngs' interface reconstruction method [87], and assess whether it offers any significant advantages over the current CORVUS treatment for applications. An interesting research topic which would complement the work done here by the author in improving the Lagrangian treatment of multi-material cells, would be to attempt to take the multi-material advection above first order. In practical terms the main deficiency that has been noted with the current method is with large aspect ratio zones where the interface sometimes still appears a little noisy. This may be due to the switch over from serial to parallel advection. Possibly there could be some merit in smoothing this transition.

In chapter 8 the gas bubble was painted on top of the mesh at the start of the problem using a dedicated subroutine. This allowed a more ideal initial mesh to be generated and higher quality meshes to be generated during the problem by the mesh movement. This is quite a powerful approach would could be exploited in many problems, by developing a general ALE generator that would allow the problem geometry to be separated from the mesh generation. This would greatly simplify the set-up of many complex engineering problems, as well as significantly improving both the level of engineering detail that can realistically be represented, and the

mesh quality that can be achieved in many problems. In terms of the usability of CORVUS this is a high priority.

The highest priority at present is to reduce the run time for large, high resolution, calculations on CORVUS. Two routes are being pursued to achieve this: distributed parallelism and the development of an adaptive mesh refinement (AMR) capability. The main challenge in parallelising CORVUS is to achieve scalability, given the more limited amount of work available in a 2D code, and the local nature of the algorithms used in CORVUS. The Lagrangian step has been parallelised, and work is now focussed on parallelising the advection step.

In the author's opinion, the best way to introduce time dependent mesh adaption into a hydrocode may be through the development of a hybrid ALE and AMR capability. The less expensive ALE capability principally would be used to maintain mesh quality and the other benefits that have already been discussed. The AMR capability would provide additional resolution for the physics and features of interest such as shock, detonation waves and material interfaces. Given the ALE framework, the refinement could be anisotropic in nature to improve its efficiency. Although a number of Eulerian AMR codes have been developed, the author is aware of only one other ALE/AMR code which is under development at LLNL [100], and this has still to address the challenges introduced by the presence of slide lines and multi-material interfaces. The coding of a prototype version of CORVUS which combines its current ALE capability with AMR has just begun. Whilst it is unlikely that such an AMR capability could compete with parallelisation, in terms of the reduction in run times that can be achieved for most pure hydro problems, it may offer greater benefits for problems including additional physics which must be solved using implicit methods, that do not offer the same parallel scalability as explicit hydrodynamics.

Many of the important engineering and physics problems to which hydrocodes are applied are 3D in nature. The adaptive multi-material ALE algorithm currently is being implemented in a 3D code at AWE known as PEGASUS.

Bibliography

[1] C. E. Anderson. An overview of the theory of hydrocodes. *Internat. J. Impact Engng.*, 5:33–59, 1987.

[2] A. A. Ammsden and C. W. Hirt. YAQUI: An Arbitrary Lagrangian-Eulerian computer program for fluid flow at all speeds. Technical Report LA-5100, Los Alamos National Laboratory, 1973.

[3] C. W. Hirt A. A. Ammsden and J. L. Cook. An Arbitrary Lagrangian-Eulerian Computing Method for all flow speeds. *Journal of Computational Physics*, 14:227–253, 1974.

[4] A. A. Ammsden H. M. Ruppel and C. W. Hirt. SALE: A simplified ALE computer code for fluid flow at all speeds. Technical report, Los Alamos National Laboratory, 1980.

[5] A. A. Ammsden and H. M. Ruppel. SALE-3D: A simplified ALE computer code for calculating three-dimensional fluid flow. Technical report, Los Alamos National Laboratory, 1981.

[6] R. W. Sharp and R. T. Barton. HEMP advection model. Technical Report UCID-17809, Rev. 1, Lawrence Livermore National Laboratory, 1981.

[7] R. B. Demuth L. G. Margolin B. D. Nichols T. F. Adams and B. W. Smith. SHALE: A computer code for solid dynamics. Technical Report LA-10236, Los Alamos National Laboratory, 1985.

[8] F. L. Addessio D. E. Carroll J. K. Dukowicz F. H. Harlow J. N. Johnson B. A. Kaskiwa M. E. Maltrud and H. M. Ruppel. CAVEAT: A computer code for fluid dynamics problems with large distortion and internal slip. Technical Report UC-32, Los Alamos National Laboratory, 1988.

[9] D. J. Benson. An Efficient, Accurate, Simple ALE Method for Nonlinear Finite Element Programs. *Comp. Meth. App. Mech. Engng., 72*, pages 305–350, 1989.

[10] W. F. Noh and P. Woodward. SLIC (Simple Line Interface Calculation). *Lecture Notes in Physics*, 59, 1976.

[11] J. von Neumann and R. D. Richtmeyer. A Method for the Numerical Calculations of Hydrodynamical Shocks. *Journal of Applied Physics*, 21, 1950.

[12] R. Landshoff. A Numerical Method for Treating Fluid Flow in the Presence of Shocks. Technical Report LA-1930, Los Alamos Scientific Laboratory, 1955.

[13] G. Maenchen and S. Sack. The TENSOR Code. *Methods in Computational Physics Volume 3*, pages 181–210. Academic Press, New York, 1964.

[14] M. L. Wilkins. Use of artificial viscosity in multidimensional fluid dynamics calculations. *Journal of Computational Physics, 36*, pages 381–403, 1980.

[15] W. F. Noh. Errors for Calculations of Strong Shocks Using an Artificial Viscosity and an Artificial Heat Flux. *Journal of Computational Physics, 72*, pages 78–120, 1987.

[16] R. B. Christensen. Private Communication. 1990.

[17] D. J. Benson. A new two-dimensional flux-limited shock viscosity. *Comp. Meth. App. Mech. Engng., 93*, pages 39–95, 1991.

[18] D. J. Benson. Computational methods in Lagrangian and Eulerian hydrocodes. *Comp. Meth. App. Mech. Engng., 99*, pages 235–394, 1992.

[19] B. Tipton. Private Communication. 1994.

[20] J. P. Boris and D. L. Book. Flux-corrected transport, i. SHASTA, a fluid transport algorithm that works. *Journal of Computational Physics*, 11:38–69, 1973.

[21] S. K. Godunov. A difference scheme for numerical computation of discontinuous solutions of equations of fluid dynamics. *Mat. Sb.*, 47:271–290, 1959.

[22] B. P. Howell. *An Investigation of Lagrangian Riemann Methods Incorporating Material Strength.* PhD thesis, Southampton University, 2001.

[23] P. D. Palma P. Roe R. Struijs, H. Deconinck and K. Powell. Progress on multidimensional upwind euler solvers for unstructured grids. *AIAA-91-1550*, 1991.

[24] M. J. Marchant and N. P. Weatherill. Adaptivity Techniques for Compressible Inviscid Flows. *Computer Methods in Applied Mechanics and Engineering*, 106:83–106, 1993.

[25] P. Collela and M. J. Berger. Local adaptive mesh refinement for shock hydrodynamics. *Journal of Computational Physics, 82*, pages 64–84, 1989.

[26] M. J. Berger. Data Structures for Adaptive Grid Generation. *SIAM J. Sci. Stat. Comput. Vol. 7, No. 3*, 1986.

[27] J. Bell M. J. Berger J. Saltzman and M. Welcome. Three-Dimensional Adaptive Mesh Refinement for Hyperbolic Conservation Laws. *SIAM J. Sci. Comput. Vol. 15, No. 1*, 1994.

[28] J. J. Quirk. *An Adaptive Grid Algorithm for Computational Shock Hydrodynamics.* PhD thesis, Cranfield University, 1991.

[29] J. D. Whittle. Private Communication. 1998.

[30] E. Hinton and D. R. Owen. *An Introduction to Finite Element Computation.* Pineridge Press, 1979.

[31] O. C. Zienkiewicz and R. L. Taylor. *The Finite Element Method, fourth edition, volumes 1 and 2.* McGraw Hill, 1991.

[32] R. Sedgewick. *Algorithms.* Addison-Wesley Publishing Company, 1988.

[33] J. O. Hallquist. DYNA 3D Course Notes, Lawrence Livermore National Laboratory. Technical Report UCID-19899, Rev. 2, Lawrence Livermore National Laboratory, 1987.

[34] D. L. Hicks. Stability Analysis of WONDY for a Special Case of Maxwell's Law. *Mech. Comp. 32*, page 1123, 1978.

[35] R. L. Bowers and J. R. Wilson. *Numerical Modelling in Applied Physics and Astrophysics.* Jones and Bartlett Publishers, 1991.

[36] D. P. Flanagan and T. Belytschko. A uniform strain hexhedron and quadrilateral and orthogonal hourglass control. *Comp. Meth. App. Mech. Engng., 17*, pages 679–706, 1981.

[37] L. G. Margolin and J. J. Pyun. A method for treating hourglass patterns. In *Proceedings of the 4th International Conference on Numerical Methods in Laminar and Turbulent Flow, Montreal*, 1987.

[38] M. L. Wilkins. Calculation of Elastic-Plastic Flow. *Methods in Computational Physics Volume 3*, pages 211–263. Academic Press, New York, 1964.

[39] S. Hancock. PISCES 2DELK Theoretical Manual. Technical report, (Pysics International), 1985.

[40] G. A. Sod. A Survey of Several Different Methods for Systems of Nonlinear Hyperbolic Conservation Laws. *Journal of Computational Physics, 27*, pages 1–31, 1978.

[41] A. J. Barlow. MSc Thesis - Feasibility Study for Code to Model Hypervelocity Launcher. Technical report, Cranfield University, 1999.

[42] D. J. Steinberg S. G. Cochrran and M. W. Guinan. A Constitutrive Model for Metals Applicable at High Strain Rates. *Journal of Applied Physics*, 51:1498–1504, 1980.

[43] F. L. Addessio D. E. Carroll J. K. Dukowicz F. H. Harlow J. N. Johnson B. A. Kaskiwa M. E. Maltrud and H. M. Ruppel. CAVEAT: A computer code for fluid dynamics problems with large distortion and internal slip. Technical Report LA-10613-MS, Los Alamos National Laboratory, 1986.

[44] D. Verney. Evaluation de la Limite Elastique du Cuivre et de l'Uranium par des Experiences d'Implosion 'Lentre. In *Behaviour of Dense Media Under High Dynamic Pressures, Symposium HDP Paris 1967*. Gordon and Breach, Paris and New York, 1968.

[45] I. N. Gray. Private Communication.

[46] C. W. Hirt and B. D. Nichols. Volume of fluid VOF method for the dynamics of free boundaries. *Journal of Computational Physics*, 39:201–225, 1981.

[47] J. E. Hammerberg B. L. Holian J. Röder A. R. Bishop and S. J. Zhou. Nonlinear dynamics and the problem of slip at material interfaces. *Physica D*, 123:330–340, 1998.

[48] A. J. Barlow. Friction in corvus a 2D ALE code. In *Proceedings of the 22nd International Symposium on Shock Waves, Imperial College, London*, pages 653–657, 1999.

[49] A. B. Chaudhary and K. J. Bathe. A Lagrange multiplier segment procedure for solution of three dimensional contact algorithms. Technical Report BRL-CR-544, U.S. Ballistic Research Laboratory, 1985.

[50] N. Kikuchi and J. T. Oden. Contact problems in elasticity: A study of variational inequalities and finite element methods for a class of contact problems in elasticity. *SIAM Stud.*, 8, 1986.

[51] A. J. Barlow and J. Whittle. Mesh Adaptivity and Material Interface Algorithms. *Chem. Phys. Reports*, 19(2):233–258, 2000.

[52] R. E. Winter. P. Taylor. D. J. Carley A. J. Barlow and H. Pragnell. Strain Distribution at High Pressure, High Velocity Sliding Interfaces. In *Proceedings of the American Physical Society meeting on materials under high strain rate, Atlanta*, 2001.

[53] W. P. Crowley. An Equipotential Zoner on a Quadrilateral Mesh. Technical Report Memorandum, Lawrence Livermore National Laboratory, 1962.

[54] A. M. Winslow. Equipotential zoning of two-dimensional meshes. Technical Report UCRL-7312, Lawrence Livermore National Laboratory, 1963.

[55] A. M. Winslow. Numerical Solution of the Quasilinear Poisson Computing Equation in a Nonuniform Triangular Mesh. *Journal of Computational Physics*, 2:149–172, 1967.

[56] B. Tipton. Private Communication. 1992.

[57] J. U. Brackbill and J. S. Saltzman. Adaptive Zoning for Singular Problems in Two Dimensions. *Journal of Computational Physics*, 46:342–368, 1982.

[58] A. M. Winslow. Adaptive-Mesh zoning by the Equipotential Methos. Technical Report UCID-19062, Lawrence Livermore National Laboratory, 1981.

[59] A. J. Barlow. Mesh Adaptivity and Material Interface Algorithms in a Two Dimensional Lagrangian Hydrocode. In *Proceedings of the TTCP Hydrocode Workshop, Banff, Canada.*, 1996.

[60] J. K. Dukowicz. Conservative Rezoning (remapping) for General Quadrilateral Meshes. *Journal of Computational Physics, 54*, pages 411–424, 1984.

[61] J. K. Dukowicz and W. Kodis. Accurate conservative remapping (rezoning) for arbitary Lagrangian-Eulerian computation. *SIAM J. Sci. Stat. Comput. Vol. 8, No. 3*, 1987.

[62] J. D. Ramshaw. Conservative Rezoning (remapping) for General Quadrilateral Meshes. *Journal of Computational Physics, 59*, pages 193–199, 1985.

[63] J. D. Ramshaw. Simplified second-Order rezoning algorithm for generalised two-dimensional meshes. *Journal of Computational Physics, 67*, pages 214–222, 1986.

[64] B. van Leer. Towards the Ultimate Conservative Difference Scheme. I.The Quest for Monotonicity. *Lecture Notes in Physics Physics, 18*, pages 163–168, 1973.

[65] B. van Leer. Towards the Ultimate Conservative Difference Scheme. II.Monotonicity and Conservation Combined in a Second Order Scheme. *Journal of Computational Physics, 14*, pages 263–275, 1974.

[66] B. van Leer. Towards the Ultimate Conservative Difference Scheme. III. Upstream-Centred Finite Difference Schemes for Ideal Compressible Flow. *Journal of Computational Physics, 23*, pages 263–275, 1977.

[67] B. van Leer. Towards the Ultimate Conservative Difference Scheme. IV. A New Approach to Numerical Convection. *Journal of Computational Physics, 23*, pages 276–299, 1977.

[68] B. van Leer. Towards the Ultimate Conservative Difference Scheme. V. A Second-Order Sequel to Godunov's Method. *Journal of Computational Physics, 32*, pages 101–136, 1979.

[69] W. G. Strang. On the construction and comparison of difference schemes. *SIAM J. Numer. Anal.*, 5(3):506–517, 1968.

[70] R. Courant E. Isaacson and M. Rees. On the solution of nonlinear hyperbolic differential equations by finite differences. *Comm. Pure. Appl. Math.*, 5:243, 1952.

[71] P. Collela. Multidimensional upwind methods for hyperbolic conservation laws. *Journal of Computational Physics, 87*, pages 171–209, 1990.

[72] D. J. Benson. Momentum Advection on a Staggered Mesh. *Journal of Computational Physics, 100*, pages 1–20, 1992.

[73] R. B. DeBar. Fundamentals of the KRAKEN Code. Technical Report UCIR-760, Lawrence Livermore National Laboratory, 1974.

[74] J. M. Hyman. Numerical methods for tracking interfaces. *Physics D, 12*, pages 396–407, 1984.

[75] E. S. Oran and J. P. Boris. *Numerical Simulation of Reactive Flow*. Elsevier, New York, 1987.

[76] S. A. Silling. CTH Reference Manual: Boundary Layer Algorithm for Sliding Interfaces in Two Dimensions. Technical Report SAND93-2487, Sandia National Laboratory, 1994.

[77] D. L. Littlefield. An Algorithm for modelling Contact in Three Dimensional Eulerian Hydrocodes. In *Proceedings of the American Physical Society meeting on materials under high strain rate, Snow Bird, Utah*, 1999.

[78] W. E. Johnson. TOIL (A two-material version of the OIL code). Technical Report General Atomic Report GAMD-8073, 1967.

[79] S. L. Thompson. CSQ - A two dimensional hydrodynamic program with energy flow and material strength. Technical Report SAND74-0122, Sandia National Laboratory, 1975.

[80] R. Couch E. Albright and N. Alexander. The JOY Computer Code. Technical Report UCID-19688, Lawrence Livermore National Laboratory, 1983.

[81] W. G. Sutcliffe. BBC Hydrodynamics. Technical Report UCIR-716, Lawrence Livermore National Laboratory, 1973.

[82] W. F. Noh and P. Woodward. SLIC (Simple Line Interface Calculation). Technical Report UCRL-52111, Lawrence Livermore National Laboratory, 1976.

[83] A. J. Chorin. Flame advection and propagation algorithms. *Journal of Computational Physics, 35*, pages 1–11, 1980.

[84] W. H. McMaster. Computer Codes for fluid-structure interactions, in proc. 1984 pressure vessel and piping conference, san antonio, tx. Technical Report UCRL-89724 Preprint, Lawrence Livermore National Laboratory, 1984.

[85] K. S. Holian D. A. Mandell T. F. Adams F. L. Addessio J. R. Baumgardner and S. J. Mosso. MESA: A 3-D computer code for armor/anti-armor applications,. In *Proceedings Supercomputing World Conference, San Francisco, CA,*, 1989.

[86] J. M. McGlaun S. L. Thompson and M. G. Elrick. CTH: A three-dimensional shock wave physics code. In *Proceedings 1989 Hypervelocity Impact Symposium*, 1989.

[87] D. L. Youngs. Time-Dependent Multi-Material Flow with Large Fluid Distortion. In K. W. Morton and eds. M. J. Baines, editors, *Numerical Methods for Fluid Dynamics*, pages 273–285. (Academic Press, London), 1982.

[88] D. L. Youngs. An interface tracking method for a 3D Eulerian hydrodynamics code. Technical Report AWRE/44/92/35, AWRE Design Mathematics Division, 1987.

[89] N. J. Johnson. Private Communication. 1994.

[90] J. K. Dukowicz. A General, Non-iterative Riemann Solver for Godunov's Method. *Journal of Computational Physics, 61*, pages 119–137, 1985.

[91] Ya. B. Zel'dovich and Yu. P. Raizer. *Physics of Shock Waves and High-Temperature Hydrodynamic Phenomena.* New York: Academic Press, 1966.

[92] P. Sims. *Interface Tracking using Lagrangian-Eulerian methods.* PhD thesis, Reading University, 1999.

[93] J.-F. Haas and B. Sturtevant. Interaction of weak shock waves with cylindrical and spherical gas inhomogeneities. *J. Fluid Mech*, 181:41–76, 1987.

[94] G. J. Ball. A Free-Lagrange method for unsteady compressible flow:simulation of a confined cylindrical blast wave. *Shock Waves*, 5:311–325, 1996.

[95] G. J. Ball and A. J. Barlow. ALE and Free-Lagrange Simulations of Shock/Bubble Interaction. In *Proceedings of the 23nd International Symposium on Shock Waves, Austin, Texas*, 2001.

[96] M. J. Shashkov E. J. Caramana, D. E. Burton and P. P. Whalen. The Construction of Compatible Hydrodynamics Algorithms Utilizing Conservation of Total Energy. *Journal of Computational Physics*, 146:227–262, 1998.

[97] E. J. Caramana and P. P. Whalen. Numerical Preservation of Symmetry Properties Continuuum Problems. *Journal of Computational Physics*, 142:174–198, 1998.

[98] M. J. Shashkov E. J. Caramana and P. P. Whalen. Elimination of Artificial Grid Distortion and Hourglass-Type Motions by Means of Lagrangian Subzonal Masses and Pressures. *Journal of Computational Physics*, 144:521–561, 1998.

[99] M. J. Shashkov E. J. Caramana and P. P. Whalen. Formulation of Artificial Viscosity for Multi-Dimensional Shock Wave Computations. *Journal of Computational Physics*, 144:70–97, 1998.

[100] R. W. Anderson R. B. Pember and N. S. Elliot. An Aribtrary Lagrangian-Eulerian Method with Local Structured Adaptive Mesh Refinement for Modelling Shock Hydrodynamics. In *Proceedings of the 40th AIAA Aerospace Sciences Meeting and Exhibit, Reno, Nevada*, 2002.

Appendix A

Evaluation of Shape function integrals

In chapter 3, weak integral forms of the conservation equations were reduced to simple forms, involving the integral of the shape functions and their derivatives over a square in a local coordinate space. The isoparametric space will now be derived for each element, and used to derive the various terms required in chapter 3, by exact integration. A more complete explanation can however be found in [30].

Recall the shape functions (3.35) expressed in terms of the local coordinate system.

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

$$N_2 = \frac{1}{4}(1+\xi)(1-\eta) \tag{A.2}$$

$$N_3 = \frac{1}{4}(1+\xi)(1+\eta) \tag{A.3}$$

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

An isoparametric formulation is used for the Lagrangian step in CORVUS, which means that both the geometry of the element and the functions to be integrated are represented in terms of discrete nodal values and these same shape functions. Whilst the area of an element is given by

$$dxdy = detJd\xi d\eta \tag{A.5}$$

where the Jacobian matrix **J** is defined as

$$\mathbf{J} = \begin{bmatrix} \frac{\partial x}{\partial \xi} & \frac{\partial y}{\partial \xi} \\ \frac{\partial x}{\partial \eta} & \frac{\partial y}{\partial \eta} \end{bmatrix} \tag{A.6}$$

$$= \begin{bmatrix} \sum_i \frac{\partial N_i}{\partial \xi} . x_i & \sum_i \frac{\partial N_i}{\partial \xi} . y_i \\ \sum_i \frac{\partial N_i}{\partial \eta} . x_i & \sum_i \frac{\partial N_i}{\partial \eta} . y_i \end{bmatrix} \tag{A.7}$$

and the derivatives of the shape functions are given by

$$\frac{\partial N_1}{\partial \xi} = \frac{1}{4}(\eta - 1) \tag{A.8}$$

$$\frac{\partial N_1}{\partial \eta} = \frac{1}{4}(\xi - 1) \tag{A.9}$$

...etc

Thus the x,y coordinates at any point ξ, η within an element can be obtained from

$$x(\xi, \eta) = \sum_i N_i x_i$$

$$y(\xi, \eta) = \sum_i N_i y_i \tag{A.10}$$

which can be expanded as

$$x(\xi, \eta) = \frac{1}{4}((x_1 + x_2 + x_3 + x_4) + \xi(-x_1 + x_2 + x_3 - x_4)$$

$$+ \eta(-x_1 - x_2 + x_3 + x_4) + \xi\eta(x_1 - x_2 + x_3 - x_4)) \tag{A.11}$$

and similarly for $y(\xi, \eta)$.

Using the chain rule of differentiation

$$\frac{\partial N_j}{\partial x} = \frac{\partial N_j}{\partial \xi} . \frac{\partial \xi}{\partial x} + \frac{\partial N_j}{\partial \eta} . \frac{\partial \eta}{\partial x} \tag{A.12}$$

and obtaining the partial derivatives of ξ and η from the identity

$$\mathbf{J^{-1}} = \begin{bmatrix} \frac{\partial \xi}{\partial x} & \frac{\partial \eta}{\partial x} \\ \frac{\partial \xi}{\partial y} & \frac{\partial \eta}{\partial y} \end{bmatrix} \tag{A.13}$$

$$= \frac{1}{det\mathbf{J}} \begin{bmatrix} \frac{\partial y}{\partial \eta} & -\frac{\partial x}{\partial \xi} \\ -\frac{\partial x}{\partial \eta} & \frac{\partial x}{\partial \xi} \end{bmatrix} \tag{A.14}$$

We can then write:

$$\frac{\partial x}{\partial \xi} = \sum_{i=1}^{4} \frac{\partial N_i}{\partial \xi} x_i$$

$$= \frac{1}{4}(-x_1 + x_2 + x_3 - x_4) + \frac{\eta}{4}(x_1 - x_2 + x_3 - x_4)$$

$$= A_1 + A_2\eta \tag{A.15}$$

Similarly:

$$\frac{\partial x}{\partial \eta} = \frac{1}{4}(-x_1 - x_2 + x_3 + x_4) + \frac{\xi}{4}(x_1 - x_2 + x_3 - x_4)$$

$$= A_3 + A_2\xi \tag{A.16}$$

$$\frac{\partial y}{\partial \xi} = \frac{1}{4}(-y_1 + y_2 + y_3 - y_4) + \frac{\eta}{4}(y_1 - y_2 + y_3 - y_4)$$

$$= B_1 + B_2\eta \tag{A.17}$$

$$\frac{\partial y}{\partial \eta} = \frac{1}{4}(-y_1 - y_2 + y_3 + y_4) + \frac{\xi}{4}(y_1 - y_2 + y_3 - y_4)$$

$$= B_3 + B_2\xi \tag{A.18}$$

The Jacobian matrix (A.7) and its inverse (A.14) can now be formed to give

$$\mathbf{J} = \begin{bmatrix} A_1 + A_2\eta & B_1 + B_2\eta \\ A_3 + A_2\xi & B_3 + B_2\xi \end{bmatrix} \tag{A.19}$$

$$\mathbf{J}^{-1} = \frac{\mathbf{1}}{\mathbf{C}} \begin{bmatrix} B_3 + B_2\xi & -(B_1 + B_2\eta) \\ -(A_3 + A_2\xi) & A_1 + A_2\eta \end{bmatrix} \tag{A.20}$$

where

$$C = det\mathbf{J} = [(\mathbf{A_1} + \mathbf{A_2}\eta)(\mathbf{B_3} + \mathbf{B_2}\xi) - (\mathbf{B_1} + \mathbf{B_2}\xi)(\mathbf{A_3} + \mathbf{A_2}\eta)] \tag{A.21}$$

The partial derivatives of the shape functions in the global coordinate space can then

be written totally in terms of the local coordinate system as

$$\frac{\partial N_1}{\partial x} = \frac{1}{4C}(\eta - 1)(B_3 + B_2\xi) - \frac{1}{4C}(\xi - 1)(B_1 + B_2\eta) \tag{A.22}$$

$$\frac{\partial N_1}{\partial y} = -\frac{1}{4C}(\eta - 1)(A_3 + A_2\xi) + \frac{1}{4C}(\xi - 1)(A_1 + A_2\eta) \tag{A.23}$$

...etc

The integrals that were obtained from the analysis of the weak forms of the conservation laws in chapter 3 can then be evaluated.

$$\begin{aligned}\{\int N_1\} &= \int_{-1}^{1}\int_{-1}^{1} N_1 det\mathbf{J}d\xi d\eta \\ &= \frac{1}{4}\int_{-1}^{1}(1-\xi)(B_3 + B_2\xi)d\xi \int_{-1}^{1}(1-\eta)(A_1 + A_2\eta)d\eta \\ &- \frac{1}{4}\int_{-1}^{1}(1-\xi)(A_3 + A_2\xi)d\xi \int_{-1}^{1}(1-\eta)(B_1 + B_2\eta)d\eta \\ &= \frac{1}{9}[(3B_3 - B_2)(3A_1 - A_2) - (3A_3 - A_2)(3B_1 - B_2)]\end{aligned} \tag{A.24}$$

and similarly

$$\{\int N_2\} = \frac{1}{9}[(3B_3 + B_2)(3A_1 - A_2) - (3A_3 + A_2)(3B_1 - B_2)] \tag{A.25}$$

$$\{\int N_3\} = \frac{1}{9}[(3B_3 + B_2)(3A_1 + A_2) - (3A_3 + A_2)(3B_1 + B_2)] \tag{A.26}$$

$$\{\int N_4\} = \frac{1}{9}[(3B_3 - B_2)(3A_1 + A_2) - (3A_3 - A_2)(3B_1 + B_2)] \tag{A.27}$$

The partial derivative terms are integrated in the same way, and reduce to very neat expressions in A and B.

$$\begin{aligned}\{\int \frac{\partial N_1}{\partial x}\} &= \int_{-1}^{1}\int_{-1}^{1} \frac{1}{4det\mathbf{J}}[(\eta - 1)(B_3 + B_2\xi) - (\xi - 1)(B_1 + B_2\eta)]det\mathbf{J}d\xi d\eta \\ &= -B_3 + B_1\end{aligned} \tag{A.28}$$

and similarly,

$$\{\int \frac{\partial N_1}{\partial y}\} = A_3 - A_1 \tag{A.29}$$

$$\{\int \frac{\partial N_2}{\partial x}\} = B_3 + B_1 \tag{A.30}$$

$$\{\int \frac{\partial N_2}{\partial y}\} = -A_3 - A_1 \tag{A.31}$$

$$\{\int \frac{\partial N_3}{\partial x}\} = B_3 - B_1 \tag{A.32}$$

$$\{\int \frac{\partial N_3}{\partial y}\} = -A_3 + A_1 \tag{A.33}$$

$$\{\int \frac{\partial N_4}{\partial x}\} = -B_3 - B_1 \tag{A.34}$$

$$\{\int \frac{\partial N_4}{\partial y}\} = A_3 + A_1 \tag{A.35}$$