PRACTICAL METHODS FOR SIMULATION OF COMPRESSIBLE FLOW AND STRUCTURE INTERACTIONS

A DISSERTATION
SUBMITTED TO THE DEPARTMENT OF COMPUTER SCIENCE
AND THE COMMITTEE ON GRADUATE STUDIES
OF STANFORD UNIVERSITY
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

Nipun Kwatra
April 2011

I certify that I have read this dissertation and that, in my opinion, it is fully adequate in scope and quality as a dissertation for the degree of Doctor of Philosophy.

(Ronald Fedkiw) Principal Adviser

I certify that I have read this dissertation and that, in my opinion, it is fully adequate in scope and quality as a dissertation for the degree of Doctor of Philosophy.

(Charbel Farhat)

I certify that I have read this dissertation and that, in my opinion, it is fully adequate in scope and quality as a dissertation for the degree of Doctor of Philosophy.

(Vladlen Koltun)

Approved for the University Committee on Graduate Studies

Abstract

This thesis presents a semi-implicit method for simulating inviscid compressible flow and its extensions for strong implicit coupling of compressible flow with Lagrangian solids, and artificial transition of fluid from compressible flow to incompressible flow regime for graphics applications.

First we present a novel semi-implicit method for alleviating the stringent CFL condition imposed by the sound speed in simulating inviscid compressible flow with shocks, contacts and rarefactions. The method splits the compressible flow flux into two parts – an advection part and an acoustic part. The advection part is solved using an explicit scheme, while the acoustic part is solved using an implicit method allowing us to avoid the sound speed imposed CFL restriction. Our method leads to a standard Poisson equation similar to what one would solve for incompressible flow, but has an identity term more similar to a diffusion equation. In the limit as the sound speed goes to infinity, one obtains the Poisson equation for incompressible flow.

This implicit pressure solve also lends itself nicely to solve for the pressure and coupling forces at a solid fluid interface. With this pressure solve as the foundation, we then develop a novel method to implicitly two-way couple Eulerian compressible flow to volumetric Lagrangian solids. The method works for both deformable and rigid solids and for arbitrary equations of state. Similar to previous fluid-structure interaction methods, we apply pressure forces to the solid and enforce a velocity boundary condition on the fluid in order to satisfy a no-slip constraint. Unlike previous methods, however, we apply these coupled interactions implicitly by adding the constraint to the pressure system and combining it with any implicit solid forces in order to

obtain a strongly coupled system. Because our method handles the fluid-structure interactions implicitly, we avoid introducing any new time step restrictions and obtain stable results even for high density-to-mass ratios, where explicit methods struggle or fail. We exactly conserve momentum and kinetic energy (thermal fluid-structure interactions are not considered) at the fluid-structure interface, and hence naturally handle highly non-linear phenomenon such as shocks, contacts and rarefactions.

The implicit pressure solve allows our method to be used for any sound speed efficiently. In particular as the sound speed goes to infinity, we obtain the standard Poisson equation for incompressible flow. This allows our method to work seamlessly and efficiently as the sound speed in the underlying flow field changes. Building on this feature of our method, we next develop a practical approach to integrating shock wave dynamics into traditional smoke simulations. Previous methods for doing this either simplified away the compressible component of the flow and were unable to capture shock fronts or used a prohibitively expensive explicit method that limits the time step of the simulation long after the relevant shock waves and rarefactions have left the domain. Instead, using our semi-implicit formulation allows us to take time steps on the order of fluid velocity. As we handle the acoustic fluid effects implicitly, we can artificially drive the sound speed c of the fluid to ∞ without going unstable or driving the time step to zero. This permits the fluid to transition from compressible flow to the far more tractable incompressible flow regime once the interesting compressible flow phenomena (such as shocks) have left the domain of interest, and allows the use of state-of-the-art smoke simulation techniques. We also propose an extension to Euler's equations to model combustion of fuel in explosions.

In loving memory of my father, Dr. Suresh Kumar Kwatra, who has been my greatest inspiration in life

Acknowledgement

I am profoundly thankful to my parents for their immense support, encouragement, guidance, love and affection. My father has been a great academic influence on me and I am thankful to him for instilling in me the confidence and guiding me through the tough periods of my life. Undoubtedly I would not be at this place without his extraordinary hard work. My mother has been has been a constant source of motivation and support for me. Her constant push for excellence motivated me to dream of bigger goals. I will always to thankful to her for the enormous sacrifices she has made for me. I am also immensely thankful to my brother Vivek Kwatra. He has played a huge role in my academic life from when I was a child. He has been the role model I have tried to follow my entire life. Without the example he set for me, it would not have had been possible for me to get this far. He was a constant source of inspiration and was always there to guide and help me when I needed him. I also had the opportunity to collaborate with him on research and it was an amazing learning experience for me.

I would like to thank my adivsor, Ron Fedkiw, for his extraordinary support and guidance. Ron is undoubtedly the smartest person I have worked with, and I have been fortunate to learn a lot from him. His insight into topics in mathematics, physics, simulation; and research, problem solving in general will be very useful to me in the future. I am also grateful to Ron for his encouragement and the trust he showed in me, which gave me a lot of confidence. At the same time I am very thankful to him pointing out my weaknesses which I was able to then work on. He was a great advisor, always there to help when I was stuck, and at the same time pushing me over my comfort zone. Needless to say, it was a great learning experience and I will

cherish my time spent here.

I am grateful to my colleagues in Ron's research lab who made the last five years both fun and educational. I would like to thank my co-authors Jon Gretarsson, Frank Losasso, Jon Su and Jerry Talton. Special thanks to Jon Gretarsson with whom I worked the closest with and had a great collaboration. I would also like to thank other members of Ron's group with whom I had a lot of fun working and interacting. Thanks to Michael Lentine, Craig Schroeder, Mridul Aanjaneya, Tamar Shinar, Avi Robinson-Mosher, Wen Zheng, Geoffrey Irving, Eftychios Sifakis, Elliot English, Andrew Selle and Rachel Weinstein. Special thanks to Mridul Aanjaneya with whom I worked closely in the last year and enjoyed a lot.

I would like to thank Ron Fedkiw, Charbel Farhat and Vladlen Koltun for sitting on my reading and defense committees. Thanks to Juan Alonso for chairing my defense committee and Margot Gerritsen for sitting on my thesis committee.

I would also like to thank my master's advisors at Georgia Tech: Irfan Essa, Peter Mucha and Greg Turk for their exceptional support and guidance. Also thanks to my co-author Chris Wojtan with whom I had a lot of fun working. I am also very thankful to my bachelor's advisors at Indian Institute of Technology, Delhi: Subhashis Banerjee and Prem K Kalra who were very instrumental in generating my interest in research and computer science. Special thanks to my co-authors and dear friends Sumit Jain and Dhruv Mahajan, with whom I had more fun working than ever.

I would like to thank my grandmother for her blessings and love, which I have cherished my entire life. I am very thankful to my parents-in-law for their immense love, care and support for all these years. I would like to thank my sister-in-law, Aditi, for her unconditional support and for all the fun discussions we had. I would also like to thank my sisters-in-law, Megha and Aanchal, for their friendship and for all the fun times we spent together. I am also thankful to my admirable nephews, Ishansh and Vidit, who were a constant source of fun and joy.

I am extremely thankful to my wonderful wife, Neha, without whom it would have been impossible to complete this journey. She always stood by me during times good and bad, and was a constant source of encouragement, support and motivation. I am grateful to her for the trust she showed in me and her invaluable guidance throughout.

I have deep regard for her unconditional love and for the immeasurable sacrifices she made for me. Neha, thanks for making the last five years, the best years of my life.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Physical simulation of highly nonlinear compressible flows with shocks, contacts and rarefactions is an important topic with applications in both the scientific community and computer graphics. Another important area of study in the scientific community is the interaction of compressible flow with structures. The computer graphics community is also interested in the ability to simulate both the initial states of the explosion including shock waves along with the long time behavior of rolling plumes and other incompressible flow effects. In this dissertation we address these problems and propose practical methods for the simulation of compressible flow, its interaction with solid structures, and integrating shock wave dynamics into traditional smoke simulations.

Traditionally highly nonlinear compressible flow problems are solved with explicit time integration (Runge-Kutta methods, ENO, WENO etc.). Although these methods produce high quality results, small time steps are required in order to enforce the CFL condition of information moving only one grid cell per time step. This is is too restrictive for flows where the sound speed, c, may be much larger than $|u|$. In this case, a large number of time steps are required if one is interested in the motion of the fluid particles over an appreciable distance in the low Mach number regions.

To alleviate the stringent CFL restriction, we propose a semi-implicit method where the calculation is divided into two parts: advection and non-advection. The

advection terms are treated with explicit time integration, and thus the CFL restriction on the material velocity remains. Whereas one can use a standard method such as ENO in solving the advection terms, we found that when coupled to an implicit solution of the pressure equations (that is inherently central-differenced) the standard ENO method sometimes leads to spurious oscillatory behavior. Thus we designed a new ENO method geared towards a MAC grid discretization of the data, making it more similar to incompressible flow. We call this MAC-ENO or MENO. The remaining non-advection terms are solved using an implicit equation for the pressure using a standard MAC grid type formulation. The derivation of our method is based on the pressure evolution equation as discussed in [11], which makes it valid for general equations of state, arbitrary chemical species etc. Thus, our derivation has fewer assumptions and is more straight forward than previous work. Also, our method is fully conservative and thus shocks are tracked at the right speed.

Developing on this implicit pressure solve, we propose a novel method to implicitly two-way couple Eulerian compressible flow to volumetric Lagrangian solids. Traditional methods to capture fluid-structure interactions can be broadly separated into two categories. Weakly coupled (partitioned) systems interleave the disparate subsystems by integrating them forward in time separately, using each others' results as boundary conditions in an alternating one-way coupled fashion. This approach is appealing as it permits the use of specialized numerical methods for each of the different materials with only slight modifications. There are disadvantages to this approach, however, for example new and poorly understood stability restrictions arise independent of the individual subsystems, such as the lumped-mass instability discussed in [4]. The alternative is to employ a strongly coupled (monolithic) system, which are systems where the fluid and structure are evolved forward in time simultaneously using a solver specially crafted to incorporate phenomena from both fluid and solid phases. Although these methods are more stable, they require using a uniform simulation scheme for both fluids and structures. State-of-the-art solvers typically use an Eulerian framework to treat fluid flows and a Lagrangian framework to treat solids, and so such a coupled system has to either model the solid in an Eulerian framework, or model the fluid in a Lagrangian framework. Both these methods are undesirable

as the impose significant limitations on the numerical method.

Our method is a hybrid of the two categories, the explicit components of both fluid and solid solvers are evolved forward independently, while the implicit components and interactions are coupled together in a monolithic solve. We address the challenge of coupling Eulerian fluids with Lagrangian solids by introducing an interpolation operator, which conservatively maps quantities from Eulerian boundaries to nearby Lagrangian boundary nodes, and vice versa. A time step update in our method can be briefly described in two broad stages. In the first stage, the solid position is updated and explicit forces are applied to update its velocity. At the same time the fluid is updated with the explicit components of the flux, including body forces, advection, etc. Since the fluid and structure updates are independent, it allows us to use state-of-the-art Lagrangian schemes for the structure, and Eulerian schemes for the fluid. The second stage handles the complex transfer of information at the fluid-structure interface. Here we exploit the structure of our implicit pressure solver for compressible flow, which allows us to construct a coupled monolithic solve for the implicit fluid-structure coupling forces.

Finally, using our implicit pressure solve, we propose a practical method for integrating shock wave dynamics into traditional smoke simulations for graphics applications. Previous methods in computer graphics used simplified models to simulate effects like blast waves, and explosion phenomena. Some methods rely on the incompressible formulation and add an artifitial divergence to model explosions. Although they generate nice fireball-style effects, but the underlying incompressible assumption discards the compression waves and potentially dramatic effects such as shocks. To simulate physically accurate shock waves, some papers tried a full simulation of the compressible flow equations using traditional explicit methods. As discussed above these methods impose severe restrictions on the time step, and continue to be expensive even when interesting compressible flow effects such as shock waves leave the domain. As such, these methods show shocks moving around (in slow motion), and stop their simulations/video after a short time. In the real world these shocks eventually dissipate as do the effects of compressibility, leading eventually to a plume type structure more representative of smoke and fire - governed more appropriately by

incompressible flow. It is computationally infeasible for existing methods to simulate what happens to a flow field over 5-10 seconds when a large amount of computational resources are needed to advance a fraction of a millisecond.

For graphics applications, we propose to transition the flow from compressible to fully incompressible by sending $c \to \infty$. Non-physically driving the sound speed to ∞ accelerates the behavior of the fluid in order to obtain incompressible style flow phenomena such as rolling and plumes much quicker than one would otherwise attain. One could then use fast schemes typically used for incompressible simulation, such as semi-Lagrangian, BFECC or MacCormack advection [50, 6, 28, 43]. Other simulation techniques such as vorticity confinement [51] and vortex particles [44] can also be used to enhance the visual fidelity and speed of the smoke plume simulations. We note that any explicit method would have its time step driven to zero as the sound speed is driven to ∞, and therefore would not make any progress towards the incompressible flow behavior. Thus our semi-implicit method lends itself well to this approach.

The material presented in this thesis is based on previously published works [30], [18] and [29].

Chapter 2

Semi-implicit compressible flow

We propose a novel method for alleviating the stringent CFL condition imposed by the sound speed in simulating inviscid compressible flow with shocks, contacts and rarefactions. Our method is based on the pressure evolution equation, so it works for arbitrary equations of state, chemical species etc, and is derived in a straight-forward manner. Similar methods have been proposed in the literature, but the equations they are based on and the details of the methods differ significantly. Notably our method leads to a standard Poisson equation similar to what one would solve for incompressible flow, but has an identity term more similar to a diffusion equation. In the limit as the sound speed goes to infinity, one obtains the Poisson equation for incompressible flow. This makes the method suitable for two-way coupling between compressible and incompressible flows and fully implicit solid-fluid coupling, although both of these applications are left to future work. We present a number of examples to illustrate the quality and behavior of the method in both one and two spatial dimensions, and show that for a low Mach number test case we can use a CFL number of 300 (whereas previous work was only able to use a CFL number of 3 on the same example).

2.1 Introduction

In this paper, we focus on highly nonlinear compressible flows with shocks, contacts and rarefactions, for example the Sod shock tube. Traditionally these types of problems are solved with explicit time integration (Runge-Kutta methods, ENO, WENO etc, see e.g. [47, 48, 23]). Although these methods produce high quality results, small time steps are required in order to enforce the CFL condition of information moving only one grid cell per time step. While this is understandable for very high Mach number flow where $|u|$, $|u - c|$ and $|u + c|$ are all of similar magnitude, it is too restrictive for flows where the sound speed, c, may be much larger than $|u|$. Moreover some flow fields might have both high Mach number regions where shock waves are of interest as well as low Mach number regions where the material velocities are important. In this case, a large number of time steps are required if one is interested in the motion of the fluid particles over an appreciable distance in the low Mach number regions. Thus, it can be quite useful to have methods that avoid the stringent CFL time step restriction imposed by the acoustic waves and instead use only the material velocity CFL restriction (albeit one would expect some loss of quality because of the implicit treatment of the acoustic waves).

To alleviate the stringent CFL restriction, [24] proposed both a non-conservative and a conservative scheme. Their non-conservative scheme builds on the predictor-corrector type scheme of [60] to derive an elliptic pressure equation quite similar to ours, but for an adiabatic fluid. Our method is similar in spirit to [24, 56, 57, 58] where the calculation is divided into two parts: advection and non-advection. The advection terms are treated with explicit time integration, and thus the CFL restriction on the material velocity remains. Whereas one can use a standard method such as ENO in solving the advection terms, we found that when coupled to an implicit solution of the pressure equations (that is inherently central-differenced) the standard ENO method sometimes leads to spurious oscillatory behavior. Thus we designed a new ENO method geared towards a MAC grid discretization of the data, making it more similar to incompressible flow. We call this MAC-ENO or MENO. The remaining non-advection terms are solved using an implicit equation for the pressure using a

standard MAC grid type formulation. Since the MAC grid is dual in both velocity and pressure (noting that the MAC grid pressure needs to live at cell faces for flux based methods), one needs to interpolate data back and forth.

We base the derivation of our method on the pressure evolution equation as discussed in [11], thus making it valid for general equations of state, arbitrary chemical species etc. Thus, our derivation has fewer assumptions and is more straight forward than previous work, especially those based on preconditioners. For example, [56] makes two critical assumptions in their derivation of the implicit equation for pressure. In approximating the derivative of momentum they discard a $\Delta t \frac{\nabla p}{\rho}$ term, and their pressure evolution equation is missing the advection term. Also, our method is fully conservative and thus shocks are tracked at the right speed. We present a number of traditional examples for highly non-linear compressible flows including the Sod shock tube, interacting blast waves, and in two dimensions we show Flow Past a Step, Double Mach Reflection of a Strong Shock, and a Circular Shock. We also demonstrate that the method works well for low Mach number flow, taking an example from [25] where the authors obtain reasonable results with a CFL number of 3. Notably, our method allows a CFL number of 300 (two orders of magnitude more).

2.2 Numerical Method

Let us consider the one dimensional Euler equations,

$$\begin{pmatrix} \rho \\ \rho u \\ E \end{pmatrix}_t + \begin{pmatrix} \rho u \\ \rho u^2 + p \\ Eu + pu \end{pmatrix}_x = 0,$$

with ρ being the density, u the velocity, E the total energy per unit volume and p the pressure. The flux term can be separated into an advection part and a non-advection part,

$$\mathbf{F_1}(\mathbf{U}) = \begin{pmatrix} \rho u \\ \rho u^2 \\ Eu \end{pmatrix}, \mathbf{F_2}(\mathbf{U}) = \begin{pmatrix} 0 \\ p \\ pu \end{pmatrix}. \tag{2.1}$$

We first compute the Jacobian of the advection part

$$\mathbf{J} = \begin{pmatrix} 0 & 1 & 0 \\ -u^2 & 2u & 0 \\ -\frac{Eu}{\rho} & \frac{E}{\rho} & u \end{pmatrix}.$$

All the Jacobian's eigenvalues are equal to u, and it is rank deficient with left eigenvectors of $(u, -1, 0)$ and $(E/\rho, 0, -1)$ and right eigenvectors of $(1, u, 0)^T$ and $(0, 0, 1)^T$. Since all the characteristic velocities are identical, we can apply component wise upwinding to $\mathbf{F_1}(\mathbf{U})$ without having to transform into the characteristic variables first (as in [13]). Moreover, this advection part only requires a time step restriction based on u.

2.2.1 Implicit Pressure Update

The multi-dimensional Euler equations are

$$\begin{pmatrix} \rho \\ \rho u \\ \rho v \\ \rho w \\ E \end{pmatrix}_t + \begin{pmatrix} \rho u \\ \rho u^2 \\ \rho uv \\ \rho uw \\ Eu \end{pmatrix}_x + \begin{pmatrix} \rho v \\ \rho uv \\ \rho v^2 \\ \rho vw \\ Ev \end{pmatrix}_y + \begin{pmatrix} \rho w \\ \rho uw \\ \rho vw \\ \rho w^2 \\ Ew \end{pmatrix}_z + \begin{pmatrix} 0 \\ \nabla p \\ \nabla \cdot (p\vec{u}) \end{pmatrix} = 0,$$

where $\vec{u} = (u, v, w)$ are the velocities. Here we have advection components in each of the 3 spatial dimensions, and they can be handled as outlined previously in a dimension by dimension fashion (as in [48]).

We apply a time splitting as is typical for incompressible flow formulations, first updating the advection terms to obtain an intermediate value of the conserved variables $(\rho)^*$, $(\rho u)^*$, and E^*, and afterward correct these to time t^{n+1} using an implicit pressure. Since the pressure does not affect the continuity equation, $\rho^{n+1} = \rho^*$. The non-advection momentum and energy updates are

$$\frac{(\rho\vec{u})^{n+1} - (\rho\vec{u})^*}{\Delta t} = -\nabla p \tag{2.2}$$

and

$$\frac{E^{n+1} - E^*}{\Delta t} = -\nabla \cdot (pu). \tag{2.3}$$

Taking motivation from the standard incompressible flow formulation (which uses the momentum equation to derive an implicit equation for pressure), we divide equation (2.2) by ρ^{n+1},

$$\vec{u}^{n+1} = \vec{u}^\star - \Delta t \frac{\nabla p}{\rho^{n+1}}, \tag{2.4}$$

and take its divergence to obtain

$$\nabla \cdot \vec{u}^{n+1} = \nabla \cdot \vec{u}^\star - \Delta t \nabla \cdot \left(\frac{\nabla p}{\rho^{n+1}} \right). \tag{2.5}$$

In the case of incompressible flow, we would set $\nabla \cdot \vec{u}^{n+1} = 0$, but for compressible flow we instead use the pressure evolution equation derived in [11],

$$p_t + \vec{u} \cdot \nabla p = -\rho c^2 \nabla \cdot \vec{u}. \tag{2.6}$$

If we fix $\nabla \cdot \vec{u}$ to be at time $n+1$ through the time step (making an $\mathcal{O}(\Delta t)$ error), we can substitute in equation (2.5) to get

$$p_t + \vec{u} \cdot \nabla p = -\rho c^2 \nabla \cdot \vec{u}^\star + \rho c^2 \Delta t \nabla \cdot \left(\frac{\nabla p}{\rho^{n+1}} \right), \tag{2.7}$$

which is an advection-diffusion equation with a source term. Discretizing the $\vec{u} \cdot \nabla p$ advection term explicitly, using a forward Euler time step, and defining the diffusive pressure at time t^{n+1} as is typical for backward Euler discretization, gives after rearrangement

$$p^{n+1} - \rho^n (c^2)^n \Delta t^2 \nabla \cdot \left(\frac{\nabla p^{n+1}}{\rho^{n+1}} \right) = (p^n - (\vec{u}^n \cdot \nabla p^n) \Delta t) - \rho^n (c^2)^n \Delta t \nabla \cdot \vec{u}^\star. \tag{2.8}$$

Note we have discretized ρc^2 at time t^n. This equation can be further simplified by using the advection equation for pressure,

$$\frac{p^a - p^n}{\Delta t} + \vec{u}^n \cdot \nabla p^n = 0$$

to obtain

$$p^a = p^n - (\vec{u}^n \cdot \nabla p^n)\Delta t, \tag{2.9}$$

where p^a is an advected pressure which can be computed using HJ ENO [37] or semi-Lagrangian advection [5]. Substituting in equation (2.8) we obtain

$$p^{n+1} - \rho^n (c^2)^n \Delta t^2 \nabla \cdot \left(\frac{\nabla p^{n+1}}{\rho^{n+1}} \right) = p^a - \rho^n (c^2)^n \Delta t \nabla \cdot \vec{u}^\star. \tag{2.10}$$

We discretize this equation at cell centers (which is typical for advection-diffusion equations) and thus need to define velocities at cell faces for $\nabla \cdot \vec{u}^\star$. Consider two

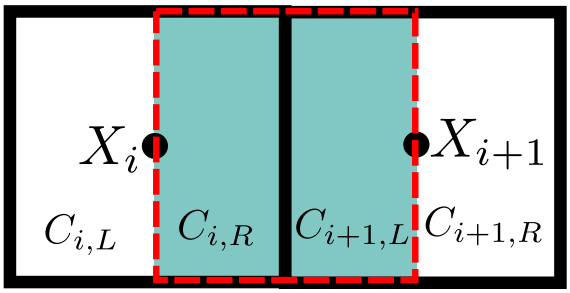


Figure 2.1: Two adjacent grid cells with the dual cell in between.

adjacent grid cells, one centered at X_i and one centered at X_{i+1}. We divide these into four regions $C_{i,L}$, $C_{i,R}$, $C_{i+1,L}$, $C_{i+1,R}$, where $(C_{i,R} \cup C_{i+1,L})$ represents a dual cell (see figure 2.1). Then equation (2.2) for $C_{i,R}$ is

$$\frac{(\rho u)^{n+1}_{i,R} - (\rho u)^*_{i,R}}{\Delta t} = -\frac{p^{n+1}_{i+1/2} - p^{n+1}_i}{\Delta x/2}. \tag{2.11}$$

Similarly for $C_{i+1,L}$ we have

$$\frac{(\rho u)^{n+1}_{i+1,L} - (\rho u)^*_{i+1,L}}{\Delta t} = -\frac{p^{n+1}_{i+1} - p^{n+1}_{i+1/2}}{\Delta x/2}. \tag{2.12}$$

Adding these equations together and dividing by $(\rho_i + \rho_{i+1})$ yields

$$\frac{\hat{u}_{i+1/2}^{n+1} - \hat{u}_{i+1/2}^{*}}{\Delta t} = -\frac{p_{i+1}^{n+1} - p_i^{n+1}}{\Delta x \hat{\rho}^{n+1}}, \tag{2.13}$$

where $\hat{u}_{i+1/2} = \frac{(\rho u)_{i,R} + (\rho u)_{i+1,L}}{\rho_i + \rho_{i+1}} = \frac{(\rho u)_i + (\rho u)_{i+1}}{\rho_i + \rho_{i+1}}$ can be thought of as a density-weighted face velocity, and $\hat{\rho}_{i+1/2} = \frac{\rho_i + \rho_{i+1}}{2}$ is the cell face density. Note that we currently use $(\rho u)_{i,R} = (\rho u)_i$ and $(\rho u)_{i+1,L} = (\rho u)_{i+1}$, although higher order approximations could be used. Using this discretization on equation (2.10) yields

$$\left[I + \rho^n (c^2)^n \Delta t^2 G^T \left(\frac{1}{\hat{\rho}^{n+1}} G \right) \right] p^{n+1} = p^a + \rho^n (c^2)^n \Delta t G^T \vec{\hat{u}}^{\star}, \tag{2.14}$$

where G is our discretized gradient operator and $-G^T$ is our discretized divergence operator. This is solved to obtain p^{n+1} at cell centers.

In summary, instead of using an equation of state (EOS) to find the pressure for use as a flux in both conservation of momentum and energy, we use equation (2.14). The EOS still plays a role because it is used to determine the time t^n pressures which factor into p^a and is also used to determine $(c^2)^n$. In figure 2.2 we show an example calculation of the pressure for our Sod shock tube example. In the picture we plot the pressure using the equation of state at time t^n, i.e. p^n, the pressure calculated using equation (2.14), i.e. our p^{n+1}, and also the pressure calculated using the EOS applied to the conservative variables at time t^{n+1}, i.e. p_{EOS}^{n+1}. Notice in the figure that the pressure calculated from equation (2.14) is a good approximation to what the pressure will be at the next time step (i.e. p_{EOS}^{n+1}) emphasizing the implicit nature of our scheme. p^n is the pressure used in a typical explicit scheme.

It is interesting to note that this derivation does not require an ideal gas assumption, and hence should be general enough to work with any EOS (even multi-species flow [11]).

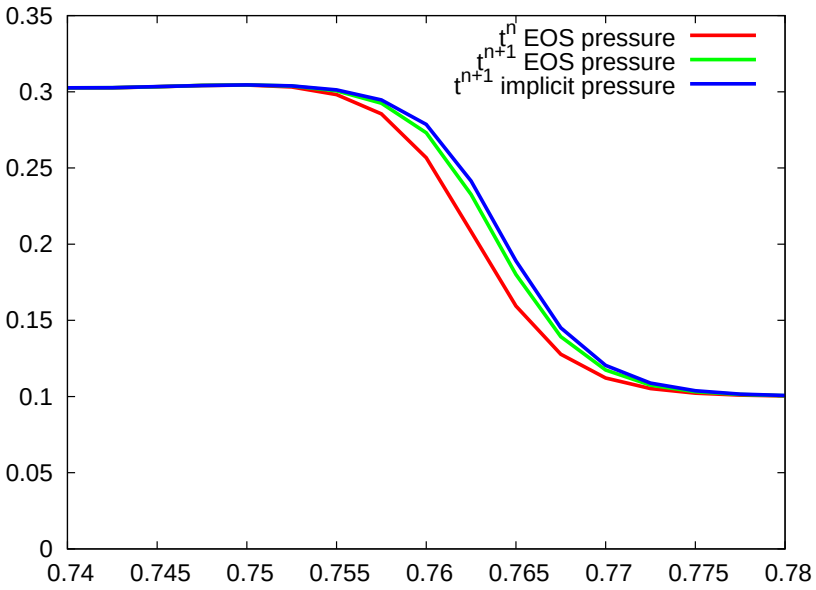


Figure 2.2: A blow-up of the pressure plot for example 6.1.1 at time $t(n) = .149s$ and $t(n+1) = .15s$, showing that the implicit pressure calculated in equation (2.14) is a good approximation to what the pressure will be at time t^{n+1} emphasizing the implicit nature of our scheme. p^n is also plotted to emphasize the difference between using an implicit and explicit pressure.

2.2.2 Updating Momentum and Energy

To obtain the correct shock speeds we use a flux based method and thus need the pressure at cell faces for equations (2.2) and (2.3), and the velocity at cell faces for equation (2.3). Applying conservation of momentum to the control volumes $C_{i,R}$ and $C_{i+1,L}$ (see figure 2.1) gives

$$Du_{i,R}/Dt = (p_i - p_{i+1/2})/(\Delta x \rho_{i,R}/2)$$

and

$$Du_{i+1,L}/Dt = (p_{i+1/2} - p_{i+1})/(\Delta x \rho_{i+1,L}/2).$$

The constraint that the interface remain in contact implies that $Du_{i,R}/Dt = Du_{i+1,L}/Dt$, which can be used with the aforementioned equations to solve for the pressure at the flux location $X_{i+1/2}$ as

$$p_{i+1/2} = \frac{p_{i+1}\rho_i + p_i\rho_{i+1}}{\rho_{i+1} + \rho_i}. \tag{2.15}$$

For solid wall boundaries, we reflect the pressure and density values as usual, and then use equation (2.15). The cell face velocity is computed via equation (2.13), and $p_{i+1/2}\hat{u}_{i+1/2}$ is used in equation (2.3).

2.3 Time Step Restriction

The eigenvalues of the Jacobian of the advection part of the flux are all u. Since we solve the acoustic component implicitly, we no longer have a severe time step restriction determined by the speed of sound c, and all that remains is to find an estimate for the maximum value of $|u|$ throughout the time step. Simply using u^n is not enough, since e.g. Sod shock tube starts out with an initial velocity identically zero and thus u^n would imply an infinite Δt. To alleviate this, we add a term that estimates the change in velocity over a time step similar to what was done in [27]. Assuming the flow is smooth, we combine conservation of mass and momentum to give an equation for the velocity, $u_t + u \cdot \nabla u + \frac{\nabla p}{\rho} = 0$. The temporal update of this equation would advect velocity based on the $u \cdot \nabla u$ term, but also increase the velocity by an amount equal to $\frac{\nabla p}{\rho}$. In one spatial dimension, we use this to estimate the velocity at the end of the time step as $\left(\frac{|u^n|_{max} + \frac{|p_x|}{\rho}\Delta t}{\Delta x}\right)$ and the CFL condition becomes

$$\Delta t \left(\frac{|u^n|_{max} + \frac{|p_x|}{\rho}\Delta t}{\Delta x}\right) \leq 1. \tag{2.16}$$

This is quadratic in Δt with solutions

$$\frac{-|u^n|_{max} - \sqrt{|u^n|^2_{max} + 4\frac{|p_x|}{\rho}\Delta x}}{2|p_x|/\rho} \leq \Delta t \leq \frac{-|u^n|_{max} + \sqrt{|u^n|^2_{max} + 4\frac{|p_x|}{\rho}\Delta x}}{2|p_x|/\rho}.$$

As the lower limit is always non positive and $\Delta t \geq 0$, we only need to enforce the upper bound. As $p_x \to 0$, both the numerator and denominator vanish and thus we obtain a more convenient time step restriction by replacing the 2^{nd} Δt in equation (2.16) with this upper bound to obtain

$$\frac{\Delta t}{2}\left(\frac{|u^n|_{max}}{\Delta x}+\sqrt{\left(\frac{|u^n|_{max}}{\Delta x}\right)^2+4\frac{|p_x|}{\rho\Delta x}}\right)\leq 1. \tag{2.17}$$

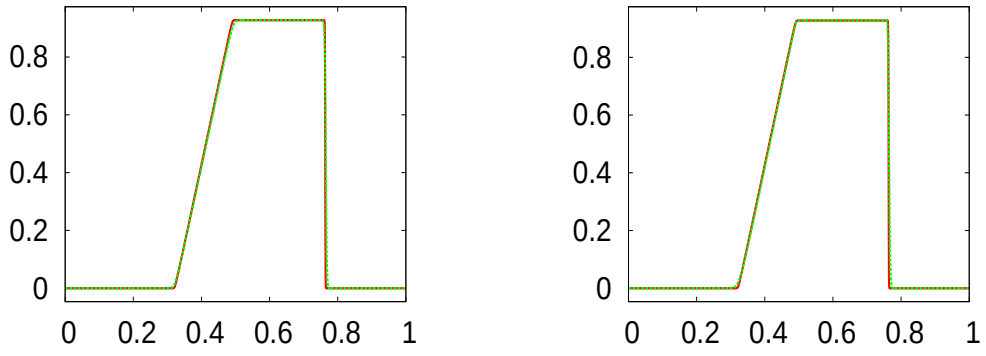
Note that this is not linear in Δx, but as $\Delta x \to 0$ we obtain a more typical CFL condition $\Delta t < \frac{\Delta x}{|u^n|_{max}}$. In two spatial dimensions our CFL follows along the lines of [27]'s equation 95 and is given by:

$$\frac{\Delta t}{2}\left(\frac{|u|_{max}}{\Delta x}+\frac{|v|_{max}}{\Delta y}+\sqrt{\left(\frac{|u|_{max}}{\Delta x}+\frac{|v|_{max}}{\Delta y}\right)^2+4\frac{|p_x|}{\rho\Delta x}+4\frac{|p_y|}{\rho\Delta y}}\right)\leq 1.$$

All of our examples are stable for CFL number $\alpha = .9$, and *all* of our examples were unstable for $\alpha = 1.3$. Some examples (e.g. example 6.1.8) blow up for $\alpha = 1$.

2.4 Modified ENO Scheme

When using traditional ENO methods for the advection part of our equations (as in [48]), we obtained excessive spurious oscillations. This seems to be related to our dual cell center and MAC grid formulation, thus we device a new ENO scheme which better utilizes that dual formulation. We call this Mach-ENO or MENO. The main idea is to replace the advection velocity with the MAC grid value defined at the flux in question, i.e. $\hat{u}$. The lowest level of the divided difference table is typically constructed with the physical fluxes, i.e. ρu, ρu^2 and Eu for $\mathbf{F_1}(\mathbf{U})$ in equation (2.1). A dissipation term is added for the local and global Lax-Friedrichs versions. Consider constructing an ENO approximation for the flux at $X_{i+1/2}$. Locally, we would use a divided difference table with base values corresponding to the physical fluxes plus or minus the appropriate dissipation. Our modification is to replace $\rho_j u_j$, $\rho_j {u_j}^2$, and $E_j u_j$ with $\rho_j \hat{u}_{i+1/2}$, $\rho_j u_j \hat{u}_{i+1/2}$, and $E_j \hat{u}_{i+1/2}$ leaving the dissipation terms unaltered.



(a) Standard ENO-LLF (b) MENO-LLF

Figure 2.3: Sod shock tube problem at $t = .15s$. Left: Standard ENO-LLF (Local Lax-Friedrichs) using 401 grid points (green) and 1601 grid points (red). Right: The base 1601 grid points solution is the same as in the left figure, but the coarse grid calculation (with 401 grid points) is done with the new MENO scheme. Velocity is shown in both figures. Both simulations were done with explicit time integration and a full characteristic decomposition in order to demonstrate that the new ENO schemes performs similar to the old one when one is not using our new implicit discretization of the pressure.

Note that $\hat{u}_{i+1/2}$ is fixed throughout the divided difference table similar to the way one fixes the dissipation coefficient.

In order to validate our new MENO scheme, we compared it to the standard scheme from [48] for the standard Sod shock tube in Figure 2.3. For this problem and other fully explicit simulations the results were fairly similar, but when we ran the simulations with our semi-implicit formulation the MENO scheme performed much better, and in fact the standard ENO scheme was not successful in producing any solution whatsoever for figure 2.11 in our examples section.

2.5 Time Integration

While the explicit component of our update is an upwind scheme, the implicit component is centrally-differenced. This tends to introduce more dispersive rather than dissipative errors to the solution (i.e. there is more of an imaginary component to the eigenvalues), which suggests the use of Runge-Kutta over forward Euler.

We use two variations of the third order TVD Runge-Kutta scheme [47] in all of our examples. The first is to perform Runge-Kutta on just the advection part, $\mathbf{F_1}(\mathbf{U})$, with only one final implicit solve for $\mathbf{F_2}(\mathbf{U})$. The second variation is to carry out both $\mathbf{F_1}(\mathbf{U})$ and $\mathbf{F_2}(\mathbf{U})$ for each Runge-Kutta stage, noting that this has three times the computational cost as far as the implicit solution of $\mathbf{F_2}(\mathbf{U})$ is concerned. In general we observed better performance, especially in controlling overshoots, when using the second variation (see figure 2.4). However, some examples (in particular the high Mach number ones) do tend to show more oscillations (see figure 2.4, bottom). These oscillations are less predominant when combined with MENO, so we show all of our examples with the second variation.

2.6 Numerical Results

2.6.1 One dimensional Validation

For the one dimensional tests, we use a computational domain of $[0, 1]$, 401 grid points, and also plot a baseline solution using 1601 grid points in the standard fully explicit ENO method as in [48]. A second order ENO was used along with the CFL number of .5. Unless otherwise noted the maximum Mach number in each example lies within the range $(.9, 2.5)$. All units are in S.I. Generally speaking our method is a perturbation of those proposed by [56, 57] and thus demonstrates similar qualitative behavior. Timings are shown in table 2.1. In particular note that the implicit scheme is generally more efficient than the explicit scheme predominantly because we avoid the characteristic decomposition and can advect all three independent variables simultaneously because they all have the same eigenvalue u. At first glance one might assume that the necessity of a pressure Poisson equation would cancel

out these efficiency gains. However, being strongly diagonally dominant, the implicit solve typically converges after only a few iterations of the conjugate gradient method. In practice we noticed that only five or six iterations were required to reach a reasonable tolerance. It is unclear whether our newly proposed semi-implicit method would have these slight efficiency gains across a wider number of examples and in multiple spatial dimensions, however for the low Mach number flow problems for which it was designed (such as example 6.1.8) it is significantly more efficient than the explicit method.

Sod Shock Tube

Our first test case is a standard Sod shock tube with initial conditions of

$$(\rho(x,0), u(x,0), p(x,0)) = \begin{cases} (1,0,1) & \text{if } x \leq .5, \\ (.125,0,.1) & \text{if } x > .5. \end{cases}$$

Our results are shown in Figure 2.5, which indicate well resolved shock, rarefaction and contact solutions. Since our method is conservative, we get the correct shock speeds. The results are comparable to that of [25] and [56].

Lax's Shock Tube

Lax's shock tube is similar in nature to Sod shock tube, except that the initial condition has a discontinuity in the velocity:

$$(\rho(x,0), u(x,0), p(x,0)) = \begin{cases} (.445,.698,3.528) & \text{if } x \leq .5, \\ (.5,0,.571) & \text{if } x > .5. \end{cases}$$

Our results are shown in Figure 2.6. Again, the results are comparable to the previous work.

Strong Shock Tube

The Strong shock tube problem poses initial conditions that generates a supersonic shock:

$$(\rho(x,0), u(x,0), p(x,0)) = \begin{cases} (1, 0, 10^{10}) & \text{if } x \leq .5, \\ (.125, 0, .1) & \text{if } x > .5. \end{cases}$$

Our results are shown in Figure 2.7. The scheme admits some oscillations near the rarefaction wave, and we see no notable difference in simulation time when compared to the explicit simulation. With that in mind, we note that the main advantage of the proposed method is to take time steps irrespective of the sound speed values; in cases of high Mach number flows (or high Mach number regions of the flow – if asynchronous time integration is used), one could use a typical ENO scheme.

Mach 3 Shock Test

The initial conditions for the Mach 3 shock test are:

$$(\rho(x,0), u(x,0), p(x,0)) = \begin{cases} (3.857, .92, 10.333) & \text{if } x \leq .5, \\ (1, 3.55, 1) & \text{if } x > .5. \end{cases}$$

Our results are shown in Figure 2.8. As above we do note some oscillations near the rarefaction wave.

High mach flow test

The initial conditions for the High mach flow test are:

$$(\rho(x,0), u(x,0), p(x,0)) = \begin{cases} (10, 2000, 500) & \text{if } x \leq .5, \\ (20, 0, 500) & \text{if } x > .5. \end{cases}$$

As noted in [25] the Mach number in this test can reach as high as 240. Our results are shown in Figure 2.9.

Interaction of blast waves

Here we present a test of two interacting blast waves. This problem was introduced by [55] and involves multiple strong shock waves. The initial conditions for the test are:

$$(\rho(x,0), u(x,0), p(x,0)) = \begin{cases} (1, 0, 10^3) & \text{if } 0 \leq x < .1, \\ (1, 0, 10^{-2}) & \text{if } .1 \leq x < .9, \\ (1, 0, 10^2) & \text{if } .9 \leq x \leq 1. \end{cases}$$

We also have solid wall boundary conditions at $x = 0$ and $x = 1$. Our results are shown in Figure 2.10 which shows that we achieve very accurate results.

Two Symmetric Rarefaction Waves

In this test there are two rarefaction waves going in opposite directions from the center of the domain. This causes very low density regions near the center of the domain. The initial conditions for the test are:

$$(\rho(x,0), u(x,0), p(x,0)) = \begin{cases} (1, -2, .4) & \text{if } x \leq .5, \\ (1, 2, .4) & \text{if } x > .5. \end{cases}$$

Our results are shown in Figure 2.11. Our results are comparable to that of [25] and [56]. Note that there is an unphysical pulse in the internal energy field near the low pressure region, caused by overheating (see e.g. [12]).

Smooth Flow Test (Mach Zero Limit)

The initial conditions for the zero mach limit test are given by:

$$\begin{aligned} u(x,0) &= 0 \\ p(x,0) &= p_0 + \epsilon p_1(x) \\ p_1(x) &= 60\cos(2\pi x) + 100\sin(4\pi x) \\ \rho(x,0) &= \left(\frac{p(x,0)}{p_0}\right)^{\frac{1}{\gamma}} \rho_0 \end{aligned}$$

Where $\rho_0 = 1$, $p_0 = 10^9$, and $\epsilon = 10^3$. Since the flow is smooth and there are no shocks in this test, we have used a single implicit solve per time step. This test is dominated by acoustic waves (as observed in [25]). We can take time steps as large as is permitted by our CFL condition in equation (2.17). This permits time steps three orders of magnitude greater than those permitted by sound-speed based CFL. However, as with all implicit schemes, taking too large a time step can lead to inaccurate results. Thus, in order to get sufficient accuracy, we clamp our time step to be a fixed multiple of the explicit time step (which is calculated using the sound-speed based CFL). In figure 2.12 we use 3 times the explicit time step and show convergence via grid resolution.

In a second suit of tests we show that we can increase the grid resolution without the need to refine the time step. The timing results for this experiment are available in table 2.2, where Δt remains fixed as the grid resolution goes up as high as $320,000$ grid cells. At that point the effective sound speed CFL is 300. Numerical results are plotted in figure 2.13 and table 2.2 summarizes the results. In particular we note that the newly proposed implicit method permits a fixed time step all the way up to $320,000$ grid points. This allows the wall clock simulation time to scale approximately linear to the size of the problem (since we solve the Poisson equation using conjugate gradients, which has superlinear complexity – however, note that one only needs the solver to converge in the sense of truncation error as opposed to round-off error). On the other hand, in explicit methods the simulation time grows quadratically, becoming impractical at $320,000$ grid points. Note that since we are not refining the time step, we do not expect to see any further convergence in the solution.

2.6.2 Flow Past a Step Test

Our first two dimensional experiment is similar to the one described in [12]. We assume an ideal gas with $\gamma = 1.4$. The test domain is 3 units long and 1 unit wide, with a .2 unit high step which is located .6 units from the left hand side of the tunnel. The initial conditions are $\rho = 1.4$, $p = 1$ and $u = 3$ and $v = 0$ everywhere in the domain. We apply an inflow boundary condition on the left hand side of the

domain, and an outflow boundary condition on the right hand side of the domain. A reflective solid wall boundary condition is applied for the top and bottom boundaries of the domain. We show numerical results at $t = 4s$ on a grid of resolution 120x40 in figure 2.14.

2.6.3 Double Mach Reflection of a Strong Shock

In a computational domain of $[0, 4] \times [0, 1]$, a planar Mach 10 shock hits a reflecting boundary that lies along the bottom wall of the domain along $x \in [\frac{1}{6}, 4]$. The plane of the shock begins at $(\frac{1}{6}, 0)$ and makes a 60° angle with the reflecting plane. The left and bottom (for $x \in (0, \frac{1}{6})$) boundary conditions are given by the postshock condition, the right boundary by a zero-gradient condition, and the top boundary is set to describe the exact motion of the Mach 10 shock. If we take $\vec{n}$ to be the unit vector that lies normal to the planar shock, then the initial values are given by:

$$(\rho(x, y, 0), u(x, y, 0), p(x, y, 0)) = \begin{cases} (1.4, \vec{0}, 1) & \text{preshock} \\ (8, 8.25\vec{n}, 116.5) & \text{postshock} \end{cases}.$$

Our method (see figure 2.15) compares well with those provided in [55], which provides a description of this example and presents numerical results comparing the performance of various methods in this problem. As is done in previous work we only show the domain of interest ($[0, 3] \times [0, 1]$).

2.6.4 Circular Shock Test

The circular shock test has an initial condition prescribed as

$$(\rho, u, v, p) = \begin{cases} (1, 0, 0, 1) & \text{if } r \leq .4 \\ (.125, 0, 0, .1) & \text{if } r > .4, \end{cases}$$

where $r = \sqrt{x^2 + y^2}$. Numerical results are shown in figure 2.16. The same test was shown in [57]. Our results indicate well resolved shock and contact solutions along

with correct speed shock calculations.

2.7 Conclusions and Future Work

We have presented a method for alleviating the stringent CFL condition imposed by the sound speed in highly non-linear compressible flow simulations. A fractional step procedure combined with the pressure evolution equation is used. The method works for arbitrary equations of state, and in the limit as the sound speed goes to infinity it yields the Poisson equation for incompressible flow. We also presented a Mach-ENO or MENO scheme which better utilizes a dual cell center and MAC grid formulation. The numerical experiments on various benchmark problems for one and two dimensions indicate that our semi-implicit method obtains well resolved shock, rarefaction and contact solutions. Since our method is conservative, we also obtain correct shock speeds. The smooth flow example illustrates the ability of our method to take significantly large time steps for low Mach number flows as compared to explicit methods. In future work we plan to extend our approach to handle two-way coupling between compressible and incompressible flows, as well as fully implicit solid-fluid coupling.

Appendix: Boundary Conditions

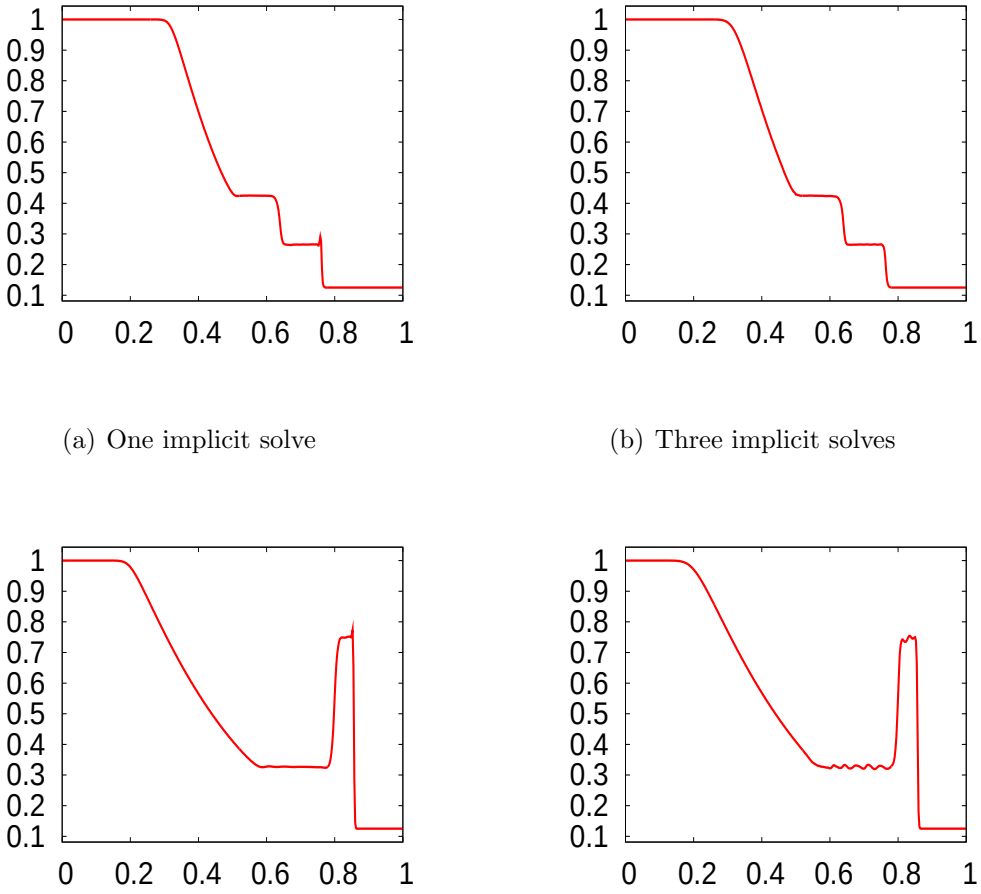
Figure 2.14 requires the handling of inflow and outflow boundary conditions. We define U_{out} to be the outgoing state and U_{in} to be the ingoing state. The outgoing state, U_{out}, is obtained by simple extrapolation whereas the ingoing state, U_{in}, is obtained by attenuating U_{out} towards specified far-field values. After defining U_{out} via extrapolation, we average the primitive variables to cell flux on the boundary of the domain, and use those values to compute a characteristic decomposition. If the p^{th} characteristic field indicates ingoing information, then when applying the ENO scheme in this characteristic field we use U_{in} for the ghost node values. Otherwise U_{out} is used. Note for higher order schemes boundary values will be needed for fluxes on the interior of the domain as well, and we choose the ghost nodes (as U_{in} or U_{out})

in the same fashion.

Our ingoing state, U_{in}, is obtained by attenuating the extrapolated state, U_{out}, towards a given far field state, U_{far}. This is accomplished by multiplying U_{out} with each of the left eigenvectors, attenuating if the eigenvalue in that characteristic field indicates an ingoing wave, and then multiplying by the right eigenvector. Defining the scalar characteristic information in each field as $\xi^p = L^p U_{out}$, we would attenuate ξ^p towards ξ^p_{far} using the analytic solution of the ODE

$$d\xi/dt = K(\xi - \xi_{far})$$

for time step Δt using initial data of $\xi = \xi_{out}$. We used an attenuation coefficient of $K = -.5$ in our examples.



(a) One implicit solve

(b) Three implicit solves

(c) One implicit solve

(d) Three implicit solves

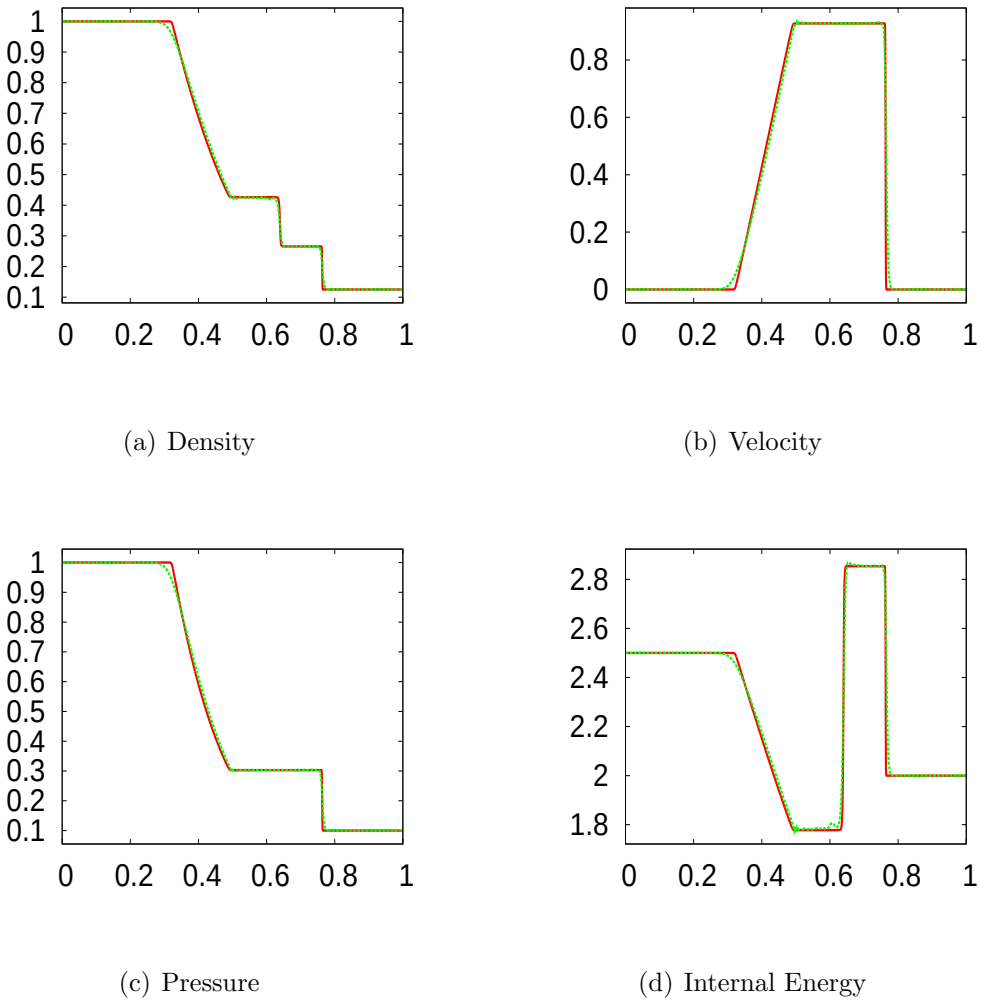
Figure 2.4: Numerical results comparing placing the implicit solve either inside each Runge-Kutta stage (b and d) or once after a full three stage Runge-Kutta cycle (a and c). The top two figures show the results for a Sod shock tube problem at $t = .15s$, the bottom two figures show the results for a strong shock tube problem at $t = 2.5 \times 10^{-6}s$. Density is shown in all figures. Note the spurious overshoots when the implicit solve is not included in the Runge-Kutta cycle (left two figures). Note that we use the standard ENO scheme from [48] (not MENO) for these four examples.

Test name	semi-implicit (seconds)	explicit (seconds)
Sod shock tube	2.95	3.69
Lax shock tube	2.71	4.53
Strong shock tube	2.43	3.43
Mach 3 shock test	2.90	3.59
High Mach flow test	3.75	3.29
Interaction of blast waves (Bang Bang)	5.28	9.86
Two symmetric rarefaction waves	3.52	4.15

Table 2.1: Wall clock times comparing the semi-implicit method with the fully explicit method, for 1-D examples. Simulations were run to the target times of each example as mentioned in their respective figures.

Grid Resolution	Effective sound speed CFL	Δt	Wall clock time (Implicit)	Wall clock time (Explicit)
3200	3	5.01e-08	63.41s	511.67s
32000	30	5.01e-08	810.03s	60498.49s
320000	300	5.01e-08	9976.58s	Impractical

Table 2.2: Timing results for smooth flow test, with Δt approximately constant. The wall clock times are shown for simulations till $t = 5 \times 10^{-5} s$.



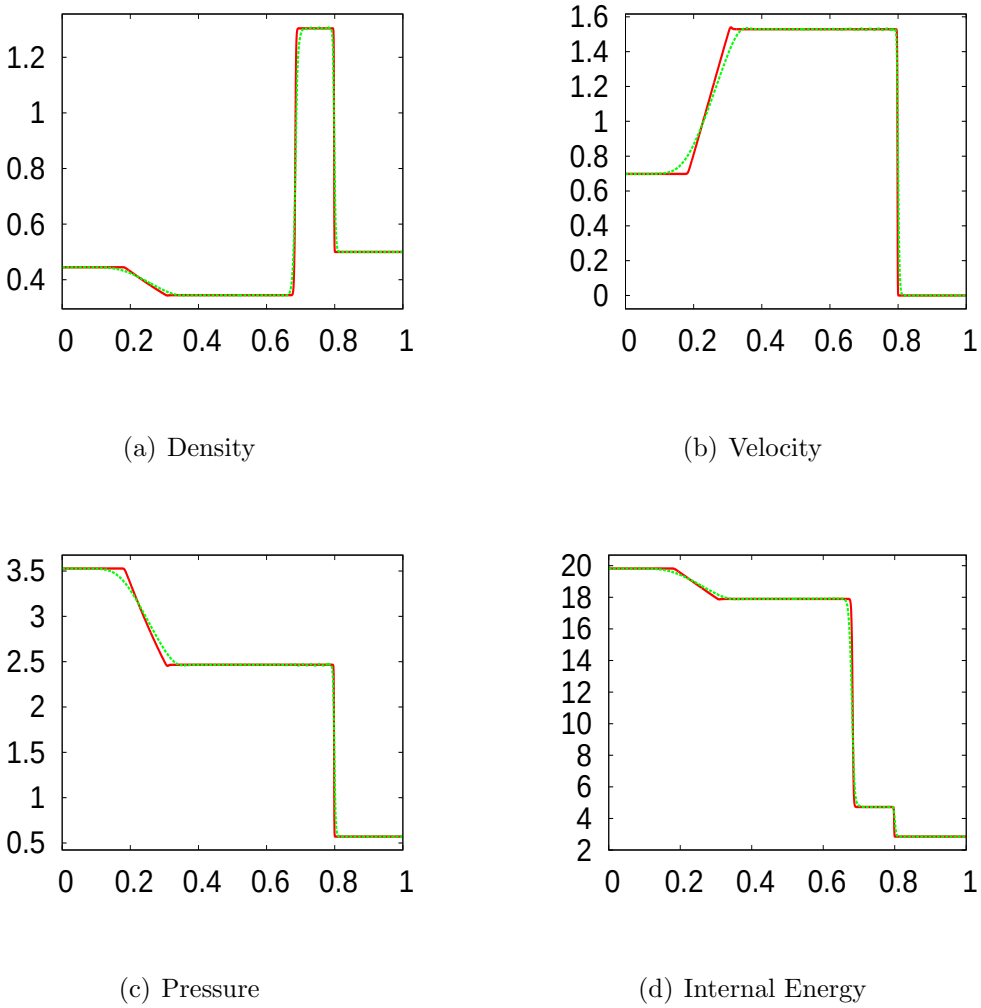
(a) Density

(b) Velocity

(c) Pressure

(d) Internal Energy

Figure 2.5: Numerical results of the Sod shock tube problem at $t = .15s$. The explicit baseline solution is plotted in red, and the solution from our method is plotted in dotted green.



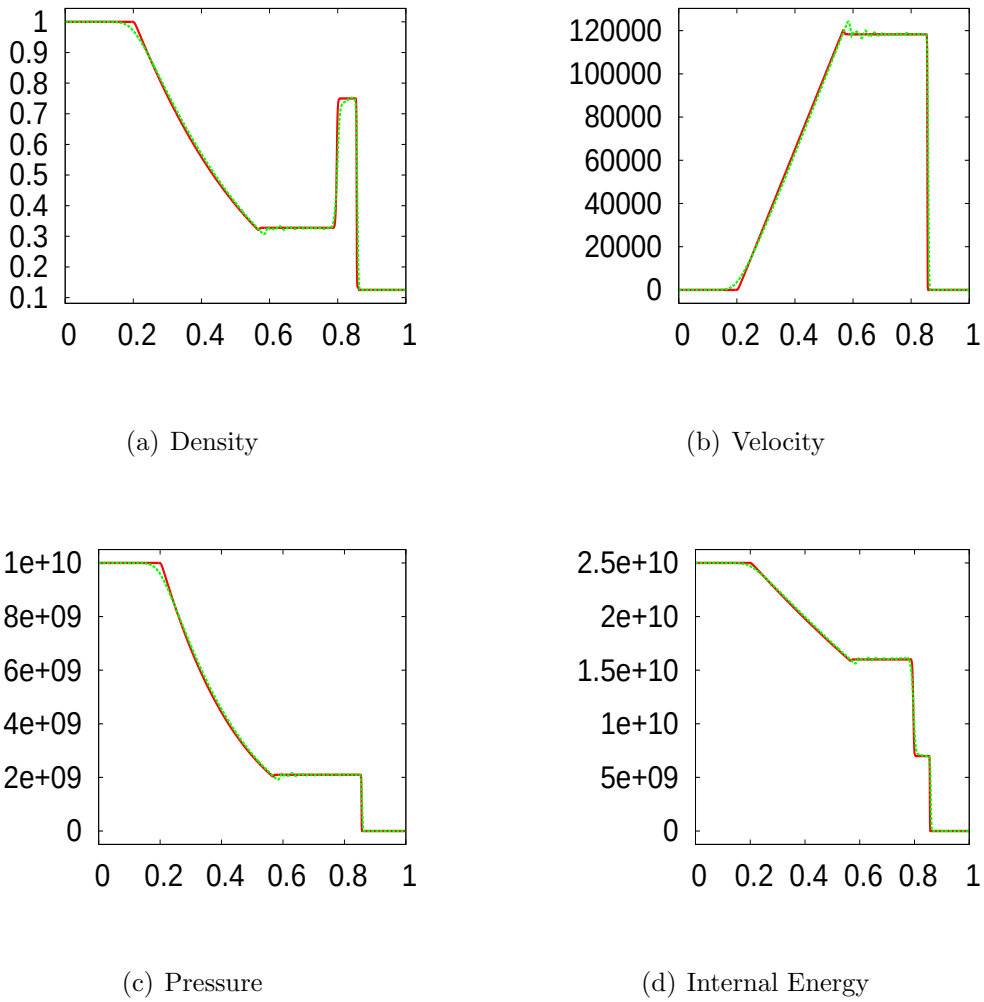
(a) Density

(b) Velocity

(c) Pressure

(d) Internal Energy

Figure 2.6: Numerical results of the Lax's shock tube problem at $t = .12s$. The explicit baseline solution is plotted in red, and the solution from our method is plotted in dotted green.



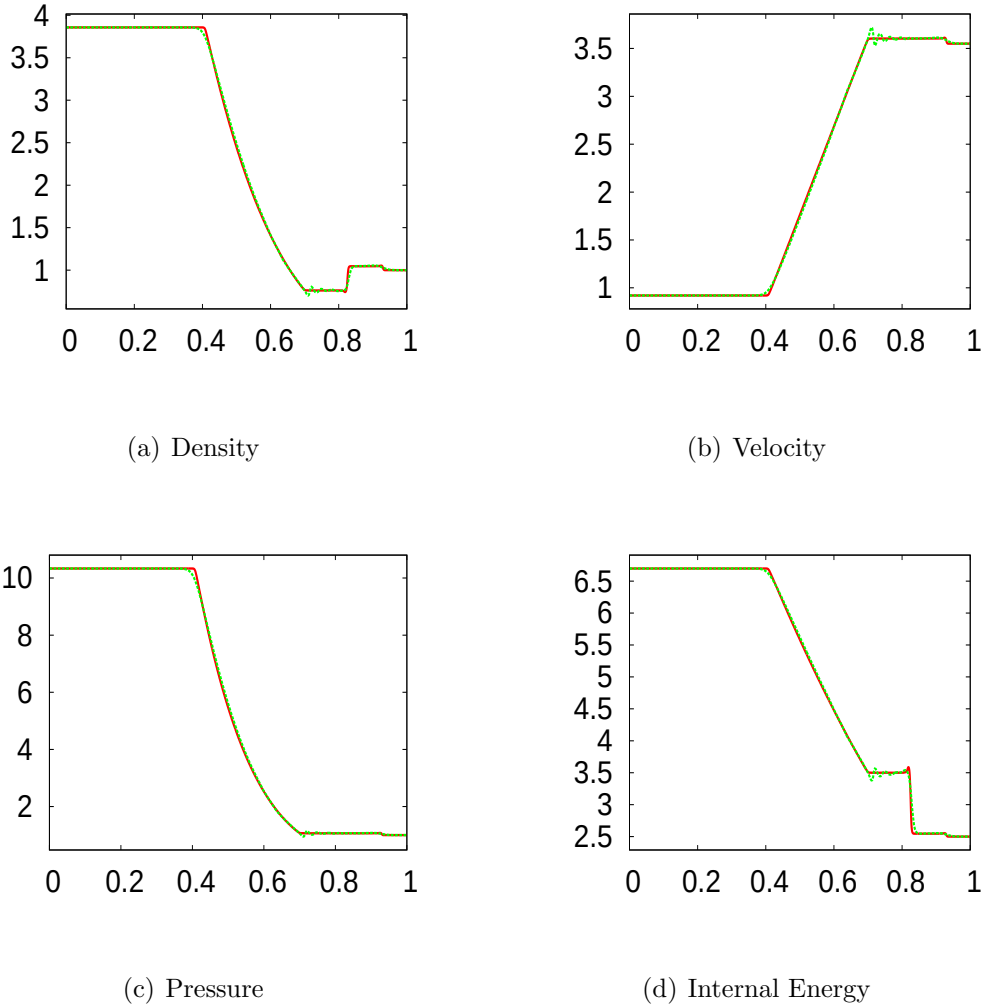
(a) Density

(b) Velocity

(c) Pressure

(d) Internal Energy

Figure 2.7: Numerical results of the strong shock tube problem at $t = 2.5 \times 10^{-6}s$. The explicit baseline solution is plotted in red, and the solution from our method is plotted in dotted green.



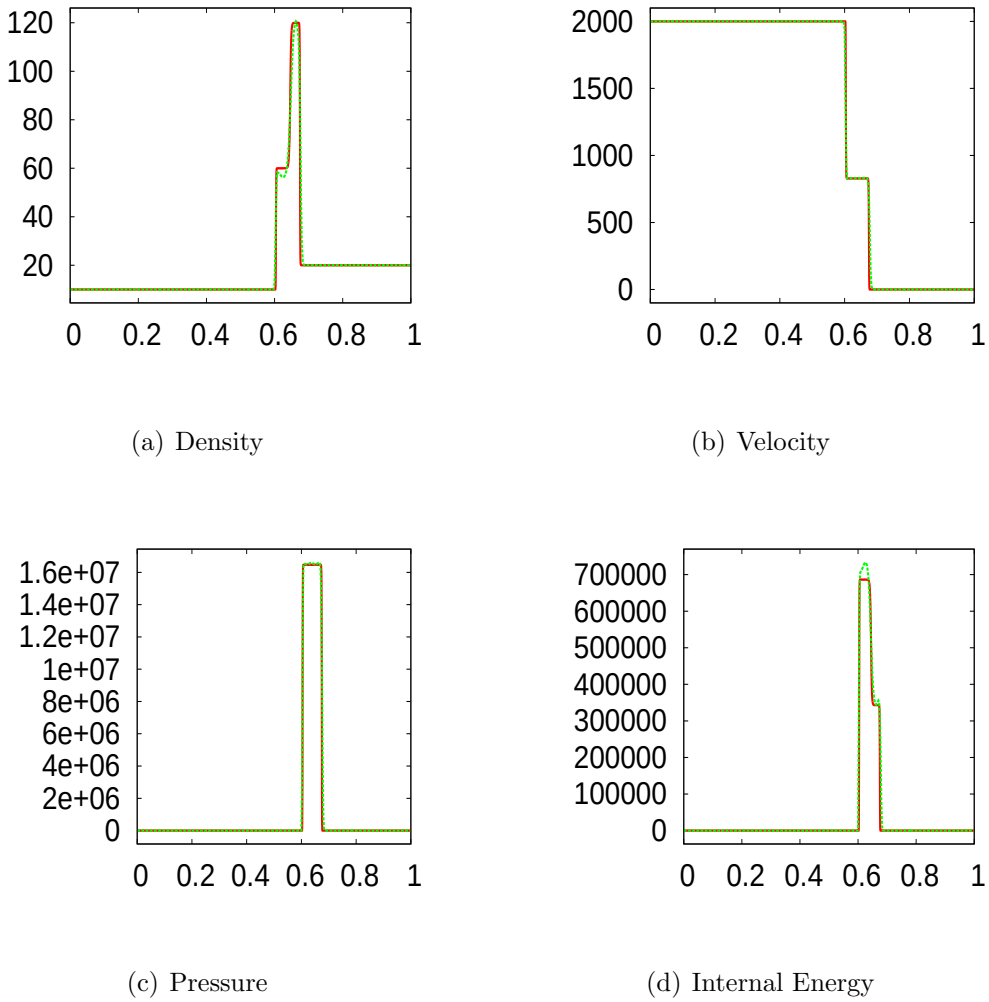
(a) Density

(b) Velocity

(c) Pressure

(d) Internal Energy

Figure 2.8: Numerical results of the Mach 3 shock tube problem at $t = .09s$. The explicit baseline solution is plotted in red, and the solution from our method is plotted in dotted green.



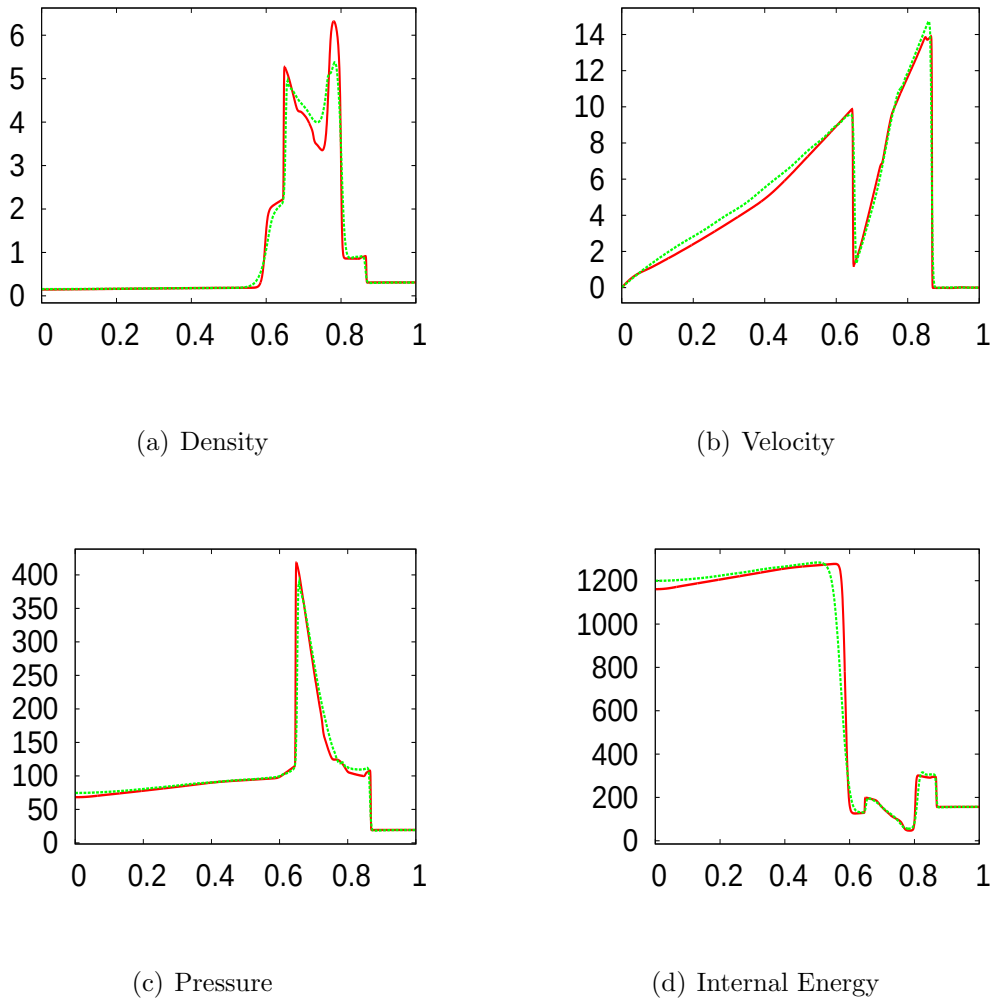
(a) Density

(b) Velocity

(c) Pressure

(d) Internal Energy

Figure 2.9: Numerical results of the High Mach shock tube problem at $t = 1.75 \times 10^{-4}s$. The explicit baseline solution is plotted in red, and the solution from our method is plotted in dotted green.



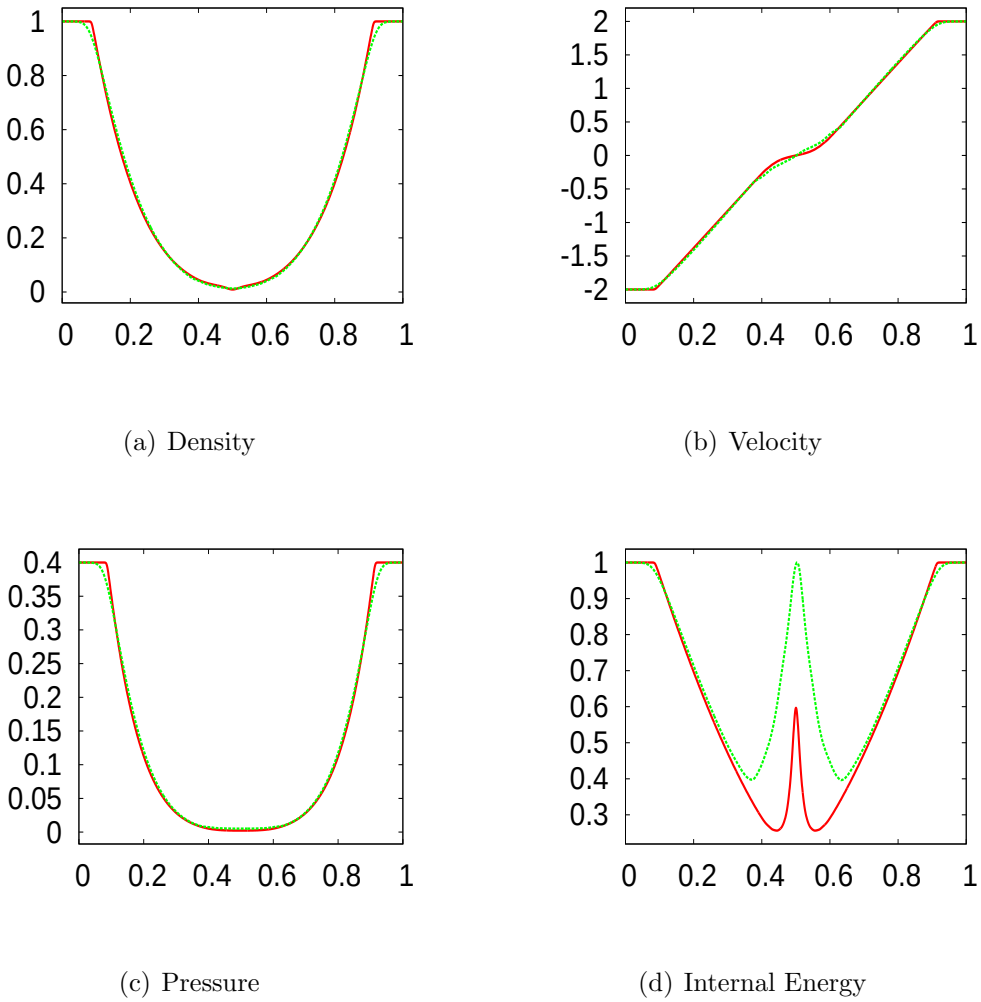
(a) Density

(b) Velocity

(c) Pressure

(d) Internal Energy

Figure 2.10: Numerical results of the interacting blasts shock tube problem at $t = .038s$. The explicit baseline solution is plotted in red, and the solution from our method is plotted in dotted green.



(a) Density

(b) Velocity

(c) Pressure

(d) Internal Energy

Figure 2.11: Numerical results of the symmetric rarefaction shock tube problem at $t = .15s$. The explicit baseline solution is plotted in red, and the solution from our method is plotted in dotted green.

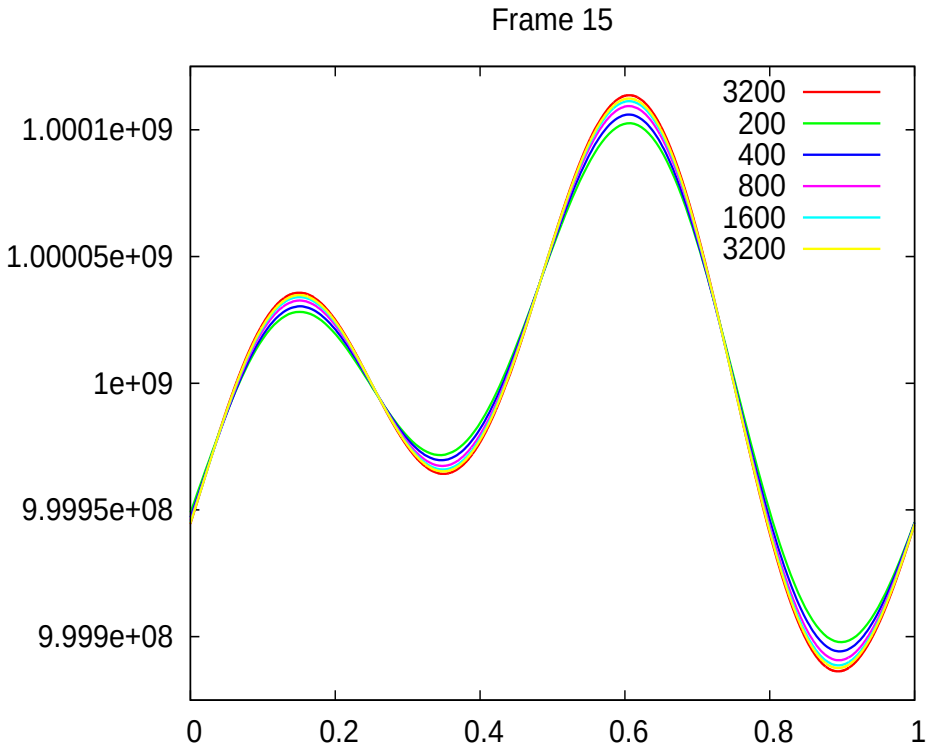


Figure 2.12: Numerical results comparing the pressure in smooth flow test at 200, 400, 800, 1600, and 3200 grid cells with an effective sound speed based CFL number 3 at $t = 1.5 \times 10^{-5}s$. The red curve is the explicit simulation run at 3200 grid cells with a CFL number .5.

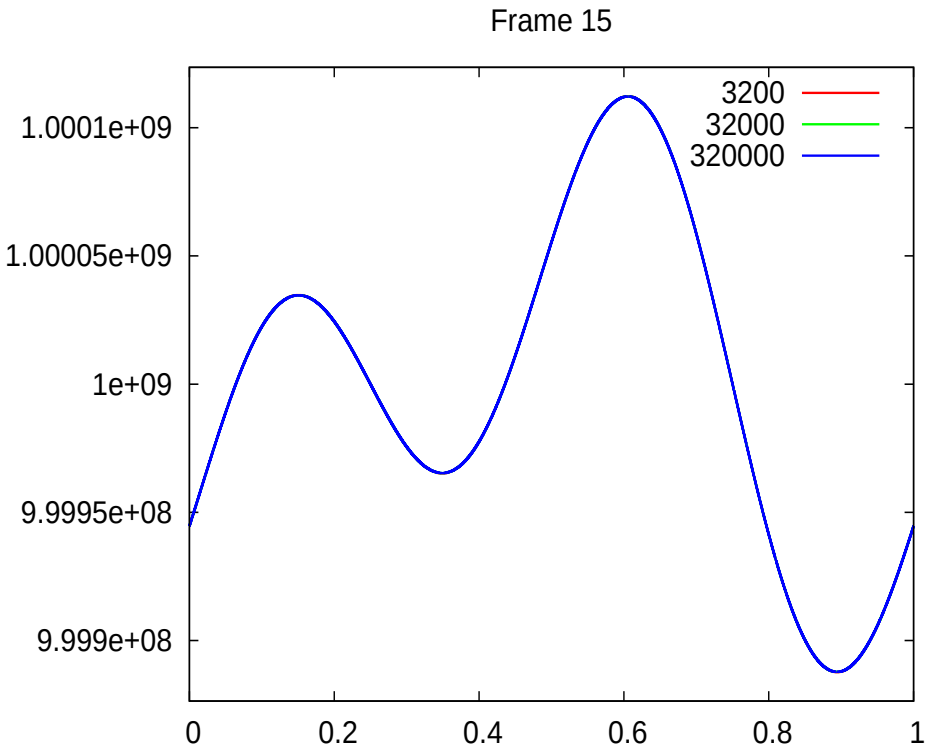


Figure 2.13: Numerical results showing pressure in the smooth flow test at 3200, 32000 and 320000 grid cells. We used an effective sound speed based CFL number of 3, 30 and 300 respectively at $t = 1.5 \times 10^{-5}s$. Since Δt stays constant, the solution remains relatively unchanged even as we get huge time step gains.

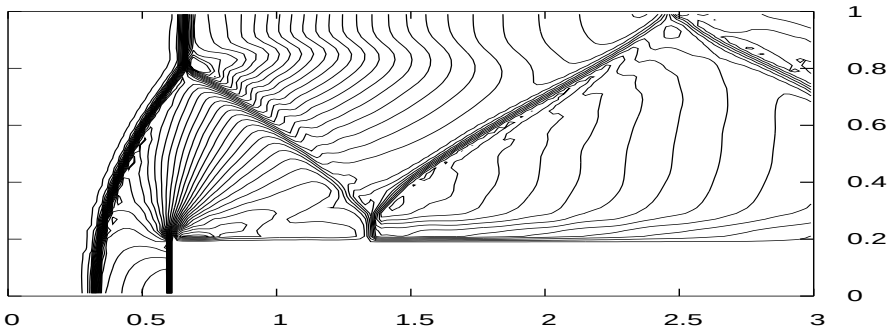


Figure 2.14: Numerical results showing the contour plots of density for the flow past a step test on a grid of size 120x40 at $t = 4s$. 30 contours are plotted in the range $[.2568, 6.067]$.

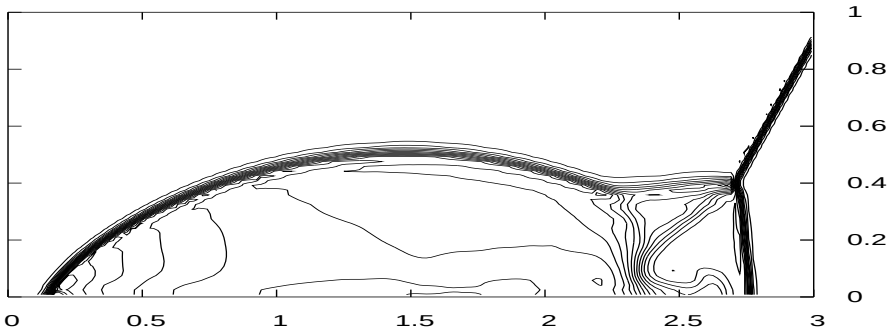
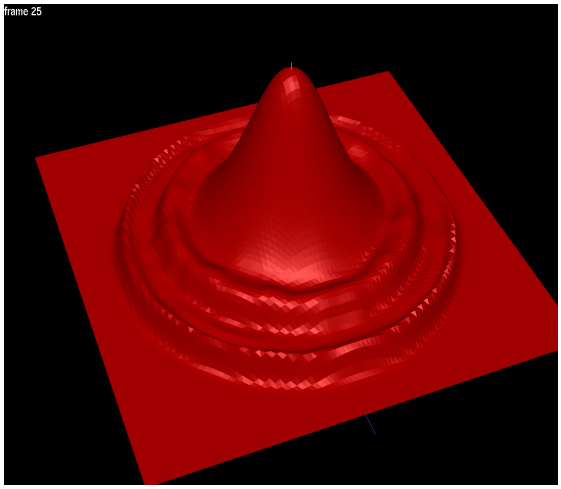
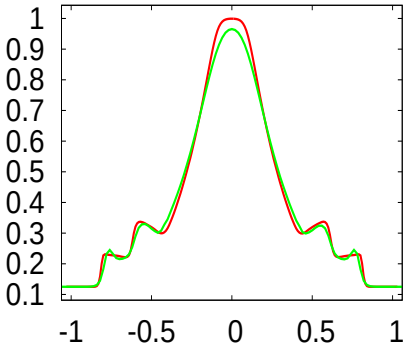


Figure 2.15: Numerical results showing the contour plots of density for the double mach reflection of a strong shock on a grid of size 240x60 at $t = .2s$. 30 contours are plotted within the range $[1.731, 20.92]$.



(a) Density



(b) 1-D slice of density, with the reference solution shown in red.

Figure 2.16: Numerical results for the circular shock test on a grid of size 100x100 at $t = .25s$.

Chapter 3

Implicit Compressible Flow and Structure Coupling

We propose a novel method to implicitly two-way couple Eulerian compressible flow to volumetric Lagrangian solids. The method works for both deformable and rigid solids and for arbitrary equations of state. The method exploits the semi-implicit formulation of compressible flow described in chapter 2, solving for the advection part explicitly and then correcting the intermediate state to time t^{n+1} using an implicit pressure, obtained by solving a modified Poisson system. Similar to previous fluid-structure interaction methods, we apply pressure forces to the solid and enforce a velocity boundary condition on the fluid in order to satisfy a no-slip constraint. Unlike previous methods, however, we apply these coupled interactions implicitly by adding the constraint to the pressure system and combining it with any implicit solid forces in order to obtain a strongly coupled, symmetric indefinite system (similar to [42], which only handles incompressible flow). We also show that, under a few reasonable assumptions, this system can be made symmetric positive-definite by following the methodology of [41]. Because our method handles the fluid-structure interactions implicitly, we avoid introducing any new time step restrictions and obtain stable results even for high density-to-mass ratios, where explicit methods struggle or fail. We exactly conserve momentum and kinetic energy (thermal fluid-structure interactions are not considered) at the fluid-structure interface, and hence naturally handle highly

non-linear phenomenon such as shocks, contacts and rarefactions.

3.1 Introduction

Direct numerical simulations (DNS) are often used to study the interactions between fluid flows and solid structural models. Under certain assumptions these can be reduced to a one-way coupled system; for example if one wishes to determine the steady-state lift of an airfoil in subsonic flow, it is often reasonable to simulate the airfoil as a kinematic body. With a clever choice of boundary conditions, one can even begin to examine two-way coupled interactions, albeit in a limited fashion. In the more general case, these assumptions miss the interesting two-way coupled interactions between the fluid and the structure. These two-way coupled interactions can be quite important and, if not properly captured in the DNS, can lead to non-physical results. It is therefore important to have a robust numerical method that accurately captures two-way coupled interactions across a fluid-structure interface.

Methods to capture fluid-structure interactions can be broadly separated into two categories. Weakly coupled (partitioned) systems interleave the disparate subsystems by integrating them forward in time separately, using each others' results as boundary conditions in an alternating one-way coupled fashion (see e.g. [61, 39, 8]). This approach is appealing as it permits the use of specialized numerical methods for each of the different materials with only slight modifications to account for the modified time integration and changing boundaries. There are disadvantages to this approach, however, for example new and poorly understood stability restrictions arise independent of the individual subsystems, such as the lumped-mass instability discussed in [4]. The alternative is to employ a strongly coupled (monolithic) system, which are systems where the fluid and structure are evolved forward in time simultaneously using a solver specially crafted to incorporate phenomena from both fluid and solid phases. Our method is a hybrid of the two; the explicit components of both fluid and solid solvers are evolved forward independently, while the implicit components and interactions are coupled together in a monolithic solve.

State-of-the-art solvers typically use an Eulerian framework to treat fluid flows

and a Lagrangian framework to treat solids, and so any coupled system must do one of three things: model the solid in an Eulerian framework, model the fluid in a Lagrangian framework, or find a way to couple Eulerian fluids with Lagrangian solids. The first two options are undesirable as they impose significant limitations on the numerical method, for example Eulerian models only capture material properties (rather than tracking them) which makes it difficult to compute time history variables important to structural simulation, such as loading and damage. Many fluid Lagrangian models have difficulty in obtaining the correct shock speeds due to the lack of discrete flux differencing, and therefore resort to artificial viscosity methods that require a number of zones within a shock in order to obtain the right speed [2, 3]. Lagrangian fluid models also struggle with high-speed and deforming flows, as large deformations can cause significant numerical errors in the flow field and can drive the time step to zero. This can be partially alleviated by applying complex and expensive remeshing, but if the flow field tangles and inverts, the simulation can cease altogether. Arbitrary Lagrange-Eulerian (ALE) methods address the problem of a deforming Lagrangian fluid grid by permitting the fluid grid to move at some velocity other than the velocity of the fluid, but this can still lead to high aspect ratios that necessitate remeshing, especially in the presence of a fluid-structure interface. We address the challenge of coupling Eulerian fluids with Lagrangian solids by introducing an interpolation operator, which conservatively maps quantities from Eulerian boundaries to nearby Lagrangian boundary nodes, and vice versa.

At the fluid-structure interface there is a transfer of information. This information transfer can be handled by weakly coupling each separate subsystem using a one-sided estimate of the transfer, or by strongly coupling subsystems together and introducing new variables to the equations. Weakly coupled approaches have been shown to give high-fidelity results [1, 10, 9], but can struggle when applied to a system with high density-to-mass ratios (and are prone to going unstable, as we discuss in Section 3.4.3). These problems can be alleviated by using a better estimate of values at the interface, as suggested by [32], but this typically involves solving expensive general Riemann problems at every fluid-structure face. These problems can be avoided entirely by handling the interface in a strongly coupled fashion, but previous work has

been limited to incompressible flows [38, 42]. Our method exploits the semi-implicit formulation of compressible flow, which treats the pressure flux of compressible flows implicitly. This permits us to treat the fluid pressure as an implicit force on the solid, and use an implicit velocity boundary condition on the Poisson solve, much like previous strongly-coupled work.

Our fluid evolution is comprised of two steps: an advection stage and a pressure solver phase. This permits us to address the complexities arising from the truly non-linear components of the flow separately from the linearly degenerate components. In the pressure phase, we freeze everything to their time t^{n+1} location and perform an implicit solve for the fluid pressure and solid velocity. It is in this phase that we handle the transfer of momentum and kinetic energy across the fluid-structure interface, and as such it is important to be conservative in transferring information between the two sets of degrees of freedom. In the advection stage no information should be transmitted across the interface, but instead we must address the issues which arise by virtue of a moving solid (i.e. the covering and uncovering of fluid cells). There are many examples of how to address these problems in the literature, for example we could track cut cells, re-discretize the fluid in an ALE formulation—all of which significantly complicate the fluid evolution. Instead we make the key observation that since the interface is a contact discontinuity we can afford to be non-conservative, but only in the linearly degenerate components of the flow.

In a traditional explicit method the linearly degenerate and truly non-linear fluxes aren't separated, and as such these methods need to deal with all of the complexities of moving boundaries and information transmission at the same time. That is, they need to be conservative when dealing with information that crosses the interface while at the same time dealing with an interface that moves. Finally, the flux needs to be re-examined carefully in order to determine what forces should be applied to the interface. One could modify traditional methods by separating the conserved quantities into their Riemann invariants, and be conservative in the truly non-linear invariants while allowing the linearly degenerate invariants to be non-conservative —however this doesn't address the moving boundary, and still leaves us with the

(poorly-understood) CFL restriction that arises from explicit fluid-structure interactions. Because of these complications, our method hinges on the existence of the semi-implicit formulation of compressible flow.

3.2 Semi-implicit compressible flow

We quickly re-derive the equations for semi-implicit evolution of compressible flow as described in 2. The derivation philosopy is slightly different here from that in chapter 2 as we want to focus on time $n+1$ face velocities in order to couple with structures. Consider the multi-dimensional Navier-Stokes equations, given by:

$$\begin{pmatrix} \rho \\ \rho\vec{u} \\ E \end{pmatrix}_t + \begin{pmatrix} \nabla \cdot \rho\vec{u} \\ \nabla \cdot (\rho\vec{u})\vec{u} \\ \nabla \cdot (E\vec{u}) \end{pmatrix} + \begin{pmatrix} 0 \\ \nabla p \\ \nabla \cdot (p\vec{u}) \end{pmatrix} = \mathbf{f} \tag{3.1}$$

where we have split the flux terms into an advection and non-advection part and lumped viscous terms into $\mathbf{f}$. The advection part (as well as any body forces) is integrated explicitly to give intermediate values $\rho^\star$, $(\rho\vec{u})^\star$ and $E^\star$. Since pressure does not affect the continuity equation, $\rho^{n+1} = \rho^\star$. The momentum update equation can be divided by ρ^{n+1} to obtain

$$\vec{u}^{n+1} = \vec{u}^\star - \Delta t \frac{\nabla p}{\rho^{n+1}}, \tag{3.2}$$

and taking its divergence gives

$$\nabla \cdot \vec{u}^{n+1} = \nabla \cdot \vec{u}^\star - \Delta t \nabla \cdot \left(\frac{\nabla p}{\rho^{n+1}} \right). \tag{3.3}$$

In the case of incompressible flow, we would set $\nabla \cdot \vec{u}^{n+1} = 0$, but for compressible flow we instead use the pressure evolution equation (see e.g. [11]),

$$p_t + \vec{u} \cdot \nabla p = -\rho c^2 \nabla \cdot \vec{u}. \tag{3.4}$$

If we fix $\nabla \cdot \vec{u}$ to be at time t^{n+1} through the time step (making an $\mathcal{O}(\Delta t)$ error), discretize $p_t + \vec{u} \cdot \nabla p$ explicitly using a forward Euler time step (i.e. $\frac{p^{n+1}-p^n}{\Delta t} + \vec{u}^n \cdot \nabla p^n$), and define the advected pressure as $p^a = p^n - \Delta t(\vec{u}^n \cdot \nabla p^n)$ we obtain

$$p^{n+1} = p^a - \Delta t \rho c^2 \nabla \cdot \vec{u}^{n+1}. \tag{3.5}$$

Substituting this in Equation (3.3) and rearranging gives

$$p^{n+1} - \rho^n (c^2)^n \Delta t^2 \nabla \cdot \left(\frac{\nabla p^{n+1}}{\rho^{n+1}} \right) = p^a - \rho^n (c^2)^n \Delta t \nabla \cdot \vec{u}^\star, \tag{3.6}$$

where we have defined ρc^2 at time t^n and the pressure p at time t^{n+1}. Discretizing the gradient and divergence operators yields

$$\left[I + \rho^n (c^2)^n \Delta t^2 G^T \left(\frac{1}{\hat{\rho}^{n+1}} G \right) \right] p^{n+1} = p^a + \rho^n (c^2)^n \Delta t G^T \vec{\hat{u}}^\star, \tag{3.7}$$

where G is our discretized gradient operator, $-G^T$ is our discretized divergence operator, and $\hat{\rho}$ and $\hat{u}$ represent variables interpolated to cell faces. This is solved to obtain p^{n+1} at cell centers. The time t^{n+1} pressures are then applied in a flux-based manner to the intermediate momentum and energy values to obtain time t^{n+1} quantities in a discretely conservative manner (thereby giving correct shock speeds). For more details see chapter 2.

3.3 Solid evolution

We give a brief treatment of solid evolution with sufficient detail to properly handle the fluid-structure interactions. A solid state is completely described by its velocity and position. We update the position and velocities in a Newmark scheme in which velocity at time $t^{n+1/2}$ is used to update the position to time t^{n+1} in a second order update. Velocity is then updated from time t^n to time t^{n+1} in a separate step. We describe below the velocity update for deformable and rigid solids. The same procedure is used twice, once with a time step of $\Delta t/2$ to obtain $V^{n+1/2}$ for position update and then with a time step of Δt for the final velocity update.

Deformable body formulation: For deformable body evolution we need to handle both elastic and damping forces. Damping forces can impose strict time step restrictions and are thus treated implicitly. We will describe a method which treats the elastic forces explicitly and damping forces implicitly although one could also incorporate implicit elasticity. The deformable body at a given time t can be described by a vector of positions of its nodes $X_s(t)$ and a vector of velocities of its nodes $V_s(t)$. The evolution of velocities can be described by Newton's second law as

$$M_s(V_s)_t = F(X_s, V_s), \tag{3.8}$$

where M_s is the mass matrix and F is the vector of all forces acting on the solid nodes. Discretizing and computing the elastic terms explicitly and damping terms explicit in position, but implicit in velocity, i.e. $F(X_s, V_s) = F(X_s^n, V_s^{n+1})$, we obtain

$$M_s V_s^{n+1} = M_s V_s^n + \Delta t F(X_s^n, V_s^{n+1}). \tag{3.9}$$

Using a Taylor series expansion on F yields

$$M_s V_s^{n+1} = M_s V_s^n + \Delta t(F(X_s^n, V_s^n) + D(V_s^{n+1} - V_s^n)). \tag{3.10}$$

where $D = \frac{\partial F}{\partial V_s}$. $F(X_s^n, V_s^n) - DV_s^n$ represents the elastic only (and, if present, any non-linear damping terms [49]) component of the force and one can write

$$M_s V_s^{n+1} = M_s V_s^\star + \Delta t D V_s^{n+1}, \tag{3.11}$$

where $V_s^\star$ denotes the velocity vector updated explicitly with the elastic terms only.
Rigid body formulation: For a rigid body we define the generalized velocity vector as $V_s = (V_{cm}^T, \omega^T)^T$, where V_{cm} is the velocity of its center of mass and ω is its angular velocity. The velocity evolution can then be described as

$$\begin{pmatrix} M_r & 0 \\ 0 & I_r \end{pmatrix} (V_s)_t = \begin{pmatrix} f \\ \tau \end{pmatrix}, \tag{3.12}$$

where M_r is a 3×3 diagonal matrix with the rigid body mass in the diagonals, I_r is the inertia tensor and f, τ are the net force and torque acting on it. Writing the mass matrix as M_s and combining f, τ into F, we get a form similar to (3.8) which can be discretized using forward Euler to obtain

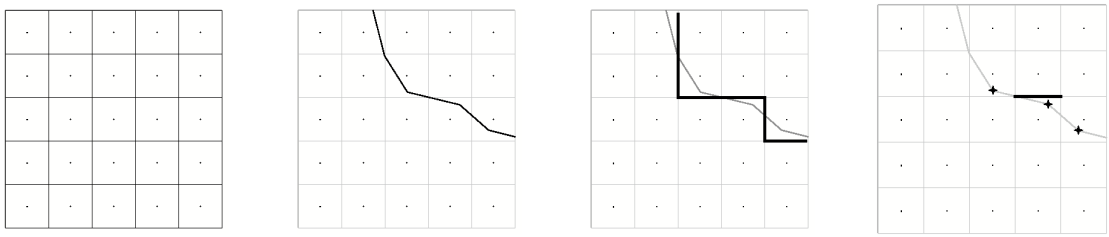
$$M_s V_s^{n+1} = M_s V_s^n + \Delta t F^n = M_s V_s^\star. \tag{3.13}$$

Where $V_s^\star$ denotes the velocity vector updated with the explicit forces. Note that this is the same as Equation (3.11) except without any damping term. We will therefore use Equation (3.11) as our general solid update equation below, as it covers both the rigid and deformable cases.

3.4 Fluid-structure interaction

We solve for the fluid on an Eulerian grid, and the solids on freely deforming Lagrangian meshes. The fluid and structure interact with each other by applying equal and opposite forces at the interface, satisfying physical boundary conditions (we use no-slip, no penetration boundary conditions) in the process. Immersed boundary methods induce extra force variables at the interface and apply a regularization operator to map these forces to fluid faces (see e.g. [53]). They also incorporate an interpolation operator to map fluid velocity to solid nodes for applying boundary conditions. We eliminate the extra interface force variables and conservatively map the fluid pressures directly to solid nodes, and solid velocities to fluid faces using an interpolation operator.

Figure 3.1 illustrates an example fluid grid which is coupled to a Lagrangian solid which occupies the upper right-hand corner of the grid. In our model, the fluid interacts with a voxelized version of the solid and the solid directly sees forces acting on its nodes. We define an interpolation operator W which maps solid node velocity to the fluid cell faces, where the rows correspond to fluid faces and the columns to solid nodes. W can be constructed in a row-by-row fashion: for each row, we identify the corresponding fluid face and locate the nearby solid nodes. The entry corresponding



(a) Eulerian fluid grid.

(b) Lagrangian solid which overlaps the fluid domain.

(c) Solid voxelized to fluid faces.

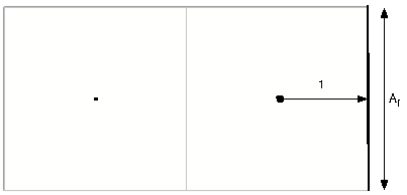
(d) Solid nodes which contribute to the rasterized face.

Figure 3.1: A common challenge with FSI problems is one of overlapping grids. We resolve this issue by voxelizing solid degrees of freedom to the fluid grid using an interpolation operator denoted by the matrix W. The row corresponding to a fluid face gets contributions from nearby solid nodes.

to each solid node is populated by a weight proportional to its contribution to the fluid face, and then finally the row is normalized to ensure that each row sums to one, making it an interpolation. This is done in a component-by-component manner, e.g. the x-component of solid velocity is voxelized to x-axis fluid faces but not y- or z-axis fluid faces, and so the solid velocity at fluid face $i + 1/2$ is $(WV_s)_{i+1/2}$. Since pressure is defined at cell centers, we also introduce an extrapolation operator B which maps cell-centered pressure to face pressures, as illustrated in Figure 3.2. These face pressures are then multiplied by the surface area of the cell face to get a force and distributed back to solid nodes using W^T. That is, W maps from solid node degrees of freedom to cell faces, and W^T maps back in the opposite direction. Note that since the rows of W sum to one, the columns of W^T sum to one and therefore the force felt due to the pressure on the face is fully and conservatively distributed to the solid node degrees of freedom.

3.4.1 The strongly coupled system

The fluid acts on solid degrees of freedom via pressure along the interface. The pressure exerts a force given by $W^T A_f B p$ on the solid degrees of freedom, where A_f is a diagonal matrix whose entries correspond to the areas of fluid-structure faces. We



$$B = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 1 \end{bmatrix}$$

Figure 3.2: Operator B maps pressure from cell centers to bordering fluid-structure faces. In this example there are x-direction faces, of which the one to the far right represents a rasterized solid face. Therefore B has three rows (one for each vertical face, with the top and the bottom rows corresponding to the far left and far right vertical faces respectively, and the middle row corresponding to the middle vertical face), and two columns (one for each pressure at each cell center). Since the only contribution to the solid is from the second pressure to the third face, B has the form shown above with a single non-zero element. Note that $(1/dx)B^T$ equals $-G_s^T$, as defined in Figure 3.3(b).

can incorporate these forces into the implicit solid system given by Equation (3.11):

$$M_s V_s^{n+1} = M_s V_s^\star + \Delta t D V_s^{n+1} + \Delta t W^T A_f B p. \tag{3.14}$$

The fluid sees a velocity boundary condition at the fluid-structure interface. To incorporate this into the fluid equations, we partition the discrete divergence operator $-G^T$ into two components. G_f^T operates over fluid-fluid faces, while G_s^T is the component of the divergence operator which operates on rasterized fluid-structure faces (as outlined in Figure 3.3), and $G^T = G_f^T + G_s^T$. We can then set fluid-structure faces to have implicit Neumann boundary conditions; that is,

$$\vec{u}^{n+1} = \begin{cases} \vec{u}^\star - \Delta t \frac{G_f p}{\hat{\rho}} & \text{at a fluid-fluid face; and} \\ W V_s^{n+1} & \text{at a fluid-structure face.} \end{cases} \tag{3.15}$$

Taking the divergence of the velocity field yields

$$G^T \vec{u}^{n+1} = G_f^T \vec{u}^\star - \Delta t G_f^T \frac{1}{\hat{\rho}} G p + G_s^T W V_s^{n+1} \tag{3.16}$$

Using this modified definition for $G^T\vec{u}^{n+1}$ in Equation (3.5) and substituting into Equation (3.3) gives

$$\left[\frac{1}{\Delta t\rho^n(c^n)^2}I + \Delta t G_f^T\frac{G}{\hat{\rho}^{n+1}}\right]p^{n+1} - G_s^T W V_s^{n+1} = \frac{p^a}{\Delta t\rho c^2} + G_f^T\vec{u}^\star. \tag{3.17}$$

If we define $V = \Delta x\Delta y\Delta z$ to be the volume of the fluid cell, then $VG_s^T = A_f B^T$. Combining equations (3.14) and (3.17), using scaled pressure $\tilde{p} = \Delta t p$ and scaled advected pressure $\tilde{p}^a = \Delta t p^a$, and rescaling the fluid equations by cell volume gives us our symmetric system

$$\begin{pmatrix} \frac{V}{\Delta t^2\rho c^2}I + VG_f^T\frac{1}{\rho}G_f & -A_f B^T W \\ -W^T B A_f & -M_s + \Delta t D \end{pmatrix}\begin{pmatrix}\tilde{p}^{n+1}\\ V_s^{n+1}\end{pmatrix} = \begin{pmatrix}\frac{V}{\Delta t^2\rho c^2}\tilde{p}^a + VG_f^T\vec{u}^\star \\ -M_s V_s^\star\end{pmatrix}. \tag{3.18}$$

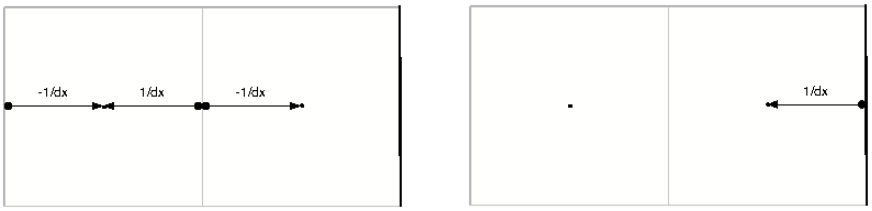
It is interesting to note that if we take the incompressibility assumption (i.e. $c\to\infty$) then this system reduces to one similar to [42].

The system in Equation (3.18) is symmetric but indefinite, and can be solved using efficient solvers such as Conjugate Residuals [34] to obtain the final time t^{n+1} solid velocity and pressure. The solid part of our update is now complete, but we still need to use the t^{n+1} pressure to update the fluid momentum and energy (noting that $\rho^{n+1} = \rho^\star$ is already done).

3.4.2 Updating fluid momentum and energy

To obtain correct shock speeds we use the flux-based method discussed above, with modifications to account for fluid-structure faces. At a fluid-structure face $i+1/2$, the fluid applied a force of $(BA_f p)_{i+1/2}$ to the solid. To conserve momentum, fluid face $i+1/2$ should apply an equal and opposite force $-(BA_f p)_{i+1/2}$ on fluid cell i. In our momentum update this is numerically equivalent to setting $p_{i+1/2} = (Bp)_{i+1/2}$ at fluid-structure faces.

Next, we need to consider the work done by the fluid on the solid at a fluid-structure face. We are applying an impulse $\Delta t(BA_f p)_{i+1/2}$ on the solid, which is equivalent to applying a constant force over the interval Δt. In order to compute the



(a) $G_f^T = \frac{1}{dx}\begin{bmatrix} -1 & 1 & 0 \\ 0 & -1 & 0 \end{bmatrix}$ (b) $G_s^T = \frac{1}{dx}\begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$

Figure 3.3: In our derivation, the divergence operator $-G^T$ is split into G_f^T (which operates only on fluid-fluid faces) and G_s^T (which operates only on fluid-structure faces). We show this splitting for a simple two cell example where the right-most face is a fluid-structure interface. The rows in the above matrices correspond to cells and columns to faces. The left most face corresponds to the first column of G_f^T and only has one non-zero element since it only borders one fluid cell. The middle face (which corresponds to the second column of G_f^T) contributes to both fluid cells and hence has two non-zero elements. The third column of G_f^T is zero, as the third face is a fluid-structure face and instead corresponds to G_s^T. Figure (b) depicts G_s^T, which is defined as $-(1/dx)B^T$ in Figure 3.2.

work done on the solid system by a single force $\vec{f}$ in the presence of other forces, we lump all forces acting on the solid into a vector $\vec{F}$ and examine

$$\int_0^{\Delta t} \vec{f}\cdot V_s(t)dt = \int_0^{\Delta t} \vec{f}\cdot(V_s^n + M_s^{-1}\vec{F}t)dt = \Delta t\vec{f}\cdot\left[V_s^n + M_s^{-1}\vec{F}\frac{\Delta t}{2}\right] = \Delta t\vec{f}\cdot\left[\frac{V_s^n + V_s^{n+1}}{2}\right], \tag{3.19}$$

where we take advantage of $\vec{F}$ and $\vec{f}$ being constant over the interval. We are interested in calculating the work done by a single fluid face on the solid, so if we take $W_{i+1/2}^T$ to be the column vector which distributes the pressure from cell face $i+1/2$ to the solid node degrees of freedom then $\vec{f} = W_{i+1/2}^T(BA_fp)_{i+1/2}$, and the work done on the solid by this face is exactly

$$\Delta t\left[W_{i+1/2}^T(BA_fp)_{i+1/2}\right]^T\left[\frac{V_s^n + V_s^{n+1}}{2}\right] = \Delta t\left[(BA_fp)_{i+1/2}\right]W_{i+1/2}\left[\frac{V_s^n + V_s^{n+1}}{2}\right]. \tag{3.20}$$

This, if $p_{i+1/2}$ is defined to be $(Bp)_{i+1/2}$ as suggested above in the momentum update, then we merely need to set $\vec{u}_{i+1/2} = (1/2)(W[V_s^n + V_s^{n+1}])_{i+1/2}$ in order to obtain a flux $p\vec{u}$ which exactly conserves the kinetic energy transferred.

3.4.3 Time step restriction

In our method fluid-structure interactions are handled implicitly and thus we avoid introducing any new time step restrictions. The time step is therefore determined by the minimum of the time steps imposed by the fluid and the structure. For the structure update the time step restriction is determined by the elastic part only, as damping terms are handled implicitly, while our semi-implicit fluid update imposes a time step restriction dependent only on its bulk velocity. The time step restriction imposed by the semi-implicit flow formulation in two spatial dimensions is described in 2.3. We note that the implicit fluid-structure coupling gives stable results even for very high density-to-mass ratios, where explicit methods struggle even when the CFL restrictions of both solid and fluid systems are obeyed. We explore this in example 3.6.1.

3.5 Unified time integration

We employ a time integration scheme which incorporates fluid evolution into a Newmark-style solid evolution scheme. The scheme works by computing an intermediate velocity for the solid $V_s^{n+1/2}$, and applying this in a second order update to get solid positions at time t^{n+1}. Velocities are then updated from time t^n to t^{n+1} (discarding intermediate values), and so two linear systems are solved.

In order to compute the intermediate solid velocity $V_s^{n+1/2}$, we begin by applying all explicit solid forces to the system, which gives $V_s^{n+1/2\star}$. Explicit body forces such as gravity and viscosity are also applied to the fluid system, yielding $t^{n+1/2\star}$ fluid quantities. The coupled system (3.18) is solved in order to obtain $X_s^{n+1} = X_s^n + \Delta t V_s^{n+1/2}$, and then the entire fluid state and all solid velocities are restored to their time t^n values.

These new positions are then used to compute an *effective* velocity for the solids, i.e. $(X_s^{n+1} - X_s^n)/\Delta t$. Using the effective velocity and then the time t^n position of the solid, we fill ghost cells. These ghost cells are used directly in the stencils of high-order methods, and provide a valid state for which to populate uncovered cells. In order

to compute the ghost cell data at location $\vec{x}_g$, we begin by identifying the closest solid interface point $\vec{x}_I$, and reflecting across the interface. Density and pressure are interpolated to the reflected point $2\vec{x}_I - \vec{x}_g$ from neighboring cells and then copied to the ghost cell. The surface normal $\vec{N}$ at the interface is used to decompose the velocity at the reflected point $\vec{V}_r$ into its normal component $V_{rN} = \vec{V}_r \cdot \vec{N}$ and its tangential component $\vec{V}_{rT} = \vec{V}_r - V_{rN}\vec{N}$. In order to remain continuous with the effective velocity of the structure at the interface $\vec{V}_I$, V_{rN} is reflected across the interface, and so we compute $V_{gN} = 2\vec{V}_I \cdot \vec{N} - V_{rN}$. Tangential velocity is decoupled from the interface and thus we can use it directly, giving the final ghost cell velocity $\vec{V}_g = V_{gN}\vec{N} + \vec{V}_{rT}$.

Once ghost cells are filled, explicit body forces such as gravity and viscosity are integrated into the system, and the advection component of flux from Equation (3.1) is applied using a conservative flux-based method (see 2). Explicit solid forces are applied in order to compute $V_s^{n+1\star}$, and then the coupled system (3.18) is solved to obtain V_s^{n+1} and p^{n+1}. This pressure is applied as per Section 3.4.2 to obtain time t^{n+1} fluid quantities.

We also fill the ghost cells inside the solid using time t^{n+1} data from the fluid and solid velocities, as described above. Although none of our examples use these ghost values, if an explicit body force such as viscosity were to be applied, its stencil would require valid ghost cells to be defined. Note that these are valid as instantaneous ghost cells, whereas the ghost cells above use the *effective* solid velocity, which is the actual motion of the solid through the mesh. Practical experience shows that this can make a meaningful difference.

3.6 Examples and validation

In order to compare our results with previous methods, we implement an explicit coupling scheme which integrates a fully explicit compressible flow evolution with a Newmark time integration for solids. This explicit method proceeds in a fashion similar to Section 3.5, except that instead of solving the system (3.18) we simply fill ghost cells inside the solid once and explicitly evolve the fluid once, while time t^n pressures along the fluid-structure interface are applied to the solid as explicit

forces. This gives us an explicitly coupled time evolution scheme, such as the one described in [9].

Although one might assume that the implicit solve would cause efficiency bottlenecks, we observed relatively few Conjugate Residuals iterations per time step. This is likely due to the strongly diagonally dominant nature of Equation (3.18), and the good initial guess for pressure provided by the equation of state at time t^n. For all of our one dimensional examples the maximum number of iterations required per time step was 3. For the two dimensional examples, the rigid body coupling example required a maximum of 4 iterations, while the deformable coupling example required a up to 24 iterations per time step.

In all of the examples we consider the fluid is simulated using an ideal gas law, with $\gamma = 1.4$.

3.6.1 One-dimensional validation

We examine several one dimensional fluid-structure interactions to validate our method. A third order ENO scheme [48] is used along with an advection-based CFL number of .6. All quantities below are in SI units, with density as kg/m^3, pressure in Pa, lengths in m, spring coefficients in N/m, etc.

Sod shock coupled with a rigid body

Our first example is a Sod shock interacting with a rigid body, with open boundary conditions. The initial condition for the fluid is

$$(\rho(x,0), u(x,0), p(x,0)) = \begin{cases} (1,0,1) & \text{if } x \leq .5, \\ (.125,0,.1) & \text{if } x > .5. \end{cases}$$

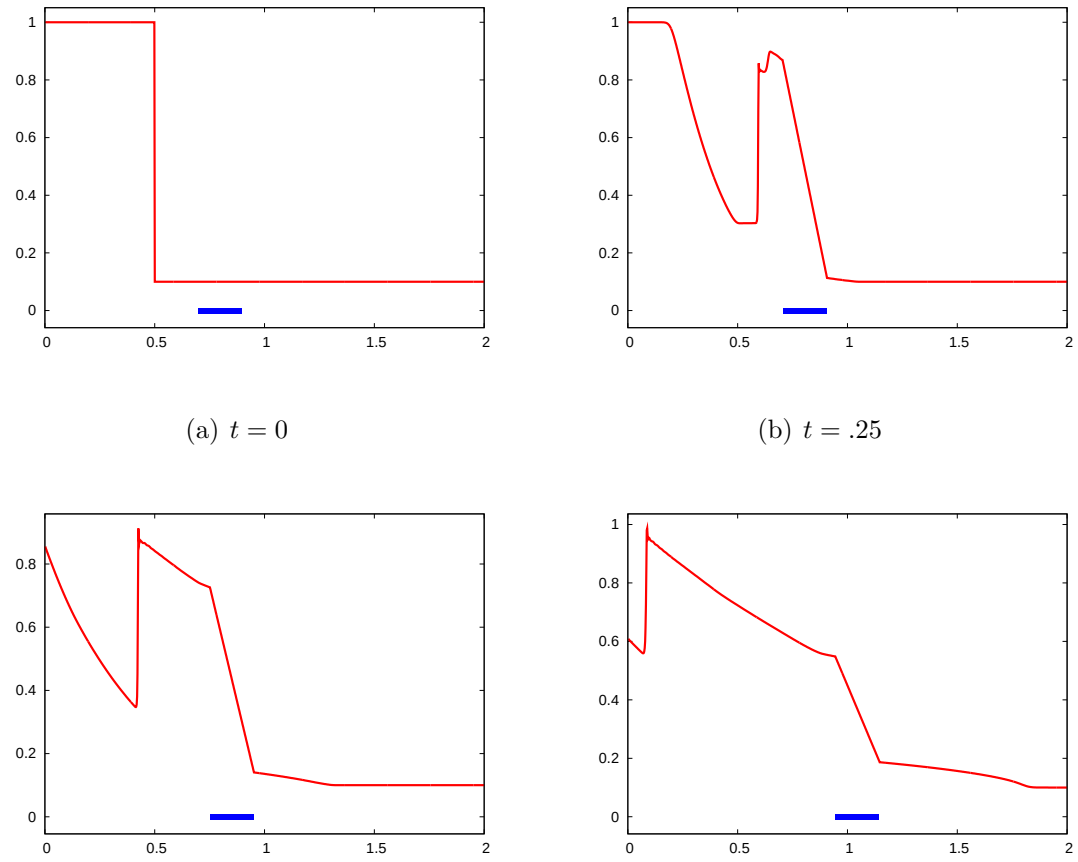
A rigid body of mass 1 and width .2 starts at rest with its center of mass a distance of .8 from the left of the domain. The domain is of length 2. The rigid body remains at rest until the shock hits it, at which point it accelerates by virtue of the pressure difference. The solid body continues to accelerate until it converges to a velocity of .927453, which is precisely the interfacial velocity of the Sod Riemann problem. Figure 3.4

shows snapshots of the pressure profile at various times through the simulation. For comparison, results with the explicit method are shown in Figure 3.5. We also do a convergence analysis of our method in Figure 3.6. The error in the position of the rigid body is computed at time .9 from the highest resolution grid simulated, which is 6401 grid cells. The convergence order of the error is estimated as 1.6.

It is interesting to consider this simple problem for a variety of density-to-mass ratios. Figure 3.7(a) shows the velocity of the rigid body as a function of time for a range of rigid body masses in the semi-implicit case. Figure 3.7(b) shows this in the explicit case. We note that the explicit simulation struggles with high density-mass ratios. In particular it appears as though the rigid body gains too much momentum in a single time step, causing the fluid on the other side to over-compress, leading to a very stiff oscillatory system, even though the time step obeyed CFL restrictions. We show snapshots of the pressure profile of simulations with a light solid of mass .0001, with semi-implicit and explicit schemes in Figure 3.8 and Figure 3.9, respectively.

Sod shock interacting with a fluid piston

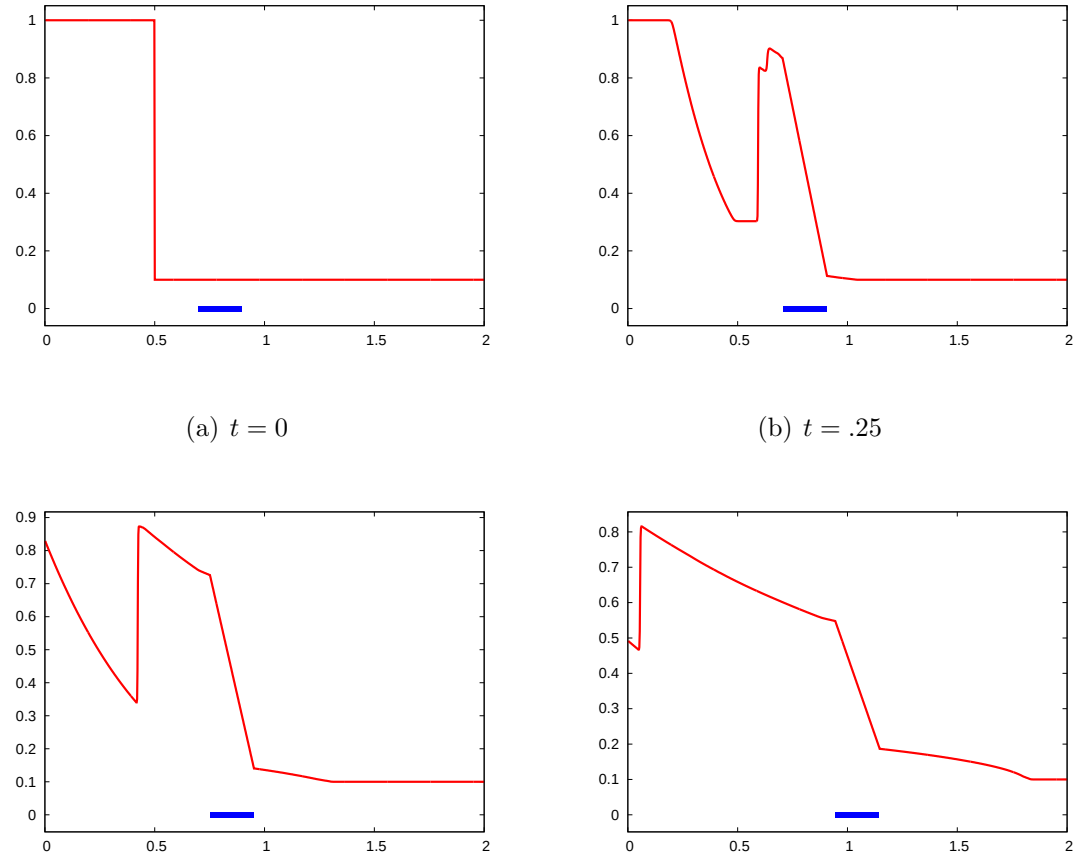
We consider a similar problem, this time with solid wall boundary conditions and a larger domain, with the initial discontinuity located at distance 1 from the left of the domain. The rigid body has a mass of 1, width .2 and starts at rest with its center of mass at 1.5 from the left of the domain. The domain is of length 3. The shock imparts momentum to the rigid body which in turn compresses the fluid on its right. This compressed fluid creates a high pressure region which pushes back on the solid, in effect creating a "fluid spring." This causes the rigid body to oscillate as shown in Figure 3.10, which plots the position of the center of mass of the rigid solid as a function of time. Figure 3.11 shows snapshots of the pressure profile at various times through the simulation. For comparison, results with the explicit method are shown in Figure 3.12. We also do a convergence analysis of our method in Figure 3.13. The error in the position of the rigid body is computed at time $4s$ from the highest resolution grid simulated, which is 6401 grid cells. The convergence order of the error is estimated as 1.03.



(a) $t = 0$ (b) $t = .25$

(c) $t = .5$ (d) $t = 1$

Figure 3.4: Semi-implicit simulation of a Sod shock hitting a rigid body of mass 1. Pressure profile of the fluid is shown at various times through the simulation. The 1-D rigid body is drawn as a blue line segment at the bottom of the plot, with pressure inside the solid shown as a linear pressure profile. The simulation was done on a grid of resolution 1601.



(a) $t = 0$

(b) $t = .25$

(c) $t = .5$

(d) $t = 1$

Figure 3.5: Explicit simulation of a Sod shock hitting a rigid body of mass 1. Pressure profile of the fluid is shown at various times through the simulation. The 1-D rigid body is drawn as a blue line segment at the bottom of the plot, with pressure inside the solid shown as a linear pressure profile. The simulation was done on a grid of resolution 1601.

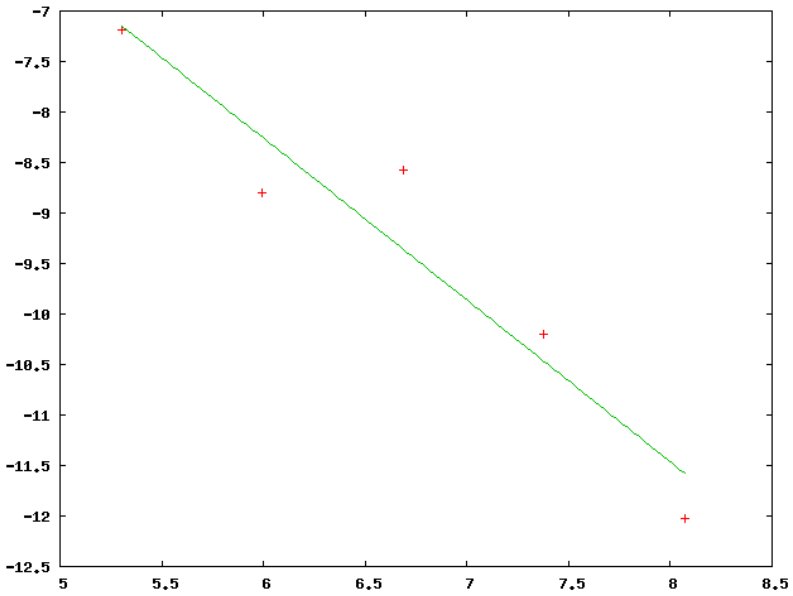
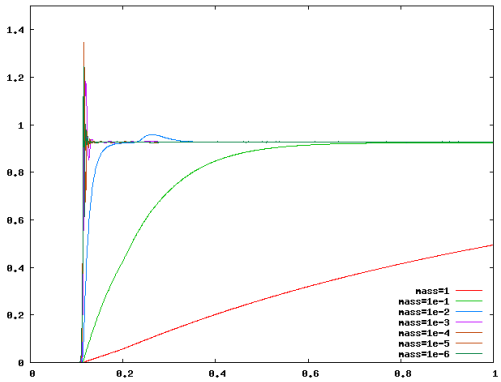
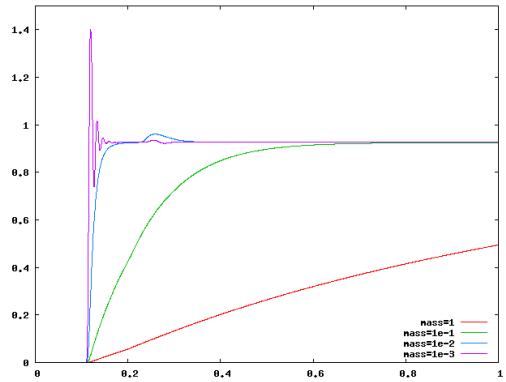


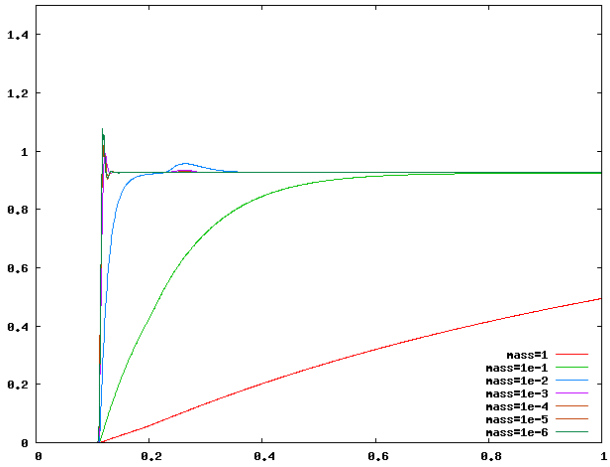
Figure 3.6: Position error of the center of mass of a rigid body hit by a Sod shock, as compared to a high-resolution simulation, at time $.9s$. We plot the log of the relative error, as a function of the log of the resolution of the underlying grid. The convergence rate is 1.6.



(a) Semi-Implicit.

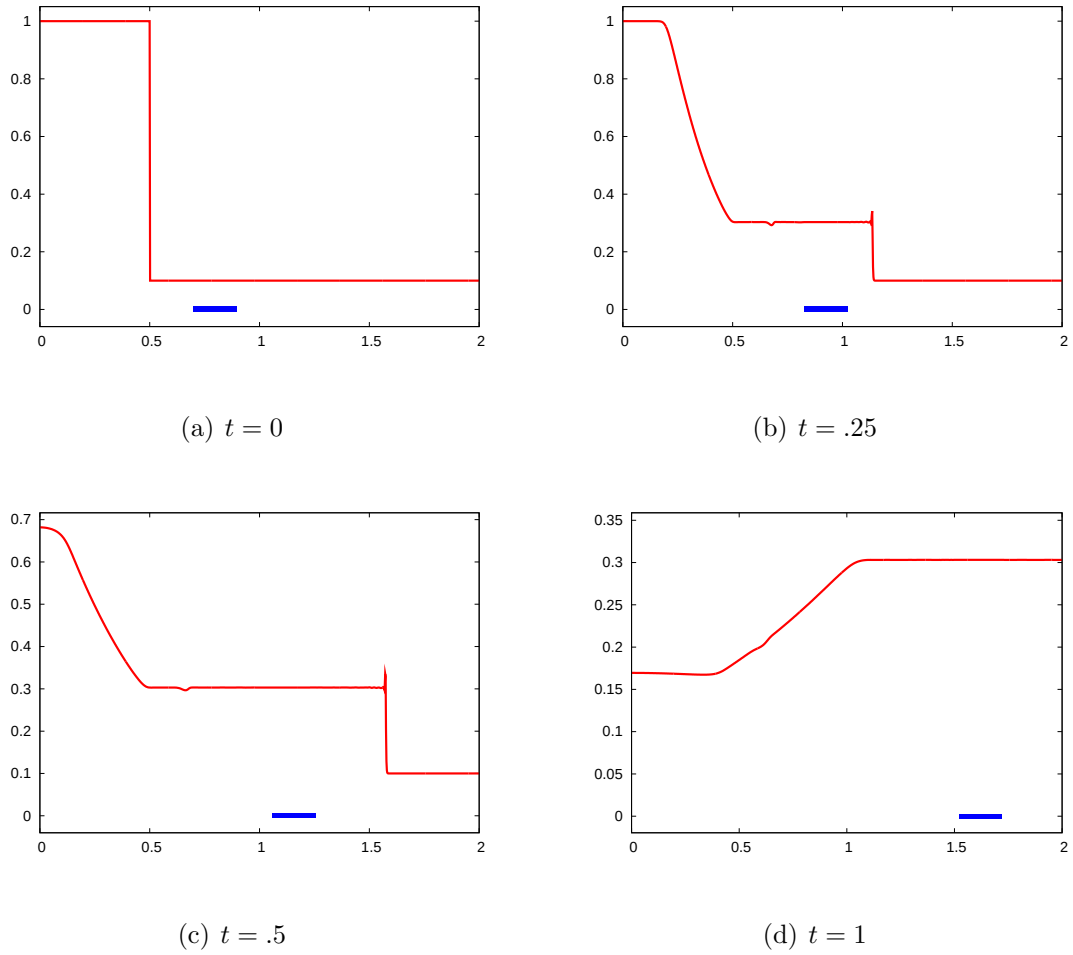


(b) Explicit.



(c) Semi-Implicit symmetric positive-definite formulation.

Figure 3.7: Velocity of a 1-D rigid body hit by a Sod shock, as a function of time. Simulations were done on a grid of resolution 1601. All simulations were run with a CFL number of .6, where the explicit simulation CFL is based on $|u| \pm c$ and the semi-implicit simulation was run with the CFL condition specified in Equation (2.3). The explicit simulations grow increasingly unstable as mass tends to zero, giving unusable results when mass reaches .0001 (these results are shown in Figure 3.9), and crashes for lighter masses. As mass tends to zero, the momentum absorbed by the solid tends to zero and the shock passes through the solid relatively unperturbed, and so the flat line to which solid velocities appear to converge is in fact the post-shock velocity of the fluid.



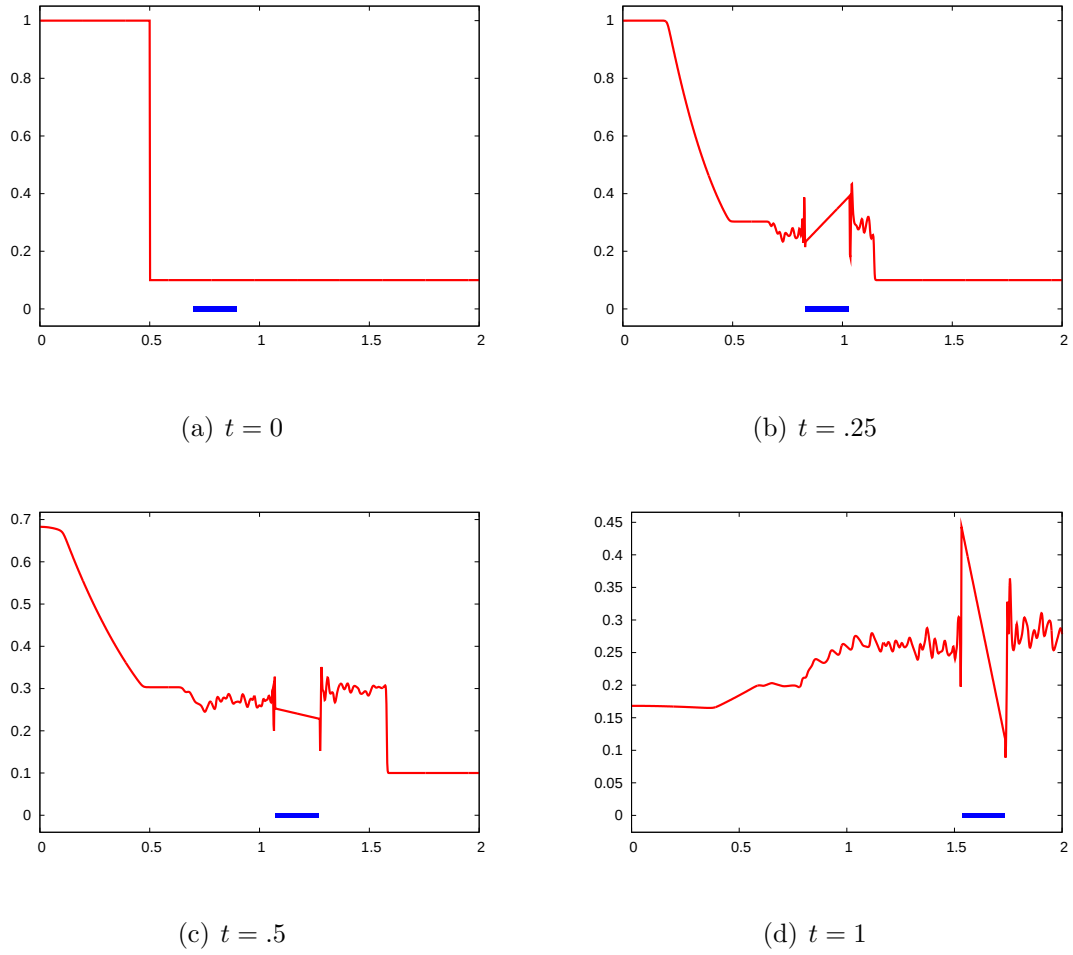
(a) $t = 0$

(b) $t = .25$

(c) $t = .5$

(d) $t = 1$

Figure 3.8: Semi-implicit simulation of a Sod shock hitting a light solid of mass .0001. Pressure profile of the fluid is shown at various times through the simulation. The 1-D rigid body is drawn as a blue line segment at the bottom of the plot. The simulation was done on a grid of resolution 1601. For this light mass, the post-shock state remains practically undisturbed as very little momentum transfers to the solid.



(a) $t = 0$

(b) $t = .25$

(c) $t = .5$

(d) $t = 1$

Figure 3.9: Explicit simulation of a Sod shock hitting a light solid of mass .0001. Pressure profile of the fluid is shown at various times through the simulation. The 1-D rigid body is drawn as a blue line segment at the bottom of the plot. The simulation was done on a grid of resolution 1601. The CFL number for this simulation is .6, and we use the standard compressible flow CFL, based on $|u| \pm c$. Despite satisfying a reasonable CFL time step restriction, a fully explicit simulation generates unstable results, and even goes unstable and crashes for masses lighter than .0001.

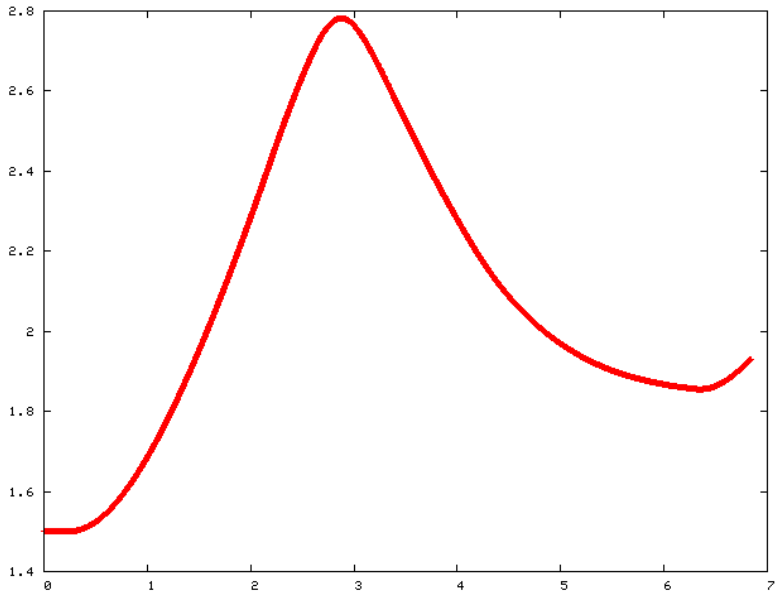


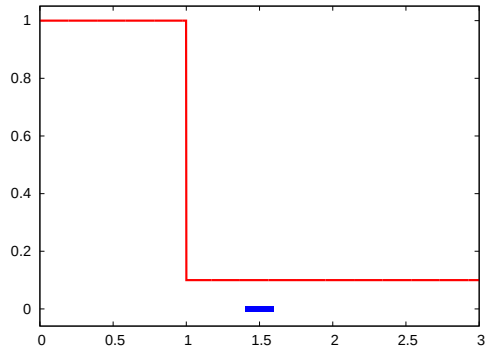
Figure 3.10: The position of the piston (Section 3.6.1) is plotted as a function of time.

Sod shock coupled with a mass-spring system

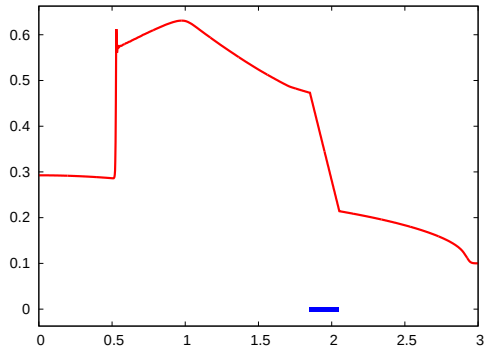
To conclude the one-dimensional examples, we consider the mass-spring system interacting with a high pressure gas described in [1] in order to provide validation for our approach against an analytic solution. The domain is of length 20, and a spring is fixed to the right side of the domain which has a rest length of 1, a stiffness of 10^7, no damping and a mass of 3. The fluid is given by

$$(\rho, p, \vec{u}) = (4, 10^6, 0)$$

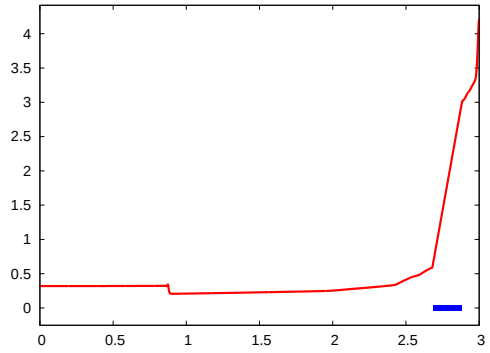
An outflow boundary condition is used for the left side of the domain. The spring starts at rest length and is compressed by the gas. Figure 3.14 shows snapshots of the pressure profile at various times through the simulation. The position of the moving end of the spring as a function of time is shown in Figure 3.15(a), and a convergence analysis in Figure 3.15(b). The error in the position of the free end of the spring is computed at time .008, and is compared against the analytic solution provided in [1]. The convergence order of the error is estimated as 1.16.



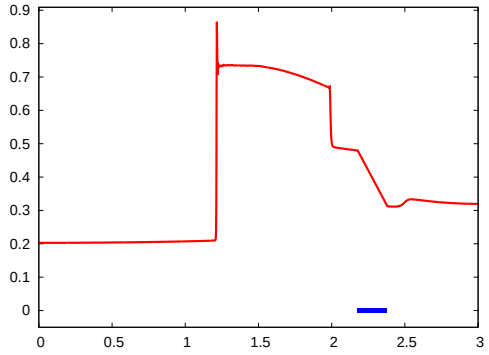
(a) $t = 0$



(b) $t = 1.5$

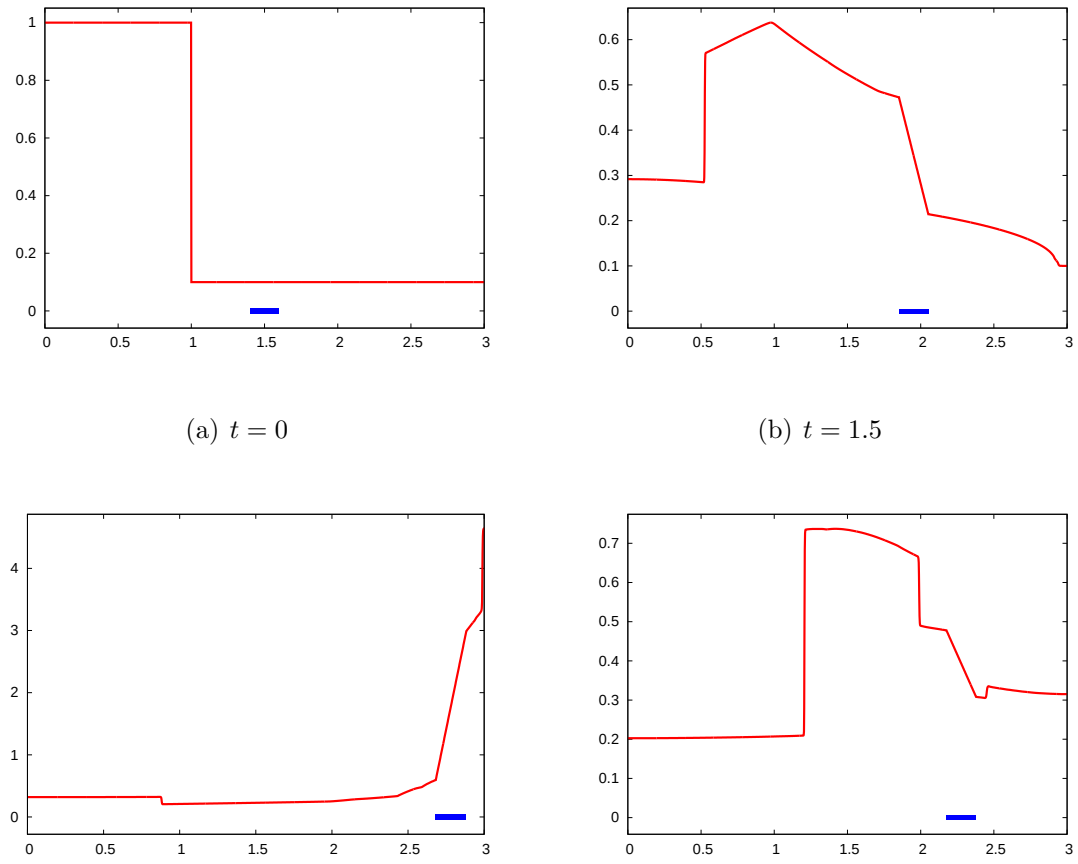


(c) $t = 2.87$



(d) $t = 4$

Figure 3.11: Semi-implicit simulation of a piston hit by a Sod shock, with closed-wall boundary conditions on both sides. Pressure profile of the fluid is shown at various times through the semi-implicit simulation. The 1-D rigid body is drawn as a blue line segment at the bottom of the plot, with pressure inside the solid shown as a linear pressure profile. The simulation was done on a grid of resolution 1601. The shock on the left pushes the rigid body and compresses the fluid on the right into a small high pressure pocket against the wall, which in turn pushes the rigid body back to the left.



(a) $t = 0$

(b) $t = 1.5$

(c) $t = 2.87$

(d) $t = 4$

Figure 3.12: Explicit simulation of a piston hit by a Sod shock, with closed-wall boundary conditions on both sides. Pressure profile of the fluid is shown at various times through the explicit simulation. The 1-D rigid body is drawn as a blue line segment at the bottom of the plot, with pressure inside the solid shown as a linear pressure profile. The simulation was done on a grid of resolution 1601. The shock on the left pushes the rigid body and compresses the fluid on the right to a very high pressure against the wall, which in turn pushes the rigid body back to the left.

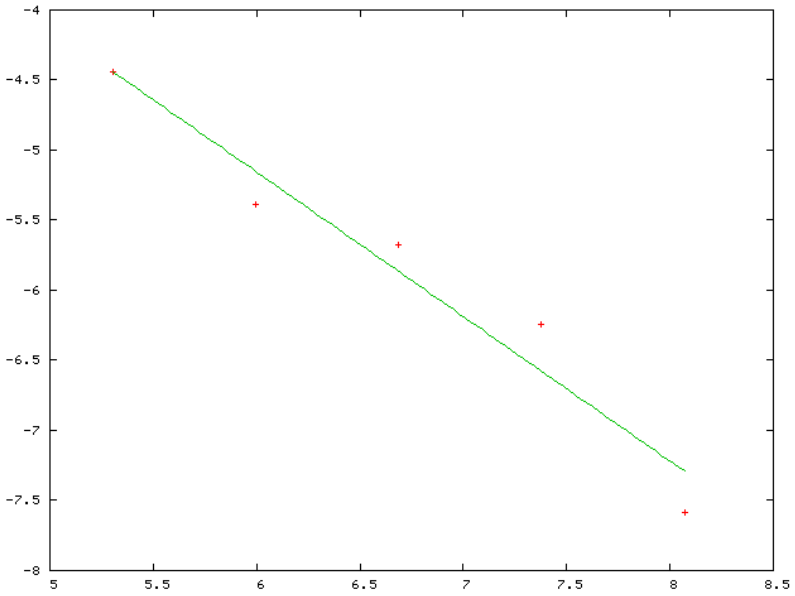
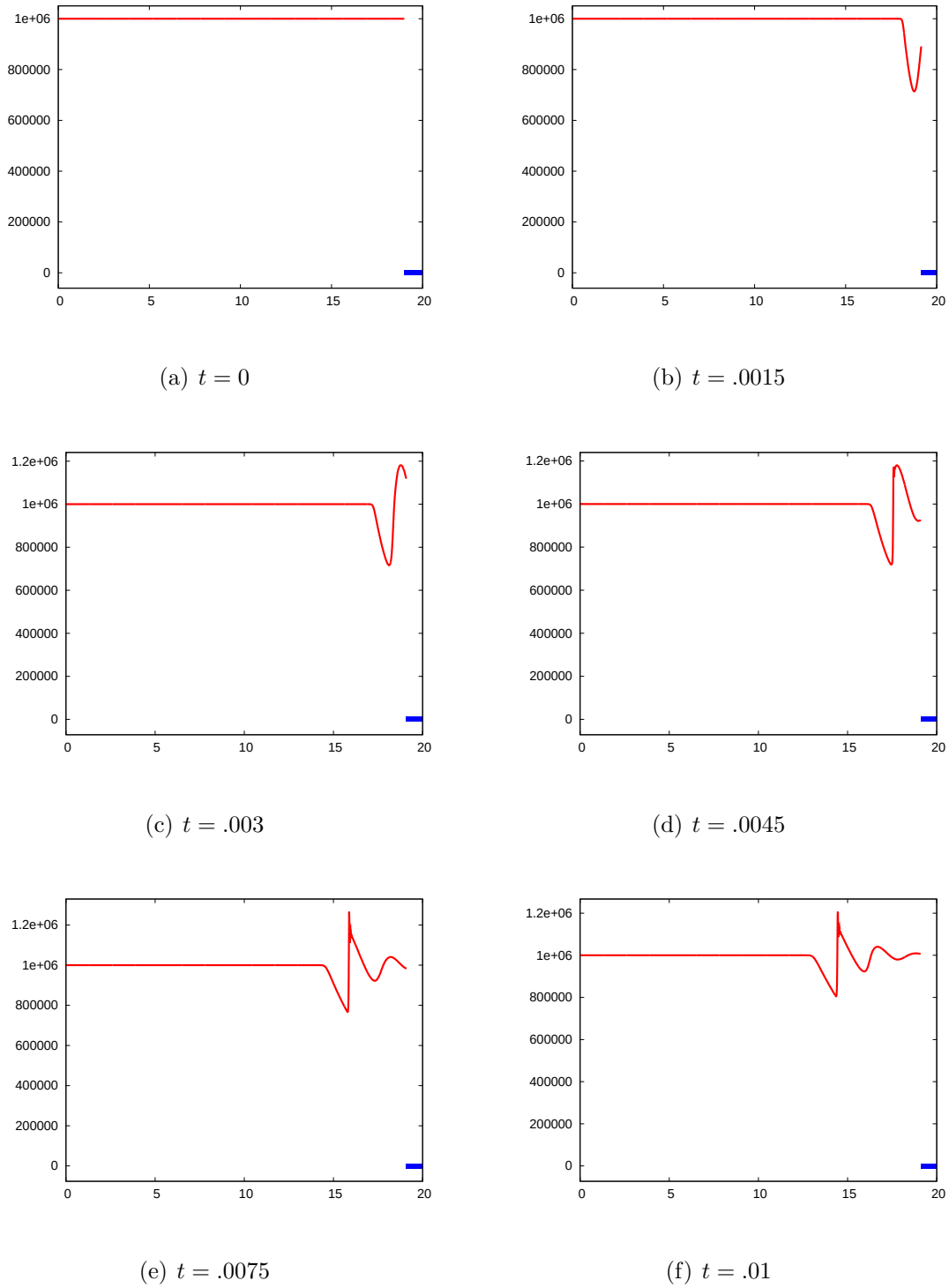


Figure 3.13: Position error of the center of mass of the piston (Section 3.6.1), as compared to a high-resolution simulation, at time $4s$. We plot the log of the relative error, as a function of the log of the resolution of the underlying grid. The convergence rate is 1.03.



(a) $t = 0$

(b) $t = .0015$

(c) $t = .003$

(d) $t = .0045$

(e) $t = .0075$

(f) $t = .01$

Figure 3.14: Semi-implicit simulation of a 1-D mass-spring system hit by a Sod shock wave. Pressure profile of the fluid is shown at various times through the semi-implicit simulation. The mass-spring system is drawn as a blue line segment at the bottom of the plot. The simulation was done on a grid of resolution 1601. Note the formation of a spontaneous shock wave.

3.6.2 Two-dimensional validation

In this section we validate our method for the multidimensional case, and briefly describe a symmetric positive-definite reformulation of the Equation (3.18). We consider interactions with both rigid and deformable solids. A second order ENO scheme was used along with an advection-based CFL number of .6.

Rigid Cylinder lift-off

This example, which is suggested by [7, 16, 1], examines the interaction of a Mach 3 shock with a rigid cylinder initially at rest on the floor of a rectangular channel. The cylinder is lifted by the shock, due to an asymmetric reflection of the incident wave. The test domain is $1 \times .2$, with the initial shock front positioned at .08 from the left boundary and the remaining domain is filled with the gas at pressure 1 and density 1.4. The top and bottom of the domains are rigid walls, the left boundary is fixed to be the post shock state and an outflow boundary condition is used for the right boundary. The cylinder has a density of 10.77, a radius of .05 and is initially located at $(.15, .05)$. Figure 3.16 shows the snapshot of the simulation for a selection of times. Our results compare favorably to those shown in [1], and converges at a rate of .93.

Deforming cylinder lift-off

This example is similar to the one described above (in Section 3.6.2), except that the rigid cylinder is replaced by a deformable mass-spring system with 222 triangles, and edge- and altitude-springs with a stiffness of .3. The density of the sphere is 10.77, has a radius of .05 and the center of mass is initially located at $(.15, .05)$. Figure 3.17 shows snapshots of the simulation for a selection of times. As the shock front passes through the deforming body, it dissipates, scatters and is partially absorbed by the body. The example converges at a rate of .99.

Heavy deforming cylinder lift-off

We next consider a heavy deforming cylinder, in the same setup as described in Section 3.6.2 and Section 3.6.2 above. In this case, the cylinder matches the cylinder

from Section 3.6.2, except the density is set to 100. As the body absorbs the shock wave, it compresses and delays the shock. Some of the shock is reflected, but most of the shock passes through the cylinder. Figure 3.18 shows snapshots of the simulation at a selection of times. The example converges at a rate of 1.01.

Shock traveling down a deformable tube

This example is similar to the inflatable bladder examples suggested in [1] and [9] in which a shock wave travels through a deformable tube causing large deformation of the walls. Our results are shown in Figure 3.19. We also do a convergence analysis of our method in Figure 3.20. The error in the position of a particle on the deformable tube is computed at time $.00049s$ (which is the approximate time of maximum deformation of that particle in the highest resolution simulation) from the highest resolution grid simulated, which is 800×600 grid cells. The convergence order of the error is estimated as 1.18.

Symmetric positive-definite reformulation

Our numerical method is symmetric, but not positive-definite. Recent developments in [41] discuss a modification of the implicit coupling methodology for incompressible flow by separating out the coupling forces as implicit variables λ (similar to immersed boundary methods), decomposing the symmetric damping force into $D = C^TC$ and solving for $\hat{V}_s = CV_s^{n+1}$. The symmetric positive-definite system they obtain can be modified for compressible flow in a manner similar to Section 3.4.1 to obtain

$$\begin{pmatrix} \frac{V}{\Delta t^2 \rho c^2} I + \hat{G}^T \beta^{-1} \hat{G} & -\hat{G}^T \beta^{-1} K^T & 0 \\ -K\beta^{-1}\hat{G} & K(\beta^{-1} + WM^{-1}W^T)K^T & KWM^{-1}C^T \\ 0 & CM^{-1}W^TK^T & I + CM^{-1}C^T \end{pmatrix} \begin{pmatrix} \tilde{p} \\ \lambda \\ \hat{V}_s \end{pmatrix} = \begin{pmatrix} \frac{V}{\Delta t^2 \rho c^2}\tilde{p}^a + \hat{G}^T u^\star \\ KWV_s^\star - Ku^\star \\ CV_s^\star \end{pmatrix}, \tag{3.21}$$

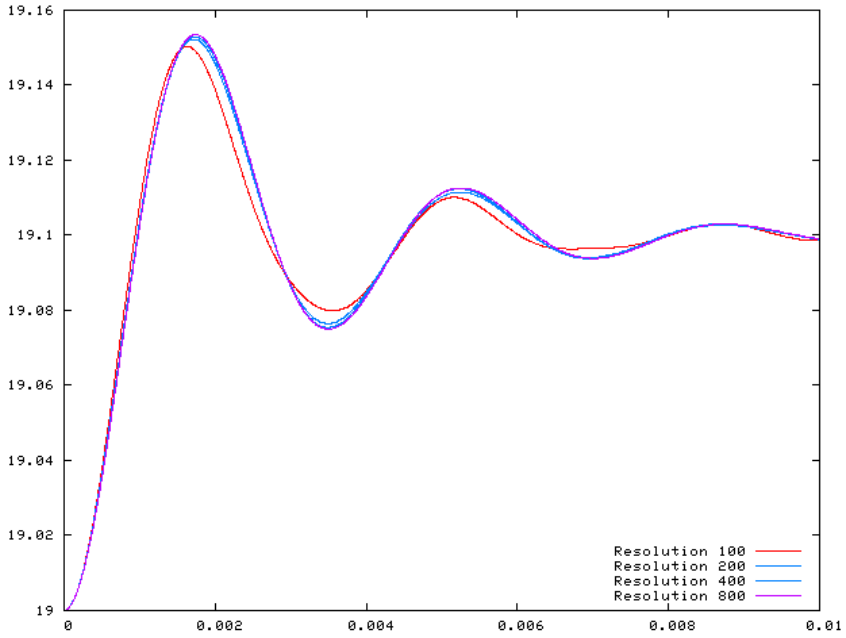
where $\hat{G}$ and $-\hat{G}^T$ are the volume weighted gradient and divergence operators respectively, β is the diagonal matrix of fluid dual cell masses, and K^T is the matrix of 1s and 0s mapping λ to the appropriate fluid velocity scalars (see [41] for more details).

Note that in order to avoid confusion in notation we renamed a few operators. In particular W and J in [41] correspond to the K and W we use here, respectively. This system is both symmetric and positive-definite. We demonstrate the viability of this modified method in another example, where we've replaced the implicit coupled solve with Equation (3.21). Our example is similar to the example in Section 3.6.2 except that the sphere is replaced with a diamond whose major axis is of length .1 and minor axis is of length .025. The diamond begins rotated by $\pi/4$, with a center of mass at $(.15, .04)$. Snapshots of the resulting simulation are shown in Figure 3.21. The convergence analysis for this example is shown in Figure 3.22 which estimates the convergence order of the error as .84.

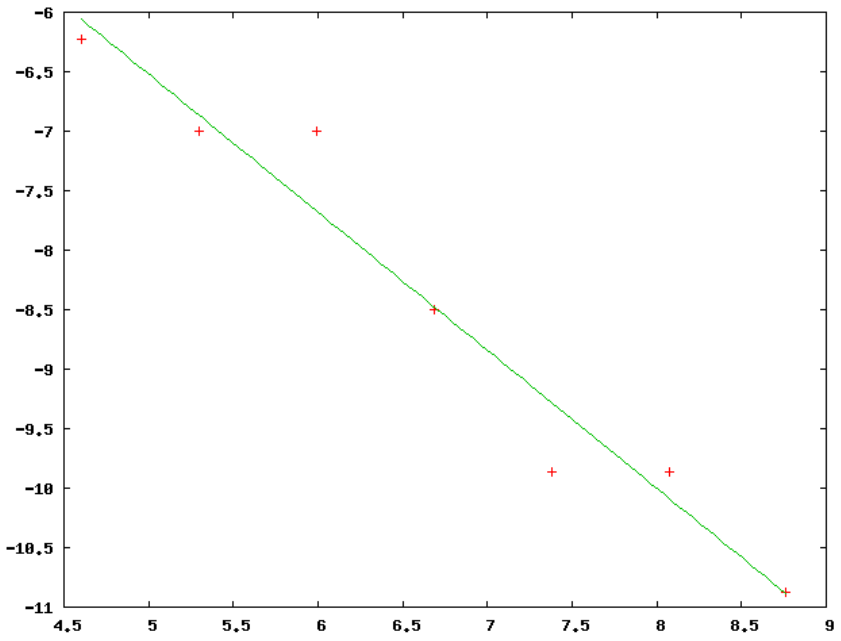
3.7 Conclusions and future work

We have presented a first order method which implicitly couples compressible flow with solid bodies with arbitrary constitutive models. We show that this method is robust, numerically conservative, and avoids the numerical instabilities which comparable explicit methods suffer from in the presence of high density-to-mass ratios. The same methodology can be applied to reformulate our implicit system into a symmetric positive-definite system.

There are several interesting avenues of future work which we wish to explore. Given the promising results which arise from handling fluid-structure interactions implicitly, we believe that an alternative approach would split the fluid flux along Riemann invariants–rather than by pressure–and solve for the Riemann invariant which interacts with the solid implicitly. Our method also relies on the assumption that the solid has some thickness where ghost cells can be filled, and we believe that the method can be made to work for thin shell structures (such as parachutes). Given the utility of the scheme proposed in 2 in handling fluid-structure interactions, it becomes imperative to address the issues of that original scheme. In particular, the implicit component of the method is overly centrally-differenced, which tends to introduce Gibbs phenomena at shocks. It would be better to add upwind biasing, although it is unclear how to do so.

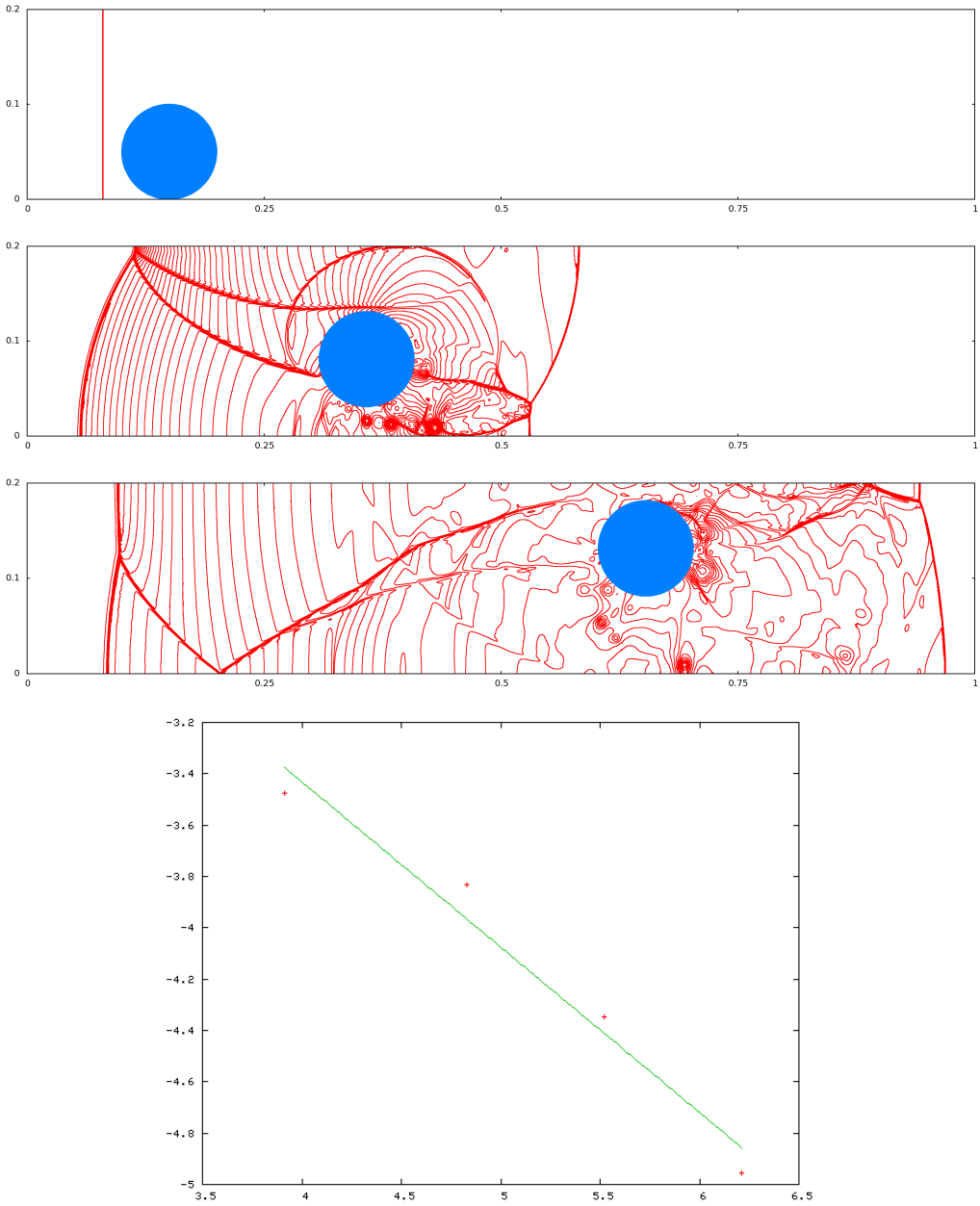


(a) Position of the free end of the spring, as a function of time.



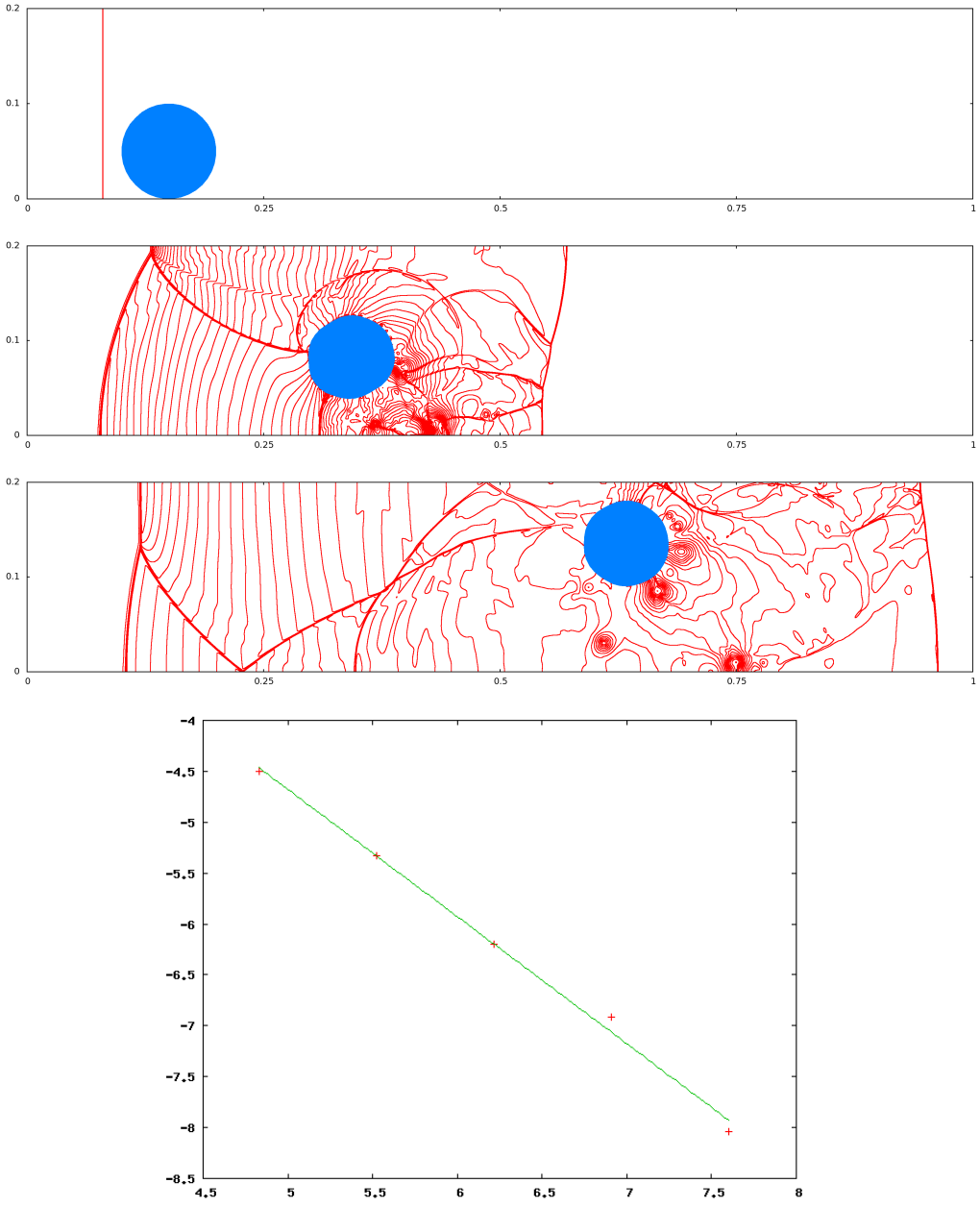
(b) Position error for the left-most side of the mass-spring system, as compared to the analytic solution provided in [1], at time .008s. We plot the log of the relative error, as a function of the log of the resolution of the underlying grid. The convergence rate is 1.16.

Figure 3.15: 1-D mass-spring system hit by a Sod shock wave.



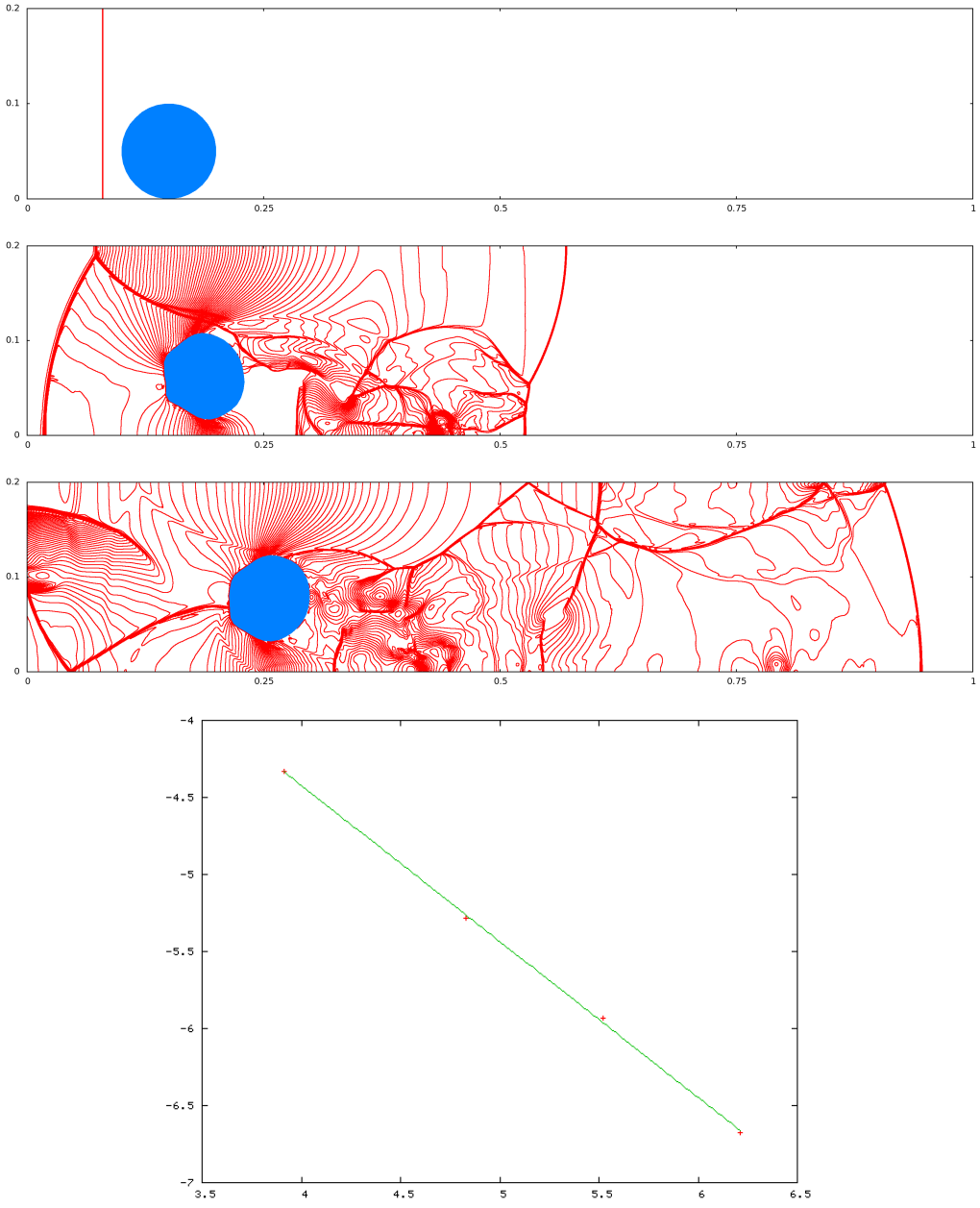
(d) Position error of the center of mass of the cylinder hit by a planar shock, as compared to a high-resolution simulation, at time $t = .15s$, with a convergence of .96.

Figure 3.16: Pressure contours for semi-implicit simulation of rigid cylinder lift off are shown at $t = 0$, $t = .164$ and $t = .301$. The simulation is run with a CFL number of .6, using the CFL restriction discussed in Equation 2.3.



(d) Position error of the center of mass of the deformable cylinder hit by a planar shock, as compared to a high-resolution simulation, at time $t = .15s$. We plot the log of the relative error, as a function of the log of the resolution of the underlying grid. The convergence rate is .99.

Figure 3.17: Pressure contours for semi-implicit simulation of deformable cylinder lift off are shown at $t = 0$, $t = .164$ and $t = .301$. The simulation is run with a CFL number of .6, using the CFL restriction discussed in Equation 2.3.



(d) Position error of the center of mass of the heavy deformable cylinder hit by a planar shock, as compared to a high-resolution simulation, at time $t = .15s$. We plot the log of the relative error, as a function of the log of the resolution of the underlying grid. The convergence rate is 1.01.

Figure 3.18: Pressure contours for semi-implicit simulation of deformable cylinder lift off are shown at $t = 0$, $t = .164$ and $t = .301$. The simulation is run with a CFL number of .6, using the CFL restriction discussed in Equation 2.3.

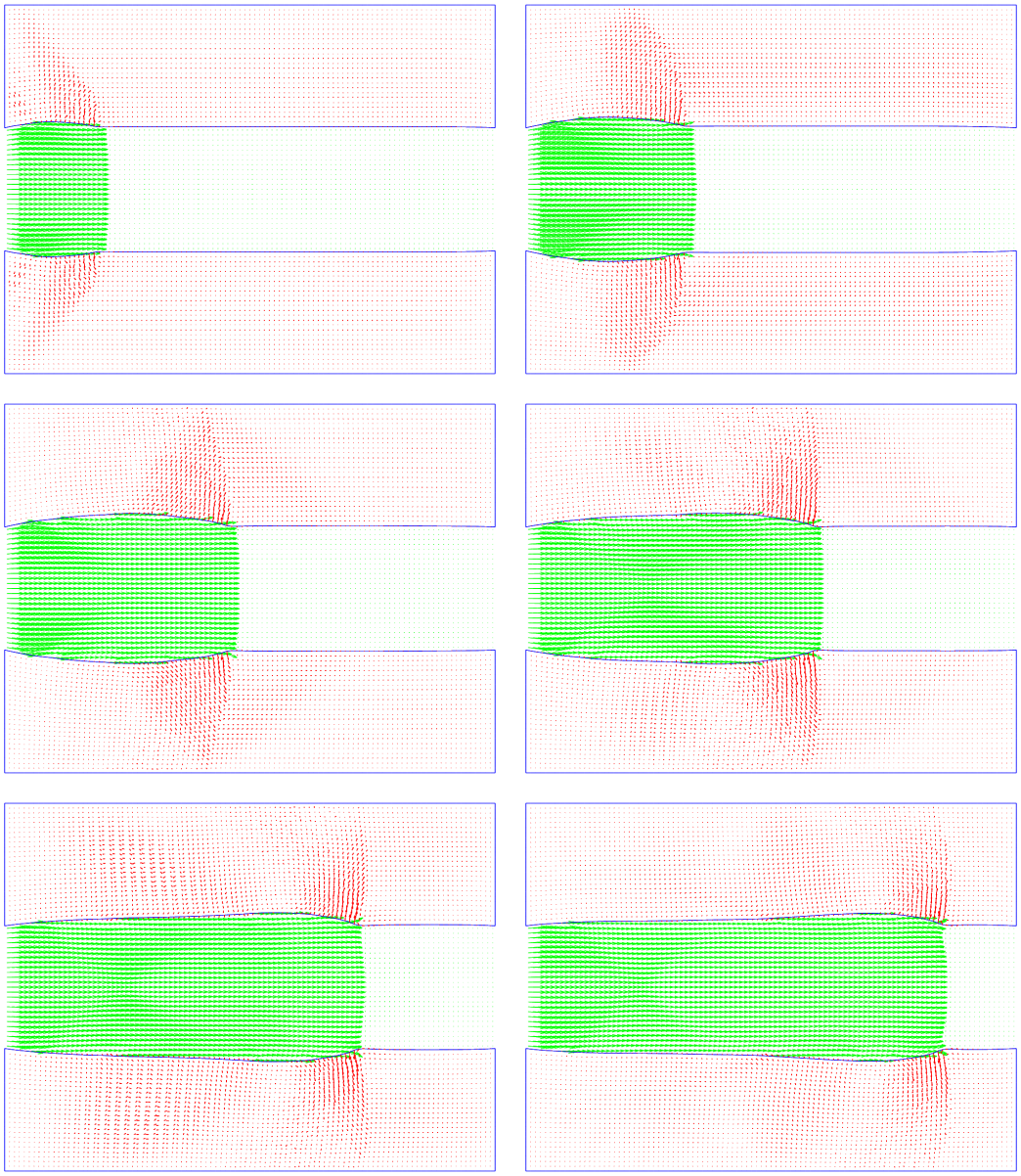


Figure 3.19: A planar shock travels down a deformable bladder. Shown are the velocity field of the fluid in green and the velocities of the deformable nodes in red at times $t = .0001$, $t = .0002$, $t = .0003$, $t = .0004$, $t = .0005$ and $t = .0006$.

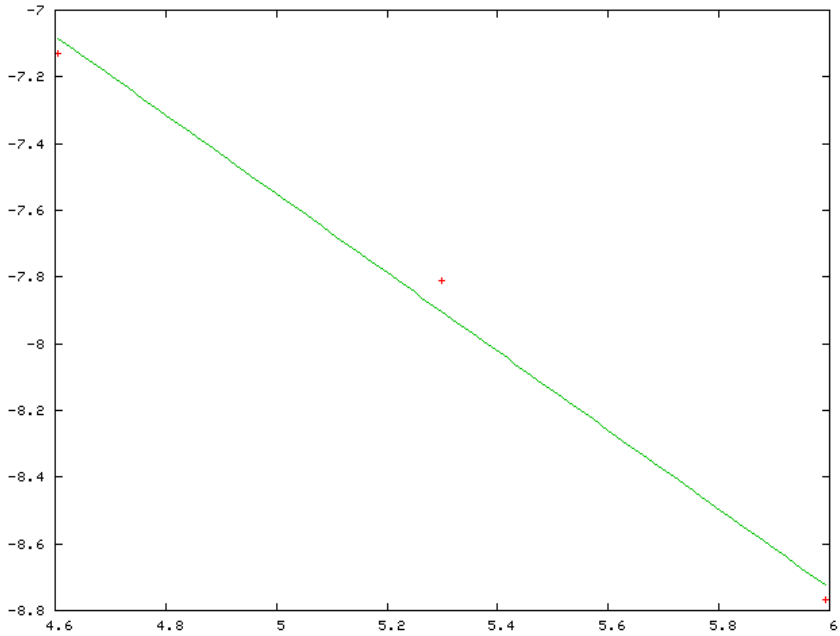


Figure 3.20: Position error of the position of a particle on the deformable tube hit by a planar shock, as compared to a high-resolution simulation, at time .00049s. We plot the log of the relative error, as a function of the log of the resolution of the underlying grid. The convergence rate is 1.18.

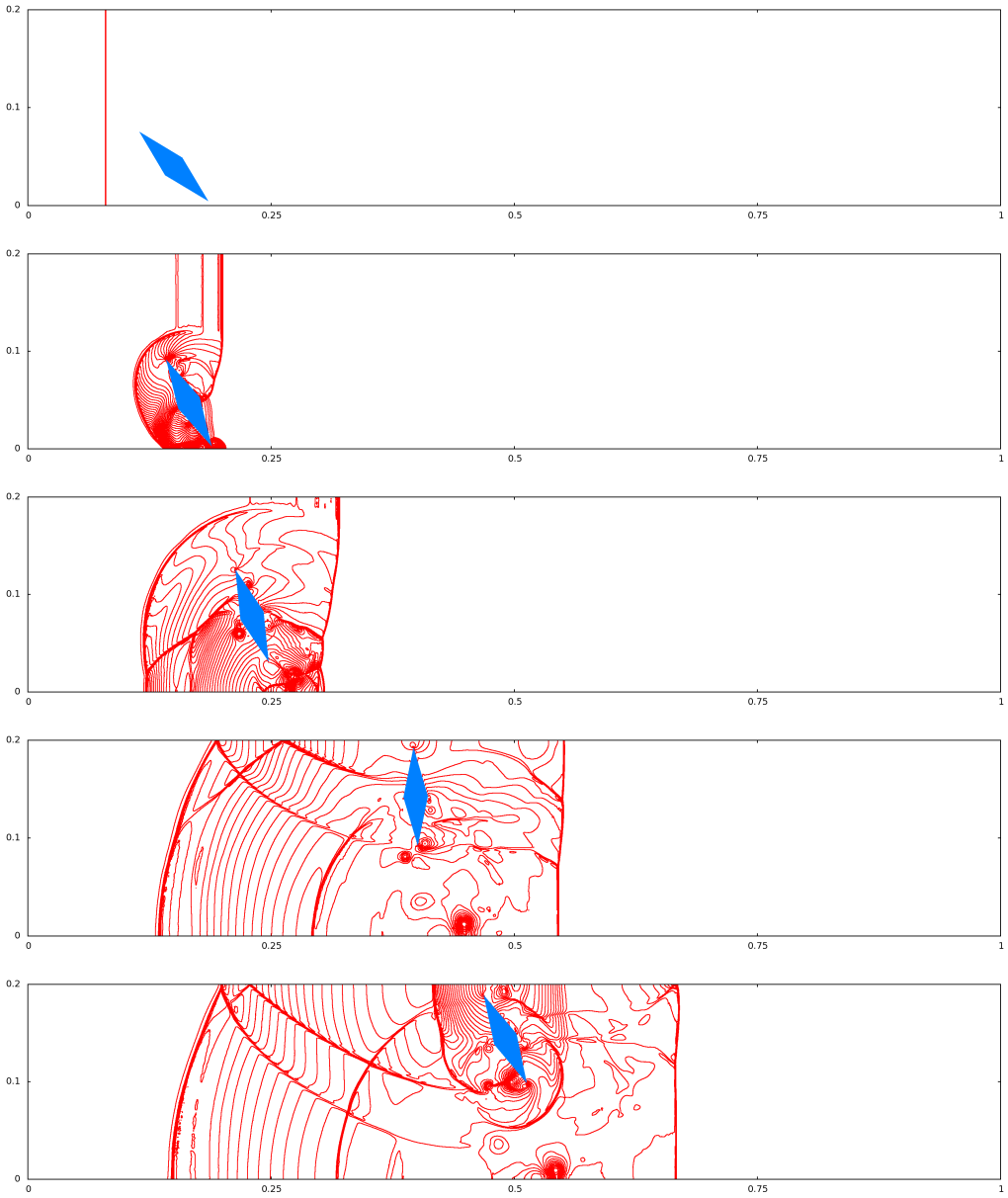


Figure 3.21: A diamond is hit by a planar shock, and then collides with the top of the channel. Shown are pressure contours at $t = 0$, $t = .04$, $t = .08$, $t = .16$ and $t = .2$.

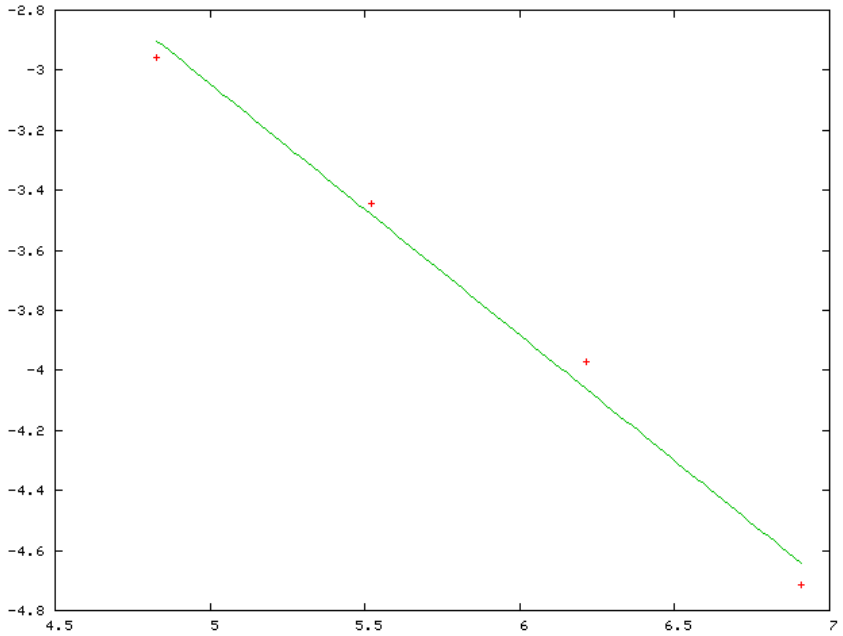


Figure 3.22: Position error of the center of mass of the diamond hit by a planar shock, as compared to a high-resolution simulation, at time .15s. We plot the log of the relative error, as a function of the log of the resolution of the underlying grid. The convergence rate is .84.

Chapter 4

Integrating Shock Wave Dynamics into Smoke Simulations

We propose a practical approach to integrating shock wave dynamics into traditional smoke simulations. Previous methods either simplify away the compressible component of the flow and are unable to capture shock fronts or use a prohibitively expensive explicit method that limits the time step of the simulation long after the relevant shock waves and rarefactions have left the domain. Instead, we employ a semi-implicit formulation of Euler's equations, which allows us to take time steps on the order of the fluid velocity (ignoring the more stringent acoustic wave-speed restrictions) and avoids the expensive characteristic decomposition typically required of compressible flow solvers. We also propose an extension to Euler's equations to model combustion of fuel in explosions. The flow is two-way coupled with rigid and deformable solid bodies, treating the solid-fluid interface effects implicitly in a projection step by enforcing a velocity boundary condition on the fluid and integrating pressure forces along the solid surface. As we handle the acoustic fluid effects implicitly, we can artificially drive the sound speed c of the fluid to ∞ without going unstable or driving the time step to zero. This permits the fluid to transition from compressible flow to the far more tractable incompressible flow regime once the interesting compressible flow phenomena (such as shocks) have left the domain of interest, and allows the use of state-of-the-art smoke simulation techniques.

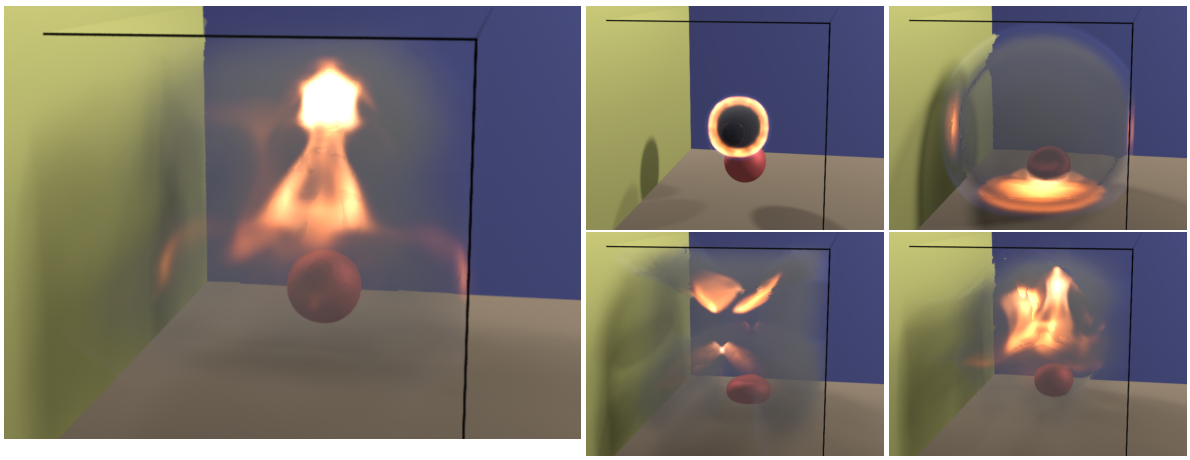


Figure 4.1: A charge is detonated near a deformable ball. The ball compresses and bounces off the ground as it interacts with the shock. The soot heats up near the shock front and emits blackbody radiation. This was solved on a $150 \times 100 \times 100$ grid.

4.1 Introduction

Shock waves have had a deep and varied impact across multiple disciplines within the graphics community. The solids community, for example, has put significant effort into capturing the destructive effect that blasts have on rigid bodies, realistically fracturing [35, 36] and generating interesting small-scale debris and dust [22]. Unfortunately, these methods suffer from an over-simplified model of the blast itself, making them useful only in the very limited venue where the dynamic effect of the fluid is negligible.

In the fluids community focus has been more on modeling the after-effects of an explosion, e.g. the smoke plumes of [17, 50, 14]. A few papers have simulated phenomena related to the explosion itself. For example [40, 54, 15, 21, 19, 26] modify the incompressible flow equations in various ways, such as by adding a divergence term to approximate the expansion due to chemical reactions. These generate fantastic fireball-style effects, but are held back by the underlying modeling assumptions. In particular, by simulating the fluid as *incompressible* they discard the compression waves and the potentially dramatic effects therein–such as shocks.

In order to capture the physics that drive shock waves, we must instead consider the compressible Euler equations, as done in [59, 46, 45]. The equations which drive the fluid flow naturally yield the information necessary to generate physically accurate shock waves, albeit at a significantly increased computational cost. Compressible flows require conservative advection schemes such as ENO-Roe [47] in order to capture shocks at the correct speeds and properly account for the highly non-linear, discontinuous nature of compressible flow. This excludes the fast schemes typically used for incompressible simulation, such as semi-Lagrangian, BFECC or MacCormack advection [50, 6, 28, 43]. Moreover, the time step of a compressible flow simulation is constrained by the sound speed c in addition to bulk velocity; this severe restriction is necessary to properly resolve the shock wave and related phenomena, but is unduly limiting once these effects have left the domain of interest.

Shock and other compressible flow phenomena impose small time steps and therefore require a large amount of computational effort to simulate a mere fraction of a second. Other authors who have carried out these types of simulations show shocks moving around (in slow motion) etc., and stop their simulations/video after a short time. If they would have continued simulating, one would see more of the same, shocks moving, etc. for a few more fractions of a second. In the real world these shocks eventually dissipate as do the effects of compressibility, leading eventually to a plume type structure more representative of smoke and fire - governed more appropriately by incompressible flow. It is computationally infeasible for existing methods to simulate what happens to a flow field over 5-10 seconds when a large amount of computational resources are needed to advance a fraction of a millisecond.

Instead we propose to transition the flow from compressible to fully incompressible by sending $c \to \infty$. Non-physically driving the sound speed to ∞ accelerates the behavior of the fluid in order to obtain incompressible style flow phenomena such as rolling and plumes much quicker than one would otherwise attain. Any explicit method would have its time step driven to zero as the sound speed is driven to ∞, and therefore would not make any progress towards the incompressible flow behavior we are after. Thus our semi-implicit method disucussed in chapter 2 lends itself well to this approach as the formulation naturally yields the Godunov splitting methods

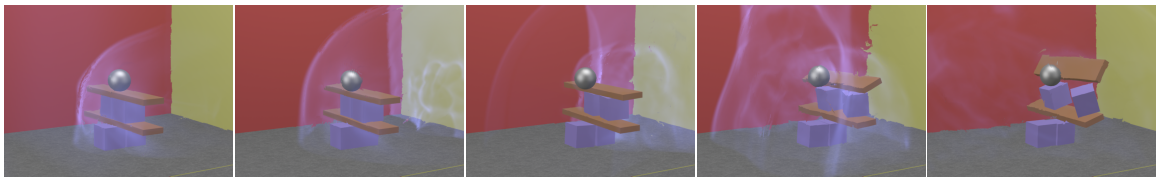


Figure 4.2: A planar shock enters an enclosed domain and disrupts a stack of rigid bodies. It reflects off of the back wall, hits the stack of objects again, and exits the domain. Times $.0011s$, $.0018s$, $.003s$, $.0043s$ and $.0053s$ are shown. The grid size is $225 \times 150 \times 150$.

intrinsic to incompressible flow. Once the flow is fully incompressible, there are many mature simulation techniques that can be used to enhance the visual fidelity and speed of traditional smoke simulations. Vorticity confinement [51] and vortex particles [44] help to reduce the numerical viscosity introduced by fast, low-order advection schemes. Non-uniform mesh refinement techniques such as Octrees [33] and RLE [20] grids permit faster simulations by discarding information away from the area of interest.

One of the main contribution of our paper is the ability to show both the initial states of the explosion including shock waves along with the long time behavior of rolling plumes and other incompressible flow effects. To the best of our knowledge this has not been previously addressed and other authors merely stop their simulations after shocks have moved around a little bit.

4.2 Flow Regime Transition

One drawback of existing methods is that the small time steps required for simulation, coupled with the complexity of simulating compressible flow, result in simulations that are relatively short. Shock waves travel across the domain and a tiny plume starts to form, just before the simulation ends. Obviously, transitioning from compressible to incompressible flow allows one to take bigger time steps and show more of the interesting incompressible flow-style smoke effects which persist long after the shocks have exited the domain. Simply initializing an incompressible flow from the output of a compressible flow simulation leads to significant velocity discontinuities and unusable

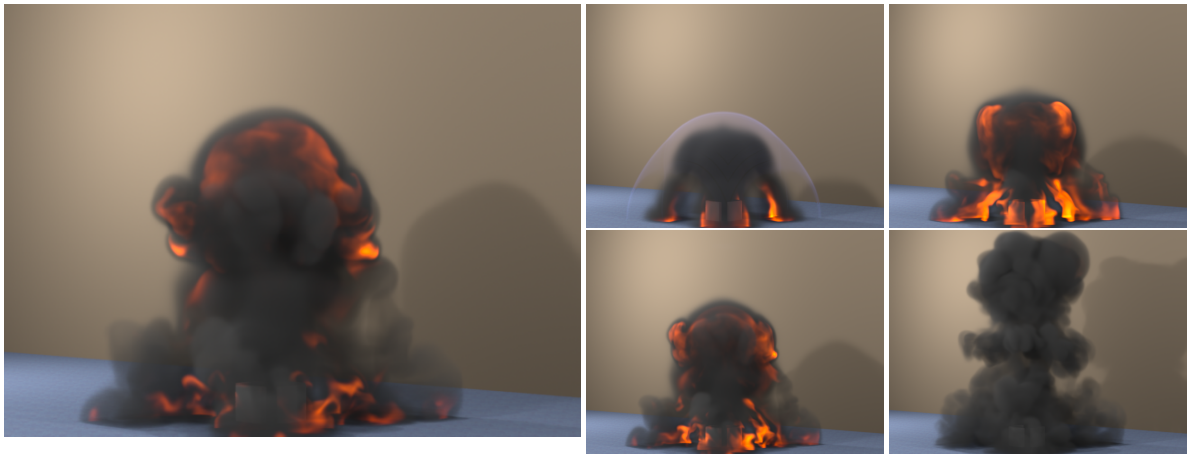


Figure 4.3: A charge is detonated within a small four-walled chamber at $t = 0$. Shown on the left is a smoke plume at $t = 0.04s$, and on the right its development, at times $.005s$, $.02s$, $.03s$ and $.115s$. This was solved on a $200 \times 300 \times 200$ grid.

results as the compressible flow velocities can be far from divergence-free. Therefore one needs a smooth transition, which can be achieved by sending $c \to \infty$. Unfortunately when using an explicit time step, pushing c to ∞ drives the time step to zero and no progress can be made whatsoever. This is not a concern for a semi-implicit method.

As $c \to \infty$, the EOS decouples entirely from the solve and the pressure evolution equation becomes $\nabla \cdot \vec{u} = 0$, which is exactly incompressibility. This in turn decouples E from the simulation, and sends $\nabla \cdot (\rho \vec{u}) \to \vec{u} \cdot \nabla \rho$, giving the more familiar advection equations that drive incompressible flow. Substituting the density equation

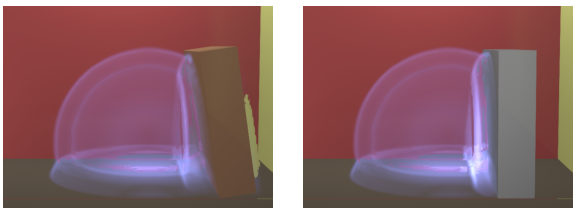


Figure 4.4: A shock interacts with a light wall (left) and a heavy wall (right) respectively, at $t = 0.316s$. Note how the shock passes through the light wall, and strongly reflects off of the heavy wall. The grid resolution is $225 \times 150 \times 150$.

into the momentum equation (see equation (3.1)) and taking into account the above simplifications gives:

$$\begin{cases} \vec{u}_t + \vec{u} \cdot \nabla \vec{u} + \frac{\nabla p}{\rho} = 0 \\ \nabla \cdot \vec{u} = 0 \end{cases} \tag{4.1}$$

It remains, then, to chose how to send $c \to \infty$. When a flow becomes incompressible it forcibly damps out discontinuities such as shock waves, potentially causing drastic changes in the flow field. Consider the speed of a shock U, given for a gamma-law gas [31] as

$$U = \left(1 + \frac{\gamma + 1}{2\gamma} \frac{p_1 - p_0}{p_0}\right)^{\frac{1}{2}} c_{\text{EOS}}, \tag{4.2}$$

where c_{EOS} is the sound speed as determined by the EOS (as opposed to c, which we artificially accelerate). By artificially driving $c \to \infty$ over an interval of time (t_s, t_f), we force shock waves to travel faster and faster, effectively dispersing them before going fully incompressible. Equation (3.18) only contains $(1/c)$ terms, so it is more convenient numerically to send this term to 0. A naïve approach might linearly interpolate between $1/c_{\text{EOS}}$ and 0, however this simply does not accelerate the sound speed sufficiently fast, being only a $10\times$ amplification by the time we are 90% through the transition. Experimentally, we found however that

$$(1/c)(t) = \frac{1}{c_{\text{EOS}}} \left(1 - \frac{t - t_s}{t_f - t_s}\right)^3 \tag{4.3}$$

gives reasonable results. Note that since we are handling the acoustic component of the flow implicitly, this artificial acceleration of shock waves does not affect the stability of our method.

Once the flow has fully transitioned, we can switch to using a traditional incompressible flow solver and take advantage of the rich body of literature which has been invested in making fast, visually stunning smoke simulations.

As we artificially change c during our transition, the EOS pressure becomes increasingly inaccurate estimate of p^n. One could alleviate this by keeping the pressure

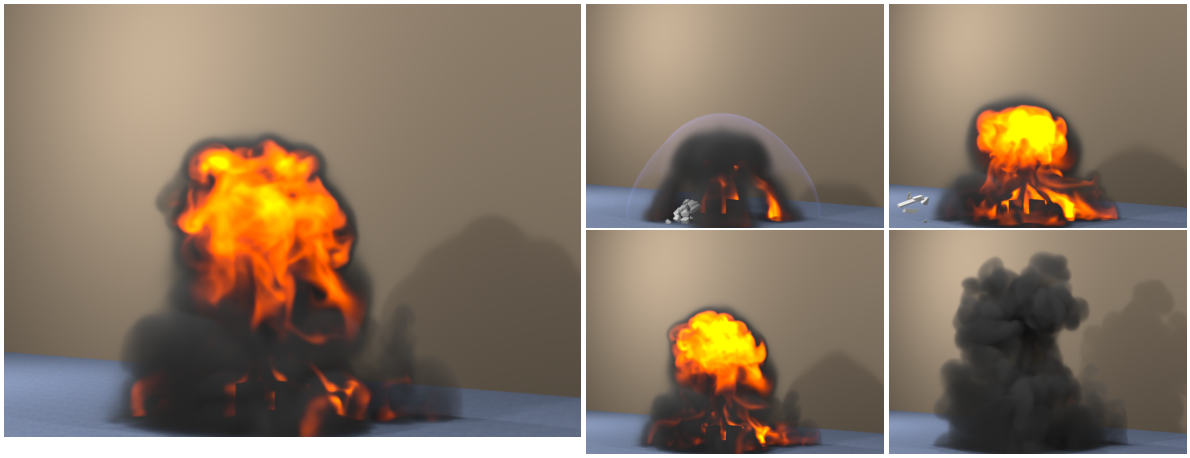


Figure 4.5: A charge is detonated within a small four-walled chamber with a fragile wall at $t = 0$. Shown on the left is a smoke plume at $t = 0.04s$, and on the right its development, at times $.005s$, $.02s$, $.03s$ and $.115s$. This was solved on a $200 \times 300 \times 200$ grid.

from the previous time step rather than reinitializing p^n, however this choice does not appear to have any effect on the flow, likely because the contribution from p^n vanishes rapidly, disappearing entirely from the governing equations in the limit. Instead we prefer to reinitialize p^n from the EOS until the flow is fully transitioned.

4.3 Combustion

The method described above models compressible flow and the related non-linear phenomena like shocks. However, a lot of visual detail in explosions also comes from chemical reactions due to burning of fuel. We follow an approach similar to [15] for modeling combustion. We track fuel in the domain by using a passively advected scalar fuel field F, defined as the fraction of mass in the cell that is fuel, using

$$F_t + \vec{u} \cdot \nabla F = S_F, \tag{4.4}$$

where S_F denotes the source terms. If the temperature at a cell i is greater than the ignition point T_I of the fuel, the fuel will burn at a specified rate b, i.e. $S_F(i) =$

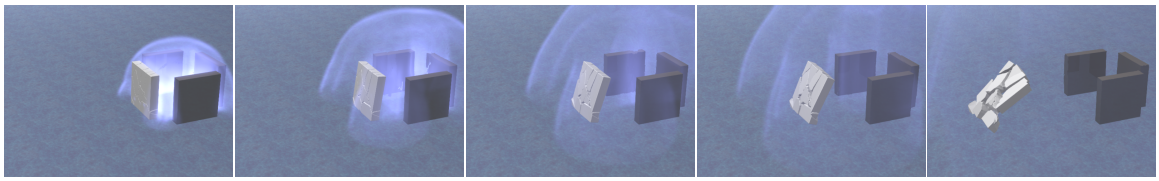


Figure 4.6: A charge detonates, fracturing a fragile wall. Shown are $t = .0005s$, $t = .0010s$, $t = .0015s$, $t = .0020s$, $t = .0040s$.

$-b/\rho$ for a cell with temperature greater than T_I and 0 otherwise. The burning fuel generates heat at a rate given by $r_h b$, where r_h is the calorific value of the fuel. This generation of heat is easily accounted for in Euler's equations by adding $r_h b$ as an energy source term to the right-hand side of the bottom row of Equation (3.1). Note that this source term will increase the internal energy/temperature of the compressible fluid, which will in turn increase pressure causing an expansion. This phenomena was modeled in [15] by adding an artificial divergence to their incompressible flow solver yielding impressive results; however, our semi-implicit compressible flow formulation allows us to model this expansion due to burning fuel in a more physical manner.

Another secondary effect of burning fuel is the generation of carbon particles or soot. We model soot by tracking the soot field C, defined as the fraction of mass in the cell that is soot, via

$$C_t + \vec{u} \cdot \nabla C = S_c, \tag{4.5}$$

where S_c denotes the source terms. The soot generated by burning fuel can be modeled by setting $S_c = r_c b/\rho$, where r_c denotes the mass of soot produced per unit mass of fuel burnt. We initialize both the soot and fuel to be non-zero at the detonation site and zero everywhere else. Note that ρC is the total soot in a control volume, ρF is the total fuel and $\rho(1 - C - F)$ is air.

4.4 Fracture

Our stable, two-way, strongly coupled methodology lends itself naturally to computing explosive phenomena like fracture. Like [59], we have access to the pressure forces (computed as $W^T B A_f p^{n+1}$), which can then be plugged into existing fracture codes

such as the one proposed in [52] to produce visually stunning special effects. We demonstrate this by incorporating said fracture framework, which uses fracture patterns, a threshold and a point of impact to break an object into debris. One could designate the point of impact by searching through the fluid faces coupled to a given solid and choosing the face whose pressure force is maximized. However, as our geometries are simple, we simply cast a ray back along the direction of force and use that intersection with the body surface as the point of impact.

4.5 Rendering

Soot and Heat: We use a standard volumetric smoke rendering [14] algorithm for the visualization of soot. The soot also emits light with intensity proportional to its density, and color given by blackbody radiation.

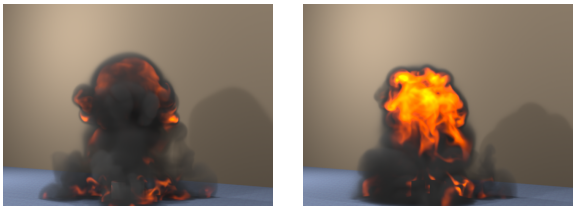


Figure 4.7: Smoke plumes which result from a detonation within an enclosed chamber (left), and from a detonation within a chamber whose front left wall is fragile (right).

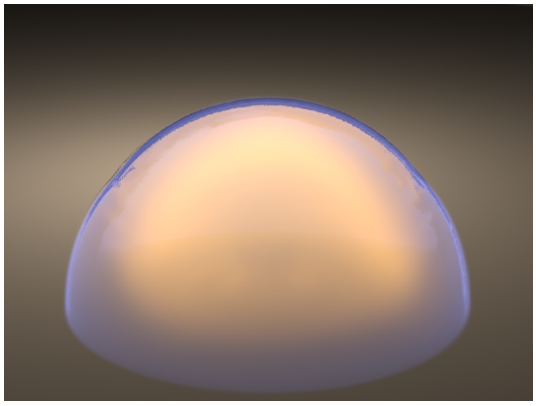


Figure 4.8: The 1945 Trinity Test, simulated on a $200 \times 100 \times 200$ grid.

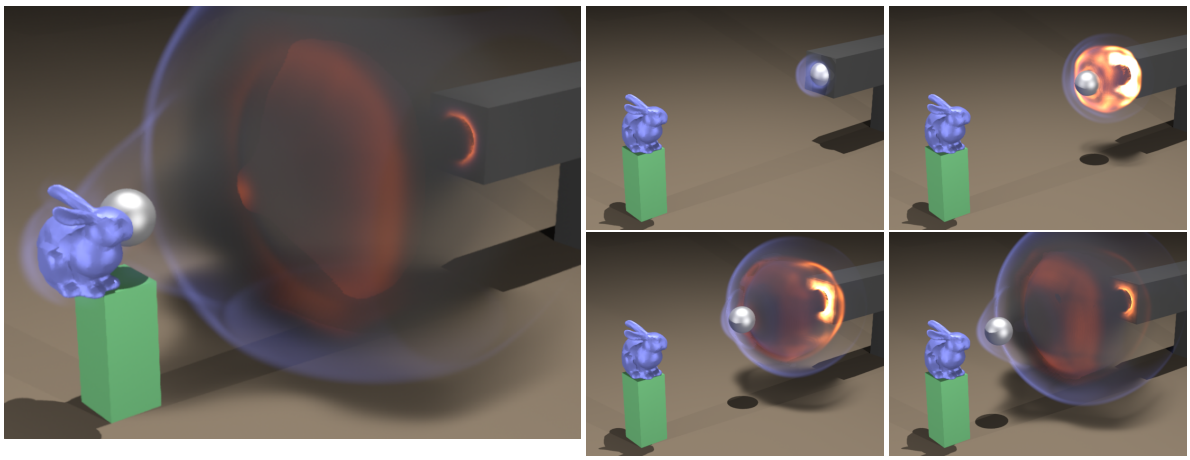


Figure 4.9: A cannonball is accelerated by an explosion in the barrel of the cannon. It reaches a super-sonic speed of 1500 m/s, and generates a secondary shock wave as it compresses the air in front of it. The grid resolution is $300 \times 120 \times 120$.

Shock Fronts: Shocks are detected in the flow field by examining $|\nabla p|$ and noting any location where the gradient is above some threshold to be the location of a shock front. They are then used to refract light (as in [59]), using $\nabla p/|\nabla p|$ as the surface normal and bending rays according to the difference in refraction index η across the shock front. This effect can have a dramatic visual effect on the simulation, as in Figure 4.8 where the strength of a nuclear explosion bends light significantly. In a more typical scenario, The effect is subtle but distinct (as seen in Figure 4.1). We further enhance the visual impact of the shock by adding a blue emittance which scales with $|\nabla p|$, demonstrated in Figure 4.2.

4.6 Results

We simulate air as an ideal gas with $\gamma = 1.4$, with a rest state temperature of $T_{atm} = 290\ K$, zero initial velocity, and pressure of $p_{atm} = 1.01325 \times 10^5\ Pa$, or atmospheric pressure. This gives a fluid of density $\rho = 1.4\ kg/m^3$, comparable to that of air. Unless otherwise noted we initialize a shock by instantaneously depositing a high internal energy into an initial blast location, corresponding to a temperature of

$10 \times T_{atm}$ and pressure $10^3 \times p_{atm}$. Boundary conditions are set to be atmospheric, permitting shocks to smoothly flow out of the domain.

All of our simulation are run with second order ENO [47] and third order Runge-Kutta. The examples took between 30 minutes to several hours on our unoptimzied research code with a lot of I/O. For the purposes of comparison we set up a simulation similar to Figure 11 in [45] and Figure 2 in [59]. The explicit version of our code ran in 4 minutes and 24 seconds and the semi-implicit one in 3 minutes 11 seconds, which is comparable to the numbers reported in [45]. Even though the semi-implicit method was faster, if one only cares about the short time simulations with no rolling smoke, etc, explicit methods are just fine.

Trinity Test: Figure 4.8 shows a simulation of the trinity test of 1945, which we model by depositing an initial temperature of $2.62497 \times 10^8\ K$ and pressure of $9.41831 \times 10^{10}\ Pa$ into an initial blast of radius $6.5\ m$. This corresponds to approximately $90\ KJ$ worth of internal energy being introduced to the simulation.

Enclosed Detonation: We show in Figure 4.3 a detonation that goes off in an enclosed blast chamber composed of four massive walls, as suggested by [45]. The detonation drives air out the top of the chamber and through the small openings at the four corners. After the initial shock waves exit the domain, we transition the flow from compressible to incompressible over the time interval $t \in (.15, .16)$, and simulate the resulting smoke plume using a traditional incompressible flow solver, incorporating vorticity confinement.

Shock Hitting Smoke: Transitioning from incompressible flow to compressible flow is a relatively easy task, and can be done by setting ρ, T and p to their atmospheric values at the time of transition. To show this we create a smoke plume and then hit it with a shock wave. The results are shown in Figure 4.10. The smoke plume is driven by buoyancy dynamics, but as the effects of buoyancy are vanishingly small in the time scale of the shock wave we neglect them while the flow is compressible.

Shock Affecting a Light/Heavy Solid: In Figure 4.4 we show a shock interacting with a heavy object, and a shock interacting with a light object. The shock mostly reflects off of the heavy object, generating a strong secondary shock that reflects off the wall. The light object instead absorbs most of the shock wave, rather

than reflecting it. Once the light object collides with the static right wall, it creates a secondary shock due to the sudden change in velocity.

Shock Driving a Stack of Rigid Bodies: Figure 4.2 shows a planar shock wave interacting with a stack of rigid bodies, reflecting off of a wall, and hitting them again before exiting the domain.

Shock Interacting with a Deformable Body: The two-way coupling technique we use is quite general, and works with deformable bodies with arbitrary constitutive models in addition to the rigid bodies shown above. Figure 4.1 shows a shock interacting with a deformable ball which is modeled as a mass-spring system. It has 21528 elements, edge springs with $k = 10^4\ N/m$, and we use altitude springs with $k = 10^4\ N/m$.

Enclosed Detonation with a Fragile Wall: The two-way coupled effects of interacting solids and fluids are demonstrated in Figure 4.5 and Figure 4.6, where one of the walls from Figure 4.3 is replaced with a light wall that fractures as a result of pressure forces. The resulting smoke plumes are compared in Figure 4.7. After the initial shock waves exit the domain, we transition the flow from compressible to incompressible over the time interval $t \in (.15, .16)$, and simulate the resulting smoke plume using a traditional incompressible flow solver, incorporating vorticity confinement.

Cannon Fired at a Bunny: In Figure 4.9 we use an explosive charge to fire a cannonball at an unsuspecting bunny. The cannonball is initially at rest in the chamber of the cannon, creating a seal separating the high-energy blast charge from the outside air. This charge is detonated at $t = 0$ and accelerates the cannonball to a velocity over $1.5km/s$. As the cannonball exits the barrel it is followed by a dynamic, automatically-generated cloud of soot, fuel and fire.

4.7 Conclusions

We present a novel approach to incorporate the ability to handle both the initial states of an explosion (including shock waves) along with the long time behavior

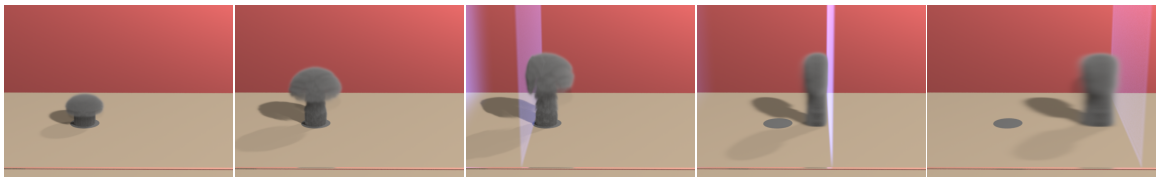


Figure 4.10: After a smoke plume develops, it is hit by a planar shock. Times shown are $.83s$, $1.67s$, $2.0835s$, $2.0836s$, $2.0838s$. The grid resolution is $512 \times 256 \times 256$.

of rolling plumes and other incompressible flow effects. Our method handles compressible flow in a semi-implicit manner, permitting the fast and stable simulation of complex dynamical phenomena, including shock waves and combustion. It supports two-way coupled interactions in a way that permits the integration of complex solid-fluid interactions such as fracture.

Bibliography

[1] M. Arienti, P. Hung, E. Morano, and J.E. Shepherd. A level set approach to Eulerian–Lagrangian coupling. *J. Comput. Phys.*, 185(1):213–251, 2003.

[2] D. Benson. A new two-dimensional flux-limited shock viscosity for impact calculations. *Comput. Meth. Appl. Mech. Engng.*, 93(1):39–95, 1991.

[3] D. Benson. Computational methods in Lagrangian and Eulerian hydrocodes. *Comput. Meth. Appl. Mech. Engng.*, 99:235–394, 1992.

[4] P. Causin, J.-F. Gerbeau, and F. Nobile. Added-mass effect in the design of partitioned algorithms for fluid-structure problems. *Comp. Meth. Appl. Mech. Engng.*, 194(42-44), 2005.

[5] R. Courant, E. Issacson, and M. Rees. On the solution of nonlinear hyperbolic differential equations by finite differences. *Comm. Pure and Applied Math*, 5:243–255, 1952.

[6] T. Dupont and Y. Liu. Back and forth error compensation and correction methods for removing errors induced by uneven gradients of the level set function. *J. Comput. Phys.*, 190/1:311–324, 2003.

[7] J. Falcovitz, G. Alfandary, and G. Hanoch. A two-dimensional conservation laws scheme for compressible flows with moving boundaries. *J. Comput. Phys.*, 138(1):83–102, 1997.

[8] C. Farhat, K. G. van der Zee, and P. Geuzaine. Provably second-order time-accurate loosely-coupled solution algorithms for transient nonlinear computational aeroelasticity. *Comp. Meth. Appl. Mech. Engng.*, 195(17-18):1973 – 2001, 2006.

[9] R. Fedkiw. Coupling an Eulerian fluid calculation to a Lagrangian solid calculation with the ghost fluid method. *J. Comput. Phys.*, 175:200–224, 2002.

[10] R. Fedkiw, T. Aslam, B. Merriman, and S. Osher. A non-oscillatory Eulerian approach to interfaces in multimaterial flows (the ghost fluid method). *J. Comput. Phys.*, 152:457–492, 1999.

[11] R. Fedkiw, X.-D. Liu, and S. Osher. A general technique for eliminating spurious oscillations in conservative schemes for multiphase and multispecies euler equations. *Int. J. Nonlinear Sci. and Numer. Sim.*, 3:99–106, 2002.

[12] R. Fedkiw, A. Marquina, and B. Merriman. An isobaric fix for the overheating problem in multimaterial compressible flows. *J. Comput. Phys.*, 148:545–578, 1999.

[13] R. Fedkiw, B. Merriman, and S. Osher. Efficient characteristic projection in upwind difference schemes for hyperbolic systems (the complementary projection method). *J. Comput. Phys.*, 141:22–36, 1998.

[14] R. Fedkiw, J. Stam, and H. Jensen. Visual simulation of smoke. In *Proc. of ACM SIGGRAPH 2001*, pages 15–22, 2001.

[15] B. Feldman, J. O'Brien, and O. Arikan. Animating suspended particle explosions. *ACM Trans. Graph. (SIGGRAPH Proc.)*, 22(3):708–715, 2003.

[16] H. Forrer and M. Berger. Flow simulations on Cartesian grids involving complex moving geometries. In *Hyperbolic Problems: Theory, Numerics, Applications; Seventh International Conference in Zürich, February 1998*, page 315. Birkhauser, 1999.

[17] N. Foster and D. Metaxas. Controlling fluid animation. In *Comput. Graph. Int.*, pages 178–188, 1997.

[18] J.T. Grétarsson, N. Kwatra, and R. Fedkiw. Numerically stable fluid-structure interactions between compressible flow and solid structures. *J. Comput. Phys.*, 230:3062–3084, April 2011.

[19] J.M. Hong, T. Shinar, and R. Fedkiw. Wrinkled flames and cellular patterns. *ACM Transactions on Graphics (TOG)*, 26(3):47, 2007.

[20] B. Houston, M. Nielsen, C. Batty, O. Nilsson, and K. Museth. Hierarchical RLE level set: A compact and versatile deformable surface representation. *ACM Trans. Graph.*, 25(1):1–24, 2006.

[21] I. Ihm, B. Kang, and D. Cha. Animation of reactive gaseous fluids through chemical kinetics. In *Proc. of the 2004 ACM SIGGRAPH/Eurographics Symp. on Comput. Anim.*, pages 203–212, 2004.

[22] T. Imagire, H. Johan, and T. Nishita. A fast method for simulating destruction and the generated dust and debris. *The Visual Computer*, 25(5):719–727, 2009.

[23] G.-S. Jiang and C.-W. Shu. Efficient implementation of weighted ENO schemes. *J. Comput. Phys.*, 126:202–228, 1996.

[24] S.Y. Kadioglu and M. Sussman. Adaptive solution techniques for simulating underwater explosions and implosions. *J. Comput. Phys.*, 227:2083–2104, 2008.

[25] S.Y. Kadioglu, M. Sussman, S. Osher, J. Wright, and M. Kang. A second order primitive preconditioner for solving all speed multi-phase flows. *J. Comput. Phys.*, 209:477–503, 2005.

[26] B. Kang, Y. Jang, and I. Ihm. Animation of chemically reactive fluids using a hybrid simulation method. In *Proceedings of the 2007 ACM SIGGRAPH/Eurographics symposium on Computer animation*, page 208. Eurographics Association, 2007.

[27] M. Kang, R. Fedkiw, and X.-D. Liu. A boundary condition capturing method for multiphase incompressible flow. *J. Sci. Comput.*, 15:323–360, 2000.

[28] B.-M. Kim, Y. Liu, I. Llamas, and J. Rossignac. Using BFECC for fluid simulation. In *Eurographics Workshop on Natural Phenomena 2005*, 2005.

[29] N. Kwatra, J.T. Grétarsson, and R. Fedkiw. Practical animation of compressible flow for shockwaves and related phenomena. pages 207–215, 2010.

[30] N. Kwatra, J. Su, J.T. Grétarsson, and R. Fedkiw. A method for avoiding the acoustic time step restriction in compressible flow. *J. Comput. Phys.*, 228(11):4146–4161, 2009.

[31] J. Lighthill. *Waves in fluids*. Cambridge Univ Pr, 2001.

[32] T.G. Liu, B.C. Khoo, and W.F. Xie. The modified ghost fluid method as applied to extreme fluid-structure interaction in the presence of cavitation. *Commun. Comput. Phys*, 1(5):898–919, 2006.

[33] F. Losasso, F. Gibou, and R. Fedkiw. Simulating water and smoke with an octree data structure. *ACM Trans. Graph. (SIGGRAPH Proc.)*, 23:457–462, 2004.

[34] D.G. Luenberger. The conjugate residual method for constrained minimization problems. *SIAM J. on Numer. Anal.*, pages 390–398, 1970.

[35] O. Mazarak, C. Martins, and J. Amanatides. Animating exploding objects. In *Proc. of Graph. Interface 1999*, pages 211–218, 1999.

[36] M. Neff and E. Fiume. A visual model for blast waves and fracture. In *Proc. of Graph. Interface 1999*, pages 193–202, 1999.

[37] S. Osher and C.-W. Shu. High order essentially non-oscillatory schemes for Hamilton-Jacobi equations. *SIAM J. Num. Anal.*, 28:902–921, 1991.

[38] C. Peskin. The immersed boundary method. *Acta Numerica*, 11:479–517, 2002.

[39] S. Piperno, C. Farhat, and B. Larrouturou. Partitioned procedures for the transient solution of coupled aroelastic problems part i: Model problem, theory and two-dimensional application. *Comp. Meth. Appl. Mech. Engng.*, 124(1-2):79 – 112, 1995.

[40] N. Rasmussen, D. Nguyen, W. Geiger, and R. Fedkiw. Smoke simulation for large scale phenomena. *ACM Trans. Graph. (SIGGRAPH Proc.)*, 22:703–707, 2003.

[41] A. Robinson-Mosher, C. Schroeder, and R. Fedkiw. A symmetric positive definite formulation for monolithic fluid structure interaction. *J. Comput. Phys.*, 2010. in review.

[42] A. Robinson-Mosher, T. Shinar, J.T. Grétarsson, J. Su, and R. Fedkiw. Two-way coupling of fluids to rigid and deformable solids and shells. *ACM Trans. on Graphics*, 27(3):46:1–46:9, August 2008.

[43] A. Selle, R. Fedkiw, B. Kim, Y. Liu, and J. Rossignac. An unconditionally stable MacCormack method. *J. of Sci. Comp.*, 35(2):350–371, 2008.

[44] A. Selle, N. Rasmussen, and R. Fedkiw. A vortex particle method for smoke, water and explosions. *ACM Trans. Graph. (SIGGRAPH Proc.)*, 24(3):910–914, 2005.

[45] J. Sewall, N. Galoppo, G. Tsankov, and M. Lin. Visual simulation of shockwaves. *Graphical Models*, 2009.

[46] J. Sewall, P. Mecklenburg, S. Mitran, and M. Lin. Fast fluid simulation using residual distribution schemes. In *Eurographics Workshop on Natural Phenomena*, pages 47–54, 2007.

[47] C.-W. Shu and S. Osher. Efficient implementation of essentially non-oscillatory shock capturing schemes. *J. Comput. Phys.*, 77:439–471, 1988.

[48] C.-W. Shu and S. Osher. Efficient implementation of essentially non-oscillatory shock capturing schemes II (two). *J. Comput. Phys.*, 83:32–78, 1989.

[49] P. Smereka. Semi-implicit level set methods for curvature and surface diffusion motion. *J. Sci. Comput.*, 19(1):439–456, 2003.

[50] J. Stam. Stable fluids. In *Proc. of SIGGRAPH 99*, pages 121–128, 1999.

[51] J. Steinhoff and D. Underhill. Modification of the Euler equations for "vorticity confinement": Application to the computation of interacting vortex rings. *Phys. of Fluids*, 6(8):2738–2744, 1994.

[52] J. Su, C. Schroeder, and R. Fedkiw. Energy stability and fracture for frame rate rigid body simulations. In *Proceedings of the 2009 ACM SIGGRAPH/Eurographics Symp. on Comput. Anim.*, pages 155–164, 2009.

[53] K. Taira and T. Colonius. The immersed boundary method: A projection approach. *J. Comput. Phys*, 225(2):2118–2137, 2007.

[54] D. Takeshita, S. Ota, M. Tamura, T. Fujimoto, K. Muraoka, and N. Chiba. Particle-based visual simulation of explosive flames. In *Computer Graphics and Applications, 2003. Proceedings. 11th Pacific Conference on*, pages 482–486, 2003.

[55] P. Woodward and P. Colella. The numerical simulation of two-dimensional fluid flow with strong shocks. *J. Comput. Phys.*, 54:115–173, April 1984.

[56] F. Xiao. Unified formulation for compressible and incompressible flows by using multi-integrated moments I: one-dimensional inviscid compressible flow. *J. Comput. Phys.*, 195:629–5654, 2004.

[57] F. Xiao, R. Akoh, and S. Ii. Unified formulation for compressible and incompressible flows by using multi-integrated moments II: Multi-dimensional version for compressible and incompressible flows. *J. Comput. Phys.*, 213:31–56, 2006.

[58] T. Yabe and P.Y. Wang. Unified numerical procedure for compressible and incompressible fluid. *Journal of the Physical Society of Japan*, 60:2105–2108, July 1991.

[59] G. Yngve, J. O'Brien, and J. Hodgins. Animating explosions. In *Proc. SIGGRAPH 2000*, volume 19, pages 29–36, 2000.

[60] S.Y. Yoon and T. Yabe. The unified simulation for incompressible and compressible flow by the predictor-corrector scheme based on the CIP method. *Comput. Phys. Commun.*, 119:149–158, 1999.

[61] H. Zhang, X. Zhang, S. Ji, Y. Guo, G. Ledezma, N. Elabbasi, and H. deCougny. Recent development of fluid-structure interaction capabilities in the adina system. *Comput. and Struct.*, 81(8-11):1071 – 1085, 2003. K.J Bathe 60th Anniversary Issue.